T.C.
HACETTEPE ÜNİVERSİTESİ
SAĞLIK BİLİMLERİ ENSTİTÜSÜ

MİKRODİZİLİM GEN İFADE ÇALIŞMALARINDA GENELLEŞTİRME YÖNTEMLERİNİN REGRESYON MODELLERİ ÜZERİNE ETKİSİ

Selen YILMAZ IŞIKHAN

Biyoistatistik Programı
DOKTORA TEZİ

ANKARA
2014

T.C.
HACETTEPE ÜNİVERSİTESİ
SAĞLIK BİLİMLERİ ENSTİTÜSÜ

MİKRODİZİLİM GEN İFADE ÇALIŞMALARINDA GENELLEŞTİRME YÖNTEMLERİNİN REGRESYON MODELLERİ ÜZERİNE ETKİSİ

Selen YILMAZ IŞIKHAN

Biyoistatistik Programı
DOKTORA TEZİ

TEZ DANIŞMANI
Prof. Dr. C. Reha ALPAR

ANKARA
2014

Anabilim Dalı	**: Biyoistatistik**
Program	**: Biyoistatistik**
Tez Başlığı	**: Mikrodizilim Gen İfade Çalışmalarında Genelleştirme Yöntemlerinin Regresyon Modelleri Üzerine Etkisi**
Öğrenci Adı-Soyadı	**: Selen YILMAZ IŞIKHAN**
Savunma Sınavı Tarihi	**: 20.06.2014**

Bu çalışma jürimiz tarafından **Doktora** tezi olarak kabul edilmiştir.

Jüri Başkanı ve Tez danışmanı:	**Prof. Dr. Celal Reha ALPAR** **Hacettepe Üniversitesi**
Üye:	**Prof. Dr. Osman SARAÇBAŞI** **Hacettepe Üniversitesi**
Üye:	**Prof. Dr. Atilla Halil ELHAN** **Ankara Üniversitesi**
Üye:	**Doç. Dr. Erdem KARABULUT** **Hacettepe Üniversitesi**
Üye:	**Doç . Dr. Pınar ÖZDEMİR** **Hacettepe Üniversitesi**

ONAY

Bu tez Hacettepe Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca yukarıdaki jüri tarafından uygun görülmüş ve Sağlık Bilimleri Enstitüsü Yönetim Kurulu kararıyla kabul edilmiştir.

Prof.Dr. Ersin FADILLIOĞLU

Müdür y.

TEŞEKKÜR

Tez çalışmasının gerçekleştirilmesinde bana her aşamada bilgi ve deneyimleriyle büyük katkı sağlayan ve eğitim sürecimde hiçbir konuda desteğini esirgemeyen danışman hocam Sayın Prof. Dr. Reha ALPAR'a çok teşekkür ederim.

Tez izleme komitemde yer alarak eleştiri, görüş ve önerileriyle teze büyük katkı sağlayan hocalarım Sayın Prof. Dr. Osman SARAÇBAŞI, Prof. Dr. Atilla Halil ELHAN ve Doç. Dr. Pınar ÖZDEMİR'e çok teşekkür ederim.

R yazılımında gereksinim duyulan kodların yazımında ve benzetim çalışmasında yoğun desteği olan hocam Sayın Doç. Dr. Erdem KARABULUT'a ve çalışma konusunun belirlenmesinde ve gerçek veri setlerinin elde edilmesinde büyük katkısı olan Dr. Erdal COŞGUN'a çok teşekkür ederim.

Tezin son biçimini almasında görüş ve önerileriyle katkıda bulunan tez jürisindeki değerli hocalarıma teşekkür ederim.

Her zaman yanımda olan anneme, aileme ve eşim Murat IŞIKHAN'a çok teşekkür ederim.

ÖZET

Selen, Y.I. Mikrodizilim Gen İfade Çalışmalarında Genelleştirme Yöntemlerinin Regresyon Modelleri Üzerine Etkisi. Hacettepe Üniversitesi Sağlık Bilimleri Enstitüsü Biyoistatistik Programı Doktora Tezi, Ankara, 2014. Genetik araştırmalarda az sayıda hastaya ait binlerce gen verisi bulunması, klasik istatistiksel yöntemlerin (doğrusal regresyon vb.) kullanımında sorunlar ortaya çıkarmaktadır. Ancak yakın zamanda mikrodizilim gen ifade çalışmalarında çok fazla sayıdaki genin aynı anda analizi destek vektör makinaları (DVM), karar ağaçları, *boosted tree* gibi veri madenciliği yöntemlerinin de kullanılmasıyla mümkün hale gelmiştir. Bu çalışmada veri yapısı hakkında varsayım gerektirmeyen ve çok sayıda kestiriciyi modelleyebilen bu yöntemlerin gen verisi ile kestirim performansları incelenmiştir. Gen ifade verilerinde gerçekleştirilen analizler için en temel adımlardan biri analiz modellerinin genelleştirilmesidir. Bağımsız bir test verisi bulunmuyor ise, kestirim doğruluğunu belirlemek için, orijinal verinin yeniden örneklenmesi gibi yaklaşımlar kullanılmalıdır. Çalışmanın diğer amacı, genelleştirme yöntemlerinden bootstrap, çapraz geçerlik ve birini dışarıda bırakma yöntemlerinin DVR ve regresyon ağacı model performansları üzerine etkisini karşılaştırmaktır. Regresyon modellerinin genelleştirme yöntemleri ile performans karşılaştırmasında iki farklı Monte Carlo benzetim çalışması gerçekleştirilmiştir. Genel olarak bootstrap çapraz geçerlikten daha iyi performans vermiştir. Verilen bir model kurma tekniğinin kestirim performansını geliştirmede kullanılan araçlar ise model birleştirme (ensemble) yöntemleridir. Çalışmada ayrıca bagging ve boosting yöntemlerinin incelenen regresyon yöntemleri üzerinde performansı karşılaştırılmıştır. Bagging, gözlem sayısı n≥25 olan veri setleri için RA'da gelişme sağlamıştır. Gerçek gen verileri uygulaması benzetim çalışması ile uyumlu sonuçlar göstermiştir.

Anahtar Kelimeler: Destek Vektör Regresyon, Regresyon Ağacı, Genelleştirme Yöntemleri, Mikrodizilim Gen Verisi, Kestirim Performansı.

ABSTRACT

Selen, Y.I. The Effects of Generalization Methods On Regression Models in Microarray Gene Expression Studies. Hacettepe University Institute of Health Science, Ph.D. Thesis in Biostatistics, Ankara, 2014. The presence of thousands of gene data belonging to a few number of patients in genetic researches leads to problems in the use of classical statistical methods (linear regression analysis etc.). However, analysis of large number of genes in microarray gene expression studies simultaneously has become possible recently by using data mining methods such as support vector machine, decision tree and boosted tree. In this study, prediction performances of these methods which don't require assumptions about the data structure and can model a large number of predictors were examined on gene data. One of the basic steps for analyses which were performed on gene expression data is generalization of models of analysis. If an independent test data is not available, the approaches such as resampling the original data should be used to estimate the accuracy of prediction. Another purpose of the study is to compare the effect of bootstrap and cross validation generalization methods on model performances of support vector regression and regression trees. Two different Monte Carlo simulations were carried out for performance comparison of regression models with generalization methods. Overall, bootstrap has given more optimistic performance than cross validation. The tools that are used in the development of prediction performance of a given model building technique are model aggregating (ensemble) methods. In this study, the performances of bagging and boosting methods were also compared on the examined regression methods. Bagging has provided the improvement of regression tree for datasets having at least a number of 25 observations, "i.e." $n \geq 25$. The application of real gene data has shown consistent results with the simulation study.

Key words: support vector regression, regression trees, generalization methods, microarray gene data, prediction performance.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

B	:Bootstrap tekrar sayısı
BDBCG	:Birini dışarıda bırakma çapraz geçerlik
BRA	:Boosting regresyon ağacı
CG	:Çapraz geçerlik
ÇDND	:Çok değişkenli normal dağılım
DVM	:Destek vektör makinası
DVR	:Destek vektör regresyon
DVR-Bag	:Destek vektör regresyon bagging performansı
DVR-BDB	:Destek vektör regresyon-birini dışarıda bırakma çapraz geçerlik
DVR-Boot	:Destek vektör regresyon-bootstrap performansı
DVR-CG	:Destek vektör regresyon-çapraz geçerlik
DVRLIN	:Destek vektör makinası doğrusal fonksiyon çözümü
DVRPOL	:Destek vektör makinası polinomiyal fonksiyon çözümü
DVS	:Destek vektör sayısı
EKK	:En küçük kareler
gbm	:Gradyan boosting makinası
HKO	:Hata kareler ortalaması
K	:Çapraz geçerlik parça sayısı
KHKO	:Karekök hata kareler ortalaması
lr	:Öğrenme oranı (küçültme parametresi)
OMS	:Ortalama mutlak sapma
RA	:Regresyon ağacı
RA-Bag	:Regresyon ağacı bagging performansı
RA-BDB	:Regresyon ağacı birini dışarıda bırakma çapraz geçerlik
RA-Boot	:Regresyon ağacı- bootstrap performansı
RA-CG	:Regresyon ağacı-çapraz geçerlik
SRA	:Sınıflandırma ve regresyon ağacı
SDS	:Son düğüm sayısı
tc	:Etkileşim derecesi

ŞEKİLLER

TABLOLAR

1. GİRİŞ

Veri madenciliği yöntemlerinin yaygın şekilde kullanıldığı klinik çalışmalar mikrodizilim gen ifade çalışmalarıdır. Genetik araştırmalarda az sayıda hastaya ait binlerce gen verisi bulunması, klasik istatistiksel yöntemlerin (Lojistik Regresyon,varyans analizi, doğrusal regresyon analizi vb.) kullanımında sorunlar ortaya çıkarmaktadır. Özellikle mikrodizilim deneyleri için regresyon yaklaşımlarını geliştirme ihtiyacı temel olarak dikkate alınan ya da potansiyel ilgi çekici açıklayıcı değişken sayısının (p) çoğu zaman örneklem genişliğini (n) aşması (yani büyük p, küçük n) probleminden türemiştir (1). Mikrodizilim verileri ile regresyon analizini uygulama zorlukları birçok yeni yaklaşım doğurmuştur. Yakın zamanda bu tip veriler üzerinde Destek vektör makinası (DVM), karar ağaçları, boosted tree, random forest gibi birçok veri madenciliği yöntemi denenmiş ve iyi sonuçlar elde edilmiştir.

Başka bir zorluk, elde edilen çeşitli sınıflama, ayırma ya da regresyon yöntemlerine dayalı olarak en uygun modelin seçimidir. Gen ifadesi verileri analizinin ikinci adımı seçilen modelin hata oranını ya da genelleştirilebilirliğini değerlendirmektir. Bağımsız bir değerlendirme (test) verisi bulunmuyor ise, kestirim doğruluğunu tahmin etmek için yaygın bir yaklaşım; orijinal verinin yeniden örneklenmesi ya da parçalara bölünmesi gibi yaklaşımlara dayalıdır. Bu amaçla kullanılan en popüler yöntemler; b*ootstrap ve çapraz geçerlik* yöntemleridir.

Genelleştirme yöntemleri ya da model birleştirmeye dayalı (ensemble) yöntemler, dağılımların normal olması veya örneklem genişliklerinin büyük olması gibi varsayımları gerektirmez. Model birleştirme yöntemleri, aynı eğitim seti üzerinden elde edilen kestirimlere göre kestiricinin ya da sınıflayıcı yöntemin doğruluğunu arttırırlar (2).

Bu çalışmanın ilk aşamasında yeniden örneklemeye dayalı genelleştirme yöntemlerinden bootstrap, çapraz geçerlik ve birini dışarıda bırakma (leave-one-out) yöntemleri incelenerek yöntemlerin kendi özel parametrelerine göre değişimler karşılaştırılmıştır. İkinci bir aşama olarak model birleştirme (ensemble) yöntemlerinden *bagging ve boosting* yöntemlerinin dikkate alınan temel regresyon yöntemleri üzerindeki etkisi incelenmiştir.

Genelleştirme ve model birleştirme yöntemlerinin her biri, veriyi öğrenme ve test seti olarak ikiye bölmeye dayanır. *Bootstrap* yönteminde amaç veri setinden yerine koyarak tekrarlı örneklemler çekerek örneklem değişimini değerlendirmektir. Bu yöntem ile çok küçük veri setlerinden, klasik yönteme göre daha fazla bilgi sağlanır.

İlk olarak Larson'ın 1931 yılındaki çalışmasında kullandığı *K–parçalı çapraz geçerlik* yönteminde veri seti rastgele eşit gözlemli olarak "K" gruba ayrılır (3). Literatürde genel olarak "K" değerinin 10 olarak seçildiği görülmektedir. Her uygulamada bu gruplardan biri dışarıda bırakılarak geriye kalan (training) kısım ile model oluşturulur. Oluşturulan modelin sınıflama/regresyon performansı dışarıda bırakılan veriler üzerinde test edilir. Sırasıyla bu işlem dönüşümlü olarak "K" grup için de tekrarlanır (3).

Son olarak elde edilen "K" tane değerlendirme ölçütünün (R^2, ROC eğrisi alanı vb.) ortalaması alınarak modelin genel performansı değerlendirilir. Bu yöntem, regresyonda kaç tane açıklayıcı değişken bulunması gerektiğine karar vermede de kullanılır.

Çapraz geçerlik yöntemi olmadan modele kestirici eklemek her zaman artık kareler toplamını azaltır (ya da sabit tutar). Aksine; eğer değerli kestiriciler eklenirse çapraz geçerliğin hata kareler toplamı azalır; fakat anlamsız değişkenler eklenirse hata kareler toplamı artar.

Birini dışarıda bırakma çapraz geçerlik yöntemi (BDBCG) ilk olarak Lachenbruch ve Mickey tarafından 1968'de ele alınan yeniden örneklemeye dayalı geniş ölçüde kullanılan bir genelleştirme yöntemidir (3).

Çapraz geçerlik algoritmasında K=N olması yani her seferinde bir örneğin çıkarılıp diğerleri üzerinde model kurulması ve bu işlemin gözlem sayısı (N) kadar tekrarlanması ile elde edilir. BDBCG yöntemi, bu nedenle K parçalı çapraz geçerliğe göre daha yavaş çalışan bir yöntemdir. K-Parçalı çapraz geçerlik yöntemi BDBCG'e göre daha iyi bir seçenektir. Çapraz geçerlikte bir veri noktasından daha büyük bir oran dışarıda bırakılır.

Model birleştirme (ensemble) yöntemi, verilen bir istatistiksel öğrenmenin ya da model kurma yönteminin kestirim performansını geliştirmeyi amaçlar. Birleştirme yöntemlerinin temel ilkesi, modeli tek bir biçimde kullanmak yerine bazı model oluşturma yöntemlerinin doğrusal bir kombinasyonunu oluşturmaktır. *Bagging ve boosting* yöntemleri, sınıflandırıcının ya da kestirimin doğruluğunu arttırmak için kullanılan birleştirme yöntemlerinin iki örneğidir.

Bagging (bootstrap aggregating) yöntemi Breiman tarafından 1994'de rastgele üretilmiş eğitim setlerinin sınıflamalarını birleştirerek sınıflandırmayı geliştirmek amacıyla önerilmiştir (4). Bagging'in kestirim doğruluğunda gelişme yaratıp yaratmaması, herbir kestiriciyi üreten yöntemin kararlılığı (stable) ile ilgilidir. Öğrenme algoritması kararsız (unstable) ise bagging her zaman performansı arttırır. Öğrenme algoritmasının kararsız olması, eğitim setindeki küçük değişiklikler için öğrenme kestiricisinde büyük değişimlere neden olur. Kararsız kestirim yöntemine bazı örnekler: karar ağaçları, yüzeysel karar ağaçları (decision stump), regresyon ağacı (RA) ve yapay sinir ağlarıdır. Bunun yanı sıra, en küçük kareler (EKK), en yakın komşuluk regresyonu, ridge regresyon ve destek vektör regresyonu gibi daha kararlı yöntemlerin bagging'den yarar sağlaması beklenmez (4-5).

Boosting yöntemi ilk olarak Freund tarafından 1990'da ve Schapire tarafından 1995'de tanıtılmıştır (6). Bagging'de olduğu gibi boosting yönteminde de birleştirmeyi (ensemble) oluşturan kestiriciler, verilerin yeniden örneklenmesi ile elde edilmekte ve daha sonra çoğunluk oyları ile birleştirilmektedir. *Bagging*'de eğitim setinin yeniden örneklenmesi daha önceki kestiricilerin performansına bağlı değildir.

Buna rağmen *boosting,* şu anki kestirim performansının zayıf olduğu örnekleri daha doğru bir şekilde tahmin edecek yeni kestiriciler üretmeye yönelir. Boosting'in ilk algoritması Adaboost.R yanıt değişkeninin sadece iki değer aldığı sınıflama problemleri (-1/1) (lojistik regresyon vb.) için geliştirilmiştir. Friedman ve diğerleri, 2000 yılında boosted lojistik regresyon algoritmasını geliştirmiş, daha sonra Hastie ve diğerleri 2001'de boosting algoritmasını tüm hata dağılımları için uyarlayabilmiştir. Boosted regresyon ağaçları ilk olarak hem istatistiksel hem de makina öğrenmesindeki görüş ve yöntemlere dikkat çeken Schapire tarafından 2003'de çalışılmıştır (6). Boosted regresyon ağaçları, iki algoritmanın gücünü birleştirir: *regresyon ağaçları* (ardışık ikili bölünmeler yardımıyla kestiriciler için sonuç veren modeller) ve *boosting* (geliştirilmiş kestirim performansı vermek için birçok basit modelin birleştirilmesine dayalı yöntem).

Bu çalışmada temel kestirim yöntemi olarak destek vektör regresyon modelleri ve regresyon ağacı yöntemleri incelenmiştir. Destek vektör makinesi (*DVM*), ilk olarak Vapnik ve diğerleri tarafından 1992'de önerilen danışmanlı öğrenme algoritmalarının bir sınıfıdır (7). *DVM*, orijinal olarak sınıflandırma için geliştirilmesine rağmen Vapnik, yöntemi elde edilecek destek vektörlerin seyrek bir dizisini sağlayan uygun bir en iyileme (optimizasyon) fonksiyonu seçerek regresyon problemlerini çözmek için etkin hale getirmiştir.

DVM'nin regresyon için uyarlaması 1997'de Vapnik, Goldwich ve Smola tarafından önerilmiştir (7). Destek vektör regresyonun (DVR) genelleştirme hatası diğer makine öğrenme yöntemlerinden farklı olarak problemin orijinal girdi boyutuna (ölçeğine) doğrudan bağlı değildir. DVR, klasik regresyon yöntemlerinden farklı olarak gözlenen eğitim hatasını minimize etmek yerine genelleştirme hatasını en aza indirmeye çalışır. Bu ilgili genelleştirme hatası, eğitim hatası ve hipotez uzayının karmaşıklığını kontrol eden düzenleme teriminin bir kombinasyonudur.

DVR yöntemi, yüksek boyutlu uzayda yararlı bir yöntemdir, çünkü DVR optimizasyonu girdi uzayının boyutuna bağlı değildir. Bu yöntem klasik regresyon yöntemi ile karşılaştırıldığında belirli avantajlara sahiptir. DVR, çeşitli türlerde kayıp fonksiyonlarının kullanımına izin verdiği için daha esnek olduğu kanıtlanmıştır, ayrıca doğrusal olmama, çekirdek fonksiyonlar kullanılarak kolayca modele tanıtılır (7).

Karar ağaçları, sınıflandırma ve regresyon için güçlü ve popüler araçlardır. Yöntemin amacı birçok girdi değişkenine dayalı olarak yanıt değişkeninin değerini kestiren bir model oluşturmaktır. Karar ağaçları, karar düğümleri, dallar ve yapraklardan meydana gelmektedir. Kısaca karar ağacı işlemi kök düğümde başlayarak, yukarıdan aşağıya doğru ardışık düğümler takip edilerek yaprağa ulaşıncaya kadar devam eden bir süreçtir.

Sınıflandırma ağacı analizi kestirim değişkeni, verinin ait olduğu sınıfı belirten niteliksel veri iken kullanılır. Regresyon ağacı analizi ise, kestirim değişkeni gerçel bir sayı olarak dikkate alınabildiğinde kullanılır.

Bu çalışmada amaç, dikkate alınan temel regresyon yöntemlerinin yüksek boyutlu mikrodizilim veri setleri üzerindeki kestirim performanslarını karşılaştırmaktır. Uygulama sonucunda, farklı genelleştirme yöntemlerinin hangi regresyon yöntemlerinde daha iyi performans gösterdiği, model birleştirme yöntemlerinin regresyon modellerine etkileri ve ilgili regresyon yöntemlerinin özel parametrelerinin ayarlanması gibi konulara açıklık kazandırmak hedeflenmiştir.

Model performansları bootstrap tekrar sayısı (B), çapraz geçerlik için parça sayısı (K), örneklem genişliği (n), genler arası korelasyon ve hatanın standart sapması gibi faktörler açısından değerlendirilmiştir. Daha önceki çalışmaların benzetim düzeninden farklı olarak hem modele bağlı hem de genlerin gerçek dağılımına uyacak şekilde ayarlanan iki farklı düzende modellerin hata kareler ortalamasının karekökü, ortalama mutlak sapma ve belirtme katsayısı açısından karşılaştırılması için 1000 tekrarlı Monte Carlo benzetim tekniği kullanılmıştır.

2. GENEL BİLGİLER

2.1. Mikrodizilim Teknolojisi

Mikrodizilim teknolojisi (gen çipi, DNA çipi ya da biyoçip olarak da bilinir) biyolojik bir örnekteki binlerce genin ifade (ekspresyon) düzeyini aynı anda izleyebilen yeni bir (1995) yöntemdir. DNA mikrodizilimlerinin son birkaç yıl içinde kullanımı artmıştır ve genomik bir ölçekteki biyolojik bilgimizi ilerletmemize yardımcı olacak bir potansiyele sahiptir.

DNA mikrodizilimleri aktif proteinlere çevrilebilen ya da çevrilemeyen RNA'ların saptanmasında kullanılabilir. Bu tip analizler ifade analizi ya da ifade profili belirleme şeklinde adlandırılır. Daha önce tanımlanmamış birçok genin ifade profili incelenerek ve diğer bilinen gen profilleri ile karşılaştırarak o genlerin olası fonksiyonları hakkında ipuçları elde edilmektedir (1,8). Bu biçimde olan gen ifade dizisi verileri için çok değişkenli istatistiksel yöntemler geniş çapta kullanılır.

2.2. Gen İfade Verileri için Regresyon Yöntemleri

Tüm genom ölçeği üzerinde gen ifade profilleri ve fenotip arasındaki ilişkiyi incelemek amacıyla DNA mikrodizilimlerin potansiyel kullanımı üzerine çok sayıda çalışma yapılmıştır. İlk deneme, sınıflama ve ayırma yöntemlerinin çalışıldığı farklı kanser sınıfları gibi (Dudoit ve diğerleri, 2002) kategorik fenotipler üzerine kurulmuştur; ancak daha yakın zamanda regresyon çerçevesinin uygun olduğu sürekli (Li ve Hong, 2001) ya da sağkalım (Hastie ve diğerleri, 2001) fenotiplerinin araştırması da yapılmıştır (1).

Mikrodizilim düzeni için regresyon yaklaşımlarını geliştirme ihtiyacı temel olarak elde bulunan ya da potansiyel olarak ilgi çekici kestiricilerin (p) çoğu zaman örneklem genişliğini (n) aştığı "büyük p, küçük n" probleminden türemiştir (1).

Diğer bir neden ise, yolak (pathway) ve gen ağı ilişkilerinden dolayı örneklemler üzerindeki çeşitli genlerin ifade düzeyleri arasında büyük olasılıkla güçlü ve karmaşık ilişki olmasıdır (1).

Mikrodizilim verileri ile regresyon analizini uygulama zorlukları birçok yeni yaklaşım doğurmuştur. Mikrodizilim verileri gibi yüksek boyutlu verilerde, çok bilinen "Temel bileşenler analizi (principal component analizi)" yerine veri madenciliği yöntemlerinden bağımsız bileşenler analizi (independent component Analizi)"nin daha iyi bir faktörizasyon sağladığı gözlemlenmiştir (9).

Gen ifadesi verilerinin regresyon analizi için "*Lasso* regresyon" 1996'da Tibshirani tarafından, *"Least Angle* regression" 2002'de Efron ve diğerleri tarafından ve destek vektör makinaları 1997'de Vapnik tarafından, 2000'de ise Brown ve diğerleri tarafından regresyon yöntemleri için alternatif araçlar olarak incelenmiştir (1).

Lojistik regresyon, varyans analizi, doğrusal regresyon analizi gibi klasik istatistiksel yöntemler az sayıdaki bireye ait binlerce geni açıklamakta sorun yaşamaktadır. Yakın zamanda bu tip veriler üzerinde destek vektör makinaları (DVM), karar ağaçları, boosted tree ve random forest gibi birçok veri madenciliği yöntemi denenmiş ve iyi sonuçlar elde edilmiştir. Literatürde sağkalım analizi ve lojistik regresyonun da gen verilerinde sık kullanımına rastlanılmaktadır.

Klein, Altman, Eriksson gibi araştırmacıların 2009 yılında komite üyesi oldukları "*Estimation of the Warfarin Dose with Clinical and Pharmacogenetic Data***"** isimli uluslararası varfarin farmakogenetik konsorsiyumda bazı çevresel faktörlerin yanı sıra gen ifade değerleri gibi genetik faktörler de bağımsız değişken olarak kullanılarak *teröpatik varfarin dozunun tahmini* elde edilmiştir (10).

Tez çalışmasında gen ifade verilerinden doz kestirimi elde etmede Klein ve diğerlerinin bu çalışması referans alınmıştır (10).

Veri madenciliği modelleri, tanımlayıcı ve kestirici modeller olarak iki kategoriye ayrılmaktadır. Tanımlayıcı modeller, veri tabanındaki verinin genel niteliklerini tanımlamaktadır. Kestirici modeller ise veriyi kullanarak çıkarsamalarda bulunmakta ve kestirim yapmaktadır. Sınıflama ve regresyon, kestirici yöntemler içerisine; tanımlama, kümeleme ve birliktelik ise tanımlayıcı modeller içerisinde yer almaktadır.

Sınıflandırma, hangi sınıfta olduğu bilinmeyen bir veri örneğinin sonlu sayıda sınıftan hangisine ait olduğunun kestirilmesi işlemidir. Sınıflandırmada modelin öğrenilmesi genellikle danışmanlı olarak yapılır. Esas olarak sınıflandırmanın iki amacı bulunmaktadır: veriyi sadeleştirmek ve kestirim yapmak.

Kümeleme sınıflandırmanın tersine, önceden belirlenmiş bir sınıf etiketi olmaksızın veri grubundaki sınıfları belirleme amacı güder. Kümeleme, eldeki verileri sınıflara ya da kümelere ayırma işlemidir. Genellikle her bir eğitim örneğinin sınıf etiketinin bilinmediği bu öğrenme kalıbı danışmansız öğrenme olarak da bilinir.

Regresyon yöntemleri ise, değişkenler sayısal iken bir ya da daha fazla sayıda açıklayıcı değişkenden yanıt (response) değişkeninin değerini kestirmek için kullanılır. Sınıflama ile regresyon arasında önemli farklılıklar bulunmaktadır. Sınıflamada hedef değişken kategorik veri tipinde iken regresyonda sürekli sayısal veri tipindedir.

Klasik istatistiksel yöntemlere göre oluşturulan regresyon ve sınıflandırma uygulamaları, olasılık dağılım modelleri veya olasılık dağılım fonksiyonları olarak bilinen kesin varsayımlara göre çalışırlar. Bu varsayımların pratikte sağlanması ise zordur. Bu yüzden, olasılık dağılımının bilinmediği durumlarda dağılımdan bağımsız yöntemlerin kullanılmasına ihtiyaç duyulur. DVM yöntemi, dağılımdan bağımsız olarak çalışabilir ve istatistiksel öğrenme teorisine göre eğiticili veya yarı eğiticili olarak sınıflandırma ve regresyon işlemlerini gerçekleştirebilir. Güncel uygulamaların çoğundaki yegâne bilgi sonlu veri kümesidir ki, o bilgi de çok boyutlu ve sınırlı sayıdadır.

Bu yüzden öğrenme makinasının bu koşullarda dahi işlem yapabilmesi istenir. Yapay sinir ağları ve bulanık sistemler gibi geleneksel yöntemler bu yeteneğe sahip değillerdir ve bu tip durumlarda güvenilir olmayan sonuçlar üretirler. Ayrıca, yüksek boyutlu uzay ise boyut sıkıntısı (curse of dimensionality) problemine neden olur (1, 11, 12-13).

Özellikle biyolojik uygulamalardaki yüksek boyutlu ve az sayıda gözlem içeren veri kümeleri için başarılı sonuçlar sağlayan birçok makina öğrenmesi algoritması yaygın şekilde kullanılmaktadır. Bunlar DVM ve regresyon ağaçları gibi karar ağacı algoritmalarını içeren yöntemlerdir. Tez çalışmasında bu iki yöntem incelenecek olmakla birlikte bu bölümde ilk olarak aşırı uyum sorununa değinilecektir.

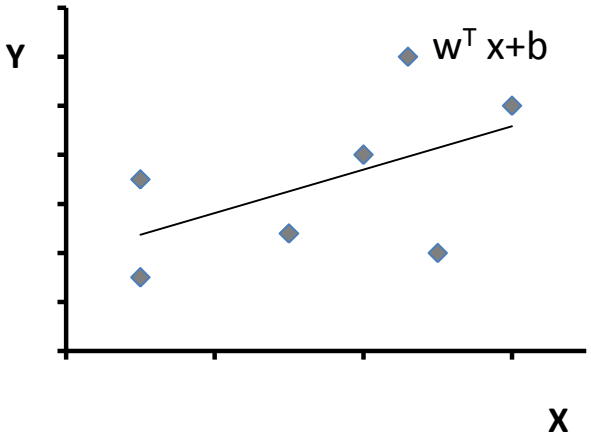
2.2.1. Doğrusal Regresyon

Şekil 2.1'de x_i'ler girdi değişkenleri y_i'ler ise x_i ile bağlantılı yanıt değişkenini belirtmek üzere $(x_1, y_1), \ldots, (x_n, y_n)$ eğitim verileri verildiğinde klasik doğrusal regresyon, (w,b)'nin en uygun çözüm olduğu doğrusal bir $w^T x+b$ fonksiyonu bulur.

$$_{w,b} min \sum_{i=1}^{n} (y_i - (w^T x_i + b))^2 \qquad (2.1)$$

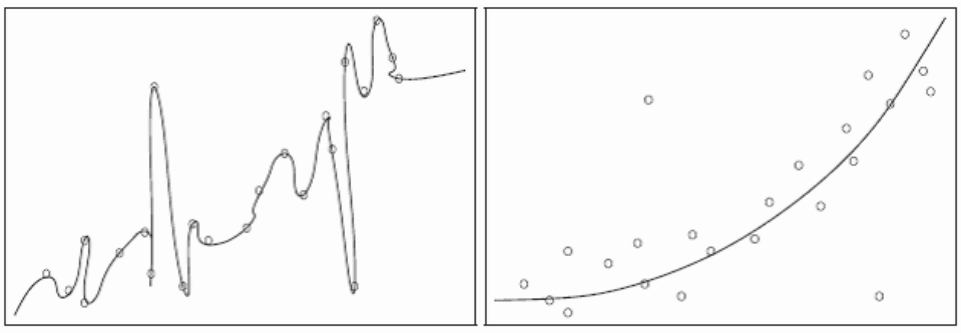
Diğer deyişle $w^T x + b$, eğitim verilerini hata kareler toplamını en küçük yaparak tahmin eder.

Değişken sayısı p, genellikle gözlem sayısı n'den daha küçüktür. Ancak bunun tersi gerçekleştiğinde yani gözlem sayısı değişken sayısına göre yetersiz kaldığında doğru tüm noktalardan geçer ve (2.1) Formülü sıfır olur, yani $w^T x + b$ en küçük hale gelir. Bu gibi durumlarda (overfitting) aşırı uyum sorunu ortaya çıkar.



Şekil 2.1. Doğrusal regresyon gösterimi.

Veri, doğrusal olmayan dağılıma sahip ise doğrusal bir fonksiyon yeterince iyi değildir. Sonraki bölümde açıklanacak olan DVR'unda sınıflandırmaya benzer şekilde bu veri, φ(x) fonksiyonu yardımıyla daha yüksek boyutlu bir uzaya eşleştirilir. Bu durumda n, φ(x)'in boyutundan küçük ise tekrar aşırı uyum ortaya çıkar. Bunun bir örneği Şekil 2.2'de verilmiştir (14).

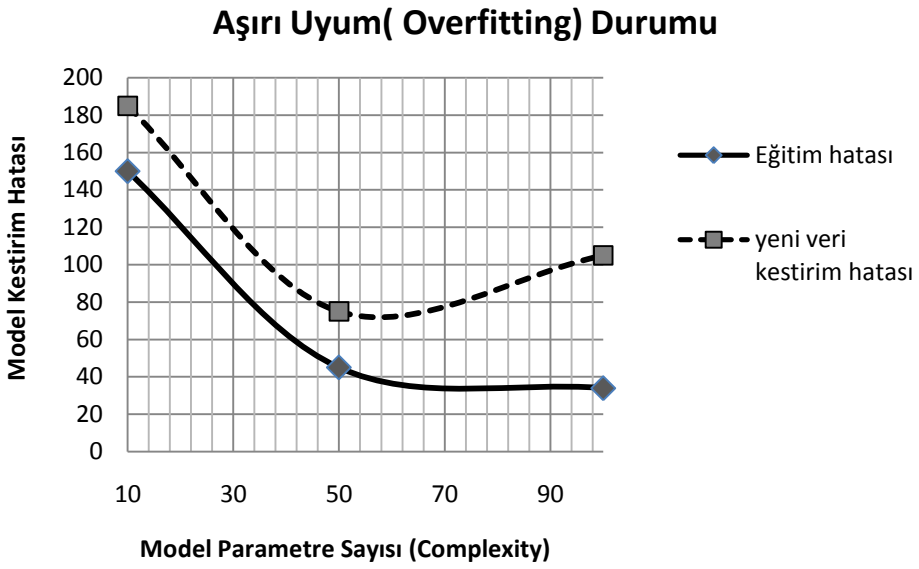


Şekil 2.2. Doğrusal olmayan regresyon

a) Aşırı uyum (overfitting) b) Daha iyi bir kestirim

Kestirim modelleri oluşturulurken, temel amaç yeni veriler için yanıt değerini en doğru şekilde kestiren bir model oluşturmaktır. Kullanılan model hatası ölçüsü bu amacı gerçekleştiren bir ölçü olmalıdır. Ancak uygulamada bir çok araştırmacı yeni veriler için model hatasının bir ölçüsünü kullanmak yerine modeli eğitmek için kullanılan aynı veri seti üzerinde model hatası ölçüsünü elde eder. Bu yanlış hata ölçüsünün kullanımı ise kalitesiz ve yanlış model seçimine neden olabilir.

Doğal olarak herhangi bir model, eğitilmiş olduğu veri için optimize edilmiştir. Modelin yeni veriler üzerinde sergilediği beklenen hata, modelin eğitim veri seti üzerinde gösterdiği hatadan daima daha yüksek olmaktadır. Aşırı uyumu belirlemek oldukça kolaydır, bunun en temel yolu modelin karmaşıklığı arttıkça (açıklayıcı değişken sayısı) eğitim hatasının değişmemesi ya da giderek azalmasıdır. Dolayısıyla aşırı uyumda, model karmaşıklığı arttıkça iyimserlik (optimizm) de artmaktadır (15-17). Oysa farklı bir veri seti üzerinden elde edilen kestirim hatası model karmaşıklığı arttıkça yükselmektedir (Şekil 2.3).



Şekil 2.3. Aşırı uyum durumunda eğitim ve test performansları

Aşırı uyum durumunu gösteren yukarıdaki şekil p=10, 50 ve 100 değişken dikkate alınarak gen verileri için regresyon ağacı yönteminden hesaplanan HKO değerlerini belirtmektedir. Eğitim seti üzerinden elde edilen HKO değerleri (düz çizgi) değişken sayısı arttıkça azalma göstermektedir.

2.2.2. Destek Vektör Regresyonu

DVM güçlü istatistiksel teoriler üzerine inşa edilmiş eğiticili bir makina öğrenmesi yöntemidir. İlk kez 1995 yılında Vapnik tarafından sınıflandırma tipi problem çözümleri için önerilmiştir.

Geleneksel makina öğrenmesi yöntemlerinde çok sayıda eğitim verisine sahip olma isteği, düşük yakınsama oranı, yerel minimuma takılma ve aşırı uyum/yetersiz uyum (overfitting/underfitting) problemleriyle karşılaşılmaktadır. Bu problemlerin çoğu geleneksel yöntemlerin deneysel risk minimizasyonuna bağlı olarak çalışmasından kaynaklanır. DVM, yapısal risk minimizasyonu (YRM) prensibine bağlı olarak öğrenir ve bu problemlerin çoğunu böylece aşmıştır. DVM, YRM prensibinin ve VC (Vapnik-Chervonenkis) teorisinin uygulandığı bir yakınsama yöntemidir (18-19).

Bu özelliğiyle diğer makina öğrenme yöntemlerinin çoğundan farklıdır. DVM, beklenen riskin minimuma ulaşması için hem deneysel riski hem de VC boyutunu minimum tutmaya çalışır. DVM, yapısal risk minimizasyonu temelinde çalışarak bu problemlerin üstesinden gelmiştir.

DVM'nin sınıflandırma uygulamalarında kullanılan çeşidi DVS (destek vektör sınıflandırma), regresyon uygulamalarında kullanılan çeşidi ise DVR (destek vektör regresyon) olarak bilinir. DVR, ilerleyen bölümde ayrıntılı olarak incelenmiştir. DVM, yüksek boyutlu fakat az sayıda veri içeren uygulamalarda da başarılıdır (7, 12).

Bu özelliklerinden dolayı DVM'leri,

- Görüntü (pattern) tanıma (sınıflama) problemleri,
- Mikrodizilim gen ifadesi sınıflandırma
- Protein katlanmasını belirleme,
- Protein yapısal sınıf kestirimi,

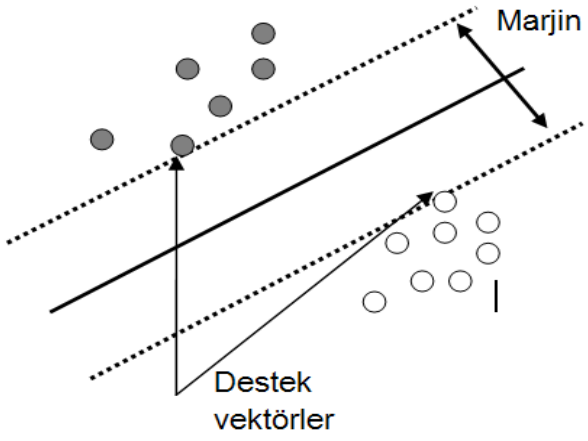
- Protein ayrılma bölgelerinin belirlenmesi
- Farmasötik veri analizleri

gibi birçok uygulama alanında başarıyla kullanılmıştır.

DVM'leri, sınıfları birbirinden ayıran marjini en büyük, doğrusal bir ayırt edici fonksiyon bulmayı amaçlar. Doğrusal olarak ayrılamayan veri, doğrusal olarak ayrılabileceği daha yüksek boyutlu başka bir uzaya taşınır ve sınıflandırma o uzayda yapılır. Destek vektör makinaları için düşük genelleştirme hatası veren en iyi ayırma düzlemini belirlemek temel husustur (20-21).

Normal regresyon yöntemleri, genellikle tüm eğitim örnekleri için kestirilen ve deneysel olarak gözlenen yanıtlar arasındaki en az sapmaya sahip f(x) fonksiyonunu türeten süreçler olarak ifade edilmektedir. DVR ise gözlenen eğitim hatasını minimize etmek yerine genelleştirme hatasını en aza indirmeye çalışır. Bu ilgili genelleştirme hatası, eğitim hatası ve hipotez uzayının karmaşıklığını kontrol eden düzenleme teriminin bir kombinasyonudur. Düzenleme terimi doğrudan hipotez uzayının VC-boyutuna bağlı olsa da hata (kayıp) fonksiyonu genellikle eldeki amaç doğrultusunda seçilir. Şekil 2.4'de gösterildiği gibi DVM, sınıflar arası maksimum aralığa sahip ayırıcı düzlemi bularak sınıflandırma yapar. Bulduğu düzlemin denklemi Formül (2.2) ile ifade edilir (22).

$$f(x) = \langle w,x \rangle + b \qquad (2.2)$$



Şekil 2.4. DVM sınıflandırıcı yapısı.

Burada, w ϵ R^n ağırlık vektorü, b ise skaler sabittir. Eğitim verileri n boyutlu x vektörleriyle temsil edilir. w ve x iç çarpımı ile b skaleri toplanıp fonksiyon sonucu elde edilir. Her bir eğitim verisi N boyutlu bir vektör aracılığıyla temsil edilir. Veri kümesini oluşturan m sayıda veri, y ϵ {+1,-1} kümesindeki elemanlardan birisine etiketlenirler ve Formül (2.3)' deki koşulu sağlamak zorundadırlar.

Burada $\xi_i \geq 0$ şartı sağlanır.

$$y_i [<w,x> + b] \geq 1- \xi_i, \qquad i=1,\ldots.m \tag{2.3}$$

Optimum ayırıcı düzlemin bulunması için (2.3) Formülündeki koşula bağlı olarak (2.4) Formülündeki amaç fonksiyonunun en küçük değeri bulunmalıdır.

$$C \sum_{i=1}^{n} \xi_i + \frac{1}{2} \|w\|^2 \tag{2.4}$$

Formül (2.3) formundaki koşullar kullanılarak Formül (2.4) en küçük yapılmalıdır.

Buradaki C, ayırıcı düzlemin karmaşıklığı ile sınıflandırma doğruluğu arasında denge kurmayı sağlayan kullanıcı tanımlı pozitif parametredir. Formül (2.3) ve (2.4) ile verilen klasik problem, Lagrange çarpanlarıyla birlikte bir iki yönlü (dual) optimizasyon problemi yapısında yazılırsa aşağıdaki denklem grubu elde edilir. Burada, Formül (2.5)'in ikinci kısmında bulunan eşitlik ve eşitsizlik halindeki koşullara göre birinci kısımdaki amaç fonksiyonu en büyük yapılmaya çalışılır. Hedef fonksiyon değeri girdi veri kümesinin iç çarpımına da bağlıdır (7).

Amaç fonksiyonu (en büyük)

$$Q(\alpha)=\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,k=1}^{n} \alpha_i y_i \alpha_k y_k < x_i, x_k > \tag{2.5}$$

Koşul

$\sum_{i=1}^{n} \alpha_i y_i = 0 \ , \ \alpha_i \geq 0, \forall i$

Pozitif değerli α_i Lagrange çarpanları kullanılarak aşağıdaki karar fonksiyonu oluşturulur. Bu tip çarpan değerlerine sahip veriler "destek vektör" adını alır. Bu veriler, veri kümesi içinde ayırıcı düzlemi en iyi temsil eden verilerdir.

$$f(x) = \text{sign}(\sum_{i=1}^{\#sv} \alpha_i y_i K(x, x_i) + b) \quad (2.6)$$

Formüller (2.2-2.6) kullanılarak kurulan optimizasyon probleminden en iyi Lagrange çarpan değerleri bulunur. Daha sonra bu değerlerle de ayırıcı düzlem oluşturulur. Bu düzlem sadece destek vektör noktaları esas alınarak oluşturulduğu için genel veri kümesinden çok daha seyrektir.

Buraya kadar anlatılan DVM sınıflandırıcısı doğrusal yapıdaki uygulamalarda başarıyla kullanılabilir. Ancak, doğrusal olmayan uygulamalarda da DVM'nin sınıflandırma yapabilmesi için çekirdek (kernel) fonksiyonlarına ihtiyaç duyulur.

Doğrusal olmayan girdi uzayındaki veriler çekirdek fonksiyonu ile doğrusal yüksek boyutlu nitelik uzayına dönüştürülür. Formül (2.7)'de verilen iç çarpım çekirdek fonksiyonu da bu dönüşümü sağlar (14, 19).

$$K(x, x_i) = K(x_i, x) = \varphi(x)^T \varphi(x_i) \quad (2.7)$$

DVM iyi bir şekilde (optimum) eğitildiği zaman sınıflandırma ve regresyon işlemlerinde çok iyi sonuçlar veren çekirdek fonksiyonlarına sahiptir. DVM yöntemi dağılımdan bağımsız olarak çalışabilir ve istatistiksel öğrenme teorisine göre eğiticili ve yarı eğiticili olarak sınıflandırma ve regresyon işlemlerini gerçekleştirebilir.

Regresyon için DVM'nın bir versiyonu Vapnik, Golowich ve Smola tarafından 1997 yılında ileri sürülmüştür. Bu yöntem, Destek vektör regresyonu (DVR) olarak adlandırılır. Destek vektör sınıflamasına göre üretilen model sadece eğitim verisi alt setine bağlıdır çünkü modeli oluşturmak için amaç (cost) fonksiyonu marjinin dışında kalan eğitim noktalarını dikkate almaz. Benzer şekilde DVR tarafından üretilen model de sadece eğitim verisinin bir alt setine bağlıdır, çünkü modeli oluşturmak için amaç fonksiyonu, model kestirimine yakın (ϵ eşik değeri içinde) herhangi bir eğitim noktasını göz ardı eder (23).

Özellikle gen ifadesi verileri yüksek boyutlu ve çoğu zaman doğrusal olmayan ilişkiye sahiptir. Genler arasındaki ilişkileri incelemek bu nedenle klasik regresyon yöntemleriyle oldukça zordur. Ayrıca dağılım varsayımlarının pratikte sağlanması oldukça zordur. DVM yöntemi kullandığı çekirdek fonksiyonlar sayesinde genler arasındaki ilişkileri anlaşılabilir hale getirir.

DVR, DVM'nın son yıllarda yaygın kullanılan biçimidir. N, girdi verilerinin uzayını ifade ettiğinde R^d, $(x_1, y_1), \ldots, (x_l, y_l)$ C N×R eğitim verisi olarak alınsın.

Ε-DVR'da amaç, tüm eğitim verileri için mevcut y_i hedeflerinden en fazla ε kadar sapmaya sahip olan f(x) fonksiyonunu bulmaktır. Diğer bir deyişle, hatalar ϵ'dan küçük olduğu sürece dikkate alınmaz (yani sıfır sayılır) ancak bu değerden daha büyük olan herhangi bir sapma kabul edilmemektedir. f'nin doğrusal fonksiyon durumu aşağıdaki gibi tanımlanmıştır:

$$f(x)= \langle w,x \rangle + b \qquad w \epsilon N,\ b \epsilon R \qquad (2.8)$$

<.,.> gösterimi, noktaların çarpımını ifade etmektedir. (2.8) Formülündeki düzlük (flatness), küçük w anlamına gelir. Bunu elde edebilmek için "$\|w\|^2$" öklid biçimini en küçük yapmak gerekir. Biçimsel olarak bu problem, dışbükey (konveks) bir optimizasyon problemi olarak aşağıdaki gibi yazılabilir:

Amaç

$$\min \ \frac{1}{2}\|w\|^2 \qquad (2.9)$$

Kısıtlar $y_i - \langle w, x_i \rangle - b \leq \epsilon$

$\langle w, x_i \rangle + b - y_i \leq \epsilon$

Yukarıdaki dış bükey optimizasyon problemi, f'nin gerçekte varolduğu ve ϵ kesinliği ile tüm (x_i, y_i) çiftlerine yakınsadığı varsayımından dolayı mümkündür. Ancak bu optimizasyon probleminde ek olarak bazı hatalar da belirlenebilir.

Cortes ve Vapnik tarafından 1995'de destek vektör makinaları için uyarlanan "soft marjin" amaç fonksiyonuna benzer şekilde, optimizasyon probleminin zorluk çıkaran kısıtlarını ele almak için ξ_i ve ξ_i^* esnek (slack) değişkenlerin tanıtılmasıyla formülizasyon aşağıdaki gibi olur (19, 23):

$$\min \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \qquad (2.10)$$

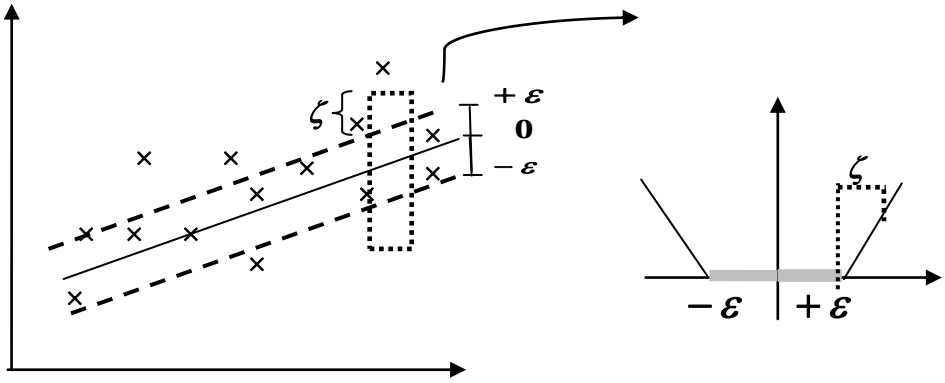
Kısıtlar $y_i - \langle w, x_i \rangle - b \leq \epsilon + \xi_i$

$\langle w, x_i \rangle + b - y_i \leq \epsilon + \xi_i^*$

$\xi_i, \xi_i^* \geq 0$

C>0 düzenleme parametresi, f'nin düzlüğü (flatness) ve tolere edilen ϵ'dan sapma miktarı arasındaki dengeyi belirler. (2.10) Formülasyonunun avantajlı bir özelliği de çözümün eğitim setindeki küçük değişikliklere karşı duyarlı olmasıdır. ϵ'a duyarsız kayıp fonksiyonu $|\xi|_\varepsilon$ aşağıdaki gibi tanımlanır:

$$|\xi|_\varepsilon = \begin{cases} 0 & |\xi| < \varepsilon \\ |\xi| - \varepsilon & \text{dd.} \end{cases} \qquad (2.11)$$



Şekil 2.5. Doğrusal destek vektör makinasına karşılık gelen yumuşak (soft) marjin amaç fonksiyonu ayarlaması.

Şekil 2.5, durumu grafiksel olarak göstermektedir. Sapmalar, doğrusal bir şekilde kısıtlandığı için kesikli çizgilerin dışındaki noktalar hataya (cost) önemli ölçüde katkıda bulunur. Çoğu zaman (2.10) ile verilen optimizasyon problemi, çift (dual) formülasyon için de kolay şekilde çözülebilir (19,20,22-23). Üstelik çift yönlü formülasyon, DVM'nı doğrusal olmayan fonksiyonlara genişletmek için anahtar sağlar. Bu nedenle Lagrange çarpanlarının kullanıldığı standart bir çift yönlüleştirme (dualizasyon) yöntemi sonraki bölümde açıklanmıştır.

Dual Problem ve Üstel Programlar

Çift (dual) formülasyon, destek vektör makinasını doğrusal olmayan fonksiyonlara genişletmek için bir anahtar sağlar. Temel esas, amaç fonksiyonu ve uygun kısıtlardan (çift değişkenler setini tanıtarak) Lagrange fonksiyonunu oluşturmaktır. Bu fonksiyonun tek ve çift değişkenler açısından bir eyer noktasına sahip olduğu gösterilebilir:

$$L:\frac{1}{2}\|w\|^2+C\sum_{i=1}^{l}\left(\xi_i+\xi_i^*\right)-\sum_{i=1}^{l}\left(\eta_i\xi_i+\eta_i^*\xi_i^*\right) \qquad (2.12)$$

$$-\sum_{i=1}^{l}\alpha_i(\varepsilon+\xi_i-y_i+<w,x_i>+b)$$

$$-\sum_{i=1}^{l}\alpha_i^*(\varepsilon+\xi_i^*+y_i-<w,x_i>-b)$$

Yukarıda L, lagrange fonksiyonu ve $\eta_i, \eta_i^*, \alpha_i, \alpha_i^*$ ise lagrange çarpanlarıdır. Dolayısıyla (2.12)'deki çift (dual) değişkenler pozitiflik şartını sağlamalıdır.

$$\alpha_i^*, \eta_i^* \geq 0 \qquad (2.13)$$

Eşik noktası koşulundan L'nin tekli değişkenlere göre (w, b, ξ_i, ξ_i^*) kısmi türevleri en uygun çözüm için sıfıra eşitlenmelidir.

$$\frac{\partial L}{\partial b}=\sum_{i=1}^{l}\left(\alpha_i^*-\alpha_i\right)=0 \qquad (2.14)$$

$$\frac{\partial L}{\partial w}=w-\sum_{i=1}^{l}\left(\alpha_i^*-\alpha_i\right)x_i=0 \qquad (2.15)$$

$$\frac{\partial L}{\partial \xi_i^*}=C-\alpha_i^*-\eta_i^*=0 \qquad (2.16)$$

(2.14), (2.15) ve (2.16)'yı (2.12)'de yerine koymak çift optimizasyon problemini verir.

Amaç (en büyük)

$$-\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)<x_i,x_j>-\varepsilon\sum_{i=1}^{l}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{l}y_i(\alpha_i-\alpha_i^*) \qquad (2.17)$$

Kısıtlar $\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0 \text{ ve } \alpha_i, \alpha_i^* \in [0, C]$

Çift değişkenler η_i, η_i^*, (2.17) Formülünü türetmek için (2.16) koşulu ile elimine edilmiştir. Formül (2.15) yeniden aşağıdaki gibi yazılabilir:

$$w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)x_i$$

ve bundan dolayı,

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) < x_i, x > +b \qquad (2.18)$$

(2.18) Formülü, destek vektör genişlemesi olarak adlandırılır. w, x_i eğitim örneklerinin doğrusal bir kombinasyonu olarak yazılabilir (7, 18, 20, 22-23).

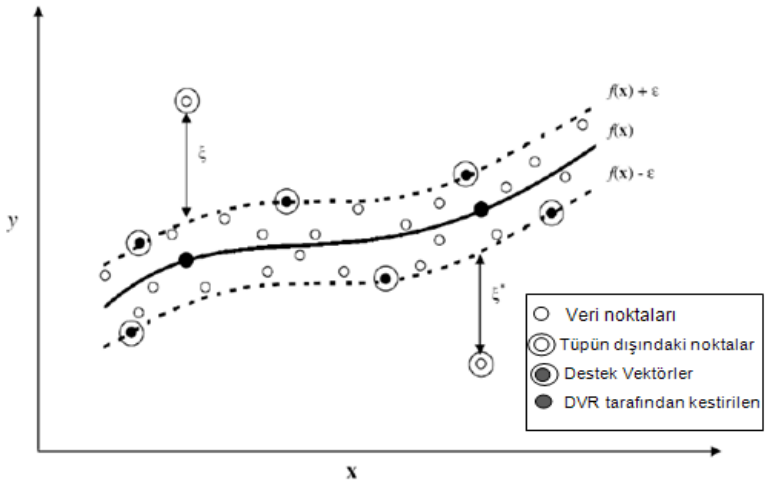
Bir anlamda fonksiyonun destek vektörler tarafından temsil karmaşıklığı girdi uzayı X'in boyutundan bağımsızdır ve sadece destek vektörlerinin sayısına bağlıdır. f(x)'i değerlendirirken w'nin hesaplanmasına gerek duyulmaz, algoritmanın tamamı veriler arasındaki çarpım sonuçlarına göre tanımlanabilir. Bu durum, doğrusal olmayan tipinin formülasyonu için yararlıdır (7, 23).

2.2.2.1. Doğrusal Olmayan Durumda Destek Vektör Regresyon

Doğrusal olmayan koşullar için çözüm, orijinal problemin çarpım sonuçlarının bir çekirdek fonksiyon tarafından $K(x_i,x_j)= <\Phi(x_i), \Phi(x_j)>$ temsil edilebildiği, yüksek boyutlu karakteristik bir uzaydaki doğrusal koşullara eşleştirerek bulunabilir. $K(x_i,x_j)= <\Phi(x_i), \Phi(x_j)>$ ifadesini Formül (2.18)'de yerine koyarak doğrusal olmayan DVR algoritması aşağıdaki gibi elde edilir.

$$w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)x_i,\ f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\,K < x_i, x > +b \quad (2.19)$$

DVM regresyon modelleri regresyon doğrusu etrafında bir tünel "funnel" oluşturur. Bu tünelin içindeki artıklar parametre kestiriminde sayılmaz (yani sıfır sayılır), ve tünelin dışındaki artıkların kareleri değil toplamları amaç fonksiyonu olarak kullanılır. Doğrusal olmayan DVR'nun şekilsel gösterimi aşağıda verilmiştir (Şekil 2.6).



Şekil 2.6. ϵ'a duyarsız kayıp fonksiyonu kullanan doğrusal olmayan DVR diyagramı.
Lahiri ve Ghanta (24)'den alınmıştır.

Uygulamada yaygın kullanılan çekirdek fonksiyonları; doğrusal, radyal tabanlı (gaussian), üstel (exponential) radyal tabanlı fonksiyon, polinomiyal, ve sigmoid çekirdek fonksiyonlarıdır.

Doğrusal Çekirdek Fonksiyonu:

$$K(x_i, x_j) = x_i^T x_j \quad (2.20)$$

➢ Doğrusal olmayan modelleme için *polinomiyal çekirdekler*, homojen ve homojen olmayan polinom çekirdekler olmak üzere d Є N, ve c ≥0 ikiye ayrılır*:*

$K(x_i, x_j) = (\gamma x_i^T x_j)^d$ (2.21)

$K(x_i, x_j) = (\gamma(x_i^T x_j + r))^d$ (2.22)

➢ *Radyal tabanlı fonksiyonlar* özellikle de gaussian radyal tabanlı fonksiyon oldukça ilgi çekmektedir:

$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$ (2.23)

$= \exp\{-\gamma \|x_i - x_j\|^2\}$

➢ *Exponential radyal tabanlı fonksiyon:*

$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|}{2\sigma^2}\right)$ (2.24)

➢ Bir diğer çekirdek fonksiyon ise, yapay sinir ağları'na (Multi-layer perceptron-çok katmanlı algılayıcı) benzerliği ile dikkat çeken s*igmoid* çekirdek fonksiyonudur:

$K(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$ (2.25)

Spline çekirdek fonksiyon ise esnekliğinden dolayı modelleme için popüler bir seçimdir (20, 25):

$K(x_i, x_j) = 1 + x_i^T x_j + \frac{1}{2} x_i^T x_j \, en\ küçük(\boldsymbol{x_i}, \boldsymbol{x_j}) - \frac{1}{6} en\ küçük(\boldsymbol{x_i}, \boldsymbol{x_j})^3$ (2.26)

Tez çalışmasında DVR uygulaması içinde doğrusal, polinomiyal, radyal tabanlı ve sigmoid fonksiyonların performans sonuçları üzerinde durulacaktır.

2.2.2.2. En Uygun Parametrelerin Ayarlanması

DVR uygulamasında dikkate alınması gereken birkaç husus vardır. Hepsinden önce belirli bir algoritmayı çalıştırmadan önce bazı parametrelerin belirlenmesi gerekir. DVR'nin genelleştirme performansı C, ε ve kullanılan çekirdek fonksiyona özel parametrelerin iyi şekilde ayarlanmasına bağlıdır. En uygun parametre seçim problemi, DVR modeli karmaşıklığı nedeniyle daha da zorlaşır ve her üç parametreye de bağlıdır. Aşağıda C, ε ve çekirdek fonksiyona bağlı parametreler R fonksiyonundaki varsayılan değerleri ile birlikte kısaca açıklanmıştır:

d (derece): Polinomiyal tipi çekirdek için gerekli parametre. Varsayılan değeri 3'tür.
γ (Gamma): Çekirdek tanım fonksiyonunda yer alan doğrusal hariç tüm çekirdekler için gerekli parametredir. Pozitif, gerçel bir sayı değeri alabilir. Varsayılan değeri "1/ (verinin açıklayıcı değişken boyutu)" dur.
r (Coef0): Polinomiyal ve sigmoid tipi çekirdekler için gerekli parametre. Varsayılan değeri sıfırdır.
C (Cost): Kısıtlamanın ihlal edilme maliyetidir. Lagrange fonksiyonundaki düzenleme terimi "C" sabitidir. Program tanımlı değeri "1" dir.
ε (Epsilon): ε- duyarsız kayıp fonksiyonundaki epsilon değeri. Belirli bir değer seçilmez ise programdaki varsayılan değeri 0.10'dur.

C parametresi, model karmaşıklığı ile ε'dan ne kadar büyüklükteki sapmanın (2.10) Formülü ile verilen optimizasyon denkleminde tolere edilebileceği arasındaki dengeyi belirler. Örneğin C, aşırı büyük ise (sınırsızlık), bu durumda amaç, (2.10) optimizasyon formülasyonundaki model karmaşıklığı kısmına bakılmaksızın yalnızca deneysel riski (2.9) en küçük yapmaktır.

C değerini artırmak kestirim hatası maliyetini arttırır ve belki de iyi genelleştirilemeyen daha yüksek doğrulukta bir model oluşmasını zorlar (26).

ε parametresi, eğitim verilerini modellemede kullanılan duyarsızlık bölgesinin genişliğini kontrol eder. ε değeri, regresyon fonksiyonunu oluşturmak için kullanılan destek vektörlerin sayısını etkileyebilir. Daha büyük ε için daha az sayıda destek vektörü seçilir. Diğer yandan daha büyük ε değerleri daha düz (flat) kestirimlere yol açar. Dolayısıyla hem C, hem de ε değerleri model karmaşıklığını farklı yollarla etkiler.

Mattera ve Haykin, ε değerinin DVM regresyon modelindeki destek vektörlerin yüzdesinin yaklaşık olarak örneklem sayısının %50'si olacak şekilde seçilmesini önermiştir (21). Ancak araştırmacı, en iyi (optimum) genelleştirme performansını %50'den daha küçük ya da daha büyük destek vektörlerin sayısı ile elde edildiği örnekleri kolaylıkla gösterebilir.

DVR modelleri oluşturulurken özellikle cost (C), gamma (γ) ve ε parametrelerinin optimizasyonuna özen gösterilmelidir. Sonlu sayıda öğrenme verisine (N) dayalı bir f modelinin herhangi bir kestirimi, düşük öğrenme yanlılığı (small bias) ve küçük varyanslı bir model arasında uzlaşma şeklinde seçilir. DVM çözümünde bu uzlaşma gamma (γ) parametresinin yeterli bir seçimi yoluyla uygulanmaktadır. Eğer gamma (γ) değeri çok büyük ayarlanırsa model gürültü de dahil olmak üzere veriyi aşırı belirleyecektir (21).

Çalışmada incelenen çekirdek fonksiyonların tümü için en iyi üç parametrenin optimizasyonu R programındaki tune() fonksiyonu yardımıyla gerçekleştirilmiştir. C'nin tune.svm() komutunda arama yapılan değer aralığı; 10^{-2} ile 10^{2} arası (yani 5 arama noktası) şeklinde belirlenmiştir.

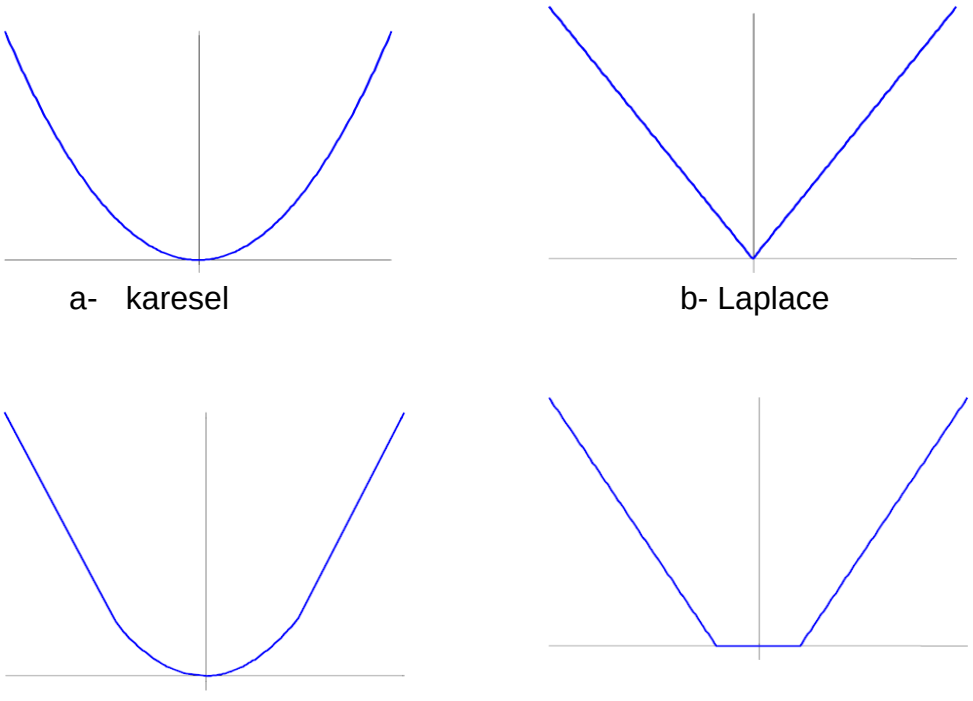
γ'nın arama yapılan değer aralığı ise 10^{-6} ile 10^{-1} aralığı (yani 6 arama noktası) olarak belirlenmiştir. Dolayısıyla tune.svm() fonksiyonu bu iki parametre için modeli toplam 5*6= 30 arama noktası için çapraz geçerlik kestirim hatası açısından değerlendirmiştir. R kodu, son olarak en küçük HKO değerli parametreleri tespit etmiştir.

C ve γ (gamma) parametreleri ile eş zamanlı olarak ε'nun 0.05, 0.10 ve 0.20 değerleri için de arama yapılmış ve tespit edilen en iyi değerler, benzetim senaryolarında gösterilmiştir (19-20, 27).

Hata (Loss) Fonksiyonu

Є'a duyarsız hata fonksiyonun yanı sıra DVR için kullanılabilen bazı alternatif hata fonksiyonları da mevcuttur. Bu hata (loss) fonksiyonu bir uzaklık ölçüsü içerecek şekilde değiştirilmelidir. Sık kullanılan dört fonksiyon Şekil 2.7'de verilmiştir.

İkinci dereceden hata fonksiyonu (Şekil 2.7-a), klasik EKK hata kriterine uymaktadır. Şekil 2.7-b'deki Laplace hata fonksiyonu aşırı değerlere karşı karesel hata fonksiyonuna göre daha az duyarlıdır. Üçüncüsü, verinin temel dağılımı bilinmediğinde iyi özelliklere sahip olan Huber'in hata fonksiyonu sağlam bir fonksiyondur.



a- karesel b- Laplace

c- Huber d- ε'a duyarsız

Şekil 2.7. Hata fonksiyonları.

Bu üç hata fonksiyonu destek vektörlerinde seyrekliğe yol açmamaktadır. Bu soruna çözüm olarak Vapnik, ϵ-duyarsız hata fonksiyonunu önermiştir. Bu fonksiyon (Şekil 2.7-d) diğerlerinden farklı olarak elde edilecek destek vektörlerin seyrek bir setini sağlamaktadır (28).

2.2.3. Regresyon Ağaçları

Sınıflandırma ve regresyon için ağaca dayalı yöntemler istatistiksel açıdan Breiman tarafından 1984'de tanıtılmıştır (29). Sınıflandırma ve regresyon analizi (SRA), özellikle biyolojik araştırmalarda son yıllarda popülerliği büyük ölçüde artmış parametrik olmayan yoğun hesaplama gerektiren yöntemlerdir. SRA, ayırma analizi ve klasik EKK regresyon yöntemi gibi modele ilişkin varsayımlara bağlı olmayan sınıflandırma ve regresyon için alternatif yaklaşımlardır. Sınıflandırma ağaçları kategorik yanıt değişkeni için kullanılırken regresyon ağaçları, yanıt değişkeni sürekli olduğunda kullanılır. SRA yöntemi, hem çok sayıda gözleme hem de çok sayıda değişkene sahip veri setlerine uygulanabilir ve aşırı değerlere karşı da oldukça dayanıklıdır (29).

Ağaca dayalı modeller, açıklayıcı değişkenler için en homojen yanıtlara sahip bölgeleri belirlemek için bir dizi kural kullanarak açıklayıcı değişken uzayını dikdörtgenlere bölümlerler. Ardından her bir bölgeye; sınıflandırma ağaçları için bir sabit olarak en olası sınıfı atayarak ve regresyon ağaçları için normal dağılımlı hataları varsayarak o bölgedeki gözlemlerin ortalama yanıtını oluşturarak her bir bölge için bir sabit modellerler. Sonuçta elde edilen dikdörtgen bölgeler son düğüm ya da yapraklar ve t_1, t_2'ler ise bölünme noktalarıdır. Kestiriciler ve bölünme noktaları kestirim hatalarını en aza indirmek için seçilirler.

Bir ağacı geliştirmek özyinelemeli ikili bölünmeleri içerir. Bu ikili bölünme bazı durdurma kriterlerine ulaşana kadar kendi çıktısına tekrarlı olarak uygulanır. Tekbir karar ağacı oluşturmak için etkili bir strateji, büyük bir ağaç geliştirmek ve ardından çapraz geçerlik yöntemi ile tespit edilen en zayıf bağlantıları kırarak onu budamaktır (30, 28).

Bir ağaç ilk olarak tüm gözlemleri içeren bir "kök" düğüm dikkate alınarak geliştirilir. Bu düğümdeki gözlemler, tek bir açıklayıcı değişken üzerinde bir bölünme noktası "split" kullanarak biri sol ve diğeri sağda olmak üzere iki alt düğümden birine gönderilir. Sürekli bir açıklayıcı değişken için bölünme tek bir bölünme noktası ile belirlenir. Açıklayıcı değişken değeri bu bölünme noktasından daha küçük olan gözlemler sola giderler, kalan gözlemler ise sağa giderler. Kategorik bir açıklayıcı değişken için bölünme, kategorilerin bir alt setini sola, diğerlerini ise sağa gönderir. Bir ağacın belirli bir bölünmesi, bir düğümü her açıklayıcı değişken üzerinde her olası bölme noktasını dikkate alarak seçilen iki alt düğüme bölme işlemini uygular. Bazı kriterlere göre en iyi değeri veren bu kestirici ve bölünme noktası kombinasyonu düğümü bölümlemede kullanılır (29,31).

Bölünme kriterlerinin örnekleri, Breiman ve diğerleri tarafından 1984'te verilmiştir ve regresyon ağaçları (RA) için *artık kareler toplamını;* sınıflandırma ağaçları için ise *Gini İndex ve Entropy gibi* homojenlik ölçülerini içerir. Bölünme işlemleri tekrarlanmış bölümleme "recursive partitioning" olarak adlandırılan alt düğümler için de sırayla uygulanır (32).

Genellikle ağaçlar bir durdurma kriteri ile karşılaşana kadar geliştirilir. Örneğin tüm düğümler belirlenmiş bir sayıdan daha az sayıda gözlem içerdiğinde büyüme sona erer. Daha sonra aşırı uyumu önlemek için ağaç geri budanır. Bir ağaç geliştirildikten ve budandıktan sonra "son düğüm" olarak adlandırılan bazı bölünmemiş düğümlere sahip olacaktır. Bir düğümdeki gözlemlerden kestirim değerleri elde edilir. Bu, regresyon problemlerinde yanıt değişkeninin ortalaması alınarak ya da sınıflandırma problemleri için ya sınıf üyelikleri oranlarını hesaplayarak ya da en sık görülen sınıfı bularak elde edilir (31-32).

Temel olarak RA, açıklayıcı değişkenler üzerinde karar kurallarının bir setini oluşturan yöntemdir. Regresyon ağacı analizinin amacı, bir sürekli “Y” yanıt değişkeninin, sürekli, ordinal ve nominal değişkenlerin rastgele bir karışımı olabilen $X=X_1, X_2,\ldots,X_n$ açıklayıcı değişken vektörü tarafından açıklanması olarak ifade edilebilir.

Yanıt değişkeni ve açıklayıcı değişkenler arasındaki ilişkinin önceden belirlendiği ve bu ilişkiyi kanıtlamak ya da çürütmek için test yapılan klasik regresyon yöntemlerinden farklı olarak RA, böyle bir ilişkiyi yok sayar (32-34).

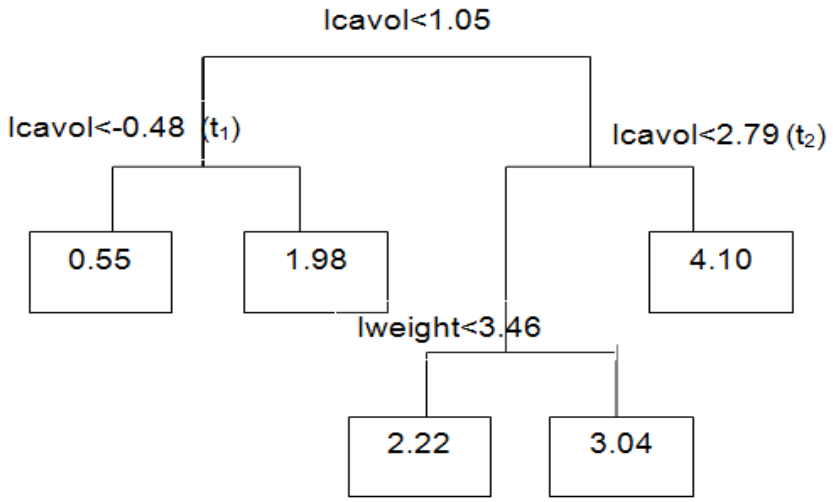
Karar kuralları, verinin tek bir açıklayıcı değişkene dayalı ikili bölünmeler ile ardışık olarak daha küçük gruplara bölümlenmesi şeklinde oluşturulur. Daha sonra bu alt bölümler yeniden bölümlenir, yapılan bu işleme özyinelemeli bölümleme denir. Açıklayıcı değişkenlerin tümü için bölünmeler kapsamlı bir arama prosedürü tarafından incelenir ve en iyi bölünme seçilir. Örneğin CART algoritması bu amaçla kullanılan algoritmalardandır.

Regresyon için CART, sınıflandırmadan farklı olarak safsızlık ölçüsü olarak artıkların kareler toplamını kullanır ve eğitim örneği ortalaması tarafından oluşturulan her bir yaprak düğüm ile parça parça sabit bir model oluşturur. Regresyon ağaçları için seçilen bölünme, iki sonuç grubunun yanıt değişkenine göre kendi içindeki homojenliğini en büyük yapan bölünmedir. Bu bölünme varyans analizindeki (ANOVA) gibi gruplar arası kareler toplamını en büyük yapar. Bu bölünme hiçbir bölünme kuralı düğümlerin homojenliğini önemli ölçüde geliştiremediği zaman durur.

Sonuçta, dalları bölünme kuralları ve ortalama yanıtları içeren son düğümlerin bir dizisi ile belirlenmiş ağaç diyagramı elde edilir. Bu yöntem, ilk başta maksimum ağacı büyütür ve daha sonra bu aşırı belirlenmiş ağacı (overfitted tree) uygun bir boyuta budamak için çapraz geçerlik gibi yöntemleri kullanır (33,29).

Bölünme noktalarına içsel düğümler de denir ve alt grupları olmayan düğümler ise son düğümlerdir. M son düğümlü ikili (binary) bir ağaç M-1 tane içsel düğüme sahiptir. Son düğümlerin sayısı modeldeki hasta gruplarının sayısını temsil ederken, içsel düğümlerin sayısı hastaların sınıflandırılması için gerekli kuralların sayısı olarak yorumlanabilir.

Regresyon problemleri için son düğümler, yanıt değişkenlerinin ortalamaları tarafından oluşturulur (33). Şekil 2.8, Stamey ve diğerlerinin 1989'daki, prostat kanseri ile ilgili çalışmasından elde edilen veriler için bir regresyon ağacını göstermektedir. Bu veriye Hastie ve diğerleri tarafından da 2001'de çalışılmıştır (35). Yanıt değişkeni prostata özgü antijen seviyesidir (lpsa).



Şekil 2.8. Regresyon ağacı örneği.
Stamey ve diğerleri (36)'den alınmıştır.

Gösterim amacıyla sadece iki açıklayıcı değişken "log kanser hacmi (lcavol) ve log prostat ağırlığı (lweight)" alınmıştır. Herbir düğümde eşitsizliği sağlayan olgular sola, eşitsizliği sağlamayan olgular ise sağa gitmiştir. Herbir son düğüm tek bir kestirim değerinde sonuç verir, yani bu değer o düğüme düşen gözlemler için yanıt değerlerinin ortalamasıdır (31, 35).

Bir regresyon ağacı, ortak değişken etkisinin farklı bölgeleri üzerinde parçalı sabit bir modeldir. Veri setinin, N gözlem i=1, 2, ….,N, p açıklayıcı değişken ve bir yanıt değişkeninden; $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$, (x_i, y_i) oluştuğu varsayılsın. M bölge şeklinde $R_1, R_2, \ldots, R_M$ bir bölünmede her bir bölgedeki yanıt değişkeni bir c_m sabiti olarak modellenir:

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \quad (2.27)$$

Burada I(), gösterge fonksiyon ve c_m, R_m'deki kestirim değeridir. Hata kareler toplamını en küçükleme kriteri kullanılarak $\sum(y_i - f(x_i))^2$ en iyi $\widehat{c_m}$ değerinin R_m bölgesindeki y_i değerlerinin ortalaması olduğu görülebilir:

$$\widehat{c_m} = ortalama(y_i \backslash x_i \in R_m) \quad (2.28)$$

Regresyon ağacının oluşumu aşamasında her bir j açıklayıcı değişkeni ve s bölünme noktası için Greedy algoritması izlenerek iki bölge oluşturulur:

$R_1(j,s)=\{X\backslash X_j \leq s\}$ ve $R_2(j,s)= \{X\backslash X_j > s\}$ (2.29)

Daha sonra ilk bölünme "j" bölünme değişkeni ve "s" bölünme noktası Formül (2.30) çözülerek elde edilir:

$$\min_{j,s} \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right] \quad (2.30)$$

Birinci bölünmenin gerçekleştirilmiş olması ile Formül (2.30), verinin ardışık bölümleri üzerinde özyinelemeli olarak uygulanır. Bu süreç tarafından üretilen bölümler ağacın düğümleri olarak adlandırılır ve en son bölünmeler ise ağacın son düğümleri olarak isimlendirilir. M son düğümlü bir regresyon ağacı en yüksek etkileşim derecesinin M-1 olduğu bir model kurar (30, 35, 37).

RA'nın Avantajlı Özellikleri

- ✓ Kestirimleri yapmak hızlıdır, karmaşık hesaplamalar yoktur, sadece ağaçtaki sabitlere bakılır.
- ✓ Bu yöntem, aşama sıralı (hiyerarşik) ya da eklemeli olmayan (nonadditive) veri yapısını ortaya çıkarmada etkilidir.

- ✓ Yanıt değişkeni ve açıklayıcı değişkenler arasındaki ilişkilerin yapısı hakkında hiçbir varsayım yapmadığından RA, değişkenler arası etkileşimleri ve doğrusal olmayan ilişkileri de dikkate alır.
- ✓ RA'nın bölünme kuralları, kestiricilerin konumsal etkisine ilişkin daha çok bilgi sağlayarak dağılımlar üzerinde en büyük etkiye sahip açıklayıcı değişkenlerin belirlenmesini sağlar.
- ✓ Bu ağaçları öğrenmek için hızlı ve güvenilir algoritmalar mevcuttur.

Regresyon ağaçları klasik istatistiksel yöntemlerin üzerinde yukarıda belirtilen avantajlara sahiptir ancak belirli dezavantajları da vardır (30,34).

RA'nın Dezavantajlı Özellikleri

- ✓ Basit doğrusal fonksiyonlar RA ile büyük ölçüde kestirilebilir.
- ✓ Belirli veri kümeleri için çapraz geçerlik kullanarak en uygun budama parametrelerini seçerek modeli sınırlamak zordur.
- ✓ Yanıt değişkeni, regresyon ağaçlarında kurulan eşik değere bağlı olarak süreksiz, kesikli ya da az örneklemli olabilir.
- ✓ Sonuç kararsız olabilir, yani verideki küçük değişiklikler son derece farklı ağaçları üretebilir (34).

2.3. Genelleştirme Yöntemleri

Gen ifade analizinin en temel adımlarından biri analizlerin genelleştirilmesidir. Kestirim yapılacak ise, önce bir kestirim modeli oluşturulur sonra bu modele göre yeni gelen hastaya ait gen ifade düzeyi ya da hastaya verilecek ilaç dozu tahmin edilmeye çalışılır. Eğer veri setindeki hastaların tamamına ait veriler bir kerede analiz edilirse, sonuçların şansa bağlı olma olasılığı yüksek olur. Başka bir deyişle algoritma veri setine göre model oluşturduğu için, hasta grubunu tanır ve sadece o hasta grubunda işe yarayan bir model oluşturur. Aynı populasyondan alınacak yeni hastalarda yanlış sonuçlar verir.

Bu nedenle hastaların bir kısmı dışarıda bırakılıp analiz modelleri oluşturulmalı, daha sonra dışarıda kalan kısım ile elde edilen modeller test edilmelidir. Bunu yapabilmek için kullanılan en popüler iki yöntem; bootstrap ve çapraz geçerlik yöntemleridir (2).

Gen ifade verilerini analiz etmedeki temel zorluk, elde edilen çeşitli sınıflama/ayırma ya da regresyon modelleri içinden model seçimidir. Sonuçları raporlamada önemli bir adım, seçilen modelin hata oranını ya da genelleştirilebilirliğini değerlendirmektir. Bağımsız bir değerlendirme (test) verisi bulunmuyor ise, kestirim doğruluğunu tahmin etmek için yaygın bir yaklaşım, orijinal verinin yeniden örneklenmesi (resampling) ya da parçalara bölünmesi (splitting) gibi biçimlere dayalıdır.

Yeniden örnekleme yöntemlerinden rasgeleliğe dayalı testler (randomization test, randomization exact test, permutation test, rerandomization test) 1935'de Fisher tarafından geliştirilmiştir (11). O yıllardan bu güne uzanan çalışmaların hala artan bir şekilde devam etmesi, yeniden örnekleme yöntemlerinin tam olarak her alanda eksiklerinin giderilmediğini göstermektedir. Çalışmaların son yıllarda artışı ise ucuz ve hızlı bilgisayarların geliştirilmesine bağlanabilir.

Gelişen bilgisayar teknolojisinin sonucu olarak yeniden örnekleme yöntemleri istatistik bir araç olarak yaygın bir şekilde kullanılma eğilimindedir. Çünkü bu yöntemler oldukça güçlü, kolay ve hesaplamaları uygulamada yaygın olarak yer bulmaktadır. Yeniden örnekleme yöntemlerinin sahip olduğu avantajlardan biri de çok fazla matematik bilgisi ya da istatistik formül ihtiyacının olmamasıdır. Kavramların ve yöntemlerin anlaşılmasını sağlayacak kadar bir bilgi yeniden örnekleme yaklaşımının uygulamaları için yeterlidir.

Yeniden örnekleme yaklaşımının temelde klasik testlerden ayrıldığı noktalar şöyle açıklanmaktadır. Klasik testlerin teorik örnekleme dağılımlarına ait gözlenen istatistikleri karşılaştırdığı bilinmektedir. Yeniden örnekleme yöntemi aynı örnek içinde tekrarlamalı örneklemeye dayandırılır. Ayrıca şansa dayalı Monte Carlo benzetim çalışmasında veriler tamamen varsayım üzerine yani kuramsal temeller üzerine oluşturulabilir.

Yeniden örnekleme yaklaşımında ise, veri seti gerçek verilere dayandırılmalıdır. Yeniden örnekleme yaklaşımı şansa dayalı (Monte Carlo) benzetim ile işte bu noktalarda ayrılmaktadır.

Literatürde holdout, rastgele alt örnekleme, çapraz geçerlik ve bootstrap gibi yöntemler, mevcut verilerin rastgele örnekleme bölümlerine dayalı doğruluğunu değerlendirmek için yaygın kullanılan yöntemlerdir. Bu gibi yöntemlerin kullanımı doğruluk tahmininde toplam hesaplama süresini uzatır; ancak model seçimi için yararlıdır. Tez çalışmasında yaygın kullanılan iki yöntem detaylı incelenecektir. Bunlar;

1- Bootstrap
2- Çapraz Geçerlik

yöntemleridir.

2.3.1. Bootstrap

Doğruluk tahmini için sık kullanılan bir yeniden örnekleme yöntemi bootstrap'tir. Efron tarafından 1979'da önerilen bootstrap yöntemi, küçük örneklem boyutunu ele almada doğru bir yoldur (3).

Birini dışarıda bırakma çapraz geçerliğe (BDBCG) benzer şekilde bootstrap yöntemi de yaklaşık n büyüklüğündeki bir eğitim seti ile algoritmanın beklenen hatasını yakalamaya çalışır.

Hesapsal olarak BDBCG'den daha etkindir. Bootstrap yöntemi sıklıkla BDBCG'nin daha büyük varyansa sahip olduğu küçük veri setleri üzerinde kullanılır (11). Bootstrap'in potansiyel başarısı da çapraz geçerlik yaklaşımına benzer şekilde temel veri madenciliği algoritmasının sabit kalması varsayımına dayalıdır (8,38).

"n" genişlikli bir veri seti verildiğinde, veriden yerine koyarak "n" örneklemin rastgele seçilmesiyle bir bootstrap eğitim seti oluşturulur. Her seferinde bir örnek seçilir ve bu örneğin yeniden seçilmesi eşit olasılıklıdır. Çekilen örnek eğitim setine yeniden eklenir. Yani bir eğitim setinde aynı örneğin birden fazla sayıda seçilmesi ya da hiç seçilmemesi olasıdır.

Bootstrap örneklemesinden sonra herbir veri noktasının seçilme olasılığı $(\frac{1}{n})^n$ iken, seçilmeme olasılığı ise $(1-\frac{1}{n})^n = e^{-1} \approx 0.368$ 'dir. Sonuçta ortalama olarak orijinal veri dizilerinin % 63.2'si bootstrap eğitim seti için elde edilirken, kalan % 36.8'i test setini oluşturmaktadır.

Güvenilir ve kestirilebilir modeller oluşturmak için eğitim ve test setlerinin seçimine özen gösterilmelidir. Test seti eğitim setinin iyi bir temsilcisi olursa model performansının doğru bir kestirimi elde edilebilir. Bu örnekleme yöntemi B kere tekrarlanabilir ve seçilen bu bootstrap örneklemlerinin herbiri modeli eğitmek için kullanılır.

Daha sonra model kestirim hatasını hesaplamak için elde edilen modeller ya orijinal veri setine ya da bootstrap örneklemine dahil olmayan veriye uygulanır. Bu süreç B defa tekrarlanır ve bootstrap hata kestirimi tüm bootstrap örneklemleri üzerindeki ortalama kestirim hatası şeklinde elde edilir.

Bu yöntemdeki temel sorun kestirim hatasını hesaplamada orijinal veri setinin mi ya da sadece bootstrap örnekleminde yer almayan gözlemlerin mi kullanılacağıdır. Orijinal örneklem kullanıldığında model göreli olarak iyi kestirimler verecektir; çünkü bootstrap örneklemi (eğitim seti) orijinal örnekleme (test seti) oldukça benzerdir. Bu yanlılığı azaltmak için bir çözüm Efron ve Tibshirani (1998)'nin "0.632 kestiricisi"ni kullanmaktır (33,39-40).

Bootstrap hata kestirimi, sadece test veri seti üzerinden elde edilebilmesine rağmen uygulamada diğer formüller de kullanılmıştır. Bunlar özellikle bootstrap kestirimlerindeki aşırı uyumu ve kestirimlerdeki yanlılığı gidermek için ayarlanmış formüllerdir (19). Örneğin 0.632 bootstrap hata kestirimi aşağıdaki gibi tanımlanır:

$$Hata/doğruluk_{boot} = \frac{1}{B}\sum_{i=1}^{B}[0.632 \times testhata_i + 0.368 \times tophata_i] \quad (2.31)$$

B: hata kestirimini elde etmek için kullanılan rastgele bootstrap örneklem sayısı

$testhata_i$: i'inci bootstrap örneklemi ile elde edilen model i'inci test setine uygulandığında elde edilen hata

$tophata_i$: i'inci bootstrap ile elde edilen modelin veri noktalarının orijinal veri setine uygulandığında elde edilen hatadır.

Bootstrap yöntemi küçük veri setleri ile iyi çalışır. Bunke ve Droge'un 1984'de ve Efron'un 1979 yılındaki çalışmalarında bootstrap yaklaşımlarının bazı türlerinin çapraz geçerlik gibi yöntemlere göre kestirim hatasının daha iyi kestiricilerini sağladığını ileri sürmüşlerdir (örneğin çok fazla hesaplama gerektiren yöntemler ya da stepwise regresyon gibi). Ancak bootstrap, bootstrap genelleştirme kestirimlerinin aşırı iyimser olabileceği ağaca-dayalı modeller gibi bazı yöntemler için iyi çalışmamaktadır (41).

2.3.2. Çapraz Geçerlik

Bu bölümde ilk olarak Hold-out yöntemi kısaca açıklanmaktadır.

2.3.2.1. Holdout Yöntemi ve Rastgele Alt Örnekleme

Hold-out yönteminde belirli bir veri seti rastgele olarak eğitim ve test seti olarak adlandırılan iki bağımsız veri setine bölünür. Genellikle verinin üçte ikisi eğitim seti ve kalan üçte biri ise test seti olarak tahsis edilir. Eğitim seti ile oluşturulan modelin doğruluğu test seti ile kestirilir. Bu kestirim kötümserdir (pessimistic); çünkü modeli oluşturmak için başlangıç verilerinin sadece bir kısmı kullanılır.

Hold-out yönteminde aşırı uyumu önlemek için bağımsız bir test seti tercih edilir. Yöntemin olumsuz yanı, mevcut tüm veriyi kullanmaması ve sonuçların eğitim/test bölüntüsünün seçimine yüksek derecede bağlı olmasıdır. Test setine dahil edilmek üzere seçilen örnekler çok kolay ya da çok zor sınıflanabilir/kestirilebilir. Bu ise sonuçları saptırabilir. Holdout doğrulama çok kez tekrar edilerek ve sonuçların ortalaması alınarak bu problem kısmen giderilebilir.

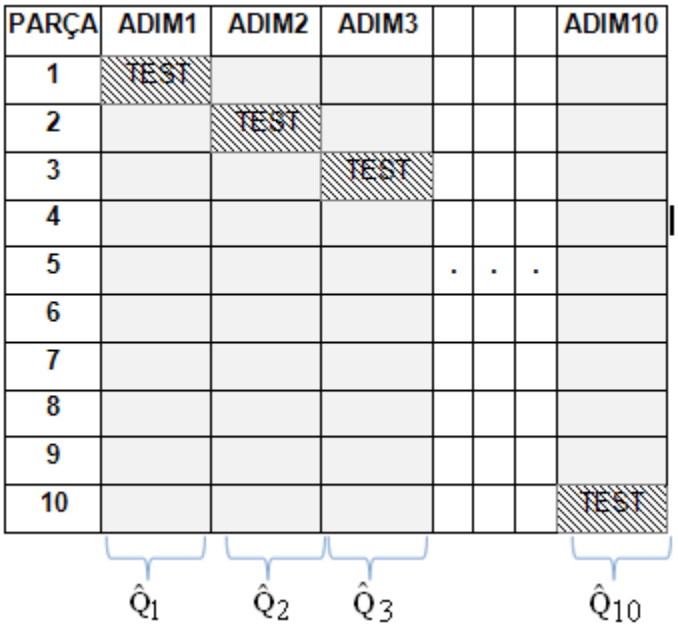
Rastgele alt örnekleme, holdout yönteminin k defa tekrar edildiği bir türevidir. Genel doğruluk kestirimi her tekrardan elde edilen doğrulukların ortalaması olarak alınır. Bu zorlukları ele almak ve mevcut veriden en fazla yararlanmak için sıklıkla k- parçalı çapraz geçerlik kullanılır (8,42).

2.3.2.2. Çapraz Geçerlik Yöntemleri

Çapraz geçerlik, veriyi iki kısma bölerek öğrenme algoritmalarını değerlendiren ve karşılaştıran istatistiksel bir yöntemdir. İlk parça, modeli öğrenmek ya da eğitmek için diğeri ise modeli değerlendirmek için kullanılır. Klasik çapraz geçerlikte eğitim ve değerlendirme setleri, her bir veri noktasının değerlendirilme şansına sahip olması için ardışık aşamalarda çapraz geçişli olmalıdır. Çapraz geçerliğin temel formu k-parçalı çapraz geçerliktir. Çapraz geçerliğin diğer türleri, k-parçalı çapraz geçerliğin özel durumlarıdır ya da k parçalı çapraz geçerliğin tekrarlanan aşamalarını içerir.

2.3.2.3. K-Parçalı Çapraz Geçerlik

K-parçalı çapraz geçerlikte veri ilk olarak k tane eşit büyüklükte kısma ya da parçaya (D_1, D_2, …, D_k) ayrılır. Herbir adım içinde verinin farklı bir parçası değerlendirme (validation) için tutulur ve kalan (k-1) parçası öğrenme için kullanılır. Ardışık k adımda eğitim ve değerlendirme gerçekleştirilir. Yani ilk adımda D_2, …, D_k altsetleri ilk modeli elde etmek için eğitim seti olarak kullanılır ve elde edilen model D_1 altseti üzerinden test edilir. İkinci adımda model D_1, D_3,…,D_k altsetleri üzerinde eğitilir ve D_2 üzerinde test edilir. Süreç bu şekilde devam eder. Holdout ve rastgele alt örnekleme yöntemlerinden farklı olarak burada her bir örneklem eğitim için aynı sayıda kullanılır ve test etme için bir kez kullanılır. Veri madenciliği ve makina öğrenmesinde 10 parçalı çapraz geçerlik (k=10) en yaygın kullanılanıdır. Bu yöntemin işleyişi Şekil 2.9'da gösterilmiştir.



Şekil 2.9. 10 parçalı çapraz geçerlik örneği.

Öğrenme algoritmalarını değerlendirme ya da karşılaştırma aşamasında her bir parça üzerindeki her bir öğrenme algoritmasının performansı, doğruluk gibi önceden belirlenmiş bir performans ölçüsü ile izlenebilir. Sınıflandırma için doğruluk kestirimi, k adımdaki doğru sınıflandırmaların başlangıç verisindeki örneklem sayısına bölünmesiyle elde edilir.

Kestirim için hata tahmini, k adımdan elde edilen toplam hatanın (loss) başlangıç örneklemlerinin toplam sayısına bölünmesi şeklinde hesaplanabilir. k(i), D'nin i'inci örneklemi içeren parçası olsun. Bu durumda kestirim hatası (HKO)'nın çapraz geçerlik kestirimi aşağıdaki gibi verilir:

$$\widehat{\mathrm{HKO}}_{\mathrm{CG}} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}(\mathrm{y_i} - \hat{\mathrm{y}}_{\mathrm{i}}^{-\mathrm{k(i)}})^2 \qquad (2.32)$$

$\hat{\boldsymbol{y}}_{\boldsymbol{i}}^{-\boldsymbol{k}(\boldsymbol{i})}$, verinin k(i)'inci parçası çıkarılarak tahmin edilen modelin i'in gözlem için kestirim değerini ifade eder (42).

2.3.2.4. Birini Dışarıda Bırakma Çapraz Geçerlik

Birini dışarıda bırakma çapraz geçerlik (BDBCG), K'nın verideki gözlem sayısına eşit olduğu (K=N) k-parçalı çapraz geçerliğin özel bir durumudur. Başka bir deyişle her bir adımda tek bir gözlem dışında hemen hemen tüm veriler eğitim için kullanılır ve model dışarıda bırakılan bu tek bir gözlem üzerinde test edilir.

i= 1,2,… N mevcut iken, i'inci örneğin her biri için:

- Eğitim setindeki i'inci gözlem dışarıda bırakılarak, model oluşturulur.
- $\hat{y}_i^{-i}$ olarak tanımlanan, i'inci gözlemin kestirim değeri hesaplanır.

Kestirim hatasının (HKO) BDBCG kestirimi aşağıdaki gibi elde edilir:

$$\widehat{\mathrm{MSE}}_{\mathrm{LOO}} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}(\mathrm{y_i} - \hat{\mathrm{y}}_{\mathrm{i}}^{-\mathrm{i}})^2 \qquad (2.33)$$

BDBCG kullanılarak elde edilen bir doğruluk kestirimi neredeyse yansızdır ancak yüksek varyansa sahiptir. Bu ise güvenilmeyen kestirimlere yol açar. Mevcut veriler seyrek olduğunda ve özellikle biyoinformatikte sadece düzine kadar örnek bulunduğunda kullanılır (30).

2.4. Model Birleştirme (Ensemble) Yöntemleri

İstatistik ve makina öğrenmesinde, model birleştirme (ensemble) yöntemleri herhangi bir model bileşeninden elde edilebilenden daha iyi kestirim performansı sağlamak için birden fazla modeli kullanır. Topluluk (Ensemble) yöntemleri, verilen bir istatistiksel öğrenmenin ya da model kurma tekniğinin kestirim performansını geliştirmeyi amaçlar. Ensemble yöntemlerin temel prensibi, yöntemi tek bir biçimde kullanmak yerine bazı model oluşturma yöntemlerinin doğrusal bir kombinasyonunu oluşturmaktır.

Bir topluluğun kendisi bir danışmanlı öğrenme algoritmasıdır; çünkü bu topluluk eğitilebilir ve daha sonra kestirim yapmak için kullanılabilir. Bu eğitilmiş topluluk bu nedenle tek bir hipotezi temsil eder; ancak bu hipotez mutlaka onun oluşturulduğu modellerin hipotez alanı içinde değildir. Böylece toplulukların temsil edebildikleri fonksiyonlar içinde daha fazla esnekliğe sahip oldukları gösterilebilir. Bu esneklik teoride onları, eğitim verilerini tek bir modelin yapacağından daha fazla (over-fit) aşırı uyum ile sonuçlandırmaya itecektir; ancak uygulamada bazı birleştirme (ensemble) yöntemleri (özellikle bagging) eğitim verilerinin aşırı uyumuyla ilgili sorunları azaltmak eğilimindedir.

Deneysel olarak birleştirme yöntemleri, modeller arasında önemli bir farklılık olduğu zaman daha iyi sonuçlar elde etmek eğilimindedir. Birçok birleştirme yöntemi bu nedenle bir araya getirecekleri modeller arasında farklılığı teşvik etmektedirler.

Bagging ve boosting yöntemleri sınıflandırıcı ya da kestirim doğruluğunu geliştirmek için kullanılan topluluk yöntemlerinin iki örneğidir. Topluluk yöntemleri, temel yöntemin kestiriciyi geliştirmek için göreli olarak popüler ve basit bir aracı olmuştur. Bu aracı oluşun farklı nedenleri vardır: Örneğin bagging yöntemi, en azından bazı yöntemler için bir varyans azaltma planı olarak ortaya çıkmıştır. Öte yandan boosting yöntemleri, temel yöntemin yanlılığını azaltır. Bagging eğitim algoritmalarının kestirim hatasını azaltmak için istenen yöntemdir (43).

2.4.1. Bagging (Bootstrap Aggregating)

Bagging, 1990'ların ortasında Leo Breiman tarafından önerilen sınıflandırma ve regresyon düzenlerindeki kestirimlerin doğruluğunu arttırmak için kullanılan bir yöntemdir (43). Bu yöntem ile bir sınıflandırma ya da regresyon yöntemi orijinal veri setinin çeşitli değişimlerine uygulanır ve bu sonuçlar tek bir sınıflandırıcı ya da regresyon modeli elde etmek için birleştirilir.

Breiman (4), yöntemi değişken seçimi ve doğrusal model oluşturan yöntemler ya da karar ağaçları gibi verilmiş bir temel prosedür için varyans azaltma tekniği olarak uyarlamıştır. Bagging yöntemi, ayrıca aşırı uyumu (overfitting) engeller.

Bagging algoritması aşağıdaki gibi tanımlanır:

Adım 1: $(X_1, Y_1),\ldots, (X_n, Y_n)$ eğitim (E) verisinden yerine koyarak n defa rastgele $(X_1^*, Y_1^*), \ldots, (X_n^*, Y_n^*)$ çekilerek bootstrap örneklemi oluşturulur.
Y yanıt değişkeni, ya bir sınıf etiketi (sınıflandırma için) ya da sayısal (regresyon için) değer olabilir. Yerine koyarak örnekleme yapıldığı için eğitim örneklemindeki bir örnek, alınan eğitim setlerinde birden fazla sayıda olabileceği gibi hiç bulunmaya dabilir.

Adım 2: Bootstrap kestiricisi $\hat{g}^*(.)$ hesaplanır:

$$\hat{\boldsymbol{g}}^*(.) = h_n\big((X_1^*, Y_1^*), \ldots, (X_n^*, Y_n^*)\big)(.). \tag{2.34}$$

Adım 3: Adım 1 ve adım2 “B” defa tekrarlanır, “B” genellikle 50 ya da 100 olarak seçilir, sonuç olarak $g^{*k}(.)\ (k = 1,2, \ldots B)$ elde edilir. Bagging kestiricisi $\hat{g}_{bag}(.) = B^{-1}\sum_{k=1}^{B} g^{*k}(.)$ biçimindedir.
Teorik olarak bagging kestiricisi:

$$\hat{g}_{bag}(.) = E^*[\hat{g}^*(.)] \tag{2.35}$$

olarak elde edilir (4, 29).

Bagging kestirici (X_i, Y_i) (i= 1,…n)’den oluşturulan tek bir kestiriciye göre daha yüksek doğruluğa sahiptir. Bu kestiricilerin regresyon için ortalaması alınır ya da sınıflandırma için modu alınarak birleştirilir. Sonuçta elde edilen kestiricinin daha küçük varyanslı (daha yüksek doğrulukta) olduğu Breiman(1996,s.129) ve Sutton(2005, s.320)’de gösterilmiştir (Formül 2.36):

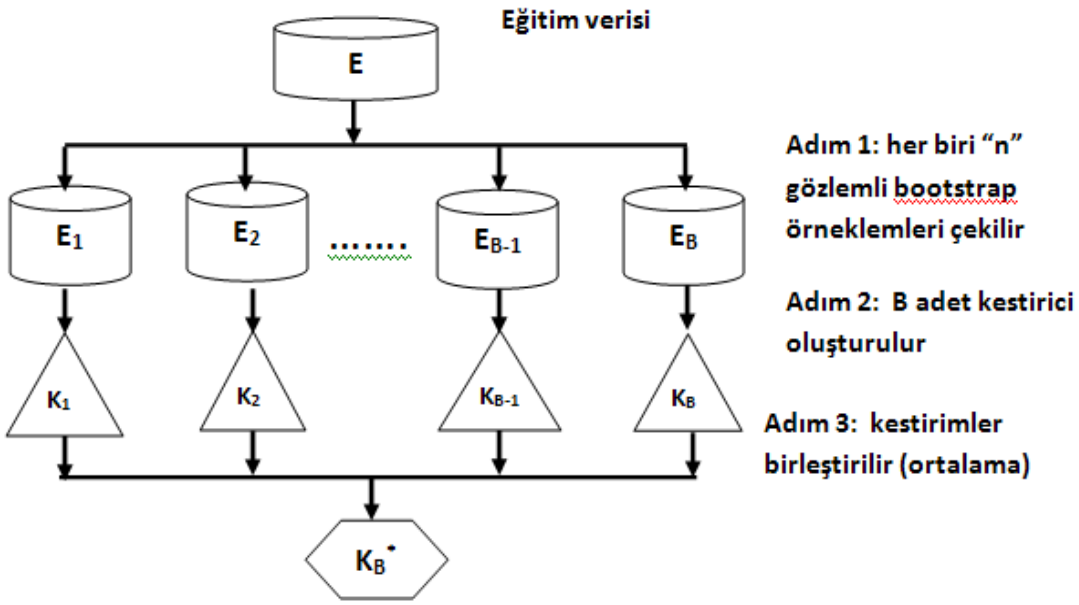
$$E([Y_x - \varphi(x)]^2) \geq E([Y_x - \mu_\varphi]^2) \tag{2.36}$$

"$\mu_{\varphi} = \mathrm{E}\,(\varphi(x))$" değeri bir kestirici olarak kullanılabilse idi $\varphi(x)$'den daha küçük kestirim hata kareler ortalamasına sahip olacaktı" (4).

Bagging Algoritması Şekil 2.10'da görüldüğü gibi herbir modelin eşit ağırlıklı kestirim verdiği öğrenme yöntemi için modellerin bir birleşimini oluşturur.

Eğer kullanılan kestirim yöntemi kararsız ise; yani eğitim setindeki ya da model oluşturmada kullanılan parametrelerdeki küçük bir değişiklik yanıt kestiriminde büyük değişikliklere sebep oluyorsa bagging kullanımında doğruluk artar.

Breiman, en iyi kestiricilerin regresyon ağaçları ve yapay sinir ağları olduğunu açıklamıştır (20). Y_x değeri, EKK regresyonu kullanılarak kestiriliyor ise, özellikle model biçiminin bilindiği, tüm değişkenlerin öğrenme örnekleminde yer aldığı hata terimi dağılımının yaklaşık olarak normal ve EKK ile ilgili varsayımların yaklaşık olarak sağlandığı ve örneklem genişliğinin aşırı küçük olmadığı durumlarda bagging'den kazanım küçük olacaktır. Çünkü kestirim yöntemi oldukça kararlıdır (4, 29).



Şekil 2.10. Bagging kestiricinin elde edilişi.

Buna karşın modelin gerçek biçiminin karmaşık ve bilinmiyor olması, çok sayıda açıklayıcı değişken olması ve bunların bazılarının yanıt değişkeni ile ilişkisiz olması, örneklem büyüklüğünün yeterince büyük olmaması durumlarında CART gibi kararsız bir regresyon yöntemi kullanılırsa bagging iyi sonuçlar verebilir. Bunun yanı sıra EKK, en yakın komşuluk regresyonu ve ridge regresyon gibi daha kararlı yöntemlerin bagging'den yarar sağlaması beklenmez (29).

Bagging, SRA'da yaygın olarak uygulanır iken, diğer pek çok sınıflandırma ve regresyon yöntemi ile birlikte de kullanılabilir. Bu genel bir yöntemdir, sınıflandırma ve regresyona özel bir yöntem değildir.

Tez çalışmasında bagging'in, incelediğimiz iki temel regresyon yöntemi olan RA ve DVR ile kullanımları incelenmiş ve gen ifade verileri üzerinde uygulaması yapılmıştır.

2.4.1.1. Bagging ve Destek Vektör Regresyon

DVR'nin Bagging ile birlikte kullanımı ve performansı üzerine literatürdeki çalışma sayısı oldukça kısıtlıdır. İlk olarak Breiman, 1996'daki çalışmasında bagging'in hangi yapıdaki modelleme yöntemleri için iyi performans sağlayacağını aşağıdaki gibi açıklamıştır:

> *"y'lerin nicel bir yanıt değişkeni olduğu, bir "E" eğitim veri seti {(y_n,x_n), n=1,2,…,N} gözlemlerini içersin. Bu eğitim seti kullanılarak x girdisi için y değişkeni φ(x,E) ile kestirilir. Yeni durumda her biri aynı temel "E" dağılımına sahip N bağımsız gözlemden oluşan {E_k} eğitim setlerinin bir dizisi verilmiş olsun. Burada amaç, tek bir eğitim seti kestiricisinden φ(x,E) daha iyi bir kestirim elde etmek için {E_k}'yı kullanmaktır. y sayısal bir değişken ise; φ(x,E), φ(x,E_k)'nın k adet ortalaması ile değiştirilmelidir; ancak bu yapılan işlemler tek bir bootstrap örneklemini oluşturur.*

E'den genellikle 20 ile 100 arasında B adet tekrarlı bootstrap örneklemleri çekilebilir. y, nicel yanıt değişkeni iken bagging kestiricisi aşağıdaki gibidir:

$$\varphi_B(x) = ort_B\, \varphi(x, E^{(B)}) \qquad (2.37)$$

Burada $\{E^{(B)}\}$, her biri N gözlemden oluşan rastgele fakat E'den yerine koyarak çekilen tekrarlı veri kümelerini oluştururlar. Herbir (y_n, x_n), herhangi bir $E^{(B)}$'de tekrarlı sayıda görünebilir ya da hiç görünmeyebilir"(4).

Bagging'in doğruluğu geliştirip geliştirmeyeceği konusundaki kritik faktör, φ'yi oluşturan temel yöntemin kararlılığıdır. E'deki değişiklikler örn. E'nin tekrarı gibi φ'de küçük değişiklikler yaratıyor ise bu durumda φ_B, φ'ye yakın olacaktır. Modeldeki iyileşme, E'deki küçük bir değişimin φ'de büyük değişimle sonuçlandığı kararsız yöntemler için ortaya çıkmaktadır. Kararsız yöntemlere örnekler; yapay sinir ağları, sınıflandırma ve regresyon ağaçları ve doğrusal regresyonda altset seçimidir.

Kararlı yöntemlere ise k- en yakın komşuluk yöntemi, ridge regresyon, DVR örnek verilebilir. Bagging genel olarak kararsız yöntemler için iyi çalışmaktadır. Breiman çeşitli regresyon verileri üzerinde özellikle regresyon ağaçları için HKO'daki azalmanın %21'den %46'ya yükseldiğini belirtmiştir. Drucker, Braga gibi bazı araştırmacılar çalışmalarında, bagging'in DVR performansında gelişme sağlamadığını (HKO artmaktadır) göstermişlerdir (4-5, 7, 44).

2.4.1.2. Bagging ve Regresyon Ağaçları

Bagging RA (RA-Bagging)'in altında yatan temel fikir, tek bir regresyon ağacındaki kestirim hatası kavramının eğitim veri setinin belirli bir seçimine bağlı olduğunun bilinmesidir. Bundan dolayı yerine koyarak örnekleme ile birçok benzer veri seti oluşturulursa ve budama olmadan regresyon ağaçları büyütülürse, kestirimlerin ortalaması alındığında kestirim hatasının varyans bileşeni azalır (4, 45). Bootstrap örneklemlerinin daha fazla sayıda kullanılması örnek ortalamalarının bağımsız gözlemlerin daha büyük bir sayısına bağlı olmasını sağlayarak varyansı düşürmektedir (29). Sonuç olarak her bir tekrar, veri setinin hafif bozulmuş bir uyarlamasıdır. Eğer farklı analizler birbirinden oldukça farklı sonuç veriyor ise ağaçlar kararsızlık sergilerler dolayısıyla ortalama almak sonuçları geliştirir.

Z= $\{(x_1,y_1),\ldots.(x_N,y_N)\}$ eğitim verisini içeren bir regresyon problemi dikkate alınsın. Bu veriden çekilen her bir $T_{B,K}$ b=1,….,B bootstrap örneklemi için model kurulur ve $Q(x,T_{B,K})$ kestirimi bulunur. Bu durumda bagging kestirimi aşağıdaki gibi tanımlanır (35):

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \widehat{\mathrm{Q}}(\mathrm{x}, \mathrm{T}_{\mathrm{B,K}}) \qquad (2.38)$$

Regresyonun özel bir durumu, $\widehat{\mathrm{Q}}(\mathrm{x}, \mathrm{T}_{\mathrm{B,K}})$'in x girdi vektöründeki ağacın kestirimini ifade ettiği bir regresyon ağacıdır. Her bir bootstrap ağacı genellikle orjinalinden farklı özellikler içerecektir ve farklı sayıda son düğüme sahip olabilir.

Bagging kestirimi bu B tane ağaçtaki x'lerin ortalama tahminidir. Çekilen bir bootstrap eğitim örnekleminde eğitim verilerinin ortalama olarak %37'si örneğin dışında kalır. Bir tekrarda örneğe çekilmiş veri "in bag veri" olarak bilinirken, çekilmeyen kısım ise "out of bag" veri olarak bilinir (34).

"Out-of-bag" veri herhangi bir ağacı oluşturmak ya da budamak için kullanılmaz fakat bagging kestiricilerin genelleştirme hatası ve düğüm hatasının daha iyi kestirimlerini sağlarlar (Breiman 1996b). Kabaca, $Q(x,T_B)$ kestiricisi için dışarıda kalan bu %37'lik örnekler kullanılmayan test örnekleridir. Böylece 100 tekrar için (K=100), eğitim setindeki her bir (y,x) örneği $T_{B,K}$'nin (y,x)'i içermediği $Q(x,T_{B,K})$'lar arasında yaklaşık 37 kestirime sahiptir. Regresyon ağaçlarına uygulandığında bir düğüm kestiriminde beklenen hatayı tahmin etmede gelişmiş bir yöntem elde edilir. Regresyonda gerçek eğitim seti yanıt değerlerini kullanmak yerine out-of-bag kestirimlerini kullanmak daha doğru ağaçlar verir.

RA-bagging'in temel dezavantajı, ortalama olarak 30-80 ağaç gerektirmesi ve bundan dolayı çok sayıda bireysel ağacın yorumlanmasının neredeyse imkansız hale gelmesidir. Birden çok ağaç arasındaki değişkenliğin az olduğu durumlarda bölünmeler birbirine benzerdir ve yorumlama için tek bir RA kullanılabilir. Aksine çok sayıda ağaç büyük değişiklik gösteriyor ise tek bir regresyon ağacı analizi modellenen ilişkinin olası yorumlamalarından sadece biri olabilir ve yorumlama belirsizliği yüksektir (4, 29).

2.4.2. Boosting

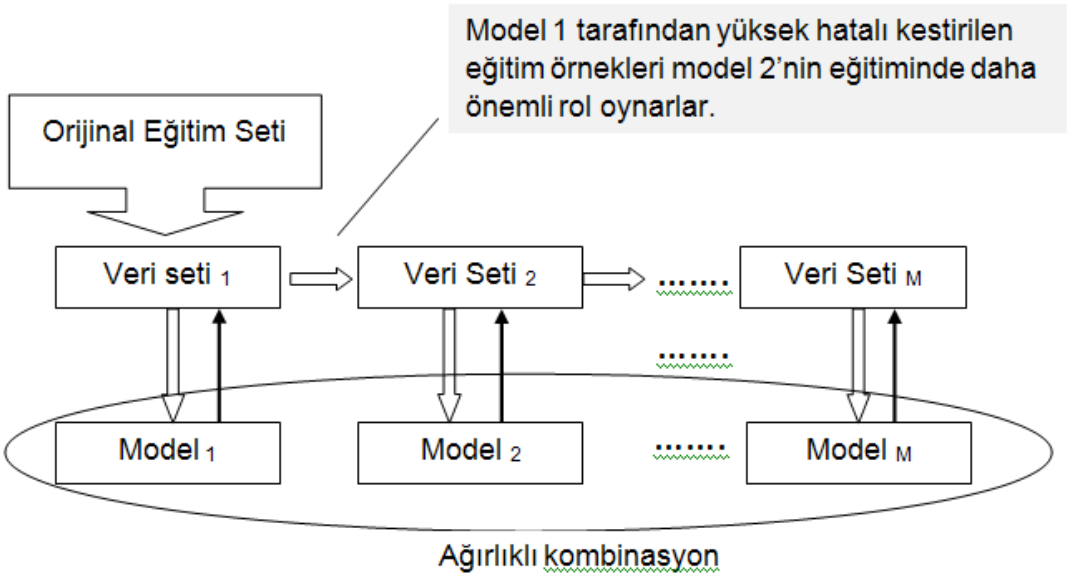
Boosting ilk olarak Freund ve Schapire tarafından 1997'de tanıtılmıştır. Hem bagging hem de boosting çok sayıda açıklayıcı değişken kullanarak daha düşük kestirim hatası ve daha düşük hata oranı elde edilebilen yöntemlerdir. Bagging'de olduğu gibi boosting yönteminde de topluluğu (ensemble) oluşturan sınıflandırıcılar/kestiriciler, verilerin yeniden örneklenmesi ile elde edilmekte ve daha sonra çoğunluk oyları ile birleştirilmektedir (46).

Sınıflamada bagging ve boosting'in çok sayıda çalışması boosting'in bagging'den daha üstün olduğunu göstermiştir (28).

Hem boosting hem de bagging'de regresyon kestiricisi (makina), eğitim setinin farklı alt setleri üzerinde eğitilir. Bagging'de her bir makina eğitim setinin orijinal N örneğinden yerine koyarak çekilen N örneklem üzerinde eğitilir. Eğitim setinden alınan her örneğin bootstrap örnekleminde görülme şansı eşittir. Herbir makina orijinal eğitim setinin farklı altsetleri üzerinde eğitilir ve bu nedenle farklı kestirimler verecektir.

Fakat Boosting sıralı ve aşamalı bir yöntemdir. Bagging'de olduğu gibi ilk makina orijinal eğitim setinden yerine koyarak çekilen N boyutlu örneklemler üzerinde eğitilir. Daha sonra eğitim örneklerinin hepsi bu makina aracılığı ile işlemden geçirilerek hangilerinin en fazla hataya sahip olduğu not edilir. Regresyon makinaları için kestirim değerleri gözlenen değerlerinden en fazla farklı olan örnekler hata açısından "fazla" olarak tanımlanır. Hatası en fazla olan bu örneklerin, ikinci makina için eğitim setinin üyeleri olarak örneklenme olasılıkları daha yüksek olacak şekilde ayarlanır. Bu nedenle makinaları oluşturmada ilerlerken zor örneklerin eğitim setinde görülme olasılığı daha yüksektir. M'inci adımdaki gözlem ağırlıkları önceki M-1 adımdaki sonuçlara bağlıdır. Böylece farklı makinalar, gözlem uzayının farklı bölgelerinde daha iyidir. Kestirimler ağırlıklı ortanca kullanılarak birleştirilir yani kestirimleri ile ilgili daha güvenilir olan kestiriciler daha büyük olarak ağırlıklandırılır (Şekil 2.11).

Boosting algoritmaları, uyum eksikliğini nasıl ölçtüklerine ve sonraki adımlar için gözlem ağırlıklarını nasıl seçtiklerine göre değişirler. Adaboost gibi orijinal Boosting algoritmaları Freund ve Schapire tarafından 1996'da iki sınıflı sınıflandırma problemleri için geliştirilmişlerdir (28). Her ne kadar sınıflandırma problemlerini geliştirmek için odaklanılmış olsa da, Freund ve Schapire 1997'de, çalışmalarının son bölümünde regresyon problemleri için adaboost algoritmasının uygulanması için düşüncelerini özetlemişlerdir (28). Ancak boosting regresyonu ilk öneren Drucker (1997) olmuştur (28,46).



Şekil 2.11. Boosting süreci.

2.4.2.1. Boosting ve Regresyon Ağaçları

Freund ve Schapire 1996'da Adaboost algoritmasını ilk olarak sınıflandırma doğruluğunu geliştirme amacıyla önermişlerdir (47).

Boosting regresyonu ve pratik yöntemleri ise ilk öneren 1997'de Drucker olmuştur. Drucker'ın Ad Hoc yöntemi zayıf kestirilen gözlemleri yüksek ağırlıklandırarak ve iyi kestirilen gözlemleri ise düşük ağırlıklandırmayı izleyen ağaç regresyonunu tekrarlı olarak gerçekleştiren Adaboost algoritmasına benzer tarzdadır. Daha sonra Breiman, Drucker'ın ad-hoc yöntemi ile boosting regresyon problemlerinin nasıl ele alınabileceğini göstermiştir. Regresyon problemlerinde Boosting ile ilgili çalışmalarda ilk kez Breiman (1999) regresyondaki kayıp (loss) fonksiyonun optimizasyonunu boosting algoritmasının bir parçası olarak kullanımını ele almıştır (47).

Genelleştirilmiş boosting yöntemleri, Adaboost algoritmasını sınıflandırmadan ziyade regresyonda kullanmak için adapte edilmişlerdir (35, 48). Hastie ve diğerlerinin 2001'de, boosting ve optimizasyon arasındaki bağlantıyı kurarak açıklığa kavuşturdukları bu yeni çalışma gradyan boosting makinasını önermektedir (47).

Boosting regresyon ağaçları (BRA), birçok model kurarak ve kestirim için bunları birleştirerek tek bir modelin performansını geliştirmeyi amaçlayan çeşitli yöntemlerden biridir. BRA iki algoritma kullanır: ilki regresyon ağaçları; sınıflandırma ve regresyon ağaçları model grubundandır ve diğeri "boosting" modellerin bir koleksiyonunu oluşturur ve birleştirir (6).

Boosting algoritmalarının regresyon problemleri için uzanımları iki tiptedir. İlk yaklaşım bir sınıflandırma probleminde orijinal veri setinin uygun bir dönüşümünü ve daha sonra bu yeni veri seti üzerinde bir sınıflandırıcının kullanımını içerir. Örneğin Freund ve Schapire (1997) tarafından, Adaboost.R algoritması; Ridgeway (47) tarafından Boosted Naive Bayes regresyon algoritması ele alınmıştır (49). Drucker (46) tarafından önerilen daha doğrudan bir strateji, regresyon için Adaboost'un değişimini dikkate alır.

Bu algoritma, eğitim örneklemindeki tüm gözlemler için $w_i^0 = 1,\ i = 1,2,\dots,N$ aynı ağırlıkları düzenleyerek başlar. Daha sonra t=1'den T'ye kadar aşağıdaki adımlar gerçekleştirilir (48).

1- İ'nci gözlemin içerilme olasılığı $p_i^t = \frac{w_i^t}{\sum_{j=1}^{N} W_j^t}$ 'ye sabitlenerek eğitim örnekleminden yerine koyarak N gözlem çekilir.

2- Kestirim fonksiyonu $h_t(x)$, çekilen örnekleme uygulanır.

3- Eğitim örnekleminin tüm örnekleri üzerinde nispi mutlak artıklar ve belirli bir kayıp fonksiyonun $L_i^t = L(r_i^t)$ artıklara ilişkin değerleri hesaplanır:

$$r_i^t = \frac{|y_i - h_t(x_i)|}{en\ buyuk\ |y_i - h_t(x_i)|} \qquad \text{i=1, 2, …, N} \qquad (2.39)$$

Drucker üç tane aday fonksiyon önermiştir: doğrusal; $L(r_i^t) = r_i^t$, karesel; $L(r_i^t) = (r_i^t)^2$, ve üstel; $L(r_i^t) = 1 - exp\,(r_i^t)$. Sonuç olarak ortalama kayıp hesaplanır: $L^{-t} = \sum_{i=1}^{N} p_i^t L_i^t$.

4- Kestirim fonksiyonu $h_t(x)$ için katsayı hesaplanır; $\beta_t = \frac{L^{-t}}{1 - L^{-t}}$.

5- Ağırlıklar güncellenir: $w_i^{t+1} = w_i^t \beta_t^{1-L_i^t}$.

T adet tahmin fonksiyonu ile elde edilen sonuçları birleştirmek için Drucker, ağırlıkların $log\left(\frac{1}{\beta_t}\right)$'ye orantılı olduğu ağırlıklı medyanın kullanımını önermiştir.

Algoritmanın yakınsaması için gerekli olan koşul $L^{-t} < 0.5$ (t=1,2,…,T) koşuludur, ancak gerçek veriler üzerinde böyle bir koşula uymak zordur. Genel olarak bu yöntemin etkinliğini kanıtlayan sonuçlar yoktur ancak kestirici olarak regresyon ağaçlarını kullanan deneysel analiz, kestirim doğruluğunda bir iyileşme göstermiştir (49).

İkinci bir yaklaşım, bir kayıp fonksiyonu üzerinde türevsel azalış algoritması olarak dikkate alınan Adaboost algoritmasına dayalıdır ve bu bakış açısından regresyon problemleri için yeni boosting algoritmaları önerilmiştir (48, 50).

Bu ikinci yaklaşıma göre regresyon problemleri için boosting, "Functional Gradient Descent" fonksiyonel türev azalmasının bir biçimidir. Bir kayıp (loss) fonksiyon göz önünde bulundurulur ise (örneğin sapma gibi) bu sapma modele bağlı olarak kestirim performansındaki kaybı temsil eder. Boosting her bir adımda bu kayıp fonksiyonu en iyi azaltan (türevin aşağı adımları ile) yeni bir ağacı ekleyerek kayıp fonksiyonu en küçük yapan bir sayısal optimizasyon tekniğidir. BRA için ilk regresyon ağacı seçilen bir ağaç boyutu için kayıp fonksiyonu (loss) en fazla azaltan ağaçtır. Takip eden her adımda ilgilenilen temel kriter artıklardır yani o adıma kadar model tarafından açıklanmamış yanıttaki değişim miktarıdır (28).

Örneğin ikinci adımdaki ağaç ilk ağacın artıkları için modellenerek oluşturulur ve bu ikinci ağaç ilk ağaçla karşılaştırıldığında oldukça farklı değişkenler ve bölünme noktaları içerebilir. Bu model daha sonra iki ağaç (iki terim) içerecek şekilde güncellenir ve bu iki terimli modeldeki artıklar hesaplanır ve süreç bu şekilde devam eder. Bu süreç "stagewise"dır ("stepwise" yani aşamalı değil). Bu da model genişletilirken varolan ağaçların değişmeden kaldığı anlamına gelir. Her bir gözlem için kestirim değeri, yeni eklenen ağacın katkısını yansıtmak için her aşamada yeniden kestirilmektedir.

Son BRA modeli her bir terimin bir ağaç olduğu bir regresyon modeli olarak düşünülebilen birçok ağacın (genellikle yüz ya da binlerce) doğrusal bir kombinasyonudur. Model oluşturma süreci, eğer türevi yavaş yavaş azaltırsa en iyi performansı gösterir, dolayısıyla her bir ağacın katkısı genellikle 1'den önemli ölçüde küçük olan bir "öğrenme oranı" (shrinkage) tarafından küçültülür. Final modeldeki kestirim değerleri, öğrenme oranı ile çarpılan tüm ağaçların toplamı olarak hesaplanır ve tekbir karar ağacı modelinden daha kararlı ve doğru kestirimler verir (28, 48).

Bu çalışmada Boosting'in regresyon ağaçlarındaki kullanımı olarak Friedman'ın stokastik türevsel (gradient) boosting makinası yaklaşımı incelenmiştir (50). Bu yaklaşıma ilişkin algoritma ve çözüm için kullanılan R programı gbm() yazılımındaki temel parametreler ilerideki bölümde açıklanmıştır.

BRA'nın bazı önemli özellikleri aşağıdaki gibidir:

- ✓ Bu süreç stokastiktir, rastgele ya da olasılıksal bir bileşen içerir. Bu stokastiklik her bir yeni ağacı kurmak için verinin sadece rastgele bir alt setini kullanarak, son modelin varyansını azaltır ve kestirim performansını arttırır (50).
- ✓ Ardışık model oluşturma süreci, daha önce oluşturulmuş ağaçlar üzerinde model kurar ve giderek kestirimi daha zor olan gözlemlere odaklanır. Bu ise süreci veri seti için tek bir büyük ağacın modellendiği süreçten ayırır.
- ✓ Değerler iki önemli parametre için sağlanmalıdır: öğrenme oranı (*lr*; aynı zamanda *küçültme parametresi "*ϑ" olarak bilinir), her bir ağacın büyüyen modele katkısını belirler. *Ağaç karmaşıklığı (tree complexity- tc)*, etkileşimlerin modellenip modellenmeyeceğini kontrol eder. 1'li tc (tekil karar ağacı; iki son düğümlü) bir katkılı model oluşturur, 2'li tc, iki yönlü etkileşimlere kadar modeli oluşturur ve bu şekilde devam eder. Bu iki parametre daha sonra en uygun kestirim için gerekli olan *ağaç sayısını (nt)* belirler (28).

BRA'nın avantajları,

- ✓ Boosting algoritmaları aşırı uyuma (overfitting) karşı dayanıklılık gösterirler.
- ✓ Bu yaklaşım hatalı ölçülmüş çok sayıda açıklayıcı değişkeni kolaylıkla ele alabilir.
- ✓ BRA, arzu edilen herhangi bir dereceden etkileşimi sağlayarak regresyon ilişkisi için esnek bir model sağlamaktadır.
- ✓ BRA ile model kurma sürecinin her bir adımında uygulaması kolay ve sayısal olarak kararlı (stable) bir algoritma veren basit hesaplamalar gerçekleştirilir (37, 47).

Gradyan Boosting Makinası

Friedman (2001), regresyon ağaçlarını kullanarak türevlenebilir bir kayıp fonksiyonun türevsel azalışına (gradient descent) dayalı fonksiyon tahmini için genel bir yaklaşım tanımlamıştır.

Gradyan boosting (gbm), her bir adımda bu kayıp fonksiyonu EKK yoluyla bulunan geçerli "pseudo" artıklarına ardışık olarak kurarak eklemeli (additive) regresyon modellerini oluşturur. Bu yalancı (pseudo) artıklar, her bir adımda değerlendirilen her bir veri noktasındaki model değerlerine göre en küçük yapılan fonksiyonel kaybın (loss) türevidirler.

Hem kestirim doğruluğunun hem de işlem hızının yönteme randomizasyon dahil edilerek önemli ölçüde geliştirilebildiği gösterilmiştir. Özellikle her adımda eğitim verilerinin bir alt örneği tüm eğitim veri setinden rastgele seçilir (50).

Bu rastgele seçilen alt örneklem daha sonra temel öğreniciyi modellemek ve geçerli adımda model güncelleştirmesini hesaplamak için kullanılmaktadır. Bu rastgele yaklaşım aynı zamanda temel öğrenicinin aşırı uyum sorununa karşı sağlamlığı arttırır (50). Friedman'ın gradyan boosting makinası algoritması aşağıda açıklanmıştır.

Gradyan Boosting Algoritması

Herhangi bir fonksiyon tahmin probleminde, bazı kayıp fonksiyonların L(y,f) beklenen değerini en küçük yapan $\hat{f}(x)$ fonksiyonu bulmak istenir (48).

$$\hat{f}(x)=\arg \min_{f(x)} E_{y,x} L(y, f(x)) \qquad (2.40)$$

Gradyan Boosting Algoritması Adımları

$\hat{f}(x)$ bir sabit olsun, $\hat{f}(x)=\arg min_{\rho} \sum_{i=1}^{N} L(y_i, \rho)$
t=1'den T'ye kadar aşağıdaki adımlar gerçekleştirilir:

1- Çalışmanın sonuç değeri olarak negatif türev hesaplanır:

$$z_i = -\frac{\partial}{\partial f(x_i)} L\big(y_i, f(x_i)\big)|_{f(x_i)=\hat{f}(x_i)} \qquad \text{i= 1, 2, …N} \qquad (2.41)$$

2- Z_i'nin x_i ortak değişkenlerine göre EKK modeli g(x) oluşturulur.

3- Türev azalış adım büyüklüğü aşağıdaki gibi hesaplanır:

$$\rho = argmin_{\rho} \sum_{i=1}^{N} L(y_i, \hat{f}(x_i) + \rho g(x_i)) \qquad (2.42)$$

4- $\hat{f}(x)$ tahmini aşağıdaki gibi güncellenir:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \vartheta \rho g(x) \qquad (2.43)$$

Bu algoritmanın ilgi çekici özelliği, adım 1'de hesaplanan genelleştirilmiş artıkların geçerli model tahminindeki, $\hat{f}(x)$, en dik azalış yönünü temsil etmesidir.

Bu türev, $\hat{f}(x)$'e her bir i'ye ayrı bir parametre gibi davranılarak hesaplanmaktadır. Bu artıklar için EKK regresyonunu kullanarak bir regresyon ağacı oluşturan kişi en dik iniş yönünde en yüksek ilişkili regresyon ağacını oluşturur. Böylece, bunun gibi bir ağacı yeniden $\hat{f}(x)$'e eklemek model kestirimini yaklaşık olarak en dik azalış yönüne kaydırır (37).

ρ, en büyük azalışın yönü boyunca adım büyüklüğüdür. ϑ parametresi 0 ve 1 arasında değişen bazı sabit değerlerdir ve her bir adımda uygulanan küçülmeyi düzenler. Friedman (2001), ϑ'nin daha küçük değerlerinin boosting kestirimini genellikle daha büyük ölçüde geliştirdiğini göstermiştir (48).

2.5. Regresyonda Model Değerlendirme Ölçüleri

Farklı regresyon modelleri oluşturulduktan sonra bu modellerin karşılaştırılabileceği çok sayıda kriter vardır. Bunlardan bazıları hata kareler ortalamasının karekökü, ortalama mutlak sapma ve belirtme katsayısı (R^2) gibi ölçülerdir.

Tez çalışmasında kullanılan model değerlendirme ölçüleri aşağıda açıklanmıştır.

1- Hata Kareler Ortalamasının Karekökü (KHKO)

KHKO, bir regresyon modelinin hata oranını ölçmek için popüler bir formüldür; ancak hataları aynı birimle ölçülen modeller arasında karşılaştırma yapılabilir.

$$\text{KHKO} = \sqrt{\frac{\sum_{i=1}^{n}(\widehat{y_i}-y_i)^2}{n}} \qquad (2.44)$$

2- Ortalama Mutlak Sapma (OMS)

Ortalama mutlak sapma, orijinal veri ile aynı birime sahiptir ve sadece hataları aynı birim üzerinden ölçülen modeller arasında karşılaştırılabilir. Büyüklük olarak KHKO'na benzer ancak biraz daha küçüktür.

$$\text{OMS} = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n} \tag{2.45}$$

3- Belirtme (Açıklayıcılık) Katsayısı (R^2)

Belirtme katsayısı (R^2), regresyon modelinin açıklayıcı gücünü özetler ve kareler toplamlarına göre hesaplanır.

$$\text{Belirtme katsayısı} \longrightarrow R^2 = \frac{\text{RKT}}{\text{TKT}} = 1 - \frac{\text{HKT}}{\text{TKT}} \tag{2.46}$$

$$\text{Toplam Kareler Toplamı} \longrightarrow \text{TKT} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$\text{Regresyon kareler Toplamı} \longrightarrow \text{RKT} = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2$$

$$\text{Hata Kareler Toplamı} \longrightarrow \text{HKT} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

R^2, bağımlı değişkenin regresyon modeli tarafından açıklanan varyans oranını ifade etmektedir. Hata kareler toplamı (HKT) küçüldükçe, modelden daha güvenilir kestirimler elde edilir.

Eğer regresyon modeli mükemmel ise HKT sıfır ve R^2 ise 1'e eşit olur. Regresyon modeli tamamen başarısız ise, HKT, TKT'na eşittir. Bu durumda regresyon ile hiçbir varyans açıklanamaz ve R^2 =0 olur (42).

3. GEREÇ VE YÖNTEM

3.1. Benzetim Çalışması

Çeşitli gen seti düzenleri için tez çalışmasında incelenen temel regresyon yöntemlerinin performanslarını değerlendirmek için kapsamlı bir benzetim çalışması gerçekleştirilmiştir. Benzetimi yapılan ilk senaryo, bağımlı değişkenle olan ilişkiler için regresyon katsayılarına, gen setinde bulunan genler arasındaki korelasyona, model hatasının standart sapmasına ve örneklem genişliğine göre değişmektedir. Her bir benzetim senaryosunda, karekök hata kareler ortalaması (KHKO), belirtme katsayısı (R^2) ve ortalama mutlak sapma (OMS) gibi ölçüleri elde etmek için 1000 veri seti türetilmiştir. Tez çalışmasında ilk senaryoya ek olarak, gen dizisi ve doz değerleri arasındaki orijinal korelasyon yapısına bağlı olarak da benzetim veri setleri oluşturulmuştur. Bu ek senaryonun temel amacı, büyük kısmı negatif değerlerden oluşan gen ifade dizisi ile doz değerleri arası orijinal korelasyon matrisini (yani gerçek ilişkiyi) temsil edebilmektir.

Farklı parametrelere göre elde edilen toplam 83 senaryonun model performansları ekte tablolar halinde özetlenmiştir (Bkz. EK 5-16). Benzetim senaryosu I ve II'de ortak olan sonuçlara grafiksel olarak bulgular bölümünde değinilmektedir.

3.1.1. Verinin Modele Bağlı Türetilmesi ile Elde Edilen Senaryo

N, bir gen veri setindeki gözlem sayısını ve p ise genlerin sayısını ifade etsin. Her bir gözlem için gen ifade verisi $X_{i(1\times p)}$ ~ ÇDND(0,Σ) i=1,2,…,n çok değişkenli normal dağılımdan türetilmiştir.

“Σ” Kovaryans matrisi, kare bir matris olduğu için ifade değerleri arasında korelasyon olmadığı durum için birim matris olarak ayarlanmış, ya da reaksiyondaki tüm genlerin belirli bir miktarda ilişkili olduğu durum için, örneğin ($\rho_{xi,xj}$=0.1 ya da $\rho_{xi,xj}$=0.3 ve varyans 1’e ayarlanmıştır) olacak şekilde düzenlenmiştir. Bağımlı değişken yani kantitatif bir fenotip (Y_i), her bir gözlem için gen ifade değerlerine bağlı olarak (3.1) Eşitliği ile türetilmiştir.

$$y_i = \beta_j X_i + \varepsilon_i \,, \qquad (3.1)$$
$$\varepsilon_i \sim N(0,\sigma)$$

Regresyon katsayılarının “β_j” belirlenmesi, benzetim senaryosuna göre değişiklik göstermektedir; β=1, β=2, β=3 sırasıyla küçük, orta ve büyük katsayı etkisini ifade etmektedir (51).

Değişen örneklem genişlikleri n=20, 100 ve 500 olarak, hatanın standart sapması bağımlı değişkendeki değişime orantılı olarak β=1 için σ=1, 6, ve 20 değerlerinde, β=3 için ise küçük ve büyük değişkenliği belirtecek şekilde σ=1 ve 50 şeklinde ayarlanmıştır. Ayrıca her bir senaryo için genlerin bağımlı değişkenle arasındaki ilişkiyi belirleyen regresyon katsayılarına (β=1, β=2, β=3) göre de değişimler dikkate alınmıştır.

$\overline{\boldsymbol{X}} = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix}$, gen ifade değerlerinin ortalamasını temsil edecek şekilde ve üç farklı korelasyon düzeyinde veri setleri üretilmiştir.

Farklı genlerin ifade düzeyleri arasında sırasıyla yüksek ilişkili (ρ=0.3), orta ilişkili (ρ=0.1) ve ilişkisiz (ρ=0) durumlar varsayılarak kullanılan korelasyon matrisleri aşağıdaki gibidir (51):

$$\mathbf{M}1=\begin{bmatrix} 1 & 0.3 & \dots & \dots & 0.3 \\ 0.3 & 1 & \dots & \dots & \dots \\ 0.3 & \dots & 1 & \dots & \dots \\ \dots & \dots & \dots & 1 & 0.3 \\ 0.3 & 0.3 & \dots & 0.3 & 1 \end{bmatrix} \qquad \mathbf{M}2=\begin{bmatrix} 1 & 0.1 & \dots & \dots & 0.1 \\ 0.1 & 1 & \dots & \dots & \dots \\ 0.1 & \dots & 1 & \dots & \dots \\ \dots & \dots & \dots & 1 & 0.1 \\ 0.1 & 0.1 & \dots & 0.1 & 1 \end{bmatrix}$$

$$\mathbf{M}3=\begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & \dots & \dots & \dots \\ 0 & \dots & 1 & \dots & \dots \\ \dots & \dots & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}$$

3.1.2. Genler Arası Orijinal Korelasyon Yapısına Dayalı Senaryo

Önceki bölümde açıklanan ilk benzetim senaryosunda gen verileri genel olarak sıfıra çok yakın ortalamaya sahip olması ve yaygınlığının da büyük olmamasından dolayı literatürdeki kullanımına da uyumlu olarak çok değişkenli standart normal dağılımdan türetilmiştir (51). Bağımlı değişken değerleri ise gen ölçümlerine bağlı doğrusal bir model varsayılarak oluşturulmuştur; $y = \beta X_i$. Regresyon hatası "ε" ise $\varepsilon \sim N(0,\sigma)$ sıfır ortalama ve σ standart sapma ile normal dağılımdan türetilmiştir. Buna karşın ikinci senaryoda gen verileri (açıklayıcı değişkenler) ile bağımlı değişken arasında belli bir model varsayımı yapılmamıştır. Yukarıda açıklandığı gibi gen verileri ile doz değerlerinin korelasyon matrisinde negatif değerler de yer almaktadır.

gds1429 ID numaralı gen verisi için doz değerleri ve gen ifade değerleri arasında elde edilen orijinal korelasyon matrisinin bir bölümü aşağıdaki gibidir (Tablo 3.1).

Tablo 3.1. Gen ifade ve doz değerleri arasındaki orijinal korelasyon matrisi (ID: gds1429).

	Doz	Gen1 AI41246	Gen2 AW5203	Gen3 Btaf1	Gen4 BE0956	Gen5 Gzf1	Gen6 AI40799	Gen7 Lmcd1	….
Doz	**1.000**	-0.009	0.058	0.015	-0.015	0.045	0.118	0.280	0.017
Gen1 AI41246	-0.009	**1.000**	0.029	-0.144	0.197	0.433	-0.399	0.152	-0.202
Gen2 AW5203	0.058	0.029	**1.000**	-0.064	0.002	0.239	-0.087	-0.114	-0.098
Gen3 Btaf1	0.015	-0.144	-0.064	**1.000**	0.074	-0.133	0.152	0.192	-0.050
Gen4 BE0956	-0.015	0.197	0.002	0.074	**1.000**	0.173	-0.202	0.132	0.224
Gen5 Gzf1	0.045	0.433	0.239	-0.133	0.173	**1.000**	-0.384	0.151	-0.142
Gen6 AI40799	0.118	-0.399	-0.087	0.152	-0.202	-0.384	**1.000**	0.083	0.266
Gen7 Lmcd1	0.280	0.152	-0.114	0.192	0.132	0.151	0.083	**1.000**	-0.109
. .	0.017	-0.202	-0.098	-0.050	0.224	-0.142	0.266	-0.109	**1.000**

Genler arasındaki gerçek ilişki yapısını çalışmada yansıtabilmek için Burton ve ark'nın aşağıda referans verilen açıklaması temel alınmıştır:

> *"Açıklayıcı değişkenler tamamıyle bağımsız olmadıkça, ki bu pratikte imkansızdır, çok değişkenli veriyi türetmek ek olarak açıklayıcı değişkenler arasındaki korelasyonların belirlenmesini gerektirir. Eğer çalışma gerçek verilere dayalı ve özellikle çok sayıda ortak değişken içeriyor ise ortalama ve ilgili kovaryans matrisinin belirlenmesi daha doğrudan bir yoldur, bu şekilde üretilen veri gerçeği yansıtacaktır. Aksine ortak değişkenler arasındaki korelasyon seçimi keyfi olabilir ancak bu durum geçerli ilişkileri belirlemek açısından sorunludur.*

> *Belirli bir ortalama ve korelasyon yapısına sahip çok değişkenli ortak değişken verisini oluşturmak için en basit yaklaşım bir çok değişkenli normal dağılım varsayımı yapmaktır. Gerçek veri içinde herhangi bir sürekli ve normal olmayan dağılımlı değişken var ise, normallik varsayımını daha uygun hale getirmek için bu değişkenler dönüştürülmelidir"* (52).

Bu bilgilere dayalı olarak gen ölçümleri ve doz değerleri arasından elde edilen korelasyon matrisi kullanılarak benzetim verisi çok değişkenli normal dağılımdan türetilmiştir.

Veriler R programı MASS yazılımı içinde yer alan mvrnorm() fonksiyonu yardımıyla sıfır ortalama vektörü $\bar{X}$ ve Tablo 3.1'deki korelasyon matrisi dikkate alınarak;

```
>setwd('C:\\')
>kor <- matrix(scan('mydata101.txt'),101,101)
>veri= mvrnorm(n=500, rep(0, 101), kor)
```

komutları ile elde edilmiştir.

Genel olarak gen ifade verilerinden elde edilen korelasyon matrisi simetrik ve pozitif tanımlı olmalıdır; ancak bazı durumlarda simetrik görülen bir matris R programındaki depolama durumlarından dolayı simetrik okunmayabilir. Dolayısıyla simetriklik ve pozitif tanımlılık gerçekleşmediği durumlarda gen korelasyon matrisleri üzerinde aşağıdaki işlemler uygulanmıştır:

a- Bir matrisi simetrik hale getirmek için:

Aşağıda verilen komutla matrisin kendisi ile transpozunun ortalaması alınmıştır:

```
> setwd('C:\\')
> sigma2 <- matrix(scan('mydata1.txt'),101,101)
> sigma.new=(sigma2+t(sigma2))/2
```

b- Matrisin pozitif tanımlı olup olmadığına bakılmıştır:

R programı bazen veri türetme satırından sonra “matris pozitif tanımlı değil” şeklinde uyarı vermektedir:

```
a = mvrnorm(n=50, rep(0, 9), sigma)
Hata oluştu: mvrnorm(n = 50, rep(0, 9), sigma) :
'Sigma' is not positive definite
 a=make.positive.definite(sigma)
```

Bu durum matrisin tüm özdeğer elemanlarının pozitif olup olmadığı ile ilgilidir. eigen(sigma)$values değerleri içinde negatif olanlar var ise matris, çeşitli işlemlerle pozitif tanımlı hale getirilebilir. Pozitif tanımlılık R'daki corpcor paketinin is.positive.definite (sigma.new) fonksiyonu ile de anlaşılabilir. Eğer matris pozitif tanımlı değil ise make.positive.definite (sigma.new) fonksiyonu ile en yakın pozitif tanımlı matris bulunur (53).

Elde edilen bu verinin ilk sütunu yanıt değişkeni olarak dikkate alınan doz değerlerinin standartlaştırılmış skorlarıdır. Sonraki adımda bu değerler, gerçek doz ortalaması ve standart sapması kullanılarak orijinal doz değerlerine dönüştürülmüştür:

$$z_i = \frac{y_i - \bar{y}}{\sigma_y} \qquad (3.2)$$

$$y_i = (z_i \times \sigma_y) + \bar{y}$$

Benzetim çalışmalarında gen sayısı yalnızca p=100 alınarak karşılaştırma yapılmıştır. Örneklem genişlikleri 2. benzetim senaryosu için de 20, 100 ve 500 olarak dikkate alınmıştır. Bootstrap yöntemi için her benzetim adımındaki bootstrap tekrar sayısı; B= 20, 100 ve 300 olarak, çapraz geçerlik yöntemi için ise K- parça sayısı ise K=3, 5 ve 10 olarak dikkate alınmıştır.

Ayrıca, kullanılan model oluşturma yöntemlerinin (DVR ve RA) özel parametreleri için en iyi çözüm veren (best parameters) parametre değerleri de tune() fonksiyonu yardımıyla elde edilmiştir.

Tüm benzetimle elde edilmiş veri setleri, az sayıda gözlem ve çok sayıda değişken içeren gen verilerinde başarılı sonuçlar verebilen makine öğrenmesi regresyon yöntemleri ve bu yöntemleri genelleştiren yeniden örneklemeye dayalı yöntemler ile analiz edilmiştir.

Analizleri gerçekleştirmek amacıyla yöntemlerin genelleştirme yöntemleri ve ayrıca model birleştirme yöntemleri ile çözümleri için R programında kod yazılmıştır. Kullanılan kod yazımı örneği ekte verilmiştir (Bkz. EK 1-4).

3.1.3. Gerçek Veri Setleri Üzerinde Uygulama

Çalışmada gerçek veri seti olarak “Gen Expression Omnibus” açık veri tabanından elde edilen orijinal 11 doz-gen ifade verisi ile literatürdeki regresyon çalışmalarından alınan Boston Housing ve prostat kanseri veri setleri (36) kullanılmıştır. Dikkate alınan bu 13 gerçek veri seti üzerinde hem genelleştirme hem de performans geliştirme yöntemlerinin çözümleri elde edilmiştir.

Klein ve diğerlerinin 2009’daki “*Estimation of the Warfarin Dose with Clinical and Pharmacogenetic Data***”** isimli Teröpatik Varfarin Dozu Konsorsiyumunda bazı çevresel faktörlerin yanı sıra gen ifade değerleri gibi genetik faktörleri de bağımsız değişken olarak kullanarak teröpatik *varfarin dozunun kestirimi* elde edilmiştir (10).

Çalışmamızda gen ifade verilerinden doz tahminini elde etmede bu konsorsiyum referans alınmıştır.

Boston Housing veri seti ilk olarak Harrison, D. ve Rubinfeld, D.L.’in 1998’deki çalışmalarında kullanılmış olup R programından aşağıda verilen komut ile çağrılabilmektedir:

```
>library(mlbench)
>data(BostonHousing)
```

Boston Housing veri setinde bağımlı değişken değeri, Boston bölgesinin banliyösündeki ev sahiplerinin oturdukları ev fiyatlarının (1000 $) ortancası olarak tanımlanmıştır. Ayrıca veri seti bu değişkeni etkileyebilecek 14 açıklayıcı değişkene (suç oranı, yaş, siyah ırk oranı, vb.) sahiptir.

Prostat kanseri veri seti, prostatektomi alan erkek hastalardaki prostata özel antijen seviyesi ile diğer klinik ölçümler arasındaki korelasyonun incelendiği bir çalışmadan alınmıştır (36).

Bağımlı değişken lpsa: prostata özel antijen seviyesi logaritması iken açıklayıcı değişkenler ise lcavol: kanser hacmi logaritması ve lweight: prostat ağırlığının logaritması olarak tanımlanmıştır. Bu veri seti, Cutler'ın 2005'deki çalışmasında bagging ve boosting RA'nın bir karşılaştırmasını vermek için kullanılmıştır (31). Veri seti 2 açıklayıcı değişken ve 97 gözlem içermesine rağmen literatürde regresyon amacıyla sık kullanılan bir veri olduğu için çalışmaya dahil edilmiştir. Veri setlerine ilişkin tanımlayıcı istatistikler Tablo 3.2'de gösterilmiştir.

Tablo 3.2. Gerçek veri setlerine ilişkin tanımlayıcı istatistikler[1].

SN	Veri Seti	Yanıt Değişkeni	Gözlem sayısı	Açıklayıcı Değ. Sayısı	Ortalama	Dağılım Aralığı	Std. Sapma
1	GDS991	doz	15	100	2.25	4.25	1.64
2	GDS3529	doz	16	100	11.75	28.00	11.28
3	GDS1309	doz	20	100	222.00	1000.00	400.84
4	GDS3643	doz	20	100	155.00	495.00	190.73
5	GDS3670	doz	25	100	1.29	3.45	1.25
6	GDS2967	doz	33	100	101.00	599.00	215.15
7	GDS3678	MSD	40	100	136.20	84.00	30.02
8	GDS3143	doz	44	100	11.10	49.95	15.39
9	GDS2998	doz	48	100	19.22	99.70	36.66
10	GDS1429	doz	69	100	13.60	26.70	9.59
11	GDS3640	doz	98	100	326.50	500.00	191.33
12	Pros.Kan.	lpsa	97	2	2.48	5.15	1.15
13	Bos.Hou.	medv	506	14	22.53	45.00	9.20

[1] MSD: Metabolik sendrom değeri

3.2. Model Performansını Geliştiren (Birleştirmeye Dayalı) Yöntemlerin Uygulaması

Bu bölümde, model birleştirmeye dayalı kestirim performansını arttıran bagging ve boosting yöntemlerinin DVR ve RA yöntemleri ile birlikte kullanım kodları yazılmıştır. Kod içinde ilk olarak DVR ve RA yöntemlerinin farklı eğitim ve test setlerine bölünmeleri sonucunda tekli (model birleştirme olmadan) çözüm kodları oluşturulmuştur. İkinci olarak DVR'da Bagging uygulaması özellikle doğrusal ve polinomiyal fonksiyonlar dikkate alınarak *25 bootstrap tekrarı* için düzenlenmiştir. RA'da *Bagging* uygulaması için R programının "ipred" yazılımından yararlanılmıştır. RA'nın *boosting* uygulaması için ise yine R programının "gbm" yazılımı kullanılmıştır. RA'da hem bagging hem de *boosting* için en iyi tekrar ya da ağaç sayısı 10 parçalı çapraz geçerlik yöntemi yardımıyla belirlenmiştir.

Breiman (4), makalesinde bagging yöntemi için öğrenme veri setinin, verinin %90 gibi yüksek bir oranını oluşturması durumunda bagging performansının geliştiğini belirtmiştir. Tez çalışmasında Breiman'ın uygulamasından farklı olarak kullanılan gerçek veri setlerinin örneklem genişliklerine göre farklı eğitim ve test oranları kullanılmıştır. Model ve kestirim setlerinin ayarlaması test olarak kullanılan veri setlerinde en az 6 gözlem bulunacak şekilde belirlenmiştir. Bagging yöntemi ile kestirim ya da sınıflama performansını inceleyen bir çok çalışmada eğitim ve test setlerinin hangi oranlarda alındığı ya da nasıl bir algoritma izlendiği çok açık şekilde belirtilmemektedir.

Bagging yönteminin DVR ve RA yöntemleri için 13 gerçek veri seti üzerinde uygulaması sırasında izlenen yollar aşağıda sıralanmıştır:

I. Gözlem sayısı 25 ve altında olan gerçek veri setlerinin % 25'i test seti olarak, kalan % 75'ini oluşturan kısım ise eğitim seti olarak rastgele bölünmüştür.

II. Gözlem sayısı 25'in üstünde olan gerçek veri setlerinin % 80'i eğitim seti, kalan % 20'lik kısmı ise test seti olacak şekilde ikiye ayrılmıştır.

III. Eğitim setinden yerine koyarak E_B bootstrap örneklemi çekilmiştir ve çekilen E_B ile model kurulmuştur. Elde edilen bu model başlangıçta ayrılan test setinde uygulanarak kestirimler elde edilmiştir. Bu adımlar 25 defa tekrarlanarak yöntemlere ilişkin kestirimler $\widehat{y_1}(x), \hat{y}_2(x), \ldots, y_{25}(x)$ elde edilmiştir.

IV. Verinin eğitim ve test olarak rastgele bölünmesi 100 defa tekrar edilmiş ve 100 tekrarda elde edilen KHKO, OMS ve R^2 performans ölçülerinin ortalaması alınmıştır.

4. BULGULAR

4.1. Benzetim Çalışmasının Bulguları

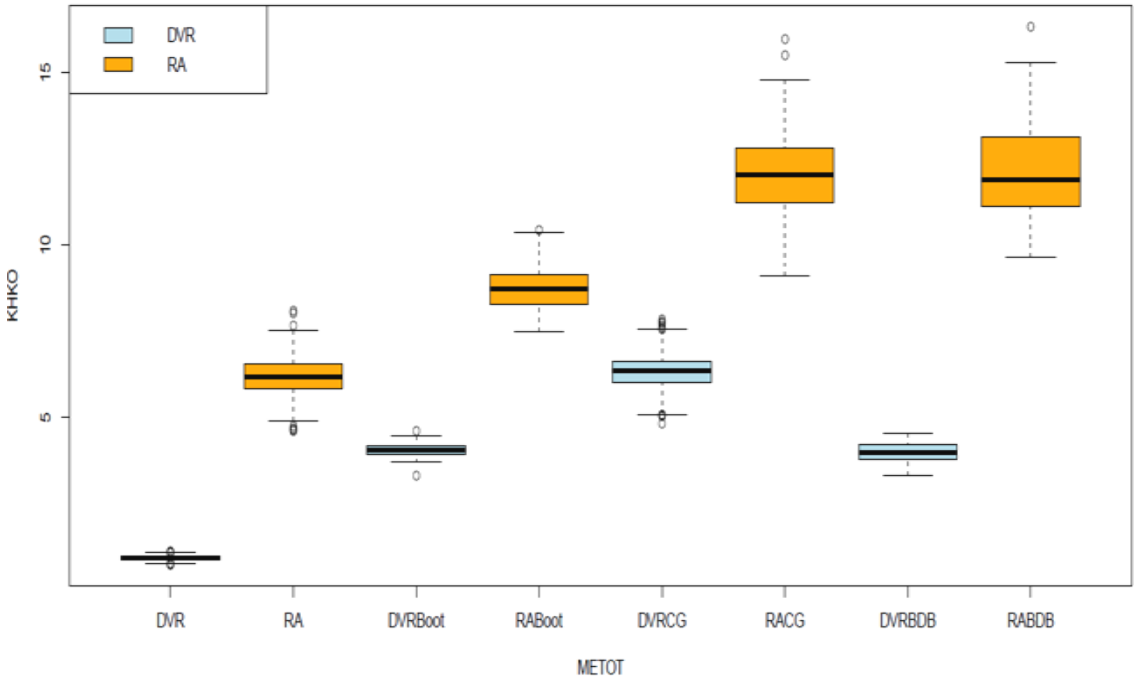
Yöntemlerin performanslarının incelenmek istendigi koşullar altında olası senaryo sayısının çok fazla olması nedeniyle, benzetim çalısmasının sonuçlarını içeren tablolarda, sunum kolaylığı olması açısından eğilimi belli edecek kadar kombinasyon seçilmiştir. Buna rağmen hala çok fazla sonuç olması nedeniyle tabloların bir bölümü bulgularda yer alırken, benzetim çalışmasının sonuçlarını içeren tabloların büyük bir bölümü ekte (Bkz. EK 5-16) sunulmuştur.

İncelenen temel regresyon yöntemlerinin 1000 benzetim tekrarındaki tekli çözümleri, bootstrap tekrar sayısının B=20 alınması ile elde edilen sonuçlar, yine parça sayısının k=3 alındığı çapraz geçerlik yöntemi ile elde edilen sonuçlar ve k=n olarak tanımlı BDBCG ile elde edilen sonuçlar Tablo 4.1'de verilmiştir.

Tablo 4.1. Modele bağlı, n=100, σ_e=1, ρ=0 ve β=1 olarak alınan senaryo sonuçları.

SS(y)=9.52	σ_e=1, β=1	p=100, $\rho_{xi,xj}$=0	n=100	ORT(y)=0.29
Yöntem	**KHKO±SS**	**R^2**	**OMS**	
DVR	0.935±0.34	0.993	0.90	DVS=80
RA	6.200±2.59	0.614	4.82	SDS= 8
DVR-Boot	4.058±1.29	0.839	2.51	---
DVR-CG	6.365±2.53	0.606	5.06	---
RA-Boot	8.729±3.20	0.331	6.22	
RA-CG	12.029±5.01	0.042	9.64	
DVR-BDB	3.998±1.54	---	2.50	
RA-BDB	11.956±6.10	---	9.55	

KHKO değerleri için hem 1000 tekrardaki ortalama değerler hem de bunlara ilişkin standart hatalar gösterilmiştir. Şekil 4.1'de ilk senaryodaki KHKO sonuçları için kutu çizgi grafiği elde edilmiştir.



Şekil 4.1. Benzetim senaryosu 1 için genelleştirme yöntemlerinin performansları

Sonuçlar incelendiğinde, hem modeldeki açıklayıcı değişken sayısının hem de örneklem sayısının n=p=100 olması, aynı veri seti üzerinden test edilen modellerin veriye aşırı uyumlu olması gibi yanıltıcı sonuçlara yönlendirmektedir. Bu nedenle yöntemlerin tekli çözüm performansları (DVR için R^2=0.993, RA için R^2=0.61) genelleştirme yöntemleriyle elde edilenlere göre daha başarılıdır. Yöntem performansları karşılaştırıldığında ise DVR'nin RA'dan hem ortalama hem de standart hata açısından daha küçük KHKO değerlerine sahip olduğu gözlenmiştir.

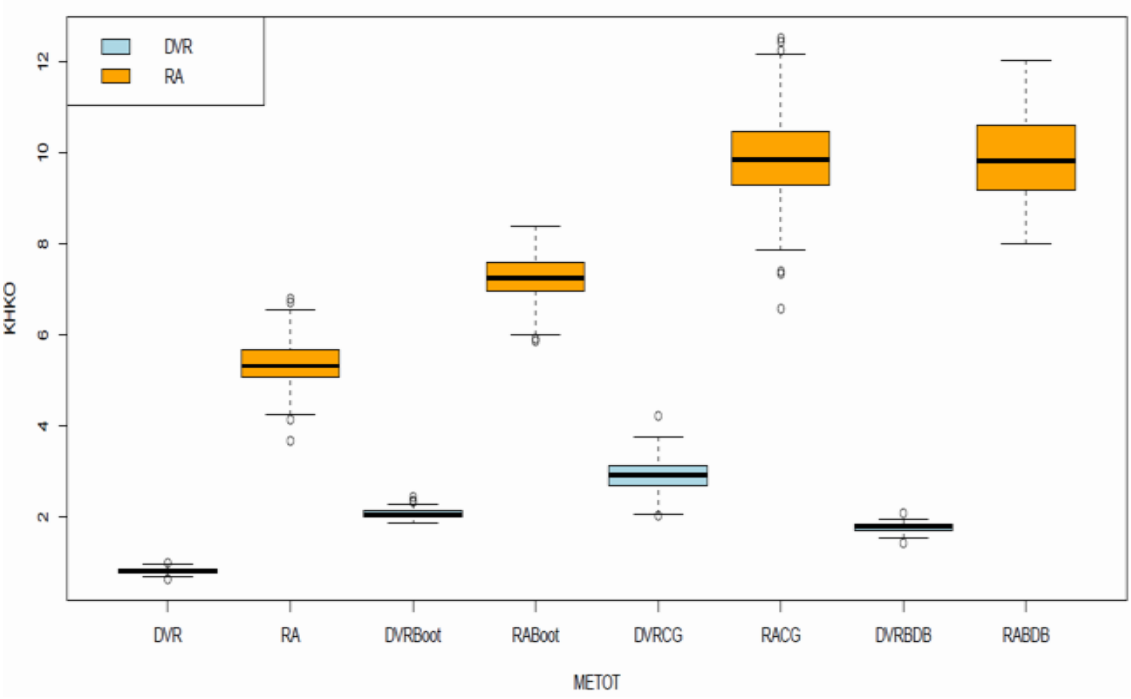
Genler arasındaki gerçek korelasyon yapısına bağlı oluşturulan senaryo 2 için performans ölçüleri Tablo 4.2'de verilmiştir. Bu sonuçlar Tablo 4.1'de elde edilen sonuçlara oldukça benzerdir. DVR için en küçük KHKO değerine sahip genelleştirme yöntemi ilk tablo için BDBCG iken benzer şekilde ikinci durum için de BDBCG yöntemi olduğu görülmektedir. Ancak KHKO'ya ilişkin elde edilen standart hatalar karşılaştırıldığında hem DVR hem de RA yöntemleri için BDBCG'in bootstrap yöntemine göre performansı zayıf bulunmuştur.

Genler arasındaki korelasyonlar senaryo 1'de $\rho_{xi,xj}=0$, iken ikinci senaryoda gerçek korelasyonlara benzer oluşturulmuştur. İlk senaryoda elde edilen ortalama destek vektör sayısı (DVS=80) iken, ikinci senaryoda DVS=58.25'e düşmüştür. Benzer şekilde ilk durumda RA yönteminden elde edilen ortalama son düğüm sayısı (SDS=8) iken ikinci senaryoda SDS= 7.13 bulunmuştur.

Tablo 4.2.Genler arası korelasyon yapısına dayalı senaryo sonuçları.

p=100,n=100	ORT(doz)=13.60,ss(doz)=9.59			
Yöntem	**KHKO±SS**	**R^2**	**OMS**	
DVR	0.809±0.298	0.993	0.749	DVS=58.25
RA	5.392±2.24	0.679	4.209	SDS= 7.13
DVR-Boot	2.044±0.64	0.956	1.454	---
DVR-CG	2.957±1.33	0.905	2.336	---
RA-Boot	7.464±2.54	0.451	5.389	
RA-CG	9.959±4.31	0.149	7.956	
DVR-BDB	1.780±0.78	---	1.408	
RA-BDB	9.603±4.29	---	7.609	

Tablo 4.1 ile Tablo 4.2 sonuçlarına dikkat edildiğinde Tablo 4.2'de kestirim performansının daha da iyileştiği şeklinde yanıltıcı sonuca ulaşılabilir.



Şekil 4.2. Benzetim senaryosu 2 için genelleştirme yöntemlerinin performansları.

Tablo 4.2 sonuçları Şekil 4.2'de görsel hale getirilmiştir. RA yöntemi için en iyi performans gösteren genelleştirme yöntemi her iki benzetim düzeni için de bootstrap yöntemi olarak belirlenmiştir. Değişkenlik açısından değerlendirildiğinde ise hem DVR hem de RA için en küçük standart hataya sahip genelleştirme yöntemi ise yine *bootstrap*'tir.

Çalışmada dikkate alınan tüm durumlar için DVR yönteminden elde edilen destek vektör sayıları ile regresyon ağacından elde edilen bölünme sayıları ya da son düğüm sayıları da raporlanmıştır.

Özellikle gözlem sayısının açıklayıcı değişken sayısına göre yetersiz olduğu durumlarda (Bkz. Şekil 2.3) yani aşırı uyumlu durumlarda söz konusu regresyon yöntemleri için genelleştirme yöntemlerinin dışında alternatifler de vardır. Bunlar RA yöntemi için gereksiz bölünmelerde budama yapmak ve DVR için ise katsayılarda küçültme yapmak (*shrinkage*) gibi yöntemleri içerir.

Karar ağacı regresyonuna her iki senaryo için 10 parçalı CG kullanılarak budama yapılmıştır.

Senaryo I ve senaryo II için regresyon ağacı ve budamalı regresyon ağacının 1000 tekrarlı çözümleri sırasıyla Tablo 4.3'te ve Tablo 4.4'te verilmiştir. Bu amaçla yazılan kod ekte verilmiştir (Bkz. EK 1-5).

Tablo 4.3. Senaryo I için RA ve budamalı RA'nın 1000 tekrarlı çözümleri. (n=100, β=1, σ_ε=1).

YÖNTEM	KHKO±SS	OMS	SDS
RA	6.20 ± 2.59	4.82	8
RA-budanmış	9.64 ± 4.1	7.67	1.21

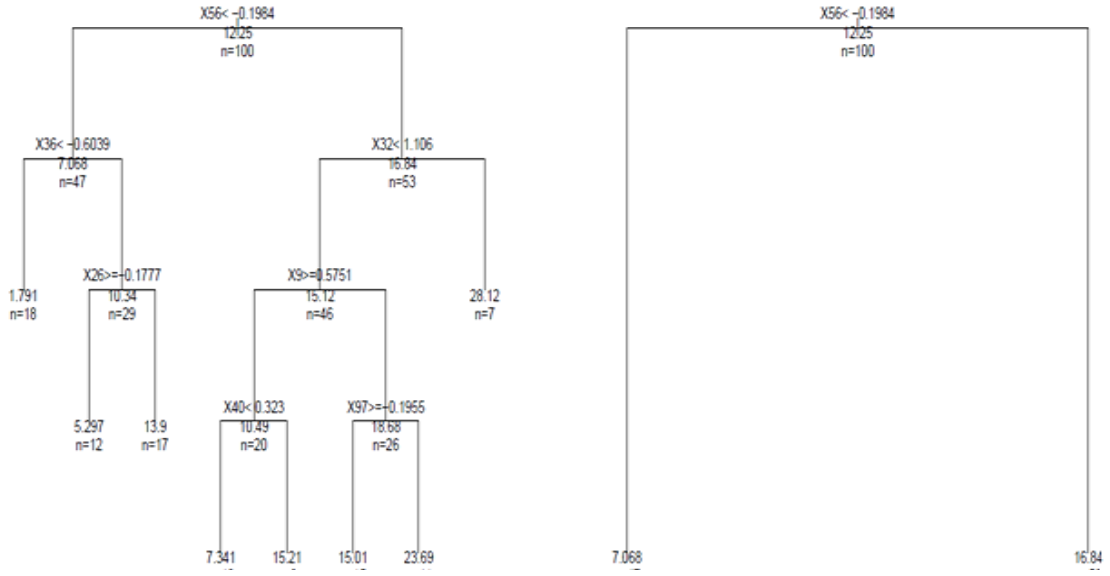
Tablo 4.4.Senaryo II için RA ve budamalı RA'nın 1000 tekrarlı çözümleri. (n=100).

YÖNTEM	KHKO±SS	OMS	SDS
RA	5.39 ± 2.24	4.20	7.13
RA-budanmış	7.84 ± 4.48	6.16	1.84

Tablo 4.3 ve Tablo 4.4 sonuçları incelenirse, RA yönteminde aşırı uyumdan kaynaklanan ve model uyumunda anlamlı fark yaratmayan bölünmeler çalışmada varsayılan değere (cp değeri) göre geri alınmıştır. Budanmış senaryolarda ilk olarak regreyon ağaçları olabilecek en büyük ağaç şeklinde geliştirilmiştir. Daha sonra "cp" değeri olarak bilinen (complexity parameter) parametre değerinin 0.01'den daha küçük olduğu bölünmeler 10 parçalı çapraz geçerlik yöntemine göre budanmıştır.

Sonuç olarak gereksiz bölünmeler yani bölünme sayısı azalmış KHKO kestirimi artmıştır. İlk senaryo için ortalama 8 son düğümden budamalı durumda son düğüm sayısı 1.21'e düşmüştür (Tablo 4.3). İkinci durumda ise ortalama 7.13 son düğümden budama yapıldıktan sonraki son düğüm sayısı 1.84'e değişmiştir.

Senaryo I için elde edilen RA Şekil 4.3 ile verilmiştir.



Şekil 4.3. Benzetim senaryosu 1 için sırasıyla RA ve budanmış RA modelleri.

Destek Vektör Makinası için model performansı değerlendirilen fonksiyon türleri; doğrusal, polinomiyal, radyal ve sigmoid fonksiyonlarıdır. Küçük (n=100) ve büyük (n=500) örneklem genişlikleri için bu fonksiyon türlerinin 1000 tekrarlı benzetim sonuçları incelenmiştir. Hem senaryo I hem de senaryo II için fonksiyonların performans ölçülerine ilişkin ortalama değerler Tablo 4.5 ve Tablo 4.6'da özetlenmiştir.

Sonuçlar incelendiğinde, hem genler arası korelasyonun "0" olarak dikkate alındığı senaryoda hem de genler arası gerçek korelasyon yapısı ile bağımlı değişken ölçeklemesi dikkate alınarak oluşturulan senaryoda en iyi ilk performansı doğrusal fonksiyon, ikinci en iyi performansı ise radyal tabanlı fonksiyon sağlamıştır.

Tablo 4.5. Benzetim senaryosu I için DVM fonksiyon performansları[2].

n=100, $\rho_{xi,xj}=0$

Perf. Ölçüsü	Fonksiyon	DVR	DVR-Boot	DVR-CG	DVR-BDB
KHKO	DOĞRUSAL	0.935	4.058	6.365	3.998
	POLİNOM	4.012	6.516	9.984	9.897
	RADYAL	3.502	5.978	9.091	8.572
	SİGMOİD	4.279	5.698	7.339	5.887
	En iyi	**DOĞ**	**DOĞ**	**DOĞ**	**DOĞ**
OMS	DOĞRUSAL	0.901	2.511	5.061	2.503
	POLİNOM	2.402	4.040	7.958	7.912
	RADYAL	2.158	3.774	7.236	6.838
	SİGMOİD	3.067	4.009	5.843	4.689
	En iyi	**DOĞ**	**DOĞ**	**DOĞ**	**DOĞ**
R^2	DOĞRUSAL	0.993	0.839	0.606	---
	POLİNOM	0.906	0.606	0.269	---
	RADYAL	0.948	0.689	0.405	---
	SİGMOİD	0.854	0.688	0.499	---
	En iyi	**DOĞ**	**DOĞ**	**DOĞ**	**---**

n=100 için fonksiyonlara ilişkin elde edilen tablolar bu bölümde verilmiş ancak n=500'ü içeren tablolar ekte yer almıştır (Bkz. EK 13). Örneklem genişliği n=100'den n=500'e değiştiğinde genel olarak kestirim performanslarında bir iyileşme mevcuttur. Ancak gözlem sayısındaki artış ya da azalış en iyi ilk fonksiyon ve en iyi ikinci fonksiyon sıralamasını etkilememiştir. Diğer taraftan genler arasında belirli miktarda korelasyon bulunması *sigmoid* fonksiyonun hatasını daha fazla arttırmıştır (Tablo 4.6). Diğer fonksiyon türleri incelenirse senaryo I'den senaryo II'ye geçişte hata değerlerinde önemli bir değişim olmadığı görülebilir.

[2] Koyu vurgu en iyi ilk performansı, açık ton vurgu ise en iyi ikinci performansı göstermektedir.

Benzer şekilde n=500 için sigmoid fonksiyona ilişkin KHKO değeri ilk senaryoda 3.172 iken ikinci senaryoda 37.062'ye değişmiş ve OMS değeri ise 2.76'dan 19.35'e yükselmiştir (Bkz. EK 14).

Tablo 4.6. Benzetim senaryosu 2 için DVM fonksiyon performansları[3].

n=100 , $\rho_{xi,xj}\neq 0$

Perf. Ölçüsü	Fonksiyon	DVR	DVR-Boot	DVR-CG	DVR-BDB
KHKO	DOĞRUSAL	0.809	2.044	2.957	1.781
	POLİNOM	4.517	6.060	8.407	8.089
	RADYAL	3.209	5.068	7.557	7.001
	SİGMOİD	7.493	8.373	6.450	7.026
	En iyi	**DOĞ**	**DOĞ**	**DOĞ**	**DOĞ**
OMS	DOĞRUSAL	0.749	1.454	2.336	1.409
	POLİNOM	2.822	3.941	6.681	6.363
	RADYAL	2.075	3.279	5.998	5.514
	SİGMOİD	5.263	5.931	5.074	5.139
	En iyi	**DOĞ**	**DOĞ**	**DOĞ**	**DOĞ**
R^2	DOĞRUSAL	0.993	0.956	0.905	---
	POLİNOM	0.812	0.633	0.291	---
	RADYAL	0.935	0.761	0.465	---
	SİGMOİD	0.456	0.397	0.566	---
	En iyi	**DOĞ**	**DOĞ**	**DOĞ**	**---**

Kullanılan temel yöntemlerin özel parametre değerlerinin ayarlanması açısından da sonuçlar karşılaştırılmıştır. DVR'de ε, C ve γ parametreleri için en küçük hatayı sağlayan değerler eşzamanlı olarak e1071 paketindeki tune() fonksiyonu yardımıyla aratılmıştır.

[3] Koyu vurgu en iyi ilk performansı, açık ton vurgu ise en iyi ikinci performansı göstermektedir.

Benzer şekilde RA için bir düğümdeki en küçük gözlem sayısı, cp değeri ve etkileşim derecesi parametreleri rpart paketindeki rpart.control() fonksiyonu kullanılarak eş zamanlı olarak en iyi performans değerleri bulunmuştur. Varsayılan parametre değerlerine göre elde edilen sonuçlar Tablo 4.7'de, en iyi parametre ayarlaması ile elde edilen sonuçlar ise Tablo 4.8'de görülmektedir.

Tablo 4.7. Doz kestirimi için varsayılan parametrelere göre benzetim sonuçları (ρxixj=0).

σ_e=1, p=100, n=100, b=1

Yöntem	KHKO±SS	R^2	OMS	DVS	SDS
DVR	0.93 ± 0.34	0.993	0.901	80	---
RA	6.20 ± 2.59	0.614	4.827	---	8
DVR-Boot	4.05 ± 1.29	0.839	2.511		
DVR-CG	6.36 ± 2.53	0.606	5.061		
RA-Boot	8.72 ± 3.20	0.331	6.227		
RA-CG	12.02 ± 5.01	0.042	9.640		

Tablo 4.8. Doz kestirimi için en iyi ayarlama parametrelerine göre benzetim sonuçları (ρ_{xixj} = 0).

σ_ε=1, p=100, n=100

Yöntem	**KHKO±SS**	**R^2**	**OMS**	**DVS**	**SDS**	**parametre**
DVR	0.93 ± 0.32	0.993	0.903	80.32		ε=0.05. C=2
RA	5.64 ± 2.75	0.681	4.462		8.5	cp=0.03,etkileşim=4
DVR-Boot	3.95 ± 1.28	0.840	2.199			ε=0.05, C=0.25
DVR-CG	6.36 ± 2.67	0.607	5.060			
RA-Boot	8.64 ± 3.21	0.328	6.155			cp=0.01
RA-CG	10.93 ± 4.84	---	8.709			cp=0.002

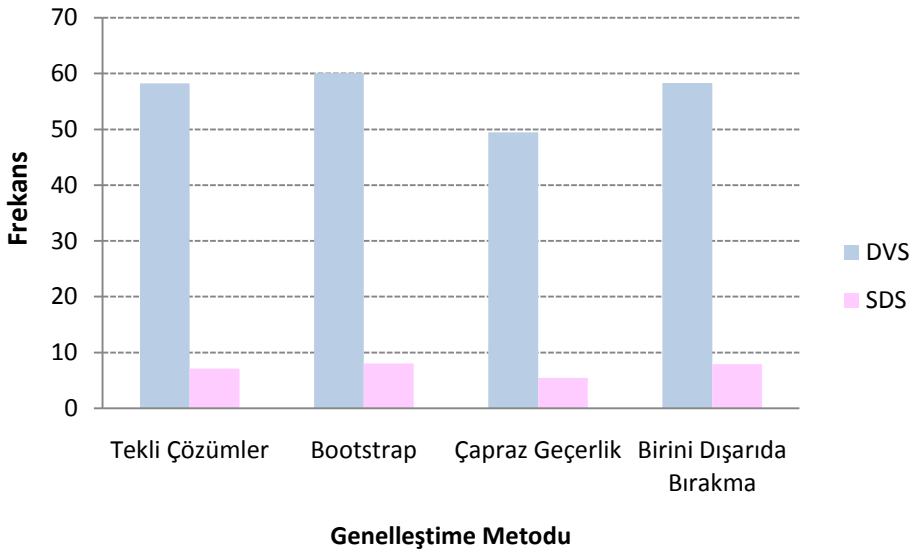
DVR yönteminde C'nin özellikle çok büyük değerleri ve ε'nun çok küçük değerleri aşırı uyuma neden olmaktadır. Genel olarak Tablo 4.8'deki hata kestirimleri Tablo 4.7'deki varsayılan parametrelere göre bulunan kestirimlere yakındır. RA yönteminde ise cp değeri daha küçük ve etkileşim derecesi ise daha büyük olacak şekilde ayarlandığında model hatası daha küçük tahmin edilmiştir. En iyi parametrelerin ayarlaması her iki regresyon yöntemi için en küçük hata kestirimi verecek şekilde yapıldığı için elde edilen ortalama destek vektör sayısı ve elde edilen ortalama son düğüm sayısında ilk duruma kıyasla artış görülmektedir.

Kullanılan genelleştirme yöntemleri için DVR ve RA modellerinin 1000 tekrardaki ortalama destek vektör sayıları ve ortalama son düğüm sayılarının nasıl bir eğilim gösterdikleri de incelenmiştir. Bu değişimleri ifade eden ilgili tablo ve grafik sırasıyla Tablo 4.9 ve Şekil 4.4'te verilmiştir.

Tablo 4.9. Genelleştirme yöntemlerine göre yönteme özel parametre değişimleri (n=100, ρ_{xixj} =0 ve σ_{ε}=1).

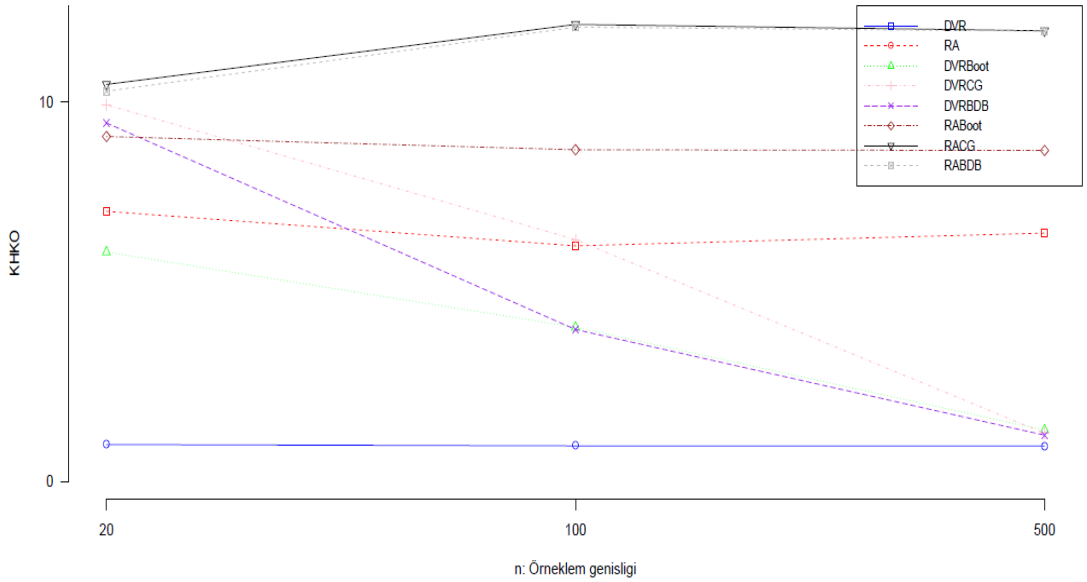
Yöntem	**DVS**	**SDS**
Tekli Çözümler	58.25	7.13
Bootstrap	60.09	8.05
Çapraz geçerlik	49.45	5.46
Birini Dışarıda Bırakma	58.31	7.94

Elde edilen destek vektör sayıları ve son düğüm sayıları genelleştirme yöntemlerinin Tablo 4.1, Tablo 4.2, Şekil 4.1 ve Şekil 4.2'deki hata performansları ile uyumlu sonuçlar sergilemiştir. Yine Tablo 4.1'de KHKO değeri diğerlerine göre en yüksek olan çapraz geçerlik yöntemi en küçük DVS ve SDS kestirimlerini vermiştir (Tablo 4.9, Şekil 4.4). Diğer taraftan en iyimser performans tahmini sağlayan bootstrap yöntemi ise DVS ve SDS özel parametrelerini diğerlerinden daha yüksek belirlemiştir.

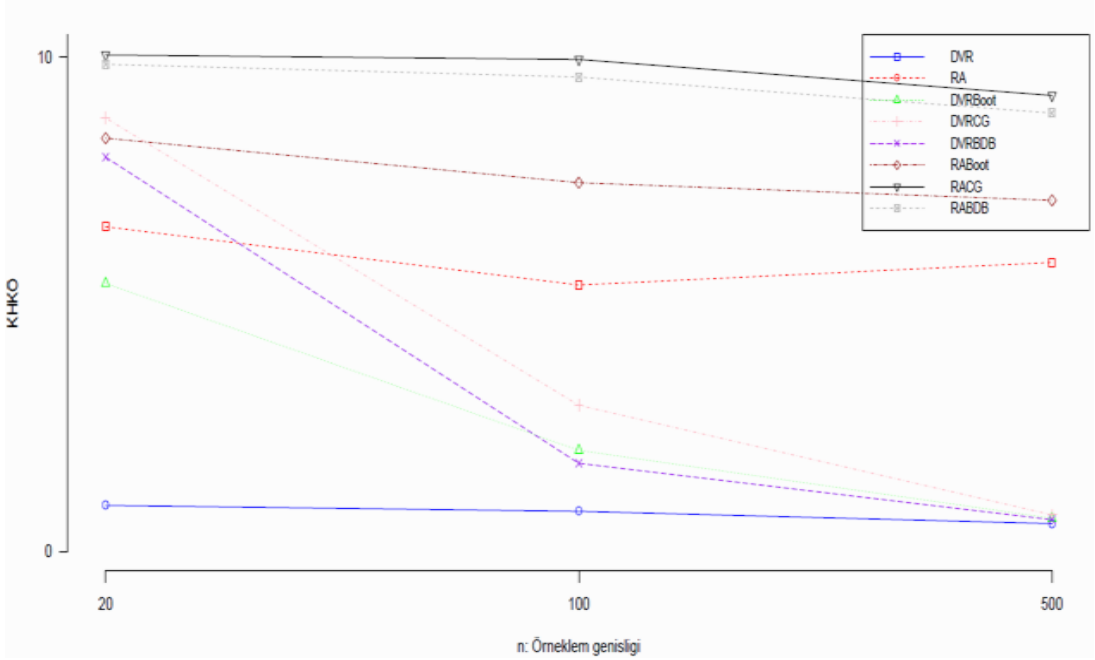


Şekil 4.4. Yönteme özel parametre değişimi.

Örneklem genişliğinin n=20, 100 ve 500 değerlerindeki değişimi açısından her iki benzetim senaryosu için performanslar incelenmiş sırasıyla Şekil 4.5 ve Şekil 4.6'da gösterilmiştir. DVR'nin eğitim seti performansına bakılırsa her iki senaryo için örneklem genişliğindeki değişimden etkilenmediği görülmektedir. Benzer şekilde RA yönteminin de hem eğitim performansı hem de genelleştirme performansları örneklem genişliğine göre önemli bir değişime sahip değildir. Buna karşın, DVR'nin genelleştirme yöntemleriyle olan çözümlerine bakılırsa, örneklem genişliği 20'den 500'e çıktığında her iki senaryoda hatada anlamlı bir azalış göze çarpmaktadır; ancak RA yönteminde örneklem genişliğinin değişimi (özellikle 100'den 500'e) hata kestirimini çok etkilememektedir (Bkz. EK 5-EK 7).

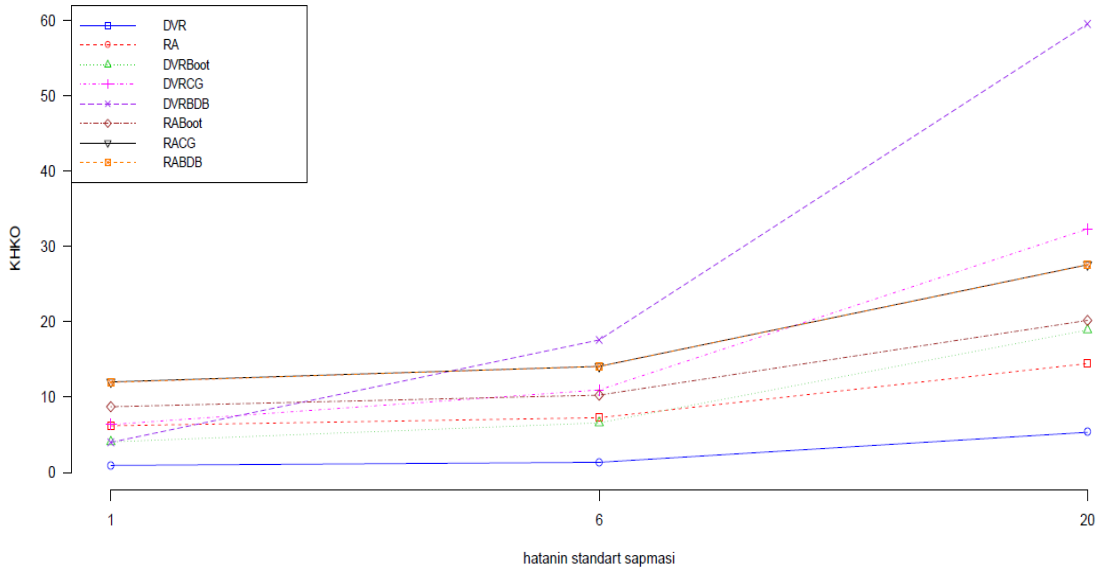


Şekil 4.5. Modele bağlı gerçekleştirilen benzetim çalışmasının örneklem genişliğine bağlı performansı.



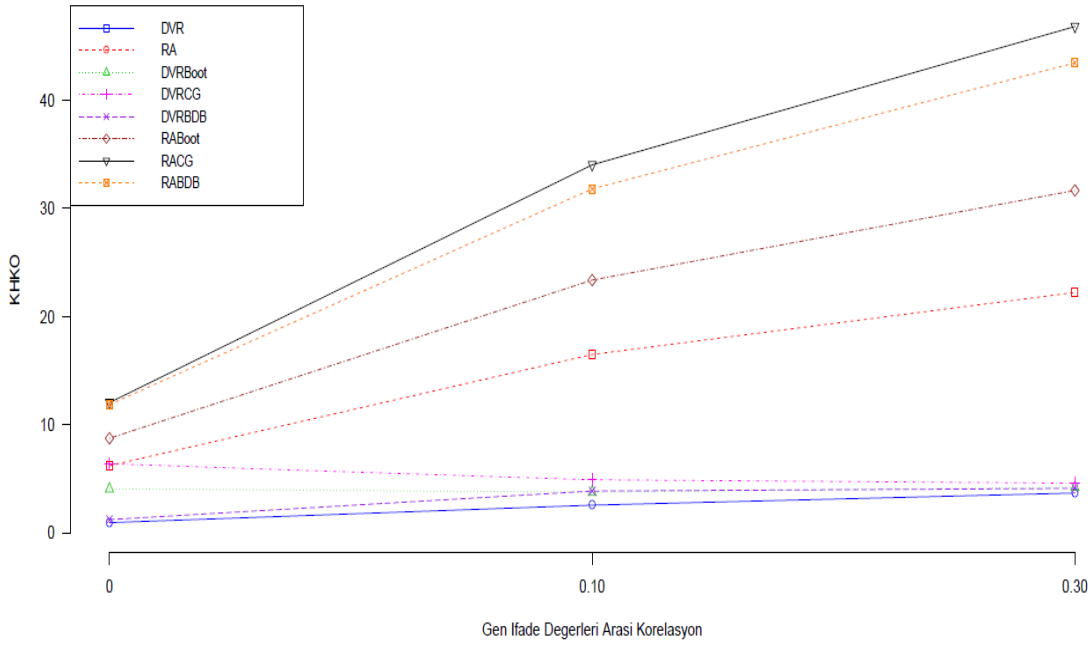
Şekil 4.6. Korelasyon yapısına bağlı gerçekleştirilen benzetim çalışmasının örneklem genişliğine bağlı performansı.

σ_ε'ya göre değişimler incelendiğinde özellikle DVR için BDBCG yöntemine ilişkin ortalama KHKO kestiriminin σ_ε'nın büyük değerlerinde aşırı arttığı dikkat çekmektedir (Şekil 4.7). Standart sapma değeri 1'den 6'ya yükseldiğinde tüm yöntemler için önemli bir değişim olmamış fakat 6'dan 20'ye olan yükselişte beklenildiği gibi KHKO kestirimlerinde belirgin bir artış görülmüştür (Bkz. EK 5- EK 11).



Şekil 4.7. Senaryo I için regresyon hatasının standart sapmasına (σ_ε) bağlı performans değişimi.

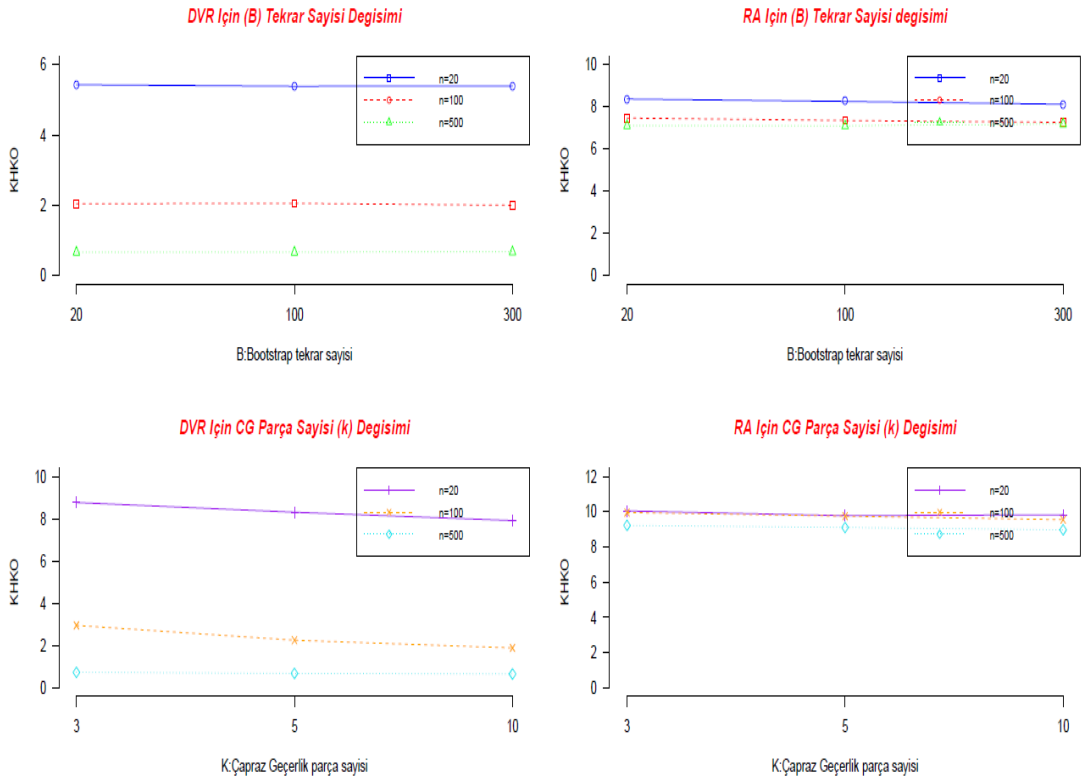
Genler arasında ayarlanan üç farklı korelasyon değeri (0, 0.10 ve 0.30) açısından performans ölçüsündeki değişimler de değerlendirilmiş ve R programında çizdirilen grafikleri Şekil 4.8'de verilmiştir.



Şekil 4.8. Senaryo I için genler arası farklı korelasyon değerlerindeki (ρ=0, 0.10, 0.30) performans değişimi.

Genler arası ilişki (ρ) arttığında DVR yöntemi KHKO sonuçları bu artıştan etkilenmemiştir. Ayrıca DVR'nin genelleştirme yöntemlerinden elde edilen kestirimleri de genler arası ilişkideki artışa göre değişim göstermemiştir; ancak RA yönteminden elde edilen KHKO değerleri ρ'ya orantılı olarak artmıştır. RA yöntemindeki artışı belirten eğim özellikle 0 ile 0.10 değerleri arasında daha diktir (Bkz. EK 6, EK 8, EK 10).

Bootstrap tekrar sayısına (B) ve çapraz geçerlik parça sayısına (k) göre yöntemlerin KHKO sonuçlarındaki değişimler R programında n=20,100 ve 500 için grafik olarak elde edilmiş ve aşağıda verilmiştir. DVR ve RA yöntemleri için oluşturulan 4 farklı grafik R programındaki mfrow() fonksiyonu yardımıyla birleştirilmiştir (Şekil 4.9).



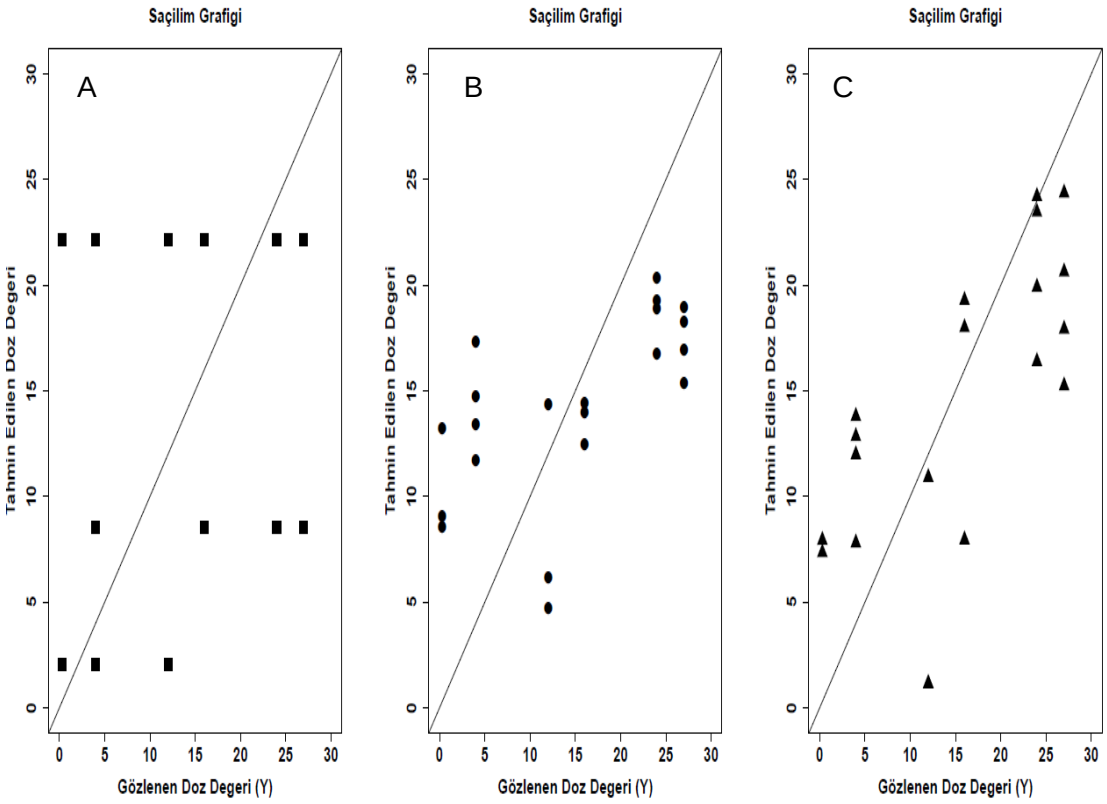
Şekil 4.9. Her benzetim tekrarında bootstrap yeniden örnekleme sayısı(B) ve çapraz geçerlik (k) parça sayısına göre değişimler.

Sonuçlara göre KHKO değerleri bootstrap tekrar sayılarına göre önemli bir değişim göstermemekte ancak çapraz geçerlik için parça sayısı 3'ten 10'a yükseldikçe KHKO değerlerinde önemli ölçüde azalışlar gözlenmektedir. Özellikle bootstrap yönteminde 100 tekrardan sonra KHKO değerlerinde çok ciddi değişimler olmadığı göze çarpmaktadır (Bkz. EK 15, EK 16).

4.2. Model Birleştirme Yöntemlerinin Performans Sonuçları

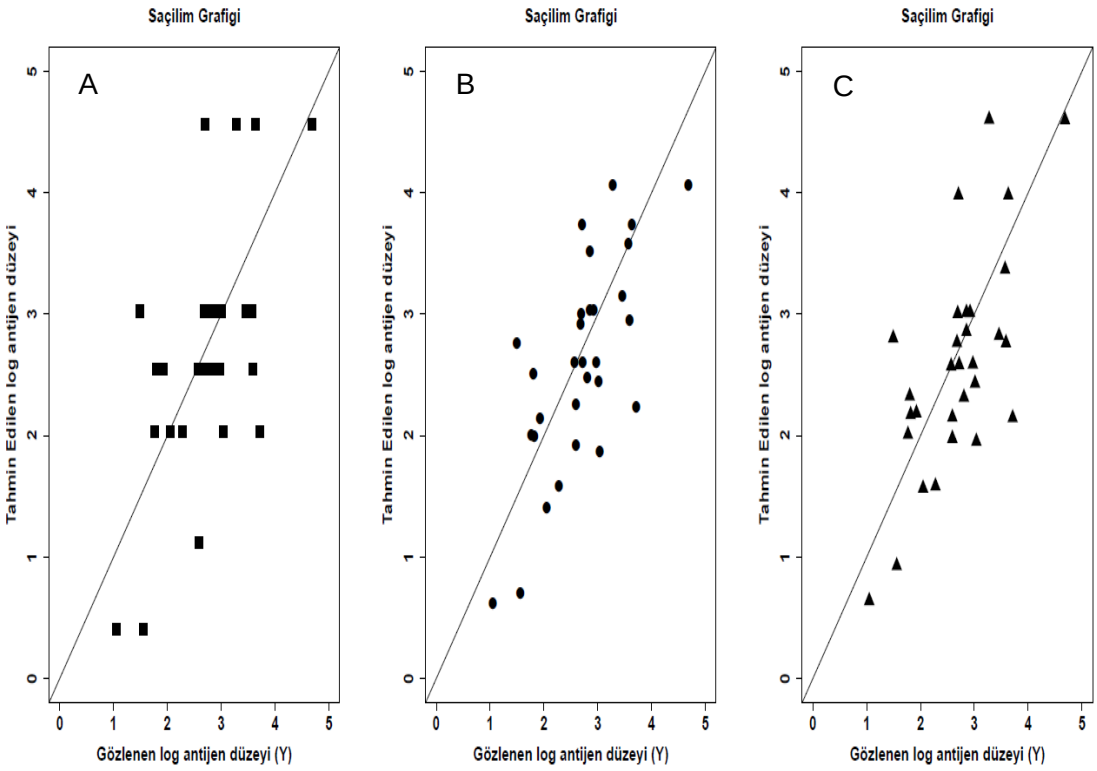
Çalışmada ele alınan model birleştirmeye dayalı bagging ve boosting yöntemlerinin incelenen tüm veri setleri için 100 tekrarlı performansları değerlendirilmiştir.

Performansları değerlendirmede KHKO, OMS ve R^2 ölçüleri hesaplatılmış, buna ek olarak KHKO değerlerine ait standart hatalar da kayıt edilmiştir. İlk olarak RA yönteminin test veri seti üzerindeki tekli performansı, *bagging* performansı ile *boosting* performansı arasındaki farklılığı görmek amacıyla yanıt değişkeninin gözlenen ve beklenen değer saçılım grafikleri değerlendirilmiştir (Şekil 4.10, Şekil 4.11).



Şekil 4.10. gds1429 ID kodlu gen verisi için sırasıyla A)RA, B)RA-bagging ve C)RA-boosting modellerinden elde edilen doz'un kestirim değerine karşı gözlenen değer saçılım grafiği.

Gds1429 gen veri seti (n=69, p=100) için 100 tekrarda elde edilen ortalama son düğüm sayısı SDS=4'tür. Beklenildiği gibi regresyon ağacı modeli için saçılım grafiği her iki veri seti için "adıma-benzer" kestirimler göstermiş ve bu adımların sayısı ise ağaçlardan elde edilen ortalama son düğüm sayısına uymak zorunda kalmıştır.



Şekil 4.11. Prostat kanseri veri seti için sırasıyla A)RA, B)RA-bagging ve C)RA-boosting modellerinden elde edilen log antijen düzeyinin kestirim değerine karşı gözlenen değer saçılım grafiği

Prostat kanseri veri seti (n=97, p=2) için 100 tekrarda elde edilen ortalama son düğüm sayısı ise SDS=5 olarak bulunmuştur. Şekil 4.11 incelendiğinde RA'daki adım sayısının bu değere eşit olduğu görülmektedir. Bunun yanı sıra RA-bagging ve RA-boosting (gbm) modellerinin saçılım grafikleri istenilen doğrusal korelasyona daha yakın görünüm sergilemiştir. Buna rağmen her üç veri seti için de RA'dan elde edilen ortalama son düğüm sayısı RA-bagging'den elde edilen ortalama son düğüm sayısından fazladır.

Bu bölümde 13 veri setinden sadece Boston Housing, Prostat Kanseri veri setleri ile yüksek boyutlu gen verilerinden gds1429 ID numaralı veri seti sonuç tabloları verilmiştir (Tablo 4.10- Tablo 4.12).

Tablo 4.10. Prostat Kanseri verisinin tekli ve model birleştirme yöntemleri ile performans sonuçları (100 tekrardaki ortalama sonuçlar).

Prostat K. verisi	**KHKO±SS**	**OMS**	**R^2**	**SDS**
DVR	0.746 ± 0.435	0.583	0.591	
RA	0.850 ± 0.458	0.704	0.489	5.49
DVR-bag	0.768 ± 0.435	0.604	0.584	
RA-bag	0.774 ± 0.424	0.628	0.562	5.28
RA-boost	0.771 ± 0.412	0.622	0.573	

Tablo 4.11. gds1429 gen verisinin tekli ve model birleştirme yöntemleri ile performans sonuçları (100 tekrardaki ortalama sonuçlar).

GDS1429	**KHKO±SS**	**OMS**	**R^2**	**SDS**
DVR	6.96 ± 4.43	5.42	0.54	
RA	10.08 ± 5.78	7.91	0.18	4.22
DVR-bag	7.82 ± 3.65	6.17	0.43	
RA-bag	8.06 ± 3.72	6.95	0.36	4.08
RA-boost	7.68 ± 3.56	6.38	0.40	

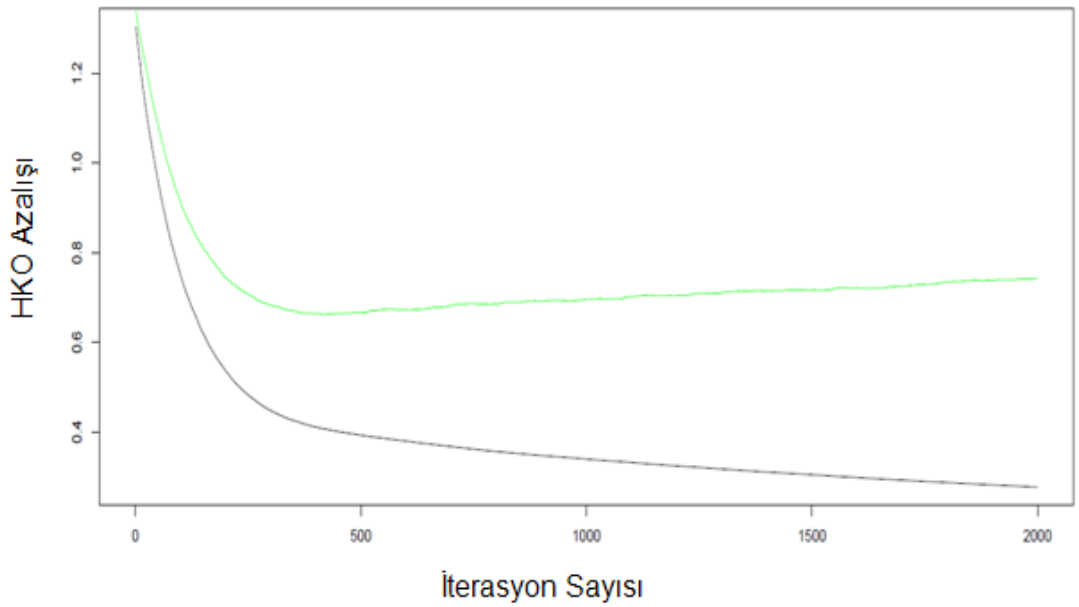
Tablo 4.12. Boston Housing verisinin tekli ve model birleştirme yöntemleri ile performans sonuçları (100 tekrardaki ortalama sonuçlar).

Boston Housing	**KHKO±SS**	**OMS**	**R^2**	**SDS**
DVR	4.94± 3.22	3.20	0.72	
RA	5.07± 3.24	3.46	0.69	6.62
DVR-bag	5.01± 3.22	3.26	0.71	
RA-bag	4.08± 2.87	2.77	0.80	6.54
RA-boost	3.89 ± 2.51	2.56	0.82	

Tablo sonuçları dikkatle incelenirse RA ve DVR yöntemlerinin test veri seti üzerindeki kestirimleri ile model birleştirme yöntemi kestirim performansları farklılık göstermiştir.

DVR yönteminin test seti KHKO performansının tüm veri setlerinde DVR-bagging'e göre daha başarılı olduğu görülmektedir. Ancak tüm veri setlerinde RA yönteminin performansı incelenirse hem bagging hem de boosting yönteminin RA'nın test seti performansını daha fazla iyileştirdiği söylenebilir. Özel olarak bagging ve boosting performansları kıyaslandığında ise veri setlerinin büyük kısmında boosting'in hem ortalama (KHKO) hem de standart hata açısından bagging yönteminden daha iyi kestirim sağladığı görülmektedir.

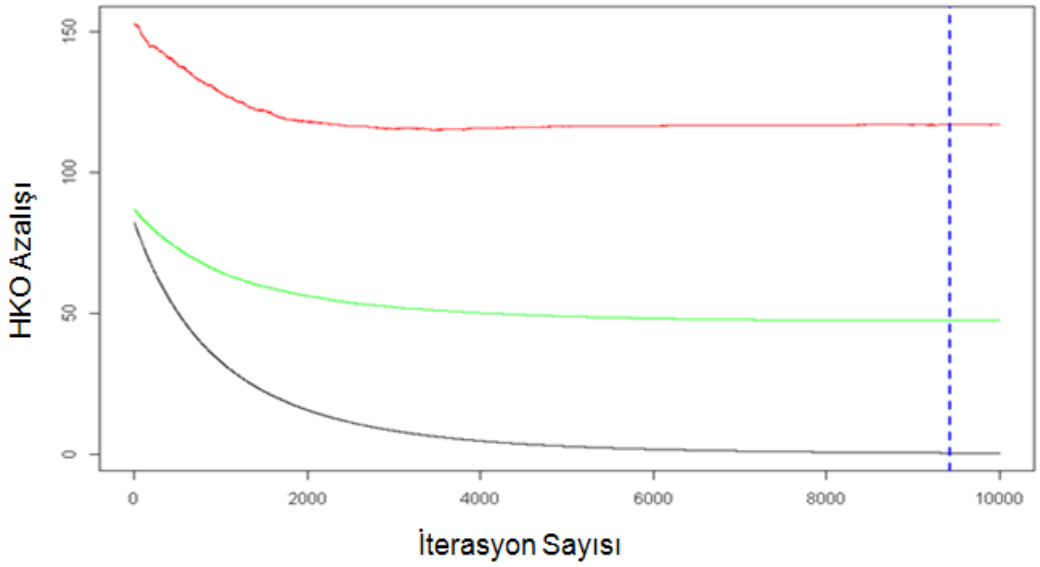
En küçük hata kestirimini sağlayan iterasyon sayısını belirlemek için best.iter() komutunun yanı sıra görsel değerlendirme de dikkate alınmıştır. Aşağıda prostat kanseri veri seti ile gds1429 gen verisi için gbm modellemesinde best.iter() fonksiyonu yardımıyla hesaplanan en uygun iterasyon (ağaç sayısı) gösterilmiştir (Şekil 4.12-4.13).



Şekil 4.12. Prostat kanseri veri seti için gbm'de en iyi iterasyon sayısı.

Grafikte siyah çizgi, eğitim verisi performansını, yeşil çizgi eğitim verisine seçilmeyen örneklemin (out-of-sample) performansını ve kırmızı çizgi ise çapraz geçerlik doğrulama performansını belirtmektedir.

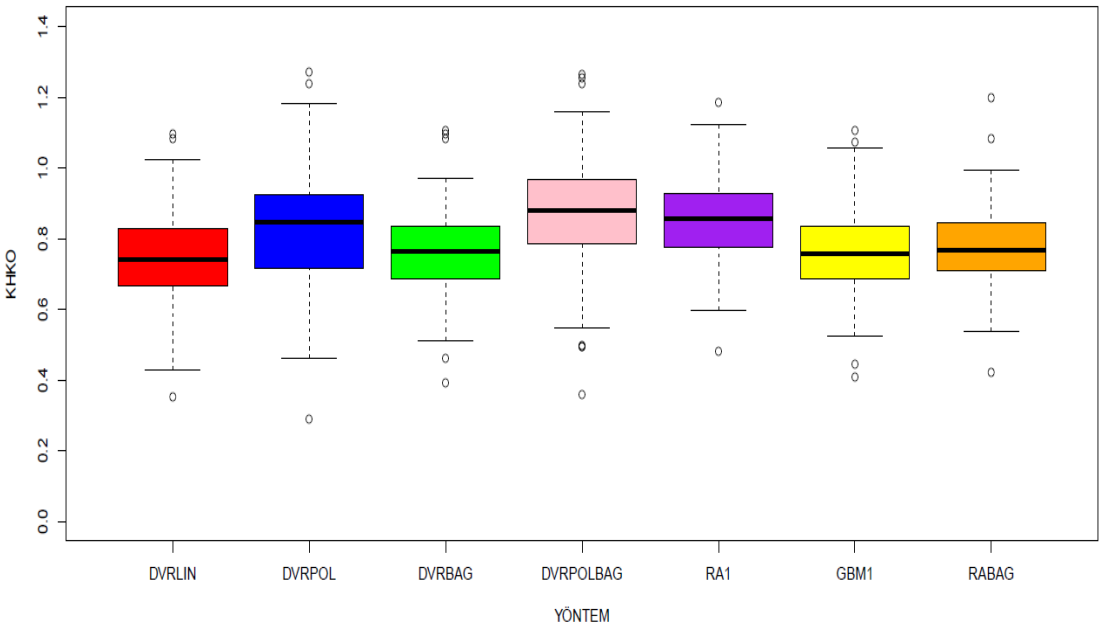
Prostat kanseri veri setinde (n=97, p=2) en iyi iterasyon 1610 olarak belirlenmiştir. Bu değer eğitim ve çapraz geçerlik performansları için hata kareler ortalamasındaki değişimin en az olduğu ağaç sayısını ifade etmektedir. Gds1429 gen verisinde ise (n=69, p=100) en iyi iterasyon 9426 olarak tespit edilmiştir.



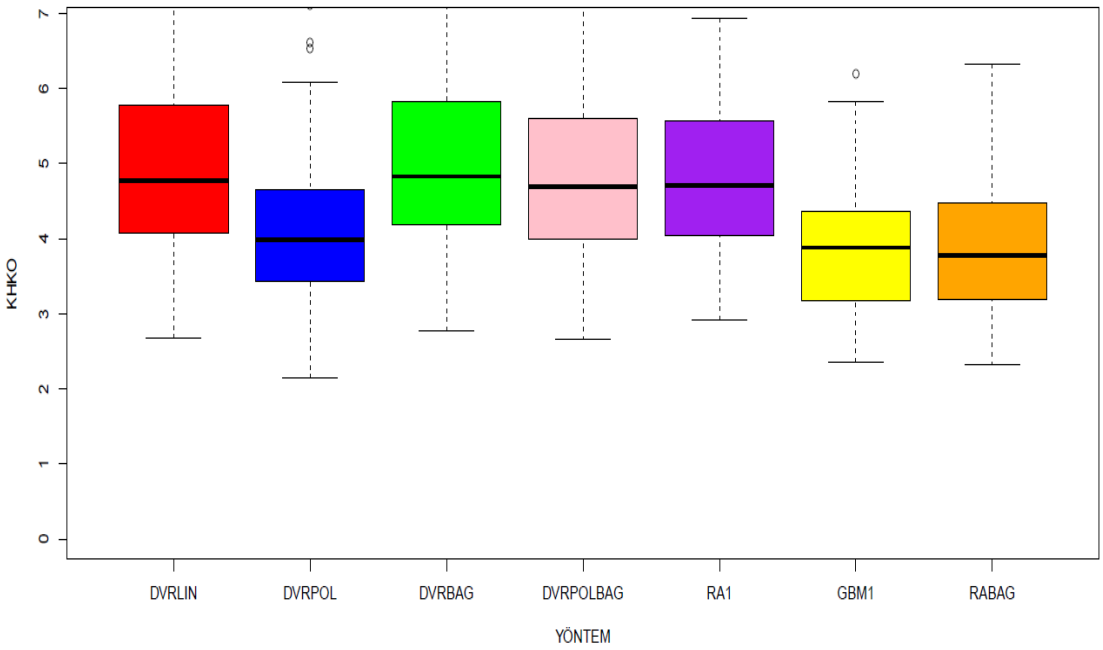
Şekil 4.13. gds1429 ID kodlu gen verisi için gbm'de en iyi iterasyon sayısı.

Tüm veri setlerinden elde edilen performanslar kutu-çizgi grafiği olarak çizdirilmiştir; ancak 13 verinin hepsi yerine yalnızca 3 veri seti için elde edilen grafikler aşağıda yer almaktadır (Şekil 4.14-Şekil 4.16).

RA yönteminin model birleştirmesine dayalı çözümlerinden elde edilen KHKO değerlerinin genellikle azaldığı görülmektedir. Bunun yanı sıra RA'ya kıyasla gbm ve bagging-RA'ya ilişkin KHKO kestirimlerinin açıklık ve standart hatalarının da azaldığı açıktır (Tablo 4.10- Şekil 4.14). Ancak DVR için elde edilen performanslar göz önüne alınırsa bagging'in hem KHKO değerini hem de standart hatasını iyileştirmediği söylenebilir.

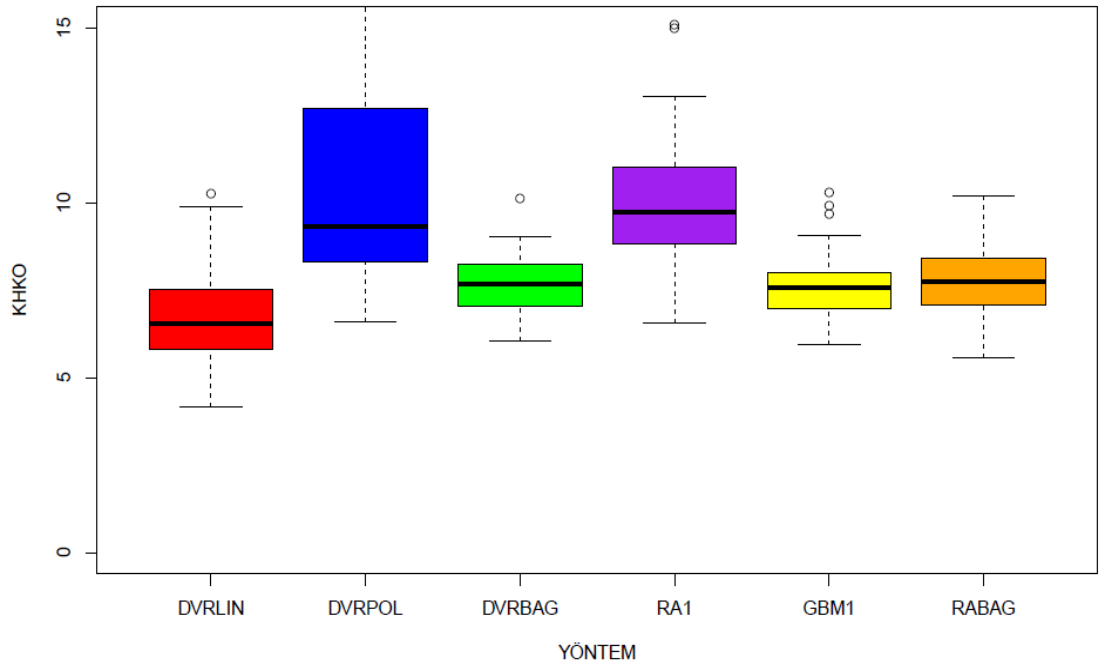


Şekil 4.14. Prostat kanseri veri seti için KHKO performansı.



Şekil 4.15. Boston Housing veri seti için KHKO performansı.

Her üç veri seti için regresyon performansı açısından elde edilen sonuçlar birbiri ile tutarlı bulunmuştur. Regresyon ağacı yöntemi için bagging ve boosting yöntemlerindeki performans artmakta iken DVR yöntemi için bagging, yöntemin tekli çözümü üzerinde gelişme sağlamamıştır.



Şekil 4.16. gds1429 ID kodlu gen veri seti için KHKO performansı.

Bu üç veri setinin yanı sıra kullanılan diğer 10 veri setinin hepsinde bagging için çekilen bootstrap tekrar sayısı 25 alınmış ve 7 farklı modelin hepsi 100 benzetim tekrarında ve farklı oranlarda eğitim ve test setlerine bölünerek sonuçlar bulunmuştur. 13 veri setinden 8'inde DVR yöntemindeki polinomiyal fonksiyon doğrusal fonksiyon kadar veriyi iyi modelleyememiştir.

RA yöntemi için elde edilen standart hata değerleri karşılaştırıldığında veri setlerinin büyük kısmında hem gbm hem de bagging-RA için elde edilen KHKO değerlerinin standart hatalarının RA'dan elde edilen standart hataya göre daha küçük olduğu anlaşılmaktadır.

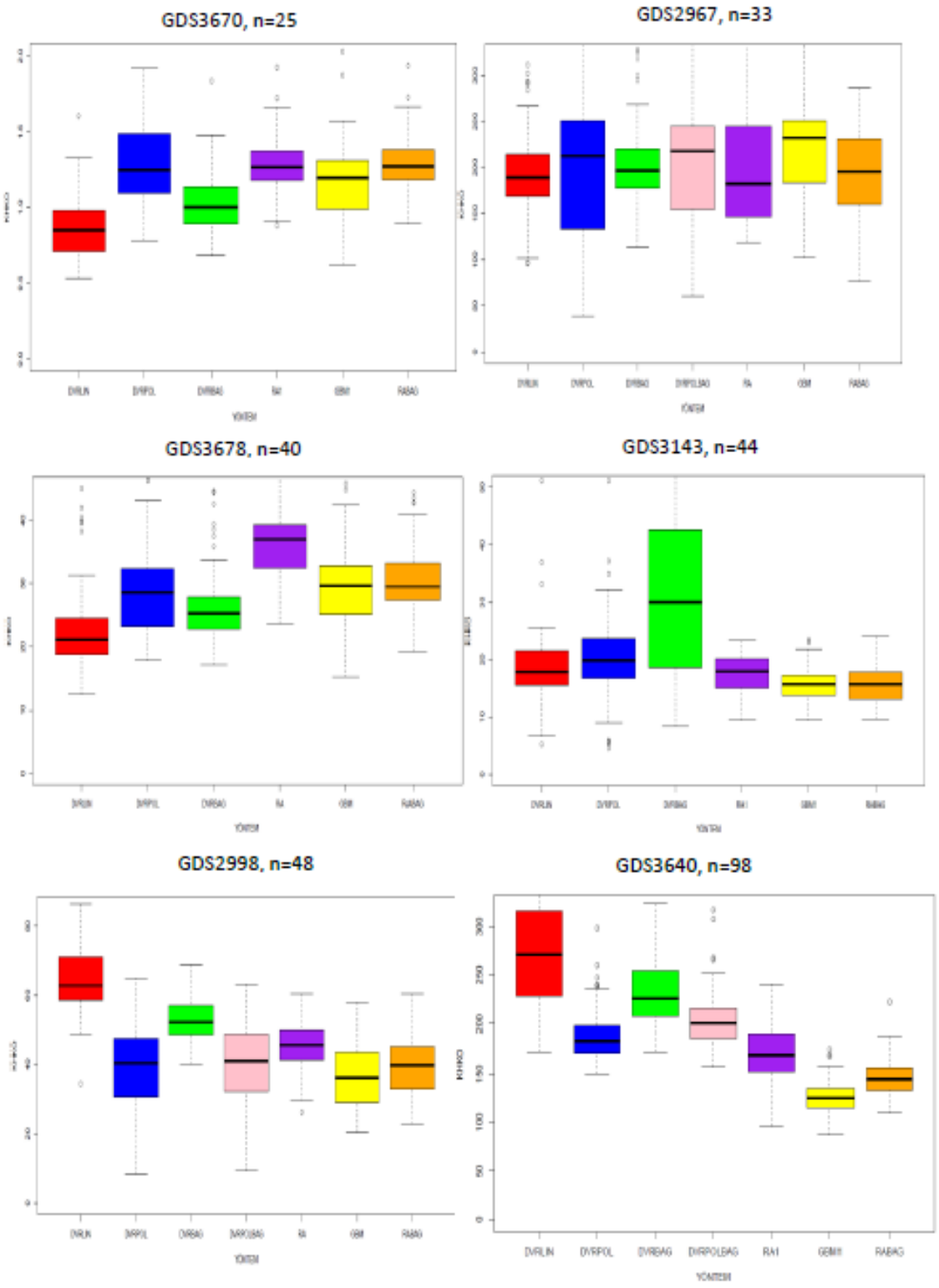
Tüm veri setleri gözönüne alınarak bagging'in hangi durumlarda KHKO değerlerinde azalma sağladığını görmek için sonuçlar kategorize edilmiştir. Bagging yönteminin RA yöntemi üzerinde gözlem sayılarına göre KHKO kestirimi açısından sağladığı yüzde değişimler Tablo 4.13'te özetlenmiştir.

Tablo 4.13. 11 adet gen ifade verisi için 100 tekrarlı RA ve RA-bagging için KHKO değişimi.

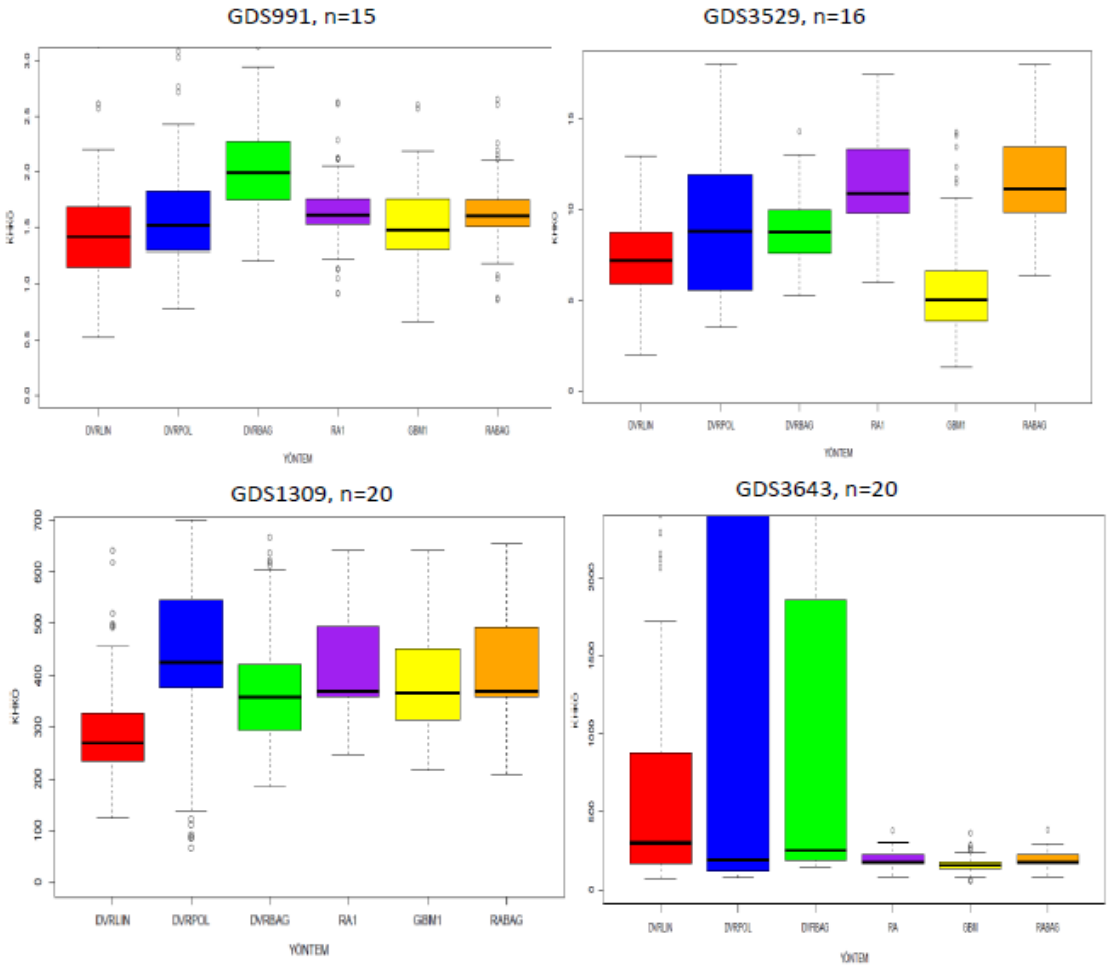
VeriSeti/Yöntem	n	$KHKO_{RA}$	$KHKO_{RA\text{-}Bagging}$	Değişim (%)
Gds991	15	1.712	1.714	0.11 artma
Gds3529	16	12.183	12.186	0.02 artma
Gds1309	20	410.410	411.949	0.37 artma
Gds3643	20	194.966	195.117	0.07 artma
Gds3670	25	1.277	1.278	0.07 artma
Gds2967	33	223.159	209.675	6.04 azalma
Gds3678	40	37.080	31.371	15.40 azalma
Gds3143	44	17.789	15.866	10.80 azalma
Gds2998	48	45.166	40.657	9.98 azalma
Gds1429	69	10.081	8.062	20.02 azalma
Gds3640	98	178.134	147.312	17.30 azalma

Bu karşılaştırmayı yapmak için tüm veri setlerinin p=100'er açıklayıcı değişken içermesi avantaj sağlamıştır. İlk başta yanıt değişkeni olan doz değerlerinin varyansı, dağılım aralığı ve değişim katsayıları açısından sonuçlar karşılaştırılmıştır. Ancak bu parametreler ile bagging sonuçları arasında herhangi bir bağlantı kurulamamıştır.

RA'dan RA-bagging'e KHKO değişimleri örneklem genişliği ile birlikte incelendiğinde örneklem genişliği 25 olan veri setine kadar Bagging'in performansı iyileştirmediği görülmüştür (Tablo 4.13). Buna karşın n=33 olan gds2967 ID kodlu veri setinden itibaren KHKO değerleri azalma göstermeye başlamıştır ve bu azalma gözlem sayısındaki artış ile daha belirgin olmuştur. Diğer 10 veri setinin yöntem performansları için çizdirilen kutu-çizgi grafikleri örneklem genişliğine göre kategorize edilerek Şekil 4.17 ve Şekil 4.18'de gösterilmiştir.



Şekil 4.17. n≥25 olan gen verileri için model birleştirme yöntem performansları.



Şekil 4.18. n<25 olan gen verileri için model birleştirme yöntem performansları.

n<25 olan dört veri seti için mor kutular (RA kestirimi) turuncu kutularla (bagging-RA kestirimi) ya aynı hizada ya da turuncu kutular mor kutulara göre daha yüksek görülmektedir (Şekil 4.18). Oysa n≥25 olan veri setlerinin büyük kısmında turuncu kutuların mor kutulardan daha küçük ortancaya sahip olduğu açıktır (Bkz. Şekil 4.17).

4.3. Gerçek Veri Bulguları

Çalışmada kullanılan veri setleri Gen Expression Omnibus açık veri tabanından elde edilen orijinal 11 doz-gen ifade verisi, prostat kanseri veri seti (Stamey,1989) ve Boston Housing (Harrison, 1978) veri setleridir (36, 54).

Bootstrap ve çapraz geçerlik yöntemlerinin gerçek verilere uygulanışı aşağıdaki kurala göre gerçekleştirilmiştir:

I. Bootstrap'da çekilen bootstrap tekrarı (b=1,2,…,20) B=20 olarak ayarlanmıştır. Bu B tane bootstrap eğitim setinin her biri üzerinde RA ve DVR modelleri kurulmuştur. Daha sonra her bir B tekrarı için KHKO, OMS ve R^2 performans ölçüleri hesaplatılarak son adımda ise bunların ortalamaları alınmıştır. Dolayısıyla bu 20 Bootstrap örneğine ilişkin ortalama performans ölçüsü tek bir benzetim adımından elde edilmiştir.

II. Çapraz geçerlik yönteminde standart olarak k=3 olarak alınmıştır. Veri setinin bölündüğü 3 parçanın her seferinde 2 parçası ile model oluşturulmuş ve kalan bir parçası ile oluşturulan model test edilmiştir. Ardından her parçada elde edilen kestirimler için ortalama performans ölçüsüleri hesaplatılmış ve daha sonra bu 3 adımdaki ölçülerin ortalaması alınmıştır.

Bu aşamalar uygulanmadan önce gen verilerinin analize hazır konuma getirilmesi bir takım işlemler sonucunda mümkün olmuştur. İlk olarak gen verileri, Orange programının Bioinformatics bölümünden GEO Data Sets bağlantısından tespit edilmiştir. Daha sonra "Gene Selection" kısmındaki birçok skorlama yöntemi içinden Foldchange kullanılarak her veri seti için en önemli 100'er gen seçilmiştir.

Daha sonra eksik verisi olan genler için eğer eksik gözlem sayısı az ise bunlara gen ölçümlerinin ortalaması atama yapılmıştır. Eksik gözlem sayısının çok olduğu durumda bu genler yerine gözlemleri tam olan başka genler veri setine dahil edilmiştir.

Gen verilerinden elde edilen yöntem sonuçları Tablo 4.14'de verilmiştir.

Tablo 4.14. Gerçek gen verileri üzerinde genelleştirme yöntemlerinin KHKO performansı.

	EĞİTİM SETİ HATASI		**GENELLEŞTİRME YÖNTEMLERİ**					
VERİ SETİ	**DVR**	**RA**	***DVR-B***	***DVR-CG***	***DVR-BDB***	***RA-B***	***RA-CG***	***RA-BDB***
gds991	0.161	1.581	*0.9*	1.6	1.2	*1.6*	1.8	1.7
gds3143	13.435	12.720	24.9	25.4	*17.9*	*14.8*	15.4	16.3
gds1429	0.906	4.856	8.7	7.0	*5.9*	12.6	10.2	9.8
gds3640	21.122	101.327	*162.6*	306.2	217.7	*129.0*	186.0	197.3
gds1309	39.149	264.327	*196.2*	295.9	221.8	*321.7*	397.2	411.3
gds3670	0.122	0.842	*0.6*	0.8	0.8	*1.0*	1.2	1.4
gds3529	1.128	10.917	*4.3*	6.9	5.8	11.14	11.08	11.6
gds2998	3.663	27.996	*35.0*	62.9	80.6	*37.1*	49.8	48.5
gds3643	18.498	96.632	1306.9	3789.2	*536.4*	*134.0*	198.8	195.7
gds2967	20.561	139.753	*129.4*	202.2	201.7	187.7	*184.1*	240.8
gds3678	2.957	22.134	*16.4*	23.1	20.7	*27.6*	32.6	38.2

Tabloda yöntemlerin eğitim seti performansları beklenildiği gibi diğer genelleştirme yöntemleri ile birlikte verdiği performanslardan çok daha iyimserdir. Bu genelleştirme yöntemleri içinde hem RA hem de DVR açısından en küçük hata kestirimini sağlayan genelleştirme yöntemi bootstrap'tir.

Prostat Kanseri veri seti ile Boston Housing veri setinin genelleştirme yöntem performansları ise Tablo 4.15 ve Tablo 4.16'da ayrı ayrı gösterilmiştir.

Tablo 4.15. Prostat kanseri verisi için genelleştirme yöntemlerinin KHKO performansları.

	DVR	**RA**	***DVR-B***	***RA-B***	***DVR-CG***	***RA-CG***
Prostat Kanseri	0.734	0.652	0.746	0.742	0.744	0.825
Verisi	***DVR-BDB***	***RA-BDB***	**DVRB-Test**	**RAB-Test**	**Hold-out-DVR**	**Hold-out-RA**
n=97, p=2	0.749	0.846	0.769	0.882	0.884	0.971

Tablo 4.16. Boston Housing veri seti için genelleştirme yöntemlerinin KHKO performansları.

	DVR	**RA**	***DVR-B***	***RA-B***	***DVR-CG***	***RA-CG***
Boston Housing	4.927	4.030	4.988	4.205	5.091	4.862
Veri seti	***DVR-BDB***	***RA-BDB***	**DVRB-Test**	**RAB-Test**	**Hold-out-DVR**	**Hold-out-RA**
n=506, p=13	5.038	4.757	5.107	4.784	5.787	5.013

13 açıklayıcı değişkene sahip Boston Housing veri seti açısından DVR için en iyi performans bootstrap ile elde edilmiştir. En büyük hata kestirimini ise hold-out yöntemi vermiştir. RA açısından ise en iyi performans yine bootstrap'ın iken en kötü sonuç hold-out ile bulunmuştur. Açıklayıcı değişken sayısının p=2 olduğu prostat kanseri veri seti dikkate alındığında, DVR'nin hold-out dışındaki hata kestirimleri birbirine çok yakındır. RA için ise en küçük hata tahminini yine bootstrap yöntemi sağlamıştır. Her iki veri seti gözönüne alındığında genelleştirme yöntemlerinden en küçük hata kestirimi veren yöntemin bootstrap ve en kötümser sonuç veren yöntemin ise hold-out olduğu söylenebilir.

Bu bulgular daha önce iki farklı benzetim düzeni ile elde edilenlere oldukça benzerdir. Bu sonuçlar yöntemlerin tekli çözümleri olduğundan burada standart hata kestirimleri verilmemiştir.

Gen verilerinden seçilen 8'i üzerinde RA ve BRA modellerinin aşırı uyumlu durumunu farklı parametreler açısından yansıtmak için aynı eğitim veri seti üzerinde test edilerek elde edilen analiz bulguları Tablo 4.17'de gösterilmiştir.

Tablo 4.17. Sekiz Gen verisine ilişkin RA ve BRA performansları.

SN	Veri Seti	Gözlem sayısı(n)	Gen Sayısı(p)	Yöntem	KHKO	R^2
1	gds630	8	100	RA	3.351	-
				BRA	2.487	0.921
2	gds991	15	100	RA	1.581	-
				BRA	1.499	0.792
3	gds3004	13	100	RA	2.170	-
				BRA	2.165	0.223
4	gds3670	25	100	RA	0.842	0.529
				BRA	0.647	0.887
5	gds3529	16	100	RA	10.916	-
				BRA	8.065	0.917
6	gds3678	40	100	RA	22.134	0.442
				BRA	10.427	0.870
7	gds3143	44	100	RA	12.719	0.300
				BRA	10.840	0.627
8	gds1429	69	100	RA	4.856	0.684
				BRA	4.587	0.782

Tablo sonucuna göre BRA yönteminin çeşitli parametre değerlerini verinin yapısına uygun olarak ayarlamak şartıyla çok küçük gözlem (hasta) sayısında bile $n<25$ RA'ya göre daha iyi performans elde edilmiştir.

Bir önceki bölümde bu yöntemlerin ayrı bir test seti ile performansları benzetim yoluyla incelenmişti. Bu bölümde gerçek uygulama olarak "gds3670" ID kodlu n=25 hastaya ilişkin p=100 genden oluşan veri seti ile "gds1429" ID kodlu n=69 hastaya ilişkin p=100 olan gen verisi için farklı parametrelerin performans değişimi gösterilmektedir.

100 ile 10000 arasında seçilen yedi ağaç sayısı, beş farklı küçültme parametresi (lr) ve 3 farklı etkileşim derecesi (1, 3 ve 5) için BRA KHKO değerleri elde edilmiştir (Tablo 4.18).

Tablo 4.18. GDS3670 Gen verisinin sırasıyla a) tc=1, b) tc=3 ve c) tc=5 alınarak elde edilen BRA performansları.

(a)

Etkileşim derecesi=1					
Ağaç sayısı	**lr=0.05**	**lr=0.01**	**lr=0.005**	**lr=0.001**	**lr=0.0005**
100	1.014	0.801	0.971	1.166	1.197
500	0.848	1.219	0.988	0.984	1.125
2000	1.201	0.812	0.834	0.825	0.904
4000	1.197	0.837	1.225	0.958	0.922
6000	0.474	0.803	0.894	0.847	0.930
8000	1.191	0.709	0.839	1.02	0.856
10000	1.205	0.742	0.786	0.886	0.901

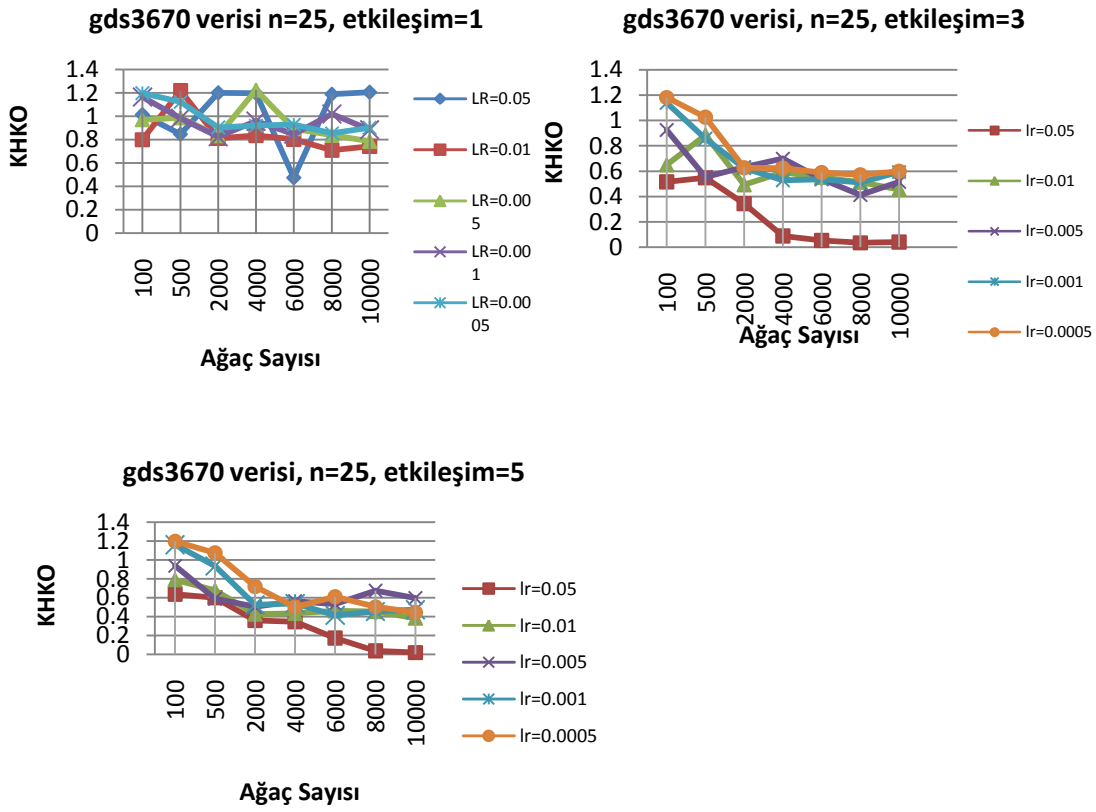
(b)

Etkileşim derecesi=3					
Ağaç sayısı	**lr=0.05**	**lr=0.01**	**lr=0.005**	**lr=0.001**	**lr=0.0005**
100	0.515	0.650	0.924	1.140	1.183
500	0.549	0.883	0.553	0.855	1.025
2000	0.344	0.492	0.631	0.624	0.627
4000	0.089	0.592	0.701	0.528	0.624
6000	0.053	0.552	0.538	0.536	0.589
8000	0.037	0.512	0.411	0.510	0.570
10000	0.04	0.456	0.515	0.595	0.599

(c)

Etkileşim derecesi=5					
Ağaç sayısı	**lr=0.05**	**lr=0.01**	**lr=0.005**	**lr=0.001**	**lr=0.0005**
100	0.638	0.792	0.941	1.163	1.197
500	0.600	0.679	0.589	0.937	1.074
2000	0.360	0.426	0.506	0.527	0.719
4000	0.345	0.434	0.563	0.540	0.501
6000	0.172	0.460	0.532	0.414	0.611
8000	0.037	0.454	0.675	0.456	0.507
10000	0.021	0.384	0.596	0.474	0.444

Sonuçlar incelendiğinde ağaç sayısı arttıkça ve aynı zamanda etkileşim derecesi arttıkça KHKO değerleri giderek azalmaktadır. Aynı öğrenme oranı için tablo sütunlarında yukarıdan aşağı inildikçe yani ağaç sayısı arttıkça KHKO değerlerinde azalma görülmektedir; ancak bu azalma etkileşim derecesi "1" iken çok belirgin değildir. Bu azalma etkileşim derecesinin artması ile birlikte daha belirgin olmaktadır (Şekil 4.19).



Şekil 4.19. gds3670 gen verisinin farklı parametrelere göre BRA performansı.

Son olarak n= 69 gözlemli gds1429 gen veri seti için elde edilen performans sonuçları Tablo 4.19'da ve ilgili grafiği ise Şekil 4.20'de verilmiştir. Tablo ve şekiller incelendiğinde her iki veri seti için KHKO kestirimlerinde en hızlı azalış lr= 0.05 değerinde gerçekleşmiştir.

BRA için ulaşılan bulgular p=100 gene karşılık n= 25 ve 69 gözlem sayısı üzerinden elde edilmiştir. Elith ve diğerlerinin (22) bulgularından farklı olarak her öğrenme oranı (lr) değeri için yavaş yavaş azalan bir grafik görünümü elde edilmiştir.

Tablo 4.19. GDS1429 Gen verisinin a) tc=1, b) tc=3 ve c) tc=5 için BRA performans sonuçları.

(a)

Etkileşim derecesi=1					
Ağaç sayısı	lr=0.05	lr=0.01	lr=0.005	lr=0.001	lr=0.0005
100	5.767	7.241	8.179	9.229	9.372
500	5.618	6.029	6.030	8.192	8.816
2000	5.509	5.946	5.750	6.368	7.231
4000	4.843	6.224	5.818	6.118	6.068
6000	4.246	6.074	5.944	5.969	6.092
8000	3.418	5.856	6.068	6.051	6.015
10000	3.209	5.666	5.963	5.882	5.856

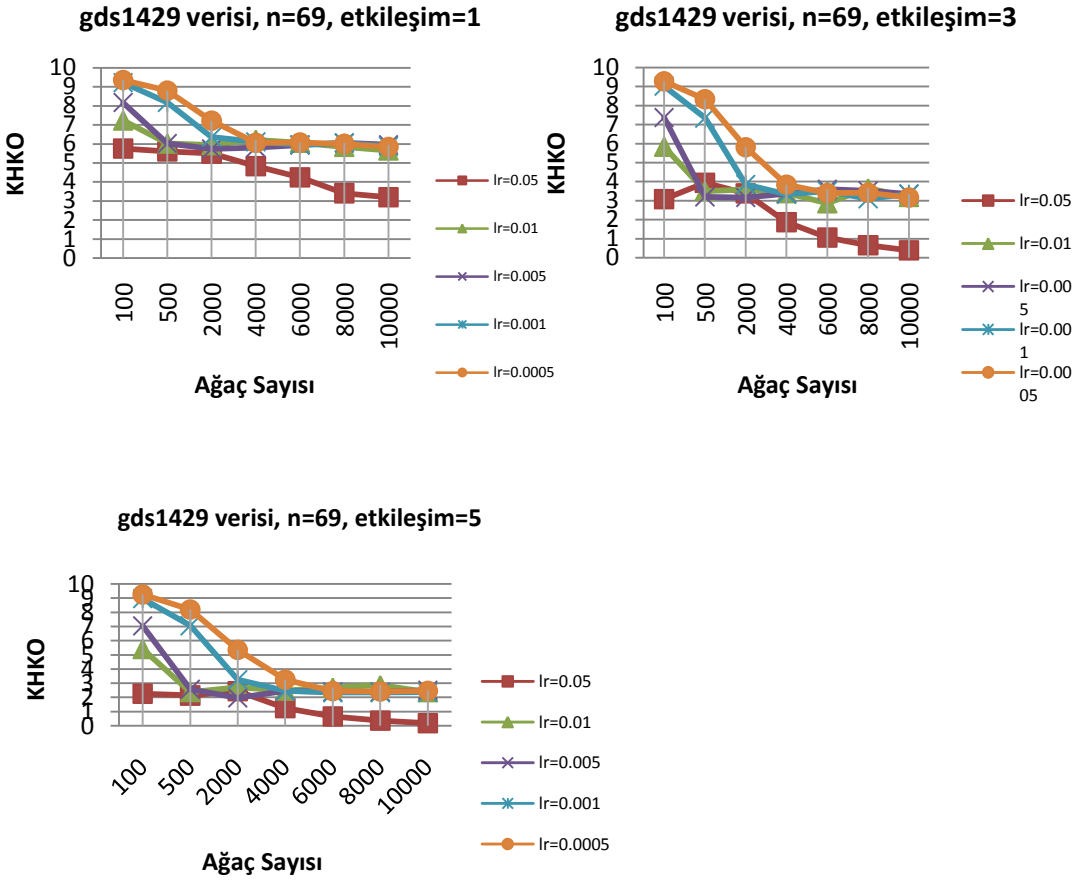
(b)

Etkileşim derecesi=3					
Ağaç sayısı	lr=0.05	lr=0.01	lr=0.005	lr=0.001	lr=0.0005
100	3.094	5.852	7.383	9.031	9.276
500	3.951	3.515	3.204	7.355	8.332
2000	3.397	3.603	3.171	3.860	5.813
4000	1.879	3.406	3.368	3.329	3.841
6000	1.072	2.871	3.604	3.500	3.397
8000	0.667	3.656	3.528	3.098	3.417
10000	0.392	3.206	3.357	3.349	3.166

(c)

Etkileşim derecesi=5					
Ağaç sayısı	lr=0.05	lr=0.01	lr=0.005	lr=0.001	lr=0.0005
100	2.271	5.422	7.049	8.961	9.243
500	2.139	2.393	2.584	7.059	8.192
2000	2.444	2.72	1.976	3.255	5.349
4000	1.250	2.494	2.474	2.462	3.252
6000	0.658	2.693	2.398	2.342	2.442
8000	0.357	2.870	2.403	2.350	2.422
10000	0.184	2.385	2.511	2.360	2.466

Ağaç sayısı arttıkça KHKO değerlerinde görülen azalma aşırı uyum olsa da olmasa da beklenen bir bulgudur. Etkileşim derecesi yani model karmaşıklığında 1'den 5'e doğru artış olduğunda genel olarak hata kestirimindeki düşüş aşırı uyumun bir sonucudur. Tablo 4.19 ve Şekil 4.20'de beklenenden farklı olan bir bulgu ise küçültme parametresi yani herbir ağacın katkısı (lr≤1) azaldıkça daha büyük hata kestiriminin elde edilmesidir.



Şekil 4.20. gds1429 gen verisinin farklı parametrelere göre BRA performansı.

5. TARTIŞMA

Literatürdeki çalışmalar incelendiğinde DVR, karar ağacı regresyonu gibi makine öğrenme yöntemlerinin çapraz geçerlik, bootstrap ve birini dışarda bırakma gibi genelleştirme yöntemleri ile birlikte kullanımları çalışmaların büyük kısmında *sınıflandırma* amaçlı gerçekleştirilmiştir. Dolayısıyla tez çalışması ele alınan temel regresyon yöntemleri ile yeniden örnekleme ya da veriyi bölmeye dayalı genelleştirme yöntemleri ve model birleştirme (ensemble) yöntemlerinin mikrodizilim verileri üzerinde performans karşılaştırmasını sağlamaktadır.

Çalışmanın bu bölümünde, daha önce yapılmış çalışmaların bulguları ile bu çalışmanın bulguları karşılaştırmalı olarak tartışılmıştır.

Luzheng ve diğerleri (55), gece görüş sistemleri ile yaya algılama olasılığını modellemek için DVR'unu kullanmışlar ayrıca, DVR'yi yapay sinir ağları, doğrusal regresyon ve polinomiyal regresyon yöntemleri ile karşılaştırmışlardır. Çalışmada ayrıca söz konusu yöntemlerin genelleştirme performansları k parçalı çapraz geçerlik yöntemi ile de değerlendirilmiştir. Sonuç olarak eğitim performansı açısından en iyi kestirimi yapay sinir ağları yöntemi vermiş, genelleştirme performansları dikkate alındığında ise DVR yöntemi diğer yöntemlere göre daha küçük hata kestirimi sağlamıştır. Luzheng ve diğerlerinin (55) yaptıkları bu çalışmada kullandıkları 3 veri setinin analizlerini LIBSVM paketi ile çözümlemişler ancak benzetim çalışması yapmamışlardır. Çalışmada ayrıca genelleştirme yöntemlerinin farklı parametrelere göre performans değişimlerine değinilmemiştir. Bu çalışma kapsamında yapılan benzetim çalışmasında ise en iyi eğitim hatası kestirimini onların çalışmasına benzer şekilde DVR sağlamıştır. Genelleştirme hatası kestirimleri karşılaştırıldığında ise en iyi sonucu BDBCG-DVR yöntemi gerçekleştirmiştir.

Tsujitani ve Tanaka (56), çalışmalarında destek vektör *sınıflandırma* makinesi için yeniden örnekleme yöntemlerinin bir uygulamasını dikkate almıştır. DVM için olabilirlik yöntemine dayalı istatistiksel çıkarsama tartışılmıştır. Kestirimdeki aşırı hata yanılgısını (bias) tahmin etmek ve parametre ayarlamasını belirlemek için birini dışarda bırakma yöntemi önerilmiştir. Bootstrap en uygun ayarlama parametreleri ile tüm uyum iyiliğinin değerlendirilmesinde kullanılmıştır. Uskumru yumurtası anketi ve karaciğer hastalığı çalışmalarından alınan veriler yeniden örnekleme yöntemlerini değerlendirmek amacıyla kullanılmıştır. Sonuç olarak hem eğitim örneklemi hem de test örneklemi için DVM'nin hata oranı diğer tüm modeller arasında en küçük kestirimi sağlamıştır.

Pers ve diğerleri (57), çalışmalarında 99 kişiye ait 8525 metabolik açıklayıcı değişkene sahip yüksek boyutlu veri seti üzerinde yağ oksidasyonu kapasitesini kestirmeyi amaçlamışlardır. Temel kestirim yöntemi olarak DVR, LASSO ve random forest yöntemleri kullanılmış ayrıca "*bootstrap-cross validation*" yöntemi ile tüm kestirim yöntemlerinin genelleştirme performansları da karşılaştırılmıştır. Uygulamada 99 kişilik örneklemden yerine koymadan 80 genişlikli 100 bootstrap alt örneklemi çekilmiştir. Her bir kestirim yöntemi (DVR, LASSO ve random forest) b'inci bootstrap alt örneklemi ile modellenmiş ve b'inci bootstrap alt örnekleminde bulunmayan 19 gözlemin kestirici matrisi ile test edilmiştir. Sonuç olarak bootstrap-cross validation yöntemi negatif yanlı ve modellerin tüm gözlemler üzerindeki performanslarına göre daha kötümser sonuçlar vermiştir. Pers ve diğerleri (57), bir sonraki çalışmada genelleştirme performansını kestirmek için alt örneklemlerin uygun büyüklüğü ya da çapraz geçerlikteki uygun adım sayısının belirlenebileceğini belirtmişlerdir. Bu tez çalışmasında ise bootstrap alt örneklem genişliği ve çapraz geçerlik parça sayısına göre kestirimlerdeki değişimler detaylı olarak incelenmiştir.

Rao ve diğerleri (17) çalışmalarında, yüksek boyutlu veri setlerinde sınıflandırma performansını genelleştirmek için kullanılan çapraz geçerlik ve birini dışarda bırakma-çapraz geçerlik yöntemlerini incelemişlerdir. Sonuç olarak ÇG ve BDBCG yöntemlerinin yüksek boyutlu veri setlerinde aşırı uyum gösterdiğini ve örneklem genişliği azaldıkça ve verinin boyutu arttıkça bu aşırı uyumun daha da kötüleştiğini ifade etmişlerdir. BDBCG yönteminin gerçek hatayı düşük kestirdiği ancak elde edilen hata varyansının ise oldukça büyük olduğunu belirterek böyle bir sınıflandırıcının sonuç olarak güvenilmez kestirimler üreteceğini vurgulamışlardır. Bu çalışmada ise hem DVR hem de RA ile elde edilen genelleştirme performanslarında; KHKO'nın standart hata kestirimleri 3 parçalı çapraz geçerlik yöntemi ve BDBCG yöntemlerinde hemen hemen birbirine yakın bulunmuştur. Bu durum test veri setindeki gözlem sayısının azalmasına da (örneğin BDBCG'de 1 gözlem) bağlıdır.

Drucker ve diğerleri (7) çalışmalarında, EKK regresyonu, ileriye yönelik altküme seçimi, DVR ve bagging yöntemlerini dört doğrusal olmayan veri seti üzerinde karşılaştırmıştır. Bu veri setlerinin hemen hemen hepsinde DVR, hem EKK yönteminden hem de Bagging ile elde edilen performansından daha iyi sonuç vermiştir. Çalışmada yüksek boyutlu veri yapısı kapsam dışında olmasına rağmen özellikle girdi uzayının boyutu ve kestirim derecesi, özellik uzayının boyutunu örnek sayısından daha büyük bir uzayda temsil eder ve oluşturur ise DVR'nin büyük bir kullanıma sahip olduğu vurgulanmıştır.

Bagging'in DVR üzerindeki etkisi için; Chen ve Hall (58) benzetim sonuçlarında bagging'in hem varyansı hem de hata kare ortalamasını arttırdığını belirtmişlerdir. Braga ve diğerleri (5) ise bagging'in özellikle doğrusal regresyon, çok katmanlı algılayıcı sinir ağları ve regresyon ağaçları gibi kararsız yöntemleri geliştirdiğini ancak DVR gibi kararlı yöntemlerde performansı arttırmadığı sonucuna ulaşmışlardır. Çalışmada ayrıca bagging'in regresyon ağaçları ile kullanımının yapay sinir ağları ya da DVR ile kullanımına göre anlaşılması çok daha kolay olduğu vurgulanmıştır.

Tez çalışmasında karşılaştırma olarak temel alınan iki çalışma Molinaro ve diğerleri (8) ile Borra ve Ciaccio'nun (59) makaleleridir.

Molinaro ve diğerleri (8) çalışmalarında, Lymphoma and lung **(**lenf ve akciğer kanseri) mikrodizilim veri setleri ile çok değişkenli normal dağılımdan türettikleri yine yüksek boyutlu benzetim verileri üzerinde; *doğrusal ayırma analizi, en yakın komşuluk, diyagonal ayırma sınıflandırıcıları, sınıflandırma ve regresyon ağaçları (CART) gibi* sınıflandırma yöntemlerinin yeniden örneklemeye dayalı (bootstrap, BDBCG, k-parçalı çapraz geçerlik gibi) çeşitli kestiriciler yardımıyla performanslarını karşılaştırmıştır. Eğitim veri setlerinin genişlikleri n= 40, 80 ve 120 olarak, test seti oranları p = 0.5, p = 0.2, p = 0.1 olarak ve Monte Carlo tekrar sayıları ise 20, 50 ve 1000 alınarak sonuçlar değerlendirilmiştir. Sonuç olarak küçük örneklem genişlikleri için 2 parçalı çapraz geçerlik ve örneklem bölme (split sample) yöntemi zayıf performans göstermiştir. BDBCG yöntemi yanlılık ve hata kare ortalaması açısından iyi bir performansa sahip olmakla birlikte 10 parçalı CG yöntemi daha düşük varyans kestirimi verdiği için BDBCG'ye tercih edilebilir bulunmuştur. Çalışmanın sınıflandırma analizine dayalı olması tez çalışmasından farklı olan yönüdür. Tez çalışmasında çapraz geçerlik açısından parça sayısı arttıkça KHKO kestirim performansının iyileşmesi ise onların çalışmasıyla benzer bir bulgudur.

Borra ve Ciaccio (59), detaylı bir benzetim yaklaşımıyla, farklı örneklem genişliklerinde BDBCG, parametrik ve parametrik olmayan *bootstrap*, çapraz geçerlik gibi yeniden örneklemeye dayalı kestiricileri dikkate alarak r*egresyon ağaçları, izdüşüm regresyonu ve* yapay *sinir ağları* modellerinin kestirim hatalarını karşılaştırmıştır.

Çalışmada modellerin eğitildiği örneklem genişlikleri 120 ve 500 olarak, sinyal-gürültü oranları ise 1, 2.5 ve 5 olarak alınmıştır. Tekrarlı düzeltilmiş 10 parçalı çapraz geçerlik yöntemi diğer kestiricilerin üstünde performans sergilemiştir. Örneklem genişliği büyük olduğunda ve kararlı yöntemler ile kullanıldığında en iyi sonuç veren kestirici parametrik bootstrap yöntemidir.

Molinaro ve diğerlerinin (8) sonuçlarına benzer şekilde BDBCG yöntemi ile yetersiz sonuçlara ulaşılmıştır. BDBCG yöntemi, oldukça yansız sonuçlar verir gibi görünse de hesaplama zorluğu ve yüksek değişkenliğe sahip olması olumsuz yanları olarak gösterilmiştir.

Bugüne kadar yapılan çalışmalara bakıldığında, DVR ve RA yöntemlerinin çeşitli genelleştirme yöntemlerine göre kestirim performanslarının karşılaştırılmadığı tespit edilmiştir. Ayrıca bu genelleştirme yöntemlerinin bootstrap tekrar sayısı, çapraz geçerlik parça sayısı gibi kendi özel parametrelerine göre oluşturduğu değişimleri inceleyen çalışmalar da oldukça azdır.

Ele alınan genelleştirme yöntemlerinden çapraz geçerlik yönteminden önceki çalışmaların büyük kısmında sınıflandırma amacıyla sadece değişken seçimi ya da yöntemlere ilişkin en uygun parametrelerin seçimi gibi amaçlar için yararlanılmıştır. Bu çalışmada ise yöntemlerin az sayıda hastaya sahip gen verileri için eğitim seti üzerindeki performansları ile yeniden örnekleme ya da veriyi bölmeye dayalı olarak elde edilen genelleştirme performansları arasındaki farklılıklar ortaya konulmuştur.

Tez çalışmasında yapılan benzetim senaryolarının planlanması, gerekli kodların yazılması, deneme tekrarları, hataların belirlenip giderilerek senaryoların tamamlanması yaklaşık 14 ay sürmüştür.

Özellikle Bootstrap ve birini dışarda bırakma yöntemlerine ilişkin kodların çalışma süresinin uzun olması çalışmanın kısıtlılıkları arasında sayılabilir. Modele bağlı ilk senaryo ile genler arası korelasyona bağlı olarak gerçekleştirilen ikinci senaryonun bulguları hem birbirine hem de gerçek veri bulgularına uyumlu elde edilmiştir.

Bu çalışma kapsamında kullanılan veri setleri yüksek boyutlu mikrodizilim gen ifadesi ile bağımlı değişken olarak da doz ve metabolik sendrom gibi nicel ölçümleri içeren açık erişimli gerçek veri setleridir. İncelenen diğer prostat kanseri ve boston housing veri setleri ise Breiman (4), Stamey (36) ve Cutler (31) gibi araştırmacıların bagging, boosting gibi analizleri gerçekleştirdikleri bilinen veri setleridir.

Breiman (60), Boston Housing veri seti için elde ettiği sonuçlarda RA yönteminin HKO kestirimini 20, bagging RA için ise 11.6 bulmuştur. Bu çalışmada ise aynı veri setini kullanarak RA için HKO değeri 25.80 ve Bagging- RA için ise 16.71 olarak elde edilmiştir. Sonuçlar alınan eğitim ve test oranları ile RA'nın budama kriterlerindeki farklılıktan dolayı aynı değildir ancak oldukça paraleldir.

Boston Housing veri setini Drucker ve diğerleri (7), EKK, DVR, RA ve bagging'in performanslarını karşılaştırmak amacıyla da kullanmıştır. Ancak Drucker ve diğerleri (7), Boston Housing veri seti için DVR'nin HKO kestirimini 7.2 olarak ve Bagging-DVR'nin kestirim hatasını ise 12.4 olarak bulmuştur. Bu çalışmada ise aynı veri seti için DVR kestirim hatası 24.41, Bagging DVR kestirim hatası ise 25.11 olarak elde edilmiştir. Onların çalışmasında her benzetim adımında test veri seti 25 büyüklüğünde alınmış, tez çalışmasında yazılmış olan algoritma kodunda ise her adımdaki test seti oranı örneklem genişliği yeterince büyük veri setleri için % 10 olarak belirlenmiştir.

Sonuç olarak bu çalışmada DVR'ye göre bagging-DVR'den elde edilen hata artışı % 2.86 iken Drucker ve diğerlerinin (7) saptamış olduğu hata farkı % 72'dir. Drucker ve diğerleri (7), toplam 100 denemenin 71'inde DVR'nin bagging ile kullanımından daha iyi performans gösterdiğini açıklamıştır. Tez çalışmasında ise bunların 11 tanesi az sayıda gözleme sahip gen-ifade verisi olmak üzere toplam 13 veri setinden 11'inde (% 84.6) DVR, bagging ile kullanımından daha iyi model kestirimi sağlamıştır.

Bu sonuçlara ek olarak kullanılan veri setlerinin büyük kısmı için bagging'in 100 tekrardaki standart hata kestirimleri yöntemlerin tekli performansına kıyasla daha küçük elde edilmiştir. Bu sonuçlar ise Breiman'ın (60) bagging'in yanlılığı azaltamayan ancak varyansı azaltabilen bir yöntem olduğu şeklindeki önerisi ile uyumludur.

Bu çalışmada farklı olarak yöntemlerin model birleştirme yöntemleri ile performansları kategorize edilerek bir kural belirlenmiştir.

Eğitim verisinde bulunan gözlem sayısı n<25 olan veri setleri için RA-Bagging'in RA performansını iyileştirmediği görülmüştür. Gözlem sayısı n≥25 olan veri setleri için ise RA-bagging'in kullanımıyla hem KHKO hem de KHKO standart hata kestirimleri azalmıştır. Buna rağmen boosting regresyon ağaçları (gbm) ise 8'den 98'e değişen örneklem genişliğine sahip her veri seti için RA'dan daha iyi performans göstermiştir. Dolayısıyla sonuçlara göre bagging yönteminin yüksek boyutlu verilerde n≥25 gibi bir kısıtlılığı varken boosting yönteminin herhangi bir kısıtlaması bulunmamaktadır.

Elith ve diğerleri (28), ekolojik veriler üzerinde uyguladıkları boosting regresyon (gbm) yöntemi için farklı parametreler açısından ve örneklem genişliği açısından değişimleri incelemişlerdir. Çalışmalarında 13369 siteden topladıkları ekolojik veriler üzerinde gbm yöntemini uygulamışlardır. Model kurulumu için genellikle 1000 site kullanmış ve kalan veri ile model değerlendirmesi yapmışlardır. Tüm modeller için 12 açıklayıcı değişken kullanmışlardır. Çalışmada ele aldıkları belirli aralıktaki ağaç sayıları, etkileşim dereceleri ve öğrenme oranları için örneklem genişliği arttıkça hata performanslarında hızlı bir düşüş gözlenmiştir. Ayrıca her etkileşim derecesi tc= 2, 4, 6, 8, 10 değeri ile, küçültme parametresi (learning rate=0.05, 0.01, 0.005, 0.001, 0.0005) değerlerinin aynı ağaç sayısında aynı hatayı vermesi için aralarındaki ilişkinin ters orantılı olduğundan bahsedilmiştir.

Tez çalışmasında ise n=25 ve n=69 hastaya ait gen verileri üzerinde aşırı uyumlu durumu yansıtmak için aynı eğitim verisi üzerinde kestirim yapılarak farklı parametreler ile gbm oluşturulmuştur. 100-10000 arasında seçilen yedi ağaç sayısı, beş farklı küçültme parametresi (lr) ve 3 farklı etkileşim derecesi (tc) için BRT'nin KHKO değerleri elde edilmiştir. Sabit bir küçültme oranı için ağaç sayısı arttıkça KHKO değerlerinde azalma olduğu belirlenmiştir. Ancak bu azalmanın oranı etkileşim derecesine de bağlıdır. Sonuçlar incelendiğinde ağaç sayısı arttıkça ve aynı zamanda etkileşim derecesi arttıkça KHKO değerleri giderek azalmaktadır.

Elith ve diğerlerinin (28), farklı öğrenme oranları ve artan ağaç sayısı için çizdirmiş olduğu grafik daha çok hiperbolik bir görünüm vermiştir. Bu çalışmada ise giderek azalma gösteren bir grafik elde edilmiştir. Özellikle seçilen ağaç sayısı 4000 ve üstüne çıktığında hem n=25 gözlem hem de n=69 gözlem için en küçük KHKO değerleri lr=0.05 parametresi ile bulunmuştur. Diğer çalışmada 12 açıklayıcı değişkene karşı 1000 gözlem ile model oluşturulurken tez çalışmasında 100 gene karşılık en az 8 ve en fazla 98 arasında değişen veri setleri kullanılmıştır.

6. SONUÇ VE ÖNERİLER

Bu tez çalışması çeşitli genelleştirme yöntemlerinin özellikle yüksek boyutlu mikrodizilim verileri için çoklu regresyonda kapsamlı bir karşılaştırmasını sunmaktadır. Çalışmada iki farklı benzetim düzeninin yanı sıra, GEO'dan ulaşılan 11 mikrodizilim gen ifadesi verisi ile de modeller test edilmiştir. Çünkü orijinal veri setlerinin kullanımı yeni yöntemlerin test edilmesinde en çok tercih edilen yaklaşımlardandır.

Eğitim performansı açısından DVR yöntemi RA yönteminden daha iyi sonuç vermektedir. Genelleştirme performansı açısından 11 orijinal veri setinden 8'inde bootstrap yöntemi daha küçük hata kestirimi sağlamıştır. P=2 açıklayıcı değişken içeren prostat kanseri verisinde DVR için *çapraz geçerlik*, RA için ise *bootstrap* daha iyi performans vermiştir. Benzetim senaryosu I ve II'de DVR için en iyimser sonuç BDBCG ile elde edilirken, RA yöntemi için en iyimser sonuç bootstrap genelleştirme yöntemi ile bulunmuştur. Ayrıca bu genelleştirme yöntemlerinden en küçük standart hatalı kestirimler *bootstrap* yöntemi ile elde edilmiştir. Bunun temel nedeni ise modelin test edildiği veri setinin yeterli sayıda örneklem içermesidir. Örneğin BDBCG yönteminde her adımda 1 veri seti üzerinde model test edildiği için varyans kestirimi artmıştır. Benzer şekilde çapraz geçerlik yönteminde de verinin belli bir parçası test seti olarak kullanıldığı için varyans Bootstrap'a kıyasla daha büyük bulunmaktadır.

Regresyon ağaçlarına budama yapmanın etkisi yöntemin performansını genelleştirme yöntemlerinden Bootstrap ile çapraz geçerlik arasına taşımaktadır. Her iki benzetim sonucuna göre DVR fonksiyonlarından ilk sırada doğrusal fonksiyon ikinci sırada radyal tabanlı fonksiyon iyi performans göstermiştir. Örneklem genişliğinin artması DVR'nin genelleştirme hatasında azalmaya sebep olmuş ancak RA'ın performansını etkilememiştir.

Tam aksine genler arası korelasyonun artması DVR'nun genelleştirme hatası sonuçlarını etkilememiş ancak RA hatası korelasyon ile orantılı olarak artış göstermiştir. Çalışmada genelleştirme yöntemlerinin kendi özel parametrelerine göre değişimler de incelenmiş ve bootstrap tekrar sayısının özellikle de 100 tekrardan sonra KHKO değerinde ciddi değişim oluşturmadığı gözlenmiştir. Çapraz geçerlikte ise k "parça sayısı" 3'ten 10'a yükseldikçe KHKO değerlerinde azalma belirlenmiştir. Ancak bu azalma benzetim sonuçlarına göre n=100 ya da n= 500 alındığında gerçekleşmiştir. N=20 iken KHKO performansında parça sayısına (k) göre iyileşme bulunmamıştır. Gözlem sayısı yeterli büyüklükte iken çapraz geçerlik yönteminin k=3, 5 ve 10 parça sayıları arttıkça hatada yavaş bir azalma olmasına rağmen en küçük hata kestirimini BDBCG yöntemi sağlamıştır.

Sonuç olarak inceleme konusu olan veri, mikrodizilim gen ifade verileri gibi çok sayıda değişken (gen) içeren ancak az hastaya sahip veri setleri ise, araştırmacı bu yapıdaki verilerle regresyon modellemesi yaparken dikkat etmelidir. Genelleştirme performansı daha yüksek bir sonuç elde edilmek isteniyor ise DVR yöntemi tercih edilmelidir. Çünkü çalışmada DVR'nin genelleştirme yöntemlerinin her türü ile hata kestirimi daha küçük bulunmuştur.

Bir regresyon yönteminin dikkate alınan veri seti üzerindeki (tek model olarak) çözümü yerine kestirim performansını arttırmada dikkate alınacak bir seçenek ise birden fazla modeli birleştirerek daha güçlü bir kestirici elde etmektir. Bu durumda araştırmacı, regresyon ağacı yönteminin performansını geliştirmek amacıyla gözlem sayısının değişken sayısına göre yetersiz olduğu veri setleri üzerinde özellikle n<25 iken RA-bagging yöntemi yerine RA-boosting'i tercih edebilir. Sonuç olarak RA-boosting'in gözlem sayısının çok küçük olduğu veri setlerinde bile iyi kestirim sağlayan bir yöntem olduğu söylenebilir.

KAYNAKLAR

1. Segal, M.R., Dahlquist, K.D. ve Conklin, B.R. (2003). Regression approaches for microarray data analysis. *Journal of Computational Biology,* 10, 961-980.

2. Coşgun, E., Karaağaoğlu, E. (2011). Veri Madenciliği Yöntemleri ile Mikrodizilim Gen İfade Analizi [*Microarray gene expression data analysis with data mining methods*]. *Hacettepe Tıp Dergisi*, 42, 180-189.

3. Efron, B. ve Tibshirani, R. (1993). *An Introduction to the bootstrap*. London: Chapman and Hall.

4. Breiman, L. (1996). Bagging predictors. *Machine Learning,* 24, 123–140.

5. Braga, P.L., Oliveira, A. L. I., Ribeiro, G.H.T. ve Meira, S. R. L., (2007). Bagging predictors for estimation of software project effort. *International Joint Conference on Neural Networks: IJCNN 2007*: 12-17 August 2007-USA: Bildiriler (s.1595-1600).

6. Buhlmann, P. ve Hothorn, T. (2007). Boosting algorithms: regularization, prediction and model fitting. *Statistical Science*, 22 (4), 477-505.

7. Drucker, H., Burges, C.J.C., Kaufman, L., Smola, A. ve Vapnik, V. (1997). Support vector regression machines. *Advances in Neural Information Processing Systems*, 9, 155-161.

8. Molinaro, A.M., Simon, R. ve Pfeiffer R.M. (*2005).* Prediction error estimation: a comparison of resampling methods. *Bioinformatics,* 21(15), 3301-3307*.*

9. Karabulut, E. ve Karaağaoğlu, E., (2010). Biyoinformatik ve Biyoistatistik [Bioinformatics and Biostatistics]. *Hacettepe Tıp Dergisi*, 41, 162-170.

10. Klein, T.E, Altman, R.B, Eriksson, N., Gage, B.F, Kimmel, S.E, Lee M.T. ve diğerleri. (2009). Estimation of the warfarin dose with clinical and pharmacogenetic data. *The New England Journal Of Medicine*, 360, 753-764.

11. Boulesteix, A.L., Strobl, C., Augustin, T. ve Daumer, M. (2008). Evaluating microarray-based classifiers: An overview. *Cancer Informatics*, 6, 77-97.

12. Çomak, E. (2008). ***Destek Vektör Makinelerinin Etkin Eğitimi için Yeni Yaklaşımlar***. Doktora Tezi, Selçuk Üniversitesi, Konya.

13. Ye, N. (2003). *The Handbook of Data Mining*. London: Lawrence Erlbaum Associates, Inc.

14. Hsu, C.-W., Chang, C.-C. ve Lin, C.-J. (t.y.). A practical guide to support vector classification. Erişim: 10.09.2013, **http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.**

15. Freedman, L.S. ve Pee, D. (1989). Return to a note on screening regression equations. *The American Statistician*, 43(4), 279-282.

16. Hawkins, D.M., (2004). The problem of overfitting. *Journal of Chemical Information and Computer Sciences*, 44 (1), 1-12.

17. Rao, R.B., Fung, G. ve Rosales, R. (April 2008). On the dangers of cross-validation. overfitting and overestimation of performance: An experimental evaluation. In Proceedings of the SIAM International Conference on Data Mining (SDM), Georgia.

18. Vapnik, V., Golowich, S. ve Smola, A. (1996). Support vector method for function approximation, regression estimation, and signal processing. *Advances in Neural Information Processing Systems*, 9, 281–287.

19. Song, M., Breneman, C.M., Bi, J., Sukumar, N., Bennett, K.P., Cramer, S.M., ve diğerleri. (2002). Prediction of protein retention times in anion-exchange chromatography systems using support vector regression. *Journal of Chemical Information and Computer Sciences,* 42(6), 1347-1357.

20. Kim, H.-C., Pang, S., Je, H.-M, Kim, D., Bang, S.Y. (2003). Constructing support vector machine ensemble. *Pattern Recognition*. 36(12), 2757-2767.

21. Nalbantov, G., Groenen, P.J.F. ve Bioch, J.C. (2005). Support vector regression basics. *Medium Econometrische Toepassingen*, 13(1),16-19.

22. Gunn, S.R. (1998). Support vector machines for classification and regression. London: University of Southampton, Faculty of Engineering, Science and Mathematics School of Electronics and Computer Science, **http://users.ecs.soton.ac.uk/srg/publications/.**

23. Basak, D., Pal, S., Patranabis, D.C. (2007). Support vector regression. *Neural Information Processing – Letters and Reviews*, 11 (10), 203-224.

24. Lahiri, S.K. ve Ghanta, K.C. (2008). The Support vector regression with the parameter tuning assisted by a differential evolution technique: study of the critical velocity of a slurry flow in a pipeline. *Chemical Industry & Chemical Engineering Quarterly,* 14(3), 191-203.

25. Garcia, M.A. (2005). Study of Support vector machines and comparison with the multi-layer perceptron for 3D nonlinear data regression. Yüksek lisans tezi, University of Rovira Virgili, Automatical and Industrial Electronic Engineering, Spain.

26. DTREG Software, svm- support vector machines. (t.y.). Erişim: 15.05.2014, **http://www.dtreg.com/svm.htm.**

27. Chang, C.-C. ve Lin C.-J. (2001). LIBSVM: a library for support vector machines. Erişim: 20.04.2014, **http://www.csie.ntu.edu.tw/~cjlin/ libsvm.**

28. Elith, J., Leathwick, J.R, Hastie, T. (2008). A working guide to boosted regression trees. *Journal of Animal Ecology*, 77(4), 802-813.

29. Sutton, C.D. (2005). Classification and regression trees, bagging and boosting. *Handbook Of Statistics, Data Mining and Data Visualization*, 24, 303-329.

30. Hastie, T., Tibshirani, R. ve Friedman, J. (2008). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Newyork: Springer.

31. Cutler, A., Cutler, D.R., and Stevens J.R. (2009). Tree-based methods. Li, X. ve Xu, R. (Ed.) *High-Dimensional Data Analysis in Cancer Research, Applied Bioinformatics and Biostatistics in Cancer Research series*. Newyork: Springer.

32. Breiman, L., Friedman, J., Olshen, R., ve Stone, C. (1984). *Classification and Regression Trees*. California: Wadsworth, Belmont.

33. Grubinger, T., Kobel, C. ve Pfeiffer, K.P. (2010). Regression tree construction by bootstrap: Model search for DRG-systems applied to Austrian health-data. *BMC Medical Informatics and Decision Making,* 10 (9),1-11.

34. Prasad, A.M., Iverson, L.R. ve Liaw, A. (2006). Newer classification and regression tree techniques: bagging and random forests for ecological prediction. *Ecosystems,* 9, 181–199*.*

35. Hastie, T., Tibshirani, R. ve Friedman, J. (2001). *The Elements of Statistical Learning*. Newyork: Springer.

36. Stamey, T., Kabalin, J.N., McNeal, J.E., Johnstone, I.M., Freiha, F., Redwine, E.A. ve diğerleri. (1989). Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate. II. Radical prostatectomy treated patients. *Journal of Urology*, 16**,** 1076-1083.

37. Sexton, J. ve Laake, P. (2007). Boosted regression trees with errors in variables. *Biometrics*, 63(2), 586-592.

38. Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *International Joint Conference on Artificial Intelligence*, 2, 1137–1145.

39. Leisch F. (2002). Sweave part I: Resampling methods in R: boot package. *R News*. December, 2(3), 2-10.

40. Davison, A. C. ve Kuonen, D. (2002). An introduction to the bootstrap with applications in R. *Statistical Computing and Statistical Graphics Newsletter*, 13 (1), 6-11.

41. SAS (R) Enterprise Miner (TM) 12.1 extension nodes: Developer's Guide. (t.y.). Erişim:02.09.2013,**http://support.sas.com/documentation/cdl/en/emxndg/65358/HTML/default/viewer.htm#n002icfvzhfd57n1c2pmqx1ygovl.htm.**

42. Han, J., Kamber M. (2006). *Data Mining Concepts and Techniques Details of Book: Data Mining: Concepts And Techniques*. USA: Elsevier.

43. Coşgun, E., Limdi, N.A. ve Duarte C.W. (2011). High dimensional pharmacogenetic prediction of a continuous trait using machine learning techniques with application to warfarin dose prediction in African American. *Bioinformatics*, 27(10), 1384-1389.

44. Buhlmann, P. (2004). Bagging, boosting and ensemble methods. Gentle, J., Härdle, W. ve Mori, Y. (Ed.). *Handbook of Computational Statistics: Concepts and Methods* (s. 877-907). Newyork: Springer.

45. Buhlmann, P. ve Bin Y. (2002). Analyzing bagging. *The Annals of Statistics.* 4, 927-961.

46. Drucker, H. (1997). Improving regressors using boosting techniques. In *Proceedings of the Fourteenth International Conference on Machine Learning* Morgan-Kaufman, San Mateo, CA, 107–115.

47. Ridgeway, G. (1999). The state of boosting. *Computing Science and Statistics*. 31, 172-181.

48. Ridgeway, G. (2007). gbm: Generalized boosted regression models*. R package* version 1.6-3. Erişim: 10.05.2014, **http://cran.uvigo.es/ web/ packages/gbm/gbm.pdf.**

49. Ciaccio, A.D., Borra, S., (2002). Improving nonparametric regression methods by bagging and boosting. *Computational Statistics & Data Analysis*, 4, 407-420.

50. Friedman, J.H. (2002). Stochastic gradient boosting. *Computational Statistics and Data Analysis*. 38(4), 367-378.

51. Fridley, B.L., Jenkins, G.D. ve Biernacka, J.M. (2010). Self-contained gene-set analysis of expression data: An evaluation of existing and novel methods. *PLoS ONE,* 5(9), 1-9.

52. Burton, A., Altman, D.G., Royston, P. ve Holder, R.L. (2006). The design of simulation studies in medical statistics. *Statistics in Medicine*, 25, 4279–4292.

53. Schaefer, J., Opgen-R. R., Zuber V., Ahdesmaki, M., Silva, A.P.D. ve Strimmer K. (2006). The corpcor package: Efficient estimation of covariance and correlation. Erişim: 10.05.2014, **http://cran.r-project. org/web/packages/corpcor/index.html.**

54. Harrison, D. ve Rubinfeld, D.L. (1978). Hedonic prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5, 81-102.

55. Luzheng, Bi, Tsimhoni, O. Yili, Liu. (2011). Using the support vector regression approach to model human performance. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 41(3), 410-417.

56. Tsujitani, M. ve Tanaka, Y., (2011). Cross-Validation, bootstrap, and support vector machines. *Advances in Artificial Neural Systems*, 2011, 1-6.

57. Pers, T.H, Albrechtsen, A., Holst, C., Sørensen, T.I.A., Gerds, T.A. (2009). The validation and assessment of machine learning: A game of prediction from high- dimensional data. *PLoS ONE,* 4(8), 1-8.

58. Chen, S. X. and Hall P. (2003). Effects of bagging and bias correction on estimators defined by estimating equations. *Statistica Sinica,* 13 (1), 97-110.

59. Borra S., Ciaccio A.D., (2010). Measuring the prediction error. A comparison of cross-validation, bootstrap and covariance penalty methods. *Computational Statistics and Data Analysis*, 54 (12), 2976-2989.

60. Breiman, L. (2001). Using iterated bagging to debias regressions. *Machine Learning*, 45(3), 261-277.

EKLER

EK 1. Modele bağlı gerçekleştirilen benzetim senaryosu 1'de yöntemlerin regresyon performanslarının elde edilmesi için R programında yazılan algoritma.

```
library(e1071)
library(rpart)
library(MASS)
#set parameters
n.iter<-1000
n<-100
p<-100
setwd('c:\\')
cor2 <- matrix(scan('cor10003.txt'),p,p)
beta=rep(1,p)
hko.svmlin=numeric(n.iter)
mad.svmlin=numeric(n.iter)
r2.svmlin=numeric(n.iter)
hko.svmpol=numeric(n.iter)
mad.svmpol=numeric(n.iter)
r2.svmpol=numeric(n.iter)
hko.svmsig=numeric(n.iter)
mad.svmsig=numeric(n.iter)
r2.svmsig=numeric(n.iter)
hko.svmrad=numeric(n.iter)
mad.svmrad=numeric(n.iter)
r2.svmrad=numeric(n.iter)
hko.rt=numeric(n.iter)
mad.rt=numeric(n.iter)
r2.rt=numeric(n.iter)
SVLIN=numeric(n.iter)
SVPOL=numeric(n.iter)
SVSIG=numeric(n.iter)
SVRAD=numeric(n.iter)
estimates<-data.frame(matrix(NA,nrow=n.iter,ncol=20))
colnames(estimates, do.NULL = FALSE)
colnames(estimates) <-
c("hko.svmlin","mad.svmlin","r2.svmlin","hko.svmpol","mad.svmpol","r2.svmpol","hko.svmsig"
,"mad.svmsig","r2.svmsig","hko.svmrad","mad.svmrad","r2.svmrad","hko.rt","mad.rt","r2.rt","s
v.lin","sv.pol","sv.sig","sv.rad","TER.NOD")
for(i in 1:n.iter){
 #generate data
x = mvrnorm(n, rep(0, 100), cor2)
eps<-rnorm(n,m=0,sd=1)
y=(x%*%beta)+eps
data=data.frame(y,x)
#estimate things
mod1<-svm(y~.,data=data,type="eps-regression",kernel="linear"
```

EK 1 (Devam)

```
mod2<-svm(y~.,data=data,type="eps-regression",kernel="polynomial")
mod3<-svm(y~.,data=data,type="eps-regression",kernel="sigmoid")
mod4<-svm(y~.,data=data,type="eps-regression",kernel="radial")
mod5<-rpart(y~.,data=data)
aa<-summary(mod5)
predict1<-predict(mod1,data)
predict2<-predict(mod2,data)
predict3<-predict(mod3,data)
predict4<-predict(mod4,data)
predict5<-predict(mod5,data)
hko.svmlin[i]=sum((data$y-predict1)^2)/n
mad.svmlin[i]=sum(abs(data$y-predict1))/n
r2.svmlin[i]=(cor(data$y,predict1))^2
hko.svmpol[i]=sum((data$y-predict2)^2)/n
mad.svmpol[i]=sum(abs(data$y-predict2))/n
r2.svmpol[i]=(cor(data$y,predict2))^2
hko.svmsig[i]=sum((data$y-predict3)^2)/n
mad.svmsig[i]=sum(abs(data$y-predict3))/n
r2.svmsig[i]=(cor(data$y,predict3))^2
hko.svmrad[i]=sum((data$y-predict4)^2)/n
mad.svmrad[i]=sum(abs(data$y-predict4))/n
r2.svmrad[i]=(cor(data$y,predict4))^2
hko.rt[i]=sum((data$y-predict5)^2)/n
mad.rt[i]=sum(abs(data$y-predict5))/n
r2.rt[i]=(cor(data$y,predict5))^2
SVLIN[i]=length(mod1$coefs)
SVPOL[i]=length(mod2$coefs)
SVSIG[i]=length(mod3$coefs)
SVRAD[i]=length(mod4$coefs)
TERNOD[i]=length(aa[[5]][,1])
estimates[i,1]<-hko.svmlin[i]
estimates[i,2]<-mad.svmlin[i]
estimates[i,3]<-r2.svmlin[i]
estimates[i,4]<-hko.svmpol[i]
estimates[i,5]<-mad.svmpol[i]
estimates[i,6]<-r2.svmpol[i]
estimates[i,7]<-hko.svmsig[i]
estimates[i,8]<-mad.svmsig[i]
estimates[i,9]<-r2.svmsig[i]
estimates[i,10]<-hko.svmrad[i]
estimates[i,11]<-mad.svmrad[i]
estimates[i,12]<-r2.svmrad[i]
estimates[i,13]<-hko.rt[i]
estimates[i,14]<-mad.rt[i]
estimates[i,15]<-r2.rt[i]
estimates[i,16]<-SVLIN[i]
estimates[i,17]<-SVPOL[i]
estimates[i,18]<-SVSIG[i]  }
```

EK 2. Modele bağlı gerçekleştirilen benzetim senaryosu 1'de yöntemlerin Bootstrap genelleştirme performanslarının elde edilmesi için R programında yazılan algoritma

```
library(e1071)
library(rpart)
library(bootstrap)
library(MASS)
n.iter<-1000
n<-100
p<-100
setwd('c:\\')
cor2 <- matrix(scan('cor10003.txt'),p,p)
beta=rep(1,p)
bootstrap=function(data,nboot){
hko.svmlin=numeric(nboot)
mad.svmlin=numeric(nboot)
r2.svmlin=numeric(nboot)
hko.svmpol=numeric(nboot)
mad.svmpol=numeric(nboot)
r2.svmpol=numeric(nboot)
hko.svmsig=numeric(nboot)
mad.svmsig=numeric(nboot)
r2.svmsig=numeric(nboot)
hko.svmrad=numeric(nboot)
mad.svmrad=numeric(nboot)
r2.svmrad=numeric(nboot)
hko.rt=numeric(nboot)
mad.rt=numeric(nboot)
r2.rt=numeric(nboot)
for (i in 1:nboot){
bootindex=sample(1:n,n,replace=T)
mod1<-svm(y~.,data=data[bootindex,],type="eps-regression",kernel="linear")
mod2<-svm(y~.,data=data[bootindex,],type="eps-regression",kernel="polynomial")
mod3<-svm(y~.,data=data[bootindex,],type="eps-regression",kernel="sigmoid")
mod4<-svm(y~.,data=data[bootindex,],type="eps-regression",kernel="radial")
mod5<-rpart(y~.,data=data[bootindex,])
predict1<-predict(mod1,data)
predict2<-predict(mod2,data)
predict3<-predict(mod3,data)
predict4<-predict(mod4,data)
predict5<-predict(mod5,data)
hko.svmlin[i]=sum((data[,1]-predict1)^2)/n
mad.svmlin[i]=sum(abs(data[,1]-predict1))/n
r2.svmlin[i]=(cor(data[,1],predict1))^2
hko.svmpol[i]=sum((data[,1]-predict2)^2)/n
mad.svmpol[i]=sum(abs(data[,1]-predict2))/n
r2.svmpol[i]=(cor(data[,1],predict2))^2
```

EK 2(Devam)

```
hko.svmsig[i]=sum((data[,1]-predict3)^2)/n
mad.svmsig[i]=sum(abs(data[,1]-predict3))/n
r2.svmsig[i]=(cor(data[,1],predict3))^2
hko.svmrad[i]=sum((data[,1]-predict4)^2)/n
mad.svmrad[i]=sum(abs(data[,1]-predict4))/n
r2.svmrad[i]=(cor(data[,1],predict4))^2
hko.rt[i]=sum((data[,1]-predict5)^2)/n
mad.rt[i]=sum(abs(data[,1]-predict5))/n
r2.rt[i]=(cor(data[,1],predict5))^2
}
hko.svmlin1=mean(hko.svmlin)
mad.svmlin1=mean(mad.svmlin)
r2.svmlin1=mean(r2.svmlin)
hko.svmpol1=mean(hko.svmpol)
mad.svmpol1=mean(mad.svmpol)
r2.svmpol1=mean(r2.svmpol)
hko.svmsig1=mean(hko.svmsig)
mad.svmsig1=mean(mad.svmsig)
r2.svmsig1=mean(r2.svmsig)
hko.svmrad1=mean(hko.svmrad)
mad.svmrad1=mean(mad.svmrad)
r2.svmrad1=mean(r2.svmrad)
hko.rt1=mean(hko.rt)
mad.rt1=mean(mad.rt)
r2.rt1=mean(r2.rt)
list(hko.svmlin1=hko.svmlin1,mad.svmlin1=mad.svmlin1,r2.svmlin1=r2.svmlin1,hko.svmpol1
=hko.svmpol1,mad.svmpol1=mad.svmpol1,r2.svmpol1=r2.svmpol1,hko.svmsig1=hko.svmsig
1,mad.svmsig1=mad.svmsig1,r2.svmsig1=r2.svmsig1,hko.svmrad1=hko.svmrad1,mad.svmr
ad1=mad.svmrad1,r2.svmrad1=r2.svmrad1,hko.rt1=hko.rt1,mad.rt1=mad.rt1,r2.rt1=r2.rt1)
}
boot = 20
sonuc<-data.frame(matrix(NA,n.iter,ncol=15))
colnames(sonuc) =
c("hko.svmlin","mad.svmlin","r2.svmlin","hko.svmpol","mad.svmpol","r2.svmpol","hko.svmsig"
,"mad.svmsig","r2.svmsig","hko.svmrad","mad.svmrad","r2.svmrad","hko.rt","mad.rt","r2.rt")
w = list()
for(i in 1:n.iter){
 #generate data
x = mvrnorm(n, rep(0, 100), cor2)
eps<-rnorm(n,m=0,sd=50)
y=(x%*%beta)+eps
data1=data.frame(y,x)
w[[i]] = bootstrap(data1,boot)
sonuc[i,1]<-w[[i]]$hko.svmlin1
sonuc[i,2]<-w[[i]]$mad.svmlin1
sonuc[i,3]<-w[[i]]$r2.svmlin1
sonuc[i,4]<-w[[i]]$hko.svmpol1
sonuc[i,5]<-w[[i]]$mad.svmpol1
```

EK 2 (Devam)

```
sonuc[i,6]<-w[[i]]$r2.svmpol1
sonuc[i,7]<-w[[i]]$hko.svmsig1
sonuc[i,8]<-w[[i]]$mad.svmsig1
sonuc[i,9]<-w[[i]]$r2.svmsig1
sonuc[i,10]<-w[[i]]$hko.svmrad1
sonuc[i,11]<-w[[i]]$mad.svmrad1
sonuc[i,12]<-w[[i]]$r2.svmrad1
sonuc[i,13]<-w[[i]]$hko.rt1
sonuc[i,14]<-w[[i]]$mad.rt1
sonuc[i,15]<-w[[i]]$r2.rt1

}

sonuc.MEAN = colMeans(sonuc)
```

EK 3. Modele bağlı gerçekleştirilen benzetim senaryosu 1'de yöntemlerin çapraz geçerlik genelleştirme performanslarının elde edilmesi için R programında yazılan algoritma

```
library(e1071)
library(rpart)
library(bootstrap)
library(MASS)
numbersim=1000
n<-100
p<-100
setwd('c:\\')
cor2 <- matrix(scan('cor10003.txt'),p,p)
beta=rep(1,p)
k<-3
crossval=function(data, k) {
fold<-sample(rep(seq_len(k), length.out=n))
if (k > n)
stop("Fold sayısı n'den büyük olamaz")
if (k < 2)
stop("Fold sayısı 2'den küçük olamaz")
hko.svmlin <- numeric(k)
mad.svmlin <- numeric(k)
r2.svmlin <- numeric(k)
hko.svmpol <- numeric(k)
mad.svmpol <- numeric(k)
r2.svmpol <- numeric(k)
hko.svmsig<- numeric(k)
mad.svmsig <- numeric(k)
r2.svmsig<-numeric(k)
hko.svmrad<-numeric(k)
mad.svmrad <- numeric(k)
r2.svmrad<-numeric(k)
hko.rt<-numeric(k)
mad.rt <- numeric(k)
r2.rt<-numeric(k)
for (i in 1:k) {
mod1<-svm(y~., data = data[fold!=i,],type="eps-regression",kernel="linear")
mod2<-svm(y~., data = data[fold!=i,],type="eps-regression",kernel="polynomial")
mod3<-svm(y~., data = data[fold!=i,],type="eps-regression",kernel="sigmoid")
mod4<-svm(y~., data = data[fold!=i,],type="eps-regression",kernel="radial")
mod5<-rpart(y~., data = data[fold!=i,])
predict1<-predict(mod1,data[fold==i,])
predict2<-predict(mod2,data[fold==i,])
predict3<-predict(mod3,data[fold==i,])
predict4<-predict(mod4,data[fold==i,])
predict5<-predict(mod5,data[fold==i,])
hko.svmlin[i]=sum((data[fold==i,1]-predict1)^2)/(n/k)
```

EK 3 (Devam)

```
mad.svmlin[i]=sum(abs(data[fold==i,1]-predict1))/(n/k)
r2.svmlin[i]=(cor(data[fold==i,1],predict1))^2
hko.svmpol[i]=sum((data[fold==i,1]-predict2)^2)/(n/k)
mad.svmpol[i]=sum(abs(data[fold==i,1]-predict2))/(n/k)
r2.svmpol[i]=(cor(data[fold==i,1],predict2))^2
hko.svmsig[i]=sum((data[fold==i,1]-predict3)^2)/(n/k)
mad.svmsig[i]=sum(abs(data[fold==i,1]-predict3))/(n/k)
r2.svmsig[i]=(cor(data[fold==i,1],predict3))^2
hko.svmrad[i]=sum((data[fold==i,1]-predict4)^2)/(n/k)
mad.svmrad[i]=sum(abs(data[fold==i,1]-predict4))/(n/k)
r2.svmrad[i]=(cor(data[fold==i,1],predict4))^2
hko.rt[i]=sum((data[fold==i,1]-predict5)^2)/(n/k)
mad.rt[i]=sum(abs(data[fold==i,1]-predict5))/(n/k)
r2.rt[i]=(cor(data[fold==i,1],predict5))^2
}
hko.svmlin1=mean(hko.svmlin)
mad.svmlin1=mean(mad.svmlin)
r2.svmlin1=mean(r2.svmlin)
hko.svmpol1=mean(hko.svmpol)
mad.svmpol1=mean(mad.svmpol)
r2.svmpol1=mean(r2.svmpol)
hko.svmsig1=mean(hko.svmsig)
mad.svmsig1=mean(mad.svmsig)
r2.svmsig1=mean(r2.svmsig)
hko.svmrad1=mean(hko.svmrad)
mad.svmrad1=mean(mad.svmrad)
r2.svmrad1=mean(r2.svmrad)
hko.rt1=mean(hko.rt)
mad.rt1=mean(mad.rt)
r2.rt1=mean(r2.rt)
list(hko.svmlin1=hko.svmlin1,mad.svmlin1=mad.svmlin1,r2.svmlin1=r2.svmlin1,hko.svmpol1
=hko.svmpol1,mad.svmpol1=mad.svmpol1,r2.svmpol1=r2.svmpol1,hko.svmsig1=hko.svmsig
1,mad.svmsig1=mad.svmsig1,r2.svmsig1=r2.svmsig1,hko.svmrad1=hko.svmrad1,mad.svmr
ad1=mad.svmrad1,r2.svmrad1=r2.svmrad1,hko.rt1=hko.rt1,mad.rt1=mad.rt1,r2.rt1=r2.rt1)
}
sonuc = data.frame(matrix(NA,numbersim,15))
colnames(sonuc) =
c("hko.svmlin","mad.svmlin","r2.svmlin","hko.svmpol","mad.svmpol","r2.svmpol","hko.svmsig"
,"mad.svmsig","r2.svmsig","hko.svmrad","mad.svmrad","r2.svmrad","hko.rt","mad.rt","r2.rt")
w = list()
for (i in 1:numbersim) {
#generate data
x = mvrnorm(n, rep(0, 100), cor2)
eps<-rnorm(n,m=0,sd=1)
y=(x%*%beta)+eps
data1=data.frame(y,x)
w[[i]] = crossval(data1,k)
sonuc[i,1]=w[[i]]$hko.svmlin1
```

EK 3 (Devam)

```
sonuc[i,2]=w[[i]]$mad.svmlin1
sonuc[i,3]=w[[i]]$r2.svmlin1
sonuc[i,4]=w[[i]]$hko.svmpol1
sonuc[i,5]=w[[i]]$mad.svmpol1
sonuc[i,6]=w[[i]]$r2.svmpol1
sonuc[i,7]=w[[i]]$hko.svmsig1
sonuc[i,8]=w[[i]]$mad.svmsig1
sonuc[i,9]=w[[i]]$r2.svmsig1
sonuc[i,10]=w[[i]]$hko.svmrad1
sonuc[i,11]=w[[i]]$mad.svmrad1
sonuc[i,12]=w[[i]]$r2.svmrad1
sonuc[i,13]=w[[i]]$hko.rt1
sonuc[i,14]=w[[i]]$mad.rt1
sonuc[i,15]=w[[i]]$r2.rt1
}
sonuc.MEAN = colMeans(sonuc)
```

EK 4. Yöntemlerin gds1429 gen verisi üzerindeki tekli ve model birleştirme yöntemlerinden bagging ve boosting için 100 tekrarda gösterdiği performansların elde edilmesi için R programında yazılan algoritma

```
library(e1071)
library(rpart)
library(gbm)
library(ipred)
n.sim=100
iterations<-25
bagging=function(data,iterations){
TERNOD=numeric(iterations)
SVLIN=numeric(iterations)
SVPOL=numeric(iterations)
TERNOD1=numeric(iterations)
SVLIN1=numeric(iterations)
SVPOL1=numeric(iterations)
hko.svmlin2=numeric(iterations)
mad.svmlin2=numeric(iterations)
r2.svmlin2=numeric(iterations)
hko.svmpol2=numeric(iterations)
mad.svmpol2=numeric(iterations)
r2.svmpol2=numeric(iterations)
hko.rt2=numeric(iterations)
mad.rt2=numeric(iterations)
r2.rt2=numeric(iterations)
hko.gbm=numeric(iterations)
mad.gbm=numeric(iterations)
r2.gbm=numeric(iterations)
hko.bag=numeric(iterations)
mad.bag=numeric(iterations)
r2.bag=numeric(iterations)
hko.svmlin=numeric(iterations)
mad.svmlin=numeric(iterations)
r2.svmlin=numeric(iterations)
hko.svmpol=numeric(iterations)
mad.svmpol=numeric(iterations)
r2.svmpol=numeric(iterations)
hko.svmsig=numeric(iterations)
mad.svmsig=numeric(iterations)
r2.svmsig=numeric(iterations)
hko.svmrad=numeric(iterations)
mad.svmrad=numeric(iterations)
r2.svmrad=numeric(iterations)
hko.rt1=numeric(iterations)
mad.rt=numeric(iterations)
r2.rt1=numeric(iterations)
```

EK 4 (Devam)

```
trainindex<-sample(nrow(data),size=(nrow(data)/100)*75)
traindata=data[trainindex,]
testdata=data[-trainindex,]
svm_fitlin1<-svm(dose~.,data=traindata,type="eps-regression",kernel="linear")
svm_fitpol1<-svm(dose~.,data=traindata,type="eps-regression",kernel="polynomial")
rt_fit1<-rpart(dose~.,data=traindata,method="anova", control=rpart.control(cp=0.01,xval=10))
c=rep(0,100)
gbm1<-gbm(dose~.,data = traindata, var.monotone=c, distribution="gaussian",
n.trees=2000, shrinkage=0.01,interaction.depth=1, bag.fraction=0.75, train.fraction=1,
n.minobsinnode=2,cv.folds=10, keep.data=TRUE,verbose=TRUE)
best.iter<-gbm.perf(gbm1,method="cv")
yy=traindata[,101]
xx=traindata[,1:100]
bag1=ipredbagg(yy, xx, nbagg=25, control=rpart.control(cp=0.01,xval=10),ns=length(yy))
aa<-summary(rt_fit1)
predlin1=predict(svm_fitlin1,testdata)
predpol1=predict(svm_fitpol1,testdata)
predrt1=predict(rt_fit1,testdata)
f.predict<-predict.gbm(gbm1, testdata, best.iter)
predbag1=predict(bag1,testdata)
for (j in 1:iterations){
bootindex <- sample(nrow(traindata), size=floor((nrow(traindata))),replace=T)
bootsamp<-traindata[bootindex,]
svm_fitlin<-svm(dose~.,data=bootsamp,type="eps-regression",kernel="linear")
svm_fitpol<-svm(dose~.,data=bootsamp,type="eps-regression",kernel="polynomial")
svm_fitsig<-svm(dose~.,data=bootsamp,type="eps-regression",kernel="sigmoid")
svm_fitrad<-svm(dose~.,data=bootsamp,type="eps-regression",kernel="radial")
rt_fit<-rpart(dose~.,data=bootsamp,method="anova", control=rpart.control(cp=0.01,xval=10))
bb<-summary(rt_fit)
predlin=predict(svm_fitlin,testdata)
predpol=predict(svm_fitpol,testdata)
predsig=predict(svm_fitsig,testdata)
predrad=predict(svm_fitrad,testdata)
predrt=predict(rt_fit,testdata)
TERNOD1[j]=length(bb[[5]][,1])
SVLIN1[j]=length(svm_fitlin$coefs)
SVPOL1[j]=length(svm_fitpol$coefs)
hko.svmlin[j]=sum((testdata$dose-predlin)^2)/nrow(testdata)
mad.svmlin[j]=sum(abs(testdata$dose-predlin))/nrow(testdata)
r2.svmlin[j]=(cor(testdata$dose,predlin))^2
hko.svmpol[j]=sum((testdata$dose-predpol)^2)/nrow(testdata)
mad.svmpol[j]=sum(abs(testdata$dose-predpol))/nrow(testdata)
r2.svmpol[j]=(cor(testdata$dose,predpol))^2
hko.svmsig[j]=sum((testdata$dose-predsig)^2)/nrow(testdata)
mad.svmsig[j]=sum(abs(testdata$dose-predsig))/nrow(testdata)
r2.svmsig[j]=(cor(testdata$dose,predsig))^2
hko.svmrad[j]=sum((testdata$dose-predrad)^2)/nrow(testdata)
mad.svmrad[j]=sum(abs(testdata$dose-predrad))/nrow(testdata)
```

EK 4 (Devam)

```
r2.svmrad[j]=(cor(testdata$dose,predrad))^2
hko.rt1[j]=sum((testdata$dose-predrt)^2)/nrow(testdata)
mad.rt[j]=sum(abs(testdata$dose-predrt))/nrow(testdata)
r2.rt1[j]=(cor(testdata$dose,predrt))^2
}
TERNOD=length(aa[[5]][,1])
SVLIN=length(svm_fitlin1$coefs)
SVPOL=length(svm_fitpol1$coefs)
TERNOD1=length(bb[[5]][,1])
hko.svmlin2=sum((testdata$dose-predlin1)^2)/nrow(testdata)
mad.svmlin2=sum(abs(testdata$dose-predlin1))/nrow(testdata)
r2.svmlin2=(cor(testdata$dose,predlin1))^2
hko.svmpol2=sum((testdata$dose-predpol1)^2)/nrow(testdata)
mad.svmpol2=sum(abs(testdata$dose-predpol1))/nrow(testdata)
r2.svmpol2=(cor(testdata$dose,predpol1))^2
hko.rt2=sum((testdata$dose-predrt1)^2)/nrow(testdata)
mad.rt2=sum(abs(testdata$dose-predrt1))/nrow(testdata)
r2.rt2=(cor(testdata$dose,predrt1))^2
hko.gbm=sum((testdata$dose-f.predict)^2)/nrow(testdata)
mad.gbm=sum(abs(testdata$dose-f.predict))/nrow(testdata)
r2.gbm=(cor(testdata$dose,f.predict))^2
hko.bag=sum((testdata$dose-predbag1)^2)/nrow(testdata)
mad.bag=sum(abs(testdata$dose-predbag1))/nrow(testdata)
r2.bag=(cor(testdata$dose,predbag1))^2
Ternod2=mean(TERNOD1)
svlin11=mean(SVLIN1)
svpol11=mean(SVPOL1)
hko.svmlin1=mean(hko.svmlin)
mad.svmlin1=mean(mad.svmlin)
r2.svmlin1=mean(r2.svmlin)
hko.svmpol1=mean(hko.svmpol)
mad.svmpol1=mean(mad.svmpol)
r2.svmpol1=mean(r2.svmpol)
hko.svmsig1=mean(hko.svmsig)
mad.svmsig1=mean(mad.svmsig)
r2.svmsig1=mean(r2.svmsig)
hko.svmrad1=mean(hko.svmrad)
mad.svmrad1=mean(mad.svmrad)
r2.svmrad1=mean(r2.svmrad)
hko.rt11=mean(hko.rt1)
mad.rt1=mean(mad.rt)
r2.rt11=mean(r2.rt1)
list(hko.svmlin2=hko.svmlin2,mad.svmlin2=mad.svmlin2,r2.svmlin2=r2.svmlin2,….
SVPOL=SVPOL,TERNOD1=TERNOD1,svlin11=svlin11,svpol11=svpol11)
}
sonuc<-data.frame(matrix(NA,n.sim,ncol=36))
```

EK 4 (Devam)

```
colnames(sonuc) =
c("hko.svmlin2","mad.svmlin2","r2.svmlin2","hko.svmpol2","mad.svmpol2","r2.svmpol2",…Te
rnod2","svlin11","svpol11")
w = list()
for (i in 1:n.sim) {
a<-read.delim("C:/gds1429.txt")
d = data.frame(a)
n<-nrow(d)
w[[i]] = bagging(d,iterations)
sonuc[i,1]=w[[i]]$hko.svmlin2
sonuc[i,2]=w[[i]]$mad.svmlin2
sonuc[i,3]=w[[i]]$r2.svmlin2
sonuc[i,4]=w[[i]]$hko.svmpol2
sonuc[i,5]=w[[i]]$mad.svmpol2
sonuc[i,6]=w[[i]]$r2.svmpol2
sonuc[i,7]=w[[i]]$hko.rt2
sonuc[i,8]=w[[i]]$mad.rt2
sonuc[i,9]=w[[i]]$r2.rt2
sonuc[i,10]=w[[i]]$hko.gbm
sonuc[i,11]=w[[i]]$mad.gbm
sonuc[i,12]=w[[i]]$r2.gbm
sonuc[i,13]=w[[i]]$hko.bag
sonuc[i,14]=w[[i]]$mad.bag
sonuc[i,15]=w[[i]]$r2.bag
sonuc[i,16]=w[[i]]$hko.svmlin1
sonuc[i,17]=w[[i]]$mad.svmlin1
sonuc[i,18]=w[[i]]$r2.svmlin1
sonuc[i,19]=w[[i]]$hko.svmpol1
sonuc[i,20]=w[[i]]$mad.svmpol1
sonuc[i,21]=w[[i]]$r2.svmpol1
sonuc[i,22]=w[[i]]$hko.svmsig1
sonuc[i,23]=w[[i]]$mad.svmsig1
sonuc[i,24]=w[[i]]$r2.svmsig1
sonuc[i,25]=w[[i]]$hko.svmrad1
sonuc[i,26]=w[[i]]$mad.svmrad1
sonuc[i,27]=w[[i]]$r2.svmrad1
sonuc[i,28]=w[[i]]$hko.rt11
sonuc[i,29]=w[[i]]$mad.rt1
sonuc[i,30]=w[[i]]$r2.rt11
sonuc[i,31]=w[[i]]$TERNOD
sonuc[i,32]=w[[i]]$SVLIN
sonuc[i,33]=w[[i]]$SVPOL
sonuc[i,34]=w[[i]]$Ternod2
sonuc[i,35]=w[[i]]$svlin11
sonuc[i,36]=w[[i]]$svpol11
}
sapply(sonuc,sd)
```

EK 5. Genlerin kendi arasında ilişkili olmadığı (ρxixj=0) ve genler ile bağımlı değişken arasında ilişki katsayılarının b=1 alındığı durum için genelleştirme yöntemlerinden elde edilen performans sonuçları.

		n=20			n=100			n=500		
σ_ε	Yöntem	KHKO	R^2	OMS	KHKO	R^2	OMS	KHKO	R^2	OMS
1	DVR	0.969	0.996	0.944	0.935	0.993	0.901	0.929	0.991	0.774
	RA	7.109	0.465	5.612	6.200	0.614	4.827	6.536	0.577	5.186
	DVR-Boot	6.039	0.654	1.767	4.058	0.839	2.511	1.351	0.982	1.067
	DVR-CG	9.914	0.237	7.894	6.365	0.606	5.061	1.267	0.984	1.011
	RA-Boot	9.078	0.280	6.760	8.729	0.331	6.227	8.711	0.319	6.435
	RA-CG	10.449	-	8.323	12.029	0.042	9.640	11.861	0.027	9.483
6	DVR	1.137	0.995	1.108	1.347	0.986	1.161	5.599	0.769	4.139
	RA	8.342	0.462	6.596	7.275	0.606	5.668	7.667	0.566	6.075
	DVR-Boot	7.139	0.634	4.081	6.589	0.692	3.852	6.630	0.692	4.861
	DVR-CG	11.827	0.220	9.401	10.961	0.273	8.715	7.618	0.601	6.076
	RA-Boot	10.557	0.276	7.881	10.272	0.320	7.329	10.256	0.303	7.573
	RA-CG	12.080	-----	9.631	14.087	0.037	11.285	13.908	0.019	11.118
20	DVR	2.200	0.995	2.144	5.336	0.945	2.736	18.793	0.297	13.616
	RA	16.308	0.450	12.910	14.464	0.579	11.275	15.167	0.535	12.017
	DVR-Boot	14.594	0.595	8.234	18.924	0.425	10.434	22.229	0.182	16.025
	DVR-CG	24.413	0.182	19.493	32.332	0.041	25.688	25.267	0.064	20.175
	RA-Boot	20.645	0.261	15.389	20.198	0.289	14.420	20.215	0.263	14.917
	RA-CG	23.492	---	18.754	27.590	0.031	22.157	27.622	0.009	22.128

EK 6. Genlerin kendi arasında ilişkili olmadığı (ρxixj=0) ve genler ile bağımlı değişken arasında ilişki katsayılarının b=2 alındığı durum için genelleştirme yöntemlerinden elde edilen performans sonuçları.

		n=20			n=100			n=500		
σ_ε	Yöntem	KHKO	R^2	OMS	KHKO	R^2	OMS	KHKO	R^2	OMS
1	DVR	1.940	0.996	1.889	1.849	0.993	1.782	1.282	0.997	1.093
	RA	14.248	0.461	11.280	12.326	0.612	9.590	13.033	0.576	10.337
	DVR-boot	11.829	0.652	3.987	7.972	0.843	4.947	1.629	0.995	1.352
	DVR-CG	19.509	0.250	15.548	12.456	0.621	9.888	1.878	0.993	1.497
	RA-Boot	18.288	0.276	13.614	17.363	0.333	12.388	17.285	0.319	12.775
	RA-CG	20.726	---	16.498	23.935	0.044	19.183	23.598	0.028	18.841
6	DVR	2.035	0.996	1.983	2.010	0.991	1.963	5.525	0.930	4.225
	RA	14.835	0.469	11.719	12.992	0.610	10.121	13.651	0.573	10.820
	DVR-Boot	12.568	0.646	7.159	9.473	0.793	5.741	6.531	0.903	4.920
	DVR-CG	20.709	0.229	16.517	15.100	0.490	12.021	7.477	0.873	5.963
	RA-Boot	18.823	0.278	14.023	18.160	0.327	12.958	18.148	0.313	13.397
	RA-CG	21.545	----	17.181	24.977	0.041	20.066	24.580	0.026	19.609
20	DVR	2.770	0.995	2.699	4.410	0.976	3.046	18.741	0.560	13.678
	RA	20.482	0.452	16.201	17.855	0.595	13.900	18.838	0.553	14.927
	DVR-Boot	18.116	0.606	10.276	19.752	0.561	11.190	22.183	0.436	16.083
	DVR-CG	29.772	0.193	23.826	33.624	0.119	26.775	25.273	0.316	20.161
	RA-Boot	26.076	0.264	19.453	25.169	0.306	17.962	24.986	0.287	18.423
	RA-CG	29.562	----	23.615	34.471	0.036	27.587	34.232	0.015	27.404

EK 7. Genlerin kendi arasında ilişkili olmadığı (ρxixj=0) ve genler ile yanıt değişkeni arasında ilişki katsayılarının b=3 alındığı durum için genelleştirme yöntemlerinden elde edilen performans sonuçları.

		n=20			n=100			n=500		
σ_ε	Yöntem	KHKO	R^2	OMS	KHKO	R^2	OMS	KHKO	R^2	OMS
1	DVR	2.889	0.996	2.814	2.782	0.993	2.678	1.911	0.998	1.624
	RA	21.105	0.469	16.672	18.628	0.611	14.532	19.516	0.576	15.473
	DVR-Boot	18.081	0.657	10.388	11.898	0.844	7.381	2.418	0.996	2.007
	DVR-CG	29.254	0.244	23.385	18.582	0.626	14.772	2.803	0.995	2.231
	RA-Boot	27.432	0.283	20.522	25.994	0.332	18.533	25.844	0.318	19.111
	RA-CG	30.986	----	24.718	35.834	0.043	28.734	35.239	0.026	28.186
50	DVR	5.670	0.995	5.532	12.350	0.956	6.858	46.991	0.357	34.081
	RA	42.205	0.444	33.354	37.271	0.585	29.103	39.484	0.541	31.261
	DVR-Boot	37.896	0.602	21.353	47.549	0.451	26.400	55.557	0.236	40.139
	DVR-CG	62.506	0.181	49.796	81.080	0.052	64.426	63.611	0.103	50.764
	RA-Boot	53.853	0.265	40.111	52.607	0.292	37.539	52.709	0.269	38.887
	RA-CG	60.209	---	48.126	71.723	0.031	57.557	72.092	0.009	57.651

EK 8. Genlerin kendi arasında düşük seviyede ilişki bulunan (ρxixj=0.10) ve genler ile yanıt değişkeni arasında ilişki katsayılarının b=1 alındığı durum için genelleştirme yöntemlerinden elde edilen performans sonuçları.

		n=20			n=100			n=500		
σ_ε	Yöntem	KHKO	R^2	OMS	KHKO	R^2	OMS	KHKO	R^2	OMS
1	DVR	3.046	0.997	2.910	2.552	0.998	2.291	1.912	0.998	1.597
	RA	20.938	0.568	16.545	16.476	0.747	12.741	17.733	0.710	14.060
	DVR-Boot	9.753	0.931	6.209	3.754	0.994	3.056	2.204	0.998	1.820
	DVR-CG	15.102	0.930	11.736	4.895	0.993	3.885	2.376	0.998	1.895
	RA-Boot	27.036	0.381	20.258	23.354	0.524	16.562	22.972	0.531	17.191
	RA-CG	34.200	---	27.316	34.002	0.165	27.157	31.083	0.236	24.796
6	DVR	3.125	0.996	2.995	3.014	0.992	2.860	5.509	0.973	4.364
	RA	21.316	0.569	16.870	16.869	0.745	13.038	18.120	0.707	14.362
	DVR-Boot	10.569	0.919	6.682	5.549	0.975	4.196	6.424	0.963	4.982
	DVR-CG	16.454	0.886	12.851	8.235	0.944	6.555	7.396	0.952	5.900
	RA-Boot	27.731	0.377	20.753	23.980	0.520	17.008	23.295	0.528	17.475
	RA-CG	34.889	----	27.769	34.730	0.160	27.707	31.815	0.231	25.335
20	DVR	3.731	0.994	3.605	4.563	0.986	3.901	18.639	0.767	13.782
	RA	25.380	0.546	19.942	20.458	0.717	15.895	22.239	0.668	17.657
	DVR-Boot	16.703	0.824	10.083	17.588	0.803	10.660	22.120	0.688	16.211
	DVR-CG	27.047	0.611	21.389	29.890	0.492	23.743	25.110	0.607	20.012
	RA-Boot	32.833	0.359	24.608	29.149	0.477	20.768	28.704	0.476	21.483
	RA-CG	40.680	---	32.311	41.037	0.121	32.819	38.384	0.177	30.641

EK 9. Genlerin kendi arasında düşük seviyede ilişki bulunan (ρxixj=0.10) ve genler ile yanıt değişkeni arasında ilişki katsayılarının b=3 alındığı durum için genelleştirme yöntemlerinden elde edilen performans sonuçları.

		n=20			n=100			n=500		
σ_ε	Yöntem	KHKO	R^2	OMS	KHKO	R^2	OMS	KHKO	R^2	OMS
1	DVR	9.126	0.997	8.709	14.468	0.994	11.488	5.699	0.999	4.754
	RA	62.452	0.571	49.469	49.272	0.747	38.142	53.228	0.711	42.162
	DVR-Boot	29.403	0.931	18.769	11.179	0.995	9.134	6.530	0.999	5.393
	DVR-CG	45.448	0.930	35.132	14.468	0.994	11.488	6.971	0.999	5.560
	RA-Boot	81.494	0.378	60.949	70.723	0.526	50.193	68.988	0.531	51.649
	RA-CG	103.080	----	82.250	101.111	0.167	80.768	93.272	0.235	74.355
50	DVR	10.571	0.995	10.205	11.785	0.989	10.924	46.490	0.825	34.607
	RA	72.019	0.548	56.982	58.103	0.723	45.081	62.692	0.681	49.746
	DVR-Boot	44.159	0.852	26.958	43.996	0.844	27.280	54.835	0.765	40.447
	DVR-CG	71.672	0.683	56.997	73.714	0.607	58.569	62.700	0.695	49.977
	RA-Boot	92.712	0.361	69.272	81.688	0.487	58.049	81.183	0.491	60.870
	RA-CG	116.104	---	92.977	117.180	0.134	93.659	109.083	0.185	87.000

EK 10. Genlerin kendi arasında orta seviyede ilişki bulunduğu (ρxixj=0.30) ve genler ile yanıt değişkeni arasındaki ilişki katsayılarının b=1 alındığı durum için genelleştirme yöntemlerinden elde edilen performans sonuçları

		n=20			n=100			n=500		
σ_ε	Yöntem	KHKO	R^2	OMS	KHKO	R^2	OMS	KHKO	R^2	OMS
1	DVR	4.653	0.998	4.288	3.674	0.999	3.151	2.826	0.999	2.315
	RA	32.024	0.639	25.448	22.209	0.837	17.035	25.252	0.792	19.972
	DVR-Boot	7.669	0.993	5.814	4.213	0.999	3.539	3.081	0.999	2.513
	DVR-CG	10.540	0.995	7.920	4.585	0.999	3.644	3.131	0.999	2.496
	RA-Boot	41.485	0.465	31.467	31.637	0.682	22.421	31.278	0.686	23.736
	RA-CG	57.227	----	45.719	46.788	0.371	37.263	40.892	0.483	32.600
6	DVR	4.812	0.997	4.465	4.357	0.994	3.929	5.578	0.990	4.617
	RA	32.326	0.641	25.639	22.335	0.837	17.124	25.505	0.790	20.189
	DVR-Boot	8.459	0.988	6.360	5.590	0.991	4.622	6.377	0.987	5.170
	DVR-CG	12.193	0.979	9.315	7.190	0.985	5.723	7.528	0.982	6.008
	RA-Boot	42.311	0.468	32.115	32.049	0.682	22.757	31.704	0.684	24.042
	RA-CG	57.370	----	45.593	47.704	0.360	38.042	41.281	0.481	32.942
20	DVR	5.502	0.993	5.262	5.762	0.990	5.591	18.577	0.900	14.039
	RA	34.606	0.635	27.406	24.812	0.819	19.125	28.523	0.764	22.605
	DVR-Boot	15.395	0.936	10.295	17.076	0.917	11.329	21.776	0.866	16.265
	DVR-CG	24.245	0.846	19.210	27.701	0.790	22.014	24.899	0.826	19.875
	RA-Boot	44.819	0.454	33.925	35.341	0.653	25.114	35.267	0.649	26.778
	RA-CG	61.344	----	48.835	51.748	0.329	41.241	46.087	0.428	36.748

EK 11. Genlerin kendi arasında orta seviyede ilişki bulunduğu (ρxixj=0.30) ve genler ile yanıt değişkeni arasındaki ilişki katsayılarının b=3 alındığı durum için genelleştirme yöntemlerinden elde edilen performans sonuçları.

		n=20			n=100			n=500		
σ_ε	Yöntem	KHKO	R^2	OMS	KHKO	R^2	OMS	KHKO	R^2	OMS
1	DVR	14.162	0.998	13.063	10.969	0.999	9.414	8.425	1.000	6.894
	RA	95.777	0.650	76.090	66.006	0.839	50.875	75.750	0.792	59.912
	DVR-Boot	22.853	0.993	17.356	12.572	0.999	10.552	9.192	1.000	7.498
	DVR-CG	30.718	0.995	23.098	13.571	0.999	10.777	9.297	1.000	7.408
	RA-Boot	125.364	0.468	95.251	95.243	0.682	67.609	93.999	0.687	71.361
	RA-CG	172.053	….	136.937	141.447	0.365	112.401	122.730	0.485	97.880
50	DVR	15.959	0.994	15.207	16.481	0.991	16.025	46.114	0.929	35.221
	RA	101.015	0.640	80.646	72.071	0.824	55.552	82.642	0.773	65.458
	DVR-Boot	39.623	0.954	27.225	41.702	0.943	28.879	53.709	0.905	40.553
	DVR-CG	61.579	0.891	48.770	67.865	0.852	53.950	61.845	0.875	49.360
	RA-Boot	132.028	0.450	99.632	102.831	0.662	73.045	101.652	0.660	77.244
	RA-CG	180.404	---	144.022	151.877	0.335	120.953	133.570	0.445	106.485

EK 12. Genlerin ve doz değerlerinin orijinal korelasyon yapısı ve gerçek parametre değerlerinin dikkate alındığı benzetimdan elde edilen genelleştirme yöntemi performans sonuçları.

Varsayılan parametre değerlerine göre elde edilen sonuçlar

	n=20			n=100			n=500		
Yöntem	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**
DVR	0.929	0.993	0.904	0.809	0.993	0.749	0.561	0.997	0.469
RA	6.571	0.502	5.219	5.392	0.679	4.209	5.849	0.627	4.648
DVR-Boot	5.425	0.692	3.117	2.044	0.956	1.454	0.669	0.995	0.552
DVR-CG	8.773	0.370	6.950	2.957	0.905	2.336	0.737	0.994	0.587
RA-Boot	8.365	0.318	6.293	7.464	0.451	5.389	7.106	0.471	5.375
RA-CG	10.050	----	7.993	9.959	0.149	7.956	9.230	0.209	7.366

En iyi parametrelerin ayarlanması ile elde edilen sonuçlar:

	n=20			n=100			n=500		
Yöntem	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**
DVR	0.917	0.993	0.888	0.417	0.998	0.391	0.283	0.999	0.238
RA	6.479	0.501	5.119	5.358	0.680	4.195	5.846	0.626	4.636
DVR-Boot	5.183	0.706	2.739	1.721	0.968	1.079	0.079	1.000	0.066
DVR-CG	8.504	0.351	6.739	2.459	0.934	1.937	0.378	0.999	0.301
RA-Boot	8.142	0.326	6.086	7.352	0.459	5.364	7.096	0.476	5.331
RA-CG	9.752	----	7.766	9.494	----	7.575	9.005	0.209	7.173

EK 13. Benzetim senaryosu 1 için n=500 iken DVR'nin farklı fonksiyonlarından elde edilen performans sonuçları.

Perf.Ölçüsü	Fonksiyon	DVR	DVR-Boot	DVR-CG	DVR-BDB
KHKO	DOĞRUSAL	0.929	1.351	1.267	1.216
	POLİNOM	3.514	6.181	9.466	9.195
	RADYAL	1.359	3.650	5.681	4.535
	SİGMOİD	3.172	4.294	1.694	1.505
	En iyi	DOĞ	DOĞ	DOĞ	DOĞ
OMS	DOĞRUSAL	0.774	1.067	1.011	0.970
	POLİNOM	2.109	3.857	7.541	7.371
	RADYAL	1.070	2.365	4.468	3.540
	SİGMOİD	2.766	3.384	1.349	1.207
	En iyi	DOĞ	DOĞ	DOĞ	DOĞ
R^2	DOĞRUSAL	0.991	0.982	0.984	---
	POLİNOM	0.929	0.653	0.654	---
	RADYAL	0.993	0.904	0.882	---
	SİGMOİD	0.901	0.826	0.973	---
	En iyi	DOĞ	DOĞ	DOĞ	**---**

EK 14. Benzetim senaryosu 2 için n=500 iken DVR'nin farklı fonksiyonlarından elde edilen performans sonuçları.

Perf.Ölçüsü	Fonksiyon	DVR	DVR-Boot	DVR-CG	DVR-BDB
KHKO	DOĞRUSAL	0.561	0.669	0.737	0.639
	POLİNOM	3.793	4.958	6.844	6.342
	RADYAL	1.878	3.276	4.930	4.246
	SİGMOİD	37.062	35.824	22.209	34.185
	En iyi	DOĞ	DOĞ	DOĞ	DOĞ
OMS	DOĞRUSAL	0.469	0.552	0.587	0.509
	POLİNOM	2.393	3.256	5.321	4.908
	RADYAL	1.325	2.175	3.822	3.265
	SİGMOİD	19.359	19.436	12.285	18.004
	En iyi	DOĞ	DOĞ	DOĞ	DOĞ
R^2	DOĞRUSAL	0.997	0.995	0.994	---
	POLİNOM	0.865	0.737	0.505	---
	RADYAL	0.973	0.898	0.777	---
	SİGMOİD	0.035	0.036	0.100	---
	En iyi	DOĞ	DOĞ	DOĞ	---

EK 15. Genlerin ve doz değerlerinin orijinal korelasyon yapısı ve gerçek parametre değerlerinin dikkate alındığı benzetimden elde edilen farklı bootstrap tekrarı için yöntemlerin bootstrap performans sonuçları.

		B=20			B=100			B=300		
n	**Yöntem**	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**
20	**DVR**	5.425	0.692	3.117	5.381	0.687	3.089	5.392	0.687	3.103
	RA	5.425	0.692	3.117	8.250	0.318	6.193	8.117	0.332	6.126
100	**DVR**	2.044	0.956	1.454	2.057	0.953	1.451	2.001	0.955	1.417
	RA	7.464	0.451	5.389	7.348	0.443	5.289	7.260	0.446	5.223
500	**DVR**	0.669	0.995	0.552	0.671	0.995	0.553	0.680	0.995	0.565
	RA	7.106	0.471	5.375	7.096	0.481	5.360	7.187	0.475	5.442

EK 16. Genlerin ve doz değerlerinin orijinal korelasyon yapısı ile gerçek parametre değerleri dikkate alınarak benzetimden elde edilen farklı parça (fold) sayıları için yöntemlerin ÇG performans sonuçları.

		Fold=3			Fold=5		
n	**Yöntem**	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**
20	**DVR**	8.773	0.370	6.950	8.313	0.483	6.594
	RA	10.050	----	7.993	9.779	-----	7.805
100	**DVR**	2.957	0.905	2.336	2.256	0.943	1.785
	RA	9.959	0.149	7.956	9.754	0.177	7.791
500	**DVR**	0.737	0.994	0.587	0.686	0.995	0.546
	RA	9.230	0.209	7.366	9.110	0.220	7.265
		Fold= 10			**LOO (K=n)**		
n	**Yöntem**	**KHKO**	**R^2**	**OMS**	**KHKO**	**R^2**	**OMS**
20	**DVR**	7.919	1.000	6.260	7.980	---	6.351
	RA	9.831	---	7.854	9.861	---	7.807
100	**DVR**	1.890	0.959	1.509	1.781	---	1.409
	RA	9.552	0.223	7.562	9.603	---	7.609
500	**DVR**	0.656	0.995	0.523	0.639	---	0.509
	RA	8.978	0.234	7.159	8.873	---	7.059