PREDICTING SOLAR ENERGY PRODUCTION USING INCREMENTAL MACHINE LEARNING TECHNIQUES

KADEMELİ MAKİNE ÖĞRENMESİ TEKNİKLERİ KULLANARAK GÜNEŞ ENERJİSİ ÜRETİMİ TAHMİNİ

SEMANUR KAPUSIZOĞLU

ASST. PROF. DR. DERYA DİNLER

Advisor

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fullfillment to the Requirements

for the Award of the Master’s Degree

2023

ABSTRACT

Predicting Solar Energy Production Using Incremental Machine Learning Techniques

Semanur KAPUSIZOĞLU

Master's Degree, Department of Industrial Engineering

Supervisor: Asst. Prof. Dr. Derya DİNLER

June 2023, 118 pages

Energy is a significant part of life and the economy, with an aggressively increasing demand due to population growth. Non-renewable sources, such as fossil fuels, are rapidly depleting and cannot meet the demand, leading to a reliance on different energy sources. Considering the environmental effects of fossil fuels, many individuals are leaning towards cleaner and renewable energy sources, such as solar and wind power. Solar power holds an important share due to its abundance and ease of implementation. The amount of solar energy produced depends on various factors, such as temperature, photovoltaic radiation, cloud cover, and location. Predictive models considering those factors for solar energy play a crucial role in creating efficient production and distribution networks. Machine learning models are becoming increasingly popular among other predictive approaches thanks to technological advancements. Machine learning is an area of programming that creates mathematical algorithms and models, enabling computers to learn and make predictions without explicit programming. There are different training approaches for machine learning models. The traditional approach divides data into training and testing sets and uses all training data at once. Online (Incremental) learning is the principle of feeding the prediction model with one data point from a training set at a time, often used in sectors where data patterns are variable. This principle can be adapted to various data mining algorithms, including supervised and unsupervised learning. In this study, the suitability of the incremental training approach is tested on solar energy production using six different machine learning models (Linear Regression, Lasso Regression, Ridge Regression, Decision Tree, Random Forest, and Artificial Neural Network). An open-source competition on the Kaggle platform, provided by the American Meteorological Society, is utilized to assess whether online models can outperform traditional models in solar energy predictions. Incremental training methods found to perform better than traditional methods in terms of Mean Absolute Error.

Keywords: Machine Learning, Solar Power Prediction, Incremental Training, Renewable Energy

ÖZET

Kademeli Makine Öğrenmesi Teknikleri Kullanarak Güneş Enerjisi Üretimi Tahmini

Semanur KAPUSIZOĞLU

Yüksek Lisans, Endüstri Mühendisliği Bölümü

Tez Danışmanı: Dr. Öğr. Üyesi Derya DİNLER

Haziran 2023, 118 sayfa

Enerji, hayatın ve ekonominin önemli bir parçası olup, nüfusa bağlı olarak hızla artan bir talebe sahiptir. Fosil yakıtlar gibi yenilenemez kaynaklar hızla tükenmekte ve talebi karşılayamamaktadır, bu da enerji sistemlerinin farklı kaynaklar aramasına neden olmaktadır. Fosil yakıtların kullanımıyla ortaya çıkan çevresel sorunlar ve dünyaya olan etkileri göz önüne alındığında, güneş ve rüzgar gibi temiz ve yenilenebilir enerji kaynaklarına olan eğilim artmaktadır. Güneş enerjisi, bol miktarda bulunması ve uygulanabilirliği kolay olması nedeniyle yenilenebilir enerji piyasasında önemli bir paya sahiptir. Üretilen güneş enerjisi miktarı, sıcaklık, güneşin açısı, fotovoltaik radyasyon, bulut miktarı ve konum gibi birçok faktöre bağlıdır. Güneş enerjisi için bu faktörleri dikkate alarak geliştirilen tahmin modelleri, verimli üretim ve dağıtım ağı oluştururken önemli bir role sahiptir. Teknolojik ilerlemelerle birlikte, makine öğrenmesi modelleri diğer tahminleme yöntemleri arasında giderek daha popüler hale gelmektedir. Makine öğrenmesi, bilgisayarların açıkça kodlanmadan tahmin yapmasını sağlayan matematiksel algoritmalar ve modeller oluşturmayı hedefler. Makine öğrenimi modelleri farklı şekillerde eğitilebilir. Geleneksel eğitim yaklaşımı, veriyi eğitim ve test verisi olarak ayırarak eğitim verisinin tümünü tek seferde kullanır.. Çevrimiçi (Kademeli) öğrenme, tahmin modelini eğitim için ayrılan veriyle birer birer besleme prensibidir. Bu, verilerdeki desenlerin değişken olduğu ve bu varyasyonları yakalamanın önemli olduğu birçok sektörde kullanılır. Çevrimiçi eğitim prensibi, denetimli, denetimsiz ve birçok diğer veri madenciliği algoritmasına adapte edilebilir. Bu çalışmada, çevrimiçi eğitim yaklaşımının güneş enerjisi üretimindeki uygunluğu, 6 farklı makine öğrenmesi modeli (Doğrusal Regresyon, Lasso Regresyon, Ridge Regresyon, Karar Ağacı, Rastgele Orman ve Yapay Sinir Ağı) kullanılarak test edilmiştir. Çevrimiçi modellerin güneş enerjisi tahminlerinde geleneksel modellere göre daha iyi performans gösterip göstermediğini değerlendirmek için Amerikan Meteoroloji Derneği tarafından sağlanan Kaggle platformundaki açık kaynaklı bir yarışma verisi kullanılmıştır. Ortalama Mutlak Hata değerlerine göre çevrimiçi modeller daha başarılı bulunmuştur.

Anahtar Kelimeler: Makine Öğrenmesi, Güneş Enerjisi Tahminleme, Yenilenebilir Enerji, Çevrimiçi Model Eğitimi

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AMS	American Meteorological Society
ANN	Artificial Neural Network
CSV	Comma Separated Values
ECMWF	European Center for Medium-Range Weather Forecasts
FFNN	Feed Forward Neural Network
GEFS	Global Ensemble Forecast System
lightGBM	Light Gradient Boosting Machine
LSR	Least Squares Regression
MAE	Mean Absolute Error
ML	Machine Learning
NCEP	National Centers for Environmental Prediction
NWP	Numerical Weather Prediction
PCA	Principal Component Analysis
SVM	Support Vector Machine
SVR	Support Vector Regression
VIF	Variance Inflation Factor

1. INTRODUCTION

Energy is the fuel of our universe. All life emerges from it and relies on it to flourish. It comes in many different forms. As civilizations evolve, humanity learned how to produce energy and power technology with it. In this scope, fossil fuels (coal, oil, natural gas, etc.) have been used in energy production for various purposes, such as transportation, heating, and manufacturing. Over time, as the population of the world grew, the need for energy also increased. Due to their non-renewable nature, fossil fuels rapidly started depleting and becoming unable to meet the demand. Realizing the environmental problems caused by the use of fossil fuels, naturally replenishable sources gained more interest. These sources are also called flow sources, energy is produced by the movement of them. Some examples of such resources are solar, wind, hydrothermal, and geothermal.

Renewable energy sources are environmentally friendly and can quickly replenish themselves. They are also called clean energy sources, the reason behind this is when they are consumed, they do not emit as much CO_2 as fossil fuels. Fossil fuels are formed from remnants of dead plants and animals. These materials are stuck and throughout millions of years with pressure and heat, they become fuels. It takes much longer for them to renew themselves.

One of the most popular and rapidly growing forms of renewable energy is solar energy. It is regarded as the most plentiful and easily accessible source of renewable energy, with enormous potential for utilization. It makes a substantial worldwide contribution to cutting carbon emissions. According to the International Energy Agency (IEA) Photovoltaic Power Systems Programme's (PVPS) most recent Trends in Photovoltaics Applications report, installed PV capacity avoided more than 860 million tons of CO_2 by the end of 2020, and it is anticipated that the gigatonne (Gt) milestone was crossed in 2021 [1]. Solar energy is anticipated to exceed both fossil fuels and other renewable sources including wind, hydropower, and bioenergy as the leading source of electricity generation by 2050, according to IEA [2].

The growing demand has led to scientific advancements in producing more effective and affordable PV panels. Additionally, tax incentives from governments, support of big tech companies by utilizing solar power, and companies that make it very smooth and easy to install solar plants have helped drive the costs down. Therefore, many individuals, businesses, and governments are turning to solar power to meet their energy needs. This trend is expected to continue, with considerable experts predicting that solar power will play a significant role in the transition to a clean energy future [3]. Expanding the use of solar energy is very crucial for a better, cleaner future. Solar energy will drastically reduce our reliance on dirty, non-renewable energy sources like coal and oil, due to its abundant, clean, and renewable nature. Fossil fuels might not be found in every corner of the world and might need to be exported from somewhere else. This creates a dependency on authorities having these resources. Because it can be locally produced, solar energy is one of the prime sources to reduce energy poverty and reduce such dependencies.

Solar energy brings many benefits but also introduces a complex problem due to its stochastic nature. Predicting its production is highly affected by some environmental and physical factors. Especially, the changes in weather conditions directly affect the amount of energy produced. Factors like air temperature, angle of incidence of the sun, wind speed, photovoltaic radiation, amount of clouds in the area, and location of clouds are some of these.

Electricity grids powering our life rely on a balanced distribution. Integrating solar energy into grids is hard because its production amount is highly variable. Balancing a grid means stabilizing the demand and supply of electricity. All electric devices in our homes are powered with a specific amount of voltage. Imbalances in grid systems might break the fuses causing power outages, or worse, destroying electrical devices. Additionally, storing electricity is a hard and costly method, which makes it undesirable. Therefore, predicting solar energy production is an important problem in the literature, studies on its effective use, distribution, and pricing have always been of interest. Studies focusing on the prediction of solar energy can help better understand the need, and simulate the process before taking any installment or expansion decisions. Therefore reliable estimates support the economy, growth, and development of more solar energy plants as a clean and renewable energy source.

The goal of this study is to use online machine learning to predict solar power production, which has one of the largest shares in renewable energy production. Machine learning is a computational method to create programs that can learn the complex underlying patterns from a given dataset and make predictions without explicitly programming the models. The size, quality, and complexity of the data used highly influence the training times and performance of these predictive models. Many approaches have recently been developed to improve accuracy or reduce the time taken for training. Online learning is one of these approaches. It works by gradually feeding the portion that is set aside for training, unlike the traditional approach, which uses all at once. The American Meteorological Society (AMS) Solar Power Prediction Competition dataset on the Kaggle platform is taken as a dataset to see if online models can outperform traditional models in terms of Mean Absolute Error. In previous studies which use the same dataset, many different machine learning approaches are implemented to the dataset but online (incremental) learning models have not been used to the best of our knowledge. Incremental approach is helping models iteratively adjust model parameters to incorporate the pattern changes in the source system, therefore providing more accurate results. The main novelty of this thesis is the implementation of online learning to solar energy prediction. Through extensive statistical analysis and comprehensive model runs, this study demonstrates that online learning models can contribute to the solar energy industry when compared to traditional learning models. Moreover, even with the smallest parametric configurations for online learning models, we become competitors to the traditional learning models which include heavy research behind them.

The sections of the thesis are as follows: Section 2 broadens the energy prediction studies and provides information about machine learning techniques. Explains the traditional and incremental (online) training approaches for machine learning along with use cases. The third section, Related Work, discusses previous studies on solar power prediction, and different machine learning methods used in that area, and touches on the works based on the selected dataset. Section 4 describes the selected dataset and the proposed method to test the hypotheses. The steps taken to implement the proposed method, including data preprocessing, model building, experiment design, and implementation are covered in Section 5. Based on the experimental result, the best-performing model is selected and improved in Section 5 as well. The sixth section, Conclusion & Discussions summarizes the process, discusses the results, and provides directions for future research.

2. GENERAL INFORMATION

Predicting weather has always been crucial for humanity. Early civilizations used the astronomical observations they obtained to form calendars and decide how climate & weather change when compared to the position of other celestial bodies. In the modern world, advanced mathematical prediction models are used for this purpose, the most popular being Numerical Weather Prediction (NWP) models. It is handy for meteorological forecasts, air quality modeling (especially crucial to know when the location is near an active volcano or when a serious chemical hazard is in place), hurricane and tornado formation, and to where and at what speed they will move. As the name suggests, these prediction models constitute complex equations to catch the possible patterns for weather status and help us to predict atmospheric features. They use satellite imagery, and sensor data from ground weather observatories along with other possible meteorological observations as input. Their accuracy relies on input data, as many other prediction models, along with computational power, to care for the sophisticated background operations. NWP models can be classified into two main categories:

- Models producing data for global coverage, the Global Ensemble Forecast System (GEFS) by the US government,
- Models produce data for constrained regions and use that to create global coverage, the European Center for Medium-Range Weather Forecasts (ECMWF) model, by a European entity [4].

Within the scope of this study, the selected dataset is utilizing the first NWP model, therefore we will focus on GEFS models. The National Centers for Environmental Prediction (NCEP) created the Global Ensemble Forecast System (GEFS) as a weather model to overcome the shortcomings and uncertainties in the input data and model itself. To compensate for various sources of uncertainty in the input data, such as limited coverage, instrument biases, and observing limitations, GEFS generates 21 separate forecasts or ensemble members. The generation of multiple forecasts produces a range of potential outcomes, providing a measure of the uncertainty associated with the weather forecast. Each forecast in the ensemble compensates for a different set of uncertainties [5].

Outputs of GEFS models are forecasted weather variables, which can be related to temperature at different heights, precipitation, humidity, wind speed, cloud cover, etc. They are used as input to other prediction models for various purposes, like weather forecasts, energy production amounts, and paths of tropical storms. Machine learning models are one of the most popular ways among these prediction models.

To better allocate the scarce sources of the Earth, various optimization/prediction algorithms have been developed over time. Thanks to advancements in technology, some questions that were previously impossible to answer can now be solved using increased processing power and computational ability. Especially algorithms using machine learning, deep learning, and artificial intelligence are becoming increasingly popular. To best answer real-life problems, these models are applied to the relevant situation. Machine learning models can be classified as follows:

- Depending on whether the data is labeled or not (supervised, unsupervised, semi-supervised),
- Depending on whether they can learn instantly or gradually (online or offline/batch learning),
- Depending on if they learn by comparing new data points to existing data points or by recognizing patterns in training data and developing a prediction-based model as scientists do (instance-based versus model-based learning),
- Depending on whether they work with a reward-penalty system (reinforcement learning) [6].

Machine learning algorithms use a dataset to “learn” and “predict”. They are trained with a different set of approaches to use the available data most efficiently. In the next step, based on the information learned from the patterns in the training dataset, the model makes predictions on data it has never seen before. As implementers of the algorithms, our main goal is to minimize errors when these predictions are compared with real-life situations. In this context, different learning methods are preferred for the models, taking into account the available data and the purpose of the study. The speed, characteristics, or importance of the problem at hand are significant factors in the choice of learning method. Some models can be trained with available data and statically taken into production, while others require continuous updating of the model based on the incoming data.

Algorithms that operate on the batch learning logic use the entire dataset allocated for training before creating any predictions. Model is trained by using all data available at hand, and when new data arrives, it is consolidated with the previous one. The model is re-trained with a consolidated dataset before putting on production.

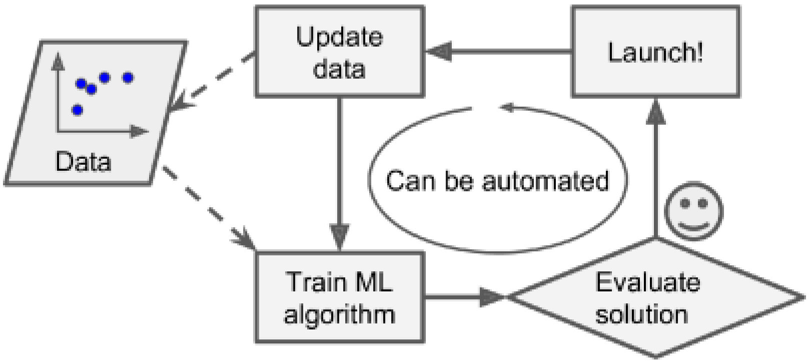


Figure 2.1 Process of Batch (Offline) Learning [6]

On the other hand, online learning algorithms differ from batch learning models in that they arrange data points according to their arrival order instead of feeding the information in the dataset to the model all at once, feeding them one by one to train the model and update the model at each step. This technique enables the system to pick up information about fresh data in real-time as it arrives, allowing for quick and cost-effective learning processes. It is especially beneficial for systems that process streaming data, requiring rapid and autonomous response to changes (such as stock prices, IoT data and solar energy prediction).

A crucial parameter in these models is learning rate which decides how quickly it should adapt to the incoming data. Higher learning rates result in rapid adaptation to new data but increases the risk of disregarding previous data, which is problematic in applications like spam filters. Conversely, a low learning rate results in slower learning but provides benefits such as decreased sensitivity to noise and outliers in the incoming data stream. Additional inherent challenge in the context of online learning lies in the potential deterioration of system performance when exposed to bad data. Instances of this problem may arise from a malfunctioning sensor on a robot or attempts to manipulate search engine rankings through spamming activities. To mitigate this risk, diligent monitoring of the system is essential, promptly deactivating the learning process and potentially reverting to a previously functional state upon detecting a decline in performance. Furthermore, monitoring the input data and employing anomaly detection algorithms can aid in identifying and responding to abnormal data patterns.

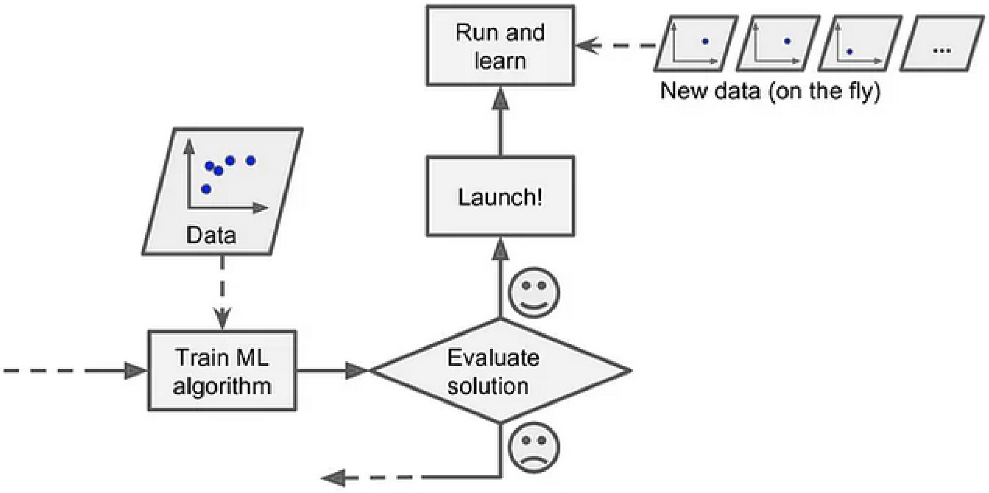


Figure 2.2 Process of Online Learning [6]

Apart from being computationally efficient, these algorithms are independent of the assumptions made before starting the analysis of the data source. The reason for their computational efficiency is that the processing time is usually constant and each data point is examined one by one. Thus, the processing time can be linearly scaled with the number of data points. These models typically run algorithms that are created by combining the good parts of multiple algorithms, known as "ensemble models," rather than using a uniform algorithm in the background. [7].

Online learning, being essentially a model training method, can be adapted to various machine learning algorithms and thus can offer successful models that can find their place in many different industries/problems. One example of supervised learning algorithms that are often adapted to this training model is email spam filtering. Malicious emails and links produced using social engineering techniques are changing and evolving in parallel with developments in cybersecurity. Therefore, models trained using offline learning, although updated at regular intervals, may become outdated and inadequate in filtering due to the speed of these developments. For this reason, online learning can be used to track changes in data and incorporate them into the process [8, 9]. When the studies conducted in the field of renewable energy prediction are examined this year, it is observed that models have been established using this training technique and successful results have been obtained [10].

Similarly, for the regression analysis cases, stock price prediction and portfolio selection models can work based on the online learning principle[11, 12]. These algorithms can provide the best response to sudden fluctuations in stock prices and different underlying patterns in the data. The fact that the data is already analyzed with time series models in such situations and progresses sequentially also increases the applicability of this training method [13]. The structure of many attributes commonly used in predicting renewable energy production also exhibits trends that are highly suitable for regression or time series models. Many prediction studies involve using these methods in multiple stages or with ensemble models and achieve impressive results [14, 15].

Unsupervised learning in machine learning refers to the clustering of data during the training process without the need for labels or tags. The unsupervised learning algorithms that are trained with an online learning approach, are quite similar to the traditional algorithm, the only difference is that they work with streaming data. Some examples include online dimensionality reduction and anomaly detection [16]. Weather prediction, which is one of the frequently used inputs in renewable energy models, is a difficult variable to predict. The quality of the results obtained from the predictions also affects the newly created model. Therefore, instead of making pinpoint predictions, some energy models cluster the weather using unsupervised learning methods and include it in the newly created model [14, 17].

Recommendation systems, also known as bandit learning, are highly suitable for online learning studies. Products that users are interested in are not static and can vary greatly depending on the last products they view. If e-commerce organizations and similar institutions cannot adapt to these changes quickly, this situation adversely affects their profits. Providing personalized recommendations based on a user's profile created with data from a certain period in the past is not logical for these systems. Users' thoughts and actions are shaped by opportunities and advertisements they encounter on the internet at any moment. Therefore, predictions based on instant user movements will result in more accurate and realistic models [18, 19]. Recommendation systems are also frequently used in studies aimed at more efficient use of generated energy in buildings and smart grid systems [20].

Reinforcement learning is an area of machine learning that studies how intelligent agents should behave in a given environment in order to maximize cumulative reward. These models observe the environment, make instant analyses based on the situations they encounter, and respond accordingly. As a result of these responses, the model learns and aims to function most efficiently (by minimizing the penalty or maximizing the reward) through a reward or punishment system [21, 22]. If we look at it from a renewable energy perspective, reinforcement models are being implemented in the management of micro-grids, and successful results are being achieved [23, 24].

3. RELATED WORK

This section shares summaries of the related work within the literature. In the first subsection, papers related to renewable energy studied in Turkey are examined to have a better understanding of the different aspects of renewable energy and how the attention is distributed within Turkey. Then the focus for the second subsection is on solar power prediction. It walks you through approaches, selected models, and metrics to better understand the solar power prediction process. Then lastly, in the third subsection, the scope is narrowed down to the papers only focusing on the dataset which is used for this study, the American Meteorological Society (AMS) 2013-2014 Solar Energy Prediction Contest on the Kaggle platform.

3.1. Studies Having a Focus on Renewable Energy

Studies in renewable energy within Turkey are mainly studied as master's thesis and mostly they are coming from Industrial Engineering Departments (4 out of 10 papers). The most recent one is [25]. 3 of the papers are directly working on renewable energy prediction problems, and the others are facility selection problems, multi-criteria decision-making problems, or the effect of numerical weather predictions in wind power prediction. Studies are either taking different renewable energy sources together or they are focusing on wind power prediction. Consolidated summary of the studies are reported in Table 3.1.

In his thesis, Ünal uses the dataset from a plant in Germany to predict the average renewable energy production incorporating solar and wind energy. During the evaluation, the long short-term memory forecasting model (LSTM) exhibited better performance when using MAE as an evaluation metric [25].

Atcı built a Markovian model to predict wind energy production. In the implementation phase, hourly average wind speed and energy production values measured between 2013-2014 at a wind power plant located in the Belen region of Hatay province were utilized. The author inferred that the future values of wind energy generated through Markov chains can be reliably predicted. [26].

Horasan created a multi-objective decision-making model for effective renewable energy planning that incorporates the major renewable energy sources such as solar, biomass, wind, hydropower, and geothermal. The developed model is being implemented in 21 Turkish cities, with four objective functions representing the technical score of regions based on renewable energy types, the unit cost of renewable energy technologies, job creation based on renewable energy types, and the environmental score of renewable energy resources. They discovered that in both stages, solar energy can meet fifty percent of the whole energy demand, while biomass energy can meet forty-seven percent of the total, and geothermal energy can meet four percent of the total. [27].

Shtewi used wind data from Libya to predict yearly energy production rates and do a technical financial analysis and greenhouse gas emissions analysis. Weibull distribution and three different methods to calculate it; graphical, experimental, and extreme possibility is used to observe their impact on the results and the Adirsiyah region founded more efficient for future wind projects, as it would be able to restore the project cost in a period that is less than wind turbines lifetime [28].

Altun examines the factors affecting the use of renewable energy sources in electricity production in Turkey and around the world. Vector Error Correction Model estimation was made using annual data in Turkey and long- and short-term relationship was determined between the variables with high impact power. As a result, it has been seen that the variables used affect electricity production, and the production affects the gross domestic product per capita, electricity consumption per capita, and the foreign dependency ratio in energy [29].

Kaya used Artificial Neural Network (ANN), Auto Regressive Moving Averages (ARIMA), and gray estimation models to forecast the 12-year electricity demand based on the consumption data between 1990-2014. Mean Absolute Percentage Error (MAPE) was used as the error function and it was discovered that the ANN models were more successful than the others [30].

A study by Çetin compares the performance of different models in the Wind Energy Monitoring and Forecasting Center (RITM) system. Comparison is made by using six wind power plants in Turkey, selected based on their high wind potential and terrain structure, and their observed wind speed data over a 3-4 year period with outputs from three different mesoscale numerical weather forecast models. Results are analyzed on diurnal, seasonal, and monthly bases using Root Mean Squared Error (RMSE), bias, and correlation coefficients to determine the best grid points for each model [31].

Turan used time series forecasting methods on 5 regions in the Marmara Region, Turkey. Mean Absolute Error (MAE), Mean Squared Error (MSE), RMSE, and MAPE scores are compared to see how time series models behave. Samples for models are collected in 3 different time intervals, with different sizes and seasons to assess the ability of time series to adapt to different situations and fluctuations in data. Obvious decreases in scores are observed when predictions are made for longer periods [32].

Derse studied a site selection problem in Turkey's provinces to completely switch from non-renewable energy sources to renewables. Biomass, geothermal, wind, solar, and hydroelectric energy are used as input. Time series models are used for energy demand forecasting, and monthly gross electricity production values between 2001-2020 are taken as monthly measurements until the end of August 2020 in kWh as training data. Weighted goal programming, fuzzy weighted goal programming, prioritized goal programming, and conic scaling methods are developed in each of the multi-objective programming models integrated with model results. Each developed multi-objective programming model used six pre-defined objectives for the process. Results of all models and sites offered for facility installation are comprehensively summarized in the study [33].

Ervural has put together a renewable energy investment strategy model that employs multi-objective decision-making techniques. For long-term choices, the model seeks to enhance competing objectives such as total power production cost, the distance between electricity production and consuming areas, renewable energy potential, social-environmental impact factor and avoided carbon emissions. One of the model inputs, renewable energy demand amounts, was generated using the Support Vector Regression (SVR) model with independent variables, followed by Holt-Winters and Genetic Algorithm (GA)-based models developed without using independent variables. Accordingly, considering the strategies pursued by the country, the current situation, and global trends, it is proposed to obtain 44% of electricity consumption from renewable energy sources for 2023, 50% for 2030, and 70% for 2050 [34].

Table 3.1 Renewable Energy Focused Studies

Author	**Year**	**Scope**	**Topic**	**Method**
Unal	2022	Msc Thesis	Predicting Solar + Wind Energy Production	LSTM
Atcı	2014	Msc Thesis	Predicting Wind Energy Production	Markov Chains
Horasan	2021	Msc Thesis	MCDM for Effective Renewable Energy Planning	MCDM
Shtewi	2021	Msc Thesis	Predicting Wind Energy Production & Financial Wind Assessment	WeiBull (visual method, empiric method and highest probability method)
Altun	2019	Msc Thesis	Factors Affecting the Use of Renewable Energy Sources in Electricity Production	Vector Error Correction Model
Kaya	2017	Msc Thesis	Forecasting Electricity Demand	ANN, ARIMA, Gray Estimation Models
Çetin	2018	Msc Thesis	Comparing Performance of NWP models in Wind Energy Monitoring and Forecasting	NWP Models, GIS Softwares (for prediction & comparison)
Turan	2019	Msc Thesis	Wind Power Prediction	AR, MA, ARMA, ARIMA, Box-Jenkins Models
Derse	2022	PhD Thesis	Site Selection Problem to Completely Switch to Renewable Energy in Turkey	MCDM (Weighted goal programming, fuzzy weighted goal programming, prioritized goal programming, and Conic Scaling methods)
Ervural	2018	PhD Thesis	Renewable Energy Investment Strategy Model	MCDM

3.2. Solar Power Prediction in the Literature

Solar energy prediction is a complex and multifaceted problem that can be approached from various perspectives. Therefore, it has attracted the attention of many researchers in the literature, resulting in valuable studies. Some authors have approached solar energy data as a time series problem to detect trends and work on models to develop appropriate solutions. Meanwhile, other authors have produced more complex ensemble methods. For the scope of the thesis, some review papers are examined to have a broader picture of what has been done, what are the features that have been widely used, metrics to measure the performance of models, and how they compare to each other. Consolidated summary of the studies are reported in Table 3.2.

Wu et al. made a comprehensive review of solar power prediction literature, grouped the models as physical and statistical, and visualized it in Figure 3.1. All solar power prediction models are using MAE, RMSE, MSE and variations of these as performance metrics. Average forecasting error found to be less when the time horizon of forecasting is shorter. Solar irradiation, temperature, and wind speed are the input parameters most frequently utilized. However, certain studies incorporate more sophisticated input variables, including global horizontal irradiance, diffused horizontal irradiance, diffused normal irradiance, and total cloud coverage. Additionally, it has been discovered that input selection and parameter optimization can improve the accuracy of machine learning models [35].

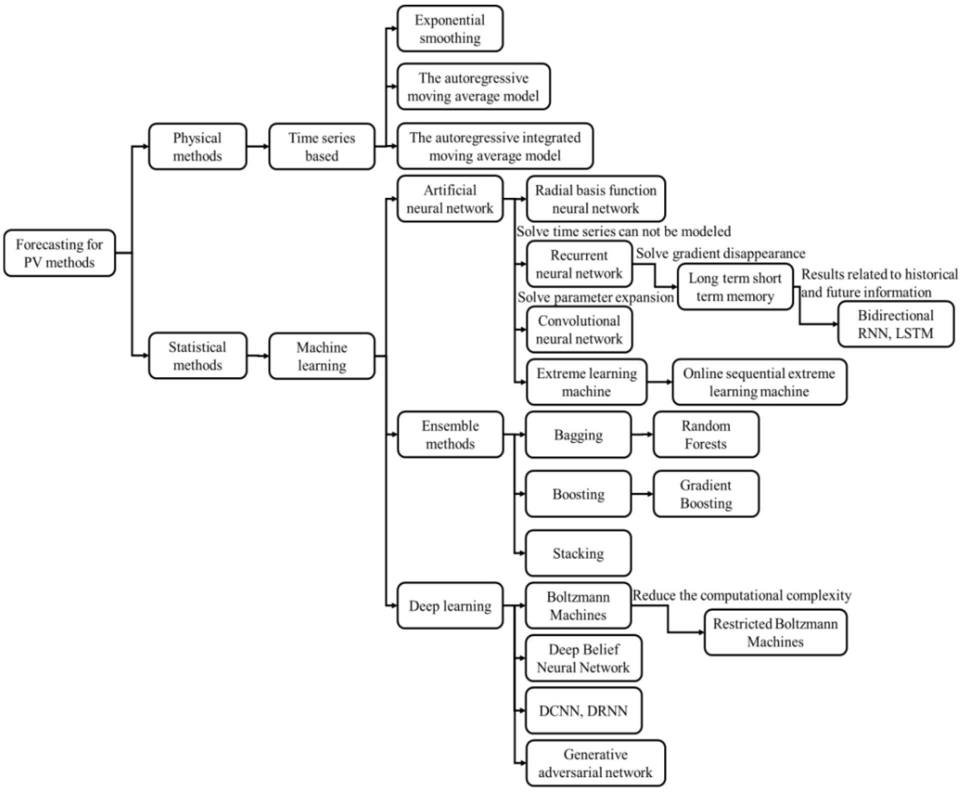

Forecasting for PV methods
Physical methods
Time series based
Exponential smoothing
The autoregressive moving average model
The autoregressive integrated moving average model
Statistical methods
Machine learning
Artificial neural network
Radial basis function neural network
Solve time series can not be modeled
Recurrent neural network
Solve gradient disappearance
Long term short term memory
Results related to historical and future information
Bidirectional RNN, LSTM
Solve parameter expansion
Convolutional neural network
Extreme learning machine
Online sequential extreme learning machine
Ensemble methods
Bagging
Random Forests
Boosting
Gradient Boosting
Stacking
Deep learning
Boltzmann Machines
Reduce the computational complexity
Restricted Boltzmann Machines
Deep Belief Neural Network
DCNN, DRNN
Generative adversarial network

Figure 3.1 PV Forecasting Models

Guermoui et al. made a comprehensive review paper summarizing how solar power is forecasted by using hybrid models in the literature. Models are divided into 5 categories:

- General ensemble learning approaches (GELA),
- Cluster based ensemble learning approach (CELA),
- Decomposition based ensemble learning approaches (DELA),
- Decomposition-clustering based ensemble learning approaches (DCELA),
- Evolutionary ensemble learning approaches (EELA).

GELA is built on the assumption that each model contributes to the forecasting process in a unique way. In this regard, numerous models combined using a variety of techniques to improve the final forecast's performance.

The main working principle of clustering-based ensemble learning models relies on data mining approaches, whereby datasets are partitioned into numerous clusters, each encompassing data samples exhibiting comparable characteristics. Subsequently, clusters are assigned to predictive models either linear or nonlinear. The resulting forecast is then derived by aggregating the predicted signals from each cluster. Typically, the unsupervised K-means algorithm and its alternatives are employed for the purpose of clustering.

The primary idea behind DELA is to dissect a non-stationary signal into a number of important signals in order to stabilize time series data. Each component is forecast independently, and the final forecasting results are calculated by integrating all anticipated components into a single signal. Non-linear models are used to estimate higher-frequency component signals, while linear models are utilized for calculating lower-frequency component signals. The final results are obtained by combining the linear and non-linear models.

The ensemble learning strategy based on decomposition clustering revolves around clustering and decomposition approaches. This particular category has shown better performance when compared to both approaches, as it leverages the advantages of both decomposition and clustering methodologies.

Evolutionary algorithms are computational approaches inspired by biological evolution that iteratively search for optimal solutions to complicated problems utilizing strategies such as mutation, selection, and reproduction. These approaches are commonly employed in the context of solar radiation assessment, as they offer effective solutions for tackling complex problems in this domain.

The forecasting approach in Residual Ensemble Learning Systems is based on the premise that solar radiation has linear and non-linear components. For linear components, a basic linear model is utilized, while for residual components, a non-linear model is used. The final forecasting signal is obtained by combining the findings of linear and nonlinear models.

In all of the analyzed situations with diverse inputs and outputs, all of the suggested hybrid models outperform the stand-alone models. The authors found it difficult to evaluate the performance of various hybrid models due to differences in the region of interest's meteorological conditions, accessible inputs-outputs, forecasting timeframes (monthly, daily, hourly, etc.), and the use of different assessment metrics. Based on the findings of the DCELA category, it surpasses the DELA category in the same region with the same data. DCELA-based hybrid model beats the DELA model for solar radiation assessments, and DCELA is the best combination strategy for solar radiation forecasts [36].

Erten et al. used four different regression techniques (lasso, linear, logistic, and elastic regression) to predict solar power production by using an open-source dataset from Kaggle. Root Mean Squared Error (RMSE) is used to compare model performance. As a feature engineering step, dimensionality reduction is implemented (Principal Component Analysis, aka PCA) and it improves the model accuracy. The best accuracy is obtained through ElasticNet Regression [37].

In their research, Sarmas et al. suggest a unique, integrated online (or incremental) learning model that tackles the dynamic character of learning settings in energy-related time-series forecasting difficulties. Suggested methodology is applied to the problem of energy forecasting, yielding models that dynamically adapt to distinctive streaming data patterns. The evaluation is carried out utilizing a real-world use scenario involving the prediction of energy consumption and renewable energy source generation. A Multi-Layer Perceptron Regressor with 4 layers is built and re-trained daily to keep the model up-to-date. Experiment findings show that online learning models outperform offline learning models in terms of MAE by 8.6% for energy demand forecasting and 11.9% for renewable energy source forecasting. [38].

Kenat et al. offer a novel approach for effectively predicting solar irradiance despite the inputs varying significantly, namely the Regression Enhanced Incremental Self-Organizing Neural Network (RE-SOINN). The proposed program works by incrementally learning time-series solar irradiance data and predicting it in real-time using an unsupervised model. It is innovative in that it uses the regression approach to convert data from discrete (as in the conventional) to continuous. It increases prediction accuracy even further by dividing the input data into two components (low and high-frequency components) before feeding it into the RE-SOINNs. The suggested approach outperforms the Persistence model, Exponential Smoothing Model, and Artificial Neural Networks based on accuracy [39].

Unlike other incremental learning papers, Balzategui et al. focus on a different problem in the solar energy industry. Authors suggest a few-shot incremental learning model to detect the defects in solar cells. While Deep Neural Networks (DNNs) show promising results in defect identification, effective classification often requires a huge amount of annotated data. The process of annotating data is complex since it is not possible to get a comprehensive sample that includes all possible defect variations due to the irregular occurrence of defects and the occurrence of particular defect types on occasion. To overcome this issue, the study focuses on the use of weight imprinting in DNNs for defect detection in industrial settings. Based on the experiments, the proposed technique enables the gradual inclusion of new defect classes using a limited number of samples, thereby enhancing the network's capabilities. Experimental results validate the effectiveness of this technique in detecting new defect classes and extending the network's detection capabilities [40].

Table 3.2 Solar Energy Focused Studies

Author	Year	Type	Highlights
Wu	2022	Review Paper on Solar Power Prediction	- All solar power prediction models are using MAE, RMSE, MSE and variations of these as performance metrics. - Average forecasting error decreases for short term predictions. - Input selection and parameter optimization increases performance. - Frequently used parameters: Solar irradiation, temperature, and wind speed
Guermoui	2020	Review Paper on Ensemble Models for Solar Power Prediction	- Models are divided into 5 categories: General ensemble learning approaches (GELA),Cluster based ensemble learning approach (CELA),Decomposition based ensemble learning approaches (DELA), Decomposition-clustering based ensemble learning approaches (DCELA),Evolutionary ensemble learning approaches (EELA).
Erten	2022	Solar Power Prediction	- Metric: RMSE - PCA implemented - ElasticNet Regression
Sarmas	2022	Solar Power Prediction	- Incremental learning Multilayer Perceptron Regressor - Online learning model outperformed traditional model by %8.6 in terms of MAE
Kenat	2020	Solar Irradiance Forecasting	- Regression Enhanced Incremental Self-Organizing Neural Network (RE-SOINN) - Suggested model outperforms ANN, Persistence Model and Exponential Smoothing based on accuracy
Balzategui	2023	Detecting Defects in Solar Cells	- Few-shot incremental learning DNN model to detect defects in solar cells - Proposed method introduces new defect classes to the model gradually and enhances performance, helping increase the model's detection capabilities.

3.3. The AMS Solar Energy Prediction Contest in the Literature

AMS Solar energy prediction dataset draws a lot of attention from the community. Despite the competition being launched 9 years ago, there are recent studies using it. One important contributor to that is the stability and reliability of the dataset. Features come from the GEFS system that is run by the US government making it a reliable source of input. When the GEFS variables, mesonet production data and metadata about these are combined, it provides researchers diverse options for feature selection. Competition being in Kaggle gives the possibility to compare models with hundreds of others. This subsection summarizes studies conducted by using AMS Solar energy prediction dataset. Consolidated summary of the studies are reported in Table 3.3.

Saad et al aimed to assess the impact of different ensemble models on prediction accuracy. In the feature extraction step, the nearest 4 GEFS grid points are incorporated into the dataset by using vincenty distance, and additional spatiotemporal features are added. 6

different experiment setting for NWP models (taking minimum, maximum, median, mean of models, taking first model only and taking all models) are tested through Light Gradient Boosting Machine (lightGBM), linear-lasso-ridge regression, decision tree, artificial neural network (ANN) and support vector machine (SVM). For hyperparameter tuning, GridSearch is used and the best score is obtained through an ANN model with the model which uses all NWP models, having a MAE of 2.01E+6 [41].

Omari et al. trained a Deep Neural Network (DNN) by using only the first ensemble model with a training dataset from 1994 to 2005, 2006 as a validation set and 2007 as testing set. Among Support Vector Regression (SVR), Keras–LeNet, and ensemble Mean Absolute Error models, the ensemble performed the best with a MAE of 2.09E+6. Both traditional models and DNNs are run on the Kaggle competition dataset. The traditional models are trained one by one for mesonets while DNN had a 98 dimensional multi-output model to predict 98 mesonets, meaning it is treated as a multi-output regression problem. On the testing dataset, SVR had 2.56E+6, DNN had 2.37E+6, and the ensemble model had 2.36E+6 MAE respectively. Ensemble models are found to be robust for the case [42].

Araf et al compared six different classification models (linear regression, ridge and lasso regressions, decision tree, random forest, and artificial neural network). While training models, they did not use elevation-related information and did not do the feature selection. A total of 75 features (15 GEFS variables observed through 5 times) are taken into account, and ensemble model outputs are averaged for a given GEFS variable. MAE is used as a performance metric, as suggested by the competition in Kaggle, and the best performing model is found to be Ridge Regression, having a MAE of 2.21E+6 [43].

Aggarwal implemented various pre-processing techniques (data cleaning, spatial pre-processing, feature segregation, data segmentation), and then implemented models based on linear least squares regression (LSR) and non-linear FFNN. Based on the comparisons, the best performance is obtained through an ensemble of LSR and FFNN [44]. Juban et al used the closest GEFS grid points as a reference. Two approaches are implemented for pre-processing, interpolating GEFS data to Mesonet sites and factoring weather data from the four nearest GEFS grid points as features for each Mesonet site [45].

Aler et al. studies the impact of the number of GEFS grids used to predict daily solar power production. SVM and Gradient Boosting Regressor (GBR) models are selected and 3 different feature selection methods are implemented to the dataset, linear correlation, ReliefF, and a new method based on local information analysis. 16 closest GEFS points are taken as reference for start and through attribute selection models, it is decreased. As preliminary steps, ensemble models are aggregated by taking the mean, mode, median, and different models; they are run by taking 5 closest GEFS points. The best MAE score is obtained by taking the mean of ensemble models. Then one of the mesonets, "ACME" station is chosen to do hyperparameter optimization for both SMO and GBR, by using 5 nearest GEFS points on 2-year validation data. The best parameters are selected based on the output of the gridsearch for further steps. The experiments involved running SMO, GBR, and RBF-SMO models using GEFS data, starting from the closest GEFS and progressing up to the 16 closest GEFS. The non-linear models outperformed the linear model significantly. The best outcomes were achieved by including observations from more than 5 nodes in the proximity of the station. Specifically, the RBF-SMO model got the lowest error using 8 GEFS sites, while GBR exhibited the lowest error when adding more grid points, 16 (though the improvement from 8 to 16 points was marginal, with only 26 percent less error). Different from expectations, feature selection did not lead to an improvement in solar energy prediction. However, with RBF-SMO, the local information algorithm managed to achieve comparable predictions using only half of the attributes. [46].

Table 3.3 AMS Competition Data Focused Studies

Author	Year	Highlights	Best Performing Model
Saad	2020	- Nearest 4 GEFS points(vincenty dist) - Spatiotemporal features - Run models for min, max, mean, median, 1st and 11th model - Linear Reg, Lasso, Ridge, Decision Tree, LightGBM, ANN, SVM - Hyperparameter tuning (gridsearch)	ANN using all ensemble members: 2.01E+6
Omari	2017	- DNN, SVR, Keras Le-Net, ensemble MAE models - Traditional models are trained one by one for each mesonet separately, DNN trained as multi-label	Ensemble model: 2.36E+6
Araf	2019	- Linear Reg, Lasso, Ridge, Decision Tree, Random Forest, ANN - Took average of hourly values at the end	Ridge Regression: 2.21E+6.
Aggarwal	2014	- Spatial processing, data cleaning, feature segregation, data segmentation - LSR, non-linear FFNN	Ensemble of LSR and non-linear FFNN: 2.41E+6
Juban	2013	- Closest GEFS points - Interpolation, factoring closest GEFS station data - Average of all ensemble models - Feature engineering (time, location and altitude features added) - Linear Reg, Lasso, Ridge, Random Forest	Random Forest: Ranked 40th (MAE haven't shared)
Aler	2015	- Feature selection (RelieF, linear correlation, new method) - Starts using all GEFS points and decreases - Ensemble models are aggregated (mean, mode, median) - SMO, GBR, RBF-SMO models - Hyperparameter optimization is done by selecting 'ACME' station as subset - RBF-SMO model got the lowest error using 8 GEFS sites, while GBR exhibited the lowest error when adding more grid points, 16	RBF-SMO: 1.93E+6

4. PROPOSED METHOD

This paper aims to test the usability of online machine learning models in the renewable energy sector, mainly for predicting production rates. As the share of solar energy is significant among other renewable energy sources, and it is commonly used all around the world, we also utilized a solar energy dataset in this thesis. We chose the solar energy competition data published by the American Meteorological Society (AMS) on the Kaggle platform. The purpose of selecting this dataset is the attention it has received from many researchers in literature and on the platform, resulting in several impressive works and the opportunity to compare them.

While developing the model, as a best practice, Cross Industry Standard Process for Data Mining (CRISP-DM) methodology is followed. It suggests following six sequential steps to set a base and standard quality for any prediction studies, as visualized in Figure 4.1. Business understanding helps to clarify what are the business objectives and how the predictive model will contribute to that. In Section 1 we explained why predicting solar energy is important and how it supports the industry. In Section 2, we covered the dynamics of solar power prediction and the place of proposed approach among other machine learning models. In Section 3, we took a closer look at the studies done in renewable energy, solar energy and the AMS Solar Energy Prediction contest, to understand the objectives of other studies, what features contributed to their models and which evaluation metrics are being used. Through these sections, we covered the Business Understanding step in CRISP-DM cycle

Data understanding refers to acquiring data to be used when building the model, verifying the quality of the data (checking missing values, ensuring the dataset can cover business scope defined in step 1). In this section, the dataset is thoroughly examined to understand these points. Data preparation and modeling steps are covered in Section 5. Once the models are created, based on the business objectives and the desired performance criteria, alternatives are tested in the evaluation step, which are also covered in Section 5. If required, parameter adjustments or further analysis is implemented to make sure the model fulfills the business needs. Deployment step covers all the activities to put models in use in live systems. Since the competition dataset is static and does not require deployment steps, our study does not include this step.

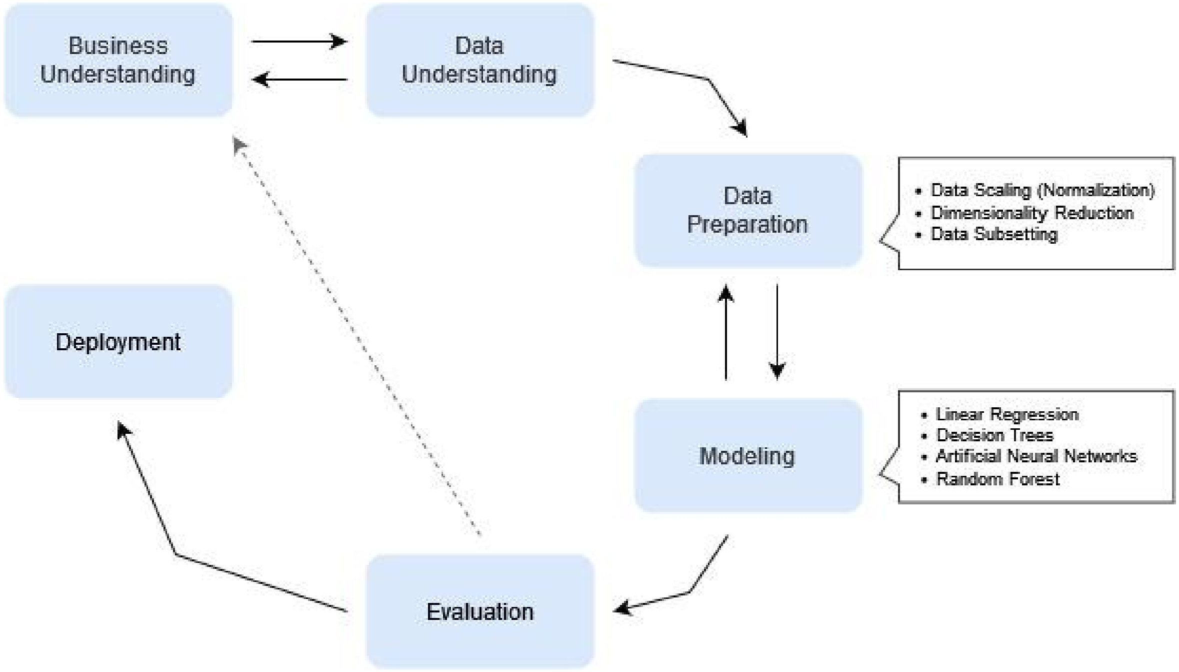


Figure 4.1 Cross Industry Standard Process for Data Mining

In the following subsections, we first explain the details of the selected dataset and then we give the information on the machine learning techniques used in the thesis.

4.1. Dataset

The dataset is open source and provided by the American Meteorological Society for 2013-2014 Solar Energy Prediction Contest. It includes 4 main files:

- gefs_train.tar.gz
- train.csv
- station_info.csv
- gefs_elevations.nc

which will be used to train models.

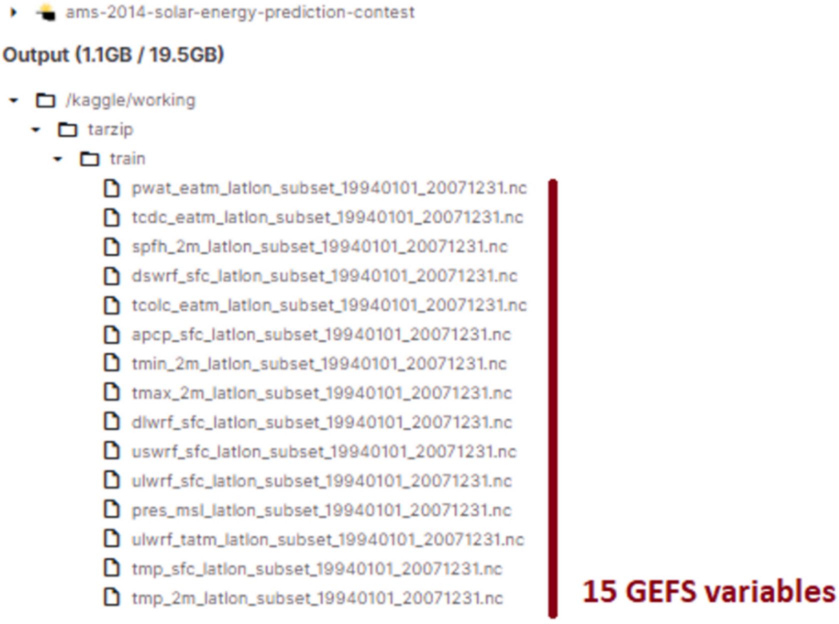
There are two other files shared to be used when testing the models:

- gefs_test.tar.gz
- test.csv

4.1.1. GEFS Weather Measurement Files (gefs_train.tar.gz, gefs_test.tar.gz)

Files containing GEFS variables, given in Figure 4.2, are in netCDF4 format and hold weather measurements (precipitation, temperature, etc.) from blue marked dots on Figure 4.3. There are 114 (9x16) GEFS locations on the given dataset. The compressed folder contains a netCDF4 file for each weather measurement (15 files in total), each having 5 dimensions:

- Date the measurement is taken
- ID number of ensemble model used to measure the given variable (ranges from 1-11)
- At what time it is measured (12:00- 15:00 - 18:00 - 21:00 - 24:00)
- Latitude of GEFS point
- Longitude of GEFS point



```
▸ ams-2014-solar-energy-prediction-contest
Output (1.1GB / 19.5GB)
▾ /kaggle/working
  ▾ tarzip
    ▾ train
        pwat_eatm_latlon_subset_19940101_20071231.nc
        tcdc_eatm_latlon_subset_19940101_20071231.nc
        spfh_2m_latlon_subset_19940101_20071231.nc
        dswrf_sfc_latlon_subset_19940101_20071231.nc
        tcolc_eatm_latlon_subset_19940101_20071231.nc
        apcp_sfc_latlon_subset_19940101_20071231.nc
        tmin_2m_latlon_subset_19940101_20071231.nc
        tmax_2m_latlon_subset_19940101_20071231.nc
        dlwrf_sfc_latlon_subset_19940101_20071231.nc
        uswrf_sfc_latlon_subset_19940101_20071231.nc
        ulwrf_sfc_latlon_subset_19940101_20071231.nc
        pres_msl_latlon_subset_19940101_20071231.nc
        ulwrf_tatm_latlon_subset_19940101_20071231.nc
        tmp_sfc_latlon_subset_19940101_20071231.nc
        tmp_2m_latlon_subset_19940101_20071231.nc
```

Figure 4.2 GEFS Files

Each file has the same five dimensions mentioned above with an array of shapes (5113, 11, 5, 9, 16). Every file includes one weather measurement for a certain day, hour, and coordinates and is predicted by an ensemble model. As mentioned in Section 2, GEFS models consist of 21 ensemble models underneath. It means, 21 different models make predictions on various atmospheric variables (can be related to pressure, temperature at different heights, precipitation etc). Outputs of GEFS models are commonly used for solar energy prediction. The reason behind having many models predict the same atmospheric variable is the complexity of the problem. Each individual ensemble model contributes process in a way and can be more/less performant when compared to others, as studied by [41]. The competition dataset provides contributors 11 of these models to use while predicting daily solar power production. Each ensemble model predicts 15 different atmospheric variables. For example, ‘pwat_eatm_latlon_subset_19940101_20071231.nc' file has data spanning 5113 days, from 1994 until 2008. It has predictions for precipitable water over the entire depth of the atmosphere, measured in kg m^{-2} . These predictions are created by 11 different ensemble models, 5 times a day (at 12:00, 15:00, 18:00, 21:00, and 00:00) for every 114 GEFS points on the map. All fifteen weather measurement variables are explained in Table 4.1.

Table 4.1 Weather Variables Contained within GEFS Files

Variable	**Description**	**Units**
apcp_sfc	3-Hour accumulated precipitation at the surface	kg m-2
dlwrf_sfc	Downward long-wave radiative flux average at the surface	W m-2
dswrf_sfc	Downward short-wave radiative flux average at the surface	Wm-2
pres_msl	Air pressure at mean sea level	Pa
pwat_eatm	Precipitable Water over the entire depth of the atmosphere	kg m-2
spfh_2m	Specific Humidity at 2 m above ground	kg kg-1
tcdc_eatm	Total cloud cover over the entire depth of the atmosphere	%
tcolc_eatm	Total column-integrated condensate over the entire atmosphere.	kg m-2
tmax_2m	Maximum Temperature over the past 3 hours at 2 m above the ground	K
tmin_2m	Minimum Temperature over the past 3 hours at 2 m above the ground	K
tmp_2m	Current temperature at 2 m above the ground	K
tmp_sfc	Temperature of the surface	K
ulwrf_sfc	Upward long-wave radiation at the surface	Vm-2
ulwrf_tatm	Upward long-wave radiation at the top of the atmosphere	Wm-2
uswrf_sfc	Upward short-wave radiation at the surface	W m-2

4.1.2. Daily Solar Power Prediction Data from Mesonets (train.csv)

In Figure 4.3, while blue dots in the map represent GEFS locations, red ones represent Oklahoma Mesonet. "Mesonet" is an abbreviation for "mesoscale network." In meteorology, "mesoscale" refers to weather occurrences with diameters ranging from one mile to 150 miles. Mesoscale events can last anywhere from a few minutes to several hours. Mesoscale occurrences include thunderstorms, wind gusts, heat bursts, and drylines. A "network" is a system that is linked together. Thus, the Oklahoma Mesonet is a system designed to measure the environment at mesoscale weather events of varying size and duration [47].

The training dataset includes the total daily incoming solar energy in (J m-2) measurements belonging to 98 stations located in Oklahoma Mesonet. The solar energy measurements are taken by a pyranometer at each Mesonet site in the Oklahoma grid. Measurements are collected in 5-minute intervals and summed from sunrise to 23:55 UTC to obtain daily solar power produced. The dates for the training set range from 1994 to 2007, and for the testing set from 2008 to 2012.

4.1.3. Station Information File

Station information file includes the latitude, longitude, and elevation of each mesonet in the Oklahoma grid. Elevation features contained in the station information file are not used for our study, as we are also not using GEFS elevation related information, which is explained in the following subsection.

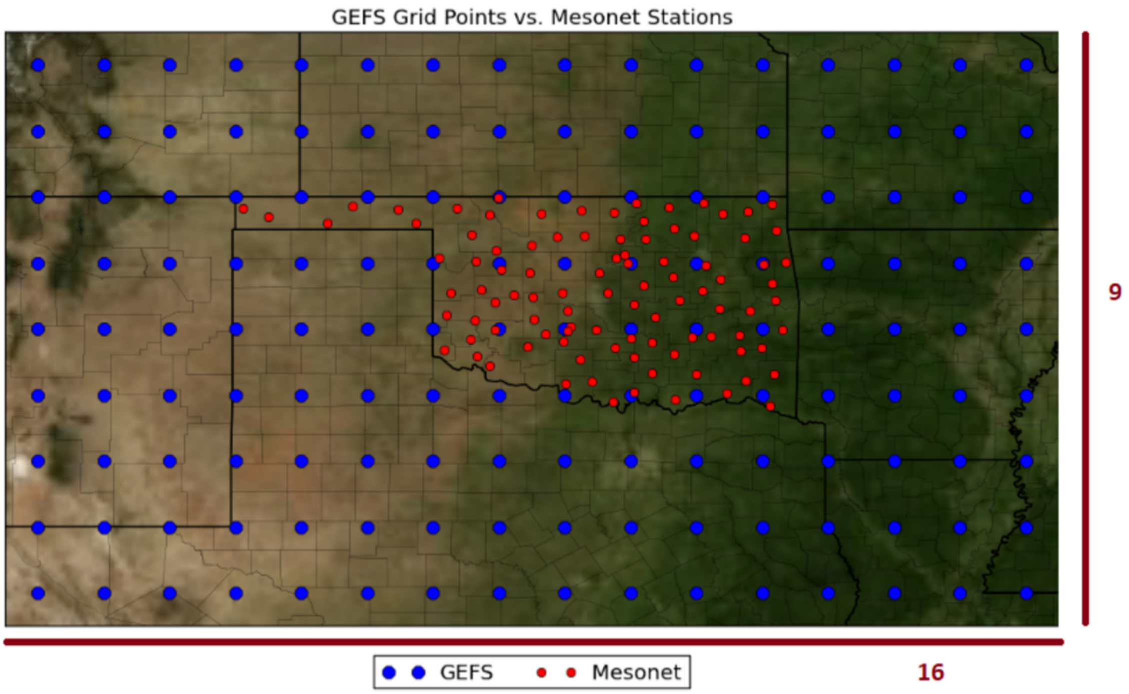


Figure 4.3 Map of Oklahoma Mesonet and Surrounding GEFS Points

4.1.4. GEFS Elevation File

Like the station information file, similar data is available for GEFS grid points in gefs_elevations.nc file. In GEFS, the model terrain undergoes smoothing which can result in a mismatch between the elevation at a given lat-lon point in the model and the actual elevation. The file includes two elevation variables:

- elevation_control: Provides the elevations for the first ensemble member (the GEFS control run)
- elevation_perturbation: Provides the elevations for the other ensemble members (the GEFS perturbations).

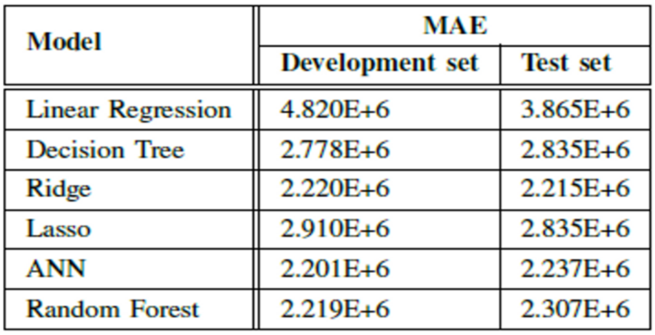
It is said that these variables can differ by up to 300 meters, and choosing one over the other could potentially affect your model. In the literature, some papers [41, 44, 45] implemented spatial interpolation methods as a pre-processing step while others [43] did not. In the scope of this work, we decided not to take elevation data into consideration.

Files shared for testing (gefs_test.tar.gz, test.csv) follow the same structure as training (gefs_train.tar.gz, train.csv) ones. The contest gives training sets (data from GEFS points, and mesonets) and expects participants to develop a model. Then use testing files to predict solar power produced by each mesonet on unknown data, gather these in a submission file, and upload them to the platform. The platform uses mean absolute error (MAE) as metric when calculating scores. It takes submissions and calculates scores, this way it is easier to see how models compare to each other.

4.2. Definition of The Proposed Method and Model Selection

To assess the effectiveness and efficiency of online machine learning models, an extensive investigation was conducted on the literature pertaining to solar power prediction, as well as previous studies on the same dataset, as summarized in the previous section. When we think about the sequential nature of the problem, one of the first models that will come to mind is time series models. Despite their applicability in renewable energy prediction, as evidenced by previous studies [30, 32], we have found no explicit implementation of time series models within the AMS Solar Power Prediction Competition. Time series models are mainly classified as physical-statistical approaches for solar power prediction [35], and the competition expects the participants to use machine learning models to create their forecasts. The aim of this study is to prove online learning based machine learning models can outperform traditional machine learning models, and we are utilizing AMS Dataset for the cause. This led us to pick a subset of more commonly used methodologies in the literature where the same dataset is used and high performance is achieved. In Table 4.2, an example study by Araf et al. [43] shows how different models perform on the selected dataset. In the scope of this work, we will also be focusing on these models. This set of models includes some simple models like LR, Ridge & Lasso, tree-based models like DT, ensemble models like RF, and finally a Neural Network. Moving forward with this set of models will help us to make a fair comparison of the difference between traditional and online training approaches.

Table 4.2 MAE of Different Regression Models Studied by Araf et al. [43]



Model	MAE	
	Development set	Test set
Linear Regression	4.820E+6	3.865E+6
Decision Tree	2.778E+6	2.835E+6
Ridge	2.220E+6	2.215E+6
Lasso	2.910E+6	2.835E+6
ANN	2.201E+6	2.237E+6
Random Forest	2.219E+6	2.307E+6

Versions of Linear Regression, Lasso Regression, Ridge Regression, Decision Tree, Random Forest and Artificial Neural Network models are trained using both traditional and incremental learning methods. During training and testing, the same pre-processing methods and chunks of data are used, and the MAE scores are recorded.

The recorded scores are checked to see if the difference between them is statistically significant or not by implementing a 2 step Design of Experiments (DOE). This comparison process is explained in more detail in Section 5. For the sake of completeness, in the following subsections, we explain the main principles and way of workings of the selected machine learning models.

4.2.1. Models

Linear Regression

Linear Regression is a supervised machine learning algorithm that is used to predict the value of a continuous dependent variable by fitting a linear relationship between the independent variables and the dependent variable. Linear regression is a simple and powerful model. It is often used in cases where the relationship between the independent variables and the dependent variable is linear [6]. The model is given as follows.

$$Y_i = f(X_i, \beta) + e_i \tag{4.1}$$

Where:

- Y_i is dependent variable
- f represents function
- X_i is independent variables
- β is unknown parameters
- e_i is Error terms

Lasso Regression

The Least Absolute Shrinkage and Selection Operator (LASSO) is a linear regression regularization technique that prevents overfitting by decreasing model coefficients. It adds to the residual sum of squares a penalty equal to the sum of the absolute values of the non-intercept beta coefficients multiplied by a parameter λ that slows or accelerates the penalty. When an optimal λ parameter is found, it can eliminate irrelevant predictors, improve accuracy, and reduce variance [48].

Ridge Regression

Ridge regression is also a regularization method used to prevent overfitting in linear regression. It regularizes the model by adding a penalty equivalent to the square of the magnitude of the coefficients [6]. The key difference between ridge regression and lasso regularization is that ridge regression always includes all of the model's features, whereas lasso regularization may exclude some. This is because the penalty for having a non-zero coefficient in lasso regularization is greater than in ridge regression. Ridge regression is often utilized when strong predictive performance is important, but lasso regularization works best when good interpretability is preferred.

Decision Tree

A decision tree is a machine learning model that uses a tree-like structure to make predictions. The tree is made up of nodes, being root, interior or leaf. Root & interior nodes represent a decision that needs to be made, while leaf nodes represent outcomes/predictions [6]. Decision trees are a powerful tool for classification and regression problems. They are relatively easy to understand and interpret, and they can be used to model complex relationships between variables.

Random Forest

Random forest is an ensemble machine learning model that uses multiple decision trees to make predictions for classification and regression tasks. Each decision tree is trained on a particular portion of the data and allowed to grow to a different size before their predictions are combined to generate a final prediction [6]. Random forests are less prone to overfit the data and more resilient to noise than individual decision trees, resulting in improved accuracy.

Artificial Neural Network

Artificial neural networks (ANNs) are machine learning models inspired by the structure and function of the human brain. They are made up of many interconnected nodes or neurons that work together to process and learn from data. The network may learn to detect complicated patterns and generate predictions or classifications by modifying the strength of the connections between neurons [6].

4.2.2 Evaluation Metric

There are many different metrics to measure the accuracy of a model, like RMSE, MAE, and MSE. To evaluate all models, Mean Absolute Error will be used within the scope of this study. The main reason behind it is that the Kaggle platform uses MAE to compare developed model performances on the test data. After deciding a set of model types and other factors, final models will be loaded there to get test scores. Formula of MAE is as follows.

$$MAE = \frac{\sum\limits_{i=1}^{n} |y_i - x_i|}{n} \tag{4.2}$$

Where:

- yi is the prediction created by the model
- xi is real-observed value
- n is the total number of data points

5. IMPLEMENTATION OF THE METHOD

In the following subsections, we explain the steps taken to implement the method to the dataset. In Subsection 5.1 all the basic pre-processing steps implemented are described. Subsection 5.2 includes multi-collinearity controls and the subsequent subsection explains how this issue is addressed. Then, Subsection 5.4 and the following describe how the experimental design is structured/implemented. Subsection 5.5 describes the post-processing steps and finally Subsection 5.6 summarizes the implementations.

5.1. Pre-Processing: Preparing Data for Analysis

The files given for the competition can be grouped into two categories. One category includes GEFS variable-related files, which will serve as features of our model, and the other category includes mesonet data, which are labels of our dataset. There are 114 GEFS points and 98 mesonet stations. These two groups must be mapped to each other so, at the end of the day a dataset with n features and a label (daily solar power prediction) can be formed to feed the model.

All GEFS files have the same dimensions apart from the one including the weather variable measurements (temperature, pressure, etc.). By flattening or aggregating them on the common dimensions (date, hour, coordinates, ensemble models), a dataset containing all 15 weather measurements can be put into the same file. Based on the setting to be used (min, max, mean of ensembles or taking ensembles separately), these files need to be pre-processed and reshaped into the desired format for the models.

5.1.1. Reshaping GEFS Files

GEFS files are presented within a zipped file for the competition. To start further processing, files need to be unzipped. Then, the NetCDF library is used to read files. As described in the previous section, all GEFS files consist of 5 dimensions: date, ensemble model id, hour, latitude, and longitude. The date dimension for each GEFS point on the map is left as it is, as the goal of the competition is to predict “daily” solar power production.

The hour dimension includes 5 measurements taken in 3 hourly intervals. This dimension is aggregated by taking the mean, to obtain one daily value for the related weather variable.

Latitude and longitude dimensions are combined into a coordinate tuple as (x,y), just so every pair represents a unique GEFS point (blue dots) in Figure 5.1. There are 114 different points within each GEFS file. To map each mesonet with a GEFS point, a distance-based method is selected which is described in detail in the following subsection. Therefore, only the points closer to Oklahoma mesonets are used, which corresponds to 36 points marked within the gray rectangle in Figure 5.1 (36 points, 4x9). Data is flattened based on coordinates, meaning, for each latitude-longitude pair, the dataset is re-structured. 36 different coordinate values became the basis for creating new rows, the remaining columns became variables for each row.

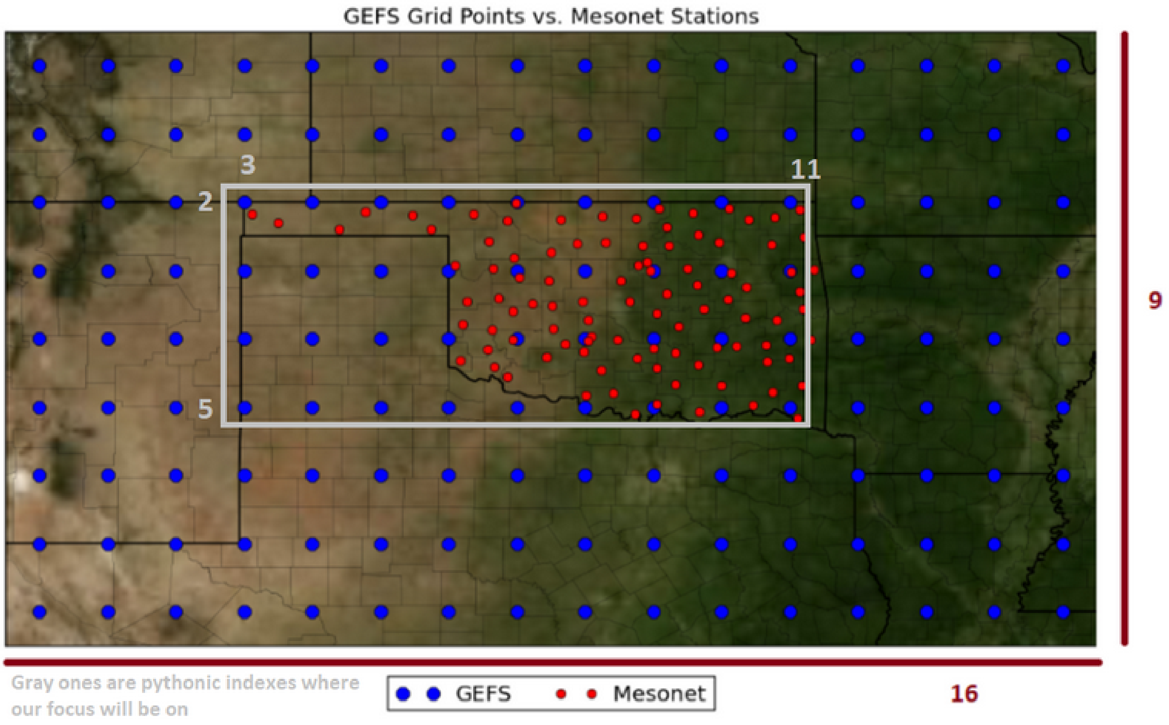

GEFS Grid Points vs. Mesonet Stations
3
11
2
5
9
16
Gray ones are pythonic indexes where our focus will be on
● ● GEFS
● ● Mesonet

Figure 5.1 Map of Oklahoma Mesonet and Surrounding GEFS Points

The ensemble model dimension is treated differently based on which model will use the pre-processed data. 4 different approaches are applied to aggregate this dimension:

- Taking all ensemble models separately (using the weather variables from only the first ensemble model, only the second, third, etc.).
    - Model 1
    - …
    - Model 11
- Taking the mean of 11 ensemble model measurements.
    - Model 12
- Taking the minimum of 11 ensemble model measurements.
    - Model 13
- Taking the maximum of 11 ensemble model measurements.
    - Model 14

Every GEFS file is processed by following the same flow and they are appended to the same dataset. The base dataset is formed by using dates and coordinates, and then the weather measurements are added as columns. Sample structures for approaches are given in Table 5.1 (for aggregated: Models 12-14, for each model we have a separate table) and Table 5.2 (for non-aggregated: Models 1-11).

Table 5.1 Aggregated Ensemble Models Dataset Example (GEFS Data)

Date	coordinates	pres_msl_1	tmp_sfc_1	…x15 GEFS variables
19940101	(33.0, 257.0)	…	…	
19940101	(33.0, 258.0)	…	…	
… Dates from 1994 to 2007	… 36 GEFS coordinates			

Table 5.2 Non-Aggregated Ensemble Models Dataset Example (GEFS Data)

Date	coordinates	pres_msl_1	tmp_sfc_1	…x15 GEFS variables	…x11 ensemble models(_1, …, _11)
19940101	(33.0, 257.0)	…	…		
19940101	(33.0, 258.0)	…	…		
… Dates from 1994 to 2007	… 36 GEFS coordinates				

5.1.2. Haversine Distance to Map GEFS and Mesonets

A distance-based method is selected to map GEFS points and mesonet stations. Based on the latitude and longitude of each point, the closest GEFS measurements are taken as features for the model. Since both points (GEFS and mesonet) are coordinates, the haversine distance method is chosen for calculation. Compared to straightforward Euclidean distance computations, which assume a flat surface, the Haversine distance formula accounts for the curvature of the Earth's surface, making it more accurate as shown in Equation 5.1 . Its formula is given below:

$$d = 2r{*}arcsin(\sqrt{sin^2(\frac{(lat_2 - lat_1)}{2}) + cos(lat_1) * cos(lat_2) * sin^2(\frac{(lon_2 - lon_1)}{2})})) \tag{5.1}$$

Where:

- d is the distance between the two points
- r is the radius of the Earth (or the sphere being used)
- lat_1 and lon_1 are the latitude and longitude of the first point
- lat_2 and lon_2 are the latitude and longitude of the second point

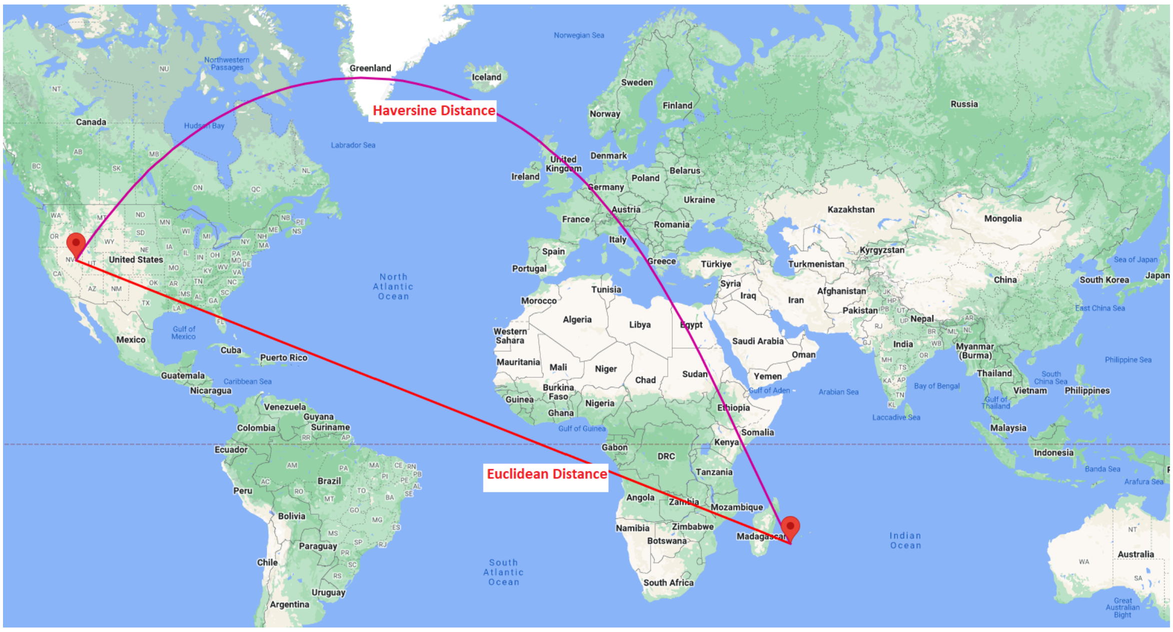


Figure 5.2 Difference Between Haversine Distance and Euclidean Distance

For every mesonet in the Oklahoma grid, this distance is calculated and results are stored in a dictionary. GEFS location coordinates (latitude, longitude) are normalized before doing the calculations as their values are greater than 180. The python library used for haversine distance calculation requires coordinates to be normalized and fit into 0-180 range. For mesonets, this pre-processing step is not needed since their values are already within desired range. Coordinates of the GEFS point having a minimum distance to related mesonet is assigned to a column as a tuple next to the station information file.

5.1.3. Mapping the Closest GEFS point to the Mesonets

Training file includes the amount of solar power produced on a given mesonet for a given date and the shape of it is (5113,98) representing 5113 days of measurements from 98 mesonet stations in the grid as shown in Table 5.3.

Table 5.3 Train.csv Dataset Example (Mesonet Data)

Date	**ACME**	**ADAX**	**…x98 Mesonet**
19940101	12384900	…	
19940102	11908500	…	
… Dates from 1994 to 2007			

The data frame is melted to have one label, daily solar production and Table 5.4 is obtained.

Table 5.4 Melted Train.csv Dataset Example (Mesonet Data)

Date	**stid (station id)**	**Daily_Production**
19940101	ACME	12384900
19940102	ACME	11908500
… Dates from 1994 to 2007 x 98 Mesonet	…x98 Mesonet	

With the help of the station name column, the melted training file and station information file (which includes mapping to the closest GEFS point based on haversine distance) (given in Table 5.5) are joined.

Table 5.5 Minimum Haversine Distance Calculated Dataset Example (Mesonet Metadata)

stid	**nlat**	**elon**	**elev**	**coord**	**min_dist_node**
ACME	34.80833	-98.02325	397	(34.80833, -98.02325)	(35.0, 262.0)
ADAX					
…x98 Mesonets					

Obtained file now includes both daily solar power prediction values, the date these measurements are taken, the station name within mesonet, and the coordinates of the closest GEFS point as given in Table 5.6.

Table 5.6 Minimum Distances - Mesonets Dataset Example (Mesonet Data)

Date	**stid (station id)**	**Daily_Production**	**min_dist_node**
19940101	ACME	12384900	(35.0, 262.0)
19940102	ACME	11908500	(35.0, 262.0)
… Dates from 1994 to 2007 x 98 Mesonet	…x98 Mesonet		

In earlier sections, GEFS files and labels have been reshaped and preprocessed accordingly. The next step is to use processed GEFS files and labels and join them based on the predetermined rule, which is referencing the closest GEFS measurements for each mesonet station. To join these two data frames, a combination of dates and coordinates are taken. Features of the resulting dataset are given in Table 5.7.

Table 5.7 Summary of Resulting Dataset

	Features	**Non-Null Count**	**Data Type**	**Type of Feature**
0	Date	501074 non-null	int64	metadata
1	stid	501074 non-null	object	metadata
2	min_dist_node	501074 non-null	object	metadata
3	Daily_Production	501074 non-null	int64	independent
4	tcolc_eatm_ensid	501074 non-null	float32	dependent
5	ulwrf_tatm_ensid	501074 non-null	float32	dependent
6	dlwrf_sfc_ensid	501074 non-null	float32	dependent
7	tmp_sfc_ensid	501074 non-null	float32	dependent
8	tcdc_eatm_ensid	501074 non-null	float32	dependent
9	dswrf_sfc_ensid	501074 non-null	float32	dependent
10	tmax_2m_ensid	501074 non-null	float32	dependent
11	tmin_2m_ensid	501074 non-null	float32	dependent
12	pwat_eatm_ensid	501074 non-null	float32	dependent
13	uswrf_sfc_ensid	501074 non-null	float32	dependent
14	spfh_2m_ensid	501074 non-null	float32	dependent
15	ulwrf_sfc_ensid	501074 non-null	float32	dependent
16	tmp_2m_ensid	501074 non-null	float32	dependent
17	apcp_sfc_ensid	501074 non-null	float32	dependent
18	pres_msl_ensid	501074 non-null	float32	dependent

GEFS table containing weather measurements is joined with Table 5.6 as a last step for basic pre-processing steps, resulting in Table 5.8 where we both have features and labels in one data frame.

Table 5.8 Resulting Dataset Example After GEFS and Mesonets Mapped

Date	stid	min_dist_node	Daily_Production	…x15 GEFS variables
19940101	ACME	(35.0, 262.0)	12384900	
19940101	ACME	(35.0, 262.0)	11908500	
… Dates from 1994 to 2007	… x98 Mesonets			

Table 5.8 is used as input for upcoming pre-processing steps which are covered in subsequent sections. By taking GEFS files and Mesonet data, and implementing the basic pre-processing steps described above, we obtain a dataset that will be used for checking multicollinearity and dimensionality reduction. The steps are summarized in below Figure 5.3, where blue color corresponds to raw files, green corresponds to the intermediary datasets, and orange is the output dataset. Transformation logic and functions are described in arrows.

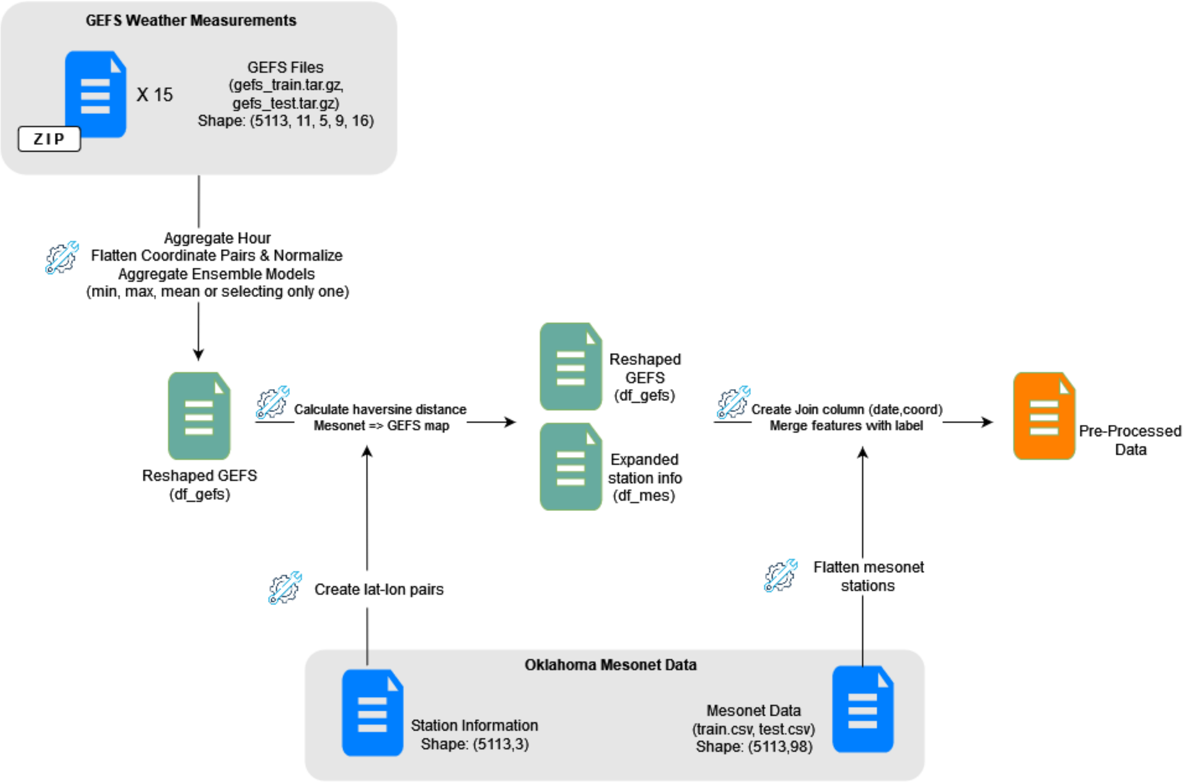


Figure 5.3 Process Flow Map for the Basic Pre-Processing Steps

5.2. Pre-Processing: Checking For Multicollinearity

Multicollinearity happens when there is a strong correlation between two or more independent variables in a statistical model. As a result, it is challenging for the statistical model to discriminate between the impacts of the various independent variables. This is because the independent variables are measuring similar or related features of the dependent variable. It can reduce model stability, increase standard errors and inflate p-value scores. Therefore it is important to check against multicollinearity and take action to create a better model. Looking at different GEFS weather variables within the dataset, we can see the possibility of multicollinearity. An example of a possible correlation can be between temperature-related variables (maximum - minimum - surface temperatures) and pressure, as the temperature increases, air pressure decreases. To statistically test this hypothesis, variance inflation factors (VIF) are calculated and the results are given in Table 5.9. Many VIF values turned out to be greater than 5, which indicates multicollinearity within independent variables [49]. Another common limit is 10, which is less strict when compared to selecting 5. Given the scores, selecting either 5 or 10 does not make a big difference.

Table 5.9 Variance Inflation Factor Calculation Results

Feature	Description	VIF
const	NaN	296637.20
apcp_sfc_0	Total_precipitation	2.56
pres_msl_0	Pressure	1.84
tmp_sfc_0	Temperature_surface	558.94
tmin_2m_0	Minimum_temperature	227.85
spfh_2m_0	Specific_humidity_height_above_ground	13.97
dlwrf_sfc_0	Downward_Long-Wave_Rad_Flux	36.53
uswrf_sfc_0	Upward_Short-Wave_Rad_Flux	6.75
ulwrf_sfc_0	Upward_Long-Wave_Rad_Flux_surface	481.68
tcdc_eatm_0	Total_cloud_cover	64293.62
tmax_2m_0	Maximum_temperature	640.80
pwat_eatm_0	Precipitable_water	13.61
ulwrf_tatm_0	Upward_Long-Wave_Rad_Flux	3.80
tcolc_eatm_0	Total_Column-Integrated_Condensate	64149.05
dswrf_sfc_0	Downward_Short-Wave_Rad_Flux	9.87
tmp_2m_0	Temperature_height_above_ground	1116.70

To address the issue, highly correlated variables can be removed one by one (Backward Elimination), they can be selected iteratively while controlling model explainability and VIF scores (Forward Selection) or automated methods like Principal Component Analysis (PCA) can be implemented. The iterative methods (Backward Elimination and Forward Selection) requires taking a feedback-based set of actions. On the other hand, PCA allows us to perform the operation once, and determine the number of features needed to explain the data. By using this number, all subsequent model variations can be trained smoothly and automatically. Through different phases, we will be running many model variations. To have a generalized and consistent method in this thesis, we decided to use PCA.

5.3. Pre-Processing: Dimensionality Reduction by PCA

Principal Component Analysis (PCA) is a statistical approach for reducing the dimensionality of a dataset by transforming the original variables into a different set of uncorrelated variables known as principal components. A principal component is a new variable that is created as a linear combination of the original variables in a dataset. The idea behind principal components analysis (PCA) is to reduce the dimensionality of a dataset while preserving as much of the variability in the data as possible.

PCA projects the data onto these new axes after determining the directions in which the data changes the most. The highest amount of data variance is captured by the first principal component, with successive components collecting progressively less variance. Within the scope of this work, cumulative explained variances are plotted on a graph given in Figure 5.4 to determine how many principal components should be produced.

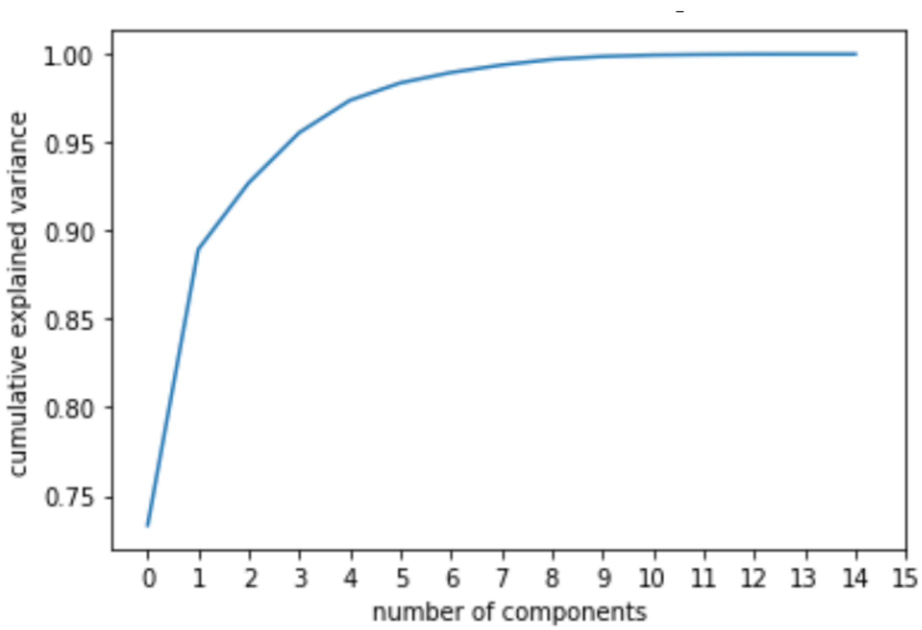


Figure 5.4 Cumulative Explained Variance for Principal Component Analysis

The purpose of PCA is to find a set of new variables (the principal components) that capture as much variability in the original dataset as possible. The variance of a dataset measures how spread out the data points are from the mean. In a dataset with high variance, the data points are spread out widely from the mean, while in a dataset with low variance, the data points are clustered more tightly around the mean. A cumulative explained variance graph helps us determine how many principal components we should have, so while reducing the dataset dimension, we will not lose too much information.

Each principle component is a linear combination of the original variables that are chosen to explain the greatest portion of the variance. In our dataset, essentially there are 15 different weather measurements. Before implementing PCA, data is processed to aggregate hour dimensions as well as ensemble models, by taking the mean across 11 models for a given day. To determine the optimal number of principal components for our dataset, the first step is to draw the cumulative explained variance graph. In the graph, an "elbow" or "knee" can represent a suitable stopping point for retaining principal components. After this point, adding extra components does not contribute much to the overall explained variance. The elbow point might be determined as the place where the slope of the curve changes dramatically, or a bit above that to ensure not much data is lost. For this scenario, in order not to lose information, a point where the slope of the line significantly decreases is selected, which corresponds to 5 principal components, explaining 97,4% of the variability in the dataset.

5.4. Mining: Comparison of Traditional and Online ML Models

As described in Section 4.2, six models are used, and different versions (with respect to ensemble model used, the inclusion of PCA or not, being traditional or online) are tested. Summary of experiments are listed in a table similar to the template given in Table 5.10 in which tr_loss and o_loss represent traditional and online learning losses for the training set, respectively (complete set of records are given in the Appendix 1).

Table 5.10 Record of Model Runs for DOE-1

id	**pca**	**ens_id**	**Linear Regression**		**Lasso Regression**		**… All models**
			tr_loss	**o_loss**	**tr_loss**	**o_loss**	
1	0	1	3,617,980.42	3,595,331.85	3,623,018.41	3,595,331.66	
…	0	…	…	…	…	…	
14	0	14	3,785,272.84	3,701,351.90	3,785,273.16	3,701,351.98	
15	1	1	3,659,116.79	3,563,595.06	3,659,117.09	3,563,595.14	
…	1	…	…	…	…	…	
28	…0, 1	…1-14					

Ensemble model dimension of the above table represents the ensemble model used to predict GEFS variables. Models are trained and tested with the following approaches:

- By taking measurements from every ensemble model (there are 11 different models in total) one by one (ex: Linear regression model is trained by using only the measurements coming from the first ensemble model, then the second, repeating the process until all ensemble models are covered.),
- By taking the **mean** of all 11 ensemble measurements,
- By taking the **maximum** of all 11 ensemble measurements,
- By taking the **minimum** of all 11 ensemble measurements.

Each approach in the above list is tested for its sensitivity to principal component analysis. All models are run both with PCA and without PCA implemented, fixing the number of features to 5, which was the suggested number of features based on cumulative explained variances, Figure 5.4.

The training dataset given for the competition is first re-shaped and normalized with Min-Max Normalization. Then it is portioned according to the approach which is currently being used. Data is normalized for all of the approaches, and %80 of the data is used for training while the remaining %20 is used for testing. In total, 56 models are run for each machine learning model mentioned in Section 4.2. Each model run is recorded as given in Table 5.10 where tr_loss holds the MAE values for models trained with traditional fashion, and o_loss holds the ones trained with incremental learning. All machine learning models are tested for their sensitivity to principal component analysis, ensemble model selection strategy, and most importantly, the training approach (traditional versus incremental) by using the run results obtained like in Table 5.10. The design of experiment (DOE), which will be covered in more detail in Section 5.4.3, is utilized for this purpose.

5.4.2. Designing Comparison Methodology for Models

As discussed in Section 5.4.1, the MAE scores obtained from model runs are recorded to be statistically compared to observe differences between traditional and online learning models. To facilitate this analysis, a variety of statistical tests are considered. Initially, the Paired-T Test is found to be a viable option. A paired samples t-test is used to compare the means of two samples when each observation in one sample can be paired with an observation in the other sample. The test has three main assumptions:

1. Independence: Observations should be independent of each other
2. Normality: The difference between pairs should be normally distributed
3. No extreme outliers: There should be no extreme outliers in differences.

By taking these into consideration, the normality of the differences is tested through the Anderson-Darling test. The following hypotheses are formed.

H_0: The data follows the normal distribution

H_1: The data do not follow the normal distribution

Based on the results, p values are found to be less than 0.05, which directs us to reject the null hypothesis. Meaning, data do not follow a normal distribution. To normalize differences, Box-Cox Transformation is implemented and rounded lambda values are used for transforming data. Even after transformation, the normality assumption couldn't be met. An example is shown in Figure 5.5, Figure 5.6, and Figure 5.7.

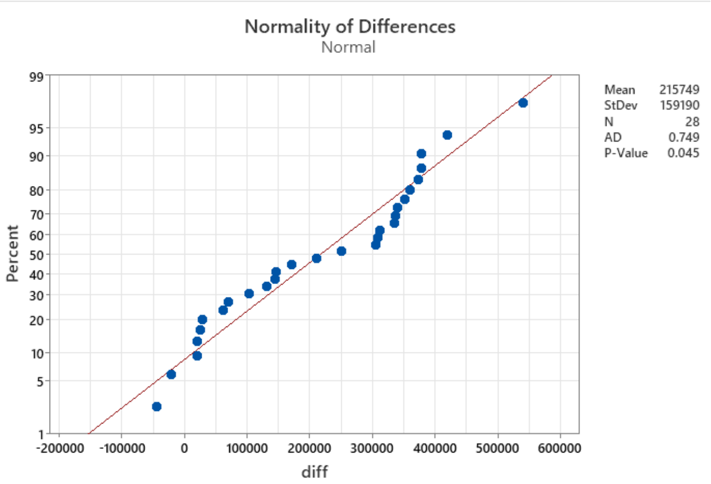


Figure 5.5 Normality Plot of Model Score Differences

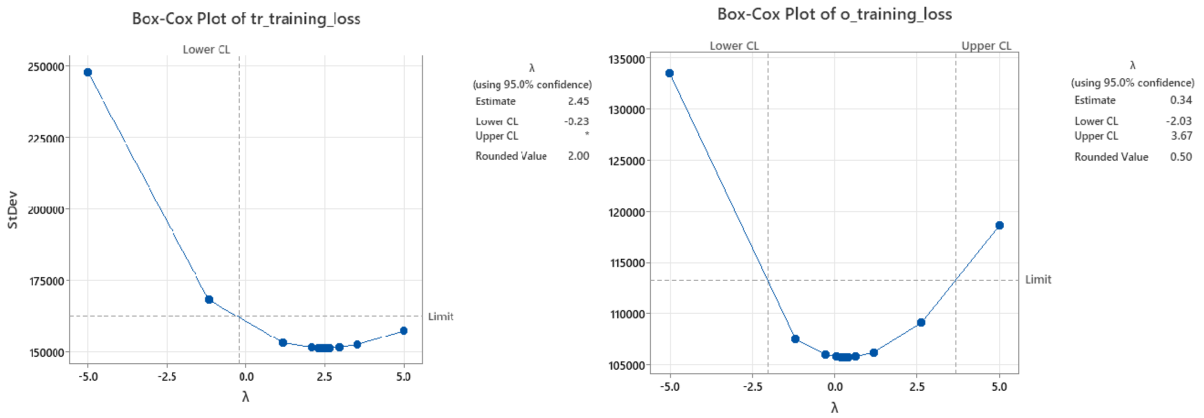


Figure 5.6 Box-Cox Transformation Results

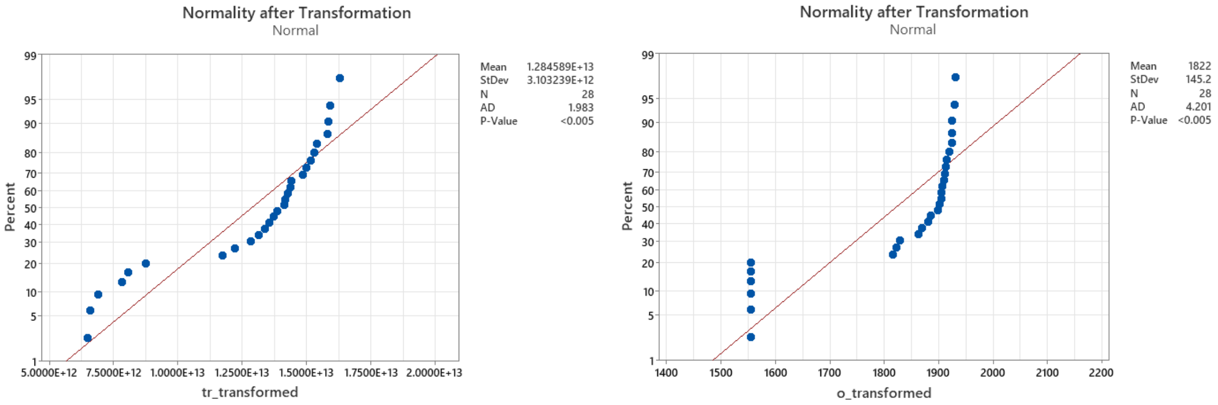


Figure 5.7 Normality Plots after Transformation

The reason behind that is the test setup includes various settings related to type, pre-processing, and ensemble model selection. Despite comparing the same model for online training and traditional training pairwise, more than one factor is changing across all runs, which makes it harder to distinguish how factors contribute to the scores. A paired-T test would be useful to see if the difference between training approaches is significant or not, however, it does not provide any information regarding the impact of different factors, such as ensemble model selection strategy, PCA implementation, and model type. For this purpose, the Design of Experiment (DOE) is implemented.

5.4.3. DOE-1: Selecting Best Factor Combination for Models on Training Data

The term "design of experiments" (DOE) refers to an area of applied statistics that focuses on the planning, carrying out, analyzing, and interpreting controlled experiments to determine the variables that affect the value of a parameter or set of parameters. DOE is an effective tool for collecting and analyzing data that may be applied in a range of experimental settings. It enables the manipulation of numerous input variables to determine their impact on a desired result (response). DOE can find significant interactions by adjusting several variables simultaneously that might be overlooked when experimenting with a single element at a time. Either each potential combination can be looked into (full factorial) or just some of them can (fractional factorial) [50].

For the thesis, a full factorial setting for DOE is implemented and checks at most two-way interactions. The experiment includes two steps, DOE-1 and DOE-2. The first step will be explained in this subsection while the second one will be discussed in the next subsection. In the first step (DOE-1), DOE is implemented for all selected predictive models, to assess the model performances by only using training data (1994-2007) and to select the factors contributing the score best. While training the models, 80% of the data is used as training and the remaining 20% is used for testing. Logic of the DOE-1 is as follows:

1. Calculate training MAE scores for all model settings.
    a. Carry out preprocessing steps based on model aggregation approach.
    b. Implement PCA if the model setting includes PCA steps.
    c. Create training & testing splits and train the model using the training portion (data: 1994-2007).
    d. Using the trained model, get MAE on the testing portion.
    e. Record the result and model parameters.
2. Select the factor combination that yields the best performance in training data for different model types.
    a. Select model types one at a time (Linear Regression, Lasso Regression, Ridge Regression, Decision Tree, Random Forest, ANN) and perform DOE on each one separately.

b. Create factor and interaction plots, comment on the results and significance of the experiment.
c. Select the factor combination to get the best MAE score on the training data.

For the DOE-1, as mentioned above, each predictive model is executed separately and results are recorded to a table template given in Table 5.10. Factorial DOE setting is used with factors and levels given in Table 5.11. Response of the experiment is selected as model performance, which are the training MAE scores. We included interactions up to two into the model, meaning we will assess the effects of two-way interactions on performance. Additionally, the experiment is set to implement Box-Cox transformation with optimal lambda and a two sided confidence level of 95%. Factorial plots are also drawn for each predictive model and are used for interpreting contributions of different factor combinations on different predictive models.

Table 5.11 Factors, Levels and Values for DOE-1

Factors	**Levels**	**Values**
type	2	traditional, online
pca	2	0 (do not implement PCA), 1 (implement PCA)
ens_id	14	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14

By using the setting described above, all DOE analyses are conducted and p-value results are recorded in Table 5.12.

Table 5.12 P-value Result Summary for DOE-1, Training Loss

	Predictive Models					
Factors	**Linear Regression**	**Lasso Regression**	**Ridge Regression**	**Decision Tree**	**Random Forest**	**ANN**
type	0.825	0.119	0	0	0.065	0
pca	0	0	0	0	0.606	0
ens_id	0	0	0	0	0	0
2-Way Interactions	0.203	0.16	0	0.003	0.004	0
- type*pca	0.001	0.009	0	0	0	0
- type*ens_id	0.589	0.67	0	0.413	0.014	0
- pca*ens_id	0.452	0.109	0.644	0.767	0.7	0.969
R-Squared	99.6	99.77	99.99	99.84	99.08	99.95

The R-Squared values for all models are greater than 99, which indicates that the model has a very high degree of explanatory power. It is able to capture a large proportion of the variation in the data.

Linear Regression

For Linear Regression models, in main effects, p value for type is 0.825, indicating type alone is not significant on model performance in the training set. However, implementation of PCA and the selection of ensemble models are found to be significant. In two-way interactions, we can see that the combination of type and pca is significant for the score. By keeping this in mind, the factorial plots for both main effects and two-way effects are analyzed.

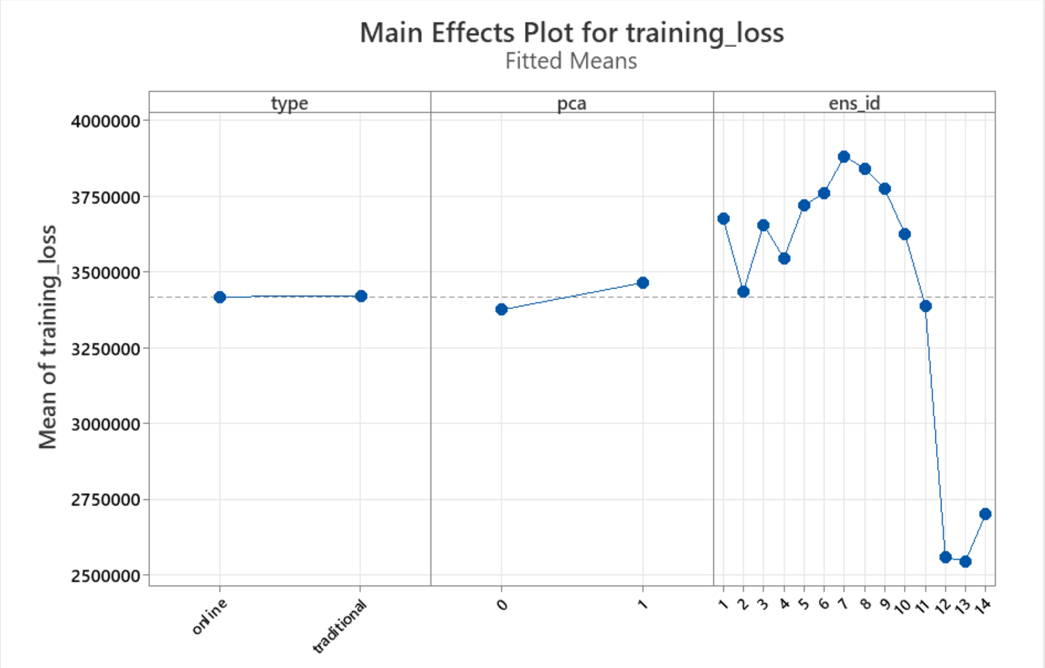


Figure 5.8 Main Effects Plot for Linear Regression - Training Data

From the main effects plot, it can be seen that type of training performances for online and traditional models are very close to each other. Based on the training set, not implementing PCA seems to yield lower training loss. Among 14 different ensemble handling approaches, taking the mean or minimum of all models seems to provide better results.

While configuring DOE, we included testing significance of interactions up to 2-way (type*pc, type*ens_id, pca*ens_id) between factors. According to ANOVA results summarized in Table 5.12 and interaction plot given in Figure 5.9, type and PCA implementation is significant for model performance when they are combined. However, other interactions are not important as the p value for them is greater than 0.05.

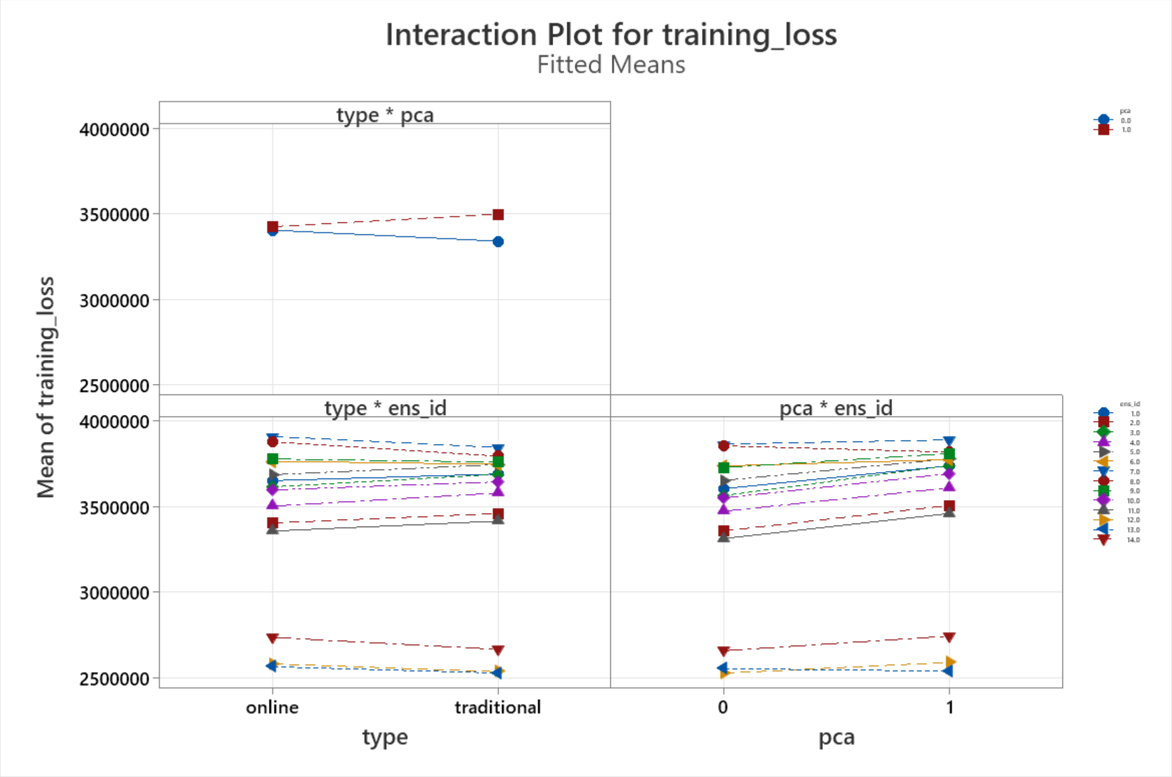


Figure 5.9 Interaction Plot for Linear Regression - Training Data

Linear regression models are trained with default settings, taking approximately 10 seconds to run when PCA is not implemented, and 5 seconds when implemented for traditional training. Training duration for the online model is 25 seconds when PCA is not implemented and 48 seconds when implemented. In light of all this data, it can be said that the best performance on training data for a Linear Regression model is obtained through implementing a traditional training approach, not doing PCA and using a minimum of ensemble models.

Lasso Regression

The pca, and ens_id factors have p-value of 0, indicating that they are significant in explaining the response variable. Type has a high p-value of 0.119, making it statistically insignificant as it is greater than 0.05. Additionally, the two-way interaction shown in Appendix 2.2, between type and pca, has a low p-value of 0.009, which suggests that this interaction is significant and should be taken into account when interpreting the results.

However, the interaction between type*ens_id has a very high p-value showing this interaction is not significant for the model. pca*ens_id also has a p-value greater than 0.05 indicating that this interaction may not be significant depending on the context.

Lasso regression models are built by taking the regularization parameter (l1) as 0.5, took approximately 6 seconds to train without PCA and 5 milliseconds when PCA implemented for the traditional training approach. For the incremental training approach the durations increase to 37 seconds without PCA implementation and to 65 seconds when implemented. Best performance for Lasso regularized regression model can be obtained by training a model with an online learning approach, not implementing PCA and taking the mean of ensemble models.

Ridge Regression

All main factors and interactions of type*pca, type*ens_id found to be significant by the model. However, pca*ens_id has a high p-value, indicating this interaction is not important for the model. From factor plots in Appendix 2.3, major difference of performance is observed especially for model training type, indicating performance of online training is worse when compared to traditional method on training data.

Training durations for traditional models are 6 milliseconds when PCA is not implemented and 2 milliseconds when implemented. Incrementally trained models have training duration of 27 seconds when PCA is not implemented and 1 minute when implemented. While training models, a regularization parameter (l2) of 0.5 is used for both online and traditional models. Best performance for Ridge regularized regression model can be obtained by training a model with a traditional approach, not implementing PCA and taking a minimum of all ensemble models.

Decision Tree

All main factors, type, PCA and ensemble id, along with interaction of type and PCA are found to be significant for the model. Interactions of type*pca and pca*ens_id have p-value greater than 0.05, which means they are not significant for the model performance and might not be considered. From the factor plots in Appendix 2.4, it can be clearly seen online training approach performs better for Decision Tree.

Decision Tree models take approximately 3 minutes when PCA is not implemented and 1 minute when implemented for the traditional approach. Incrementally trained models take 90 seconds for training when PCA is not implemented and 45 seconds when implemented. For both training approaches, models are constrained to have a maximum depth of 5. Overall, best performance on a training set can be achieved by using an online training approach, not implementing PCA and using the mean of all ensemble models for training.

Random Forest

For the performance of a Random Forest the effect of PCA alone seems to be insignificant given its high p-value of 0.6. Type has relatively low p-value, 0.06, and might be potentially significant. However interaction of type and PCA is significant for the model. Ensemble model selection and interaction of type*ens_id also seems to be a significant factor combination for model performance. When we look at main factor plots in Appendix 2.5, it seems like traditional models are slightly better when compared to online. In the interaction plots, online Random Forest yields better outcomes with PCA implemented. Also interaction of type and ensemble model selection tells us the lowest training loss is obtained through online approach using mean of all ensemble models.

Similar to Decision Tree models, Random Forest models are also constrained to have a maximum depth of 5 and a total of 3 ensemble members. Random Forest models take approximately 3 seconds to train when PCA is not implemented and 1.5 seconds when implemented for traditionally trained models. Incremental models take 12 minutes when PCA is not implemented and 10 minutes when implemented. It is important to note that traditional models support parallel processing, meaning the ensemble members of Random Forest can be trained parallelly at the same time, while this option is not available for the incremental model. Best outcome for Random Forest on training data can be obtained by following online training, not implementing PCA and taking mean ensemble models.

ANN

For ANN model performance, all main factors and interactions are significant. From the main effect plots, it can be seen online models perform significantly better when compared to the traditional ones. Not implementing PCA seems to yield lower training loss, also selecting aggregation methods that incorporate all ensembles (taking minimum, mean or maximum of models) contribute to model performance. Based on interaction plots in Appendix 2.6, it can be seen that online models have less sensitivity to PCA implementation when compared to traditional ones.

ANN models are the longest to train both for traditional and incremental approaches. For the traditional approach, batch size that is equal to the training portion of data (400,859 rows) is used with an epoch number of 15,000. For the online model, batch size is selected as 1 with an epoch number 10. Both ANN models has 4 layers:

- Input layer
- Layer 1 (50 neurons, relu activation)
- Layer 2 (25 neurons, relu activation)
- Output Layer (1 neuron, linear activation).

Models are compiled by using adam optimizer and MAE as a loss function. Models took approximately 45 minutes when PCA was not implemented and 90 minutes when implemented. For the incremental training version, model training times took approximately 40 minutes when PCA was not implemented and 70 minutes when implemented. Best performance on training data for ANN models can be obtained through following online training, not implementing PCA and taking minimum of ensemble models.

Based on the first step of DOE, suggested factors are summarized in Table 5.13.

Table 5.13 Suggested Factor Combinations by DOE-1

Suggested Factors by DOE-1	**Linear Regression**	**Lasso Regression**	**Ridge Regression**	**Decision Tree**	**Random Forest**	**ANN**
type	traditional	online	traditional	online	online	online
pca	0	0	0	0	0	0
ens_id	min	mean	min	mean	mean	min

Based on DOE-1 of the process, most of the time online training approaches seem to yield better scores on training data, only exceptions being Linear Regression and Ridge Regression. Looking at the factor plots for Linear Regression, we can see that the difference in training loss is not as significant when comparing traditional and online training approaches. Models trained with traditional approaches seem to outperform online models for Ridge Regression. This poor performance in incremental training could be related to the difficulty identifying the optimal regularization parameter, as well as potential noise or bias introduced by the incremental process. When we look at the Lasso Regression model, we don't see this problem. In fact, Lasso Regression model performed better with online training approach due to its sparsity-inducing penalty, which allows to reduce the impact of irrelevant or redundant features resulting in a more interpretable model. Compared to the L2 regularization penalty used by Ridge Regression, the L1 regularization penalty of Lasso Regression may be more adapted to the noise and bias produced in incremental training. Also, it is useful to keep in mind that in DOE-1, minimum hyperparameter configuration is made on models, which could impact the model performances. As the thesis focuses on demonstrating the applicability of an online training approach for solar power prediction, an online approach will be selected for all predictive models in the next step (even if it is not DOE-recommended in this step). When PCA is not used, all of the models appear to have lower MAE in training data. Given the potential risk of overfitting, models will be run in the following step with and without PCA to determine how they affect testing data.

Ensemble selection strategies that aggregated all models outperformed others for all models. Scores obtained by taking the minimum, maximum, or mean of all ensembles tend to be close. Therefore ensemble selection factor’s level will be dropped to 3, including minimum, maximum and mean settings for the upcoming step.

5.4.3. DOE-2: Selecting Best Performing Model on Testing Data

The second part of the procedure involves incorporating various machine learning models into the picture to see how they compare to one another. All of the models (Linear Regression, Ridge Regression, Lasso Regression, Decision Tree, Random Forest, and ANN) are trained incrementally in this step. Models run with and without PCA, by using a different ensembling aggregation as mentioned in the previous subsection. For the selected model types and factors, the models that are trained in step one are used to create predictions for the test data spanning 2008-2012.

The logic of the second step is as follows:

1. Get test MAE scores for the pre-trained & selected models in the DOE-1.
    a. Carry out the same preprocessing steps with training data.
    b. Implement PCA if the model setting includes PCA steps.
    c. Create predictions for each mesonet for the testing data and record them to submission file (data: 2008-2012).
    d. Submit results to Kaggle and obtain a competition MAE score.
    e. Record the result and model setting.
2. Perform single DOE for all recorded runs.
3. Create factor and interaction plots and comment on the results.
4. Select the best performing model among all final models.

Table 5.14 Factors, Levels and Values for DOE-2

Factors	Levels	Values
model	6	Linear Reg, Lasso, Ridge, Decision Tree, Random Forest, ANN
pca	2	0, 1
ens_id	3	12, 13, 14

Testing Logic and Creating Submission Files

For this step, trained models are submitted to the Kaggle platform to obtain testing scores. gefs_test.tar.gz file that contains GEFS variable related observations from 2008 to 2012 is processed with the same functions used for processing training data, described in Section 5.1, 5.2 and 5.3. While forming training data, it was convenient to use a join column and map GEFS variables to mesonets by that. For testing, the mesonet file only includes date dimensions. A sample submission file is provided by the competition and participants are expected to provide their predictions by abiding to that. The following logic is designed to create predictions for mesonets in the same format as the sample submission file provided by AMS Competition. The steps of testing logic:

1. Pre-process netCDF file including test portion of GEFS variables.
2. Take a mesonet
3. Get coordinates of GEFS point having a minimum haversine distance to that by using previously created df_mes file
4. Filter pre-processed GEFS variables based on extracted location in step 3
5. Scale the filtered data
6. Implement PCA if model is trained with PCA
7. Give processed test data to trained model and create predictions
8. Append predictions to submission file
9. Repeat steps 2-8 until all mesonets are covered
10. Export submission file to csv format without indexes
11. Upload the file to Kaggle and get your score.

Runs of models are submitted and their scores are recorded as given in Table 5.15 (Whole set of runs are shared in Table 5.17 while discussing model performances). Then Full Factorial DOE with the same settings described as above is conducted.

Table 5.15 Record of Model Runs for DOE-2

model	pca	ens_id	test_loss
LR	0	12	10,890,769.92
LR	0	13	7,803,050.96
LR	0	14	10,626,151.45
LR	1	12	6,719,117.59
LR	1	13	6,724,714.46
LR	1	14	6,716,157.54
Lasso	0	12	10,890,555.13
…	…	…	…
All Models	0, 1	mean, min, max	

Results of the DOE-2 are summarized in Table 5.16. R-squared value for the model is 99.39%, which means the explanatory power of the model is high.

Table 5.16 Result Summary for DOE-2 Experiments - Test Loss

Factors	P-Values
model	0
pca	0
ens_id	0.108
2-Way Interactions	0
- model*pca	0
- model*ens_id	0.626
- pca*ens_id	0.584
R-Squared	99.39

Model and PCA implementation are found to be significant contributors for the test loss. Interaction of model*pca is also found significant for the model performance. Based on the main effects plot given in Figure 5.10, it can be seen that best performance scores in testing data is obtained through Decision Tree and Random Forest models. Lasso and Linear regressions seem to have very similar testing scores, similar to their training scores. On the other hand, Ridge Regression performed better than them despite having worse training scores.

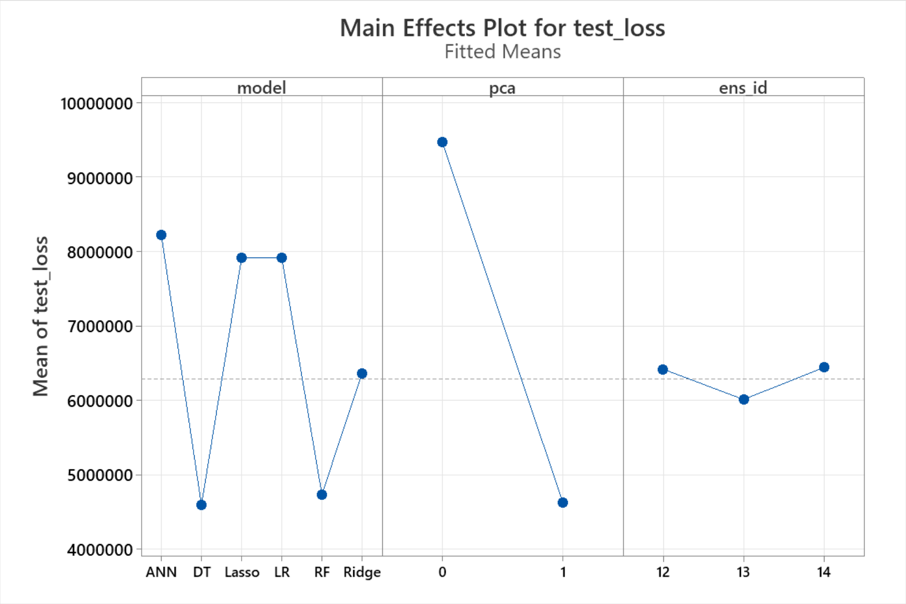


Figure 5.10 Main Effects Plot for DOE-2 - Test Data

Additionally, potential risk for overfitting seems to be on point given the testing score differences. PCA implementation drastically improves model performance on testing data. Taking a minimum of all ensemble models seems to yield slightly lower loss when compared to other strategies.

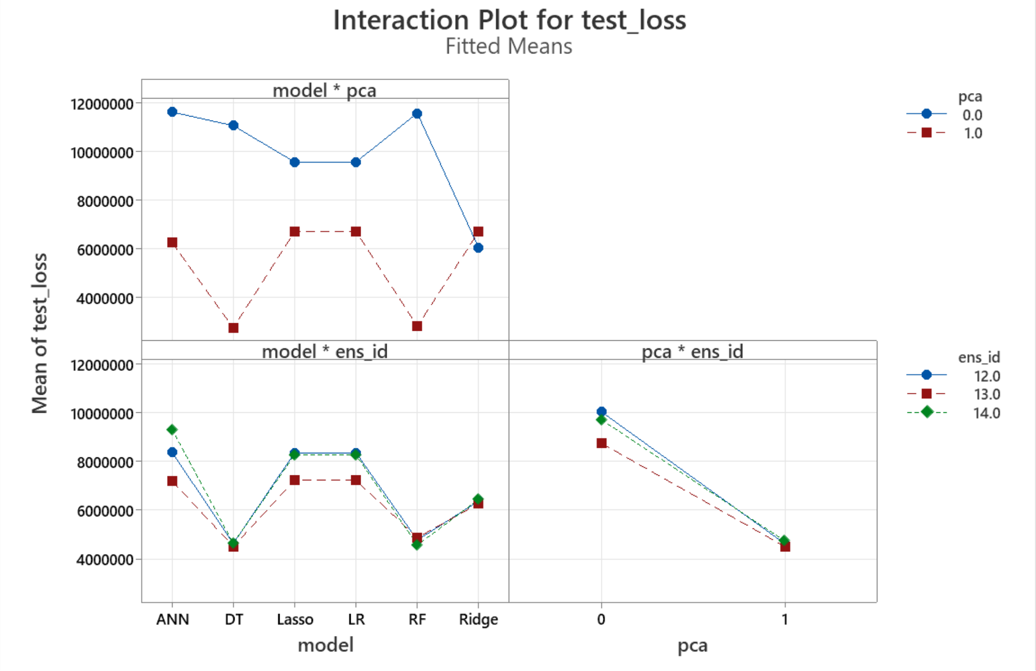


Figure 5.11 Interaction Plot for DOE-2 - Test Data

Decision Tree and Random Forest models seem to be less sensitive to ensemble model selection, especially when PCA is implemented. Based on the second step, best testing performance can be achieved by using either Decision Tree or Random Forest models, with implementing PCA and taking minimum of the ensembles. Test scores for Random Forest and Decision Tree are very close to each other, we decided to move forward with Random Forest because it is an ensemble model and it consists of more than one Decision Tree model. Which makes them stronger and more robust for overfitting.

In DOE-1 we ran models on training data and eliminated some of the factors based on results. During DOE-2, we used pre-trained models for the selected factors, created predictions on test data and submitted them to the Kaggle platform. Therefore, for the models in DOE-2, we have both training and testing scores as given in Table 5.17.

Table 5.17 Training and Testing Score Comparison of Online Models in DOE-2

model	pca	ens_id	train_loss	test_loss
ANN	0	12	2,305,371.25	11,833,125.04
ANN	0	13	2,326,162.50	10,807,474.93
ANN	0	14	2,326,672.25	12,335,264.74
ANN	1	12	2,370,540.00	6,412,608.16
ANN	1	13	2,351,943.75	5,320,625.94
ANN	1	14	2,431,919.50	7,429,573.03
Decision Tree	0	12	2,415,909.63	11,343,658.28
Decision Tree	0	13	2,415,909.63	10,366,617.10
Decision Tree	0	14	2,415,909.63	11,551,432.77
Decision Tree	1	12	2,446,872.69	2,770,233.07
Decision Tree	1	13	2,465,797.81	2,757,288.46
Decision Tree	1	14	2,513,809.64	2,762,286.46
Lasso	0	12	2,439,585.15	10,890,555.13
Lasso	0	13	2,589,781.79	7,803,350.91
Lasso	0	14	2,581,816.40	10,626,075.55
Lasso	1	12	2,560,258.81	6,719,117.61
Lasso	1	13	2,510,715.02	6,724,714.48
Lasso	1	14	2,721,861.45	6,716,157.56
Linear Regression	0	12	2,599,313.97	10,890,769.92
Linear Regression	0	13	2,615,338.00	7,803,050.96
Linear Regression	0	14	2,748,216.01	10,626,151.45
Linear Regression	1	12	2,560,258.84	6,719,117.59
Linear Regression	1	13	2,510,714.89	6,724,714.46
Linear Regression	1	14	2,721,861.61	6,716,157.54
Random Forest	0	12	2725262.39	12,409,702.95
Random Forest	0	13	2854325.73	13,321,086.17
Random Forest	0	14	2924696.01	9,616,499.15
Random Forest	1	12	2547438.26	2,805,532.21
Random Forest	1	13	2498748.26	2,820,241.74
Random Forest	1	14	2633859	2,879,446.85
Ridge	0	12	4,823,552.17	6,070,755.98
Ridge	0	13	4,707,985.60	5,876,392.55
Ridge	0	14	4,956,504.69	6,204,400.79
Ridge	1	12	4,869,873.85	6,714,252.10
Ridge	1	13	4,753,309.18	6,715,681.24
Ridge	1	14	4,994,841.65	6,713,800.33

By comparing the training and testing performance of the models, we can see Linear Regression, Lasso Regression, Decision Tree without PCA, Random Forest without PCA and ANN models are overfit. Test scores are significantly worse than training ones. However, if we look at the Ridge Regression, we can say despite the score being not as good as the best performers (Decision Tree and Random Forest), the difference between training and testing is not huge like overfit ones. This shows indication of underfitting for Ridge Regression models. It couldn't learn well on training data but it was successful in generalizing on the portion it learned.

Decision Tree with PCA and Random Forest with PCA, green marked areas in above table, are the best performing models among others. The difference between training and testing scores are very less and lower when compared to others.

5.5. Post-Processing: Hyperparameter Optimization of Best Performer Model on Test Data

In the previous subsection, among six different models, online Random Forest performed best. Until this point, while deciding the best performer, minimum hyperparameter configuration is made on models. In this subsection, we go through the hyperparameters of the online Random Forest, and try some alternatives to improve the model performance even further. Random Forest has many hyperparameters to fine-tune models for better performance. We focused on a subset of them that we thought could make a difference in performance.

n_models

As mentioned earlier, Random Forest is an ensemble model which consists of smaller Decision Tree models. It is possible to configure a number of subset models that will be used. As the number of sub-models increase, generally the model stability and performance increases, however, after a certain amount of trees the performance increase starts to diminish. Additionally, having more trees increases the training times, and can even cause overfitting. In the previous subsection, we used a Random Forest model having 3 sub-models. Now we will try different models having 3, 5 and 7 ensemble members.

max_features

The maximum_features hyperparameter specifies the maximum number of features (or variables) considered for splitting at each Decision Tree node. By limiting the amount of features, it is possible to regulate the diversity of the trees and avoid overfitting. According to Gomes et al. an appropriate number for this could be m - sqrt(m), m being the total number of features [51]. As the feature count of the dataset after PCA is 5, for this hyperparameter, 3 and 5 will be considered for tuning.

lambda_value

To generate each bootstrap sample, the lambda value defines the proportion of the original dataset that is randomly sampled with replacement. Lambda values in the range of 0.5 to 0.8 are used to create larger bootstrap samples, while lambda values around 1.0 are used as a default or baseline choice. Lambda values greater than 1.0 can be used to create smaller bootstrap samples with increased diversity, helping to reduce overfitting and improve the generalization ability of the random forest. Higher values are chosen especially when the dataset size is bigger. Gomes et al suggest using a (lambda = 6) Poisson distribution in online bagging as opposed to the more common (lambda = 1) Poisson distribution [52]. Therefore, since the dataset is big, we will consider values of 1 and 6.

max_depth

Maximum depth of the model refers to how deep a tree model is allowed to grow. In other words, it determines the number of levels in the tree from the root to the farthest leaf. Setting depth limit is helpful to prevent overfitting and for the study 3, 5 and 7 will be considered as possible limits.

splitter

The Splitter, also known as the Attribute Observer (AO), provides a strategy that monitors the class statistics of numeric features and performs splits. There are different splitters able to support regression problems. 2 of them are selected to be implemented on the model.

The quantization observer (QO) divides the feature space into bins or intervals depending on the feature's observed values throughout the training phase. These bins are then utilized as thresholds in the decision tree nodes to separate the data points, which helps the tree to make binary decisions based on categorized continuous data.

E-BST operates by storing all observations in an extended binary search tree structure between splits. It saves the input feature realizations and target statistics, allowing the split heuristic to be calculated at any moment. E-BST implements a memory management routine that prunes the binary tree's worst split candidates to save time and memory. TE-BST is an E-BST variant that rounds feature values before transferring them to the binary search tree. As a result, the attribute observer may reduce processing time and memory use because tiny variations in input values will be mapped to the same BST node [51]. A truncated version of Extended Binary Search Tree (TE-BST) and Quantization observer (QO) splitters are chosen for this hyperparameter.

Some other hyperparameters are incorporated with a fixed value to the model. Metric hyperparameter controls the method used to track tree performance within the ensemble. Since the resulting performance is measured with MAE, it is configured to optimize individual model performances based on MAE scores. Aggregation method refers to the strategy that is used to combine different trees within the forest. For the study we will be using a mean strategy to benefit from another hyperparameter, weighted voting. Weighted mean assigns weights to individual tree's predictions based on metric defined earlier (for this case MAE) and uses the arithmetic mean to combine predictions of ensemble trees. Lastly, removing poor attributes is set to true, meaning the model will disable the poor attributes to reduce memory usage and produce faster outcomes.

Table 5.18 Hyperparameters Used to Tune Online Random Forest Regressor

Hyperparameter	**Value Range Used**
n_models	3, 5, 7
max_features	3, 5
lambda_value	1, 6
splitter	TEBSTSplitter, QOSplitter
max_depth	3, 5, 7
metric	MAE
aggregation_method	mean
remove_poor_attrs	True

By taking all combinations of these hyperparameters, 72 models are trained & tested on the dataset. Runs have an average test score of 2.72E+6, while the best performer model MAE is 2.65E+2 and the worst performer model having 2.89E+6. When the MAE score of the hyperparameter optimized model is compared to the DOE-2 best performer model, there is approximately %6 decrease in MAE.

The worst score is obtained by using 3 ensemble members, 3 maximum features, lambda value of 6, maximum tree depth as 3 and with TE-BST splitter. Best performance is achieved through using 7 tree members, 5 maximum features, 6 as a lambda value for boosting and maximum depth of 3 levels, with TE-BST splitter. It was anticipated that models with fewer ensemble members and features would be less performant. Increasing the number of features a model can learn and diversifying the model through more ensemble members were anticipated to yield improved performance until reaching a critical trade-off point. After which the model loses its capability to generalize and become susceptible to overfitting.

Based on the runs, a non-parametric test, Kurskal-Wallis is implemented to the recorded scores to comment on which hyperparameter has more importance to the model. The Kruskal-Wallis test is a non-parametric statistical test used to determine if there are significant differences between multiple independent groups when the dependent variable is measured on an ordinal or continuous scale. It assesses whether the means of the groups significantly differ, providing an alternative to the parametric analysis of variance (ANOVA) when the assumptions of normality and equal variances are not met. By implementing it, we compare only the means of groups to determine if one of them dominates the other. The results given in Table 5.19 show that p-values of n_models and max_depth are low, meaning these hyperparameters have significant effect on MAE of online random forest models.

Table 5.19 Kruskal-Wallis P-Value Result Summary

Hyperparameter	P-value
n_models	**0.017**
max_features	0.281
max_depth	**0.088**
lambda_value	0.372
splitter	0.302

It is important to note that hyperparameter optimization is another important problem in machine learning and requires more attention & customization. Traditional approaches like Random Search and Grid Search could not be used for the online models. The selected ranges for the hyperparameters are limited and cover only a portion of all possible solution space. Therefore, as a thought experiment, a subset of hyperparameters are selected to see how far the model can be improved.

5.6. Summary

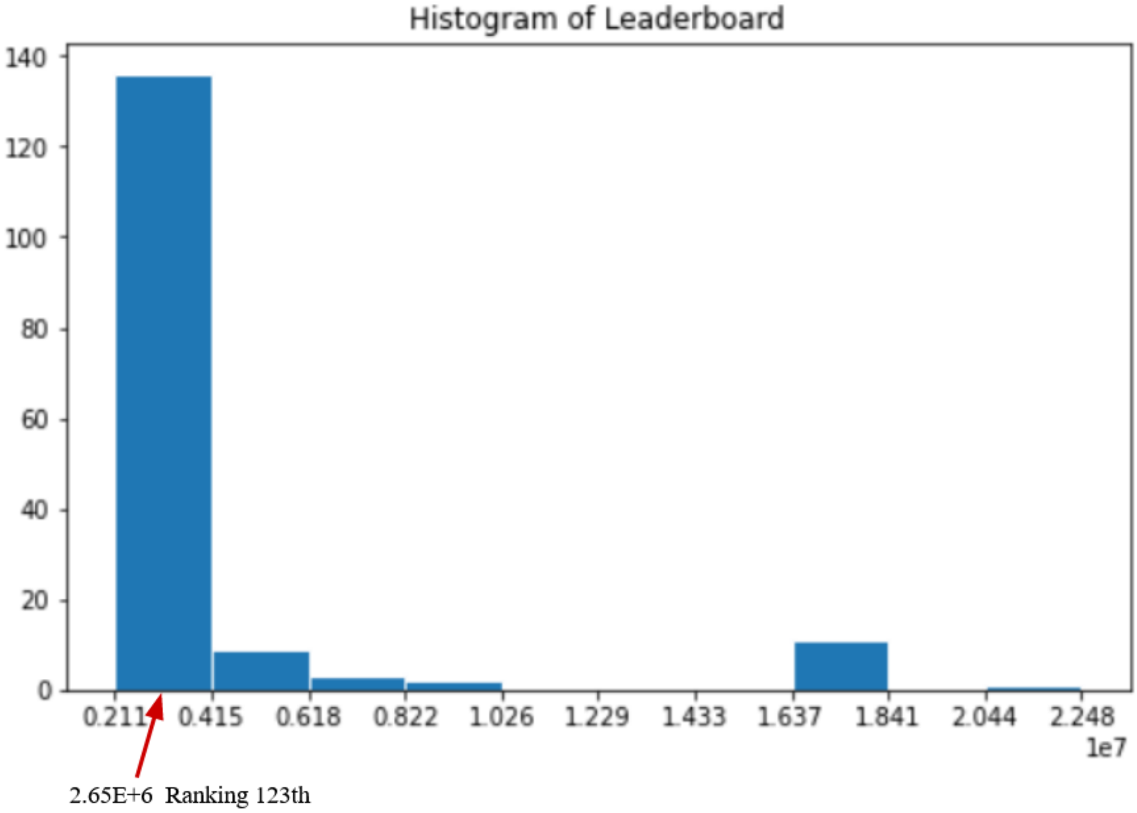
Primary focus of this study is to prove online learning models can produce good results when compared to traditional models in solar energy prediction. To prove this, we designed a two-phase method. In the first phase we run 6 predictive models on training data by using the same factor combinations for traditional and online versions. After running a total of 336 models, based on MAE scores, a set of design of experiments are conducted and insignificant factors are eliminated. Online learning models are found to be more performant in the majority of the predictive models. In the second phase, with fewer factor settings, 36 online learning models are run on test data to create daily solar power production predictions spanning a time period of 2008-2012. Predictions are submitted to the Kaggle platform to get test errors, and another design of the experiment is applied. The factors for the best performing predictive model is as follows:

- Model Type: Random Forest
- Training Approach: Online Learning
- Ensemble Strategy: Taking the Minimum of the given 11 GEFS ensembles (Model 13)
- PCA Implementation: Yes (5 principal components)

In the post-processing phase, ad-hoc hyperparameter optimization is studied with 6 hyperparameters. Additional 72 runs are taken and a non-parametric Kruskal-Wallis test is implemented to compare means. Number of ensemble models within Random Forest and maximum depth of trees are found more significant to achieve lower test errors. Through all phases, models are trained with minimum hyperparameter adjustment, until post processing, which resulted in %6 decrease in MAE. The best score which belongs to hyperparameter-configured Random Forest is recorded as 2.65E+6. Optimal parameters of the best performing hyperparameters are as follows:

- n_models = 7
- max_features = 5
- lambda = 6
- maximum_depth = 3
- splitter = TE-BST

As mentioned earlier in Section 4, the selected dataset comes from an open source competition in Kaggle platform, AMS Solar Power Prediction. 160 teams joined the competition at the time it was aired. Many teams competed with each other to create the best model that can produce the lowest MAE score. In Figure 5.12, we drew a histogram of the scores collected through competition. Most of the scores fall within the first bin, which shows the complexity of the problem. After the post-processing part, the submitted score by our best model, also falls within first bin, which would land in 123th position. Given that other participants aimed to decrease their error margins by creating custom ensemble models, configuring model hyperparameters extensively, incorporating more information by advanced feature engineering techniques, it is promising to see an online learning model perform in close range while having humble parameter optimization steps.



2.65E+6 Ranking 123th

Figure 5.12 Histogram of Leaderboard

6. CONCLUSION & DISCUSSIONS

Solar energy is one of the most rapidly growing forms of renewable energy that has become increasingly popular due to its abundance and accessibility. It has already made significant contributions to cutting carbon emissions and is projected to replace fossil fuels as the leading source of electricity generation by 2050. The growth of solar energy has been supported by various factors, including scientific advancements in PV technology, materials getting more affordable, and government incentives that support the use of it. The decreasing costs of solar panels and the support of big tech companies and having more and more companies specializing in solar installations have made solar power more accessible. Solar energy is not only beneficial for reducing dependency on non-renewable energy sources but also for promoting energy independence, as it can be locally produced anywhere.

However, solar energy is difficult to integrate to grids due to its stochastic nature and variable production that is affected by environmental factors. Changes in weather conditions have a direct impact on the amount of energy produced, making it important to accurately predict production rates, and to maintain grid stability. Therefore, accurate forecasts of solar energy production have become essential for efficient use, distribution, and pricing.

Online learning algorithms process data sequentially as it arrives, allowing for real-time adaptation to pattern changes in data. It is cost-effective and provides predictable training times as it is an iterative process. Models can be configured to optimally balance rapid adaptation and the risk of neglecting previous data. These features make it a good candidate to be used in predicting solar energy production. In this study, we focused on using an incremental training approach for machine learning models to predict solar power production. By utilizing the American Meteorological Society (AMS) Solar Power Prediction Competition dataset from the Kaggle platform, we explored the potential of online learning models as a promising approach for accurate solar energy prediction. The two-step process is designed for the selected 6 different prediction models, to demonstrate how online learning approaches outperform the traditional models in terms of solar energy prediction.

Online learning provides promising results in the solar energy industry from different aspects like energy production rates, cell defect detection. In the AMS Solar Power Prediction Competition, many different machine learning approaches have been studied but there are not any online learning implementations. To assess the superiority of online learning models over traditional approaches, a 2-phased methodology is proposed with the following post-processing step. In the first step of the study, traditional and online training approaches are compared against the same dataset based on their training scores measured by MAE metric. Among 6 different predictive models, online training performed similar to traditional training for Linear Regression, it outperformed for Lasso Regression, Decision Tree, Random Forest and ANN, and underperformed for Ridge Regression. In the second step, some factors are reduced from the experiment and only incremental training approaches are used for training models. Comparison for the second step is made on testing scores by uploading prediction results to Kaggle platform and getting MAE scores. Online Random Forest and Online Decision Tree models outperformed other models. In the post-processing phase, the online random forest model is taken as the best performer, since it is an ensemble model consisting of smaller individual decision tree models. A subset of hyperparameters are selected and additional model runs are recorded to improve the performance further. Best MAE score is recorded as 2.65E+6, which would rank at 123th position in the competition. As it can be seen, a comprehensive set of models are run throughout the process. As a result, it is statistically proven that online learning models outperform traditional models.

As a future research direction, hyperparameter optimization with more parameters could be run to improve the performance and see the effect of different hyperparameters more clearly. Additionally, in the competition traditional models with interpolation implementation seem to have better scores, this could be implemented to online models. In the scope of the thesis, we only worked with a set of predictive models with minimum hyperparameter optimization or customization of models. Most of the competition participants built custom models incorporating different ensemble techniques. Even in the minimal experimental setting, online models outperformed many traditional approaches, with custom models, model performance could be improved drastically.

The results and insights gained from this research can better inform decision-making processes related to the installment and expansion of solar energy plants. Reliable estimates of solar energy production support the economy, promote growth, and drive the development of more clean and renewable energy sources. Solar energy continues to play a pivotal role in our transition to a sustainable future, with the help of more data and recent advances in technology, researchers are able to create better predictive models, and the incremental training approach provides promising results.

8. REFERENCES

1. IEA (2022), Approximately 100 million households rely on rooftop solar PV by 2030, IEA, Paris https://www.iea.org/reports/approximately-100-million-households-rely-on-rooftop-solar-pv-by-2030, License: CC BY 4.0.

2. IEA (2021), Net Zero by 2050, IEA, Paris https://www.iea.org/reports/net-zero-by-2050, License: CC BY 4.0.

3. Gielen, D., Boshell, F., Saygin, D., Bazilian, M. D., Wagner, N., & Gorini, R. (2019). The role of renewable energy in the global energy transformation. Energy Strategy Reviews, 24, 38-50.

4. Forecasts. (n.d.). ECMWF. https://www.ecmwf.int/en/forecasts

5. Global Ensemble Forecast System (GEFS). (2020, August 10). National Centers for Environmental Information (NCEI). https://www.ncei.noaa.gov/products/weather-climate-models/global-ensemble-forecast

6. Geron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras and TensorFlow: concepts, tools, and techniques to build intelligent systems (2nd ed.). O'Reilly.

7. Cesa-Bianchi, Nicolò & Orabona, Francesco. (2021). Online Learning Algorithms. Annual Review of Statistics and Its Application. 8. 10.1146/annurev-statistics-040620-035329.

8. Sculley, D. (2008). ADVANCES IN ONLINE LEARNING-BASED SPAM FILTERING.

9. Wang, Qiang & Guan, Yi & Wang, Xiaolong. (2006). SVM-Based Spam Filter with Active and Online Learning.

10. Puah, Boon & Chong, Lee Wai & Wong, Yee Wan & Begam, K.M. & Khan, Nafizah & Juman, Mohammed Ayoub & Rajkumar, Rajprasad. (2021). A regression unsupervised incremental learning algorithm for solar irradiance prediction. Renewable Energy. 164. 908-925. 10.1016/j.renene.2020.09.080.

11. Levina, Tatsiana & Levin, Yuri & McGill, Jeff & Nediak, Mikhail. (2009). Dynamic Pricing with Online Learning and Strategic Consumers: An Application of the Aggregating Algorithm. Operations Research. 57. 327-341. 10.1287/opre.1080.0577.

12. Luo, Qi & Saigal, Romesh. (2017). Dynamic Pricing for On-Demand Ride-Sharing: A Continuous Approach. SSRN Electronic Journal. 10.2139/ssrn.3056498.

13. Ji, X., Wang, J., & Yan, Z. (2021). A stock price prediction method based on deep learning technology. International Journal of Crowd Science, 5(1), 55–72. 10.1108/ijcs-05-2020-0012.

14. Zhong, Jiaqi & Liu, Luyao & Sun, Qie & Wang, Xinyu. (2018). Prediction of Photovoltaic Power Generation Based on General Regression and Back Propagation Neural Network. Energy Procedia. 152. 1224-1229. 10.1016/j.egypro.2018.09.173.

15. C. Yang, A.A. Thatte, and L. Xie. (2015). Multitime-Scale Data-Driven Spatio-Temporal Forecast of Photovoltaic Generation, IEEE Transactions on Sustainable Energy, Vol. 6, No. 1, pp. 104-112.

16. M. N. H. Nguyen, Chuan Pham, Jaehyeok Son and C. S. Hong. (2016). Online learning-based clustering approach for news recommendation systems.18th Asia-Pacific Network Operations and Management Symposium (APNOMS), 2016, pp. 1-4, 10.1109/APNOMS.2016.7737269.

17. Feng, Cong & Cui, Mingjian & Hodge, Bri-Mathias & Lu, Siyuan & Hamann, Hendrik. (2018). Unsupervised Clustering-Based Short-Term Solar Forecasting. IEEE Transactions on Sustainable Energy. PP. 1-1. 10.1109/TSTE.2018.2881531.

18. Ashvin Nair, Abhishek Gupta, Murtaza Dalal, & Sergey Levine. (2020). AWAC: Accelerating Online Reinforcement Learning with Offline Datasets.

19. Ludovic Hofer, & Hugo Gimbert. (2016). Online Reinforcement Learning for Real-Time Exploration in Continuous State and Action Markov Decision Processes.

20. Himeur, Y., Alsalemi, A., Al-Kababji, A., Bensaali, F., Amira, A., Sardianos, C., Dimitrakopoulos, G., & Varlamis, I. (2021). A survey of recommender systems for energy efficiency in buildings: Principles, challenges and prospects. Information Fusion, 72, 1–21. 10.1016/j.inffus.2021.02.002.

21. L. Song, C. Tekin and M. van der Schaar.(2016). Online Learning in Large-Scale Contextual Recommender Systems. IEEE Transactions on Services Computing, vol. 9, no. 3, pp. 433-445, 10.1109/TSC.2014.2365795.

22. Nethra Viswanathan. (2020). Adaptive Multi-Agent E-Learning Recommender Systems.

23. R. Leo, R. S. Milton and S. Sibi, "Reinforcement learning for optimal energy management of a solar microgrid," 2014 IEEE Global Humanitarian Technology Conference- South Asia Satellite (GHTC-SAS), 2014, pp. 183-188, 10.1109/GHTC-SAS.2014.6967580.

24. Raju, L., Sankar, S., & Milton, R. S. (2015). Distributed Optimization of Solar Micro-grid Using Multi Agent Reinforcement Learning. Procedia Computer Science, 46, 231–239. 10.1016/j.procs.2015.02.016.

25. Unal, A. (2022). Forecasting of Renewable Power Generation In European Markets Via Long Short, Master’s Thesis.

26. Atcı, Y. (2014). Türkiye’deki Rüzgar Enerjisi Üreten Türbinlerin Enerji Tahmini: Markov Süreci Yaklaşımı, Master’s Thesis.

27. Horasan, M. (2021). A Multi-Objective Decision Making Model For Renewable Energy Planning: The Case of Turkey, Master’s Thesis.

28. Shtewi, F. (2021). Wind Assessment For Sites In Libya And Predict Of Annual Energy, Master’s Thesis.

29. Altun, I. (2019). Yenilenebilir Enerji Kaynaklarından Elektrik Enerjisi Üretiminin Belirleyicileri: Türkiye Örneği, Master’s Thesis.

30. Kaya, N. (2017). Zaman Serilerine Dayalı Tahmin Yöntemleri İle Türkiye’nin Yenilenebilir Enerji Kaynakları Talebinin Tahmini, Master’s Thesis.

31. Çetin, I. (2018). Effects Of Numerical Weather Predictions On Wind Power Forecasts, Master's Thesis.

32. Turan, F. (2019). Zaman Serileri Modelleri İle Yenilenebilir Enerji Sistemlerinin Güç Üretim Tahminlemesi, Master's Thesis.

33. Derse, O. (2022). Yenilenebilir Enerji Kaynakları İçin Tesis Yer Seçimi Optimizasyonu, PhD Thesis.

34. Ervural, B. (2018). Yenilenebilir Enerji Planlaması İçin Bütünleşik Çok Amaçlı Bir Karar Modeli Önerisi, PhD Thesis.

35. Wu, Y.-K.; Huang, C.-L.; Phan, Q.-T.; Li, Y.-Y. Completed Review of Various Solar Power Forecasting Techniques Considering Different Viewpoints. Energies 2022, 15, 3320. 10.3390/en15093320.

36. Guermoui, M., Melgani, F., Gairaa, K., & Mekhalfi, M. L. (2020, June). A comprehensive review of hybrid models for solar radiation forecasting. Journal of Cleaner Production, 258, 120357. https://doi.org/10.1016/j.jclepro.2020.120357

37. ERTEN, M. Y., & AYDİLEK, H. (2022). Solar Power Prediction using Regression Models. In Uluslararası Muhendislik Arastirma ve Gelistirme Dergisi (Vol. 14, Issue 3, pp. 1–1). Uluslararasi Muhendislik Arastirma ve Gelistirme Dergisi. 10.29137/umagd.1100957.

38. Sarmas, E.; Strompolas, S.;Marinakis, V.; Santori, F.; Bucarelli, M.A.(2022). Doukas, H. An Incremental Learning Framework for Photovoltaic Production and Load Forecasting in Energy Microgrids. Electronics 2022, 11, 3962. 10.3390/electronics11233962.

39. Puah, B. K., Chong, L. W., Wong, Y. W., Begam, K., Khan, N., Juman, M. A., & Rajkumar, R. K. (2021, February). A regression unsupervised incremental learning algorithm for solar irradiance prediction. Renewable Energy, 164, 908–925. 10.1016/j.renene.2020.09.080.

40. Balzategui, J., & Eciolaza, L. (2023, October). Few-shot incremental learning in the context of solar cell quality inspection. Expert Systems With Applications, 228, 120382. 10.1016/j.eswa.2023.120382

41. B. Saad, A. E. Hannani, R. Errattahi and A. Aqqal, "Assessing the Impact of Weather Forecast Models Combination on the AMS Solar Energy Prediction," 2020 Fourth International Conference On Intelligent Computing in Data Sciences (ICDS), Fez, Morocco, 2020, pp. 1-5, 10.1109/ICDS50568.2020.9268767.

42. D. D´ıaz-Vico, A. Torres-Barr´an, A. Omari, and J. R. Dorronsoro. (2017). "Deep neural networks for wind and solar energy prediction," Neural Processing Letters, vol. 46, no. 3, pp. 829–844.

43. I. Araf, H. Elkhadiri, R. Errattahi and A. E. Hannani. (2019). "AMS Solar Energy Prediction: A Comparative Study of Regression Models," 2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS), Marrakech, Morocco, 2019, pp. 1-5, 10.1109/ICDS47004.2019.8942237.

44. S. Aggarwal and L. Saini. (2014). "Solar energy prediction using linear and non- linear regularization models: A study on ams (american meteorological society) 2013–14 solar energy prediction contest," Energy, vol. 78, pp. 247–256.

45. R. Juban and P. Quach. (2013). “Predicting daily incoming solar energy from weather data,” Stanford University - CS229 Machine Learning.

46. Aler, R., Martín, R., Valls, J. M., & Galván, I. M. (2015). A Study of Machine Learning Techniques for Daily Solar Energy Forecasting Using Numerical Weather Models. In Intelligent Distributed Computing VIII (pp. 269–278). Springer International Publishing, /10.1007/978-3-319-10422-5_29.

47. About the Mesonet | Mesonet. (n.d.). About the Mesonet | Mesonet. https://www.mesonet.org/about/about-the-mesonet.

48. Least Absolute Shrinkage and Selection Operator (LASSO). (2017, September 17). Columbia University Mailman School of Public Health. https://www.publichealth.columbia.edu/research/population-health-methods/least-absolute-shrinkage-and-selection-operator-lasso.

49. Zeng, F., Huang, W.-C., & Hueng, J. (2016). On Chinese Government’s Stock Market Rescue Efforts in 2015. In Modern Economy (Vol. 07, Issue 04, pp. 411–418). Scientific Research Publishing, Inc. 10.4236/me.2016.74045.

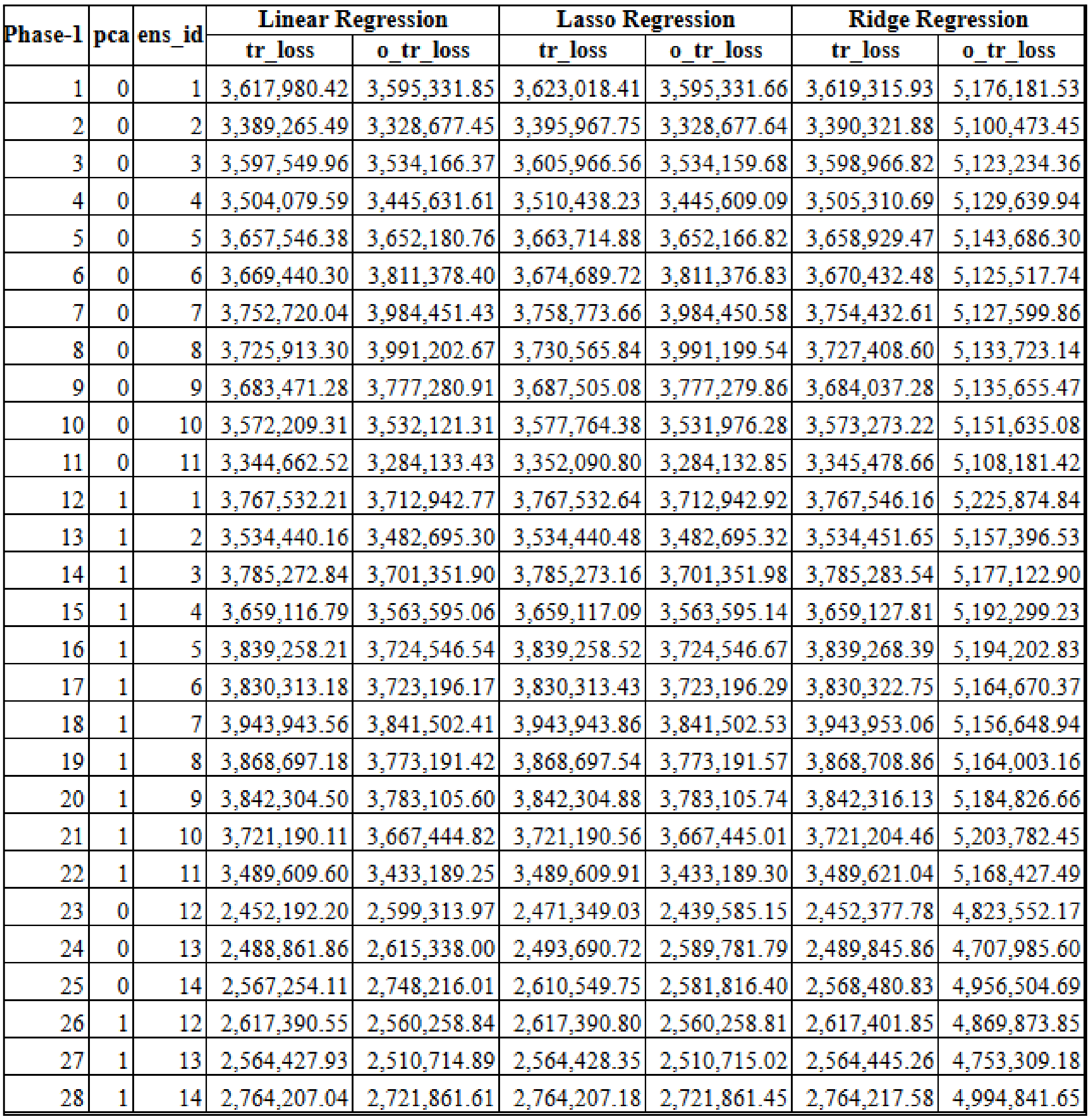
50. What Is Design of Experiments (DOE)? | ASQ. (n.d.). What Is Design of Experiments (DOE)? | ASQ. https://asq.org/quality-resources/design-of-experiments.

51. Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G., & Abdessalem, T. (2017, June 13). Adaptive random forests for evolving data stream classification. Machine Learning, 106(9–10), 1469–1495. 10.1007/s10994-017-5642-8

52. Breiman, L. (2001, October 1). Random Forests - Machine Learning. SpringerLink. 10.1023/A:1010933404324

APPENDIX

Appendix 1: DOE-1 Model Runs



Phase-1	pca	ens_id	Linear Regression		Lasso Regression		Ridge Regression	
			tr_loss	o_tr_loss	tr_loss	o_tr_loss	tr_loss	o_tr_loss
1	0	1	3,617,980.42	3,595,331.85	3,623,018.41	3,595,331.66	3,619,315.93	5,176,181.53
2	0	2	3,389,265.49	3,328,677.45	3,395,967.75	3,328,677.64	3,390,321.88	5,100,473.45
3	0	3	3,597,549.96	3,534,166.37	3,605,966.56	3,534,159.68	3,598,966.82	5,123,234.36
4	0	4	3,504,079.59	3,445,631.61	3,510,438.23	3,445,609.09	3,505,310.69	5,129,639.94
5	0	5	3,657,546.38	3,652,180.76	3,663,714.88	3,652,166.82	3,658,929.47	5,143,686.30
6	0	6	3,669,440.30	3,811,378.40	3,674,689.72	3,811,376.83	3,670,432.48	5,125,517.74
7	0	7	3,752,720.04	3,984,451.43	3,758,773.66	3,984,450.58	3,754,432.61	5,127,599.86
8	0	8	3,725,913.30	3,991,202.67	3,730,565.84	3,991,199.54	3,727,408.60	5,133,723.14
9	0	9	3,683,471.28	3,777,280.91	3,687,505.08	3,777,279.86	3,684,037.28	5,135,655.47
10	0	10	3,572,209.31	3,532,121.31	3,577,764.38	3,531,976.28	3,573,273.22	5,151,635.08
11	0	11	3,344,662.52	3,284,133.43	3,352,090.80	3,284,132.85	3,345,478.66	5,108,181.42
12	1	1	3,767,532.21	3,712,942.77	3,767,532.64	3,712,942.92	3,767,546.16	5,225,874.84
13	1	2	3,534,440.16	3,482,695.30	3,534,440.48	3,482,695.32	3,534,451.65	5,157,396.53
14	1	3	3,785,272.84	3,701,351.90	3,785,273.16	3,701,351.98	3,785,283.54	5,177,122.90
15	1	4	3,659,116.79	3,563,595.06	3,659,117.09	3,563,595.14	3,659,127.81	5,192,299.23
16	1	5	3,839,258.21	3,724,546.54	3,839,258.52	3,724,546.67	3,839,268.39	5,194,202.83
17	1	6	3,830,313.18	3,723,196.17	3,830,313.43	3,723,196.29	3,830,322.75	5,164,670.37
18	1	7	3,943,943.56	3,841,502.41	3,943,943.86	3,841,502.53	3,943,953.06	5,156,648.94
19	1	8	3,868,697.18	3,773,191.42	3,868,697.54	3,773,191.57	3,868,708.86	5,164,003.16
20	1	9	3,842,304.50	3,783,105.60	3,842,304.88	3,783,105.74	3,842,316.13	5,184,826.66
21	1	10	3,721,190.11	3,667,444.82	3,721,190.56	3,667,445.01	3,721,204.46	5,203,782.45
22	1	11	3,489,609.60	3,433,189.25	3,489,609.91	3,433,189.30	3,489,621.04	5,168,427.49
23	0	12	2,452,192.20	2,599,313.97	2,471,349.03	2,439,585.15	2,452,377.78	4,823,552.17
24	0	13	2,488,861.86	2,615,338.00	2,493,690.72	2,589,781.79	2,489,845.86	4,707,985.60
25	0	14	2,567,254.11	2,748,216.01	2,610,549.75	2,581,816.40	2,568,480.83	4,956,504.69
26	1	12	2,617,390.55	2,560,258.84	2,617,390.80	2,560,258.81	2,617,401.85	4,869,873.85
27	1	13	2,564,427.93	2,510,714.89	2,564,428.35	2,510,715.02	2,564,445.26	4,753,309.18
28	1	14	2,764,207.04	2,721,861.61	2,764,207.18	2,721,861.45	2,764,217.58	4,994,841.65

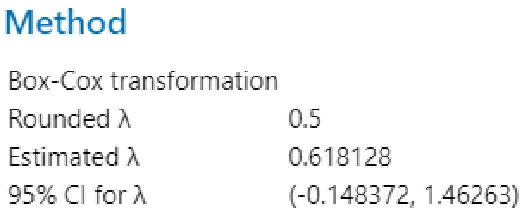
Appendix 1: DOE-1 Model Run (Continued)

Phase-1	pca	ens_id	Decision Tree		Random Forest		ANN	
			tr_loss	o_tr_loss	tr_loss	o_tr_loss	tr_loss	o_tr_loss
1	0	1	3,680,266.49	3,660,633.81	3,661,673.89	4,068,217.17	5,027,086.00	3,447,605.25
2	0	2	3,491,455.93	3,466,761.75	3,459,468.27	3,814,936.72	4,978,641.50	3,214,054.00
3	0	3	3,705,005.50	3,685,043.98	3,693,991.63	3,901,513.22	4,956,944.50	3,425,649.50
4	0	4	3,580,281.26	3,625,464.32	3,564,822.39	3,738,924.38	5,015,060.00	3,293,516.75
5	0	5	3,777,681.84	3,632,930.80	3,763,112.10	4,135,780.63	4,960,938.00	3,435,221.75
6	0	6	3,756,380.56	3,728,372.23	3,745,273.72	4,137,078.34	4,953,922.00	3,421,949.75
7	0	7	3,869,569.96	3,699,391.97	3,863,779.40	3,922,876.28	4,923,575.00	3,508,729.25
8	0	8	3,787,803.19	3,718,351.25	3,759,505.64	3,870,706.31	5,119,039.00	3,458,568.50
9	0	9	3,761,740.52	3,701,231.28	3,749,884.99	4,037,308.23	4,964,652.00	3,416,038.50
10	0	10	3,623,645.96	3,646,142.57	3,610,199.45	3,920,321.58	4,987,446.00	3,294,851.00
11	0	11	3,422,129.22	3,320,148.12	3,390,671.85	3,447,287.39	4,999,962.00	3,064,759.75
12	1	1	3,902,502.00	3,674,112.27	3,862,032.96	3,750,839.81	12,528,056.00	3,520,773.00
13	1	2	3,726,515.00	3,408,766.62	3,681,575.16	3,620,441.94	11,622,616.00	3,309,109.50
14	1	3	3,948,008.00	3,666,491.32	3,878,196.57	3,758,222.87	12,471,460.00	3,569,499.00
15	1	4	3,808,875.00	3,599,381.04	3,773,573.70	3,761,692.57	12,323,934.00	3,419,883.75
16	1	5	3,995,696.00	3,764,673.28	3,917,899.03	4,001,443.20	12,128,337.00	3,602,485.75
17	1	6	3,953,156.00	3,727,320.04	3,894,555.49	4,026,755.72	11,279,358.00	3,595,685.00
18	1	7	4,060,157.00	3,830,537.28	3,998,789.24	3,811,310.33	11,299,706.00	3,730,278.50
19	1	8	3,989,836.00	3,734,688.27	3,938,655.48	3,715,083.43	12,640,323.00	3,622,967.50
20	1	9	3,991,949.00	3,793,405.07	3,960,132.14	3,807,436.61	12,048,955.00	3,621,921.00
21	1	10	3,864,768.00	3,632,970.78	3,825,653.93	3,590,494.78	12,894,641.00	3,478,567.75
22	1	11	3,668,828.00	3,355,008.12	3,619,480.23	3,481,054.15	10,559,991.00	3,255,779.75
23	0	12	2,546,859.75	2,415,909.63	2,521,680.44	2,725,262.39	5,237,194.50	2,305,371.25
24	0	13	2,562,173.40	2,415,909.63	2,541,137.68	2,854,325.73	4,734,466.50	2,326,162.50
25	0	14	2,626,233.57	2,415,909.63	2,614,288.19	2,924,696.01	5,034,900.00	2,326,672.25
26	1	12	2,815,309.00	2,446,872.69	2,766,924.61	2,547,438.26	12,706,168.00	2,370,540.00
27	1	13	2,893,632.00	2,465,797.81	2,852,271.10	2,498,748.26	11,557,685.00	2,351,943.75
28	1	14	2,801,094.00	2,513,809.64	2,781,807.63	2,633,859.00	10,502,777.00	2,431,919.50

Appendix 2: DOE-1 Results

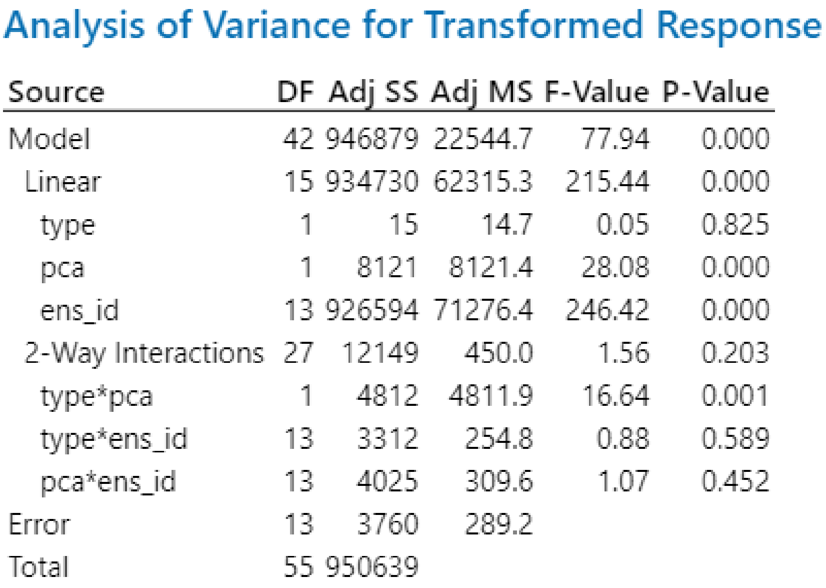
2.1 Linear Regression

ANOVA



Method

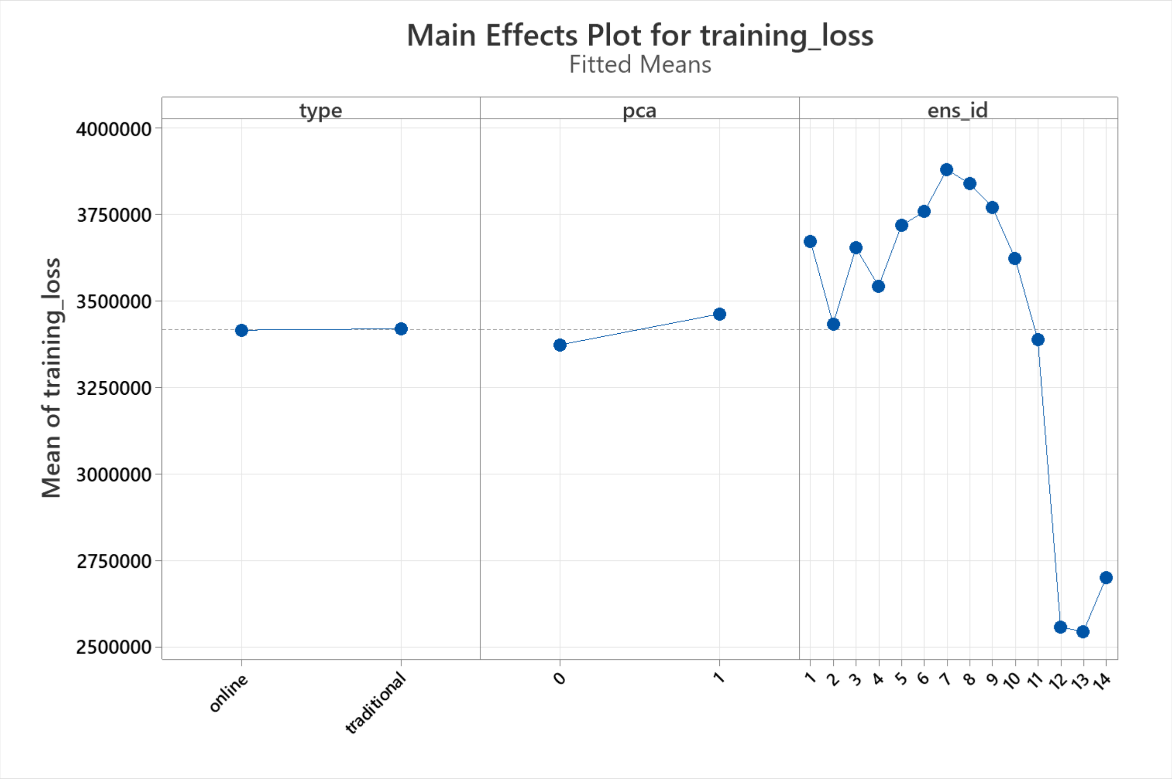
Box-Cox transformation	
Rounded λ	0.5
Estimated λ	0.618128
95% CI for λ	(-0.148372, 1.46263)

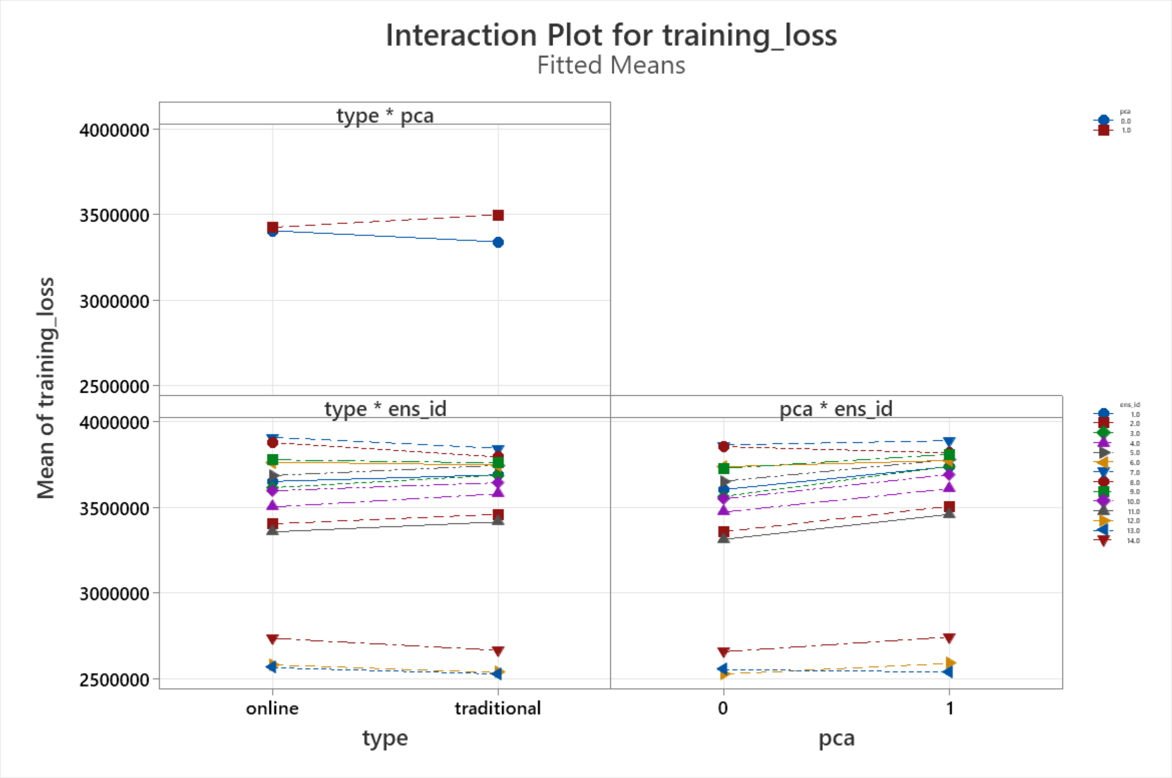


Analysis of Variance for Transformed Response

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	42	946879	22544.7	77.94	0.000
Linear	15	934730	62315.3	215.44	0.000
type	1	15	14.7	0.05	0.825
pca	1	8121	8121.4	28.08	0.000
ens_id	13	926594	71276.4	246.42	0.000
2-Way Interactions	27	12149	450.0	1.56	0.203
type*pca	1	4812	4811.9	16.64	0.001
type*ens_id	13	3312	254.8	0.88	0.589
pca*ens_id	13	4025	309.6	1.07	0.452
Error	13	3760	289.2		
Total	55	950639			

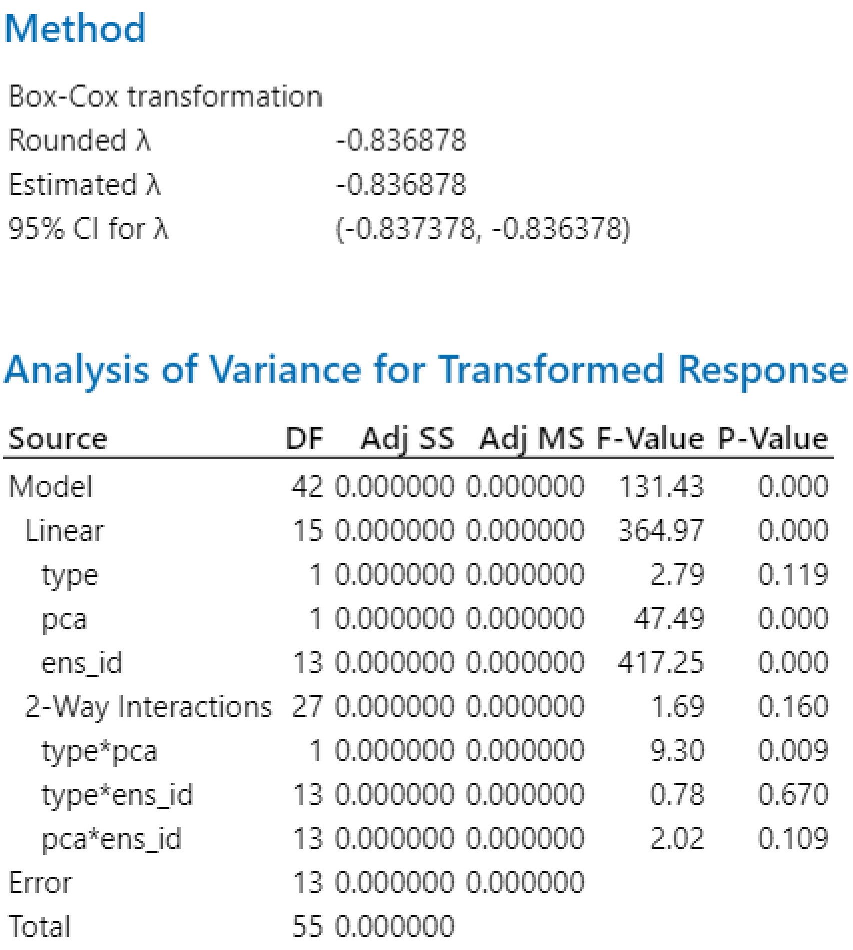
Factor Plots





2.2 Lasso Regression

ANOVA



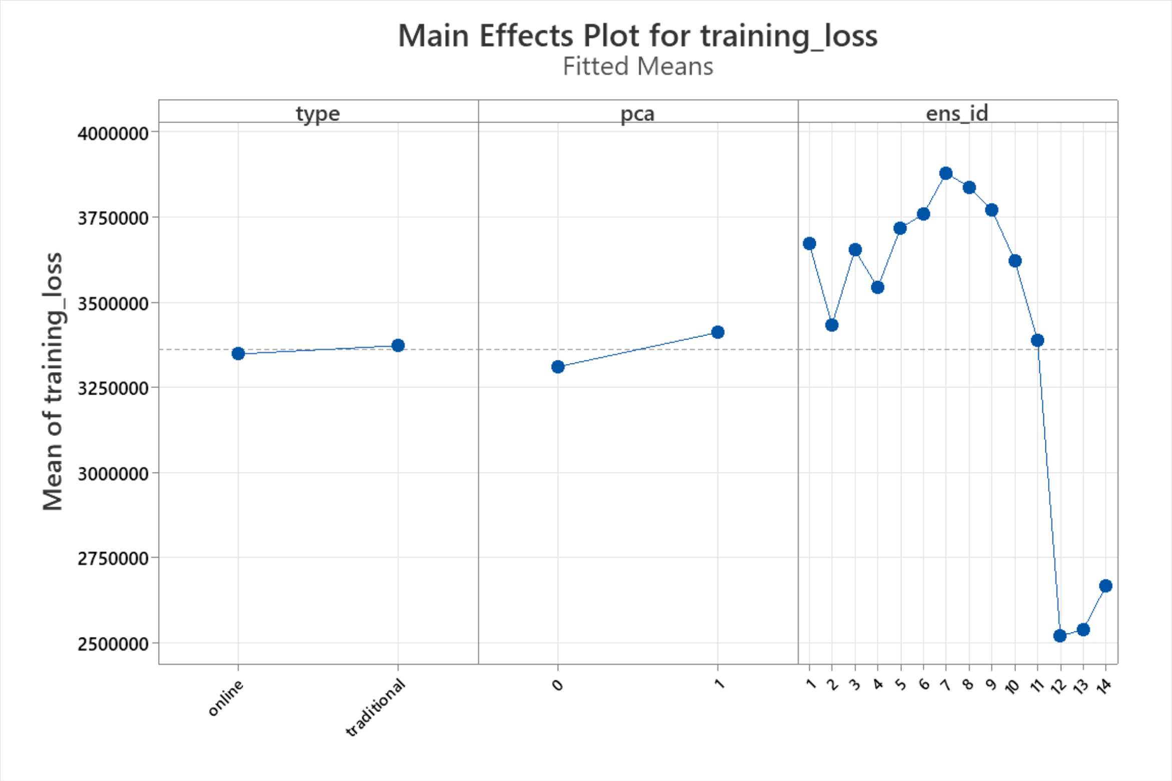
Method

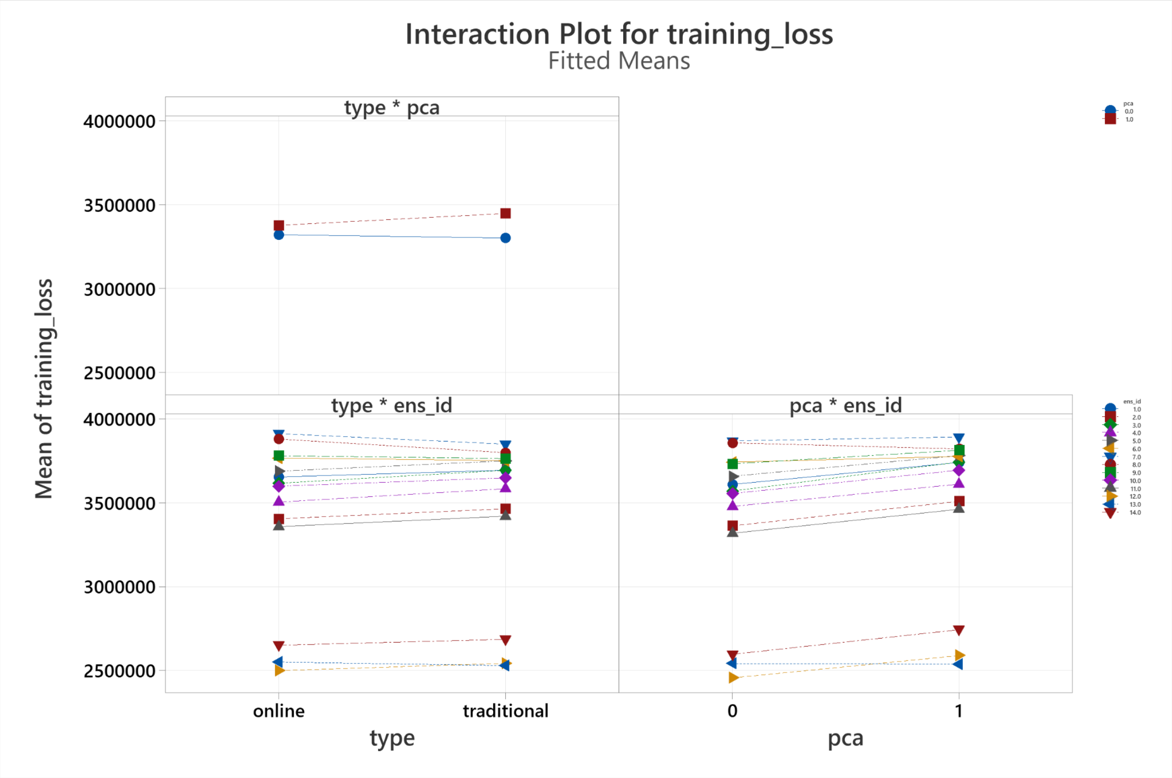
Box-Cox transformation	
Rounded λ	-0.836878
Estimated λ	-0.836878
95% CI for λ	(-0.837378, -0.836378)

Analysis of Variance for Transformed Response

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	42	0.000000	0.000000	131.43	0.000
Linear	15	0.000000	0.000000	364.97	0.000
type	1	0.000000	0.000000	2.79	0.119
pca	1	0.000000	0.000000	47.49	0.000
ens_id	13	0.000000	0.000000	417.25	0.000
2-Way Interactions	27	0.000000	0.000000	1.69	0.160
type*pca	1	0.000000	0.000000	9.30	0.009
type*ens_id	13	0.000000	0.000000	0.78	0.670
pca*ens_id	13	0.000000	0.000000	2.02	0.109
Error	13	0.000000	0.000000		
Total	55	0.000000			

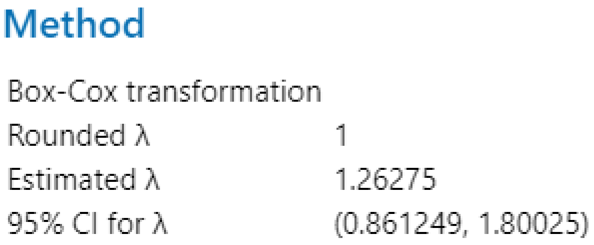
Factor Plots





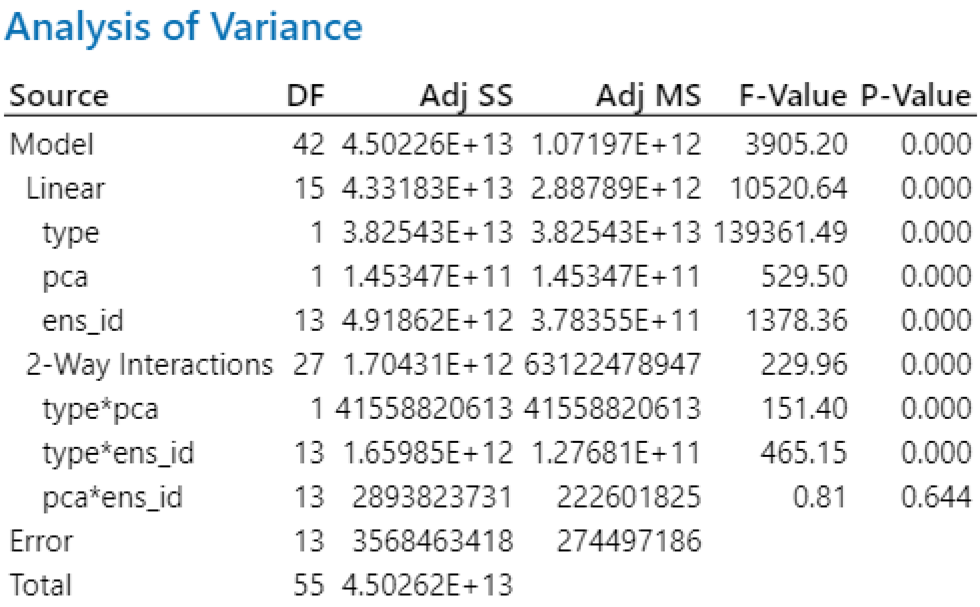
2.3 Ridge Regression

ANOVA



Method

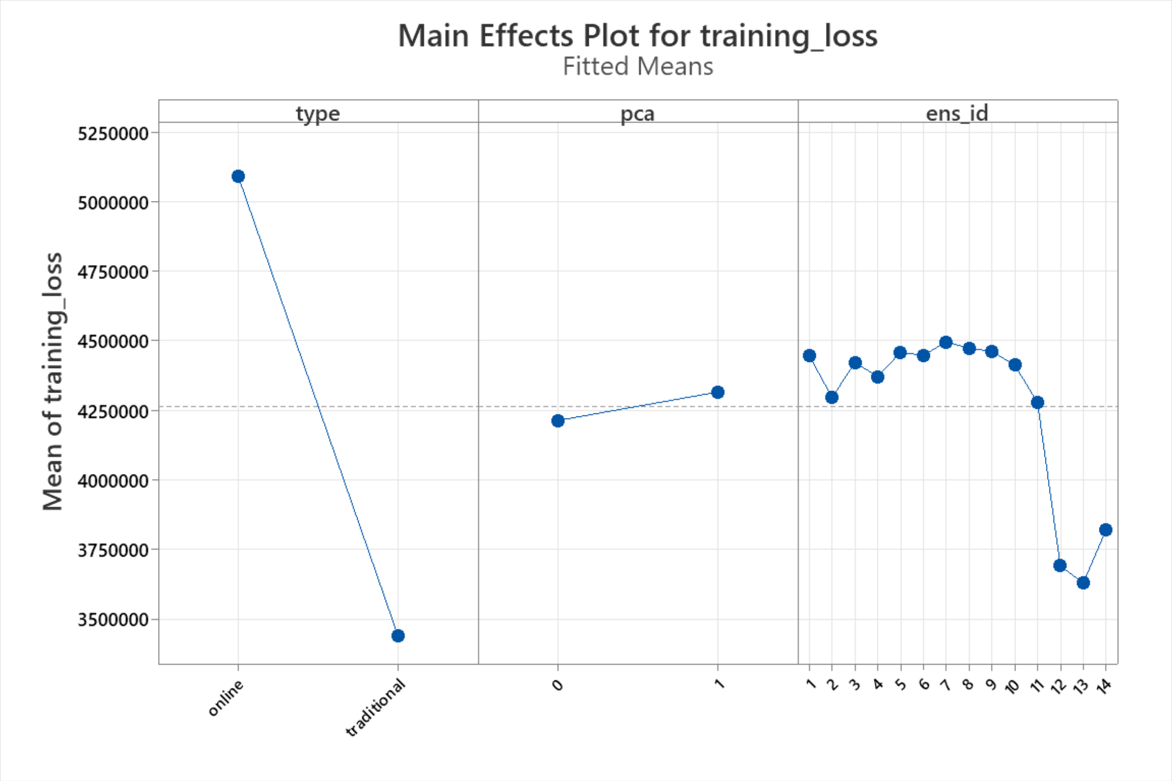
Box-Cox transformation	
Rounded λ	1
Estimated λ	1.26275
95% CI for λ	(0.861249, 1.80025)

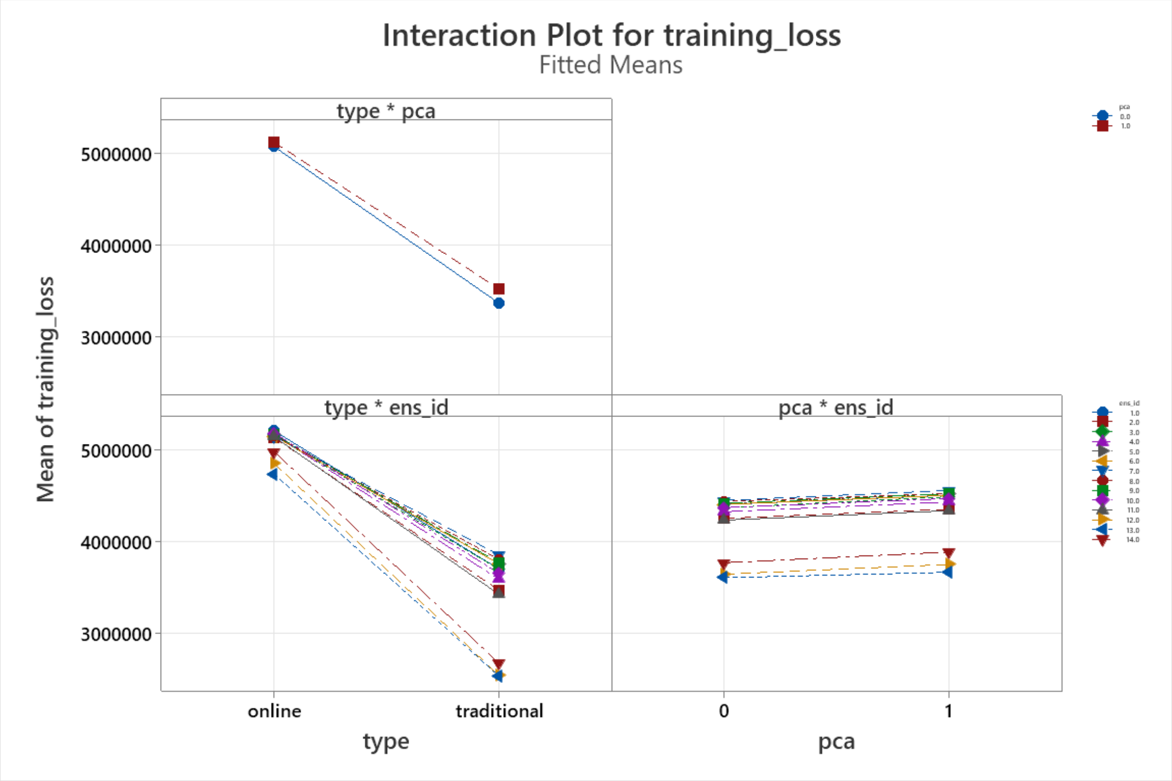


Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	42	4.50226E+13	1.07197E+12	3905.20	0.000
Linear	15	4.33183E+13	2.88789E+12	10520.64	0.000
type	1	3.82543E+13	3.82543E+13	139361.49	0.000
pca	1	1.45347E+11	1.45347E+11	529.50	0.000
ens_id	13	4.91862E+12	3.78355E+11	1378.36	0.000
2-Way Interactions	27	1.70431E+12	63122478947	229.96	0.000
type*pca	1	41558820613	41558820613	151.40	0.000
type*ens_id	13	1.65985E+12	1.27681E+11	465.15	0.000
pca*ens_id	13	2893823731	222601825	0.81	0.644
Error	13	3568463418	274497186		
Total	55	4.50262E+13			

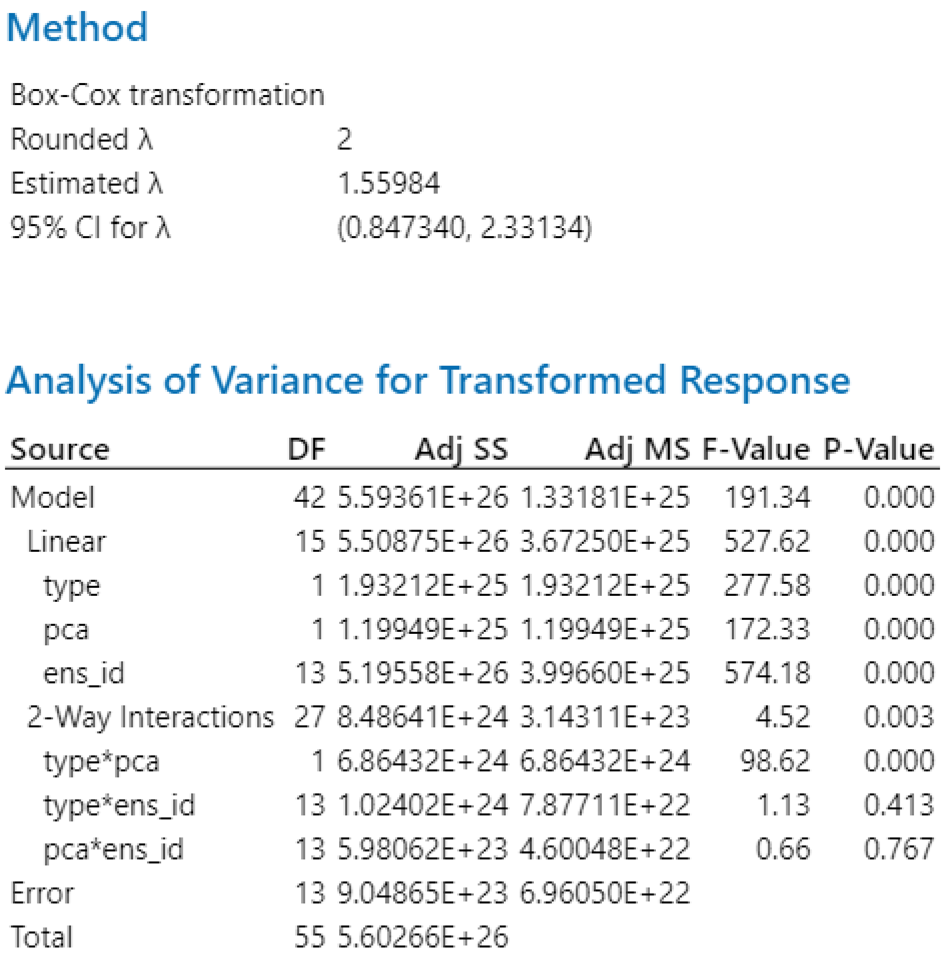
Factor Plots





2.4 Decision Tree

ANOVA



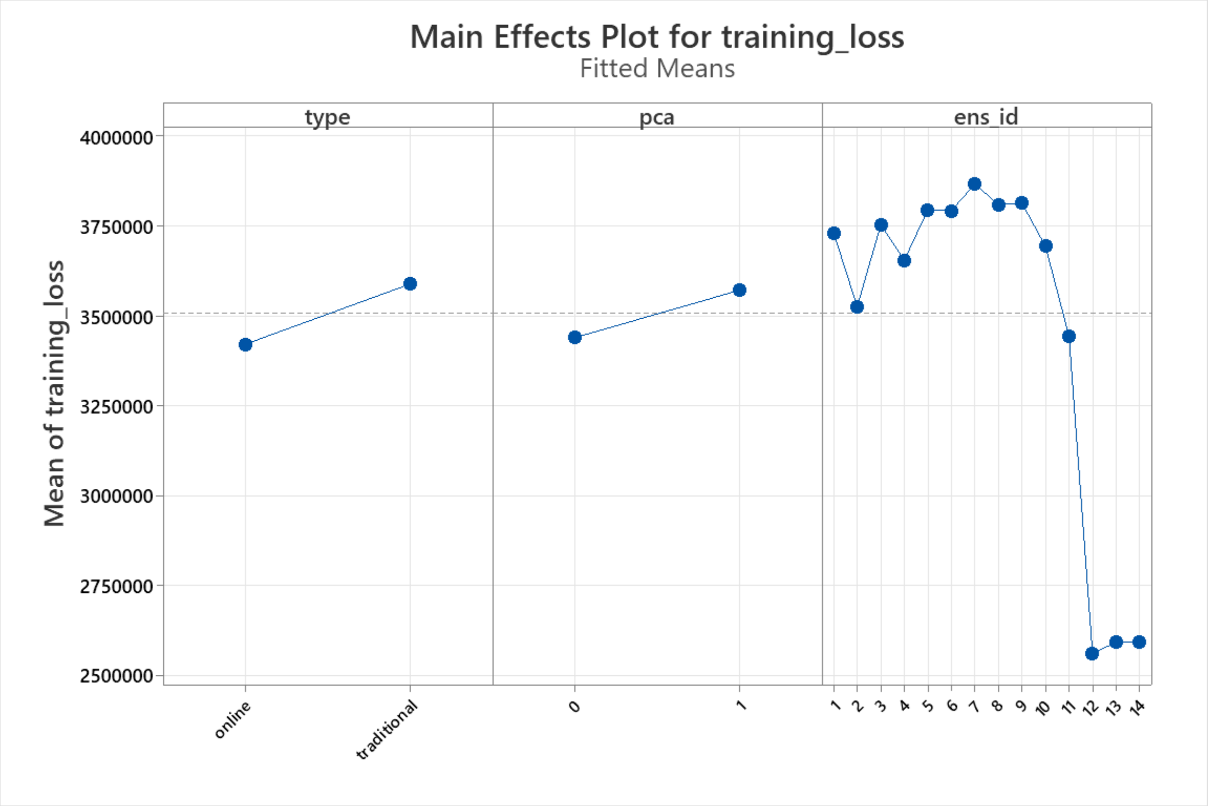
Method

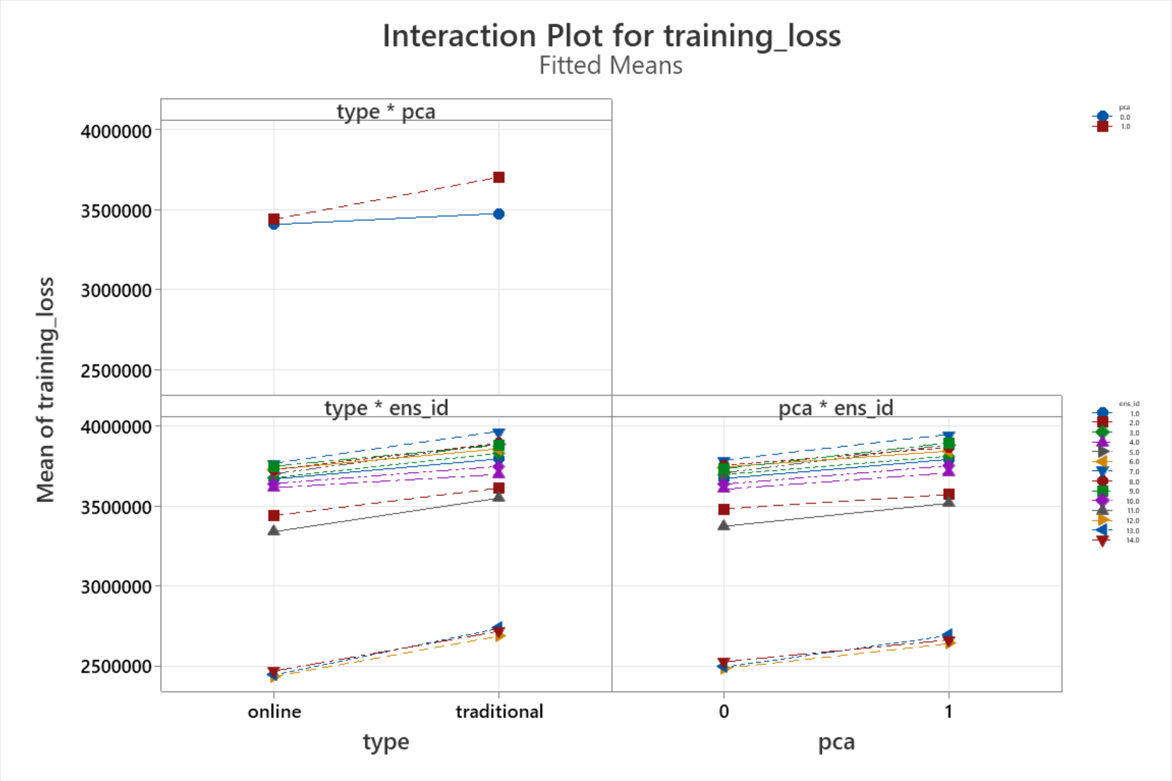
Box-Cox transformation	
Rounded λ	2
Estimated λ	1.55984
95% CI for λ	(0.847340, 2.33134)

Analysis of Variance for Transformed Response

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	42	5.59361E+26	1.33181E+25	191.34	0.000
Linear	15	5.50875E+26	3.67250E+25	527.62	0.000
type	1	1.93212E+25	1.93212E+25	277.58	0.000
pca	1	1.19949E+25	1.19949E+25	172.33	0.000
ens_id	13	5.19558E+26	3.99660E+25	574.18	0.000
2-Way Interactions	27	8.48641E+24	3.14311E+23	4.52	0.003
type*pca	1	6.86432E+24	6.86432E+24	98.62	0.000
type*ens_id	13	1.02402E+24	7.87711E+22	1.13	0.413
pca*ens_id	13	5.98062E+23	4.60048E+22	0.66	0.767
Error	13	9.04865E+23	6.96050E+22		
Total	55	5.60266E+26			

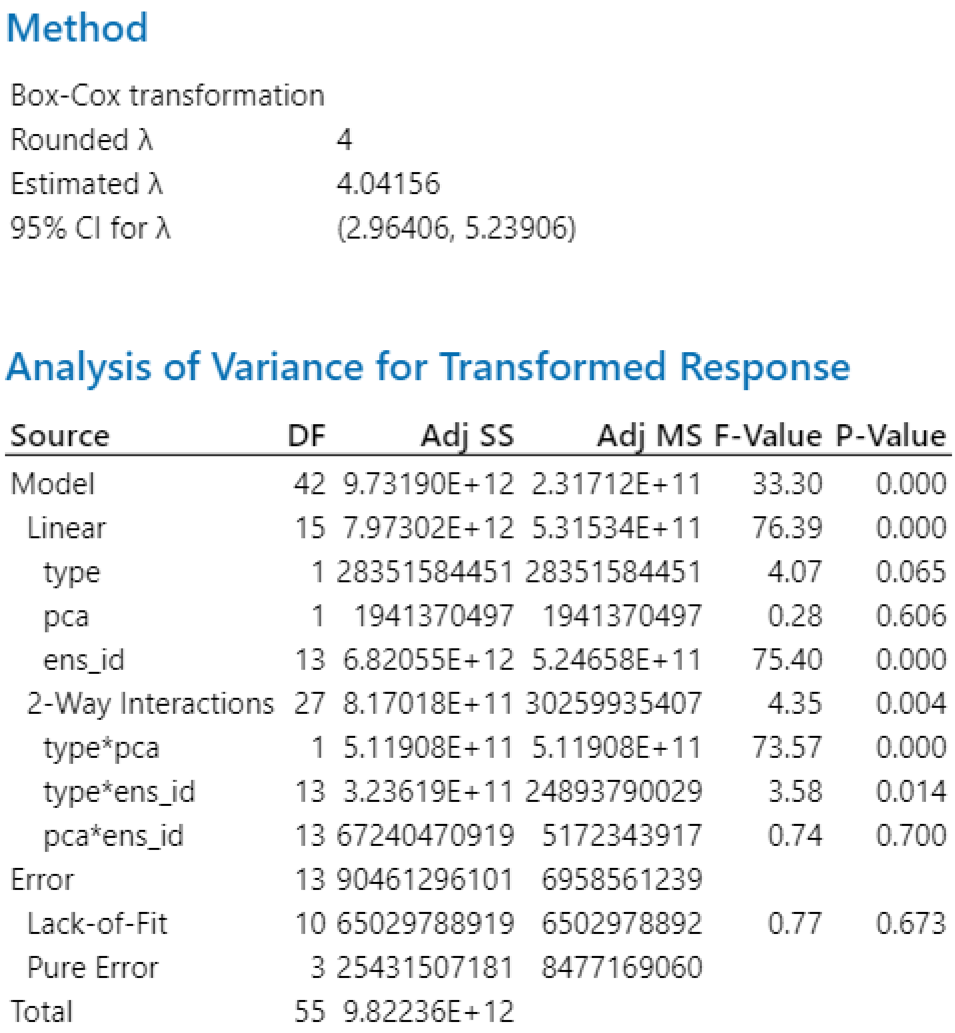
Factor Plots





2.5 Random Forest

ANOVA



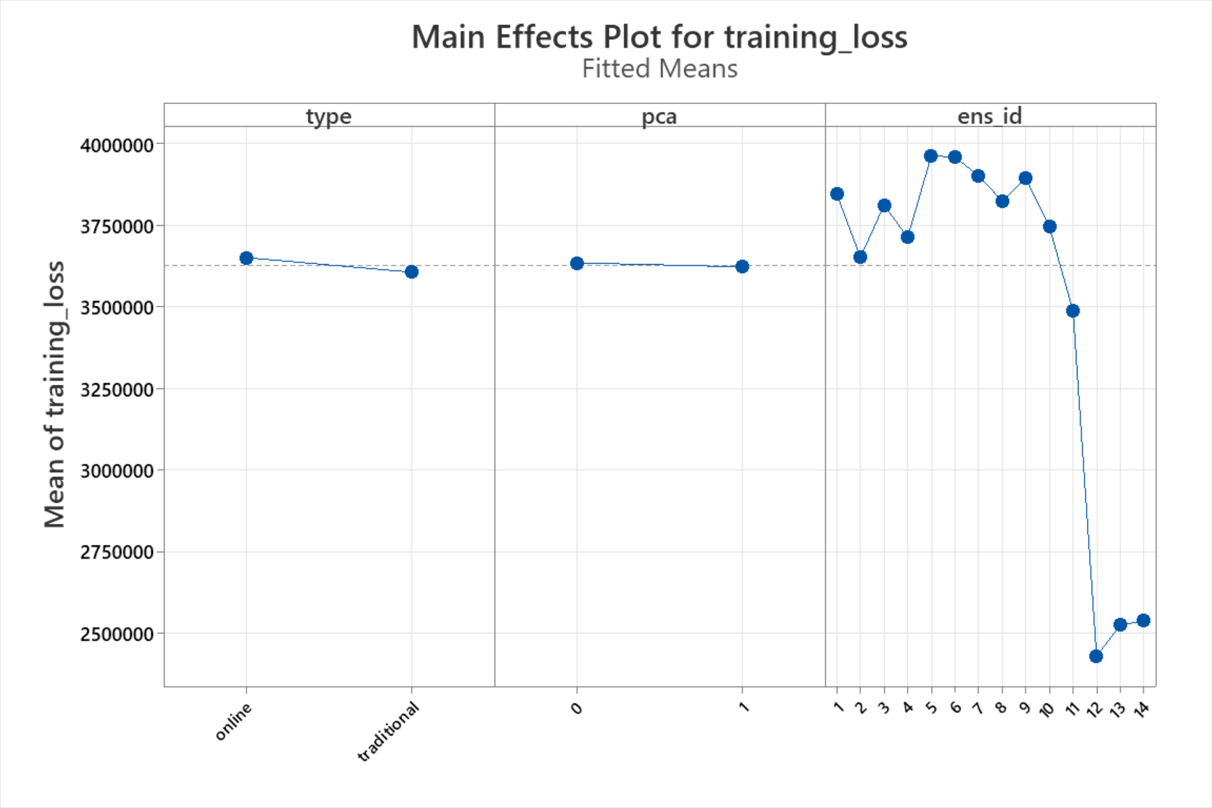
Method

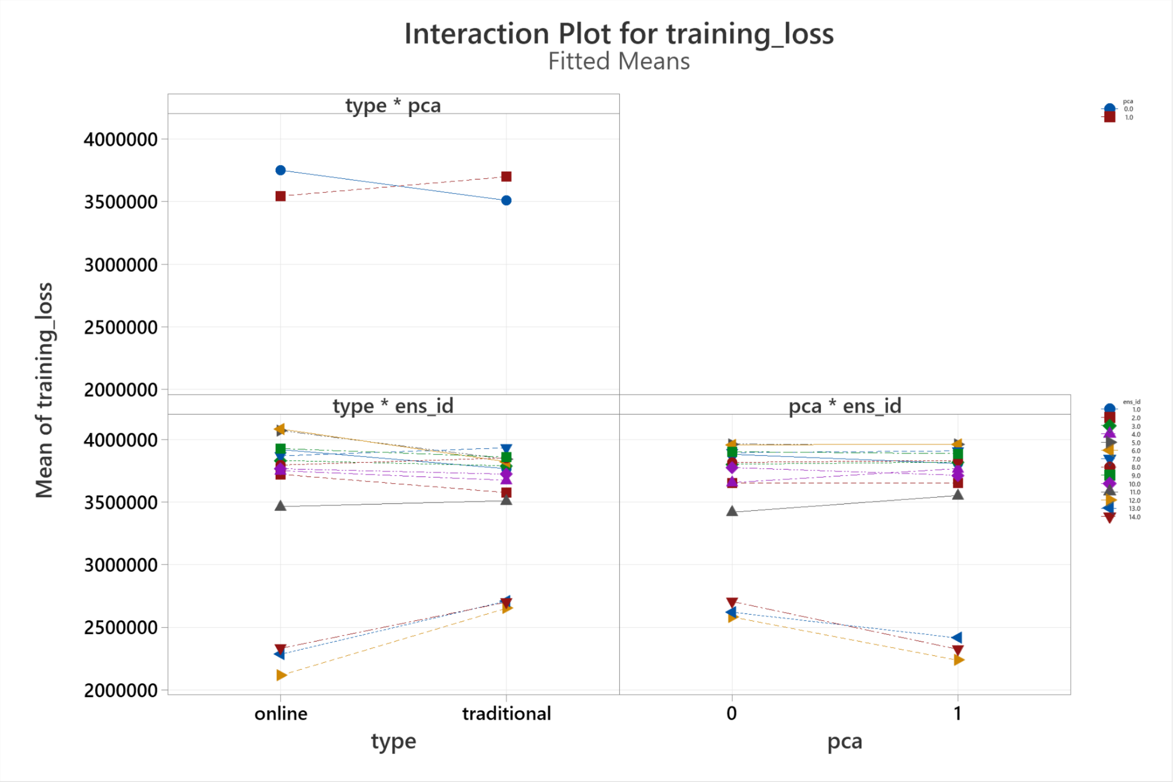
Box-Cox transformation	
Rounded λ	4
Estimated λ	4.04156
95% CI for λ	(2.96406, 5.23906)

Analysis of Variance for Transformed Response

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	42	9.73190E+12	2.31712E+11	33.30	0.000
Linear	15	7.97302E+12	5.31534E+11	76.39	0.000
type	1	28351584451	28351584451	4.07	0.065
pca	1	1941370497	1941370497	0.28	0.606
ens_id	13	6.82055E+12	5.24658E+11	75.40	0.000
2-Way Interactions	27	8.17018E+11	30259935407	4.35	0.004
type*pca	1	5.11908E+11	5.11908E+11	73.57	0.000
type*ens_id	13	3.23619E+11	24893790029	3.58	0.014
pca*ens_id	13	67240470919	5172343917	0.74	0.700
Error	13	90461296101	6958561239		
Lack-of-Fit	10	65029788919	6502978892	0.77	0.673
Pure Error	3	25431507181	8477169060		
Total	55	9.82236E+12			

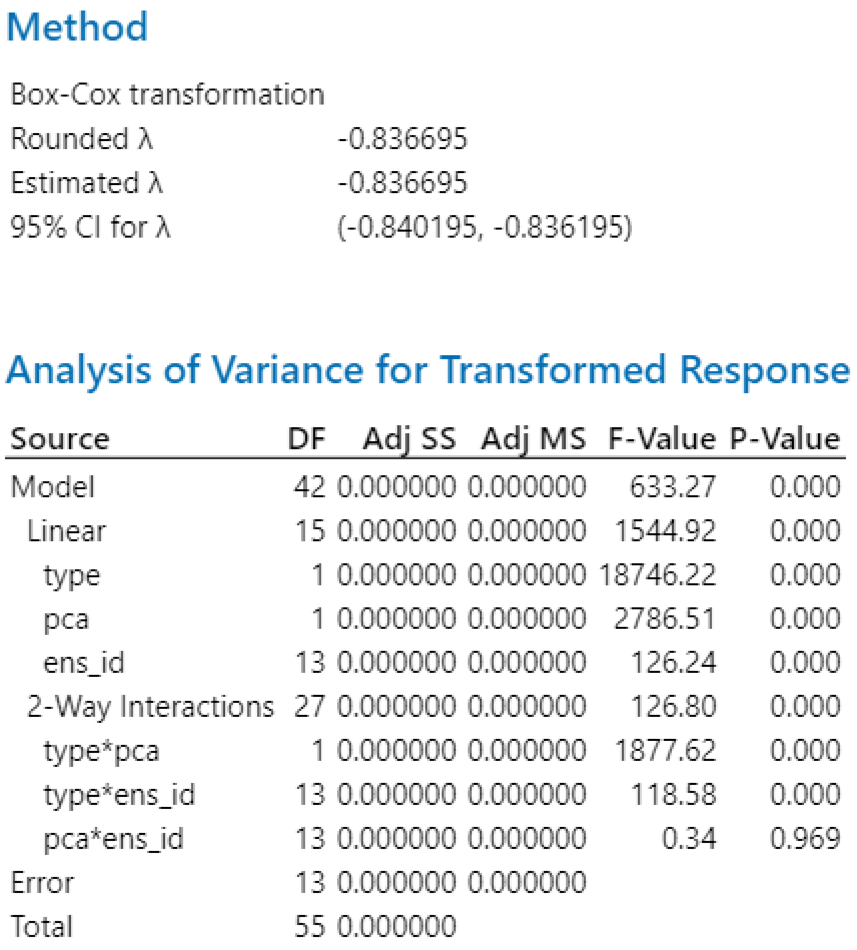
Factor Plots





2.6 ANN

ANOVA



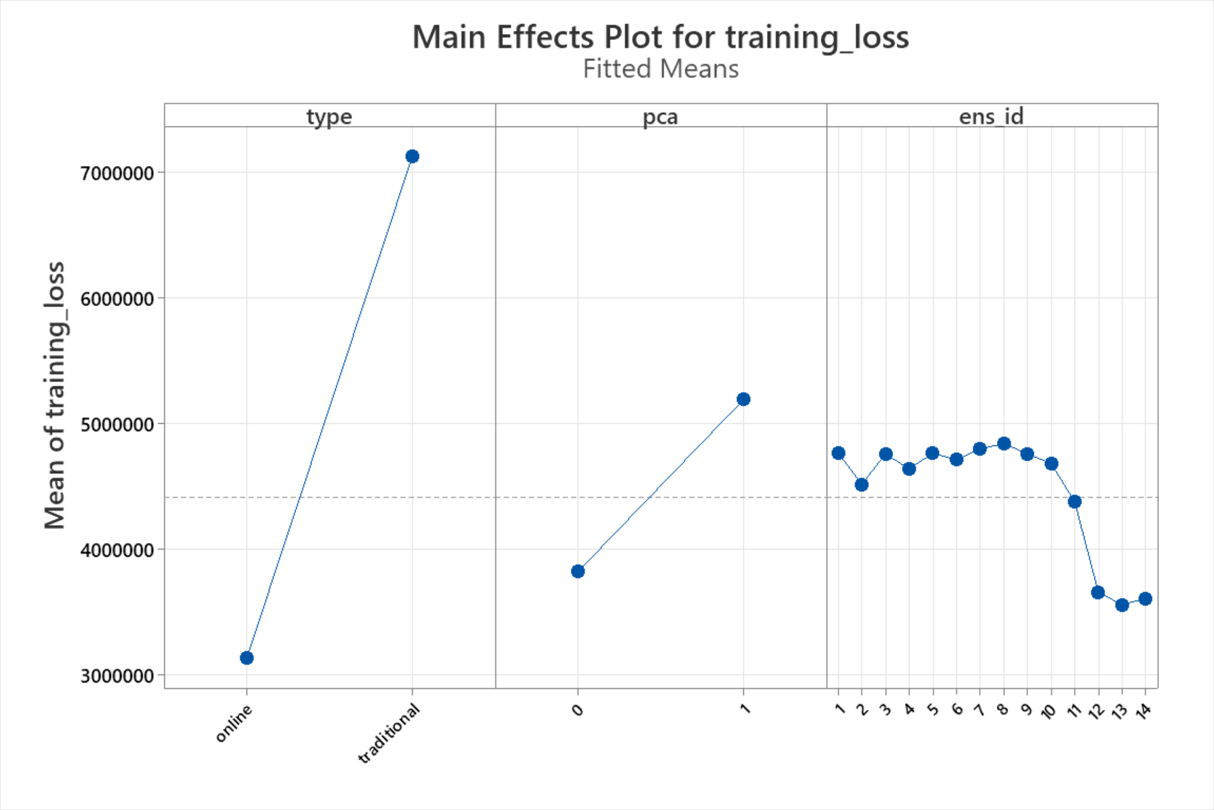
Method

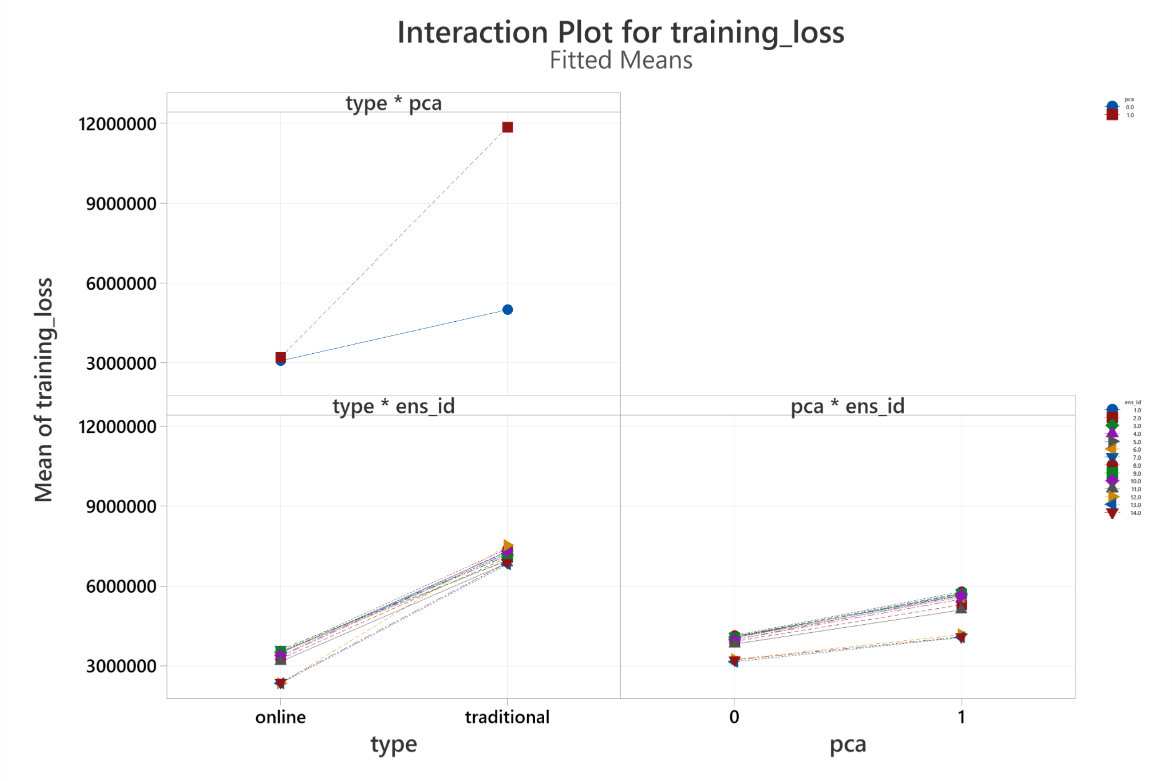
Box-Cox transformation	
Rounded λ	-0.836695
Estimated λ	-0.836695
95% CI for λ	(-0.840195, -0.836195)

Analysis of Variance for Transformed Response

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	42	0.000000	0.000000	633.27	0.000
Linear	15	0.000000	0.000000	1544.92	0.000
type	1	0.000000	0.000000	18746.22	0.000
pca	1	0.000000	0.000000	2786.51	0.000
ens_id	13	0.000000	0.000000	126.24	0.000
2-Way Interactions	27	0.000000	0.000000	126.80	0.000
type*pca	1	0.000000	0.000000	1877.62	0.000
type*ens_id	13	0.000000	0.000000	118.58	0.000
pca*ens_id	13	0.000000	0.000000	0.34	0.969
Error	13	0.000000	0.000000		
Total	55	0.000000			

Factor Plots





Appendix 3: Etik Kurul İzin Belgesi

Appendix 4: Tez Çalışması Orjinallik Raporu