Machine Learning Project - Predicting Airfare Prices

Ruilin Zhou

April 29, 2024

1 Introduction

The airline industry is marked by dynamic pricing strategies, making it challenging for travelers to accurately predict ticket prices, especially as departure dates approach. In this machine learning project, we aim to develop a predictive model that can forecast airline ticket prices. By leveraging historical flight data, with plane type and oil price, we seek to create a model capable of providing travelers with valuable insights into the expected cost of air travel on short notice. This project not only addresses the practical need for reliable ticket price predictions but also delves into the complexities of pricing dynamics within the aviation sector, showcasing the potential of machine learning to enhance decision-making in the travel industry. Through this endeavor, we aspire to empower travelers with the ability to make informed choices and optimize their travel experiences.

2 Dataset and Preprocessing

The dataset is obtained from Kaggle (https://www.kaggle.com/datasets/andrewrliu/airfare) and it consists of the following entries:

- Departure_Date (date of departure in mm/dd/yy format)
- Departure_Day (the departure day of the week)
- Departure_Time (the departure time)
- Plane_Type (type of plane, consisting of Boeing 737, Boeing 737Max Airbus A319, Airbus A320, and Embraer 175)
- Oil_Price (closing price of Brent Crude oil on the day of ticket sale)
- Days_In_Advance (number of days ahead of departure date)
- Sale_Day (the sale day of the week)
- Price_Increase (boolean value on whether the price of the ticket increased compared to the previous day)
- Price1/2/3 (price of ticket 1/2/3 days ago before purchase)
- Ticket_Price (price of ticket for the specific flight given the number of days ahead of departure date)

2.1 Feature and Target variable selection

Since the goal is to get an estimate of the ticket prices, the ticket price is selected as the target variable. The Price_Increase feature will leak information about the ticket price since it used ticket price as comparison, so it is removed from the features.

Considering the time period of the flight, the original dataset consists of flights in year 2010 and 2023. Since the year span is quite large and each year has enough data, I decided to use flights that are only in 2023.

2.2 Encoding

Many features are not numerical values, as a result, they are encoded in the following way:

- Departure_Date: the date of the month is extracted since all flights are in October.
- Departure_Day, Sale_Day: Ordinal encoding used since weekdays has natural ordering. Monday encoded as 0 and Sunday as 6.
- Departure_Time: Converted to 24 hour format and decimal representation.
- Plane_Type: One-hot encoding are there is no order in types.

Dataset before encoding:

Departure_Date	Departure_Day	Departure_Time	Plane_Type	Oil_Price	Days_In_Advance	Sale_Day	Price3	Price2	Price1	Ticket_Price
10/13/23	Friday	1:30pm	B737	82.92	57	Thursday	194	194	193	185
10/13/23	Friday	1:30pm	B737	84.24	56	Friday	194	193	185	186
10/13/23	Friday	1:30pm	B737	84.99	55	Saturday	193	185	186	194
10/13/23	Friday	1:30pm	B737	85.56	54	Sunday	185	186	194	192
10/13/23	Friday	1:30pm	B737	84.91	53	Monday	186	194	192	192

Dataset after encoding:

Departure_Date	Departure_Day	Departure_Time			Plane_Type			Oil_Price	Days_In_Advance	Sale_Day	Price3	Price2	Price1
13	4.0	13.5	0.0	0.0	1.0	0.0	0.0	82.92	57	3.0	194	194	193
13	4.0	13.5	0.0	0.0	1.0	0.0	0.0	84.24	56	4.0	194	193	185
13	4.0	13.5	0.0	0.0	1.0	0.0	0.0	84.99	55	5.0	193	185	186
13	4.0	13.5	0.0	0.0	1.0	0.0	0.0	85.56	54	6.0	185	186	194
13	4.0	13.5	0.0	0.0	1.0	0.0	0.0	84.91	53	0.0	186	194	192

3 Supervised Analysis Results and Discussion

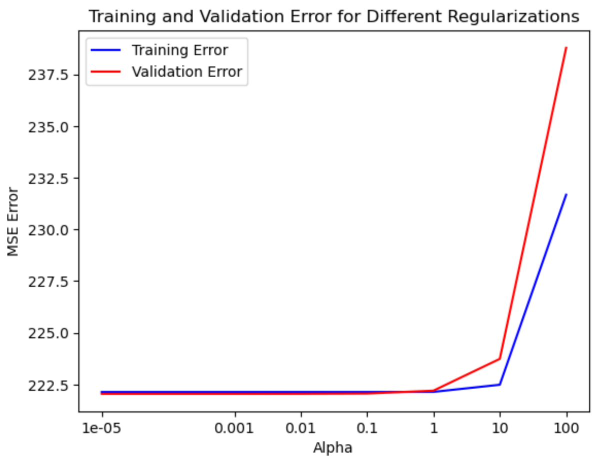
External libraries used are NumPy, scikit-learn and Matplotlib.

The models chosen are linear regression, support vector regression, and neural networks. Standardization used is zero mean and unit variance. Metrics chosen for validation is mean squared error (MSE).

Regularization parameter choice are from log scale (denoted as α for linear regression and nerual networks, C for SVR). Polynomial degree are chosen starting from 1 to 5 for linear regression and 1 to 3 for neural networks with consideration to computing time. For SVR, ϵ values are chosen with consideration to MSE error from predicted results of linear regression model. Feature transformation are commonly used RBF and polynomial kernels for linear regression and SVR. Since RBF corresponds infinite dimensional space, for neural networks, only polynomial feature transformation is considered and activation functions are commonly used logistic function and tanh function.

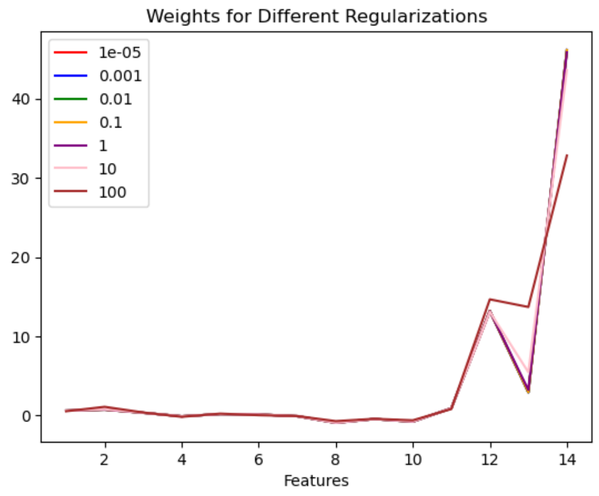
3.1 Linear regression

The figure show MSE error for different L2 regularization.



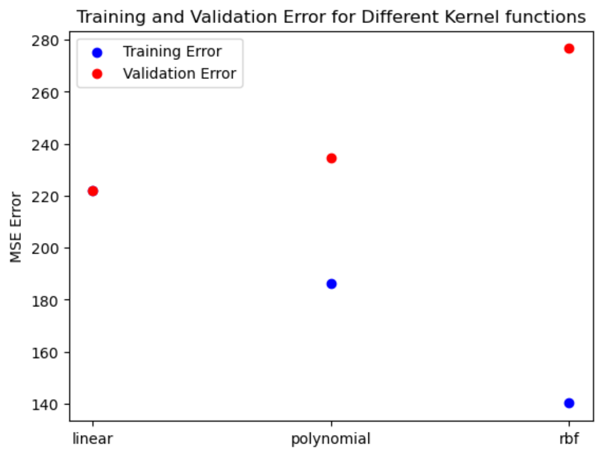
When regularization is small, Training error and validation error are both small, this indicates that we are not over-fitting nor under-fitting. When regularization increases, both errors increases, indicating that we are under-fitting, the model is too simple to capture the relationship. The variance is low but bias is high.

The figure show scale for feature weights for different L2 regularization. Corresponding features are Departure date, Departure day, Departure time, Plane type (5 slots), Oil price, Days In Advance, Sale day, Price of the ticket 3/2/1 days ago.

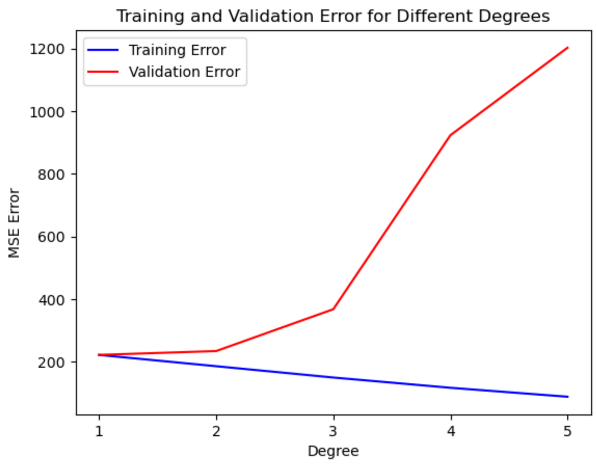


The weights suggests that while we conjectured that many features might influence the ticket price. The most influential features are the prices 3/2/1 days ago, especially the closer the date, the higher the influence. The higher the regularization, the smaller in scale the weights are.

The figure show MSE error for different feature transformations (different kernel functions).



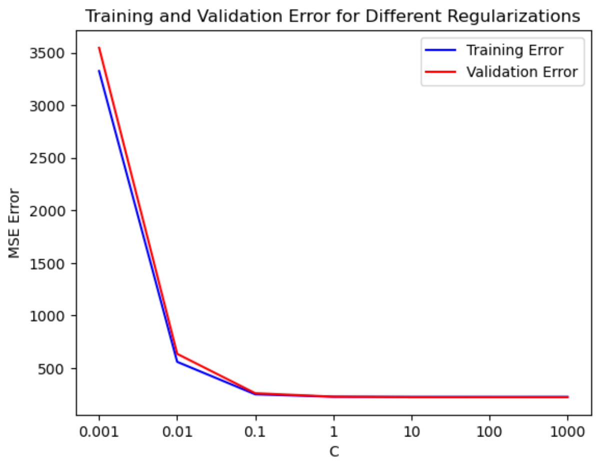
The figure show MSE error for different parameters for polynomial feature transformation.



As we apply more complicated feature transformations, we are over-fitting the data. The training error is low but validation error is high. There is high variance and low bias.

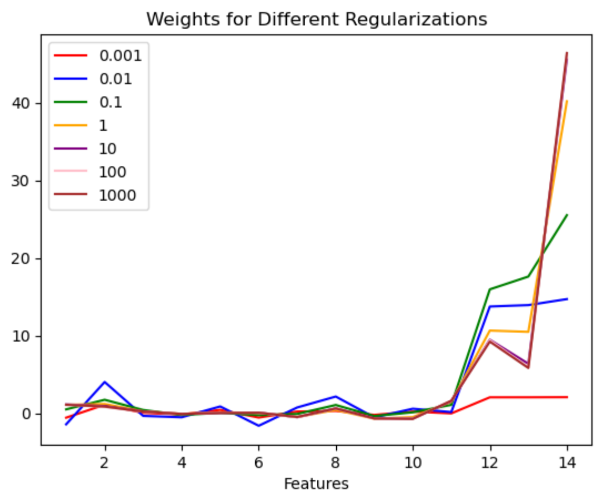
3.2 Support vector regression

The figure show MSE error for different penalty.



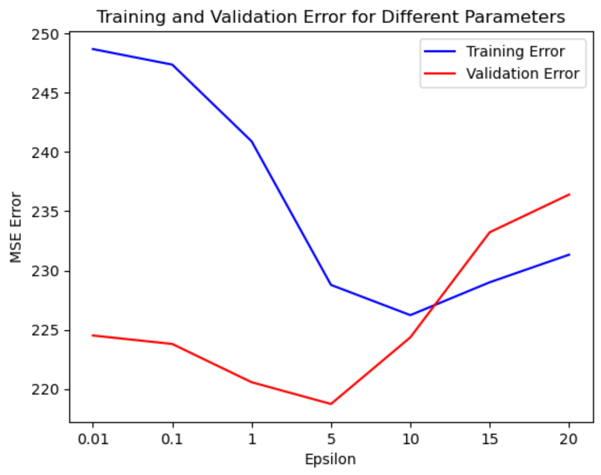
As penalty increases, regularization decreases, we move from under-fitting to the sweet spot where both errors are low.

The figure show scale for feature weights for different penalty. Corresponding features are Departure date, Departure day, Departure time, Plane type (5 slots), Oil price, Days In Advance, Sale day, Price of the ticket 1/2/3 days ago.



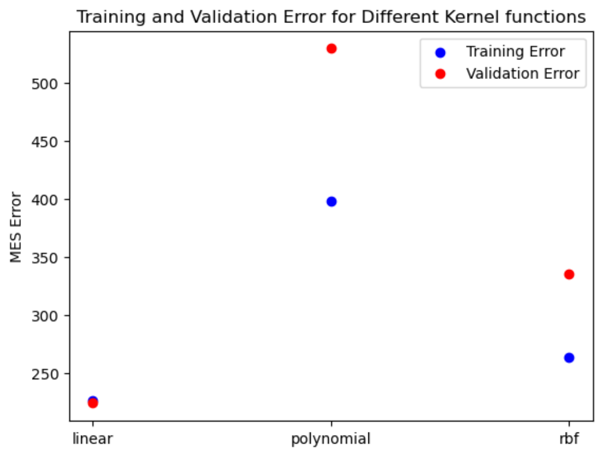
The weights suggests that while we conjectured that many features might influence the ticket price. The most influential features are the prices 3/2/1 days ago, especially the closer the date, the higher the influence. The higher the regularization, the smaller in scale the weights are.

The figure show MSE error for different parameters (ϵ is the margin that target can be from the sample without penalty).



The sweet spot for ϵ is around 5. When ϵ is too small, the model is under-fitting. The variance is low but bias is high. When ϵ is too large, the model is over-fitting. There is high variance and low bias.

The figure show MSE error for different feature transformations (different kernel functions).

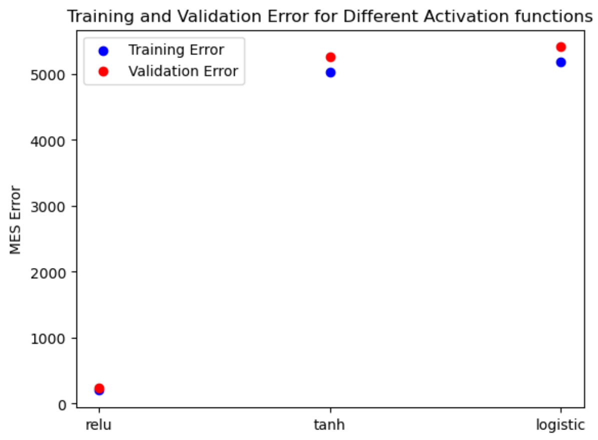


The relationship in the dataset is simple and it seems that feature transformation is not needed.

3.3 Neural networks

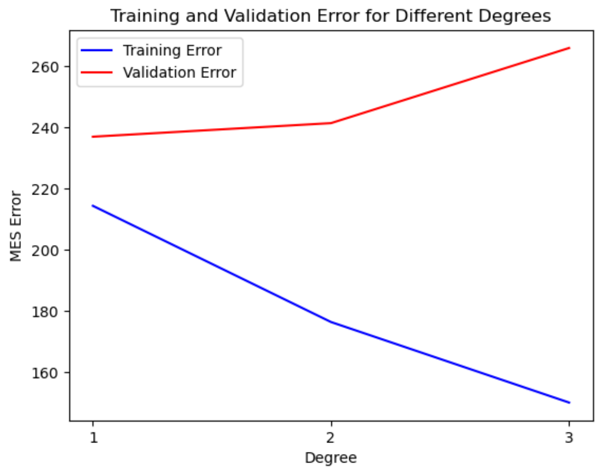
The previous models shows that the relationship is not complex, as a result, for nerual networks, only 2 hidden layers are used.

The figure show MSE error for different parameter. (different activation functions).



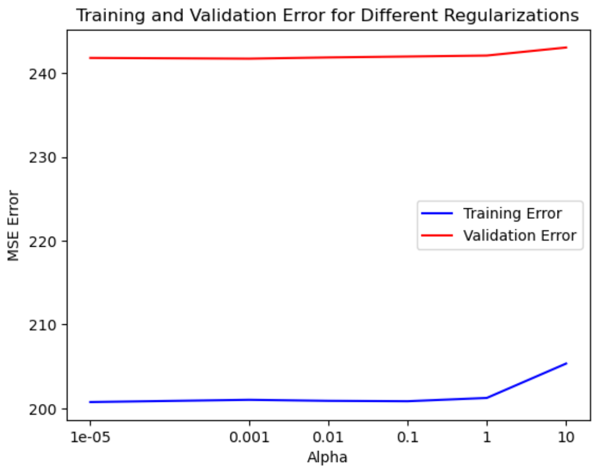
The tanh and logistic activation functions are similar in shape and they have similar performance, but it is much worse compared to Relu.

The figure show MSE error for different polynomial feature transformation.



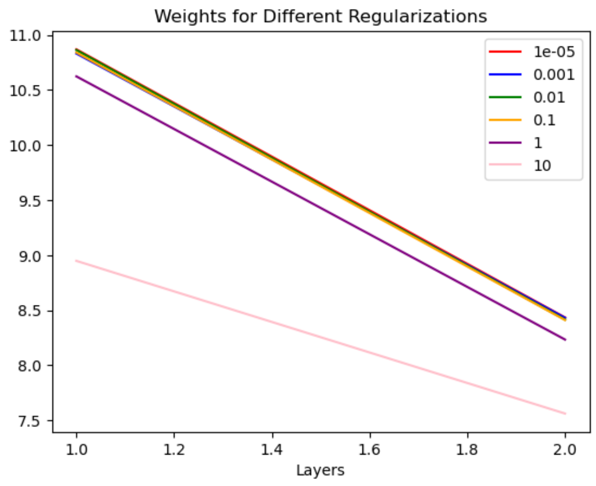
As the degree increases, we are over-fitting the data, the training error is low but validation error is high. There is high variance and low bias.

The figure show MSE error for different L2 regularization.



It is neither over-fitting nor under-fitting.

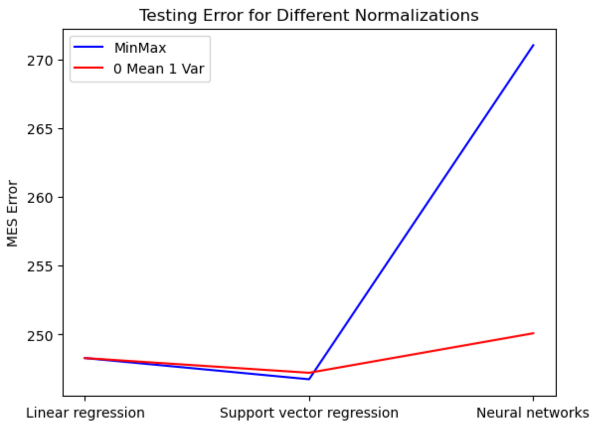
For neural networks, the weights are hard to visualize, so for each layer, I calculated the matrix norm.



As regularization increases, the scale of the weight matrix decreases.

3.4 Different normalization

Minmax normalization is also used. Using standardization with zero mean and unit variance, optimal parameters are selected and test error are calculated. Using same paramaters but with Minmax normalization, test error are also calculated and compared.



While the weights are optimized when using the standardization with zero mean and unit variance, the test error do not change when linear regression is applied. This is because linear regression library in sklearn uses closed form solution. The test error decreases when support vector regression is applied. This is because SVR is essentially solving convex optimization problem. However, since Neural Networks uses stochastic gradient descent optimizer, change normalization would affect the results as the parameters are optimized for standardization with zero mean and unit variance.

4 Table of Results

4.1 Linear regression

Table 1: Mean Squared Error for Validation Set and Training Set

α	**Validation Set**	**Training Set**
10^{-5}	222.05	222.14
0.001	222.05	222.14
0.01	222.05	222.14
0.1	222.06	222.14
1	222.20	222.14
10	223.74	222.49
100	238.78	231.68

Table 2: Mean Squared Error for Validation Set and Training Set

Kernel	**Validation Set**	**Training Set**
Linear	222.06	222.14
Polynomial	234.53	186.16
RBF	276.64	140.16

Table 3: Mean Squared Error for Validation Set and Training Set

Degree	**Validation Set**	**Training Set**
1	222.27	222.14
2	234.53	186.16
3	368.05	150.03
4	923.42	117.28
5	1202.75	88.77

From the results, the simple linear regression with no feature transformation performs the best.

4.2 Support vector regression

Table 4: Mean Squared Error for Validation Set and Training Set

Kernel	**Validation Set**	**Training Set**
Linear	224.35	226.23
Poly	529.42	398.39
RBF	335.68	263.99

Table 5: Mean Squared Error for Validation Set and Training Set

C	**Validation Set**	**Training Set**
0.001	3546.26	3325.12
0.01	635.69	559.63
0.1	261.48	250.86
1	228.75	227.39
10	224.35	226.23
100	223.94	226.21
1000	223.89	226.28

Table 6: Mean Squared Error for Validation Set and Training Set

ϵ	**Validation Set**	**Training Set**
0.01	224.51	248.69
0.1	223.79	247.37
1	220.56	240.90
5	218.73	228.78
10	224.35	226.23
15	233.22	229.00
20	236.41	231.33

From the table, the best parameters are $C = 100, \epsilon = 5$.

4.3 Neural networks

Table 7: Mean Squared Error for Validation Set and Training Set

α	**Validation Set**	**Training Set**
10^{-5}	241.83	200.75
0.001	241.74	201.02
0.01	241.89	200.90
0.1	242.00	200.85
1	242.11	201.24
10	243.07	205.34

Table 8: Mean Squared Error for Validation Set and Training Set

Activation	**Validation Set**	**Training Set**
relu	241.99	200.85
tanh	5259.77	5037.87
logistic	5411.42	5183.81

Table 9: Mean Squared Error for Validation Set and Training Set

Degree	**Validation Set**	**Training Set**
1	237.00	214.40
2	241.44	176.42
3	265.99	150.10

From the table, the optimal is to use $\alpha = 1$, degree 1 transformation and Relu.

4.4 Different normalization

Minmax normalization is also used. Using standardization with zero mean and unit variance, optimal parameters are selected and test error are calculated. Using same paramaters but with Minmax normalization, test error are also calculated and compared.

Table 10: Mean Squared Error for Test Set

Model	**MinMax**	**0 Mean 1 Var**
Linear regression	248.29	248.29
Support vector regression	246.75	247.22
Neural networks	271.02	250.09

5 Conclusion

For linear Regression, the best-performing model was the simple linear regression without feature transformation. Regularization was crucial in preventing overfitting, but for this task, a lower regularization is preferred.

For Support Vector Regression, it performed well with a linear kernel, achieving optimal results with $C = 100, \epsilon = 5$. Feature transformation didn't significantly improve performance.

For Neural Networks, using Relu activation function, $\alpha = 1$ for L2 regularization, and a degree 1 polynomial feature transformation yielded the best results. The models were sensitive to the choice of activation function.

For Normalization Comparison, while min-max normalization showed slightly better performance in SVR, it didn't significantly affect linear regression results. However, in neural networks, it performed worse.

Overall, each model had its strengths and weaknesses, but SVR with a linear kernel and appropriate parameter tuning ($C = 100, \epsilon = 5$) seemed to perform the best for this predictive task. The 3 model's performance do not differ very much with appropriate parameters. And feature transformations is not necessary due to linearity of the data. While Neural Networks is generally expected to capture the trend in the data as it can model complex functions. It turns out that airfare ticket prices are fairly predictable given previous price history. In addition, while it is generally reasonable to assume that the time when you purchased the ticket would affect the price, this is not shown through the weights. This is because the time of purchase is implicitly reflected in the prices 1/2/3 days ago. Other information about the flight and oil price are much less significant. In this project, we investigated the importance of regularization, feature transformation, and normalization techniques in enhancing model performance.