Machine Learning Mini Project report

Xinghao Wu 2708274

Nguyen Tien Dat Tran 2871790

Project_task:*develop a model to predict the EPEX Spot day-ahead auction prices in the German-Austrian market*.

We have 3 main scripts:

- **Master_1.m.**
- **Master_2.m.**
- **Master_3.m.**

  We will explain in detail each Master script below.

The following data set we used for this task:

- **Price_train.csv**:EPEX day-ahead prices in EUR/MWh. Each row contains the values for the 24 hourly prices for the date in the first columns. This is our target variable.
- **Load_train.csv**:Forecasted hourly values for the electrical load in *MWh*.
- **PV_train.csv**:Forecasted hourly values for the in-feed from PV power plants in *MWh*.
- **Wind_train.csv**:Forecasted hourly values for the in-feed from Wind power plants in *MWh*.
- **Fuels_train.csv**:Daily prices for CO2 emission allowances and natural gas in *EUR/t* and*EUR/MWh*
- **Holidays_train.csv**: A list of German federal holidays.

***The following statements are the most important factor that we always keep in mind during our model design*:**

- ***Data training size*** *(how many samples is sufficient for training and cross validation)*
- ***Features*** *(Should we use more feature or just using 1 feature like above is efficient)*
- ***Polynomial Hypothesis Fuction****(which function and to which degree is sufficient, since if the function is so simple, the error is therefore very high)*
- ***Regularization Factor Lambda*** *(it is used when the model is suffering under overfitting)*So to demonstrate the role of these factors, we will go from very basic method to advanced method by changing these factors and explain why we did that.

1. Task Analyse.

To keep it simple, we first predict the price of each hour separately. So we have 24different models to predict different 24 hours, because according to the price table, the price behave differently over time. Using 1 common model to predict all hours is very hard.

Figure1:(image of price each hour)

What Features we choose and why we choose it?

- At first we want predict the price of hour 14 (column 15 of price table), because in this time the data distribution of the energy production is relative stable in every year.
- Then we will predict the rest 23 hours similarly (just change the hyper-parameter). We use only 1 feature X = Wind + PV-Load to keep thing as simple as possible.
- The reason why we choose this feature is due to the fact that the energy left after substracting the energy production(Wind+PV) from Load has huge impact on the price.
- Indeed when we draw the “price”corresponding to “Wind+PV – Load”, we can see that the price is lower when we have more energy redundancy, and the data itself forms a curve, so we can make a curve through these data points:

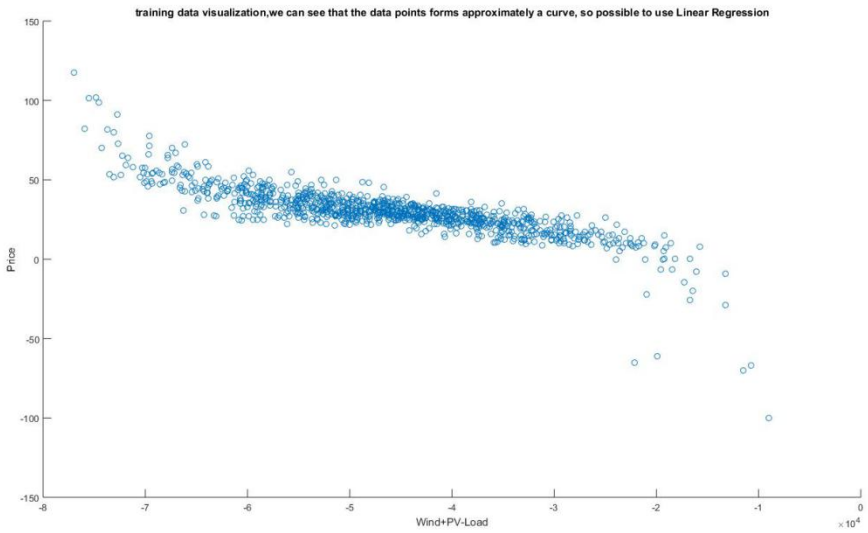


Figure 2:data_train_distribution for feature X

2. Task Implements.

Now we know that it is a simple way to use linear regression in this case, let's implement it.

(1) some details for Model designing

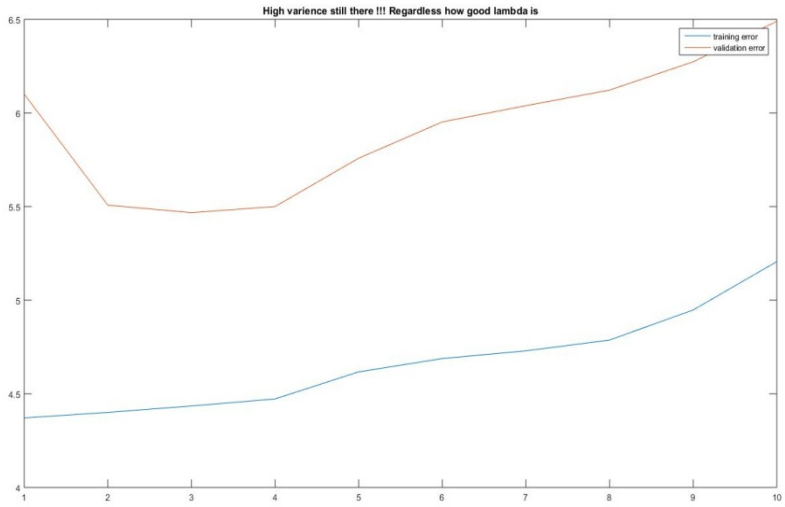
- **Algorithm:** Linear Regression with Polynomial degree
- **Feature** : Wind+PV - Load
- **Predict:** Price of hour 14.
- **Validation method:**cross-Validation

*Because we are not be sure that,how many samples that we should choose as the data train, so at first we just **run Master_1.m** and see its result:*

- **Set Data_trainsize:** 30 (**very small**)
- **Set Data_validation**: Far away from the Data_Train, in order to ensure the data is i.i.d.**(the code line 43-44)**
- **Set Polynomial function and degree:** degree 10. (to **avoid high bias**).
- **Set Different values of lambdas**:lambdas=[0 0.001 0.005 0.01 0.1 0.5 1 2 5 10].

*To make sure that our model is **not going to suffer from high bias**, we set degree on 10 for our polynomial function(very complex function for a small data_trainsets, so quite easy to fit through all training points).*

*If the behavior of the price_train is so simple, then a function can fit well for data_train andit can also fit well for validation data, and our model is well done!But unfortunately, 30 points of data is not sufficient to cover the model of price,so the behavior of the price is not simple and we **end up having high error on validation data**, because of high variance, no matter how good the lambda we choose to regularize model.*



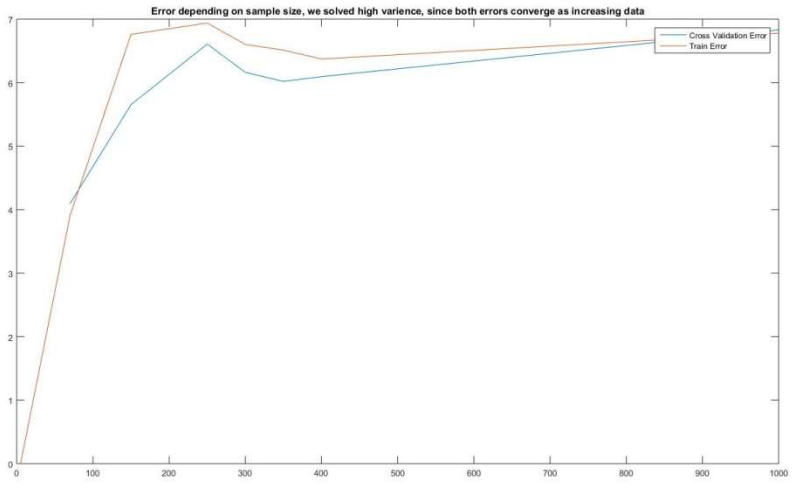
Solution: Lambda is not in option, because even with best lambda we still have high variance. So increasing number of sample might help, but to which number of sample should we increase?. To answer this question, we do increase the number of sample, do training, and plot the **LEARNING CURVE** to analyze.

(2) Model modifying

Run Master_2.m. Instead of fix sample size in **Master_1.m**, in this script we set different sample size for train data, upto 1000:

- **Data_train size**: varies from [5...upto...1000]
- **Data_validation**: obtain by splitting the Data_train into 10 folds.
- **Polynomial function**:degree 10(as the same as the **Master_Mini_project.m**).
- **Set Different values of lambdas**:lambdas=[0 0.001 0.005 0.01 0.1 0.5 1 2 5 10].

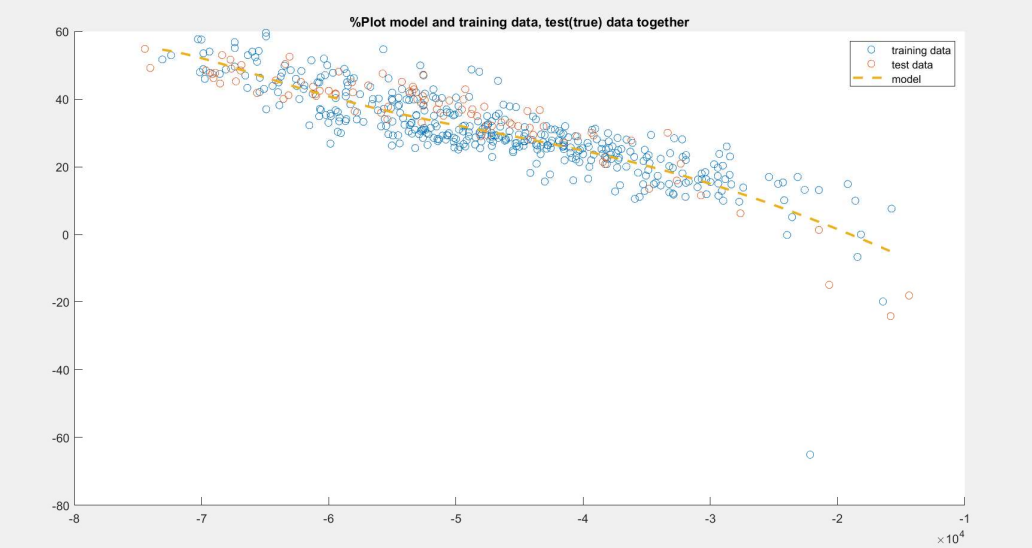
*We solved the high varience, since we could see, both errors converge as we increase the sample size, **but high bias**:(the best_lambda=0 and the data_train size:1000)*



Although it is suffering from **high bias** with RMSE error is about 6 Euro in price(picture above), we still have a good model:

Why do we have high bias?

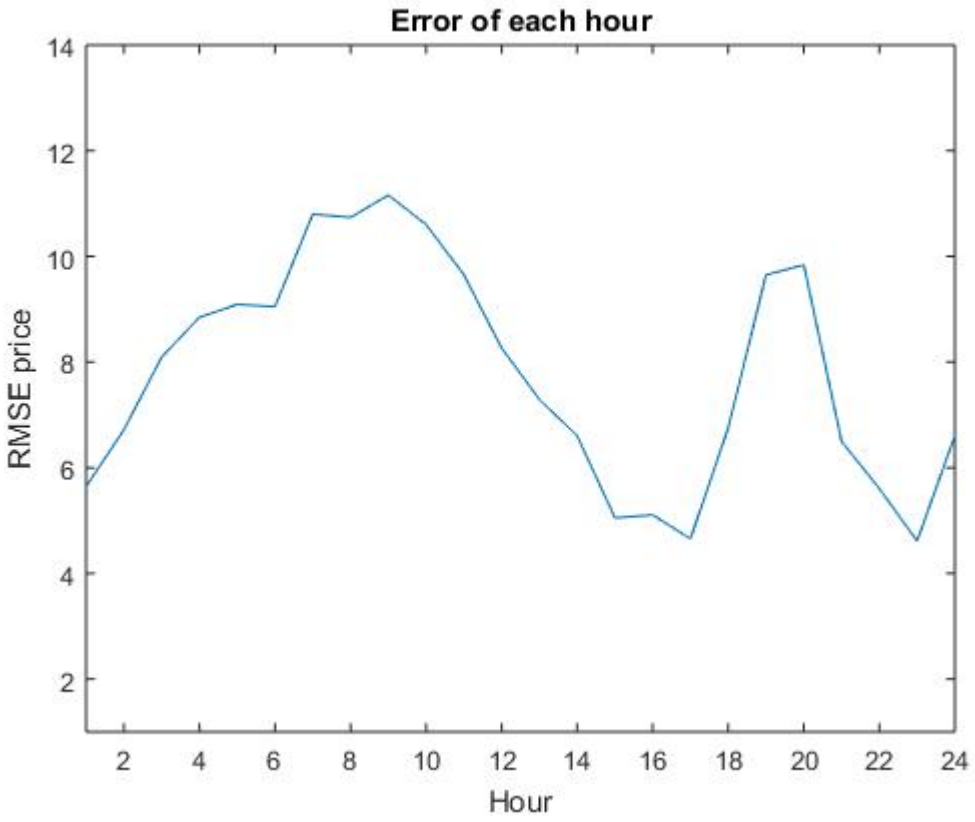
- Because there are a lot of data points that our function can not fit all of them, so we have **high bias** . So solving high variance leads to high bias.
- Solution recommended:
  - Use only **400 Sample** of Training Data, since 1000 sample is redundant.
  - Make the polynomial function more complex to fit points better, or add more feature(means more useful information) to predict more precisely.



(3) **Use modified model in step 2 to predict all 24 hours:**

Run Master_3.m to predict the price of all 24 hours by using 24 different model.

- Simular to step 2 above, we repeat the above process with the rest 23 column and we can predict all price for every hour
- RMSE error for each hour:



3. Conclusion

- Pros: Lieanr Regression is simple to implement.
- Cons: But the RMSE is still high, since the price is non-linear. Neural Network can approximate the non-linear function better than linear regression. So Neural Network is recommended.