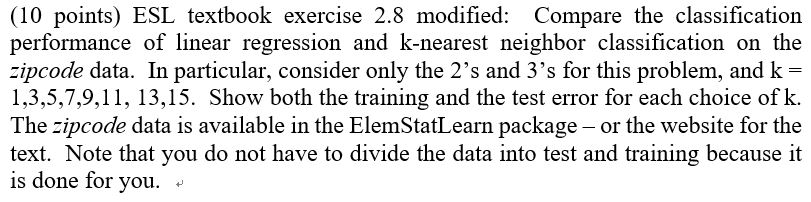
Question 2.

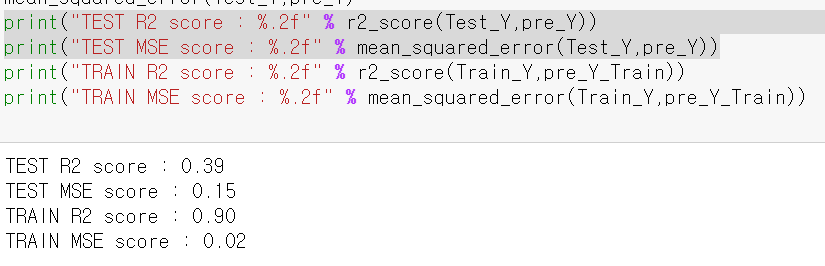


Process:

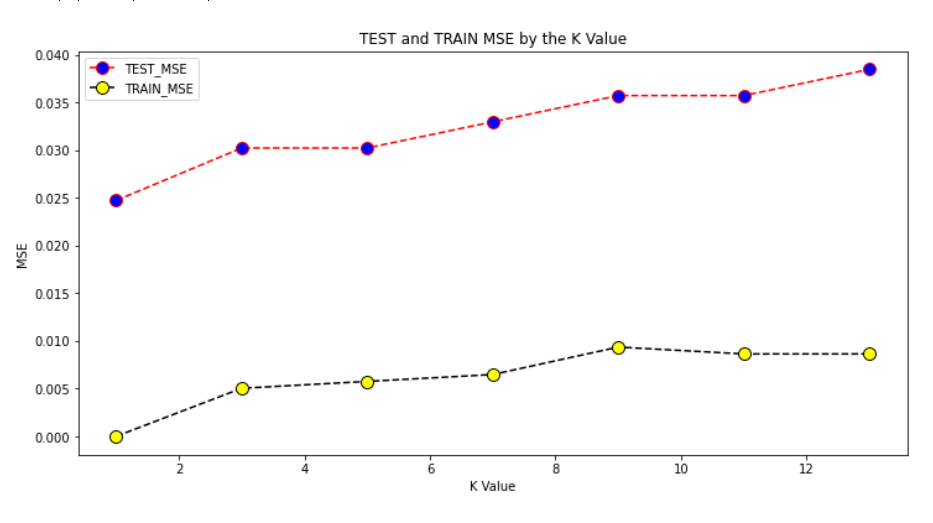
1. **Read data**: I read CSV data ‘train.csv’ and ‘test.csv’ with function ‘read\_csv’
2. **Extract targeted data**: I extracted the targeted data in the train set as well as the test set using ‘or’ method to merge two(2,3) target index.
3. **Separate target(Y):** I separated both train and test target values.
4. **Fitting the Linear model with the data**: I used the ’lm’ function to make the Linear model.
5. **Get MSE and R2\_score of the Linear model**: To analysis the accuracy of the model, I get MSE and R2 values on the linear model.
6. **Fitting the KNN model with k=1 to K=15:** K value in the KNN model should be an odd number. If the K number is an even number, there will be some undecided area.
7. **Show MSE and R2\_score to evaluate the model by K values:** By the K value, I also calculated the MSE and R2 score of Trainset and Test.

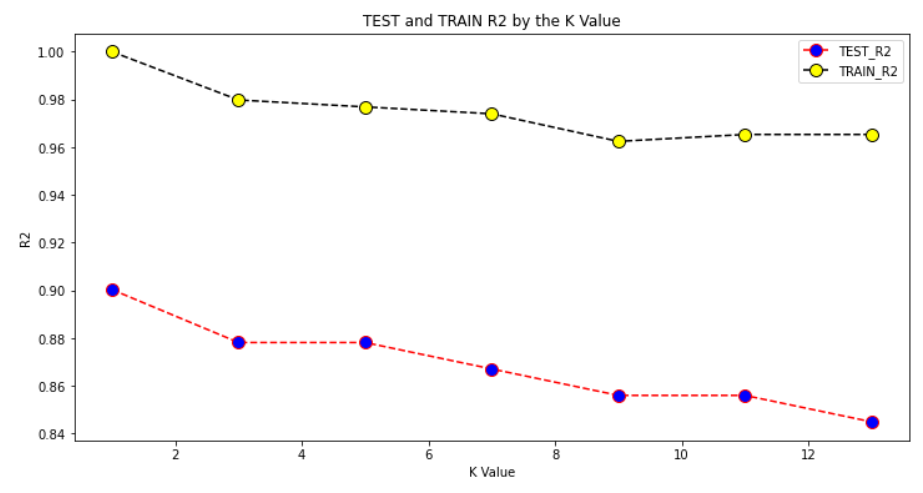
Outputs:

Linear Regression Model for the DATA (MSE, R2)



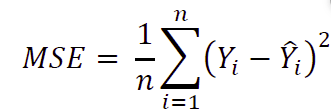
KNN Model for the data





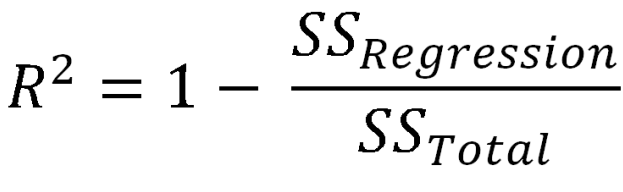
Before Consideration

**MSE(MEAN SQUARED ERROR)**



The difference between actual and predicted output is first squared and then summed up and divide it by n.

**R-SQUARED SCORE**



R-Squared score shows how much data is explained by our Regression model. A model that explains much of the data then it is a good model.

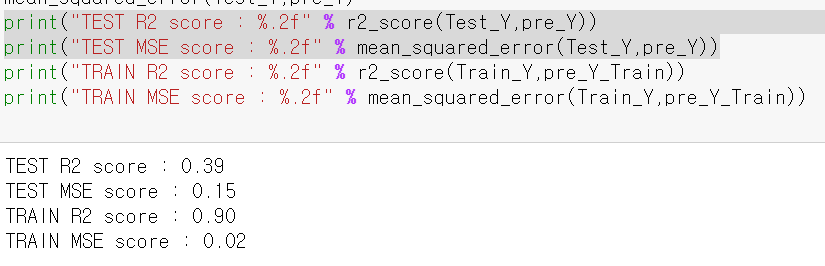
In summary.

Low MSE => Low difference between predicted data and actual data

High R2 SCORE => The data can Highly explain the model.

Discussion:

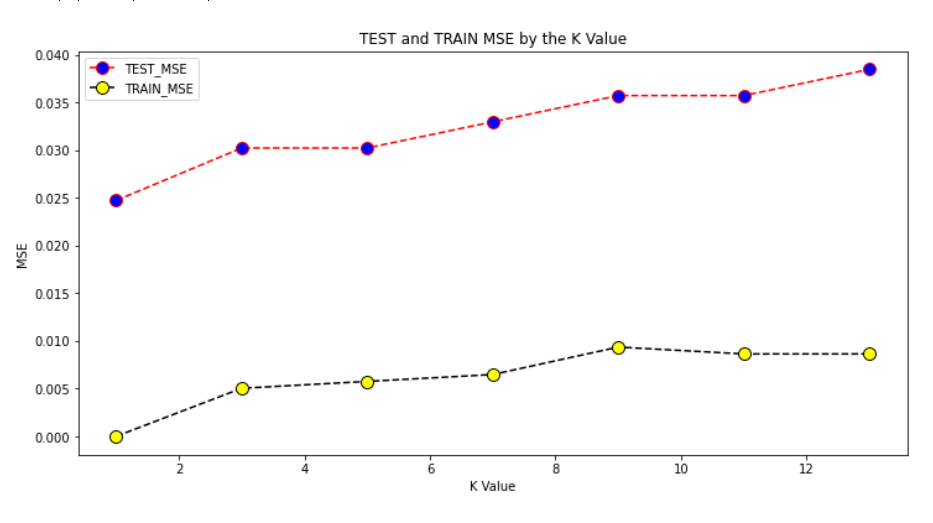
**Linear Regression:**



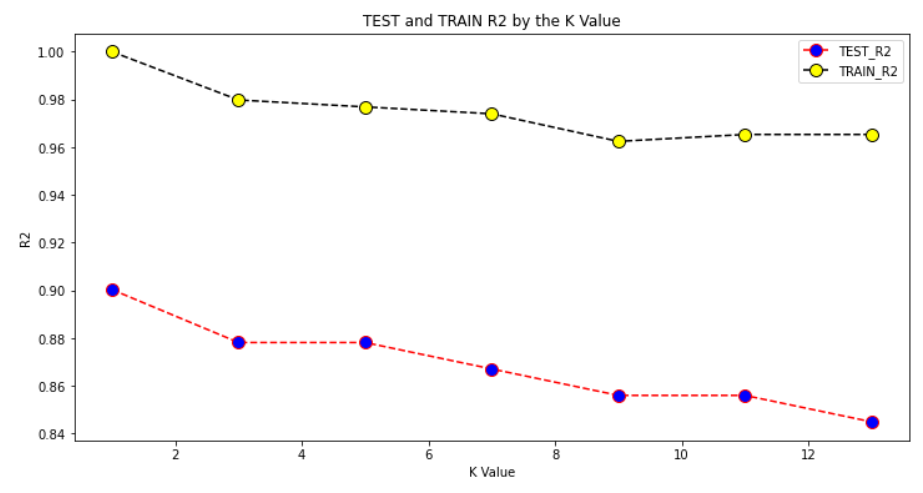
The Linear Regression model made a high R2 score and low MSE scores. However, when I apply the Test dataset in this model R2 score was significantly reduced. Also, we can see MSE become larger. This result indicates that the Linear Regression model overfitted from the train data and has less explanation

**KNN:**

In the KNN model with K=1, this implies that a point of decision boundary just looking at one the nearest neighbor to make the boundary. We can think of it as a different way that there is one criterium to divide the data. Therefore, I can guess the KNN model with K=1 is the best among the KNN model with other values. This is because our goal is making a model distinguish value ‘2’ and ‘3’.



In the graph, as K-value is going up, MSE is also going up. It means that the KNN model is becoming inaccurate as the k values are going up. Also, you can see the Test\_MSE values are greater than the Train\_MSE. This is because the model is made by test data. The model has some tendency to be overfitted by the test data.



Likewise, this indicates that the KNN model is becoming inaccurate as the k values are going up. We can also see the Test\_R2 values is lower than Train R2 Value with the same reason with MSE.

Compare the Linear Regression model and the KNN model.



I print out three models KNN(k=1), KNN(k=15), and Linear Regression.

* KNN(K=1): Best performance in KNN model.
* KNN(K=15): The worst performance in KNN model
* Linear Regression model

MSE:

KNN(K=1) TEST < KNN(K=15) TEST < Linear Regression

R2:

KNN(K=1) TEST > KNN(K=15) TEST > Linear Regression

As I mentioned, since the model should distinguish 2 variables, KNN(K=1) shows better performance than KNN(K=15) both test and train data. And even KNN(k=15) shows better performance than the Linear regression.

Reference

https://expertsteaching.com/evaluating-regression-models-with-python-scikit-learn/#MEAN\_SQUARED\_ERROR