BBL536E Homework 2 Report

704191018
Yüksel Kapan

June 7, 2020

1 Problem 1

I started the problem by examining the data and creating a data frame from the *fitbit.csv* file and as described by the problem, I dropped the **Date** column from.

After preparing the data, I extracted X and y from the data frame. Then, I utilized the built in *mutual_info_regression* function to calculate the individual contributions of each feature.

The top 4 features and their scores are as follows

1	Activity Calories	1.14535833
2	Minutes Fairly Active	0.37669184
3	Steps	0.34209196
4	Distance	0.30415666

Similarly, I did the same for the F score. Here are the result:

1	Activity Calories	24.71861361
2	Steps	11.30606088
3	Distance	11.11009364
4	Minutes Fairly Active	8.52961981

Finally, here are the result for χ^2:

1	Steps	16803.1323
2	Activity Calories	2010.95841
3	Floors	977.799808
4	Minutes Sedentary	567.536089

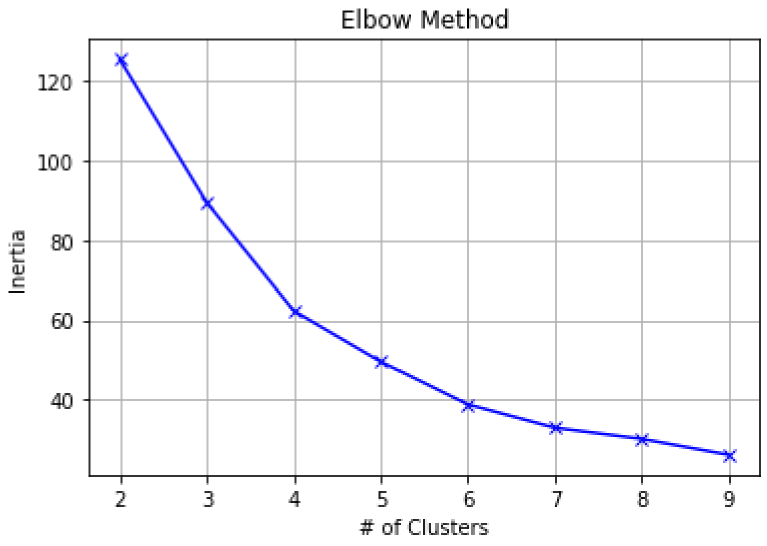
For the second part of the problem, I used the built in RFE class for Recursive Feature Elimination. Here are the top 4 features for RFE:

1	Distance
2	Minutes Lightly Active
3	Minutes Fairly Active
4	Minutes Very Active

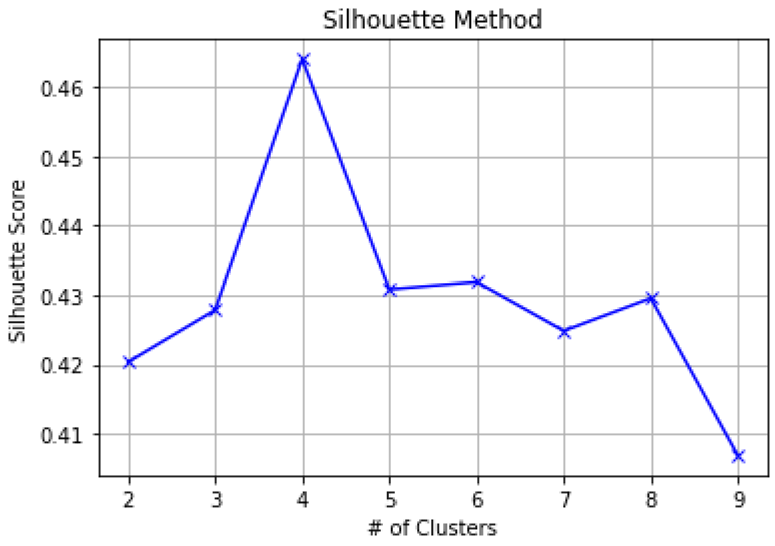
2 Problem 2

I began solving the problem with importing the data from *customer.csv* to a pandas data frame and drop the **ID** column from the data frame.

Then, I extracted X from the data frame for clustering and scale it using *StandardScaler*. With the data ready, I implement a for loop to go through all integers in $[2, 9]$ as the **k** value. For each **k** value, I store the *inertia_* attribute in an array. Here is the graph below.



Similarly, the silhouette graph is calculated with the sklearn built-in *silhouette_score* function on each **k** value. Here is the resulting graph


Silhouette Method
Silhouette Score
0.46
0.45
0.44
0.43
0.42
0.41
2
3
4
5
6
7
8
9
# of Clusters

3 Problem 3

Similar to the previous problems, I first imported *WAFn-UseC-Telco-Customer-Churn.csv* into a pandas data frame.

In this problem, I had to do some preprocessing before applying all the methods. For preprocessing, I converted all categorical values into column vectors using *et_dummies* method and dropped all rows that had missing values or empty strings.

Since I will be trying the predict the churn on many different models, I thought a function would make things easier. Therefore, I wrote a function called *calculate_avg_accuracy* and it calculates the average accuracy for a given model. (The average score is the average of accuracy scores of each cross validation section)

Here are my results:

model	train	test
Logistic	0.805567	0.80560
DecisionTree	0.997866	0.72426
SVC	0.627517	0.62388
KNN	0.829707	0.76407
MLPClassifier	0.788893	0.78640