University of Piraeus

DEPARTMENT OF DIGITAL SYSTEMS
MSc INFORMATION SYSTEMS AND SERVICES

BIG DATA AND ANALYTICS

PRACTICAL MACHINE LEARNING

FIRST NAME: GEORGIOS

LAST NAME: PANAGIOTAKOPOULOS

REGISTRATION No: ME2030

SUPERVISOR: TELELIS ORESTIS

PIRAEUS, JUNE 2021

Table of Contents

Introduction

The application creates a dataset which represents in columns all the products of the set. In total 17x7 the number of different products is 17. The dataset therefore created includes 17 columns (for products) and an initial column, which corresponds to the ID of the corresponding basket. In each row, the columns of the products that are not included in the corresponding basket, have the value 0. The other columns have a value that corresponds to the value of the product in the basket. All rows of the dataset have 7 columns with value> 0.

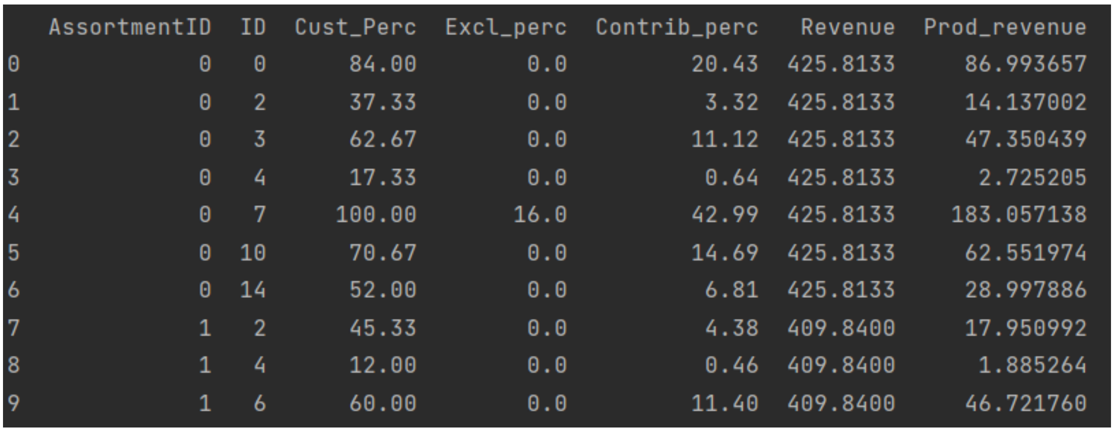
This creates a dataset that represents baskets of 7 products in any combination.

The program aims to make forecasts for future revenue from a particular product. The user enters the product code as a parameter in the application.

1. Exploratory data analysis

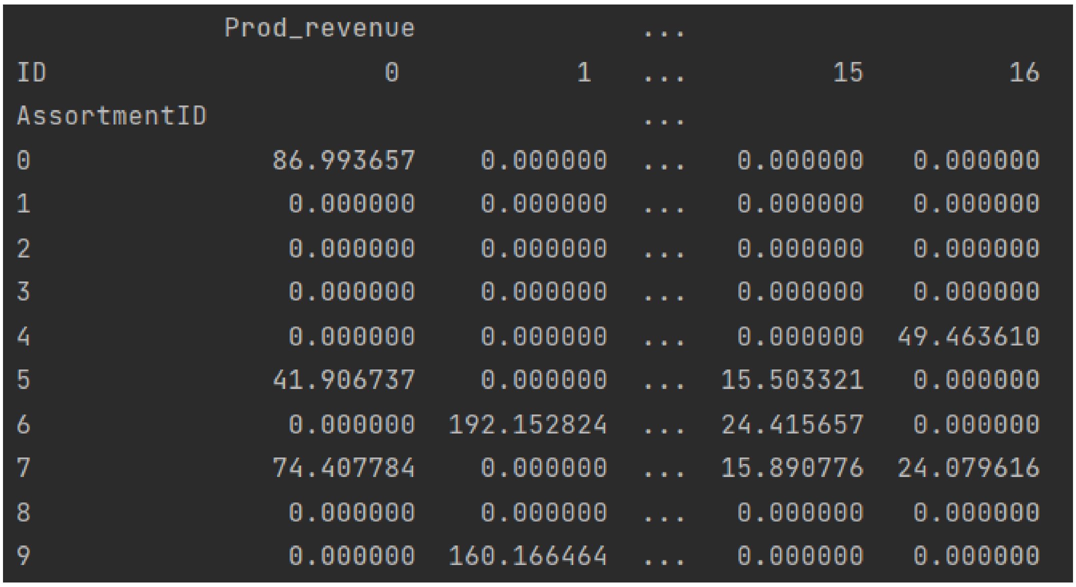
We created an auxiliary CSV file where we put the processed Dataframe with the data of each product in a line and brought it in a specific format. This form, as shown below, consists of the following columns:

- AssortmentID of the cart. There are as many lines in the file as there are products sold in the cart.
- ID which is the product code.
- Cust_Perc which is the percentage of customers who bought the product.
- Excl_perc which is the percentage of customers who exclusively bought the product.
- Contrib_perc which is the contribution percentage of the product in the revenue.
- Revenue which is the total revenue of the cart with the specific AssortmentID
- Prod_revenue which is the revenue corresponding to the product of the line.



```
   AssortmentID  ID  Cust_Perc  Excl_perc  Contrib_perc   Revenue  Prod_revenue
0             0   0      84.00        0.0         20.43  425.8133     86.993657
1             0   2      37.33        0.0          3.32  425.8133     14.137002
2             0   3      62.67        0.0         11.12  425.8133     47.350439
3             0   4      17.33        0.0          0.64  425.8133      2.725205
4             0   7     100.00       16.0         42.99  425.8133    183.057138
5             0  10      70.67        0.0         14.69  425.8133     62.551974
6             0  14      52.00        0.0          6.81  425.8133     28.997886
7             1   2      45.33        0.0          4.38  409.8400     17.950992
8             1   4      12.00        0.0          0.46  409.8400      1.885264
9             1   6      60.00        0.0         11.40  409.8400     46.721760
```

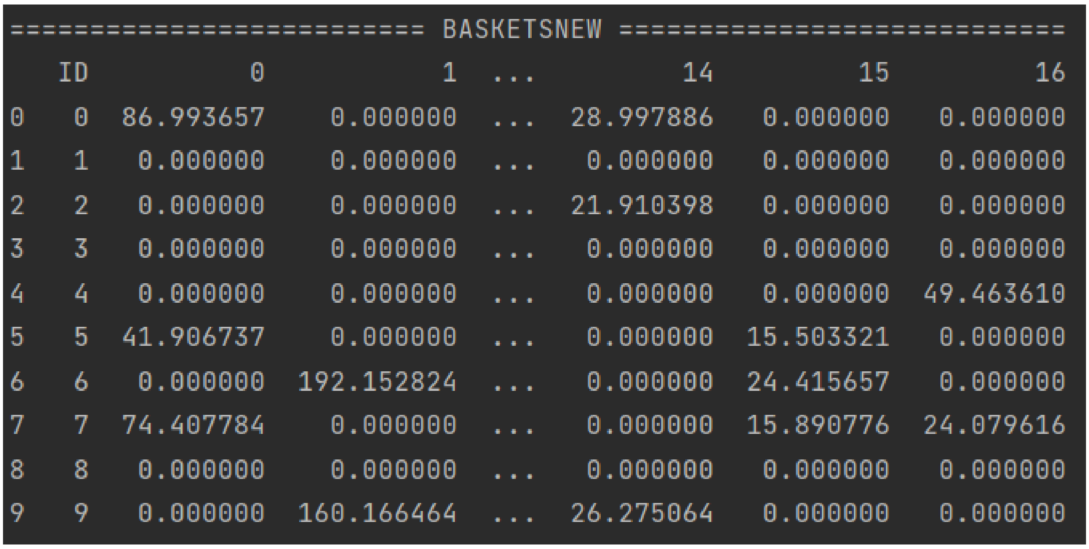
The next step was to apply groupby command to the dataframe documents by "AssortmentID" and product "ID". Grouping adds to the "Prod_revenue" column, as shown in the following figure.



```
             Prod_revenue                ...
ID                      0           1    ...         15         16
AssortmentID                             ...
0               86.993657    0.000000    ...   0.000000   0.000000
1                0.000000    0.000000    ...   0.000000   0.000000
2                0.000000    0.000000    ...   0.000000   0.000000
3                0.000000    0.000000    ...   0.000000   0.000000
4                0.000000    0.000000    ...   0.000000  49.463610
5               41.906737    0.000000    ...  15.503321   0.000000
6                0.000000  192.152824    ...  24.415657   0.000000
7               74.407784    0.000000    ...  15.890776  24.079616
8                0.000000    0.000000    ...   0.000000   0.000000
9                0.000000  160.166464    ...   0.000000   0.000000
```

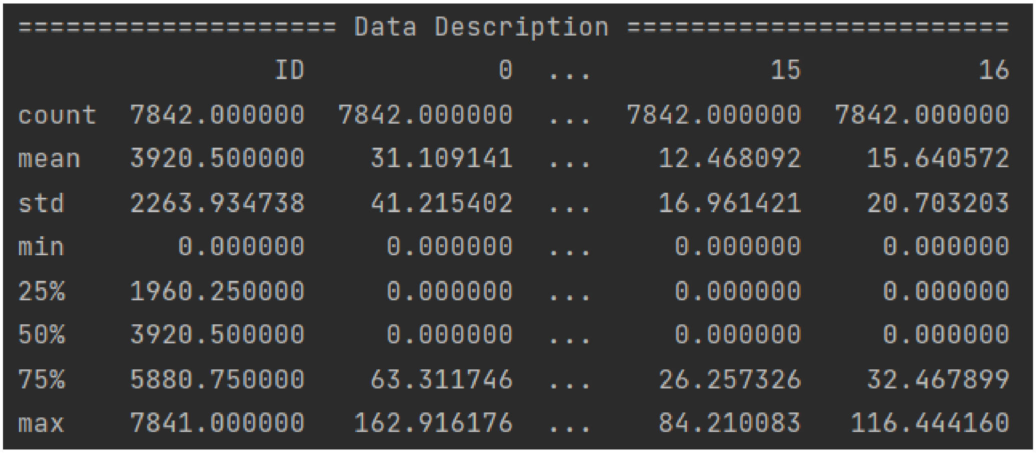
We then create a utility CSV file so that we can then exclude certain lines, which cause problems in creating a pandas dataframe.

Editing the CSV file results in creating a dataframe, which has the format shown in the following image. The first column contains the assortment ID and in the columns (17 or 20, depending on the dataset we use each time), for the products that participate in the basket, the income from them is mentioned. The other products are displayed with a value of 0.



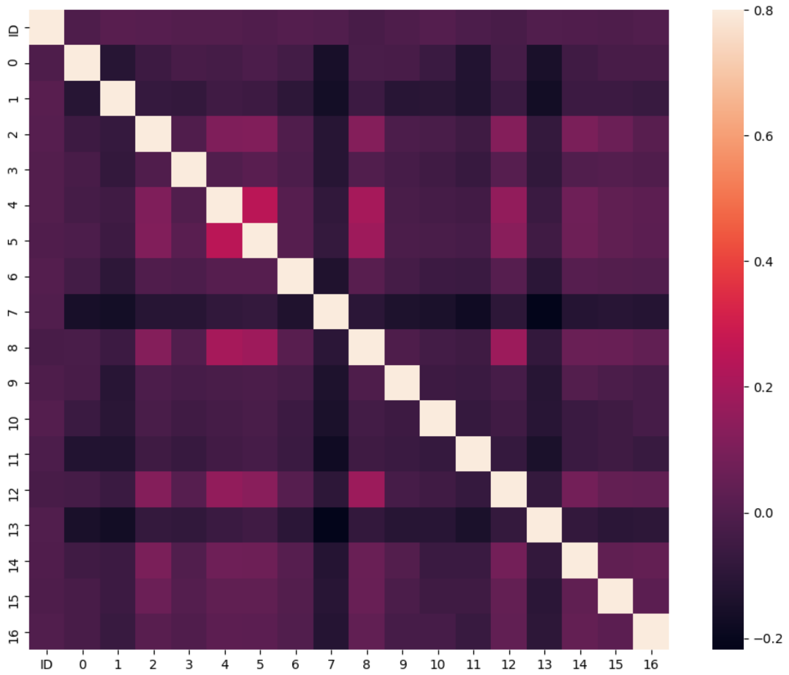
```
========================== BASKETSNEW ============================
   ID          0           1  ...         14         15         16
0   0  86.993657    0.000000  ...  28.997886   0.000000   0.000000
1   1   0.000000    0.000000  ...   0.000000   0.000000   0.000000
2   2   0.000000    0.000000  ...  21.910398   0.000000   0.000000
3   3   0.000000    0.000000  ...   0.000000   0.000000   0.000000
4   4   0.000000    0.000000  ...   0.000000   0.000000  49.463610
5   5  41.906737    0.000000  ...   0.000000  15.503321   0.000000
6   6   0.000000  192.152824  ...   0.000000  24.415657   0.000000
7   7  74.407784    0.000000  ...   0.000000  15.890776  24.079616
8   8   0.000000    0.000000  ...   0.000000   0.000000   0.000000
9   9   0.000000  160.166464  ...  26.275064   0.000000   0.000000
```

As observed by the .describe() command we have 7842 unique baskets

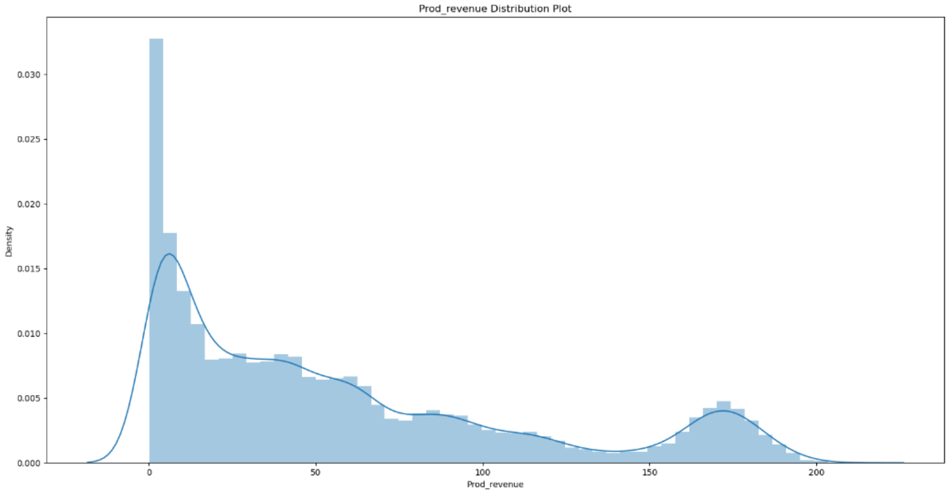


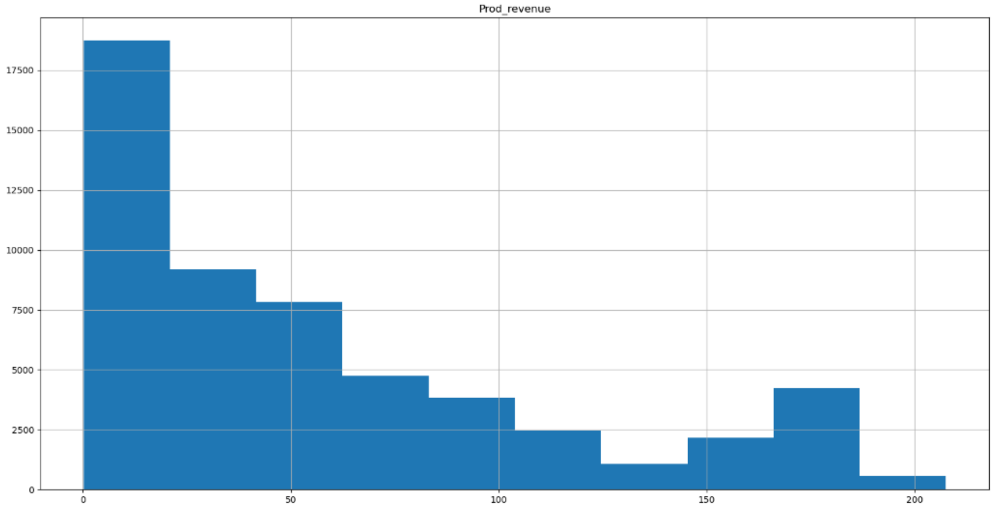
```
==================== Data Description ========================
                ID            0  ...           15           16
count  7842.000000  7842.000000  ...  7842.000000  7842.000000
mean   3920.500000    31.109141  ...    12.468092    15.640572
std    2263.934738    41.215402  ...    16.961421    20.703203
min       0.000000     0.000000  ...     0.000000     0.000000
25%    1960.250000     0.000000  ...     0.000000     0.000000
50%    3920.500000     0.000000  ...     0.000000     0.000000
75%    5880.750000    63.311746  ...    26.257326    32.467899
max    7841.000000   162.916176  ...    84.210083   116.444160
```

Calculate the correlation of the dataframe columns and create the heatmap of the following figure.



The following histogram shows the distribution of the number of products per revenue.


Prod_revenue Distribution Plot
Density
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
50
100
150
200
Prod_revenue


Prod_revenue
17500
15000
12500
10000
7500
5000
2500
0
0
50
100
150
200

2. Exercise A, Sub-questions 1st and 2nd

A pandas dataframe is used to manage the dataset. In order for the machine learning models to be executed, the column of the specific product is deleted from the dataset and the dataframe named 'X' is created. The column removed from the dataset creates a new dataframe, which contains the revenue values for that product in all the baskets of the original dataset (dataframe 'y'). These two dataframes are used to create train and test datasets. 30% is used as test data, while 70% as training data.

2.1. Results of Sub-question 1st

The following are results using the 17x7.txt dataset for product code 12.

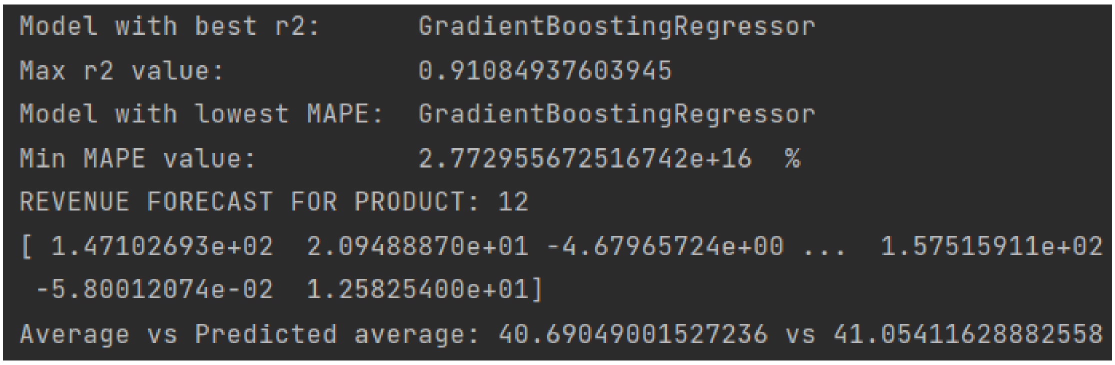
For Exercise A, sub-question 1st, the forecasts are contained in the list entitled "REVENUE FORECAST FOR PRODUCT:"

At the same time, the models that were applied were the following: Gradient Boosting Regressor, Random Forest Regressor, Linear Regression, Lasso, Ridge, ElasticNet and Adaboost Regressor, where the R^2 was calculated separately for each model, as well as the Mean Absolute Percentage Error. The following table shows the relevant results from the measurements of the specific product.

Model	R^2	Mean Absolute Percentage Error
Gradient Boosting	0.91084937603945	2.772955672516742e^{+16} %
Random Forest Regressor	0.6778576561467997	4.919726013568352e^{+16} %
Linear Regression	0.7541885232975835	5.177470654472124e^{+16} %
Lasso	0.7541550217929402	5.177168192169683e^{+16} %
Elastic Net	0.7540950000926651	5.17720390375646e^{+16} %
Ridge	0.7541881479423848	5.177469795893936e^{+16} %
Adaboost Regressor	0.20214510792201523	1.0716412617901171e^{+17}%

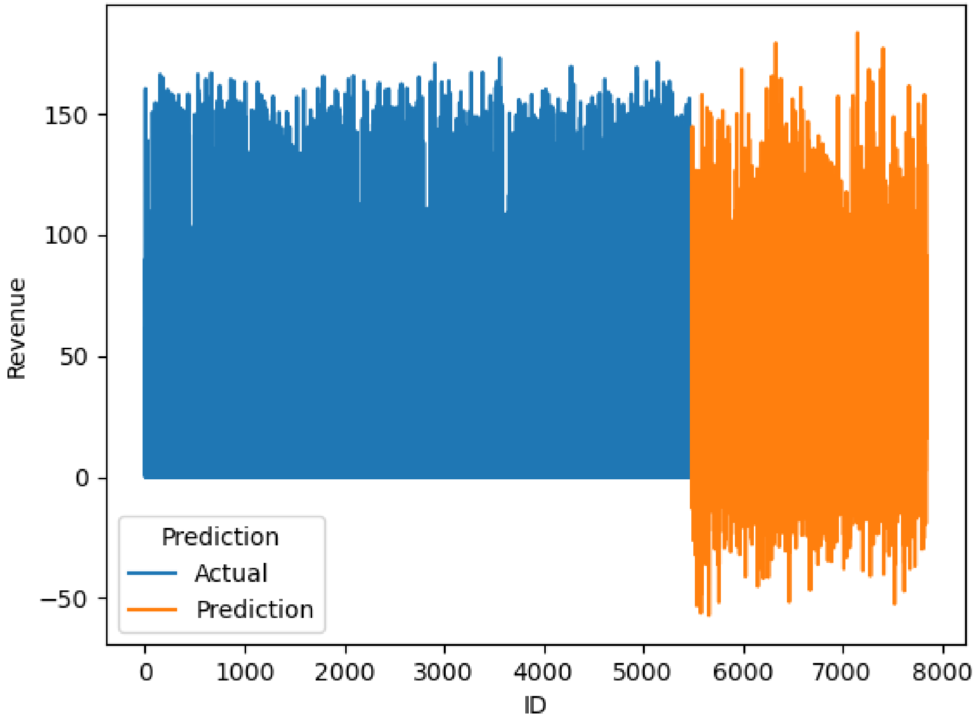
From the table above we observe that the GradientBoosting Regressor shows the best statistical results, i.e. the highest R^2 and the lowest Mean Absolute Percentage Error.

For Exercise A, sub-question 2nd, the calculation of the average value of actual income in relation to the average value of provisions is shown in the last line of the results entitled: “Average vs Predicted average:



```
Model with best r2:       GradientBoostingRegressor
Max r2 value:             0.91084937603945
Model with lowest MAPE:   GradientBoostingRegressor
Min MAPE value:           2.772955672516742e+16  %
REVENUE FORECAST FOR PRODUCT: 12
[ 1.47102693e+02  2.09488870e+01 -4.67965724e+00 ...  1.57515911e+02
 -5.80012074e-02  1.25825400e+01]
Average vs Predicted average: 40.69049001527236 vs 41.05411628882558
```

The line chart below shows the forecast (in orange) in relation to the actual data of the set using the model with the best statistical results, ie the Gradient Boosting Regressor.

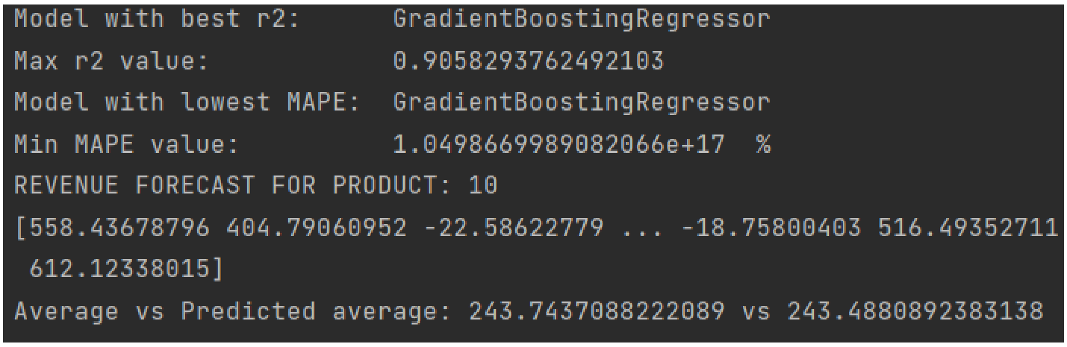


The model that achieved the best R^2 was used to display the prediction results, as the MAPE (Mean Absolute Percentage Error) size was low for all models. Based on the results, the forecasts follow upward trends for the most commercial products, while the model also produces negative prices (strong downward forecast) for products that move at levels close to zero.

2.2. Results of the sub-question 2nd

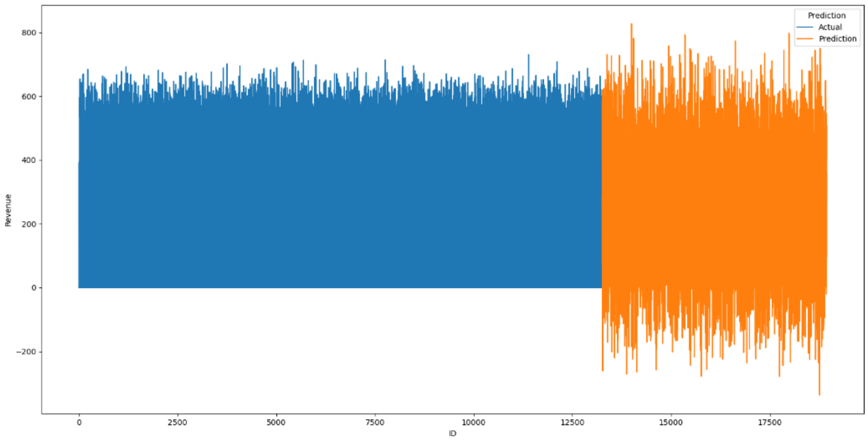
The following are results using the 20x10.txt dataset for product code 10

It turned out that among the 7 models, the Gradient Boosting Regressor model achieved the best R^2.



```
Model with best r2:      GradientBoostingRegressor
Max r2 value:            0.9058293762492103
Model with lowest MAPE:  GradientBoostingRegressor
Min MAPE value:          1.0498669989082066e+17  %
REVENUE FORECAST FOR PRODUCT: 10
[558.43678796 404.79060952 -22.58622779 ... -18.75800403 516.49352711
 612.12338015]
Average vs Predicted average: 243.7437088222089 vs 243.4880892383138
```

The model that achieved the best R^2 was used to represent the prediction results. Here again the MAPE (Mean Average Percentage Error) size was low for all models. Based on the results, the forecasts follow downward trends for the most commercial products, while the model also produces negative prices (strong downward forecast) for products that move at levels close to zero.



3. Exercise B - Sub-questions 1st and 2nd

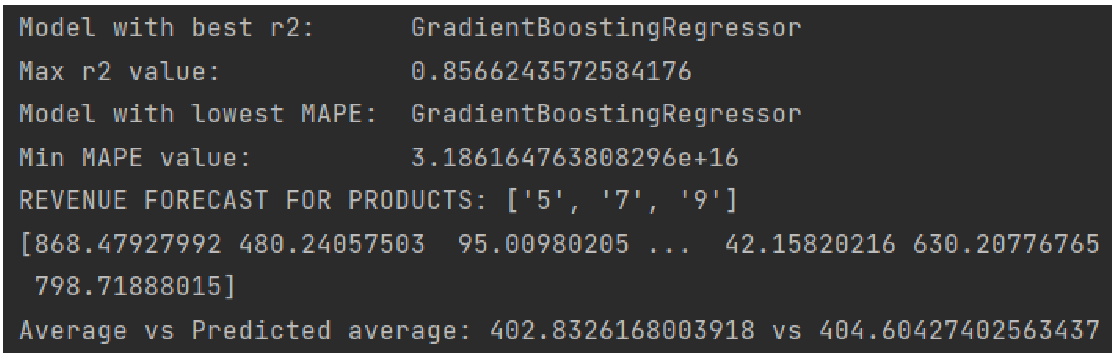
For Exercise B, the program accepts from the user a list of products, in order to create the assortment for which the revenue forecast will be made. The methodology is similar. The difference is that the columns containing the assortment revenue are removed from the dataset, and these columns create a new dataframe. In order for the assortment revenue items to create a one-dimensional NumPy array, the revenue per line is summed to give the total assortment revenue of each line. The result is a one-dimensional array, which then feeds the models.

3.1. Indicative execution in the 20x10.txt dataset

For the dataset 20x10.txt the assortment of products with IDs 5, 7 and 9 was created. The results are shown in the following table.

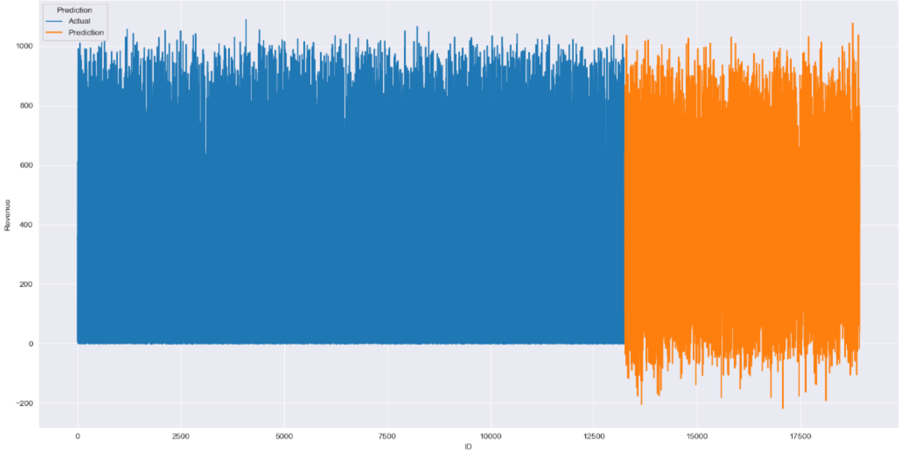
Model	R^2	Mean Absolute Percentage Error
Gradient Boosting	0.8566243572584176	3.186164763808296e^{+16}
Random Forest Regressor	0.6683649147109231	5.393765190358827e^{+16}
Linear Regression	0.7131920201547504	5.964229502337462e^{+16}
Lasso	0.7131938165085203	5.964544843204635e^{+16}
Elastic Net	0.7131976456156948	5.965320837267382e+16
Ridge	0.7131920852742855	5.964235848753719e+16
Adaboost Regressor	0.4823975634742198	1.3565569158047192e+17

In the following figure, for Exercise B, sub-question 2nd , for a specific subset of l products from m, the calculation of the average value of actual income in relation to the average value of forecasts is shown in the last line of the results entitled: “Average vs Predicted average:”.



```
Model with best r2:       GradientBoostingRegressor
Max r2 value:             0.856624357258417б
Model with lowest MAPE:   GradientBoostingRegressor
Min MAPE value:           3.186164763808296e+16
REVENUE FORECAST FOR PRODUCTS: ['5', '7', '9']
[868.47927992 480.24057503  95.00980205 ...  42.15820216 630.20776765
 798.71888015]
Average vs Predicted average: 402.8326168003918 vs 404.60427402563437
```

The model that achieved the best R^2 was used to represent the prediction results. Here again the MAPE (Mean Average Percentage Error) size was low for all models.



Based on the results, the forecasts follow upward trends for the most commercial products, while the model also produces negative prices (strong downward forecast) for products that move at levels close to zero.