EARIN
Exercise 4
Variant 1

Oleg Kim 01155526@pw.edu.pl
Seniv Volodymyr 01156240@pw.edu.pl

02.04.2023

Task description:

Write a program that predicts house prices based on the provided dataset(variant1).

Use at least two different methods (e.g. Linear Regression, Logistic Regression, SVM, ...) and compere their performance. You should also deliver report that includes the comparison of metrics for each used method and explanation of which method consider the best. The solutions you deliver will be evaluated based on the results of the chosen models and the correctness of your research methods.

Launch instruction:

Firstly, place the data set file with the program file in one folder. Secondly, you must install the appropriate packages with pip install command:

1. scikit-learn
2. pandas
3. seaborn
4. matplotlib

Then you when you launched the program you must choose what algorithm you want to use.

Theoretical background:

Linear regression:

In simple linear regression, there is only one independent variable, while in multiple linear regression, there are multiple independent variables. The objective of linear regression is to find the best fit line that represents the relationship between the independent and dependent variables.

The best fit line is determined by minimizing the sum of squared errors between the observed values of the dependent variable and the predicted values based on the independent variable(s). This is known as the least-squares method.

The linear regression model is represented as:

$Y = β0 + β1X1 + β2X2 + … + βpXp + ε$

where Y is the dependent variable, X1, X2, …, Xp are the independent variables, β0, β1, β2, …, βp are the coefficients, and ε is the error term.

The coefficients are estimated using the method of ordinary least squares (OLS), which involves minimizing the sum of squared errors. Once the coefficients are estimated, the linear regression model can be used to predict the value of the dependent variable for new values of the independent variables.

Random Forest:

The Random Forest algorithm is based on the concept of decision trees. A decision tree is a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. However, decision trees are prone to overfitting, which means that they may capture the noise in the training data and fail to generalize well on new data.

Random Forest overcomes the limitations of decision trees by using a technique called bagging. Bagging, short for bootstrap aggregating, is a method of combining the predictions of multiple models to reduce the variance of the individual models. In Random Forest, bagging is used to train many decision trees on different samples of the training data.

Each decision tree in Random Forest is trained on a random subset of the features and a random subset of the training examples. This randomness in the data and features helps to reduce overfitting and improve the generalization of the model.

To make a prediction on new data, the Random Forest algorithm takes the mean prediction of all the individual trees in the forest. This aggregation of predictions results in a more robust and accurate model compared to a single decision tree.

IQR (Interquartile Range):

IQR is a statistical method used to identify and remove outliers from a dataset. Outliers are extreme values that fall outside of the overall pattern of the dataset, which can significantly affect the results of statistical analysis.

The IQR method involves calculating the difference between the first quartile (25th percentile) and the third quartile (75th percentile) of a dataset. The IQR is then multiplied by a constant, typically 1.5, to determine the lower and upper bounds of the dataset. Any data points that fall outside of these bounds are considered outliers and can be removed from the dataset.

The formula for calculating the IQR is:

IQR = Q3 - Q1

Where Q1 is the first quartile (25th percentile) and Q3 is the third quartile (75th percentile) of the dataset.

The lower and upper bounds can then be calculated using the following formulas:

Lower Bound = Q1 - 1.5 * IQR Upper Bound = Q3 + 1.5 * IQR

Any data points that fall outside of these bounds can be considered outliers and can be removed from the dataset.

Solution:

At the beginning we have to read our data set file we the help of “pandas” library and function read_csv and store it in “db” variable. All this procedure is done inside the reading_file(filename), where “filename” is a name of our file.

Before building our models, we must preprocess the data in our data set:

1. remove rows with null values.
2. Convert the float variables (which should be int) into int
3. Apply the IQR method in order to remove outliners from our data set

All this procedure is done in prerpocessing_data(db) function, where “db” is the variable where our data set is stored.

Random forest-algorithm:

Firstly, we are defining variable which are “features” – independent one; and “price” – dependent one; and then we split this data into a training and testing sets with the help of “train_test_split” function from scikit-learn. The training set is used to train the model, while the testing set is used to evaluate the performance of the model.

Then, the random forest regressor is initialized with some hyperparameters like the number of decision trees “n_estimators” and maximum depth of the trees “max_depth”. The “RandomForestRegressor” function from scikit-learn is used for this purpose.

Next, the model is trained on the training data using the fit method of the random forest regressor object.

After the model is trained, the target variable “price” is predicted on the testing data using the predict method of the same object.

Finally, the performance of the model is evaluated using two metrics: mean squared error “mse” and r2 score “r2”.

The predicted and actual prices are also plotted to visualize the model's accuracy and the residual distribution is shown to check the assumption of normality.

Linear regression algorithm:

First, the function extracts the independent variables (i.e., features) and dependent variable (i.e., target) from the .csv file at the same time dropping the columns that are not required for the analysis (i.e., "id", "date", "price", "lat", "long", and "zipcode"). The features are stored in X, and the target variable is stored in y.

Next, the function splits the data into training and testing sets.

Then, the function fits a linear regression model to the training data and calculates the coefficients of the independent variables. The function then uses the model to predict the target variable for the testing data, and calculates the mean squared error and “r2” score of the.

Finally, the function plots a scatter plot of the actual vs predicted target values, and a histogram of the residuals.

Tests:

The output of both algorithms in form of graph.

Test will be performed by changing the test_size parameter, which will determine the amount of training and testing data.

Test 1 – testing set - 20%, training set - 80%

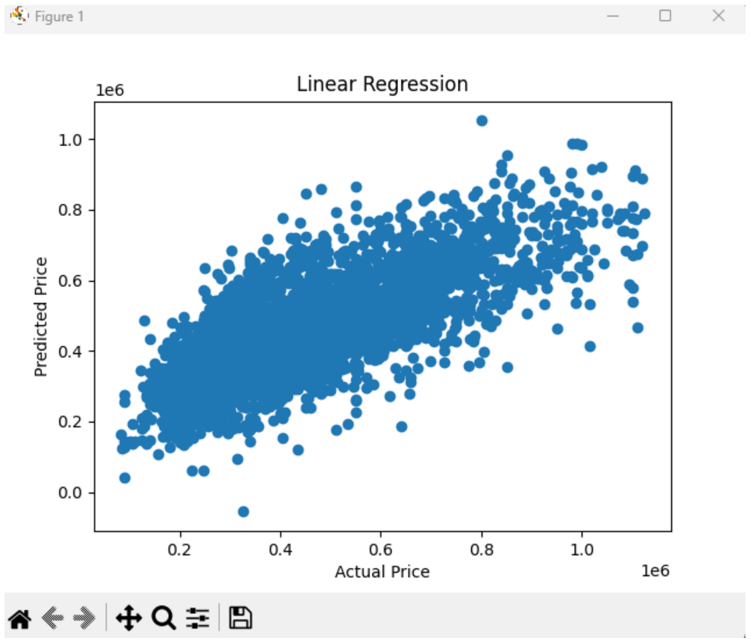
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=101)
```

Linear Regression:

Mean Squared Error: 17237292386.80978

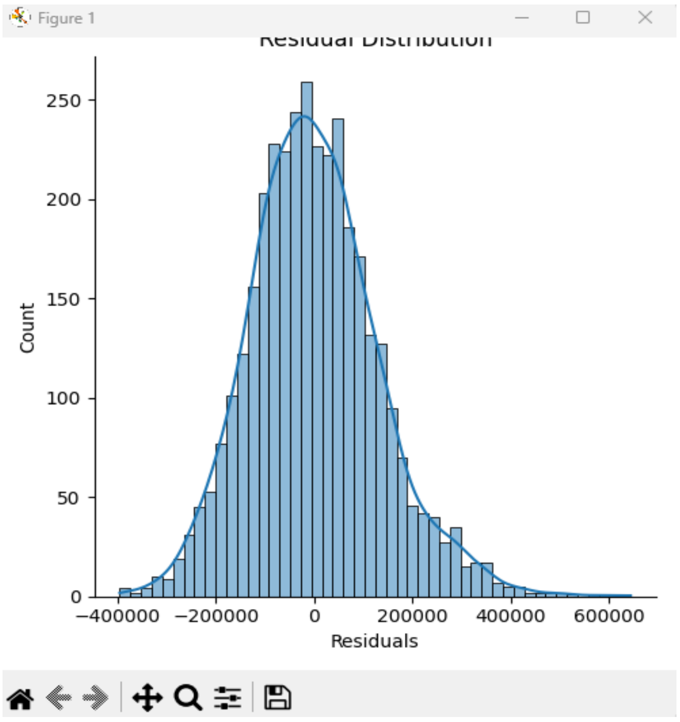
R2 Score: 0.5556051200610302

Scatter plot:



We can observe that majority if points in graph resemble a thick line going in diagonal direction, with minor amount of points located outside of it. It means that the prediction should be correct.

Distribution plot:



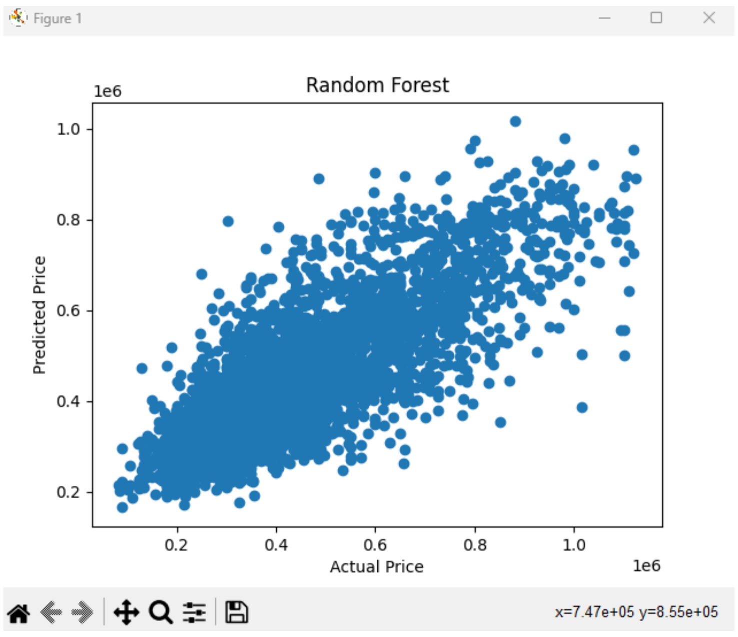
The graph appears to be in bell shape, normalized. The form appears to be correct, basically it means that model is well predicted.

Random Forest Regression:

Mean Squared Error: 14902350972.833467

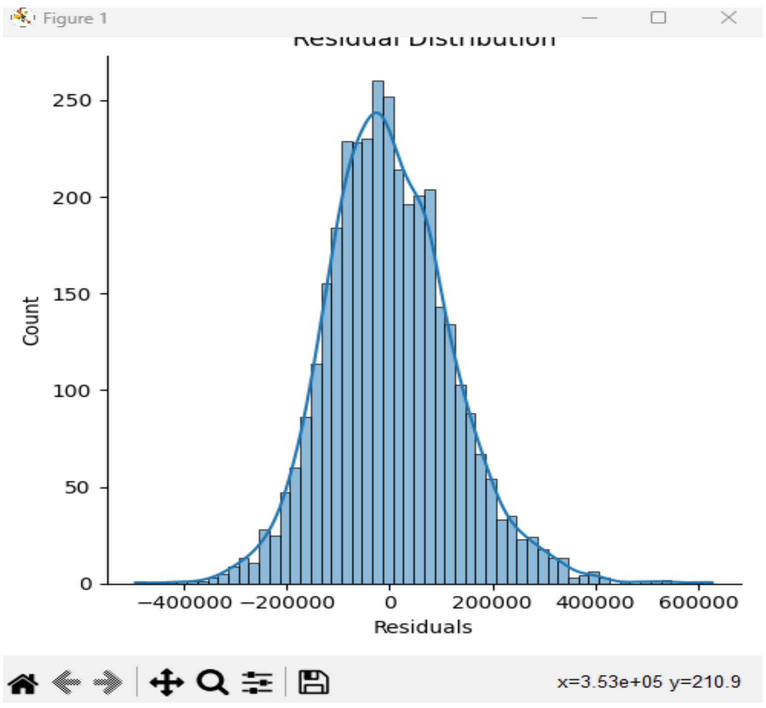
R2 Score: 0.6158022778305734

Scatter plot:



We can see similar picture to Linear model -> Scatter plot, we can observe that it is in thick line shape, with minor amount of points located outside of it. It means that the prediction should be correct.

Distribution plot:



This graph also appears to be in bell shape, normalized. The form appears to be correct.

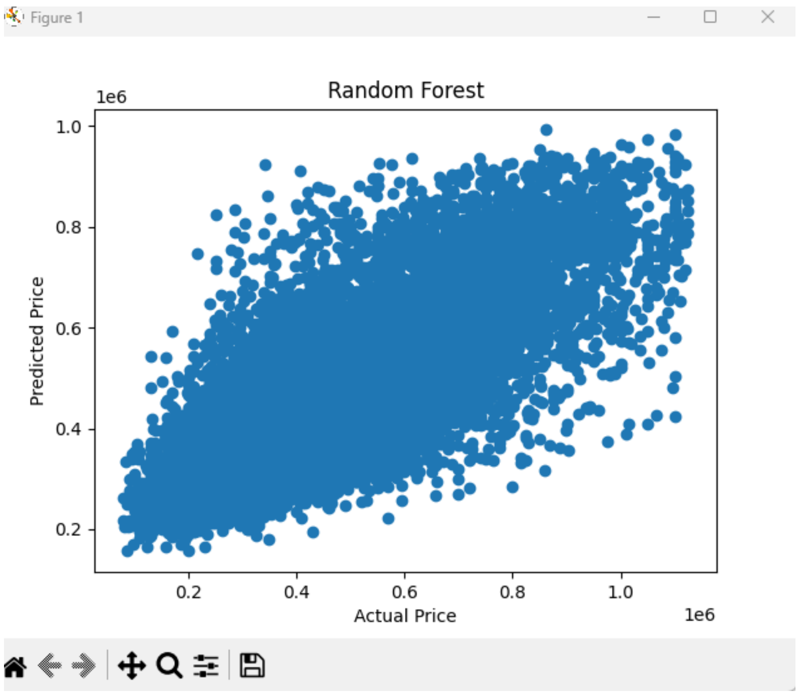
<u>Test 2 – testing set - 90%, training set - 10%</u>

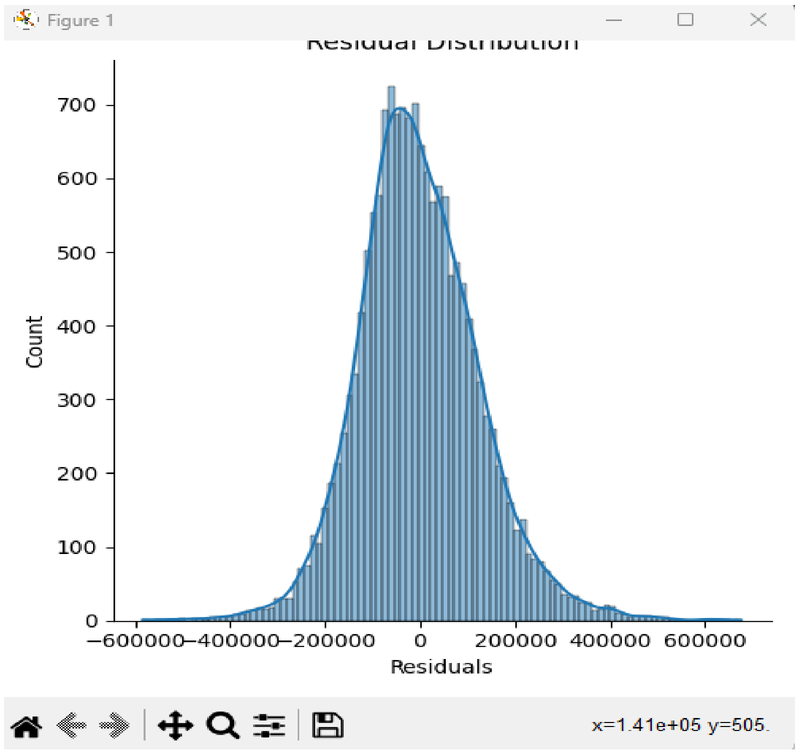
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.90, random_state=101)
```

Random Forest Regression:

Mean Squared Error: 16127340606.728706

R2 Score: 0.5849293216546005



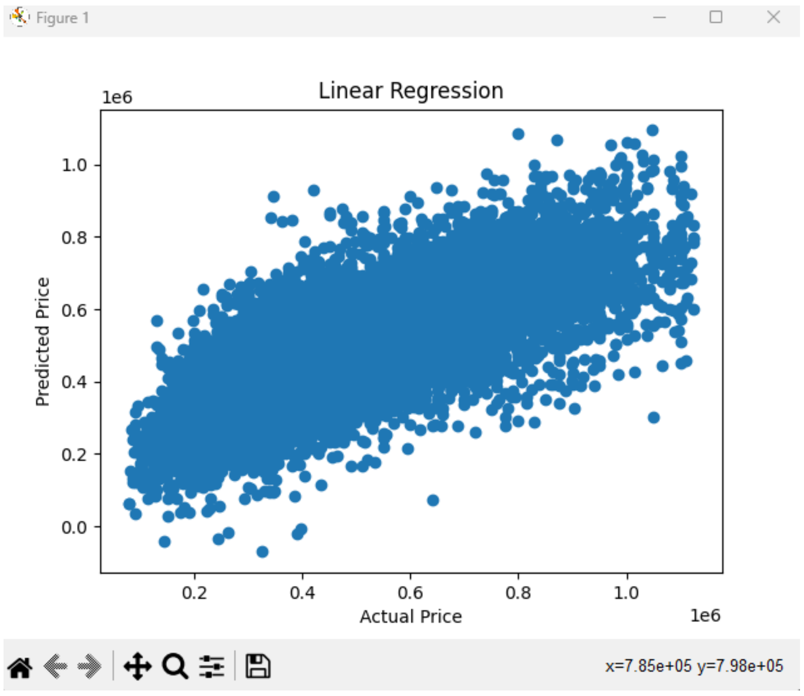


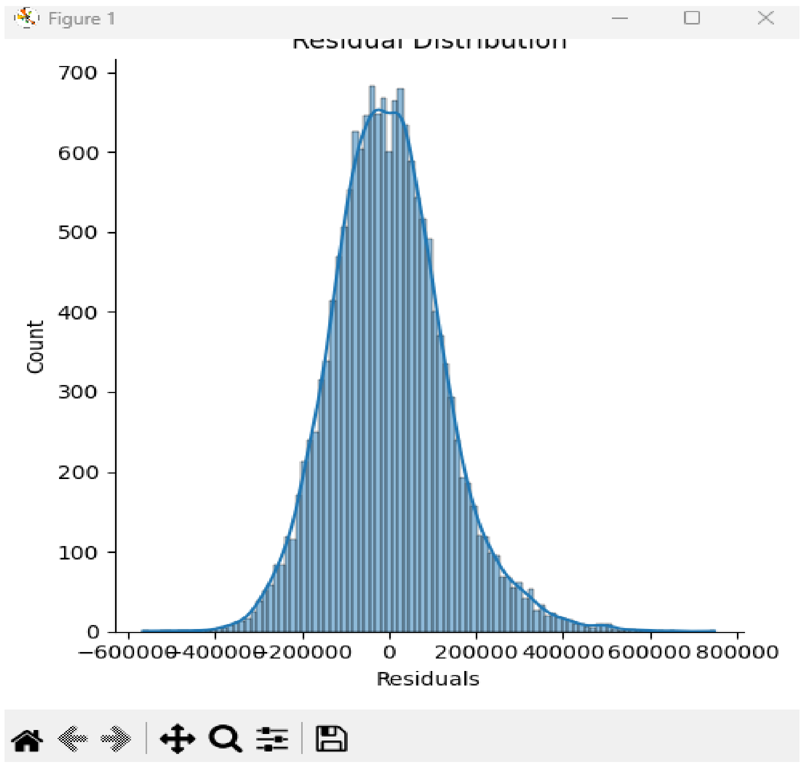
The error grew, forms appear to be similar

Linear Regression:

Mean Squared Error: 17208294045.63113

R2 Score: 0.5571087349821875





The error grew, forms appear to be similar

What can be improved:

Some minor improvements can be made, for example we can add some checkups if the variant1.csv file does not exist or in wrong format, or contains wrong data.

The path to .csv file can be made absolute. Apart from that I don't see what can be improved.

Conclusion:

The lower the mean squared error and the higher the r2 score, the better the model's performance. The predicted and actual prices are also plotted to visualize the model's accuracy and the residual distribution is shown to check the assumption of normality.

So basically, by comparing two algorithms we can see that random forest regression algorithm gives us better results. But the thing is that we can configure some parameters of Forest Regression algorithm such as 'n_estimators' to a high value, which theoretically should make it even more accurate, but computation time also grows, or we can rather set il low, then the calculations won't be so accurate. The practice shows that it is insignificant in our case If the parameter is set above 100, since we changed it multiple times, but results were similar(wasn't added as pictures).

We can see that it is better to use a smaller test size (e.g., 20-30%) to ensure that the model is trained on enough data and evaluated on a sufficient number of test samples to provide reliable estimates of model performance.

It is also known that Linear Regressor is more weak to outliers, while Random Forest is prone to them. But for the fairness of test the dataset used for both algorithms were the same, the outliers were removed form price and sqft_... columns which in our opinion are most significant in prediction. The insignificant for computation columns were removed.