MERCEDES-BENZ GREENER MANUFACTURING

ALY 6020 Predictive Analytics

CHIRAG SHAH & PRERAK SHAH

Northeastern University



Note:

Chirag Shah & Prerak Shah, College of Professional Studies, Northeastern University, Boston, MA 02115.

This report was created as a part of the project to gain hands-on experience in various machine learning algorithms using python as tool.

Chirag Shah & Prerak Shah is now a student at Department of Analytics, Northeastern University

Contact: [shah.chir@husky.neu.edu](mailto:shah.chir@husky.neu.edu) | shah.prer@husky.neu.edu

**INTRODUCTION**

Mercedes allows about 2000 patents on their cars per year. Their customers have a wide range of features and choices that allow them to choose their own Mercedes-Benz personalized car. In order to ensure the protection of each specific design model before the car hit the road, Mercedes created the rigorous test method so that any particular feature will not stop working. Yet time taken to check program is complicated and time-consuming with too many variations of functions. To support them minimize the bench time for the vehicles, Daimler (Mercedes) had provided analysts the market issue. A bench test is an operating test carried out on just an engine or other major part removed from a car.

We have worked to tackle the dimensionality and reduce the time that cars spend on the test bench. Our approach included various regressor models for more than 360 features both categorical and numerical. In this project we performed Linear, KNN, Decision Tree, Random Forest, Gradient Boosting and XGBoost Regressors, which lead improving R2 score for each. Recursive Feature Elimination (RFE), as its title suggests, recursively removes features, builds a model using the remaining attributes and calculates the accuracy of the model. RFE is able to work out a combination of attributes that contributes to the target variable's prediction. Predictive regression modeling is the task of approximating the mapping function (f) from input variables (X) to continuous output variable (y). The continuous output variable is a real value, such as an integer or a floating-point value. They are often quantities, such as quantities and sizes. As we are having the continuous target variable, so we will be using regressor machine learning algorithms over classification machine learning algorithms. In this experiment, we applied data cleaning and feature selection and extraction, along with model building and evaluation, resulting into Regression for different features w.r.t predict variable i.e. time in seconds.

**ANALYSIS**

**PART 1: LOADING DATASETS**

Using Kaggle as data source, we imported training and testing data of Mercedes-Benz Car each of 4208 observations with 365 features, which include both categorical and numerical data. We have used Python as our analysis tool and Jupyter Notebook as IDE for our model building. We used pandas data frame to import the dataset into the ide.

A close up of a keyboard

Description automatically generated

Figure 1: Loading Dataset into Jupyter Notebook using pandas

**PART 2: DATA CLEANING**

We cleaned the data by removing the outliers and unwanted features with redundant values. For the outliers, we plotted the distribution of features, which highlighted the unwanted columns in the dataset. We used matplotlib library to visualize each feature and their correlation with other features in the dataset.



Figure 2: Drop redundant and unwanted column features from the dataset

We mask the data, by filling the NA values with their respective mean values. The categorical data was encoded using Label Encoder and set dummies function in python. After finding the outliers through this graph, we removed them for better predictions.

A screenshot of a map

Description automatically generated

Figure 3: Found the outliers in the datasets

Setting the threshold as 150, to remove the outliers, affecting the model. The following method is used to set and remove the outliers in dataset.

A screenshot of a cell phone

Description automatically generated

Figure 4: Columns with threshold value greater than 150 are removed

**PART 3: EXPLORATORY ANALYSIS**

To know the data, describe () function in python was used to know the dataset. It is shown below:

A screenshot of a cell phone

Description automatically generated

Figure 5: Summary statistics for dataset

The train data has following information on features, i.e. 8 categorical variable, 1 float and 356 integer variables that are used to fit the regressor model, is shown below:

A picture containing bird

Description automatically generated

Figure 6: Information on training dataset

The target variable is a continuous variable time (in seconds) for each car benchmark, which is shown below:

A screenshot of a cell phone

Description automatically generated

Figure 7: Target variable statistics

Using matplotlib library, we plotted various distributions in terms of target variable and the features in the datasets which are shown below:

A screenshot of a cell phone

Description automatically generated

Figure 8: Distribution of Target variables

A screenshot of a cell phone

Description automatically generated

Figure 9: Distribution of y variable w.r.t various features

A close up of a map

Description automatically generated

Figure 10: Log transformation of y variable

We have also visualized the binary features in the dataset affecting the target variable. It is hown below:

A screenshot of a cell phone

Description automatically generated

Figure 11: Distribution of Binary data

**PART 4: MODEL CREATION AND EVALUATION**

**Label Encoding:**

Before splitting the dataset, we have encoded the eight categorical variables that we found during exploratory analysis. Label Encoding refers to the conversion of labels into a numeric form in order to convert them into a machine-readable form. Machine learning algorithms can then make a better decision on how to operate these labels. This is an important pre-processing step for the structured data set in supervised learning. The label encoding is done as follows:

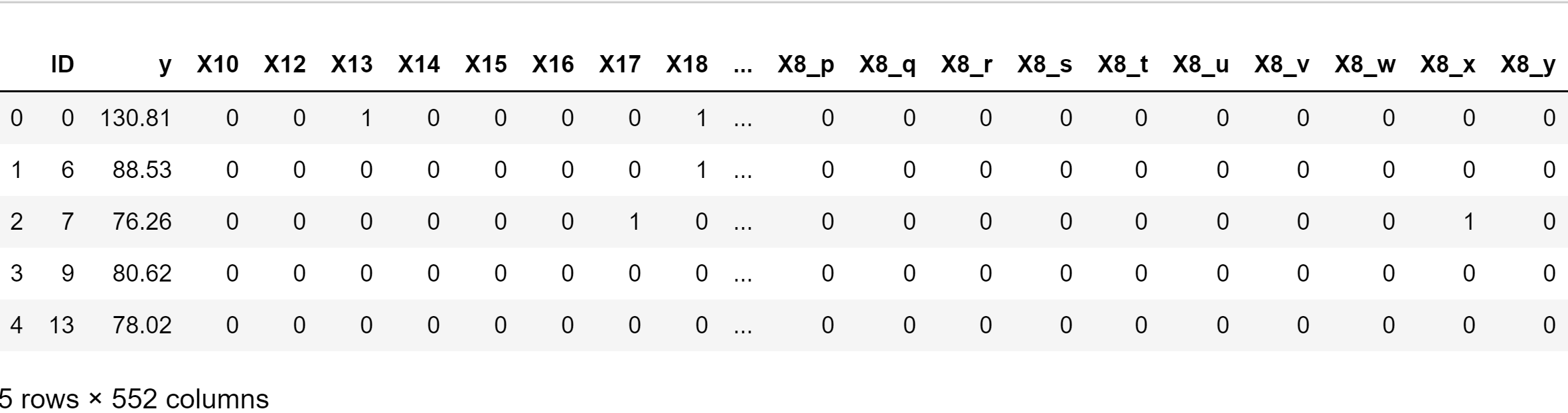


Figure 12: Encoding of categorical variables

**Splitting the dataset:**

After encoding the required variables and removing the variables which are not needed. We split the dataset as follows:



Figure 13: Splitting the Mercedes dataset

**Standardization:**

Standardization is the process of applying different variables to the same scale. This process allows you to compare scores between different variable types. Typically, to standardize variables, you calculate the mean and the standard deviation of the variable. Then, for each measured value of the variable, subtract the mean and divide it by the standard deviation.

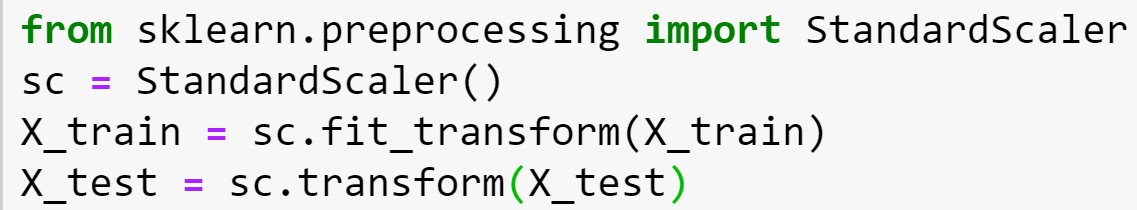


Figure 14: Standardization of the splitted data

**Model Evaluation:**

1. **Support Vector Regressor (SVR):** It uses the Support Vector Machine(SVM, a classification algorithm) algorithm to predict a continuous variable. Support Vector Regression tries to fit the best line within a predefined or threshold error value.

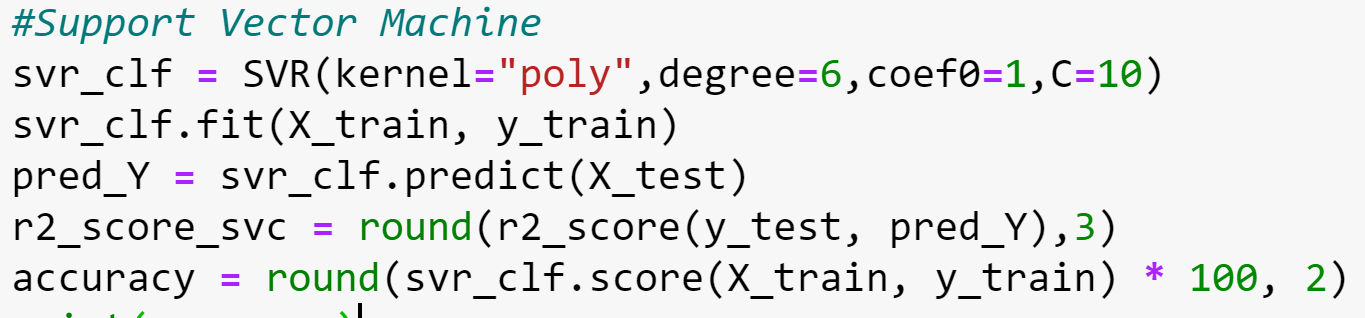


Figure 15: Support Vector Regressor model creation

1. **Random Forest Regressor:** The random forest model is very good at handling tabular data with numeric features or categorical features. Unlike linear models, random forests are capable of capturing non-linear interactions between the features and the target.

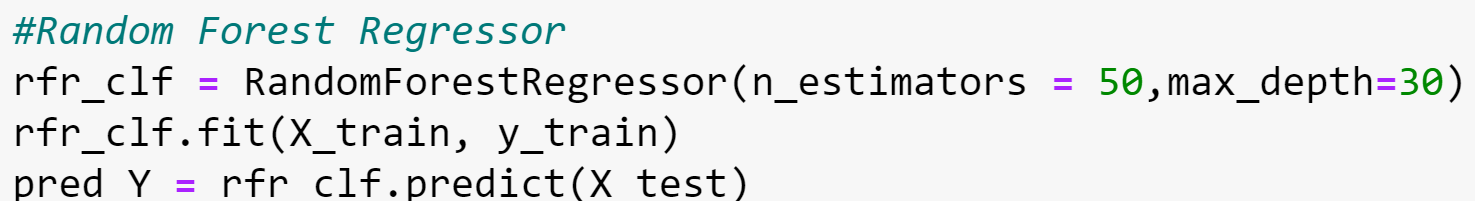


Figure 16: Random Forest Regressor model creation

1. **K Nearest Neighbor (KNN) Regressor :** When KNN is used for regression problems the prediction is based on the mean or the median of the K-most similar instances. Predictions are made for a new instance (x) by searching through the entire training set for the K most similar instances (the neighbors) and summarizing the output variable for those K instances.

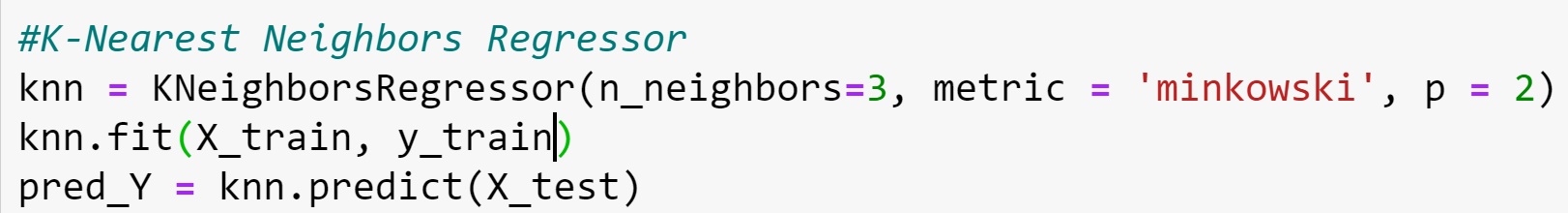


Figure 17: KNN regressor model creation

1. **Linear Regression:** Linear regression performs the task of predicting a dependent variable value (y) based on an independent variable (x). So, this regression technique finds a linear relationship between x (input) and y(output).

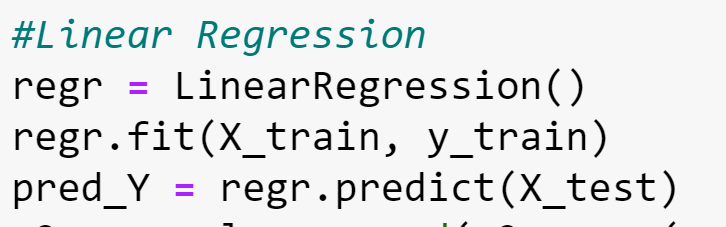


Figure 18: Linear regression model creation

1. **Decision Trees:** The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split.

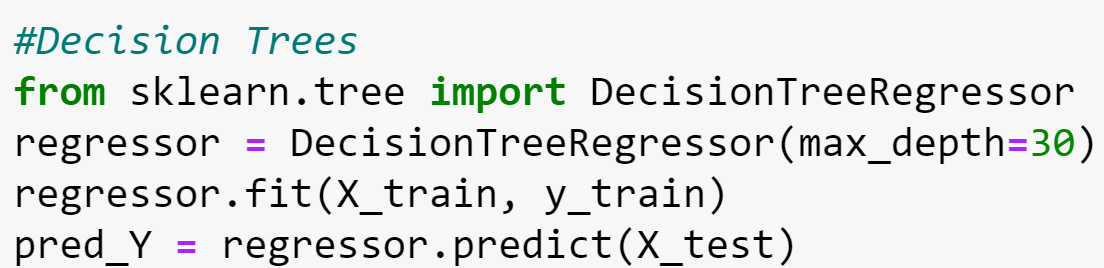


Figure 19: Decision trees model creation

1. **Extreme Gradient Boosting (XGBoost) :** XGBoost is a decision- Learning algorithm based on a gradient- framework. This is a special case of boosting where errors are minimized by a gradient descent algorithm, e.g. strategy consulting firms leverage by using case interviews to weed out less qualified candidates.

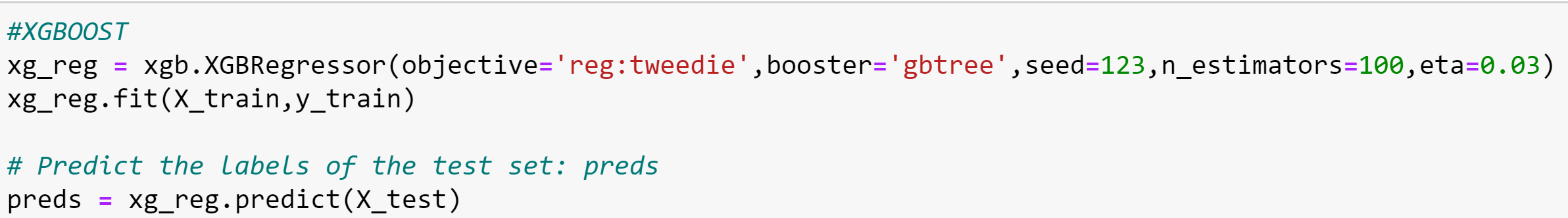


Figure 20: XGBoost model creation

**PART 5: IMPROVING MODEL PERFROMANCE**

* **Principal Component Analysis (PCA):** The main idea of the main component analysis (PCA) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the data set to the maximum extent possible. The same is done by converting the variables to a new set of variables, known as the main components (or simply, the PCs) and orthogonal, ordered in such a way that the retention of variation present in the original variables decreases as we move down in order.

1. **PCA with SVR:** The proposed method involved the extraction of multi-resolution acoustic features and robust PCA-based features for the efficient classification of three types of fish sounds. PCA-based SVM-associated approaches have been proposed as highly effective for the classification of samples.

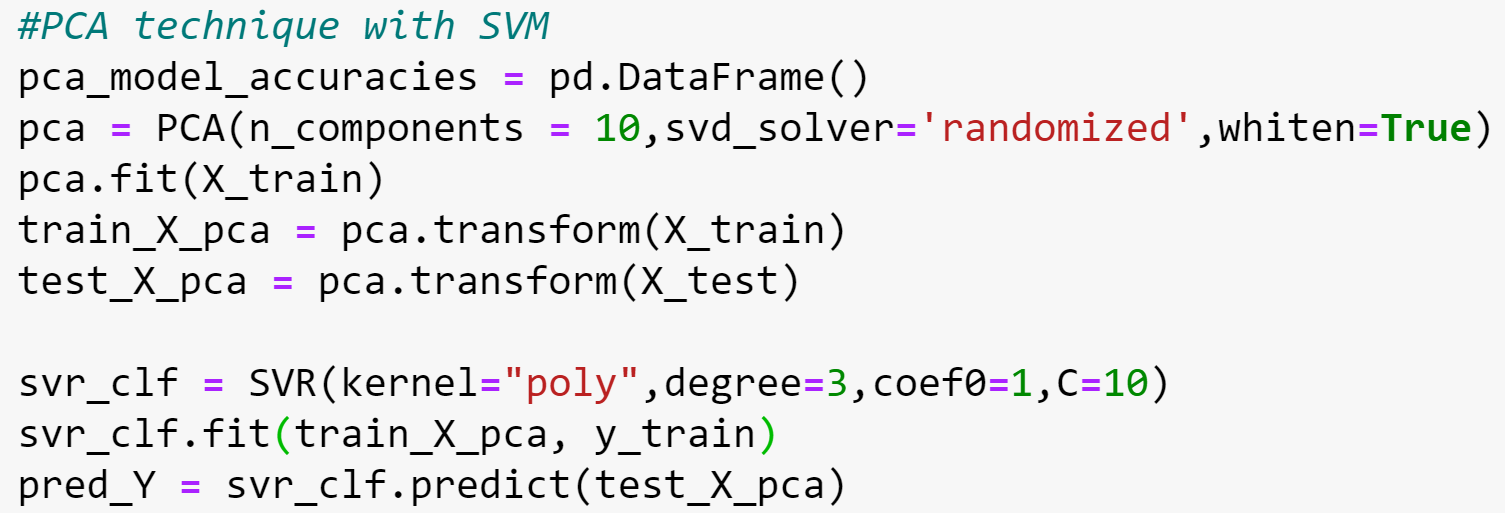


Figure 21: PCA with SVR model creation

1. **PCA with Random Forest Regressor:** When using PCA, we get rid of the two problems below that reduce the performance of Ranfom Forest: i) we reduce the number of features. ii) we're going to get rid of hill features. (All the hill features will end up in a single PCA component).

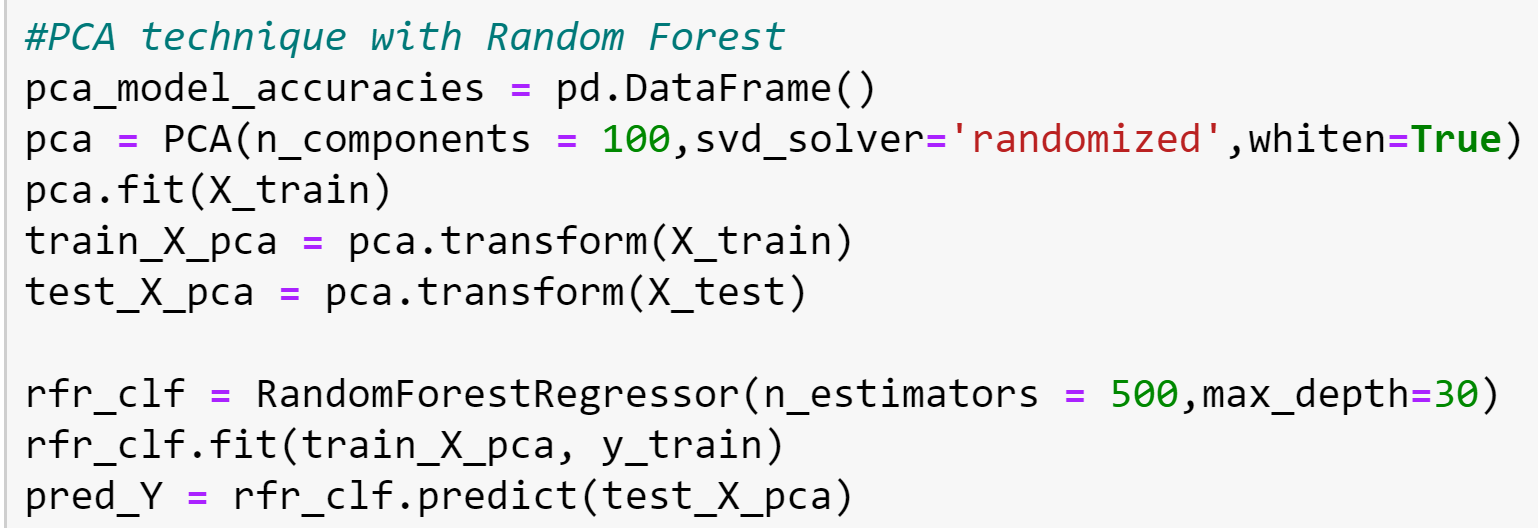


Figure 22: PCA with Random Forest Regressor model creation

1. **PCA with KNN Regressor:** In a PCA-KNN model, the historical data set as input is generated by a sliding window, transformed by PCA to principal components with rich-information, and then input to KNN for prediction.

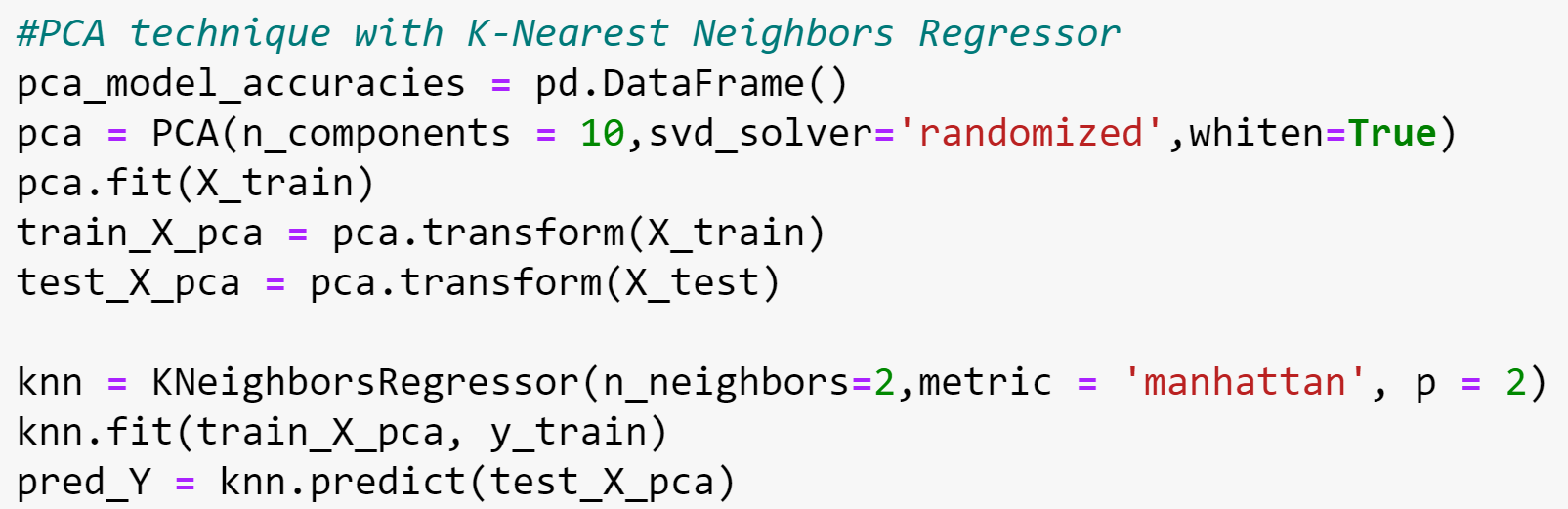


Figure 23: PCA with KNN Regressor model creation

1. **PCA with Decision Tree Regressor:** PCA helps trees to explicitly transform the data set to highlight directions that have the highest variance, which are often the directions that have the highest information gain while learning the decision tree. This makes the Decision Tree learn much faster and achieve high accuracy in the least number of decision layers.

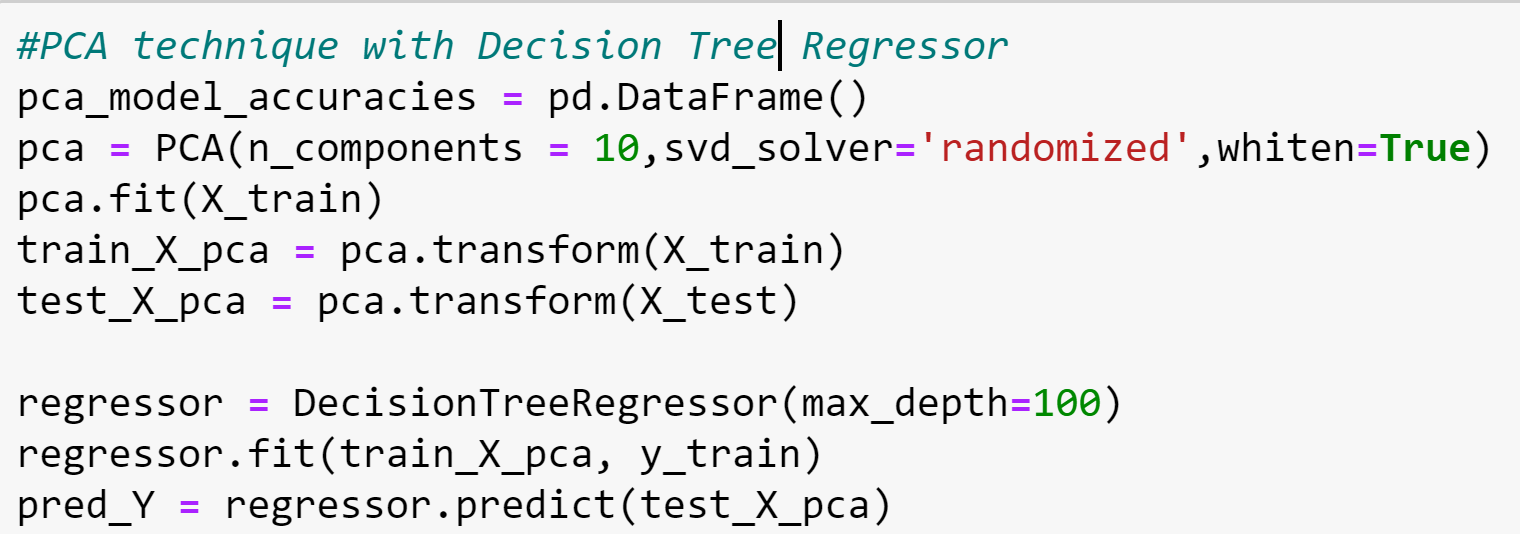


Figure 24: PCA with Decision Tree Regressor model creation

* **Recursive Feature Elimination (RFE):** It is a feature selection method that fits the model and removes the weakest features (or features) until the specified number of features is reached. Features are ranked by the model's coef or feature importances attributes, and by recursively eliminating a small number of features per loop, RFE attempts to eliminate the dependences and collinearity that may exist in the model.

1. **RFE with XGBoost:** We can recursively reject features that the model's feature\_importances\_ routine has decided are unimportant using sklearns recursive feature elimination. Or, in the exploratory phase of building your model, you can assess predictive power using visualisation or hypothesis testing.

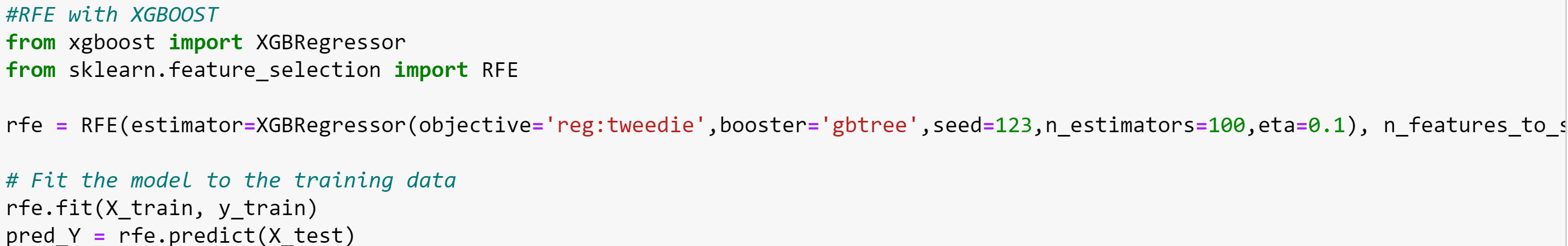


Figure 25: RFE with XGBoost Regressor model creation

Plotting the important features of the model:

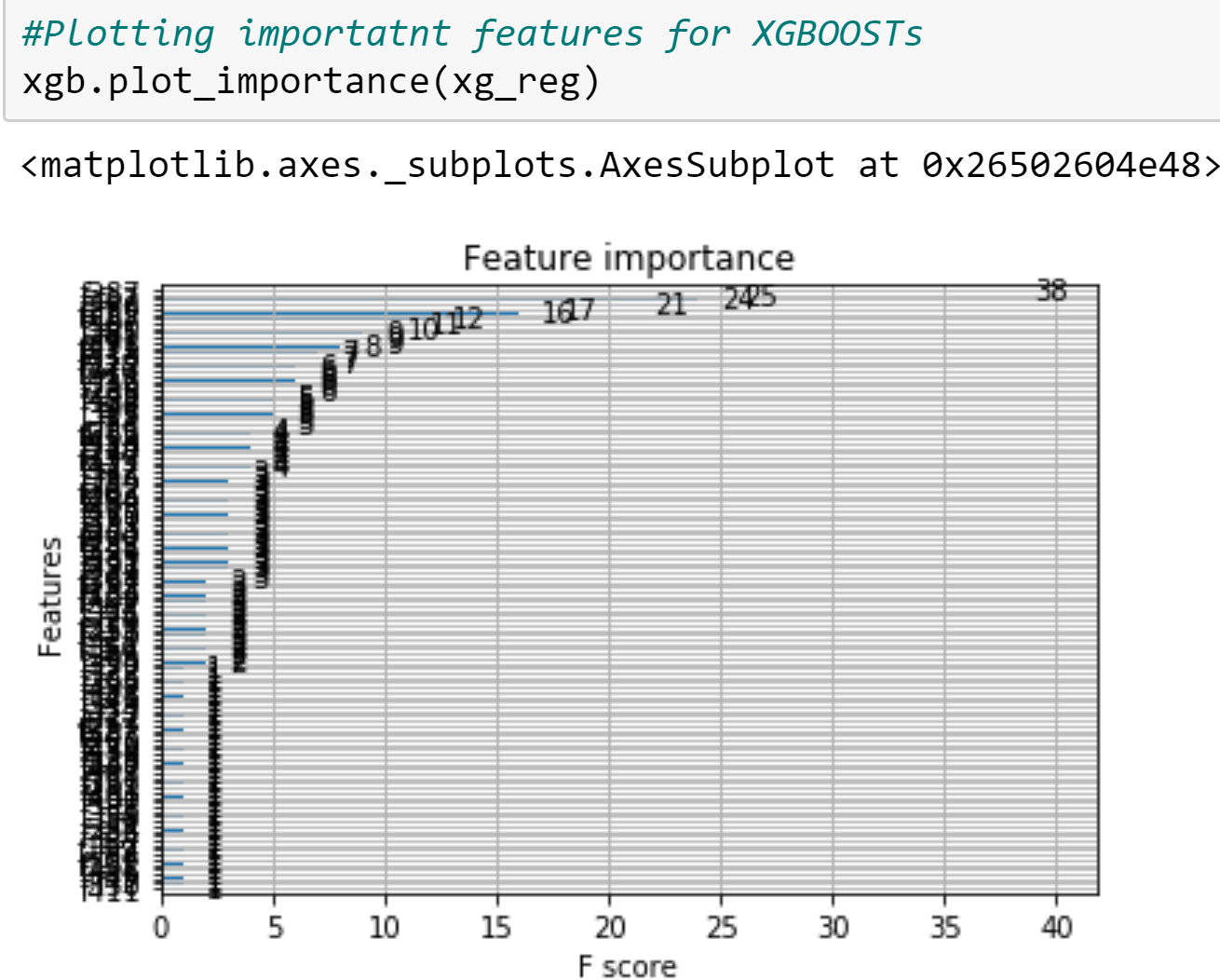


Figure 26: RFE with XGBoost -Important features

1. **RFE with Gradient Boosting Regressor:**

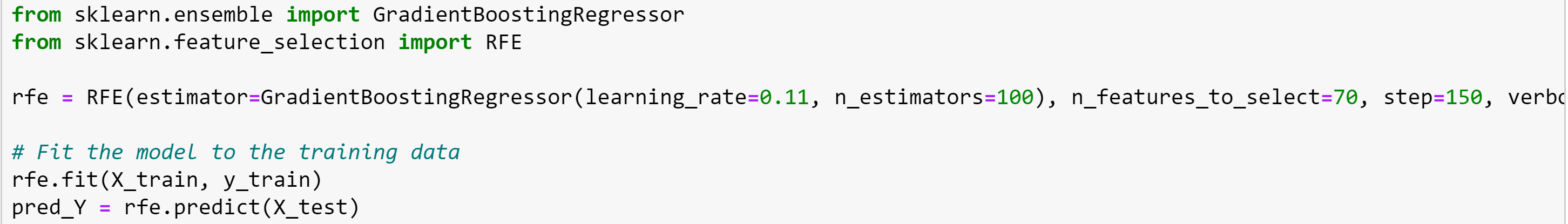


Figure 27: RFE with Gradient Boosting Regressor

1. **RFE with Random Forest Regressor**: Although RF-RFE decreased the importance of correlated variables, in the presence of many correlated variables, it also decreased the importance of causal variables, making both hard to detect. These findings suggest that RF-RFE may not scale to high-dimensional data.

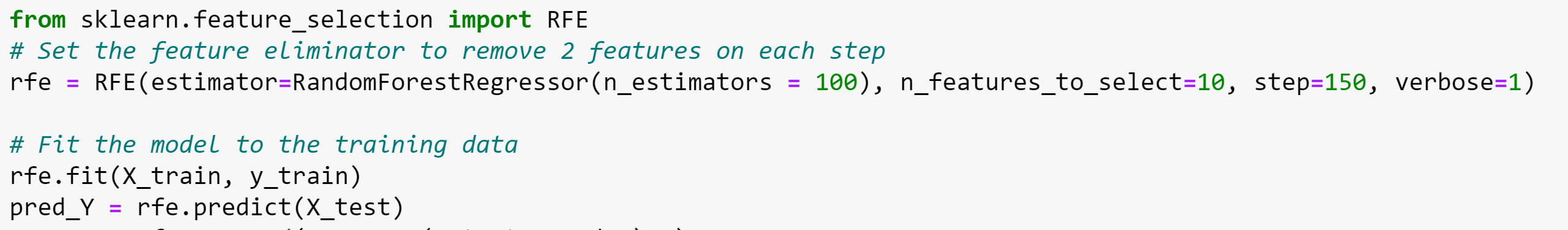


Figure 27: RFE with Random Forest Regressor

**L2 Regularization:** In L2 regularization, the regularization term is the sum of the squares of all the characteristic weights as shown in the equation above. The L2 regularization forces the weights to be small, but does not make them zero and does not produce a sparse solution.

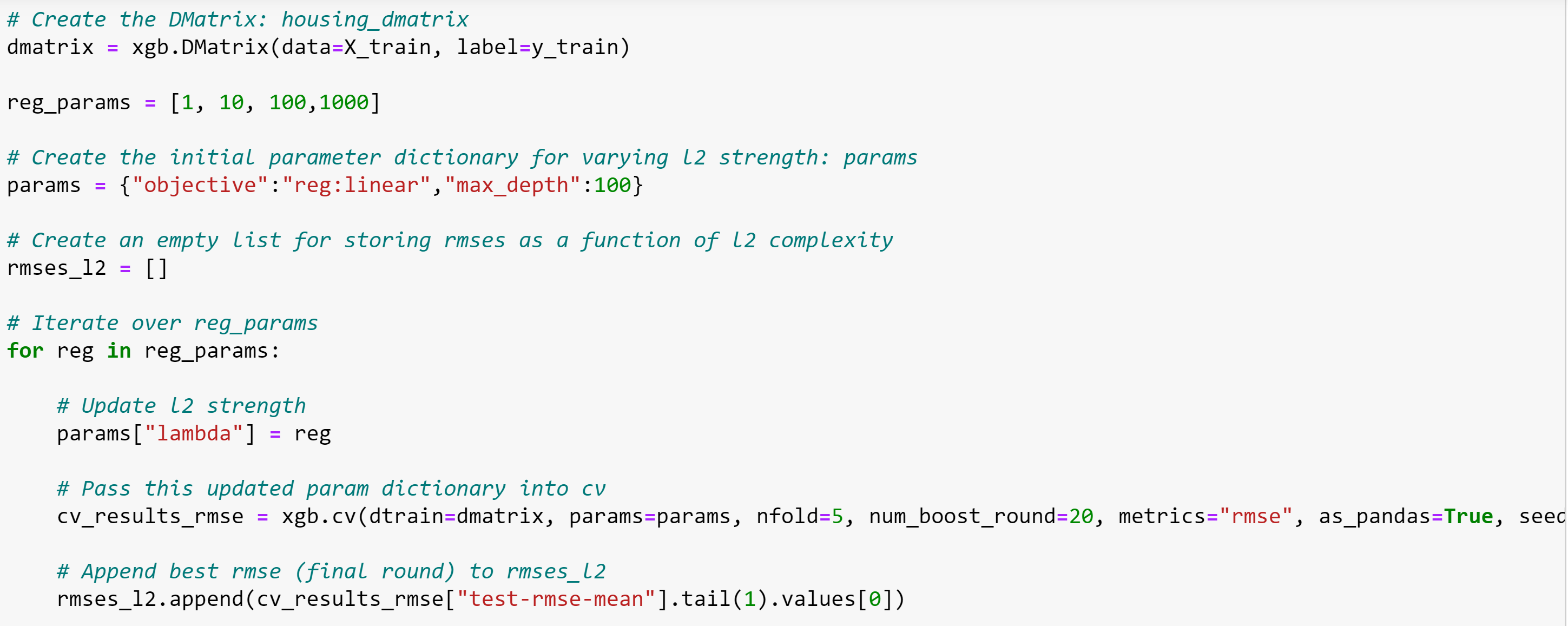


Figure 28: L2 Regularization

**PART 6: R2 EVALUATION**

|  |  |
| --- | --- |
| **Model Performed** | **R2 Score** |
| Support Vector Regressor (SVR) | 0.499 |
| Random Forest Regressor | 0.585 |
| K-Nearest Neighbors Regressor | 0.456 |
| Linear Regression | -1.040 |
| Decision Trees Regressor | 0.311 |
| XGBoost | 0.646 |
| PCA with SVR | 0.469 |
| PCA with Random Forest | 0.564 |
| PCA with KNN | 0.366 |
| PCA with Decision Tree | 0.084 |
| RFE with Gradient Boosting | 0.621 |
| RFE with Random Forest | 0.649 |
| L2 regularization | RMSE= 8.33 |

**CONCLUSION**

In this experiment, we have used regression and coefficient of determinant as our metrics to evaluate the model and predict the time in seconds for each Mercedes Benz car benchmark. Our ensemble methods like Random Forest and XGBoost model performed the best among the rest with 61% and 64.4% of R2 score with more than 360 features. It explains the variability of dependent variables on an independent variable. Our aim was to select the appropriate features for the model that resulted into better accuracy and low RMSE value. After the model evaluation with k-fold technique the rmse value kept on decreasing after 5-fold resulting into better predictions of continuous variable. After applying regularization in the features selected for the XGBoost model, it predicted the time it takes to pass testing.

**REFERENCES**

[1]. Machine Learning with R Second Edition by Brett Lantz (2015). Retrieved from <https://ia801600.us.archive.org/11/items/Machine.Learning.with.R.2nd.Edition.2015.Lantz/Machine.Learning.with.R.2nd.Edition.2015.Lantz.pdf>

[2]. Mercedes-Benz Greener Manufacturing. (n.d.). Retrieved from <https://www.kaggle.com/c/mercedes-benz-greener-manufacturing/overview/description>

[3]. Introduction to Boosted Trees¶. (n.d.). Retrieved from <https://xgboost.readthedocs.io/en/latest/tutorials/model.html>