**Shape

Description automatically generated with medium confidence**

**ESE417 Wine Quality Analysis Machine Learning Project**

Group Members: Siyi Jin, Rouxi Wang, Xiaonan Yang

Instructor: Dr. Zhang

McKelvey School of Engineering, Washington University in St.Louis

## Introduction

### 1.1 Background

Machine learning is the method that allows computers to learn automatically without any human intervention. After feeding a large amount of good-quality data, the model can be trained to make predictions or classifications by trying different algorithms.

We learned several useful machine learning models and methods from this course, including regression model, perceptron, support vector machine method, artificial neural networks method, random forest method, and so on. This final project requires us to discover a real-world dataset and apply knowledge to solve classification problems.

### 1.2 Data Set Description

The data used in this project come from a unique wine product, *Vinho Verde*, from the Minho (northwest) region of Portugal. *Vinho Verde* wines include two common categories: red wines and white wines. This project will focus on analyzing sample data for red wines.

There are 1600 instances in the dataset. Each has 11 features in data type ‘float64,’ and they jointly determine the output label ‘quality’ in data type ‘int64’.

### 1.3 Goals

It is important for manufacturers to evaluate the quality of products to provide better service for customers. However, most of the product depends on different factors. Some of the factors influence each other, so it is hard to determine the quality based on a single factor.

Given the dataset of different factors that affect wine quality, our goal is to establish a reasonable machine learning model to predict the quality level of red wines by the indicators of their physical and chemical characteristics.

### 1.4 Methods Applied

We employed three methods to train the data and chose the algorithm with the highest accuracy to improve further. The methods we applied are

* K-nearest neighbors (KNN): The class of the input point is assigned according to the class of k nearest neighbors, and the object is simply assigned to the class with the highest frequency.
* Random Forest (RF): After building a large number of decision trees, the class of the input point is the result of the majority of the trees.
* Artificial Neural Networks (ANN): Based on the coordination of neurons to communicate, the class of input point is determined by values of former layers.

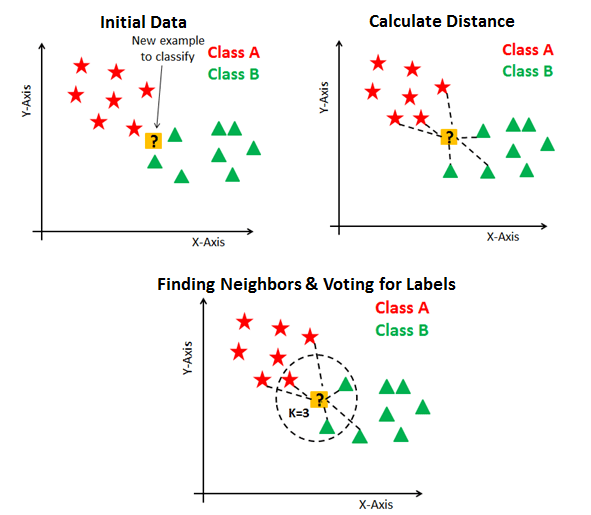
### 1.5 Brief Conclusion

The model being selected that suits the dataset best is KNN(K-nearest neighbors). However, Random forest with hyperparameters turning also has a nice performance (just 0.2% smaller than KNN).

## Methods

From preliminary analysis, there are six classifications in the range of 3 to 7. This indicates that it is not a binary but a multiple-classification problem. Thus three algorithms that aimed to solve multiple classification problems are implemented to find the primal model. They are KNN, Random Forest, and ANN.

### 2.1 K-nearest neighbors (KNN)

Chart, scatter chart

Description automatically generatedSuppose P(x,y) is the point, for which the label needs to predict. First, you need to find the k closest point to P1 and then classify points by the majority votes of its k neighbors. Each object votes for its label, and the label with the most votes is taken as the prediction. There are a few methods to calculate the distance between points, like Euclidean distance, Manhattan distance, and Minkowski distance(We use Euclidean distance in our project). The procedure of KNN is shown in the figure below:

When implemented in Python, KNeighborsClassifier is used to train the model and fit the dataset. After testing different k values in the range of 1 to 100, the one that suits best is when ,with an accuracy of .

### 2.2 Random Forest (RF)

*Random Forest* is a supervised machine learning algorithm widely used in classification and regression problems.

*Random Forest classifiers* are ensemble-based learning methods. A *Decision Tree* is a hierarchical structure that makes predictions based on a data set's features. The outcome of each feature determines which branch of the tree to follow. The fundamental principle underlying the random forest approach comprises the construction of many simple decision trees in the training stage and the majority vote across them in the classification stage.

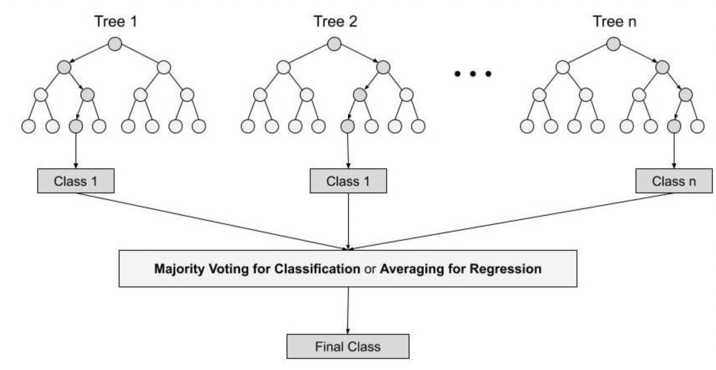
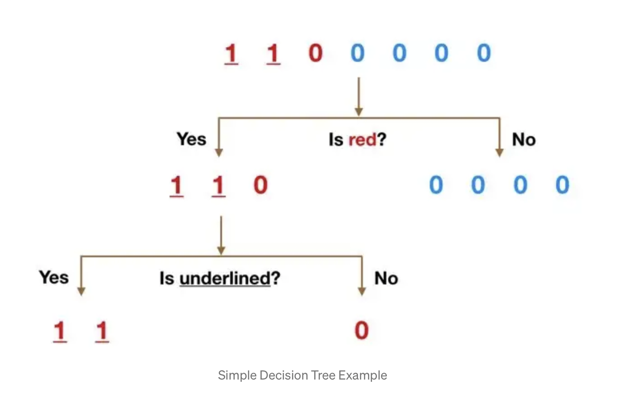
Hyperparameters:

n\_estimators: The number of trees in the forest.

criterion: A function that measures the quality of the segmentation.

max\_depth: The maximum number of nodes of the tree.

max\_features: The number of features to consider when finding the optimal split.



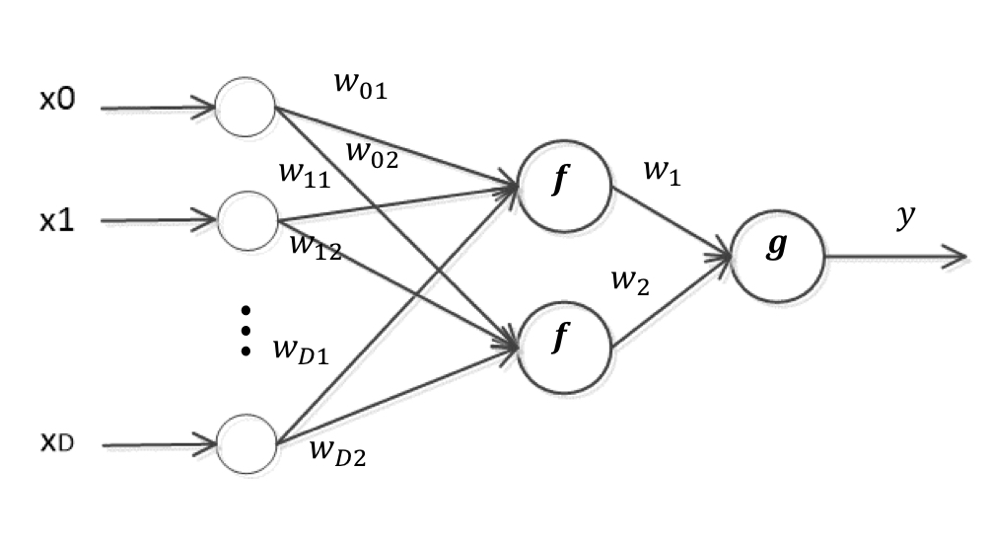
When implementing in Python, RandomForestClassifier is applied to fit the random forest model after splitting the dataset. The best estimator is when

with accuracy .



### 2.3 Artificial Neural Networks (ANN)

Artificial neural networks are inspired by biological neural networks. The process of creating a neural network begins with the perceptron. In short, the perceptron receives inputs, which is known as the input layer, multiplies them by some weights, and then passes them into an activation function (such as logistic, relu, tanh, identity) to produce a hidden layer output. Then the output of the hidden layer will repeat the procedure in the input layer to pass the data into the activation function(which might be different from the previous one) to produce the next layer. There might be multiple hidden layers. And then, the output layer receives the values from the last hidden layer and transforms them into output values.



After implementing ANN, a sigmoid function is performed to fit the model. 77 is the number of layers that gives out the best accuracy .

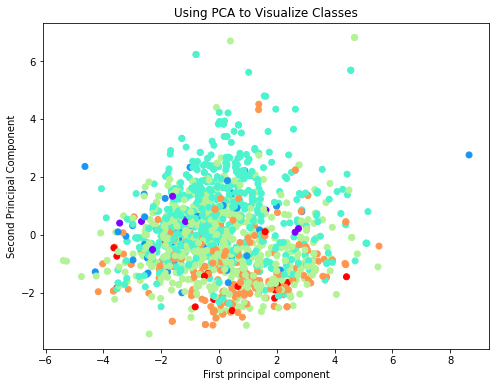
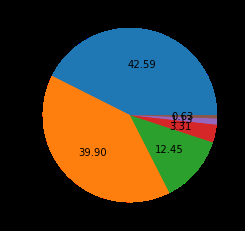
## Results and Analysis

Originally, our group applied three models to classify the given dataset. However, the accuracy results turned out relatively low. Thus we plan to make some improvements by simplifying the dataset and picking two algorithms that suit the dataset better. Finally, after comparison, our group concluded to use KNN as the final model with improved accuracy .

### 3.1 Data Analysis

Before applying different algorithms, Principal Component Analysis (PCA) is implemented to visualize the dataset. PCA is the method of mapping high-dimensional data down to 2 or 3 dimensions to plot and analyze the data more clearly.

The visualization is shown below:



The first plot in the figure above indicates the label distribution of the dataset: most wine has quality 5 or 6. It is obvious that there are many more normal wines than excellent or poor ones. Together with the results listed in Appendix, it is clear that all features and quality are imbalanced and with no null values.

Treemap chart

Description automatically generatedBy performing the PCA method, we got the first principle component responsible for 28.2% of the variance and the second one 17.5%. Based on the two components, the right plot in the figure shows the distribution of 11 features after reducing the dimension to 2 dimensions. And it is clear that this is very nonlinearly separable and is a difficult problem. The data cleaning process needs to be performed to reduce the dimensions and improve accuracy.

The matrix displayed in the figure below indicates the correlation between each variable. The darker the color, the more relevant two variables are. The lowest row indicates the correlation between features and label. All values is smaller than |0.5| which indicates they are weakly correlated. Some of the variables have large correlation such as the value of density versus fixed acidity is 0.67, meaning they are strongly correlated. This may lead to collinearity problem. To deal with it, two methods are put forward in the following section.

### 3.2 Improvement Method

1. Feature drop

The more features considered, the more difficult to fit the model and the more difficult to get a high accuracy. Reducing input dimensions is a common way to reduce the difficulty of code operation and improve the model quality.

Chi- square test is a very widely used hypothesis test method. The chi-value is larger if the deviation between actual observed value and the theoretical inferred value of the sample is larger. In this project, the ‘chi’ matrix is applied to determine which and how many features to drop to reduce the dimensions. The resulting order from highest to lowest is listed in the following table.

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Score** | **Features** | **Score** |
| total sulfur dioxide | 2755.557984 | sulphates | 4.558488 |
| free sulfur dioxide | 161.936036 | residual sugar | 4.123295 |
| alcohol | 46.429892 | chlorides | 0.752426 |
| volatile acidity | 15.580289 | pH | 0.154655 |
| citric acid | 13.025665 | density | 0.000230 |
| fixed acidity | 11.260652 |  |  |

Scores in the table represent the relationship between certain features and the label. Higher score means there is a higher possibility that this feature has a relationship with the label. This means features with higher ranks have greater influence on wine quality. It is clear that the last three features ‘density’, ‘pH’ and ‘chlorides’ have chi-values lower than 1, indicating they have little contribution in determining the wine quality. Thus these features are dropped to reduce the data dimensions.

1. SMOTE-Tomek Links Method

SMOTE-Tomek Links Method combines the capabilities of SMOTE and Tomek Links methods, with SMOTE for oversampling and Tomek Links for data cleaning.

SMOTE allows for synthesizing new samples from existing samples, oversampling the data on minority groups.

SMOTE-steps:

1. Select random data in the minority category.
2. Calculate the Euclidean distance of the random data from its neighbors.
3. Multiply the difference between the distance of the random data and its neighbors by a random number between 0 and 1.
4. add the result to the minority category as a synthetic sample.
5. Repeat this process, granting the percentage of minorities until the data set is balanced.

Tomek link is a low-sampling technique. It finds the closest data to the minority category data in the majority category by calculating the Euclidean distance and then removes it.

### 3.3 Model Comparison

Table below compares three methods applied originally.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Accuracy | Speed & Time | Advantage | Disadvantage |
| KNN | 57.1% | Fast: O(n) | No need to retrain when new points are added | Low accuracy when the sample is imbalanced |
| RF | 66.5% | Fast: parallel operation | Easy to visualize.  Able to handle large factors | Easy to overfit  Overlook data associations |
| ANN | 48.9% | Take too long to study | Strong learning ability  Strong noise tolerance | Require lots of parameters |

After comparing three algorithms, we chose KNN and Random Forest to further optimize the classification process. Since ANN takes too long to implement as well as a relatively low accuracy, it is not considered in the improvement part.

### 3.4 Improved Results

Two models both performed well after feature drop and apply SMOTE method, two models reach to accuracy higher than , as shown in the table below.



|  |  |  |  |
| --- | --- | --- | --- |
| Models | Method of Improvement | Accurracy Rate | Growth Rate of Accuracy |
| RF | Original | 66.458% | - |
| Drop Features | 66.458% | 0% |
| SMOTE-Tomek | 84.375% | +17.917% |
| Drop Features + SMOTE-Tomek | 86.042% | +19.584% |
| KNN | Original | 50.416% | - |
| Drop Features | 50.416% | 0% |
| SMOTE-Tomek | 88.75% | +38.334% |
| SMOTE-Tomek+Drop Feature | 86.66% | +36.244% |

Improved results indicate that two methods proposed by the group worked well and successfully solved the problem.

## Conclusions

For this project, we used three machine learning algorithms in total which we learned on ESE417(ANN,KNN,Random forest). For all the algorithms, we first observe the dataset, and use “quality”as the target value, others are features. And then we split the dataset into training dataset and test dataset. For all three algorithms, we first use the original training dataset to fit and the Accuracy we got are KNN: 55.8%, ANN: 48.9%,RF: 66.5%. Apparently these results do not meet our expectations. To figure out why we get such low accuracy, we first observed that there exists data imbalance by the Distribution of features and label histograms we got. Then when we implemented PCA,we get three features: 'density’, ‘pH’ and ‘chlorides’ have chi-values lower than 1. We use the SMOTE-Tomek method to deal with the imbalance problem and we tried the original dataset and the Feature-dropped dataset on it separately.We can easily see that for Random Forest, Drop Features + SMOTE-Tomek one get a better result. For KNN, SMOTE-Tomek one gets a better result.We think it’s because feature dropping would let KNN catch more noise so that the accuracy is lower. However, after implementing the SMOTE-Tomek, the accuracy of Two models improved a lot, both >80%. Generally speaking, through this project, our group realized the procedure of classification with the model in scikit-learn package, and we also learned how to figure out the dataset problems and improve the accuracy of models.

All three group members contribute to idea discussion, code programming and report writing process. The detailed division of labor is listed below:

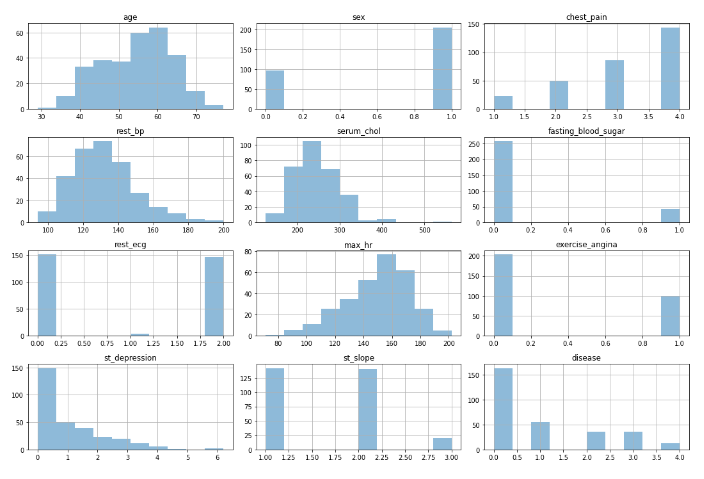
|  |  |  |
| --- | --- | --- |
| Name | Percentage | Main Responsibility |
| Siyi Jin | 33.33% | RF code, SMOTE idea proposal, partly report writing. |
| Ruoxi Wang | 33.33% | PCA code and main report writing. |
| Xiaonan Yang | 33.33% | ANN and KNN code and partly report writing. |

## References

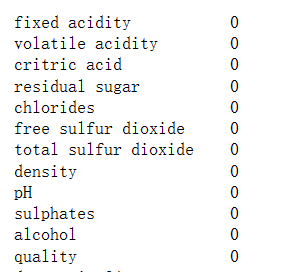
1. <https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn>
2. https://towardsdatascience.com/understanding-random-forest-58381e0602d2
3. https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/
4. https://scikit-learn.org/stable/modules/neural\_networks\_supervised.html
5. https://www.aitude.com/multiclass-classification-on-highly-imbalanced-dataset/

## Appendix

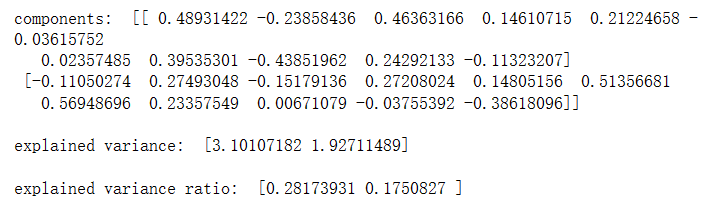
1. Distribution of features and label:



2. Distribution of features and label:



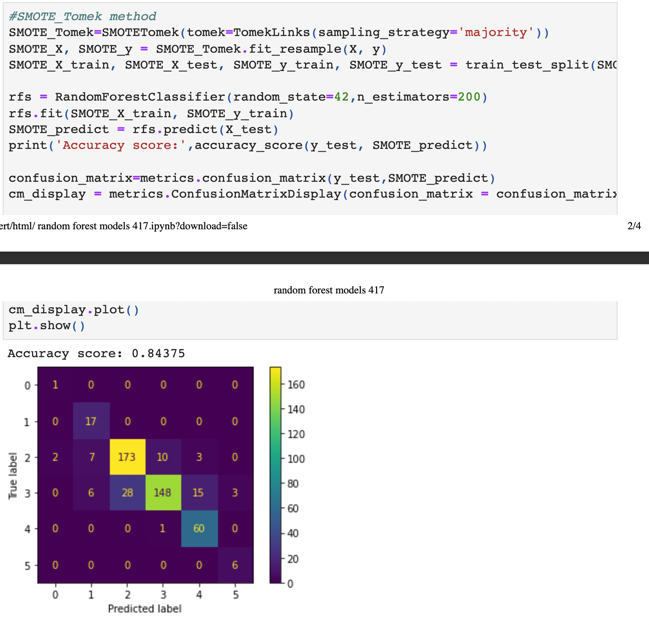
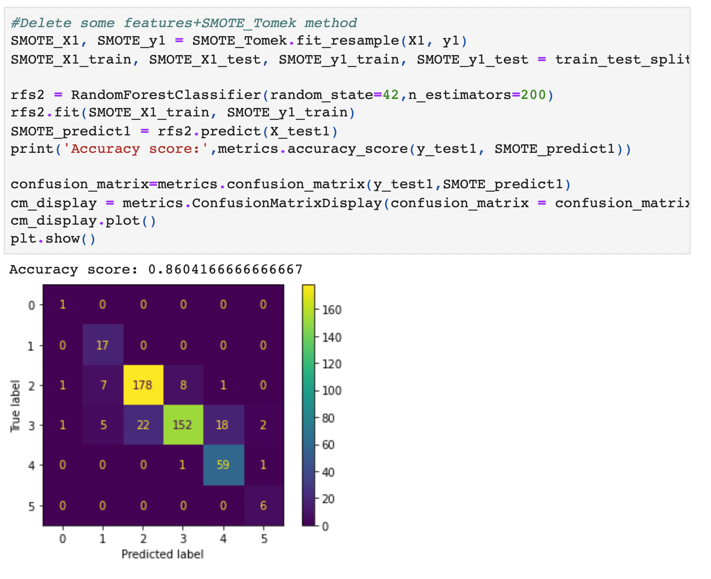
3. Principal Component Analysis



4. Random Forest Classifier



5. Random Forest Classifier with improved methods

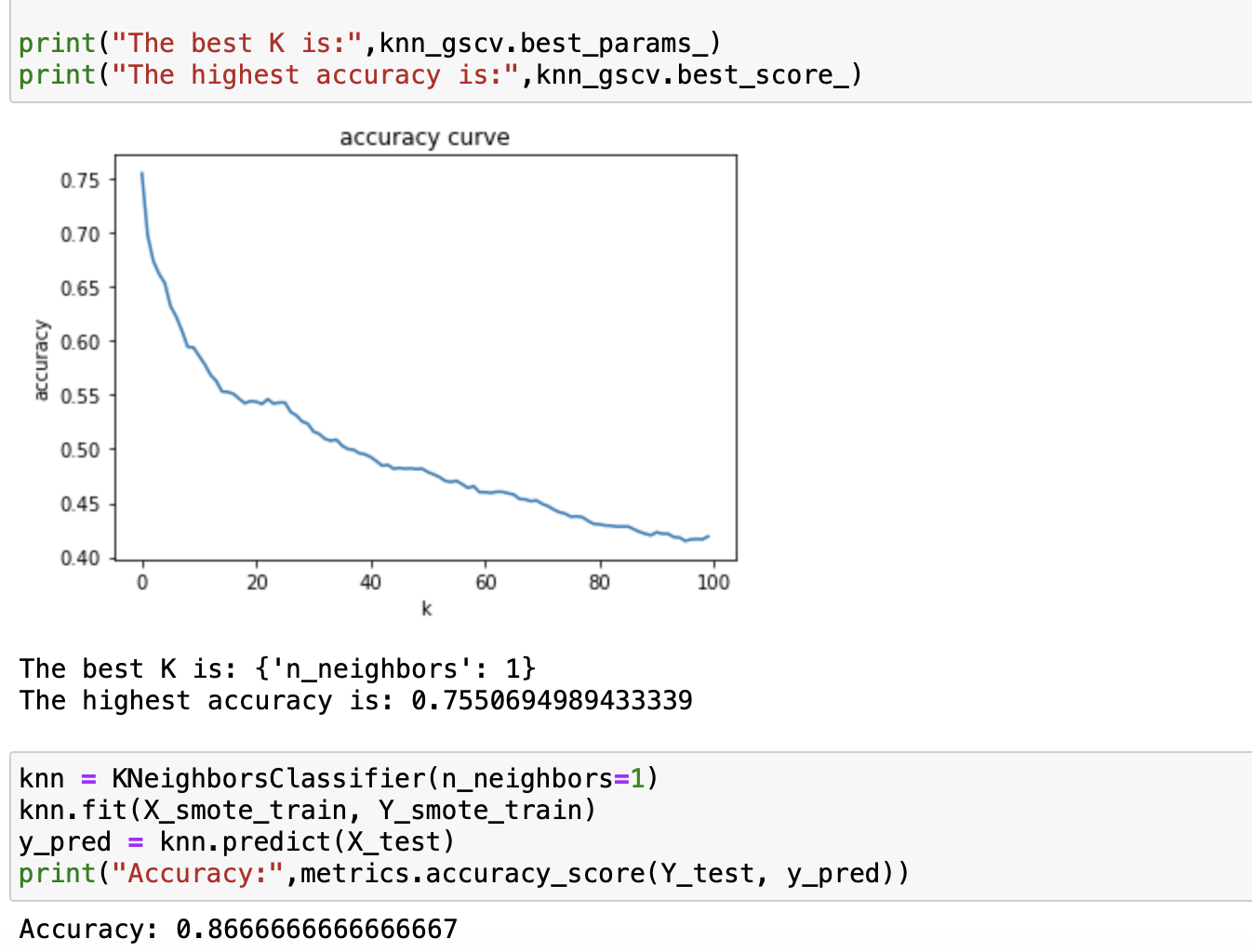


6.KNN(K-nearest neighbors)

Graphical user interface, text

Description automatically generated

7.KNN with improved methods

7.1 Smote with feature drop

7.2 Somte without feature drop

