Student ID – 201669264
Name: Rahul Arun Nawale

1.0. Introduction:

The given Python code uses the Scikit-learn library for implementing three classification methods, namely, K-Nearest Neighbors (KNN), Logistic Regression, and Random Forest. These classification methods are applied to the given dataset 'dataset_assignment1.csv' to train and test the models.

1.1. The following libraries are used in the code:

Pandas: For data manipulation and analysis.

sklearn.neighbors: A sublibrary of scikit-learn (sklearn) used for implementing K-Nearest Neighbor (KNN) algorithm, which is a non-parametric classification algorithm used for classification and regression tasks.

sklearn.model_selection: A sublibrary of scikit-learn (sklearn) that provides tools for model selection and evaluation, such as cross-validation and train-test split.

sklearn.metrics: A sublibrary of scikit-learn (sklearn) that provides tools for evaluating model performance, such as classification report, confusion matrix, accuracy, precision, recall, and F1 score.

sklearn.linear_model: A sublibrary of scikit-learn (sklearn) that provides tools for linear models, such as logistic regression.

sklearn.ensemble: A sublibrary of scikit-learn (sklearn) that provides tools for ensemble learning methods, such as random forest.

matplotlib.pyplot: A library for creating static, animated, and interactive visualizations in Python.

seaborn: A library for data visualization based on matplotlib. It provides a high-level interface for creating informative and attractive statistical graphics.

1.2. Classification Methods Used

K-Nearest Neighbors (KNN):

KNN is a supervised learning algorithm used for classification problems. It works by finding the K nearest data points in the training dataset and assigning the class label based on the majority class in the K nearest data points. In this code, the 'KNeighborsClassifier' class is used to implement KNN.

Logistic Regression:

Logistic Regression is a supervised learning algorithm used for binary classification problems. In this code, the 'LogisticRegression' class is used to implement Logistic Regression.

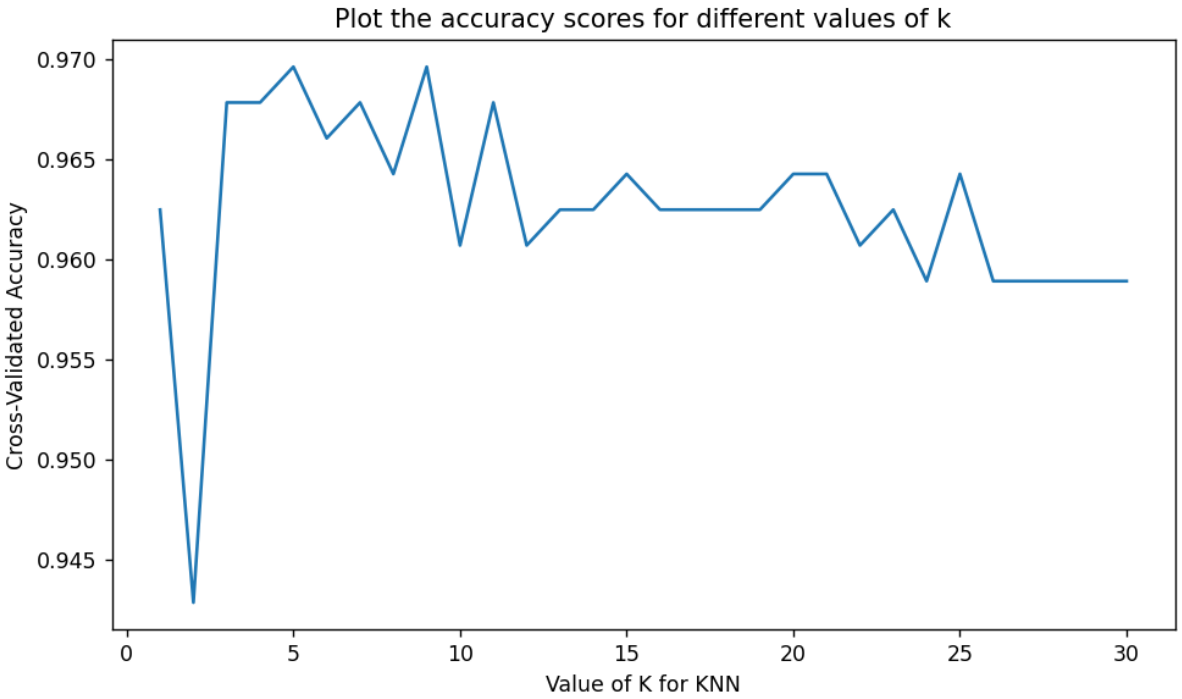
Random Forest:

Random Forest is a supervised learning algorithm used for classification and regression problems. In this code, the 'RandomForestClassifier' class is used to implement Random Forest.

1.3. Hyperparameters Selection:

The hyperparameters for each classification method are selected as follows:

KNN: The value of 'n_neighbors' is set to 5. Additionally, cross-validation is used to tune the hyperparameters of the algorithm. The range of values tested for 'n_neighbors' is 1 to 30. See below figure.



Logistic Regression and Random Forest:

For the Logistic Regression and Random Forest algorithms, the default hyperparameters are used (i.e., K=5).

1.4. Training and Testing Process:

1. The dataset is loaded using the Pandas library.
2. The number of rows, columns, column names, and data types are printed to understand the dataset's structure.
3. The number of samples for each class is also printed.
4. The dataset's features are visualized using histogram, density plot, and boxplot. The statistical description of features for each class is printed.
5. The data is split into input and output variables (X and y).
6. The data is then split into training and testing datasets using the train_test_split function from the scikit-learn library , with a test size of 0.2 and a random state of 0.

The KNN algorithm is trained using the training dataset.

1. The K-Fold Cross-Validation technique is used to tune the k value. The accuracy scores for different values of k are plotted.
2. The KNN algorithm is then tested using the testing dataset, and the confusion matrix is plotted.
3. The accuracy, precision, recall and F1-score of the model on the test data is obtained using scikit-learn library.

The same process is repeated for the Logistic Regression and Random Forest algorithms using k=5 for K-fold cross validation.

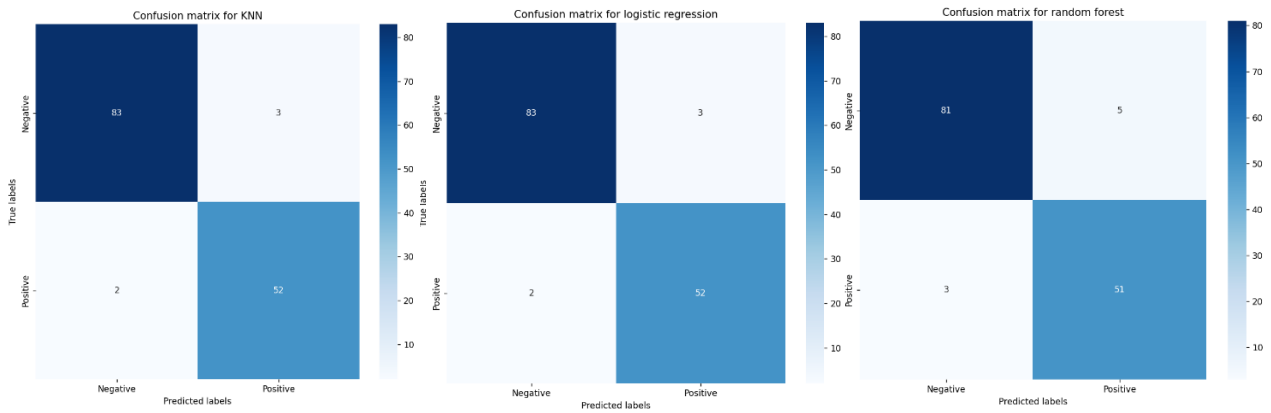
2.0. Evaluation

	KNN	Logistic Regression	Random Forest
Accuracy	0.96428	0.96428	0.94285
Precision	0.94545	0.94545	0.91071
Recall	0.96296	0.96296	0.94444
F1-Score	0.95412	0.95412	0.92727
Confusion Matrix	[83 3] [2 52]	[83 3] [2 52]	[81 5] [3 51]

Table: Comparison of evaluation metrics

The output of the code shows the classification report, confusion matrix, and evaluation metrics for three classification models: KNN, Logistic Regression, and Random Forest.

2.1. Confusion Matrix:



KNN Logistic regression Random Forest

A confusion matrix is a table that summarizes the classification results of a model by comparing predicted and actual labels. The matrix shows the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) for each class. In the case of binary classification, the matrix is a 2x2 table.

KNN:

The confusion matrix of the KNN model shows that out of 140 instances, 83 instances are true negatives, 52 instances are true positives, 3 instances are false negatives, and 2 instances are false positives.

Logistic Regression:

The confusion matrix of the Logistic Regression model is also the same as the KNN model.

Random Forest:

The confusion matrix of the Random Forest model shows that out of 140 instances, 81 instances are true negatives, 51 instances are true positives, 5 instances are false negatives, and 3 instances are false positives.

2.2. Evaluation metrics

KNN Classifier Model:

Accuracy: The KNN Classifier Model has an accuracy of 0.9643, which means that 96.43% of the predictions made by the model are correct.

Precision: The precision of the model is 0.9454, indicating that when the model predicts a positive class, it is correct 94.54% of the time.

Recall: The recall score of the model is 0.9630, which means that the model correctly identifies 96.30% of the positive class instances.

F1 score: The F1 score of the model is 0.9541, which is the harmonic mean of precision and recall scores. It gives an overall idea of the model's performance in terms of both precision and recall.

Logistic Regression Model:

The Logistic Regression Model has the same accuracy, precision, recall, and F1 score as the KNN Classifier Model, which indicates that both models have the same performance.

Random Forest Model:

Accuracy: The Random Forest Model has an accuracy of 0.9429, which is slightly lower than the other two models.

Precision: The precision score of the model is 0.9107, which is the lowest among the three models, indicating that when the model predicts a positive class, it is correct only 91.07% of the time.

Recall: The recall score of the model is 0.9444, which is higher than the precision score, indicating that the model correctly identifies 94.44% of the positive class instances.

F1-Score: The F1 score of the model is 0.9273, which is the lowest among the three models.

3.0 Conclusion:

3.1. About results

All three models have almost similar accuracy, precision, recall, and F1 scores with KNN and logistic regression being exactly identical, but there are some differences in their confusion matrices. The KNN and Logistic Regression models have similar confusion matrices with few misclassifications, while the Random Forest model has a slightly higher number of misclassifications.

1. In general, **KNN Classifier** is a simple and effective algorithm for classification problems, but it can be computationally expensive for large datasets.
2. **Logistic Regression** is also a widely used classification algorithm, particularly for binary classification problems, and it is relatively easy to implement.
3. **Random Forest** is used for both classification and regression problems, but it can be prone to overfitting if the number of trees is too high.

3.2. My views on the assignment/task:

It is a classification problem with a relatively small dataset, so the models perform quite well. It would be interesting to explore other classification algorithms and fine-tune the hyperparameters to see if we can improve

the model's performance. Additionally, it would be beneficial to perform feature selection to determine which features are the most informative in predicting the target variable.

3.3. What I have learned:

I have learned to classify any given dataset using different classification methods such as KNN, logistic regression, random forest etc. I also learned how to evaluate a model's performance using different evaluation metrics such as confusion matrix, accuracy, precision, F1- score etc.

I would try to classify more complex dataset with more than 2 classes and check which models works best.