**Kairiz CYber Technologies**

***Submitted By: Abdullah Amjad***

***Cell No: 03309772003***

***Gmail: abdullahamjad17301@gmail.com***

***Instructor: Sir Afaq and Mam Noor***

***CEO: Kainat Rizwan***

***Date: 13 July 2024.***

**Task: 2**

1. **Libraries:**

* **pandas**: For handling data structures and operations.
* **sklearn**: For machine learning tasks such as data splitting, preprocessing, and modeling.
* **SimpleImputer**: For handling missing values.
* **OneHotEncoder**: For encoding categorical variables.
* **StandardScaler**: For scaling numerical features.
* **Pipeline, ColumnTransformer**: For combining preprocessing steps.

1. **Load Dataset:**

First of all, I will load the titanic dataset means csv file then I will work on handling the missing values same as I already work in task1.

1. **One-Hot Encoding and Some Features:**

In the hot encoding first I will create the some dummy variable in Sex and Embarked columns in csv file then I will use **align()** function to ensure the size of the different columns.

Then I will define some features like ‘**FamilySize’** by summing '**SibSp'** and **'Parch'** and **adding 1**. And then second feature is '**IsAlone'** to indicate if a passenger is alone or not.

Then some preprocessing steps are applied to these feature sets.

1. **Split the Data:**

In this step I will separate the train data and test data using some parameters and then I will display the shapes of both training sets and testing sets but in parameters…

1. **Initialize and Train Models:**

Now I will use two different algorithms i.e. Random Forest and SVM but I will import RandomForeestClassifier and SVC for initializing the random forest and SVM algorithms.   
Then after the initialization now training phase is start and I am using fit function in training the algorithms.

1. **Hyperparameter Tuning:**

Basically, hyperparameters are settings that we configure before training the model to for maximize the model’s performance.

**Hyperparameters for Random Forest:**

* **n\_estimators:** The number of trees in the forest. Possible values are 100, 200, and 300.
* **max\_depth:** The maximum depth of each tree. Possible values are None, 10, 20, and 30.
* **min\_samples\_split:** The minimum number of samples required to split an internal node. Possible values are 2, 5, and 10.

A screenshot of a computer

Description automatically generated

**Hyperparameters for SVM:**

* **C:** Regularization parameter. It controls the trade-off between achieving a low training error and a low testing error. Possible values are 0.1, 1, and 10.
* **kernel:** Specifies the kernel type to be used in the algorithm. Possible values are 'linear', 'rbf' (radial basis function), and 'poly' (polynomial).
* **gamma:** Kernel coefficient for 'rbf', 'poly', and 'sigmoid'. Possible values are 'scale' and 'auto'.

**Grid Search Process** automates the process of testing different hyperparameters to find the combination that yields the best performance. It performs an exhaustive search over the specified hyperparameters.

Once the grid search is complete then I have the best hyperparameters and their corresponding performance. Then I will print the best parameters and scores for both models.

A screenshot of a computer

Description automatically generated

1. **Predictions and Evaluation:**

In predictions I will use the sklearn.model\_selection for GridSearchCV . These are some libraries that I used in prediction. And I will use the **predict** function from GridSearchCV to make predictions on the test set.

In Evaluation of models, I will use the

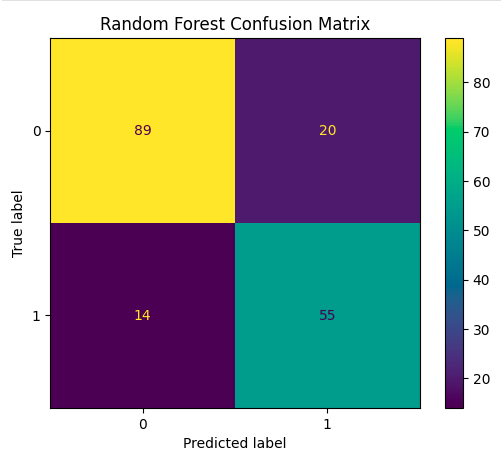
* **accuracy\_score:** Computes the accuracy of the predictions.
* **precision\_score:** Computes the precision of the predictions.
* **recall\_score:** Computes the recall of the predictions.
* **f1\_score**: Computes the F1 score of the predictions.
* **roc\_auc\_score**: Computes the ROC AUC score of the predictions.

**Display:**

At the end I will display the matrices that I will obtained from the evaluation for both Random Forest and SVM.

1. **Confusion Matrix and ROC Curve:**

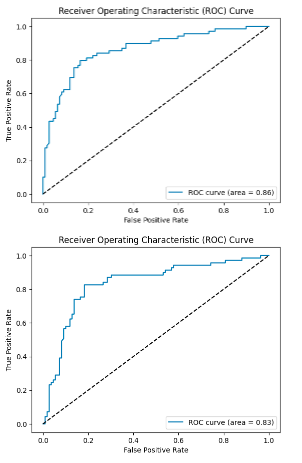
In this step I will use the **ConfusionMatrixDisplay** library to display the confusion matrix for both models. Then after that I will use the **roc\_curve** and **auc** libraries for the plotting of ROC curves for both models.



A chart of different colored squares

Description automatically generated

ROC Curve:



1. **Analyze Feature Importance:**

Feature Importance is a measure of how useful each feature is in predicting the target variable in the model. In Random Forest model feature\_importances is the attribute to get the importance of each feature.

1. **Plotting Feature Importance:**

In plotting I will use the matplotlib.pyplot library and the function I will use are

* **Barh()** function to create a horizontal bar chart.
* **Xlabel()** to label the x-axis.
* **Title()** to set the title of the plot.
* **Show()** to display the plot.

A graph with blue squares

Description automatically generated

1. **Individual Predictions:**

In this step I am using the LIME for explaining the individual predictions. **lime.lime\_tabular** this is the library that I am using in predictions. The functions that I am using are **LimeTabularExplainer** from **Lime\_tabular.**

Then I will select a sample from the test set basically I am choosing a specific instance to explain its predictions. After that I will use the **explain\_instance** to explain the predictions for the selected sample and display the results.

**Helping Material:**

<https://youtu.be/dRBYkDNkrBI?si=qyYM3qQ68YqdHQ19>

<https://youtu.be/EKk_mcobsF8?si=Wvkc5S8gya8pFKxi>

<https://youtu.be/25WeAQyoHXc?si=UkAkI-2pM74xSio9>

<https://youtu.be/fATVVQfFyU0?si=Yd9Zjy_SpGExOkrx>

<https://youtu.be/Ea_KAcdv1vs?si=A9fs39ILRJOrpws5>

<https://youtu.be/Ea_KAcdv1vs?si=A9fs39ILRJOrpws5>

<https://youtu.be/ZsM2z0pTbnk?si=jI7d3xckAXUSeDrC>

Thanks.