Model Development Phase Template

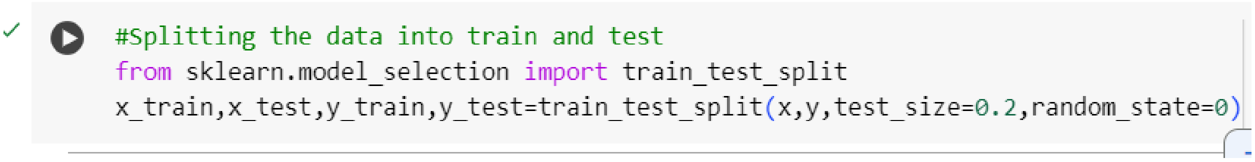
Date	29 September 2024
Team ID	LTVIP2024TMID24973
Project Title	Detection of Phishing Websites from URLs Using Machine learning
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

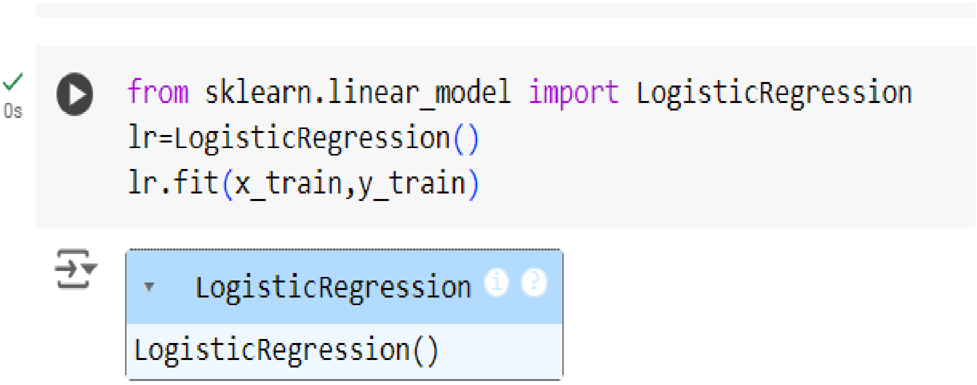
Initial Model Training Code:

Splitting the data:



```
#Splitting the data into train and test
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

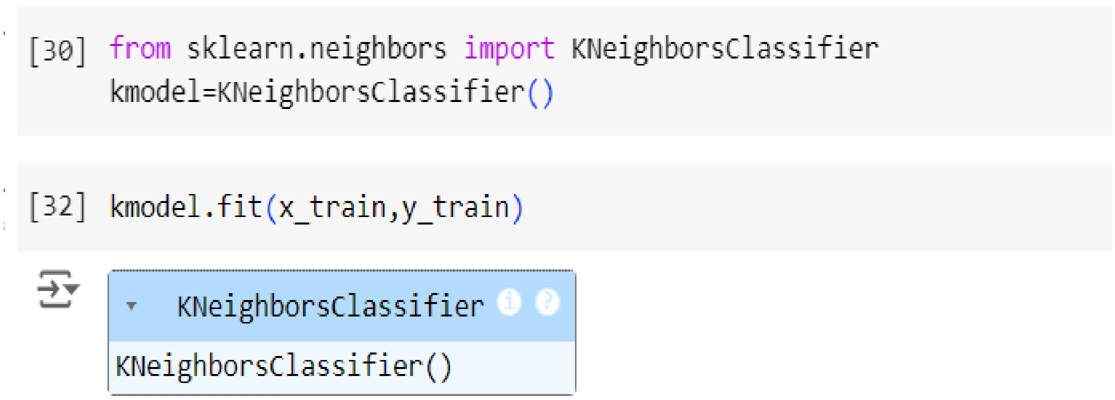
Training the model:



```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

K Nearest Neighbour

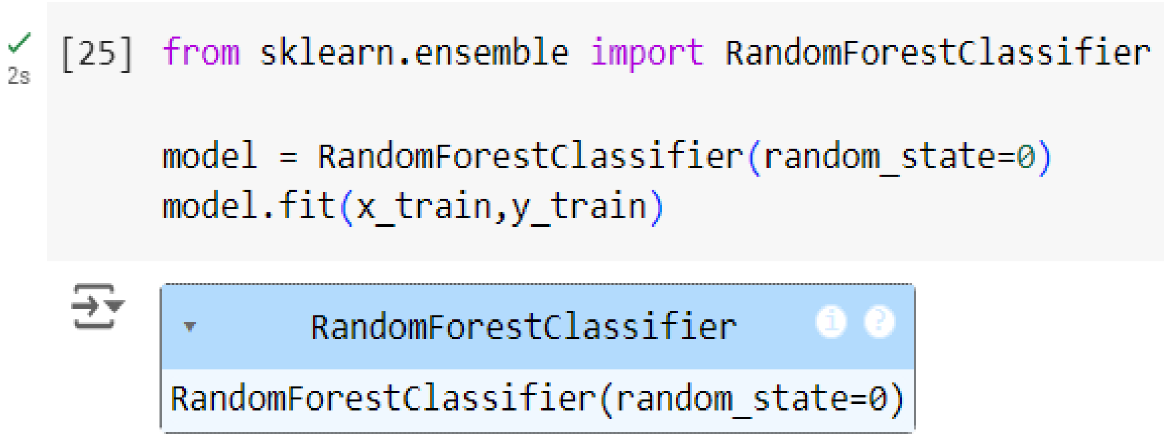


```
[30] from sklearn.neighbors import KNeighborsClassifier
     kmodel=KNeighborsClassifier()
```

```
[32] kmodel.fit(x_train,y_train)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

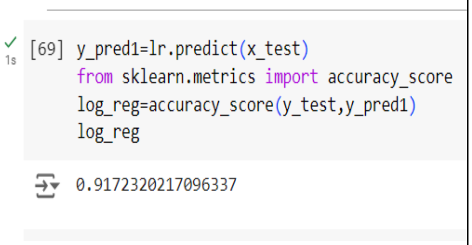
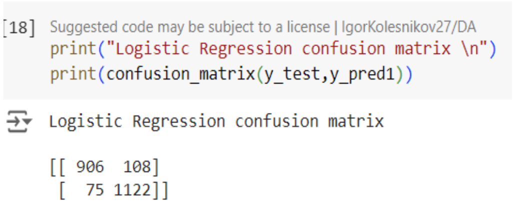
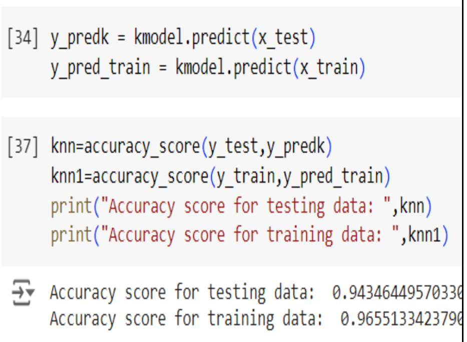
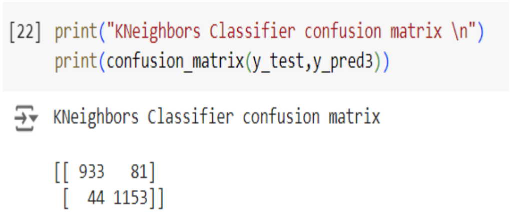
Random forest classifier

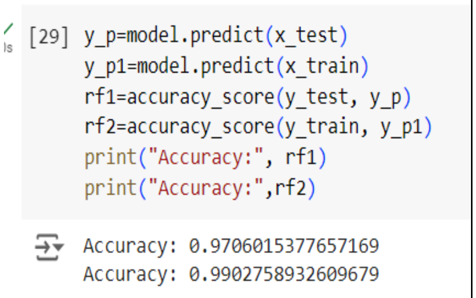
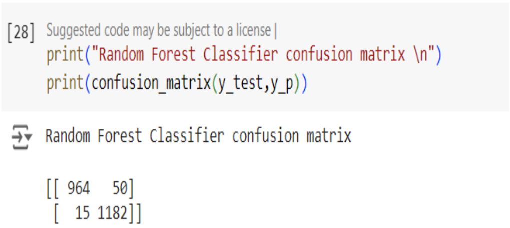


```
✓   [25] from sklearn.ensemble import RandomForestClassifier
2s
         model = RandomForestClassifier(random_state=0)
         model.fit(x_train,y_train)
```

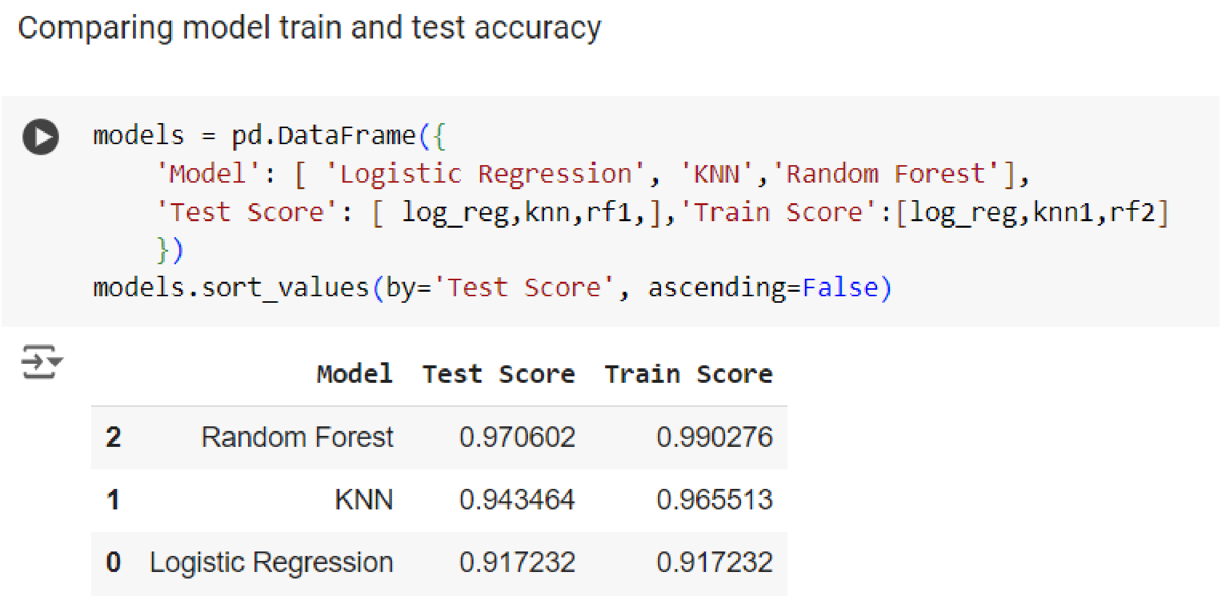
```
▾ RandomForestClassifier
RandomForestClassifier(random_state=0)
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Linear regression	 ✓ 1s [69] y_pred1=lr.predict(x_test) from sklearn.metrics import accuracy_score log_reg=accuracy_score(y_test,y_pred1) log_reg 0.9172320217096337	0.91	 [18] Suggested code may be subject to a license | IgorKolesnikov27/DA print("Logistic Regression confusion matrix \n") print(confusion_matrix(y_test,y_pred1)) Logistic Regression confusion matrix [[906 108] [75 1122]]
KNN	 [34] y_predk = kmodel.predict(x_test) y_pred_train = kmodel.predict(x_train) [37] knn=accuracy_score(y_test,y_predk) knn1=accuracy_score(y_train,y_pred_train) print("Accuracy score for testing data: ",knn) print("Accuracy score for training data: ",knn1) Accuracy score for testing data: 0.94346449570330 Accuracy score for training data: 0.9655133423790	0.94	 [22] print("KNeighbors Classifier confusion matrix \n") print(confusion_matrix(y_test,y_pred3)) KNeighbors Classifier confusion matrix [[933 81] [44 1153]]

Random Forest	
	✓ 1s [29] y_p=model.predict(x_test)
	y_p1=model.predict(x_train)
	rf1=accuracy_score(y_test, y_p)
	rf2=accuracy_score(y_train, y_p1)
	print("Accuracy:", rf1)
	print("Accuracy:",rf2)
	Accuracy: 0.9706015377657169
	Accuracy: 0.9902758932609679
		0.97	
			[28] Suggested code may be subject to a license |
			print("Random Forest Classifier confusion matrix \n")
			print(confusion_matrix(y_test,y_p))
			Random Forest Classifier confusion matrix
			[[964 50]
			 [15 1182]]

Comparing of model train test accuracy:



Comparing model train and test accuracy

```
models = pd.DataFrame({
    'Model': [ 'Logistic Regression', 'KNN','Random Forest'],
    'Test Score': [ log_reg,knn,rf1,],'Train Score':[log_reg,knn1,rf2]
    })
models.sort_values(by='Test Score', ascending=False)
```

	Model	Test Score	Train Score
2	Random Forest	0.970602	0.990276
1	KNN	0.943464	0.965513
0	Logistic Regression	0.917232	0.917232