Model Optimization and Tuning Phase Report

Date	18 June 2024
Team ID	739634
Project Title	Flight Delays Prediction Using Machine Learning
Maximum Marks	10 Marks

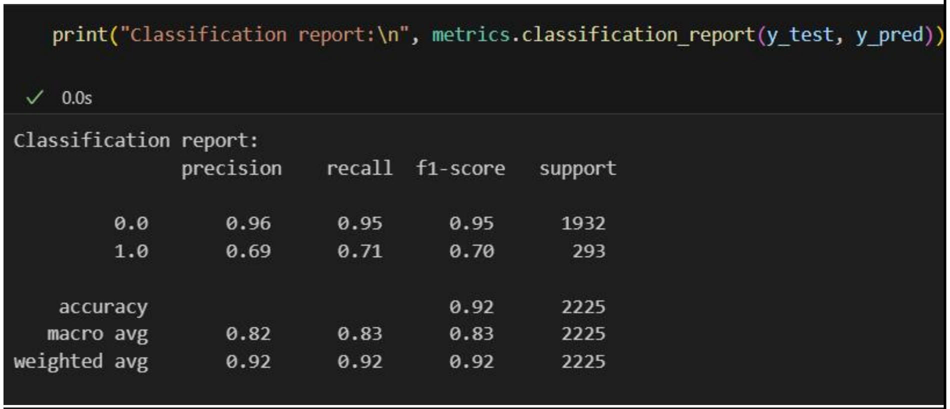
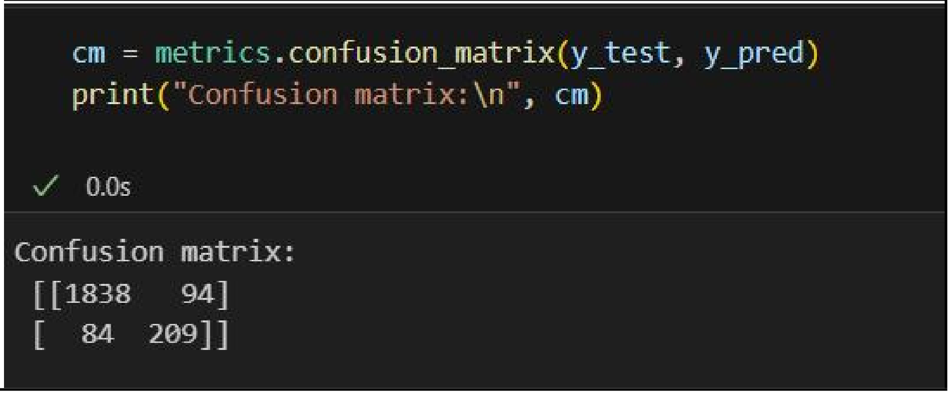
Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (6 Marks):

Model	Tuned Hyperparameters	Optimal Values
Linear regression	---	----
Random Forest	---	----
Decision tree	---	----

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric
Linear regression	 

```
print("Classification report:\n", metrics.classification_report(y_test, y_pred))
```

✓ 0.0s

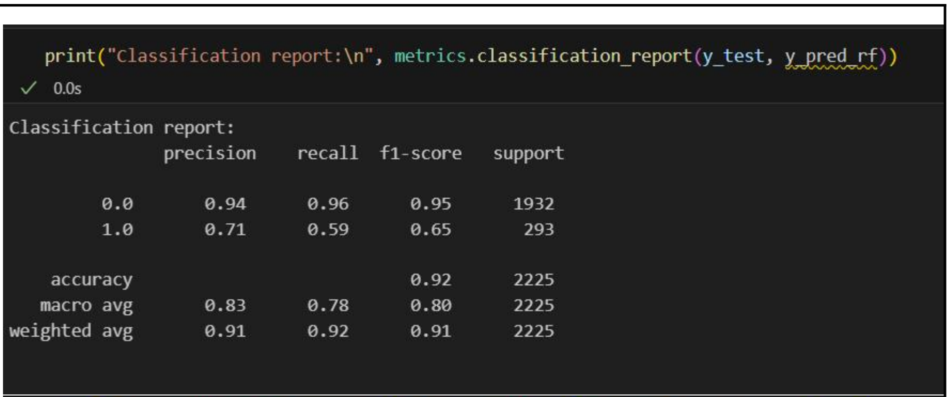
```
Classification report:
               precision    recall  f1-score   support

         0.0       0.96      0.95      0.95      1932
         1.0       0.69      0.71      0.70       293

    accuracy                           0.92      2225
   macro avg       0.82      0.83      0.83      2225
weighted avg       0.92      0.92      0.92      2225
```

```
cm = metrics.confusion_matrix(y_test, y_pred)
print("Confusion matrix:\n", cm)
```

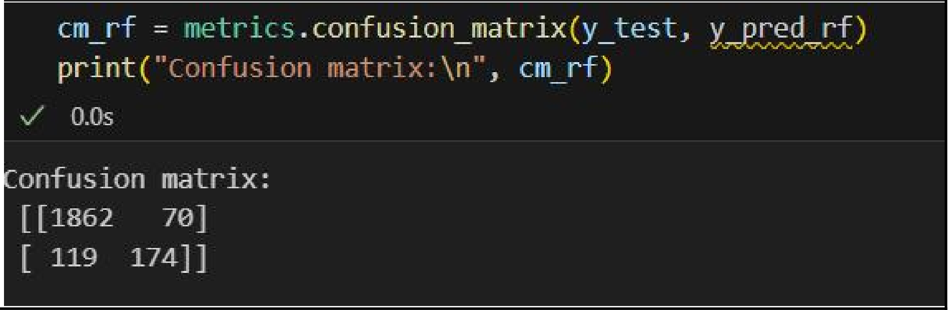
✓ 0.0s

```
Confusion matrix:
 [[1838   94]
 [  84  209]]
```

<table>
<tr>
<td>Random Forest</td>
<td>

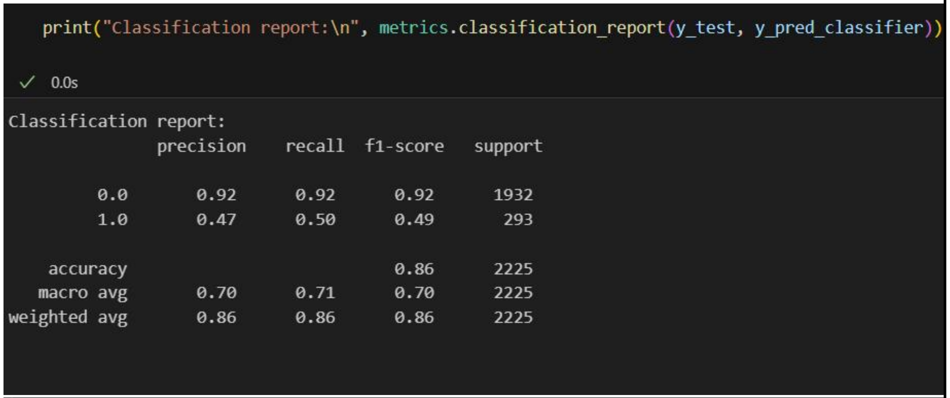


```
print("Classification report:\n", metrics.classification_report(y_test, y_pred_rf))
✓ 0.0s

Classification report:
               precision    recall  f1-score   support

         0.0       0.94      0.96      0.95      1932
         1.0       0.71      0.59      0.65       293

    accuracy                           0.92      2225
   macro avg       0.83      0.78      0.80      2225
weighted avg       0.91      0.92      0.91      2225
```

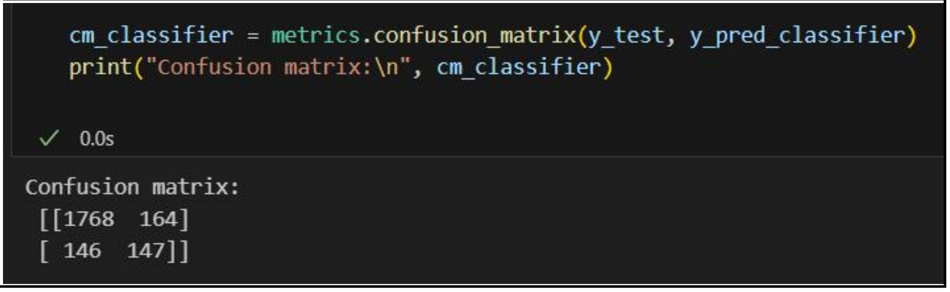


```
cm_rf = metrics.confusion_matrix(y_test, y_pred_rf)
print("Confusion matrix:\n", cm_rf)
✓ 0.0s

Confusion matrix:
 [[1862   70]
 [ 119  174]]
```

</td>
</tr>
<tr>
<td>Decision tree</td>
<td>



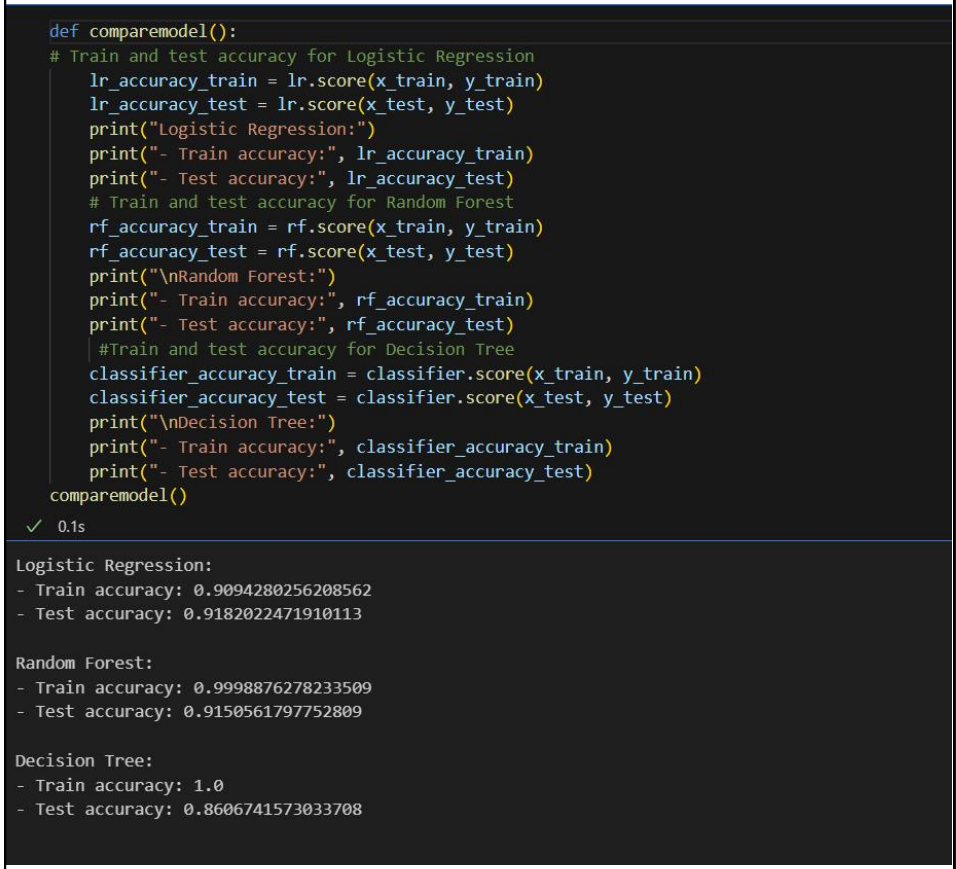
```
print("Classification report:\n", metrics.classification_report(y_test, y_pred_classifier))

✓ 0.0s

Classification report:
               precision    recall  f1-score   support

         0.0       0.92      0.92      0.92      1932
         1.0       0.47      0.50      0.49       293

    accuracy                           0.86      2225
   macro avg       0.70      0.71      0.70      2225
weighted avg       0.86      0.86      0.86      2225
```



```
cm_classifier = metrics.confusion_matrix(y_test, y_pred_classifier)
print("Confusion matrix:\n", cm_classifier)

✓ 0.0s

Confusion matrix:
 [[1768  164]
 [ 146  147]]
```

</td>
</tr>
</table>

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random Forest	 The Random Forest model was selected for its superior performance, exhibiting high accuracy during train and test. Its often more accurate than decision tree it builds multiple tree and averages their predictions,reducing the risk of overfitting.It can model non-linear relationships better than Linear Regression.Effective in detecting anomalies in datasets,useful in fraud detection and network security.

```
def comparemodel():
# Train and test accuracy for Logistic Regression
    lr_accuracy_train = lr.score(x_train, y_train)
    lr_accuracy_test = lr.score(x_test, y_test)
    print("Logistic Regression:")
    print("- Train accuracy:", lr_accuracy_train)
    print("- Test accuracy:", lr_accuracy_test)
    # Train and test accuracy for Random Forest
    rf_accuracy_train = rf.score(x_train, y_train)
    rf_accuracy_test = rf.score(x_test, y_test)
    print("\nRandom Forest:")
    print("- Train accuracy:", rf_accuracy_train)
    print("- Test accuracy:", rf_accuracy_test)
     #Train and test accuracy for Decision Tree
    classifier_accuracy_train = classifier.score(x_train, y_train)
    classifier_accuracy_test = classifier.score(x_test, y_test)
    print("\nDecision Tree:")
    print("- Train accuracy:", classifier_accuracy_train)
    print("- Test accuracy:", classifier_accuracy_test)
comparemodel()
```

✓ 0.1s

```
Logistic Regression:
- Train accuracy: 0.9094280256208562
- Test accuracy: 0.9182022471910113

Random Forest:
- Train accuracy: 0.9998876278233509
- Test accuracy: 0.9150561797752809

Decision Tree:
- Train accuracy: 1.0
- Test accuracy: 0.8606741573033708
```