Model Optimization and Tuning Phase Report

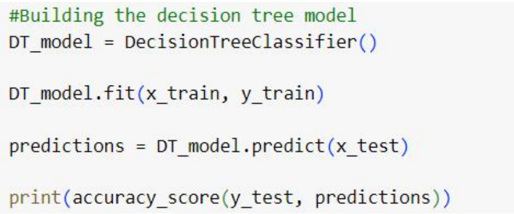
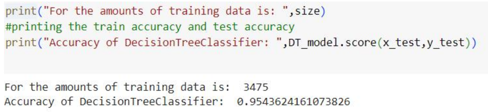
Date	05 May 2024
Team ID	Team - 737850
Project Title	FetalAI: Using Machine Learning To Predict And Monitor Fetal Health
Maximum Marks	10 Marks

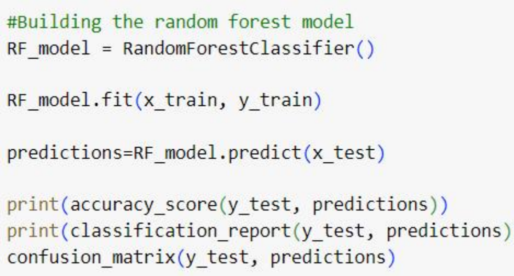
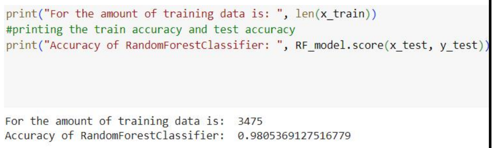
Model Optimization and Tuning Phase

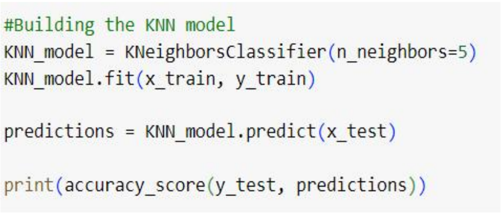
The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (6 Marks):

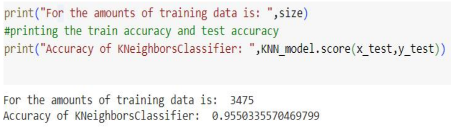
Model	Tuned Hyperparameters	Optimal Values

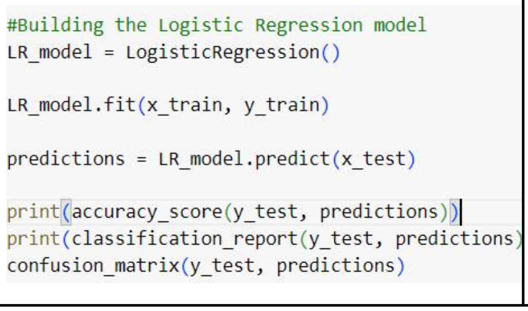
Decision Tree	
	`#Building the decision tree model`
	`DT_model = DecisionTreeClassifier()`
	`DT_model.fit(x_train, y_train)`
	`predictions = DT_model.predict(x_test)`
	`print(accuracy_score(y_test, predictions))`
		
		`print("For the amounts of training data is: ",size)`
		`#printing the train accuracy and test accuracy`
		`print("Accuracy of DecisionTreeClassifier: ",DT_model.score(x_test,y_test))`
		`For the amounts of training data is:  3475`
		`Accuracy of DecisionTreeClassifier:  0.9543624161073826`

Random
Forest
	
	`#Building the random forest model`
	`RF_model = RandomForestClassifier()`
	`RF_model.fit(x_train, y_train)`
	`predictions=RF_model.predict(x_test)`
	`print(accuracy_score(y_test, predictions))`
	`print(classification_report(y_test, predictions))`
	`confusion_matrix(y_test, predictions)`
		
		`print("For the amount of training data is: ", len(x_train))`
		`#printing the train accuracy and test accuracy`
		`print("Accuracy of RandomForestClassifier: ", RF_model.score(x_test, y_test))`
		`For the amount of training data is:  3475`
		`Accuracy of RandomForestClassifier:  0.9805369127516779`

<table>
<tr>
<td>KNN</td>
<td>
<pre>
#Building the KNN model
KNN_model = KNeighborsClassifier(n_neighbors=5)
KNN_model.fit(x_train, y_train)

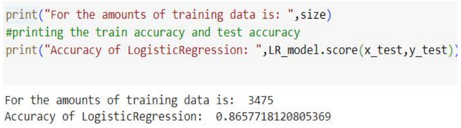
predictions = KNN_model.predict(x_test)

print(accuracy_score(y_test, predictions))
</pre></td>
<td>
<pre>
print("For the amounts of training data is: ",size)
#printing the train accuracy and test accuracy
print("Accuracy of KNeighborsClassifier: ",KNN_model.score(x_test,y_test))

For the amounts of training data is: 3475
Accuracy of KNeighborsClassifier: 0.9550335570469799
</pre></td>
</tr>
<tr>
<td>Logistic Regressi-on</td>
<td>
<pre>
#Building the Logistic Regression model
LR_model = LogisticRegression()

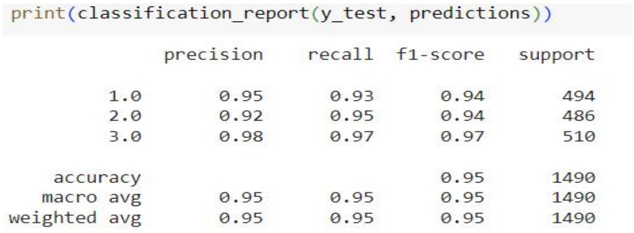
LR_model.fit(x_train, y_train)

predictions = LR_model.predict(x_test)

print(accuracy_score(y_test, predictions))
print(classification_report(y_test, predictions)
confusion_matrix(y_test, predictions)
</pre></td>
<td>
<pre>
print("For the amounts of training data is: ",size)
#printing the train accuracy and test accuracy
print("Accuracy of LogisticRegression: ",LR_model.score(x_test,y_test))

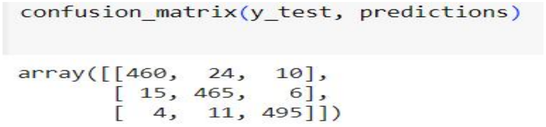
For the amounts of training data is: 3475
Accuracy of LogisticRegression: 0.8657718120805369
</pre></td>
</tr>
</table>

Performance Metrics Comparison Report (2 Marks):

<table>
<tr>
<th>Model</th>
<th>Optimized Metric</th>
</tr>
<tr>
<td>Decision Tree</td>
<td>
<pre>
print(classification_report(y_test, predictions))

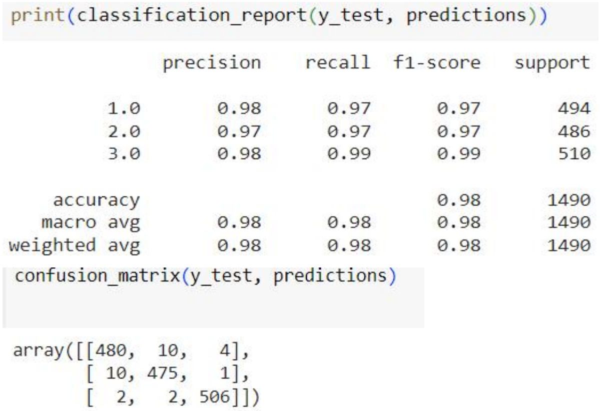
              precision   recall f1-score   support

         1.0   0.95   0.93   0.94   494
         2.0   0.92   0.95   0.94   486
         3.0   0.98   0.97   0.97   510

    accuracy   0.95   1490
   macro avg   0.95   0.95   0.95   1490
weighted avg   0.95   0.95   0.95   1490
</pre>

<pre>
confusion_matrix(y_test, predictions)

array([[460, 24, 10],
       [15, 465,   6],
       [4, 11, 495]])
</pre></td>
</tr>
</table>

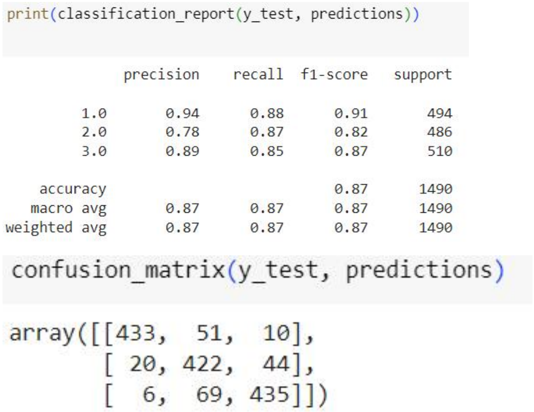
Random Forest



```
print(classification_report(y_test, predictions))

              precision    recall  f1-score   support

         1.0       0.98      0.97      0.97       494
         2.0       0.97      0.97      0.97       486
         3.0       0.98      0.99      0.99       510

    accuracy                           0.98      1490
   macro avg       0.98      0.98      0.98      1490
weighted avg       0.98      0.98      0.98      1490
```

```
confusion_matrix(y_test, predictions)

array([[480,  10,   4],
       [ 10, 475,   1],
       [  2,   2, 506]])
```

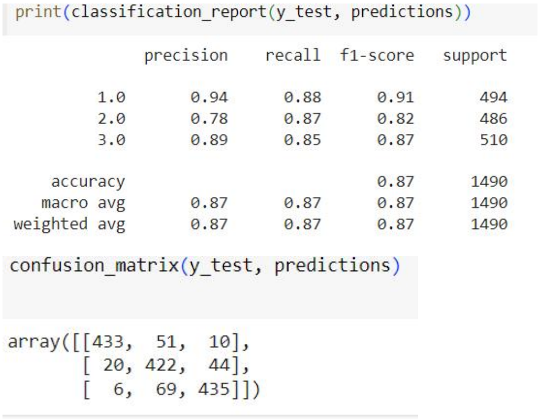
KNN



```
print(classification_report(y_test, predictions))

              precision    recall  f1-score   support

         1.0       0.94      0.88      0.91       494
         2.0       0.78      0.87      0.82       486
         3.0       0.89      0.85      0.87       510

    accuracy                           0.87      1490
   macro avg       0.87      0.87      0.87      1490
weighted avg       0.87      0.87      0.87      1490
```

```
confusion_matrix(y_test, predictions)

array([[433,  51,  10],
       [ 20, 422,  44],
       [  6,  69, 435]])
```

Logistic Regression



```
print(classification_report(y_test, predictions))

              precision    recall  f1-score   support

         1.0       0.94      0.88      0.91       494
         2.0       0.78      0.87      0.82       486
         3.0       0.89      0.85      0.87       510

    accuracy                           0.87      1490
   macro avg       0.87      0.87      0.87      1490
weighted avg       0.87      0.87      0.87      1490
```

```
confusion_matrix(y_test, predictions)

array([[433,  51,  10],
       [ 20, 422,  44],
       [  6,  69, 435]])
```

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Logistic Regression	Random Forest model is selected for fetal AI due to its ability to handle high-dimensional data, deal with non-linear relationships, and mitigate overfitting, thus providing robust and accurate predictions for fetal health assessments.