Model Development Phase Template

Date	8 July 2024
Team ID	739876
Project Title	FetalAI: Using Machine Learning To Predict And Monitor Fetal Health
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:

Random Forest:

Model Building

Random Forest

```
[ ] Rf_model = RandomForestClassifier()
    Rf_model.fit(x_train_smote, y_train_smote)
    Rf_model_pred = Rf_model.predict(x_test)
    print(accuracy_score(y_test, Rf_model_pred))
```

```
0.9435736677115988
```

Decision Tree:

Decision Tree

```
[ ] DT_model = DecisionTreeClassifier()
    DT_model.fit(x_train_smote, y_train_smote)
    DT_model_pred = DT_model.predict(x_test)
    print(accuracy_score(y_test, DT_model_pred))
```

```
0.9216300940438872
```

Logistic Regression:

Logistic Regression

```
[ ] LR_model = LogisticRegression()
    LR_model.fit(x_train_smote, y_train_smote)
    LR_model_pred = LR_model.predict(x_test)
    print(accuracy_score(y_test, LR_model_pred))
```

```
0.8605015673981191
```

K- Nearest Neighbors:

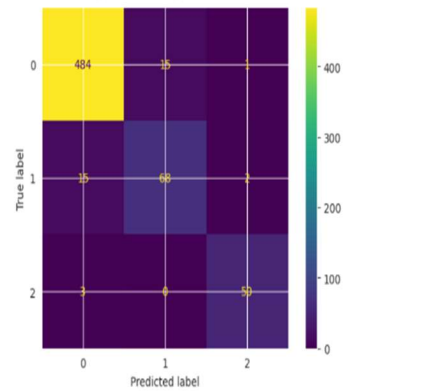
K-Nearest Neighbors

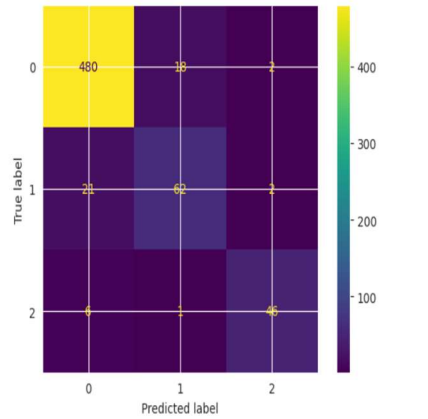
```
KNN_model = KNeighborsClassifier()
KNN_model.fit(x_train_smote, y_train_smote)
KNN_model_pred = KNN_model.predict(x_test)
print(accuracy_score(y_test, KNN_model_pred))
```

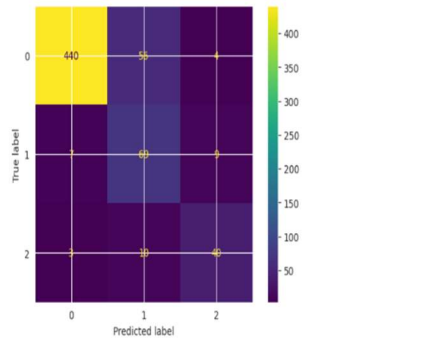
```
0.9153605015673981
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix

Random Forest	Model Building
	Random Forest
	`Rf_model = RandomForestClassifier()`
	`Rf_model.fit(x_train_smote, y_train_smote)`
	`Rf_model_pred = Rf_model.predict(x_test)`
	`print(accuracy_score(y_test, Rf_model_pred))`
	0.9435736677115988
		94%	`cm = confusion_matrix(y_test, Rf_model_pred)`
			`disp = ConfusionMatrixDisplay(confusion_matrix=cm)`
			`disp.plot()`
			<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7899dc8dd660>
			 

Decision Tree	Decision Tree
	`DT_model = DecisionTreeClassifier()`
	`DT_model.fit(x_train_smote, y_train_smote)`
	`DT_model_pred = DT_model.predict(x_test)`
	`print(accuracy_score(y_test, DT_model_pred))`
	0.9216300940438872
		92%	`cm = confusion_matrix(y_test, DT_model_pred)`
			`disp = ConfusionMatrixDisplay(confusion_matrix=cm)`
			`disp.plot()`
			<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7899dc8d
			 

Logistic Regression	Logistic Regression
	`LR_model = LogisticRegression()`
	`LR_model.fit(x_train_smote, y_train_smote)`
	`LR_model_pred = LR_model.predict(x_test)`
	`print(accuracy_score(y_test, LR_model_pred))`
	0.8605015673981191
		86%	`cm = confusion_matrix(y_test, LR_model_pred)`
			`disp = ConfusionMatrixDisplay(confusion_matrix=cm)`
			`disp.plot()`
			<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7899dceaa020>
			 

K - Nearest Neighbors	K-Nearest Neighbors
	`KNN_model = KNeighborsClassifier()`
	`KNN_model.fit(x_train_smote, y_train_smote)`
	`KNN_model_pred = KNN_model.predict(x_test)`
	`print(accuracy_score(y_test, KNN_model_pred))`
	`0.9153605015673981`
		91.5%	`cm = confusion_matrix(y_test, KNN_model_pred)`
			`disp = ConfusionMatrixDisplay(confusion_matrix=cm)`
			`disp.plot()`
			`<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7899dcf02dd0>`