

## Model Development Phase Template

Date	9 JULY 2024
Team ID	739661
Project Title	<b>Anemiasense: Leveraging Machine Learning For Precise Anemia Recognitions</b>
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:

```
from sklearn.model_selection import train_test_split
✓ 0.2s

x_train, x_test, y_train, y_test = train_test_split(X, Y , test_size=0.2, random_state=20)
✓ 0.0s
```

#### Model Validation and Evaluation Report:

Model	Classification Report	Accuracy
<b>Logistic Regression Model</b>	<pre>from sklearn.linear_model import LogisticRegression from sklearn.metrics import accuracy_score from sklearn.metrics import classification_report  logistic_regression = LogisticRegression() logistic_regression.fit(x_train, y_train) y_pred = logistic_regression.predict(x_test)  acc_lr = accuracy_score(y_test,y_pred) c_lr = classification_report(y_test,y_pred)  print('Accuracy Score: ',acc_lr) print(c_lr)</pre>	0.991

<b>Random forest model</b>	<pre> from sklearn.ensemble import RandomForestClassifier  random_forest = RandomForestClassifier() random_forest.fit(x_train, y_train) y_pred = random_forest.predict(x_test)  acc_rf = accuracy_score(y_test,y_pred) c_rf = classification_report(y_test,y_pred)  print('Accuracy Score: ',acc_rf) print(c_rf) </pre>	1.0
<b>Decision Tree Model</b>	<pre> from sklearn.tree import DecisionTreeClassifier  decision_tree_model = DecisionTreeClassifier() decision_tree_model.fit(x_train, y_train) y_pred = decision_tree_model.predict(x_test)  acc_dt = accuracy_score(y_test,y_pred) c_dt = classification_report(y_test,y_pred)  print('Accuracy Score: ',acc_dt) print(c_dt) </pre>	1.0
<b>Gaussian Navies Bayes</b>	<pre> from sklearn.naive_bayes import GaussianNB  NB = GaussianNB() NB.fit(x_train, y_train) y_pred = NB.predict(x_test)  acc_nb = accuracy_score(y_test,y_pred) c_nb = classification_report(y_test,y_pred)  print('Accuracy Score: ',acc_nb) print(c_nb) </pre>	0.979
<b>Gradient Boosting Classifier</b>	<pre> from sklearn.ensemble import GradientBoostingClassifier  GBC = GradientBoostingClassifier() GBC.fit(x_train, y_train) y_pred = GBC.predict(x_test)  acc_gbc = accuracy_score(y_test,y_pred) c_gbc = classification_report(y_test,y_pred)  print('Accuracy Score: ',acc_gbc) print(c_gbc) </pre>	1.0