Model Development Phase Template

Date	30 June 2024
Team ID	-
Project Title	Golden Harvesting: A Predictive Model For Apple Quality Assurance
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

Initial Model Training Code:

```
# Decision tree model
from sklearn.tree import DecisionTreeClassifier
model1 = DecisionTreeClassifier()
model1.fit(x_train, y_train)
dt_pred = model1.predict(x_test)
acc_score = accuracy_score(y_test, dt_pred)
print("Accuracy of Decision tree is %.2f"%accuracy_score(y_test, dt_pred))
print("Confusion matrix of Decision tree is as follows:")
confusion_matrix(y_test, dt_pred)
```

```
# Random forest model
from sklearn.ensemble import RandomForestClassifier
model2 = RandomForestClassifier(n_estimators=100)
model2.fit(x_train, y_train)
rf_pred = model2.predict(x_test)
acc_score = accuracy_score(y_test, rf_pred)
print("Accuracy of Random forest is %.2f"%accuracy_score(y_test, rf_pred))
print("Confusion matrix of Random forest is as follows:")
confusion_matrix(y_test, rf_pred)
```

```
# XGBoost model
import xgboost as xgb
model3 = xgb.XGBClassifier()
model3.fit(x_train, y_train)
xgb_pred = model3.predict(x_test)
acc_score = accuracy_score(y_test, xgb_pred)
print("Accuracy of XGBoost is %.2f"%accuracy_score(y_test,xgb_pred))
print("Confusion matrix of XGBoost is as follows:")
confusion_matrix(y_test, xgb_pred)
```

```
# Logistic Regression Model
from sklearn.linear_model import LogisticRegression
model4 = LogisticRegression()
model4.fit(x_train, y_train)
lr_pred = model4.predict(x_test)
acc_score = accuracy_score(y_test, lr_pred)
print("Accuracy of Logistic Regression is %.2f"%accuracy_score(y_test, lr_pred))
print("Confusion matrix of Logistic Regression is as follows:")
confusion_matrix(y_test, lr_pred)
```

```
# Support Vector Machine model
from sklearn.svm import SVC
model5 = SVC()
model5.fit(x_train,y_train)
svm_pred = model5.predict(x_test)
acc_score = accuracy_score(y_test, svm_pred)
print("Accuracy of Support Vector Machine is %.2f"%accuracy_score(y_test, svm_pred))
print("Confusion matrix of Support Vector Machine is as follows:")
confusion_matrix(y_test, svm_pred)
```

```
# K Nearest Neighbor model
from sklearn.neighbors import KNeighborsClassifier
model6 = KNeighborsClassifier()
model6.fit(x_train,y_train)
knn_pred = model6.predict(x_test)
acc_score = accuracy_score(y_test, knn_pred)
print("Accuracy of K Nearest Neighbor is %.2f"%accuracy_score(y_test, knn_pred))
print("Confusion matrix of K Nearest Neighbor is as follows:")
confusion_matrix(y_test, knn_pred)
```

```
# Naive Bayes model
from sklearn.naive_bayes import GaussianNB
model7 = GaussianNB()
model7.fit(x_train,y_train)
nb_pred = model7.predict(x_test)
acc_score = accuracy_score(y_test, nb_pred)
print("Accuracy of Naive Baye is %.2f"%accuracy_score(y_test, nb_pred))
print("Confusion matrix of Naive Baye is as follows:")
confusion_matrix(y_test, nb_pred)
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Decision tree model	`print(classification_report(dt_pred, y_test))` precision recall f1-score support 0 0.79 0.80 0.80 394 1 0.81 0.79 0.80 406 accuracy 0.80 800 macro avg 0.80 0.80 0.80 800 weighted avg 0.80 0.80 0.80 800	81%	Confusion matrix of Decision tree array([[317, 84], [77, 322]], dtype=int64)
Random forest model	`print(classification_report(rf_pred,y_test))` precision recall f1-score support 0 0.89 0.92 0.91 386 1 0.93 0.89 0.91 414 accuracy 0.91 800 macro avg 0.91 0.91 0.91 800 weighted avg 0.91 0.91 0.91 800	91%	Confusion matrix of Random forest array([[357, 44], [29, 370]], dtype=int64)
XGBoost model	`print(classification_report(xgb_pred, y_test))` precision recall f1-score support 0 0.91 0.91 0.91 400 1 0.91 0.91 0.91 400 accuracy 0.91 800 macro avg 0.91 0.91 0.91 800 weighted avg 0.91 0.91 0.91 800	91%	Confusion matrix of XGBoost array([[363, 38], [37, 362]], dtype=int64)

Logistic Regression Model	`print(classification_report(lr_pred,y_test))` precision recall f1-score support 0 0.76 0.75 0.75 402 1 0.75 0.75 0.75 398 accuracy 0.75 800 macro avg 0.75 0.75 0.75 800 weighted avg 0.75 0.75 0.75 800	75%	Confusion matrix of Logistic Regression array([[303, 98], [99, 300]], dtype=int64)
Support Vector Machine model	`print(classification_report(svm_pred,y_test))` precision recall f1-score support 0 0.91 0.90 0.90 402 1 0.90 0.90 0.90 398 accuracy 0.90 800 macro avg 0.90 0.90 0.90 800 weighted avg 0.90 0.90 0.90 800	90%	Confusion matrix of Support Vector Machine array([[363, 38], [39, 360]], dtype=int64)
K Nearest Neighbor model	`print(classification_report(knn_pred,y_test))` precision recall f1-score support 0 0.90 0.90 0.90 398 1 0.90 0.90 0.90 402 accuracy 0.90 800 macro avg 0.90 0.90 0.90 800 weighted avg 0.90 0.90 0.90 800	90%	Confusion matrix of K Nearest Neighbor array([[360, 41], [38, 361]], dtype=int64)
Naive Bayes model	`print(classification_report(nb_pred,y_test))` precision recall f1-score support 0 0.78 0.76 0.77 413 1 0.75 0.77 0.76 387 accuracy 0.76 800 macro avg 0.76 0.76 0.76 800 weighted avg 0.76 0.76 0.76 800	76%	Confusion matrix of Naive Baye array([[312, 89], [101, 298]], dtype=int64)