Model Development Phase Template

Date	04 June 2024
Team ID	SWTID1720096620
Project Title	E-commerce Shipping Prediction Using Machine Learning
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, GridSearchCV
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.preprocessing import LabelEncoder, StandardScaler
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
```

Random Forest

```
from sklearn.ensemble import RandomForestClassifier
```

```
rf = RandomForestClassifier(n_estimators=200, criterion='entropy', random_state=56,max_depth=5)
rf.fit(x_train, y_train)
pred = rf.predict(x_test)
accuracy = accuracy_score(y_test, pred)

print(f"Accuracy without Hyperparameter Tuning and SMOTE: {accuracy:.6f}")
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, pred))
```

K Nearest Neighbors (KNN)

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = KNeighborsClassifier(n_neighbors=10, weights='uniform', metric='minkowski', p=3)
knn.fit(x_train, y_train)
knn_pred = knn.predict(x_test)

print("Accuracy without Hyperparameter Tuning and SMOTE:", accuracy_score(y_test, knn_pred))
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, knn_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, knn_pred))
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
```

```
log_reg = LogisticRegression(solver='sag', penalty='l2', random_state=42)
log_reg.fit(x_train, y_train)
log_reg_pred = log_reg.predict(x_test)

print("Accuracy without Hyperparameter Tuning and SMOTE:", accuracy_score(y_test, log_reg_pred))
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, log_reg_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, log_reg_pred))
```

XGBoost

```
from xgboost import XGBClassifier
```

```
xgb = XGBClassifier(eval_metric='mlogloss', random_state=42)
xgb.fit(x_train, y_train)
xgb_pred = xgb.predict(x_test)
accuracy = accuracy_score(y_test, xgb_pred)

print(f"Accuracy without Hyperparameter Tuning and SMOTE: {accuracy:.6f}")
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, xgb_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, xgb_pred))
```

SVC

```
from sklearn.svm import SVC
```

```
svc = SVC(kernel='rbf', random_state=42)
svc.fit(x_train, y_train)
svc_pred = svc.predict(x_test)
accuracy = accuracy_score(y_test, svc_pred)

print(f"Accuracy without Hyperparameter Tuning and SMOTE: {accuracy:.6f}")
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, svc_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, svc_pred))
```

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
```

```
dt = DecisionTreeClassifier(random_state=42)
dt.fit(x_train, y_train)
dt_pred = dt.predict(x_test)
accuracy = accuracy_score(y_test, dt_pred)

print(f"Accuracy without Hyperparameter Tuning and SMOTE: {accuracy:.6f}")
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, dt_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, dt_pred))
```

Naive Bayes

```
from sklearn.naive_bayes import GaussianNB
```

```
nb = GaussianNB()
nb.fit(x_train, y_train)
nb_pred = nb.predict(x_test)
accuracy = accuracy_score(y_test, nb_pred)

print(f"Accuracy without Hyperparameter Tuning and SMOTE: {accuracy:.6f}")
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, nb_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, nb_pred))
```

AdaBoost

```
from sklearn.ensemble import AdaBoostClassifier
```

```
ada = AdaBoostClassifier(random_state=42)
ada.fit(x_train, y_train)
ada_pred = ada.predict(x_test)
accuracy = accuracy_score(y_test, ada_pred)

print(f"Accuracy without Hyperparameter Tuning and SMOTE: {accuracy:.6f}")
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, ada_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, ada_pred))
```

Gradient Boosting

```
from sklearn.ensemble import GradientBoostingClassifier
```

```
gb = GradientBoostingClassifier(random_state=42)
gb.fit(x_train, y_train)
gb_pred = gb.predict(x_test)
accuracy = accuracy_score(y_test, gb_pred)

print(f"Accuracy without Hyperparameter Tuning and SMOTE: {accuracy:.6f}")
print("Classification Report without Hyperparameter Tuning and SMOTE:\n", classification_report(y_test, gb_pred))
print("Confusion Matrix without Hyperparameter Tuning and SMOTE:\n", confusion_matrix(y_test, gb_pred))
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Random Forest Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.58 0.91 0.71 1379 1 0.89 0.53 0.67 1921 accuracy 0.69 3300 macro avg 0.74 0.72 0.69 3300 weighted avg 0.76 0.69 0.69 3300	0.690000	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[1250 129] [894 1027]]
K-Nearest Neighbors Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.58 0.75 0.65 1379 1 0.77 0.61 0.68 1921 accuracy 0.67 3300 macro avg 0.68 0.68 0.67 3300 weighted avg 0.69 0.67 0.67 3300	0.666969	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[1035 344] [755 1166]]
Logistic Regression	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.56 0.53 0.54 1379 1 0.67 0.70 0.69 1921 accuracy 0.63 3300 macro avg 0.62 0.61 0.61 3300 weighted avg 0.63 0.63 0.63 3300	0.628484	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[725 654] [572 1349]]

XGB Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.57 0.62 0.59 1379 1 0.71 0.66 0.68 1921 accuracy 0.64 3300 macro avg 0.64 0.64 0.64 3300 weighted avg 0.65 0.64 0.65 3300	0.644545	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[853 526] [647 1274]]
Support Vector Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.57 0.82 0.67 1379 1 0.81 0.56 0.66 1921 accuracy 0.67 3300 macro avg 0.69 0.69 0.67 3300 weighted avg 0.71 0.67 0.67 3300	0.668788	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[1129 250] [843 1078]]
Decision Tree Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.58 0.57 0.58 1379 1 0.70 0.70 0.70 1921 accuracy 0.65 3300 macro avg 0.64 0.64 0.64 3300 weighted avg 0.65 0.65 0.65 3300	0.647576	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[787 592] [571 1350]]
Naive Bayes Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.55 0.78 0.65 1379 1 0.78 0.55 0.64 1921 accuracy 0.65 3300 macro avg 0.67 0.67 0.65 3300 weighted avg 0.68 0.65 0.65 3300	0.646364	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[1079 300] [867 1054]]
Ada Boost Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.59 0.75 0.66 1379 1 0.78 0.63 0.69 1921 accuracy 0.68 3300 macro avg 0.69 0.69 0.68 3300 weighted avg 0.70 0.68 0.68 3300	0.679394	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[1039 340] [718 1203]]
Gradient Boost Classifier	Classification Report without Hyperparameter Tuning and SMOTE: precision recall f1-score support 0 0.58 0.87 0.70 1379 1 0.85 0.55 0.67 1921 accuracy 0.68 3300 macro avg 0.72 0.71 0.68 3300 weighted avg 0.74 0.68 0.68 3300	0.683939	Confusion Matrix without Hyperparameter Tuning and SMOTE: [[1194 185] [858 1063]]