

## Model Development Phase Template

|               |  |
|---------------|--|
| Date          | 06 JUNE 2024   |
| Team ID       | SWTID1720260935                                      |
| Project Title | Ecommerce 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:**

#### **Supportvectormachine**

```
svm_model = svm.SVC(gamma='auto',C=5,kernel='rbf')
svm_model.fit(X_train,y_train)
y_pred = svm_model.predict(X_test)
```

#### **Randomforestclassifier**

```
params = {'n_estimators':[100,150], 'criterion':['gini', 'entropy']}
#Hyper parameter tuning
rf_model =GridSearchCV(estimator=RandomForestClassifier(),param_grid=params,scoring='accuracy', cv=5)
rf_model = rf_model.fit(X_train,y_train)
y_pred=rf_model.predict(X_test)
```

### Artificialneutalnetwork

```
[39]: predictions = (ann.predict(X_test) > 0.5)
      print(classification_report(y_test,predictions))
```

### Logisticregression:

```
lg = LogisticRegression(n_jobs=-1,random_state=1234)
lg_param_grid={
    'C':[6,8,10,15,20],
    'max_iter':[60,80,100]
}
lg_cv = GridSearchCV(lg,lg_param_grid,cv=5,scoring="accuracy",n_jobs=-1,verbose=3)
lg_cv.fit(X_train,y_train)
y_pred=lg_cv.predict(X_test)
```

### XG BOOST CLASSIFIER:

```
xgb = xgb.XGBClassifier(random_state=1234)
xgb.fit(X_train,y_train)
y_pred = xgb.predict(X_test)
```

### K- Nearest neighbours classifier:

```
from sklearn.neighbors import KNeighborsClassifier

knn_model = KNeighborsClassifier()
knn_model.fit(X_train, y_train)
y_pred = knn_model.predict(X_test)
print("K-Nearest Neighbors (KNN) Classifier:")
```

## Model Validation and Evaluation Report:

| Model                     | Classification Report  | Accuracy | Confusion Matrix |
|---------------------------|--|----------|------------------|
| Support Vector Machine    | <pre>y_pred = svm_model.predict(X_test) print(classification_report(y_test,y_pred))</pre> <pre> precision    recall  f1-score   support  0       0.55       0.85       0.67       895 1       0.83       0.53       0.65      1305  accuracy:    0.69       0.69       0.66      2200 macro avg:    0.69       0.69       0.66      2200 weighted avg: 0.72       0.66       0.66      2200</pre>  | 66%      |                  |
| Artificial Neural Network | <pre>[10]: predictions = (ann.predict(X_test) &gt; 0.5) print(classification_report(y_test,predictions))</pre> <pre> 09/09 ----- 0e 3ms/step precision    recall  f1-score   support  0       0.56       0.82       0.67       895 1       0.82       0.56       0.67      1305  accuracy:    0.69       0.69       0.67      2200 macro avg:    0.69       0.69       0.67      2200 weighted avg: 0.71       0.67       0.67      2200</pre>       | 67%      |                  |
| Logistic Regression       | <pre>print(classification_report(y_test,y_pred))</pre> <pre> fitting 5 folds for each of 15 candidates, totalling 75 fits precision    recall  f1-score   support  0       0.56       0.57       0.57       895 1       0.78       0.69       0.70      1305  accuracy:    0.63       0.63       0.64      2200 macro avg:    0.63       0.63       0.64      2200 weighted avg: 0.64       0.64       0.64      2200</pre>                          | 64%      |                  |
| Random Forest Classifier  | <pre>print(classification_report(y_test,y_pred))</pre> <pre> precision    recall  f1-score   support  0       0.57       0.85       0.68       895 1       0.75       0.60       0.67      1305  accuracy:    0.69       0.69       0.66      2200 macro avg:    0.65       0.65       0.65      2200 weighted avg: 0.67       0.66       0.66      2200</pre>   | 66%      |                  |
| XGBoost Classifier        | <pre>print(classification_report(y_test,y_pred))</pre> <pre> precision    recall  f1-score   support  0       0.57       0.65       0.61       895 1       0.73       0.67       0.70      1305  accuracy:    0.65       0.65       0.66      2200 macro avg:    0.65       0.65       0.65      2200 weighted avg: 0.67       0.66       0.66      2200</pre>   | 66%      |                  |
| KNN Neighbours Classifier | <pre>print("K-Nearest Neighbors (KNN) Classifier:") print(classification_report(y_test, y_pred))</pre> <pre> K-Nearest Neighbors (KNN) Classifier: precision    recall  f1-score   support  0       0.56       0.61       0.58       895 1       0.72       0.67       0.69      1305  accuracy:    0.65       0.65       0.65      2200 macro avg:    0.64       0.64       0.64      2200 weighted avg: 0.65       0.65       0.65      2200</pre> | 65%      |                  |