

Model Development Phase Template

Date	20 July 2024
Team ID	SWTID1720108903
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:

```
def models_eval_mm(x_train,y_train, x_test,y_test):
    lg=LogisticRegression (random_state=1234)
    lg.fit(x_train, y_train)
    print('--Logistic Regression')
    print('Train Score:', lg.score(x_train, y_train))
    print('Test Score:',lg.score(x_test,y_test))
    print()
    lcv= LogisticRegressionCV (random_state=1234)
    lcv.fit(x_train,y_train)
    print("--Logistic Regression CV")
    print("Train Score:",lcv.score(x_train,y_train))
    print()
    print('Test Score:',lcv.score(x_test,y_test))
    print('--XGBoost')
    xgb = XGBClassifier(random_state=1234)
    xgb.fit(x_train,y_train)
    print('Train Score:', xgb.score(x_train,y_train))
    print('Test Score:xgb',xgb.score(x_test,y_test))
    print()
    print('--Ridge Classifier')
    rg = RidgeClassifier(random_state=1234)
    rg.fit(x_train,y_train)
    print('Train Score:', rg.score(x_train, y_train))
    print('Test Score:',rg.score(x_test,y_test))
    print()
    print('--KNN')
    knn = KNeighborsClassifier()
    knn.fit(x_train,y_train)
    print('Train Score:',knn.score(x_train,y_train))
    print('Test Score:',knn.score(x_test,y_test))
    print()
    print('--Random Forest')
    rf = RandomForestClassifier(random_state=1234)
    rf.fit(x_train,y_train)
    print('Train Score:', rf.score(x_train,y_train))
    print("Test Score:",rf.score(x_test,y_test))
    print()
    print('--SVM classifier')
    svc = svm.SVC(random_state=1234)
    svc.fit(x_train,y_train)
    print("Train Score:", svc.score(x_train,y_train))
    print("Test Score:",svc.score(x_test,y_test))
    print()
    return lg,lcv, xgb, rg, knn, rf, svc
```

```
def eval(name, model):
    y_pred=model.predict(x_test_normalized)
    result =[]
    result.append(name)
    result.append("{:.2f}".format(accuracy_score(y_test, y_pred)*100))
    result.append("{:.2f}".format(f1_score(y_test, y_pred)*100))
    result.append("{:.2f}".format(recall_score(y_test, y_pred)*100))
    result.append("{:.2f}".format(precision_score(y_test, y_pred)*100))
    return result

model_list={'logistic regression': lg,
            'logistic regression CV':lcv,
            'XGBoost':xgb,
            'Ridge classifier':rg,
            "KNN":knn,
            "Random Forest":rf,
            "Support Vector Classifier":svc}

model_eval_info=[]
for i in model_list.keys():
    model_eval_info.append(eval(i,model_list[i]))
model_eval_info_df = pd.DataFrame(model_eval_info, columns=['Name', 'Accuracy', 'F1_Score', 'Recall', 'Precision'])

model_eval_info_df.to_csv("model_eval.csv", index=False)
from IPython.display import display, HTML

display(HTML(model_eval_info_df.to_html(index=False)))
```

Name	Accuracy	F1_Score	Recall	Precision
logistic regression	59.27	74.43	100.00	59.27
logistic regression CV	64.09	67.08	61.73	73.45
XGBoost	66.82	71.03	68.63	73.60
Ridge classifier	59.27	74.43	100.00	59.27
KNN	63.36	68.27	66.49	70.15
Random Forest	66.86	69.64	64.11	76.21
Support Vector Classifier	59.27	74.43	100.00	59.27