

Model Development Phase Template

| | |
|---------------|--|
| Date | 10-july-2024 |
| Team ID | 739969 |
| Project Title | Walmart Sales Analysis For Retail Industry with 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:

```
#importing and building the random forest model
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor(n_estimators=100,max_depth=30,min_samples_split=5,min_samples_leaf=5)
rf.fit(X_train,Y_train.ravel())
print('Testing Accuracy:',rf.score(X_test,Y_test.ravel())*100,'%')
Y_pred=rf.predict(X_test)
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
rms=mean_squared_error(Y_test,Y_pred,squared=False)
print('RMSE:',rms)
print('MAE:',mean_absolute_error(Y_test,Y_pred))
print("Training Accuracy:",rf.score(X_train,Y_train.ravel())*100,'%')
```

```
#importing and building the Decision tree model
from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor(random_state=0)
dt.fit(X_train,Y_train)
Y_pred=dt.predict(X_test)
print('Testing Accuracy:',dt.score(X_test,Y_test.ravel())*100,'%')
from sklearn import metrics
rms=mean_squared_error(Y_test,Y_pred,squared=False)
print('RMSE:',rms)
print('MAE:',mean_absolute_error(Y_test,Y_pred))
print("Training Accuracy:",rf.score(X_train,Y_train.ravel())*100,'%')
```





```
#importing and building the XGBoost
import xgboost as xgb
xg_reg=xgb.XGBRegressor(objective='reg:squarederror',nthread=4,n_estimators=500,max_depth=4,learning_rate=0.5)
xg_reg.fit(X_train,Y_train)
pred = xg_reg.predict(X_test)
Y_pred=xg_reg.predict(X_test)
print('Testing Accuracy:',dt.score(X_test,Y_test.ravel())*100,'%')
rms=mean_squared_error(Y_test,pred,squared=False)
print('RMSE:',rms)
print('MAE:',mean_absolute_error(Y_test,pred))
print("Training Accuracy:",rf.score(X_train,Y_train.ravel())*100,'%')
```

```
#importing and building arima
import pmdarima
from pmdarima import auto_arima
```

```
data9.Date=pd.to_datetime(data9.Date,format='%Y-%m-%d')
data9.index=data9.Date
data9=data9.drop('Date',axis=1)
data9=data9.resample('MS').mean()
train_data=data9[:int(0.7*(len(data9)))]
test_data=data9[int(0.7*(len(data9))):]
train_data=train_data['Weekly_Sales']
test_data=test_data['Weekly_Sales']
train_data.plot(figsize=(20,8),title='Weekly Sales',fontsize=14)
test_data.plot(figsize=(20,8),title='Weekly Sales',fontsize=14)
plt.show()
```

```
print('mean_squared_error(MSE) of ARIMA model:',mean_squared_error(test_data,forecast))
print('mean_absolute_error(MAE) of ARIMA model:',mean_absolute_error(test_data,forecast))
print('root_mean_squared_error(RMSE) of ARIMA model:',math.sqrt(mean_squared_error(test_data,forecast)))
```

Model Valuation and Evaluation Report

| Model | Regression Report | F1 Score |
|---------------|--|----------|
| Random Forest |  Testing Accuracy: 95.91432229869315 % RMSE: 1874.193759451107 MAE: 994.9535203649899 Training Accuracy: 97.66316734724609 % | 95% |
| Decision Tree |  Testing Accuracy: 93.44439739745214 % RMSE: 2374.0438960231972 MAE: 1230.2517804997842 Training Accuracy: 97.66316734724609 % | 93% |
| XGBoost |  Testing Accuracy: 93.44439739745214 % RMSE: 2495.706679534887 MAE: 1667.5961689900203 Training Accuracy: 97.66316734724609 % | 93% |
| ARIMA |  mean_squared_error(MSE) of ARIMA model: 192435.23031259677 mean_absolute_error(MAE) of ARIMA model: 346.47571328610394 root_mean_squared_error(RMSE) of ARIMA model: 438.6744012506278 | - |