Model Development Phase Template

Date	18 July 2024
Team ID	740111
Project Title	Unvieling Airbnb Price Patterns : Machine Learning Models For Forecasting
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:

Linear regression:

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn import metrics

# Assuming x_train_scaled, y_train, x_test_scaled, and y_test are already defined

# Initialize the Linear Regression model
lr = LinearRegression()

# Fit the model to the training data
lr.fit(x_train_scaled, y_train)

# Predict using the test data
y_pred_lr = lr.predict(x_test_scaled)

# Calculate evaluation metrics
mae_lr = metrics.mean_absolute_error(y_test, y_pred_lr)
mse_lr = metrics.mean_squared_error(y_test, y_pred_lr)
rmse_lr = np.sqrt(mse_lr)  # RMSE is already calculated correctly
r2_lr = metrics.r2_score(y_test, y_pred_lr)

# Print evaluation metrics
print('\nMean Absolute Error of Linear Regression:', mae_lr)
print('Mean Squared Error of Linear Regression:', mse_lr)
print('Root Mean Squared Error of Linear Regression:', rmse_lr)
print('R2 Score of Linear Regression:', r2_lr)
```

Random Forest Regressor

```
rf=RandomForestRegressor()
rf.fit(x_train_scaled,y_train)
y_pred_rf= rf.predict(x_test_scaled)
mae_rf= metrics.mean_absolute_error(y_test, y_pred_rf)
mse_rf= metrics.mean_squared_error(y_test, y_pred_rf)
rmse_rf =np.sqrt(metrics.mean_squared_error(y_test, y_pred_rf))
r2_rf= metrics.r2_score(y_test, y_pred_rf)
print('\nMean Absolute Error of Random Forest Regressor:',mae_rf)
print('\nMean Squarred Error of Random Forest Regressor :', mse_rf)
print('\nRoot Mean Squarred Error of Random Forest Regressor:', rmse_rf)
print('\nR2 Score of Random Forest Regressor :',r2_rf)
```

Polynomial Regression:

```
from sklearn.linear_model import Ridge
model=Pipeline([
('poly', PolynomialFeatures()),
('ridge', Ridge(fit_intercept=True))
])
param_grid ={
'poly__degree': [1, 2, 3],
'ridge__alpha': [0.1,0.5,1.0,2.0]
}
#Perform grid search with 5-fold cross-validation
poly_tuned=GridSearchCV(model, param_grid, cv=5)
#Traning and Testing
poly_tuned.fit(x_train_scaled, y_train)
y_pred_poly= poly_tuned.predict(x_test_scaled)
mae_poly= metrics.mean_absolute_error(y_test, y_pred_poly)
mse_poly= metrics.mean_squared_error(y_test, y_pred_poly)
rmse_poly= np.sqrt(metrics.mean_squared_error(y_test, y_pred_poly))
r2_poly= metrics.r2_score(y_test, y_pred_poly)
print('\nMean Absolute Error of Polynomial Regression:',mae_poly)
print('\nMean Squarred Error of Polynomial Regression:',mse_poly)
print('\nRoot Mean Squarred Error of Polynomial Regression:',rmse_poly)
print('\nR2 Score of Polynomial Regression :',r2_poly)
```

Gradient Boosting Regressor:

```
from sklearn.ensemble import GradientBoostingRegressor
from sklearn import metrics

# Assuming x_train_scaled, x_test_scaled, y_train, y_test are defined and properly scaled

# Initialize GradientBoostingRegressor
gb = GradientBoostingRegressor(n_estimators=200, learning_rate=0.2, max_depth=10)

# Fit the model
gb.fit(x_train_scaled, y_train)

# Predict using the model
y_pred_gb = gb.predict(x_test_scaled)

# Calculate evaluation metrics
mae_gb = metrics.mean_absolute_error(y_test, y_pred_gb)
mse_gb = metrics.mean_squared_error(y_test, y_pred_gb)  # Corrected variable name
rmse_gb = np.sqrt(metrics.mean_squared_error(y_test, y_pred_gb))
r2_gb = metrics.r2_score(y_test, y_pred_gb)

# Print the results
print('\nMean Absolute Error of Gradient Boosting:', mae_gb)
print('\nMean Squared Error of Gradient Boosting:', mse_gb)  # Corrected print statement
print('\nRoot Mean Squared Error of Gradient Boosting:', rmse_gb)
print('\nR2 Score of Gradient Boosting:', r2_gb)
```

Catboost Regressor:

```
from catboost import CatBoostRegressor
from sklearn.model_selection import cross_val_score, KFold
from sklearn import metrics
import numpy as np
model_CBR = CatBoostRegressor()
model_CBR.fit(x_train_scaled, y_train)
cross_val_score(model_CBR,x_train_scaled, y_train,
scoring='r2',
     cv=KFold(n_splits=5,
     shuffle=True,
     random_state=2022,
))
y_pred_cbr= model_CBR.predict(x_test_scaled)
mae_cbr =metrics.mean_absolute_error(y_test, y_pred_cbr)
mse_cbr =metrics.mean_squared_error(y_test, y_pred_cbr)
rmse_cbr = np.sqrt(metrics.mean_squared_error(y_test, y_pred_cbr))
r2_cbr =metrics.r2_score(y_test, y_pred_cbr)
print('\nMean Absolute Error of CatBoost Regressor :',mae_cbr)
print('\nMean Squarred Error of CatBoost Regressor:',mse_cbr)
print('\nRoot Mean Squarred Error of CatBoost Regressor:',rmse_cbr)
print('\nR2 Score of CatBoost Regressor:',r2_cbr)
```

:

XGBoost Rergressor:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import metrics
from xgboost import XGBRegressor
xgb = XGBRegressor(objective='reg:squarederror')
xgb.fit(x_train_scaled, y_train)
y_pred_xgb = xgb.predict(x_test_scaled)
mae_xgb = metrics.mean_absolute_error(y_test, y_pred_xgb)
mse_xgb = metrics.mean_squared_error(y_test, y_pred_xgb)
rmse_xgb = np.sqrt(mse_xgb)
r2_xgb = metrics.r2_score(y_test, y_pred_xgb)
print('\nMean Absolute Error of XGBoost Regressor:', mae_xgb)
print('\nMean Squared Error of XGBoost Regressor:', mse_xgb)
print('\nRoot Mean Squared Error of XGBoost Regressor:', rmse_xgb)
print('\nR2 Score of XGBoost Regressor:', r2_xgb)
```

Model	Classification Report	F1 Score	Confusion Matrix

Linerar regression	Mean Absolute Error of Linear Regression: 0.2496850406676611 Mean Squared Error of Linear Regression: 0.0831286214866343 Root Mean Squared Error of Linear Regression: 0.28832034525269684 R2 Score of Linear Regression: -0.00040043704639391997	-	-

Model Validation and Evaluation Report:

Forest regression	Mean Absolute Error of Random Forest Regressor: 0.25042532597042355 Mean Squarred Error of Random Forest Regressor : 0.0840184970084306 Root Mean Squarred Error of Random Forest Regressor: 0.2898594435384685 R2 Score of Random Forest Regressor : -0.011109526707709039	--	-
Polynomial regression	Mean Absolute Error of Polynomial Regression: 0.24968503845646894 Mean Squarred Error of Polynomial Regression: 0.08312861967771229 Root Mean Squarred Error of Polynomial Regression: 0.288320342115696 R2 Score of Polynomial Regression : -0.00040041527716039305	-	-
Gradient boosting	Mean Absolute Error of Gradient Boosting: 0.2570986980685457 Mean Squared Error of Gradient Boosting: 0.0904058531652557 Root Mean Squared Error of Gradient Boosting: 0.30067566107893684 R2 Score of Gradient Boosting: -0.08797732237885714	-	-

```
Mean Absolute Error of XGBoost Regressor: 0.2536936827472256

Mean Squared Error of XGBoost Regressor: 0.08700855364433946

Root Mean Squared Error of XGBoost Regressor: 0.2949721235037973

R2 Score of XGBoost Regressor: -0.04709296913538896
```

Catboost regessor: