Project Development Phase
Model Performance Test

Date	10 November 2023
Team ID	592456
Project Name	Project – T20 Totalitarian: Mastering Score Predictions
Maximum Marks	10 Marks

Model Performance Testing:

Metrics:

Linear Regression:

```
Linear Regression
Mean Absolute Error (MAE): 13.076449722754575
Mean Squared Error (MSE): 304.8531889483279
Root Mean Squared Error (RMSE): 17.460045502470145
R2 Score: 0.7104923538617827
```

Random Forest Regression:

```
Random Forest Regression
Mean Absolute Error (MAE): 1.7382122068139017
Mean Squared Error (MSE): 15.335606198995606
Root Mean Squared Error (RMSE): 3.916070249497014
R2 Score: 0.9854363496472185
```

XGB Regression:

```
XGB Regression
Mean Absolute Error (MAE): 1.50136112542292
Mean Squared Error (MSE): 8.896365431767773
Root Mean Squared Error (RMSE): 2.9826775608113882
R2 Score: 0.9915514552292479
```

Hyperparameter Tuning:

XGB Regressor with Hyperparameter:

```
from sklearn.model_selection import GridSearchCV
from xgboost import XGBRegressor
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('model', XGBRegressor(random_state=1))
])

param_grid = {
    'model__n_estimators': [500, 1000],
    'model__learning_rate': [0.1, 0.2],
    'model__max_depth': [6, 12]
}

grid_search = GridSearchCV(pipeline, param_grid, cv=5,
scoring='neg_mean_squared_error', n_jobs=-1)

grid_search.fit(X_train, y_train)

best_model = grid_search.best_estimator_

y_pred = best_model.predict(X_test)

mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)

print("XGB Regression (after hyperparameter tuning)")
print(f'Best Parameters: {grid_search.best_params_}')
print(f'Mean Absolute Error (MAE): {mae}')
print(f'Mean Squared Error (MSE): {mse}')
print(f'Root Mean Squared Error (RMSE): {rmse}')
print(f'R2 Score: {r2}')
```

```
XGB Regression (after hyperparameter tuning)
Best Parameters: {'model__learning_rate': 0.2, 'model__max_depth': 6, 'model__n_estimators': 1000}
Mean Absolute Error (MAE): 1.7997337492172327
Mean Squared Error (MSE): 8.687391135389662
Root Mean Squared Error (RMSE): 2.9474380630285792
R2 Score: 0.9915898668639289
```

XGB Regressor with validation set

```
from sklearn.model_selection import GridSearchCV, train_test_split
from xgboost import XGBRegressor
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=1)

pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('model', XGBRegressor(random_state=1))
])

param_grid = {
    'model__n_estimators': [500, 1000],
    'model__learning_rate': [0.1, 0.2],
    'model__max_depth': [6, 12]
}

X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2,
random_state=1)

# Create the GridSearchCV object with a validation set
grid_search = GridSearchCV(pipeline, param_grid, cv=5,
scoring='neg_mean_squared_error', n_jobs=-1)

# Fit the model using GridSearchCV with validation set
grid_search.fit(X_train, y_train, model__eval_metric="mae",
model__eval_set=[(X_val, y_val)], model__early_stopping_rounds=10,
model__verbose=False)

# Get the best estimator from the grid search
best_model = grid_search.best_estimator_

# Predictions using the best model
y_pred = best_model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)

# Print the results
print("XGB Regression (after hyperparameter tuning with validation set)")
print(f'Best Parameters: {grid_search.best_params_}')
print(f'Mean Absolute Error (MAE): {mae}')
print(f'Mean Squared Error (MSE): {mse}')
print(f'Root Mean Squared Error (RMSE): {rmse}')
print(f'R2 Score: {r2}')
```