```
#%% md
# Requirement: Calculate the probability of score, for example, what's the probability of score 3-1
#%% md
# Idea:
- Train 2 models to predict goals that home team can score, goals that away team can score
- Calculate probability of score 3 from home team (prob_home) and score 1 from away team (prob_away)
- Calculate the probability of score 3-1 by: final_prob = prob_home * prob_away
#%% md
# Set up
#%%
import pandas as pd
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np
from scipy.stats import poisson
#%% md
# Sourcing
#%%
df = pd.read_feather(path="../data/processed/regression/processed_train_data.feather")
#%%
df.head()
#%% md
# Train model to predict goals scored by home team
#%%
#Split the train and test data
X_home = df.drop(columns=['home_score','away_score'], axis=1)
y_home = df['home_score']
#%%
X_home_train, X_home_test, y_home_train, y_home_test = \
    train_test_split(X_home, y_home, test_size=0.2, random_state=42)
#%%
```

```
#Train the model
model = xgb.XGBRegressor()
model.fit(X_home_train.drop(columns='id', axis=1), y_home_train)
#%%
y_home_pred = model.predict(X_home_test.drop(columns='id', axis=1))
#%%
mse = mean_squared_error(y_home_test, y_home_pred)
rmse = mean_squared_error(y_home_test, y_home_pred, squared=False)
mae = mean_absolute_error(y_home_test, y_home_pred)

print("Mean Squared Error:", mse)
print("Root Mean Squared Error:", rmse)
print("Mean Absolute Error:", mae)
#%%
#Calculate probability of home team to score 3 goals each match
lambda_param = np.mean(y_home_train)
prob_3_list_home = []
for i, y_pred_i in enumerate(y_home_pred):
    prob_3 = poisson.pmf(3, mu=y_pred_i)
    prob_3_list_home.append(prob_3)
#%% md
# Train model to predict goals scored by away team
#%%
#Split the train and test data
X_away = df.drop(columns=['home_score','away_score'], axis=1)
y_away = df['away_score']
#%%
X_away_train, X_away_test, y_away_train, y_away_test = \
    train_test_split(X_away, y_away, test_size=0.2, random_state=42)
#%%
#Train the model
model = xgb.XGBRegressor()
model.fit(X_away_train.drop(columns='id', axis=1), y_away_train)
```

```
#%%
y_away_pred = model.predict(X_away_test.drop(columns='id', axis=1))
#%%
mse = mean_squared_error(y_away_test, y_home_pred)
rmse = mean_squared_error(y_away_test, y_home_pred, squared=False)
mae = mean_absolute_error(y_away_test, y_home_pred)

print("Mean Squared Error:", mse)
print("Root Mean Squared Error:", rmse)
print("Mean Absolute Error:", mae)
#%%
lambda_param = np.mean(y_away_train)
prob_1_list_away = []
for i, y_pred_i in enumerate(y_away_pred):
    prob_1 = poisson.pmf(1, mu=y_pred_i)
    prob_1_list_away.append(prob_1)
#%% md
# Calculate the probability of score 3-1
#%%
X_home_test['prob_3_home'] = prob_3_list_home
#%%
X_away_test['prob_1_away'] = prob_1_list_away
#%%
final_df = X_home_test.merge(right=X_away_test,
                             on='id',
                             how='inner',
                             )
#%%
final_df['prob_3_1_score'] = final_df['prob_3_home'] * final_df['prob_1_away']
#%% md
# Final prediction for the probability for score 3-1
#%%
final_df[['id','prob_3_home','prob_1_away','prob_3_1_score']].head()
#%%
final_df.to_csv("../data/predicted/question_3_result.csv")
#%%
```