App

July 9, 2025

1 Notebook for testing app code

[ ]:
```
import joblib
import pandas as pd
from fastapi import FastAPI
from pydantic import BaseModel

app = FastAPI()

class DataHandler:
    def __init__(self):
        self.data: pd.DataFrame = joblib.load("C:/Code/Git Repositories/
 ↪Football/Football/3_Data_Preparation/rawdata_clean.pkl")

    def team_query(self, home_team: str, away_team: str) -> pd.DataFrame:

        home_team_stats = self.data[self.data['home_team'] == home_team].
 ↪sort_values(by=['season', 'week'], ascending=False).iloc[0]
        home_team_stats = home_team_stats.loc['season':
 ↪'hlb3_interception_yards']
        away_team_stats = self.data[self.data['away_team'] == away_team].
 ↪sort_values(by=['season', 'week'], ascending=False).iloc[0]
        away_team_stats = away_team_stats.loc['aqb1':'alb3_interception_yards']

        combined_dict = {}

        # Prefix home stats
        for col, val in home_team_stats.items():
            combined_dict[f"{col}"] = val

        # Prefix away stats
        for col, val in away_team_stats.items():
            combined_dict[f"{col}"] = val

        # Create a single-row DataFrame with correct types
        combined_stats = pd.DataFrame([combined_dict])
```

```
        return combined_stats


    def preprocess_stats(self, game_stats: pd.DataFrame):

        game_stats_processed = game_stats.select_dtypes(include=['float64',␣
 ↪'int64'])

        return game_stats_processed


class ModelHandler:
    def __init__(self):
        self.lasso_pipe = joblib.load("C:/Code/Git Repositories/Football/
 ↪Football/4_5_Modeling_and_Evaluation/lasso_pipeline.pkl")

    def make_prediction(self, home_team: str, away_team: str):
        query = DataHandler()

        game_stats = query.team_query(home_team=home_team, away_team=away_team)
        game_stats_processed = query.preprocess_stats(game_stats)
        prediction = self.lasso_pipe.predict(X=game_stats_processed)
        return prediction[0]


# Define input data model for API
class Teams(BaseModel):
    home_team: str
    away_team: str

model = ModelHandler()

@app.post("/predict")
def predict_margin(teams: Teams):
    pred = model.make_prediction(teams.home_team, teams.away_team)

    if pred > 0:
        winner = teams.home_team
    else:
        winner = teams.away_team

    return {"predicted_margin": pred, "predicted_winner ": winner}



def main():
```

```
    model = ModelHandler()

    return model.make_prediction('DAL', 'CIN')

main()
```

[ ]: -0.767423328900033

[ ]: