Final Report

Introduction

In the NFL, a variety of factors affect athlete performance, and ultimately, who wins the game on a given day. For our project, we aim to isolate the effects of weather and stadium on how many points a team scores.

Existing literature outlines the correlation between stadium factors and weather on attendance, but we could not find a study describing the specific effects on performance. In a paper by Arboretti et al., ML was used to predict the attendance at Italian football club stadiums, using features such as weather and game time [4]. Sites such as FiveThirtyEight have generated NFL predictions in the past, using a variety of features including quarter back rating, home team advantage, and player rest to produce Monte Carlo simulations of the NFL season [3]. However, this site is no longer active.

For our case, we will utilize the NFL Team Stats Kaggle dataset, scraped from ESPN's team stats page, which provides useful information such as time and date of games as well as scores [1]. In particular, we'd like to somehow map time and date to factors like weather and map team to stadium, and see how those affect the resulting scores.

Problem Definition

The goal of this project is to build a machine learning model that can provide information related to a given NFL team's expected game performance. This model will provide valuable insight into factors such as home field advantage, playing surfaces, and the impact of weather conditions. This insight can be used by teams to make strategic decisions, analysts to provide insight and predictions, and fans for various forms of engagement.

Methods

Data Collection

For this project, we will be utilizing the NFL Team Stats dataset which was scraped from ESPN's team stats page [1]. This dataset contains useful information such as time and date of games as well as scores and much more that we will not be utilizing. Complementing this dataset, we will be utilizing the Open Mateo [2] historical weather API to grab the relevant weather conditions on average during each of the games, namely the precipitation, temperature, and wind.

Data Cleansing and Preprocessing

Given these two datasets, we then created a unified dataset containing all of the features we wanted, encoding each into a suitable format as inputs to the model as we went. These features were the stadium the game was played in, encoded using one-hot encoding, the encoded average temperature during the game, encoded using scalar encoding based on the quarter percentile the temperature fell in, the encoded average precipitation during the game, encoded into three bins based on the quantity of precipitation, the encoded average wind speed during the game, encoded the same way temperature was, and finally the average score of the home team over the past five games, and the average score of the away team over the past five games. The labels we used were the net score of the game which was found by summing the scores of the two teams for each game using the NFL Team Stats dataset. One hiccup in this process was the presence of domed stadiums, namely stadiums that are fully enclosed and therefore aren't affected by various weather conditions. For these stadiums, we simply filled in their weather conditions by the average conditions inside of the stadium, namely around 70 degrees for the temperature, no precipitation, and 0 wind speed. Finally, we normalized our data to ensure the difference in scale between various features wouldn't severely affect the regression models.

Machine Learning Models

For the machine learning models we chose to train on this dataset, we determined that regression models would be the most effective given that we wanted the model to take in inputs of statistics about the game and predicted weather conditions and output a predicted total score.

We started with Ridge Regression, which was implemented using the Ridge class from sklearn.linear_model with an initial alpha of 0.1 and an 80/20 train/test split. We chose Ridge Regression because of its regularization term which reduced our risk of overfitting to the data and controlled for multicollinearity between our features. We also wrote an optimization method that trained multiple ridge regression models with a variety of alphas and selected the one with the lowest Mean Absolute Error.

We then attempted to utilize XGBoost which is a gradient-boosting algorithm which we implemented by utilizing XGBRegressor from the xgboost library. For this model, we optimized hyperparameters like max_depth, n_estimators, and learning_rate using grid search and cross-validation. We chose XGBoost because it theoretically performs well in capturing complex relationships in data while also having built-in regularization to mitigate overfitting.

By comparing these models, we gained valuable insights into our data and identified the best-performing approach for predicting total points. Each model's performance was evaluated using Root Mean Squared Error and Mean Absolute Error metrics, ensuring our final selection provided both accuracy and consistency across different feature combinations.

Results and Discussion

Visualization

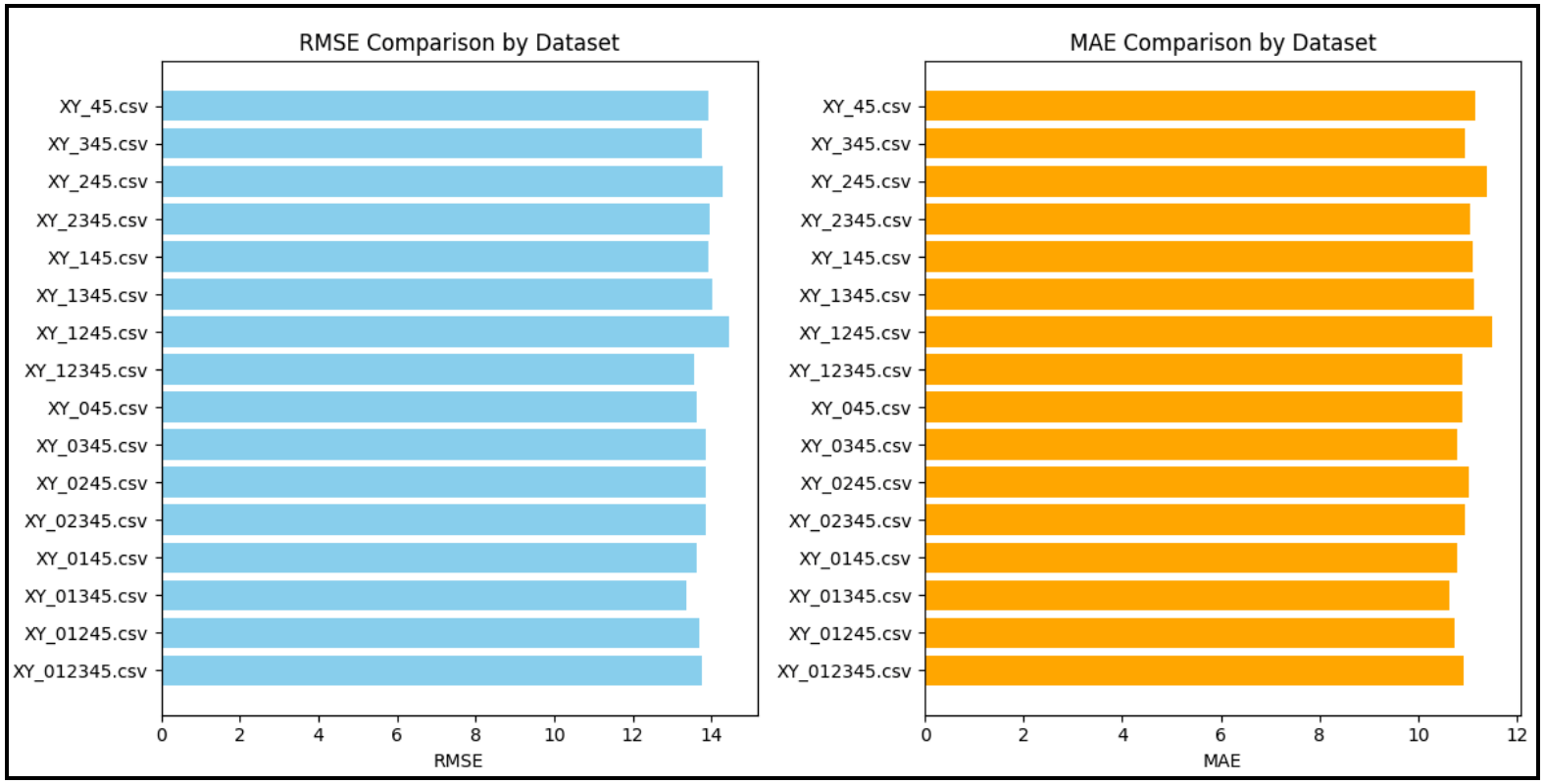

Figure 1: Ridge Regression RMSE and MAE on various combinations of features (best case: MAE = ~10.97)

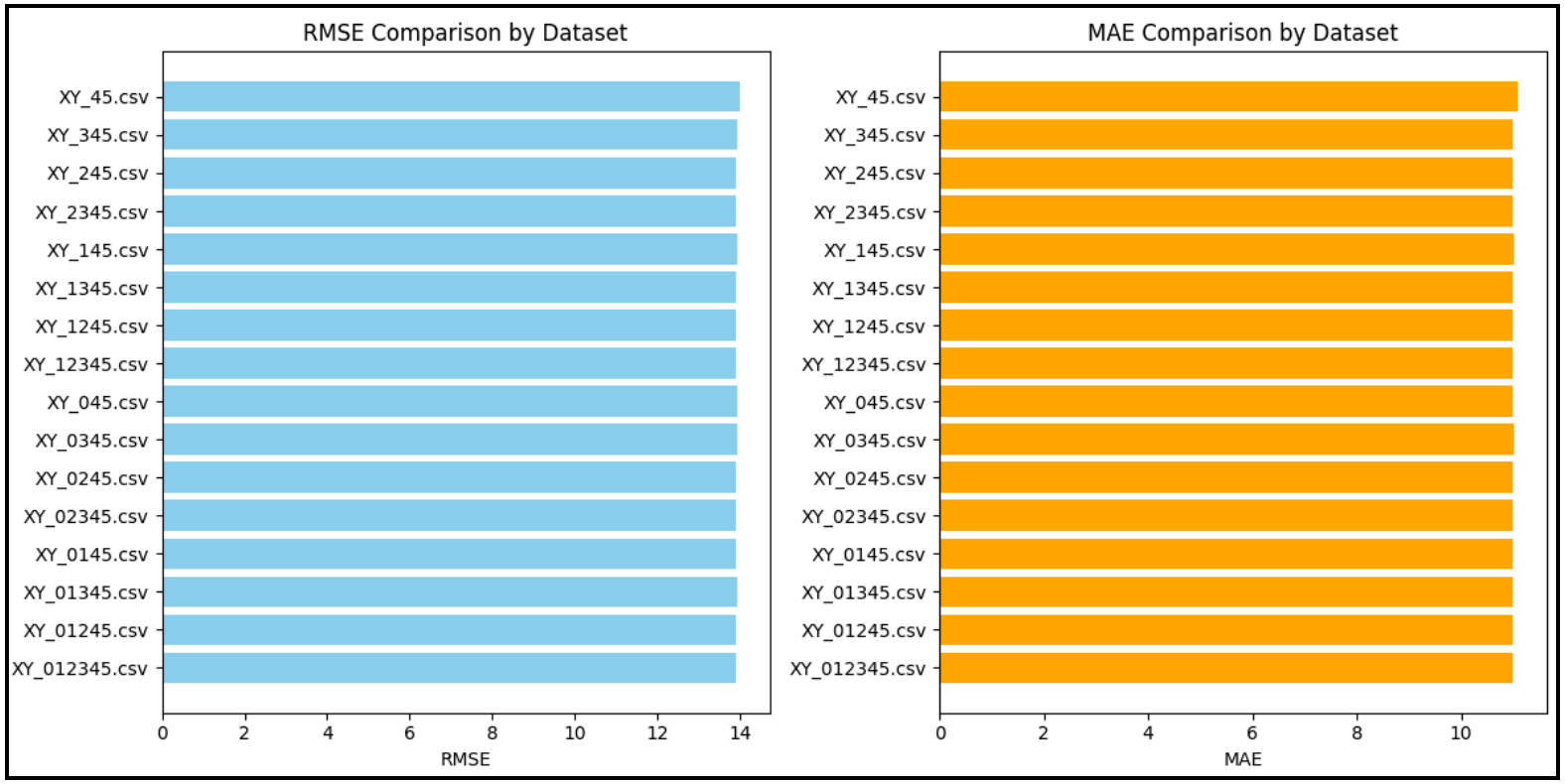

Figure 2: XGBoost Regression RMSE and MAE on various combinations of features (best case: MAE = ~11.01)

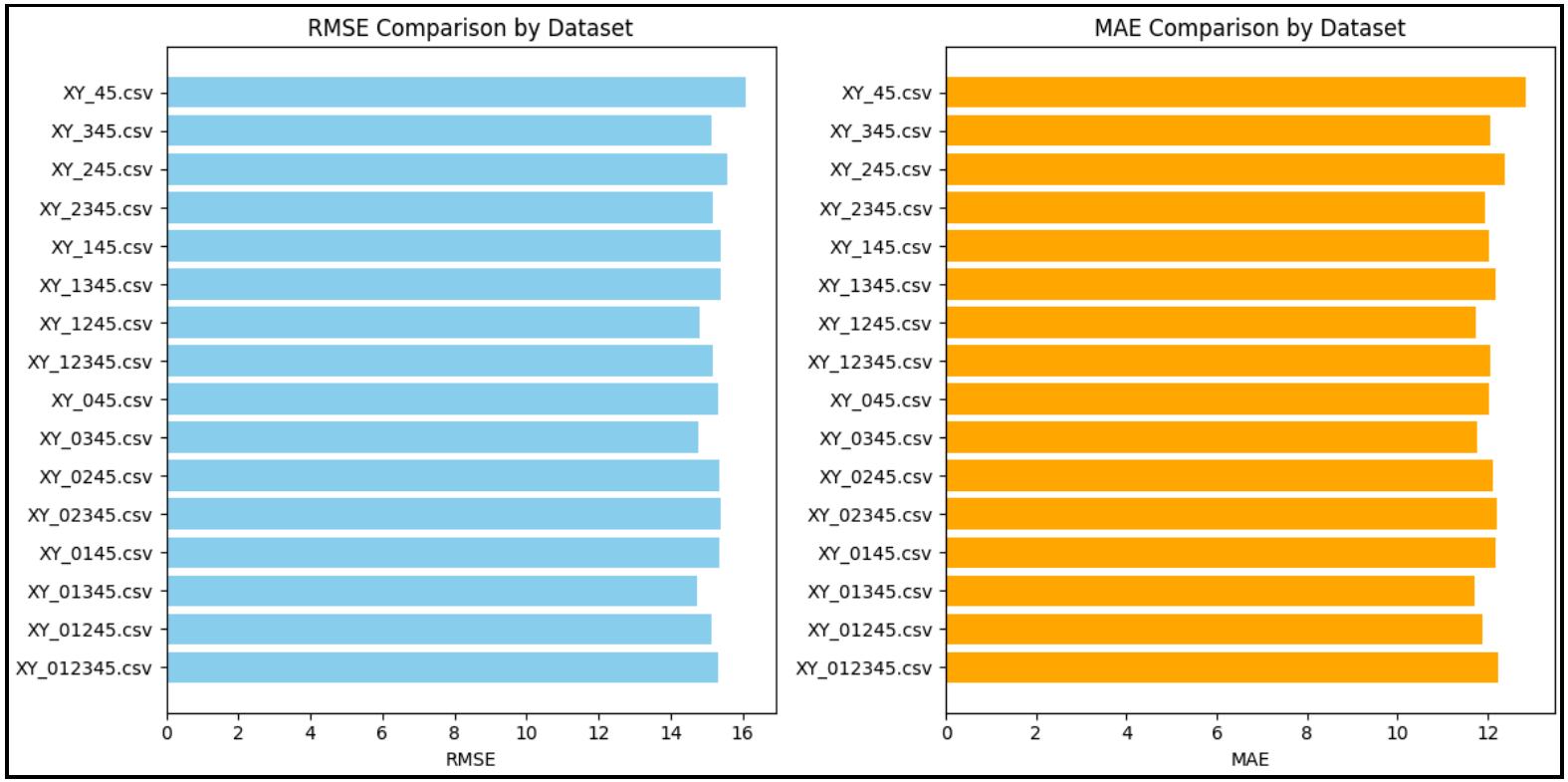

Figure 3: Random Forest Regression RMSE and MAE on various combinations of features (best case: MAE = ~11.97)

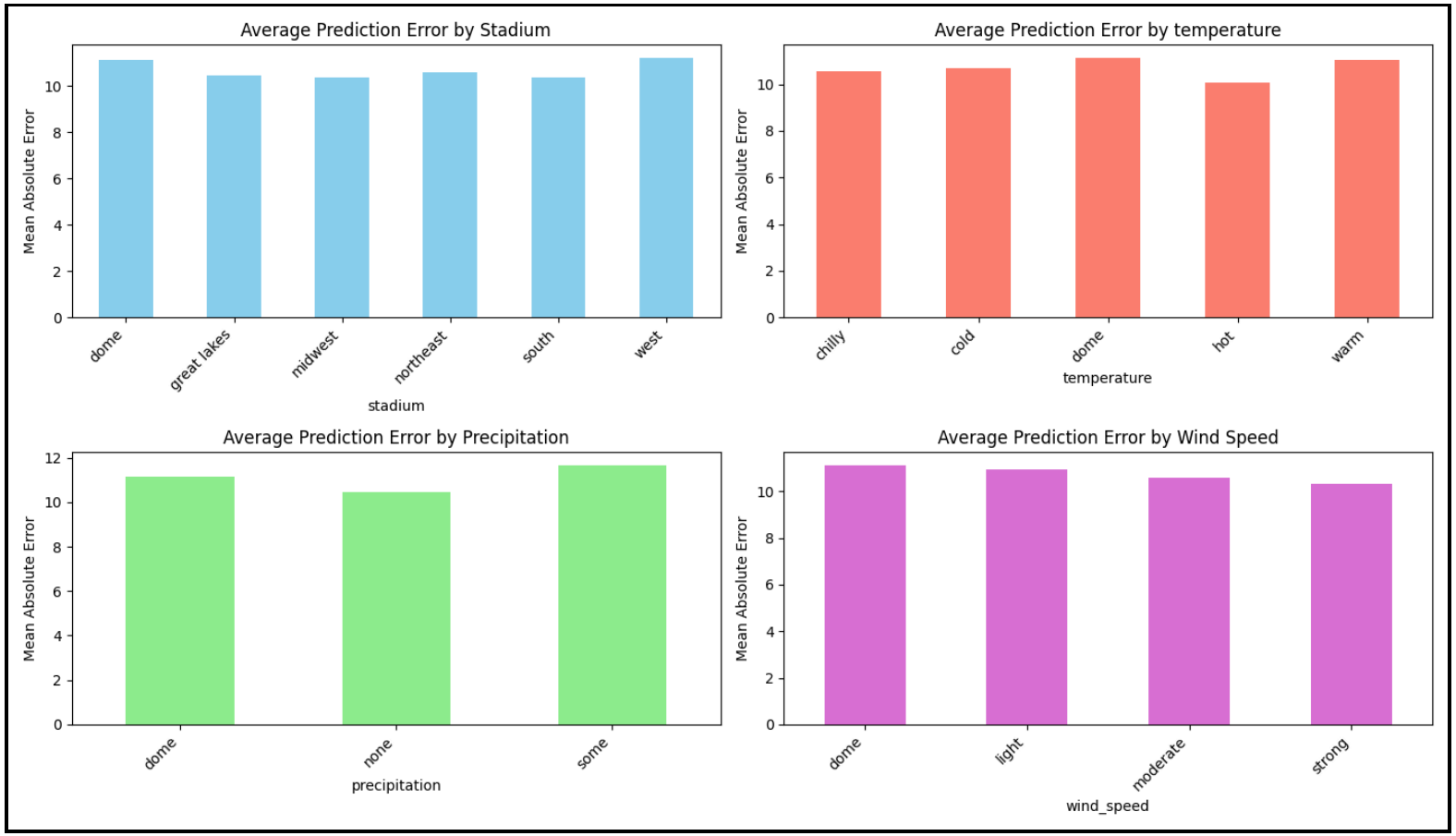

Figure 4: MAE for Ridge Regression on XY_01345.csv (best identified feature combination) by Stadium Region, Temperature, Precipitation, and Wind Speed

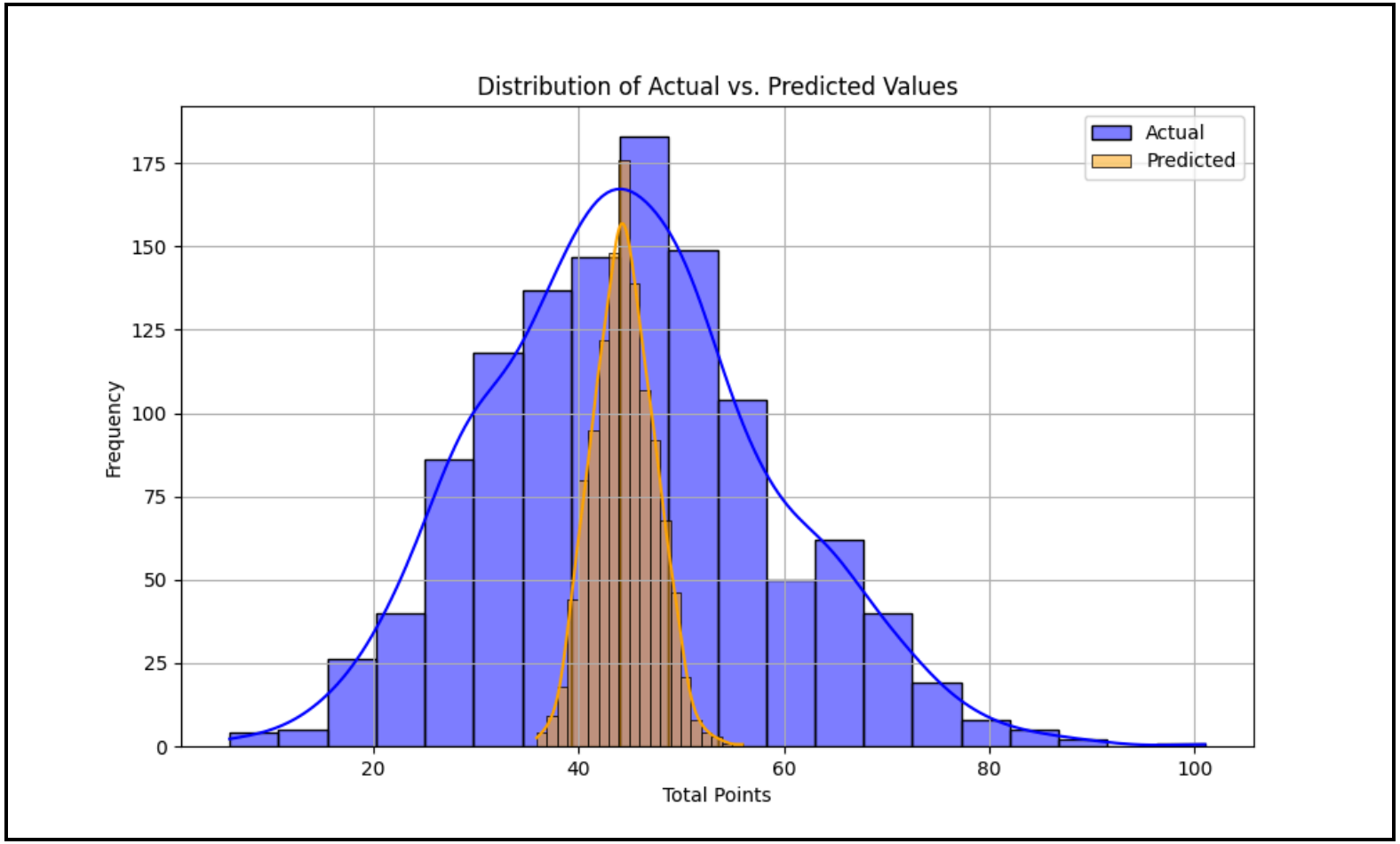

Figure 5: Distribution of Actual vs Predicted Values for Ridge Regression

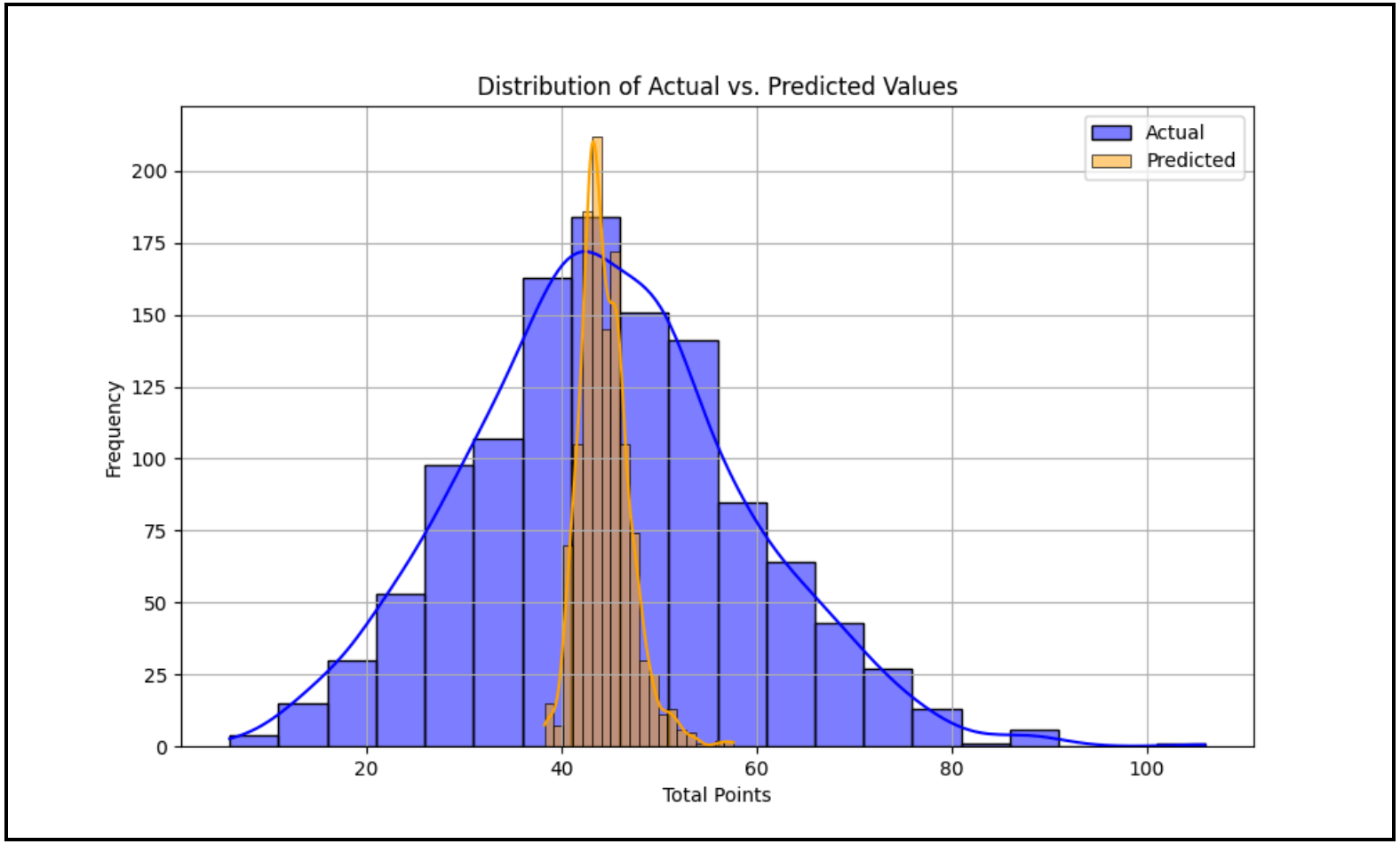

Figure 6: Distribution of Actual vs Predicted Values for XGBoost Regression

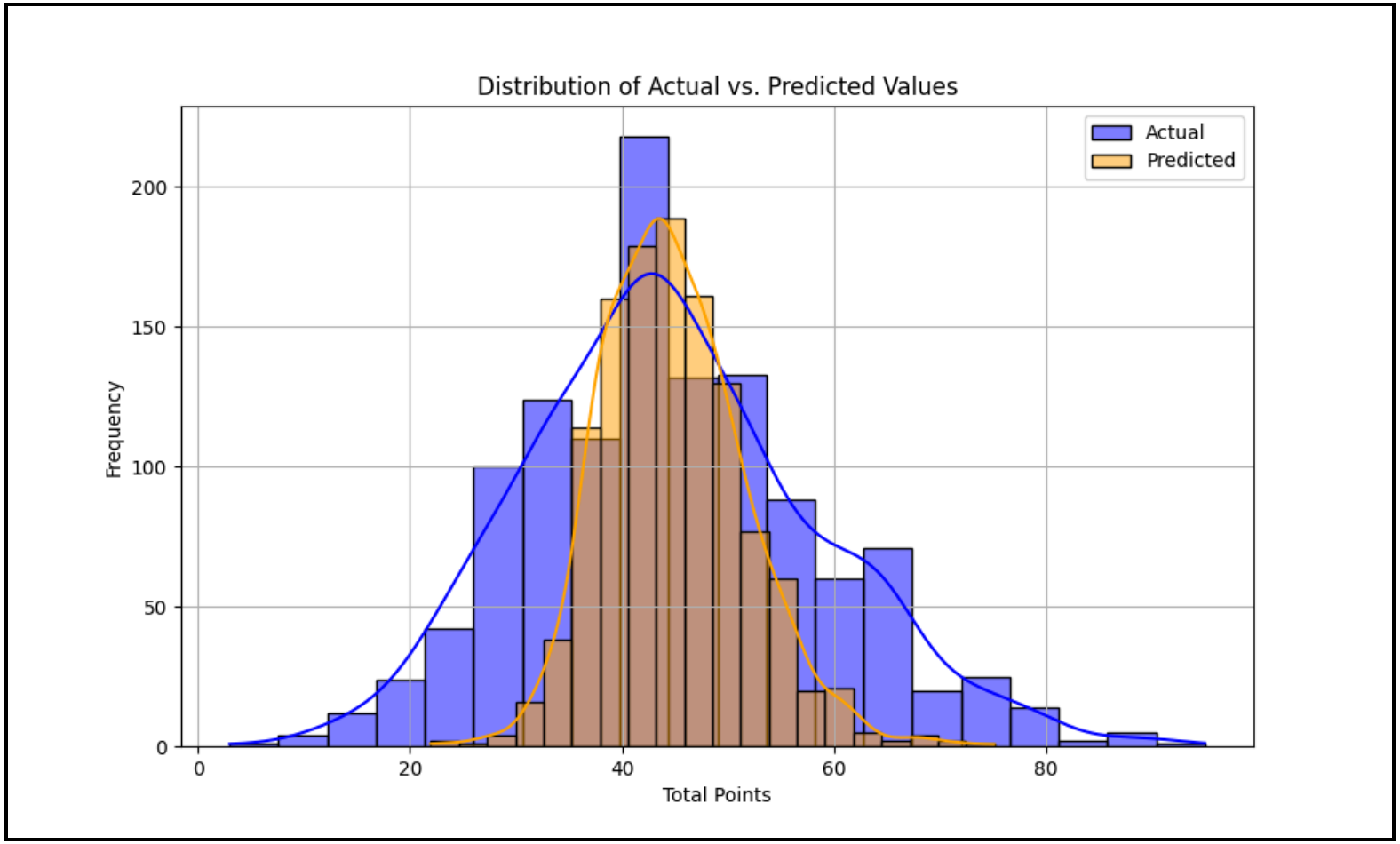

Figure 7: Distribution of Actual vs Predicted Values for Random Forest

Results sorted by file name with following format: XY_#.csv where # represents which features are present in that CSV according to this key:

0 = region of stadium, one-hot encoded

1 = average temperature, scalar encoded by percentile

2 = average precipitation, scalar encoded by percentile

3 = average wind speed, scalar encoded by percentile

4 = average score of home team over their past 5 games

5 = average score of away team over their past 5 games

Figure 8: Key for referencing input files to the ML models

Discussion

In this project, we aimed to predict the total points scored in a game using various regression models, namely Ridge Regression, XGBoost, and Random Forest Regressor. We evaluated each model on all possible feature combinations (represented by XY_#.csv files) and used RMSE and MAE metrics to assess performance. A clear pattern emerged with ~0.15 better MAE when features 4 and 5 (average score for the home team over the past 5 games and average score for the away team over the past 5 games) were included. The visualizations in Figures 1, 2, and 3 highlight the comparative performance when varying which other features were included, according to the key provided in Figure 8.

Model Performance Analysis

Ridge Regression:

As shown in Figure 1, Ridge Regression performed relatively well, achieving the lowest MAE (~10.97) on the XY_01345.csv dataset. This model's performance was consistent across different feature combinations, with minor variations. Ridge Regression's strength lies in its ability to handle multicollinearity and prevent overfitting through L2 regularization. However, its linear nature limits its ability to capture complex, non-linear relationships within the dataset, which may explain the observed prediction errors. For ridge regression, an automatic hyperparameter tuner was implemented to effectively test a variety of alpha values with the alpha resulting in the lowest MAE being selected. The alpha chosen most often was 0.1.

XGBoost Regression:

Figure 2 illustrates that XGBoost produced competitive results, with the best MAE (~11.01) on XY_245.csv. XGBoost's gradient boosting approach effectively captures non-linear patterns and interactions between features, contributing to its strong performance. Despite hyperparameter

tuning (max_depth=3, n_estimators=250, learning_rate=0.01), the model exhibited limited improvement, which indicates that further hyperparameter optimization or additional data might be required. XGBoost's consistent prediction range (40-50 points) suggests some bias towards the dataset's central values, possibly due to its ensemble learning strategy.

Random Forest Regression:

Figure 3 shows that Random Forest had the highest MAE (~11.97), indicating comparatively poorer performance. Although Random Forest handles non-linear relationships well and reduces variance through ensemble averaging, it struggled with this dataset. The model's complexity and potential overfitting may have contributed to its lower accuracy. Additionally, the distribution of predictions was broader, as can be seen in Figure 7, suggesting that Random Forest may have been more sensitive to feature variability.

Feature Importance and Visualization Insights

Figure 4 analyzed MAE across different features (stadium region, temperature, precipitation, wind speed) for Ridge Regression. No single feature stood out significantly; all contributed approximately equally to the model's performance. This indicates that the relationships between the input variables and the target (points scored) are distributed across multiple factors.

Figures 5, 6, and 7 compared the distribution of actual vs. predicted values for the three models. The actual points had a near-normal distribution, while the predicted values typically fit a narrower normal distribution with only Random Forest Regression's distribution coming close to the actual distribution (even though it had the highest MAE). This suggests that while the models capture the general trend of the data, they struggle to predict accurate scores for games, tending to predict the median of the dataset to minimize MAE, likely due to a lack of very predictive features.

What Worked and What Went Wrong

Initially, the input to the models contained a lot of raw data for temperature, precipitation, and wind speed in addition to one-hot encoded stadiums (resulting in 47 features exclusively for stadiums), resulting in extremely poor model performance. By sorting temperature, precipitation, and wind speed into bins and scalar encoding them, the models were able to understand the sequential relationship between feature values and improve their predictions. Introducing the average score over the past five games for both home and away teams significantly improved performance. This feature added temporal context, enabling models to capture trends in team performance better.

Initially, the models also struggled with overfitting, especially when using Random Forest. We attempted to address this by reducing the number of estimators and tuning hyperparameters, but the improvements were marginal. Additionally, XGBoost's consistent prediction range indicated a need for more diverse or representative training data to reduce bias.

Significant feature engineering work went into attempting to improve model performance which worked to a certain extent; however, it is our theory that the current features aren't closely correlated with the labels with too many external factors impacting overall points scored during a game (i.e. a team's quarterback getting injured, facing a team that's better at defense, etc.).

Conclusion and Next Steps

Ridge Regression emerged as the most consistent performer, with a balanced trade-off between bias and variance. XGBoost showed promise but requires further tuning or feature engineering to enhance performance. Random Forest, while robust in theory, struggled due to overfitting or dataset limitations. Future work should focus on expanding the dataset, incorporating more dynamic features (like player statistics and/or analyst predictions), and exploring advanced techniques such as ensemble stacking to combine model strengths.

References

[1] Cviaxmiwnptr, "NFL team stats 2002 - Feb. 2024 (ESPN)," Kaggle, https://www.kaggle.com/datasets/cviaxmiwnptr/nfl-team-stats-20022019-espn/data.

[2] Historical Weather API | Open-Meteo.com. (2023, September 10). https://open-meteo.com/en/docs/historical-weather-api#start_date=2023-09-10&end_date=2023-09-10&hourly=temperature_2m,precipitation,cloud_cover,wind_speed_10m&timezone=America%2FNew_York

[3] natesilver538, "How our NFL predictions work," FiveThirtyEight, https://fivethirtyeight.com/methodology/how-our-nfl-predictions-work/

[4] R. Arboretti et al., "Predictive stadium attendance using machine leaning: A case study in Italian football," Journal of Machine Intelligence and Data Science, vol. 5, 2024. doi:10.11159/jmids.2024.004

Other

Gantt Chart

Our Gantt Chart can be found here

Final Report Contribution Table

Name	Contributions Towards the Final
Joel Cave	Gantt chart, video
Ryan Corbett	Feature engineering
Shawn Coutinho	Model implementation / training, analysis
Michael Cao	Model implementation / training

Holden Casey	Documentation, data encoding

Video Presentation

Our video can be found here.