Fantasy IPL Prediction Hackathon - A Report on Approaches

Prajwal Vijay - EE23B057, Vignesh N - EE23B087

June 4, 2024

1 Best Performing Model

The Hackathon involved 2 tasks

- Regression task to predict the total_FP for a player.
- Classification task to determine the Captaincy status of a player.

We found that the best performing model is a combination of

- XGBRegression optimised with Feature Selection using the SelectKBest technique for Regression.
- Random Forest Classifier tuned with Optuna for Classification

Our best model had a:

- **Public dataset score of 0.15189**
- **Private dataset score of 0.15872**

2 Step 0 - The Dataset

We performed Exploratory Data Analysis and Feature Engineering on all the available features in the given dataset. Feature Engineering included:

- Removing null values in all columns appropriately
- Dropping the **match_name** column from the dataset
- Mapping **home_team and away_team** to integers
- Handling string entries in **Player_name**, **prev_Overs_Bowled** and **venue** using LabelEncoder()
- Mapping **Captain_Vice Captain** values to integers

Exploratory Data Analysis included:

- **Heatmap** of Correlations between all columns
- **Scatterplots** of different columns with respect to Total_FP
- **Barplots** of Correlation of different columns with Total_FP and prev_Total_FP
- **Barplot** of Correlation of different columns with Captain_Vice Captain

Some Feature Engineering was also performed, such as:

- **Player_name + venue**
- **Player_name + season + luck**

But these did not contribute to any improvements and so were not used.

3 Step 1 - Regression

The approaches taken are detailed below:

3.1 Classic Linear Regression

Done using the LinearRegression() function, taking all features in the dataset to predict Total_FP.
Reciprocal of RMSE was 0.0319 approximately

3.2 Feature Selection and XGBoost

Done using SelectKBest() and XGBRegressor() functions.
First we printed out the mutual_info_regression scores for different values of k. Since we did not know how many features to select (the value of k), we ran a loop taking all possible values of k from 1 to 23, predicting the reciprocal of RMSE using the XGBRegressor model (parameters chosen to reduce observed overfitting) and then using the value of k that gave the best RMSE reciprocal value.
Reciprocal of RMSE was 0.0330 approximately

Finally the last model was implemented after scaling the testing data appropriately

4 Step 2 - Classfication

The approaches taken are detailed below:

4.1 Neural Network

We made a Neural Network with 3 fully connected layers and 1 softmax output layer. We did not use checkpoints in the final model because there were issues with the Adam optimizer in this case. These can be fixed.
Validation accuracy was 0.923

4.2 Random Forest Classifier

The tabular nature of the data prompted us to use the RandomForestClassifier() function, which was easier to implement as well. RandomOverSampler() is also used to ensure the training dataset is well balanced to improve accuracy
Validation accuracy was 0.930 approximately

4.3 Random Forest Classifier with Optuna

To tune the hyperparameters optimally, the Optuna library is imported.
Validation accuracy was 0.997 approximately.
We however choose parameters of a model with slightly lesser accuracy, **0.98** since an accuracy greater than 0.99 means the model is perhaps overfitting to the validation set. This step is also supported since this model gives the maximum test accuracy.

Finally the last model was implemented after scaling the testing data appropriately.