FANTASY PREMIER LEAGUE PREDICTOR GROUP 15

Problem:

Fantasy sports became relevant as they helped fans become more involved in the sport and be part of something greater than themselves. Soon, this ended up being a sport in itself, and people found a new way to compete with each other. With this project, we aim to help people choose the best team that will allow them to score the maximum number of points for the game week. We forecast each player's fantasy points for the next game week based on their statistics from previous weeks and let the fan decide his best XI.

Exploratory Data Analysis and Data Preprocessing:

Recapping some critical points from our EDA:

1. Handled missing values for team scores by imputing with 0.
2. Removed multi-collinearity by removing highly correlated data features.
3. Reviewed data distributions and relationships between features and the target variable.
4. Analyzed feature importance using RandomForest and Lasso Regression.

Some critical points from the data preprocessing:

1. Considered a 3-period rolling average for player statistics.
2. Removed the outliers before feeding the data into the model.

Models:

After the preprocessing, the data had a set of numerical features corresponding to the target variable. The generic pipeline included scaling the features and performing PCA to represent the high-dimensional data followed by a regression model.

1. Linear Regression:

1.1. We start by using Linear Regression to map the player statistics with fantasy points.
1.2. Doing GridSearch for PCA yielded that the best model performance corresponded to the one that captured 95% of the variance, which is used in all the models.
1.3. Achieved a mean squared error of 5.26 on unscaled data, i.e., a mean deviation of 2.28 points in our predictions.
1.4. Analyzing the cause of such a high MSE value led us to conclude that it was due to outliers.
1.5. On removing the outliers from the dataset and rerunning the pipeline, an improved MSE of 1.76 was observed.

2. Elastic Net Regression

2.1. Considered this to promote feature selection and prevent over-fitting:
    2.1.1. The dataset is high-dimensional (26 features).
    2.1.2. The target is concentrated around zero, and learning can be affected without regularization.
2.2. Achieved a marginally improved MSE value of 1.75, i.e., the predictions deviate by 1.32 on average.

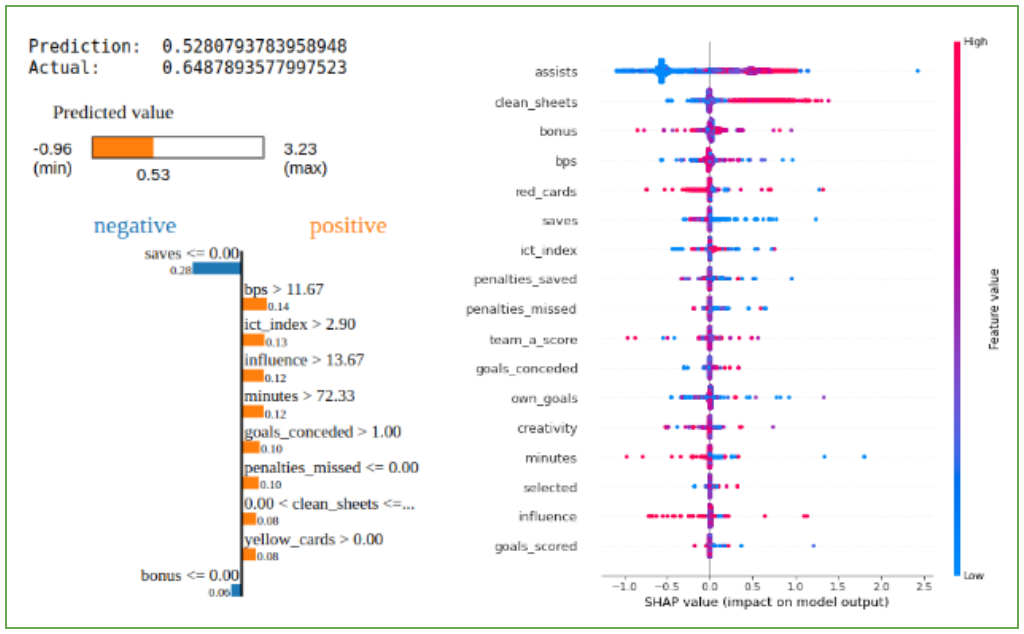
3. Decision Tree Regressor

3.1. Moving further, we tried out tree techniques to perform regression analysis.
3.2. Achieved a worse MSE of 1.78.
3.3. Tried to interpret the model using LIME and SHAP[1] and figure out the decision-making process.
3.4. The bar tells the range of the prediction, and the plot below tells how each decision positively or negatively influenced that prediction. For instance, a bps greater than 11.67 and an ict_index greater than 2.90 increased the points, while having saves less than equal to zero decreased the points.
3.5. This helped understand a particular sample, so further explore SHAP to interpret the model globally.

[1] Figure with green border

3.6. SHAP tells us perturbations in which feature affects the model output the most. For instance, features assists and clean sheets have a high model impact, i.e., on higher feature values, a higher score prediction is given. In contrast, a feature like saves doesn't have a significant impact on the target.

3.7. This helps us understand how the decision tree is giving importance to different features.



4. Random Forest Regressor

4.1. The obvious next step was to try out an ensemble technique. Consequently, we tried out Random Forest Regressor with max_depth = 8 and n_estimators = 50.

4.2. Achieved the best MSE so far of 1.72. LIME and SHAP[2] are used again to interpret the model.

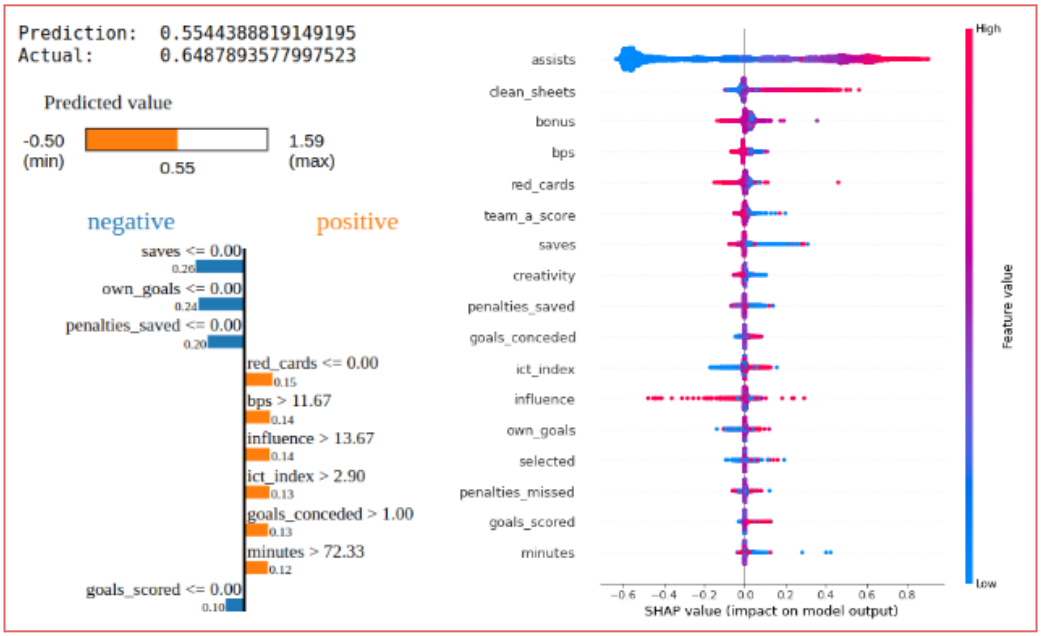
4.3. The random forest regressor for the 80th test sample gives negative weightage to decisions like non-positive saves, non-positive own goals, and non-positive saved penalties. Moreover, it assigns a positive influence to non-positive red cards, high influence. These influencers make sense in the real world, and therefore, we achieve a closer prediction than the decision tree.

4.4. The SHAP analysis for Random Forest indicates that assists and clean sheets have a significant impact on the model output, as in the case of decision trees.

5. XGBoost Regressor

5.1. Since the training set is close to 70k entries, we thought that boosting might improve the predictions and reduce the mean squared error.

5.2. It did so compared to the vanilla decision tree and linear regression models but couldn't surpass random forest. It achieved an MSE of 1.74. A similar LIME and SHAP[3] analysis were performed for XGBoost.



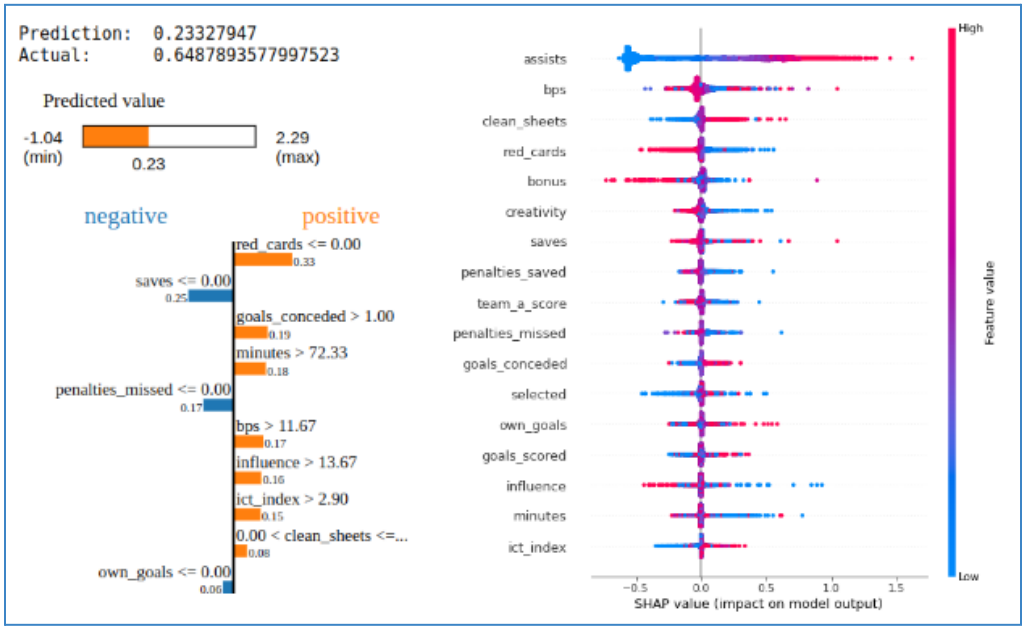
Observations:

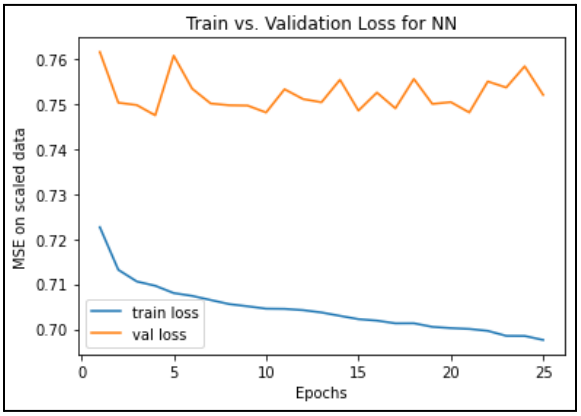
Of all these ML models, it turns out that the random forest regressor performed the best with a mean squared error of 1.72 on unscaled data. That translates to a deviation

[2] Figure with red border

[3] Figure with blue order

of approximately 1.32 points in our prediction. This error is somewhat reasonable because, essentially, the points are integers, and having a deviation of 1-1.3 points on average would not affect the overall points of the team. The chances of choosing a team such that all the players' points are +1.32 are significantly low. Therefore, we can conclude that our model is good enough for deployment. Although before that, we tried to use neural networks to boost the performance. We choose a simple Dense Network with one hidden layer totaling 11,649 parameters. On training for 25 epochs with a batch size of 32 and using the SGD optimizer, the model could attain a minimum MSE loss of value 1.73 on the unscaled test set. The model summary is shown below, along with the train/validation loss on scaled data.






```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_3 (Dense)             (None, 128)               3328

 dense_4 (Dense)             (None, 64)                8256

 dense_5 (Dense)             (None, 1)                 65

=================================================================
Total params: 11,649
Trainable params: 11,649
Non-trainable params: 0
```

Model Comparison:

S.No	Model	MSE
1	Linear Regression	1.76
2	Elastic Net Regression	1.75
3	Decision Tree Regressor	1.78
4	Random Forest Regressor	1.72
5	XGBoost Regressor	1.74
6	Neural Network	1.73

Conclusions:

Random Forest was the best model with the least mean squared error. It allows for model interpretation which makes it a good option in terms of understanding the decision-making process. In absolute terms, the ML models could do better. That is because the problem at hand is highly dynamic. Predicting the points should be challenging because there is a huge human component involved, and a lot depends on the kind of day the players have. Also, because of this variance in performance and the human sentiment involved, the whole business of betting comes into the picture.

Eklavya Jain (ej2487)
Shankh Suri (ss6625)
Marvin Limpijankit (ml4431)
Qifan Jiang (qj2172)