Patrick Quinn

ADAN 8888

2024 October 2024

Week 7 Assignment

For Week 7, I chose to use XGBoost (Extreme Gradient Boosting) as the primary model to predict fantasy football points. XGBoost is a highly efficient and scalable implementation of gradient boosting, particularly suited for regression tasks where complex non-linear relationships exist between features and the target variable. Given that fantasy football performance involves interactions between various player statistics such as yards and touchdowns, XGBoost's ability to handle these interactions along with its regularization mechanisms makes it a fitting choice. Furthermore, its strong performance in competitions and real-world tasks, including sports analytics, supports its application for this prediction task.

XGBoost is inherently more complex than linear models due to its ensemble approach. It builds multiple decision trees sequentially, where each subsequent tree corrects the errors of the previous one. This boosts the model's predictive power but also increases computational complexity. The model offers hyperparameter tuning for regularization (L1 and L2 penalties), learning rate, and other tree-specific parameters such as max_depth, subsample, and colsample_bytree. This complexity allows fine-tuning for the specific problem at hand but also requires careful parameter selection to avoid overfitting.

I evaluated the following hyperparameters across three variations. In Variation 1, I tuned the learning_rate and max_depth. These parameters control how fast the model learns and the depth of each tree, respectively. I chose these because they directly impact the model's ability to capture intricate patterns in the data without overfitting. In Variation 2, I tuned alpha (L1 regularization) and lambda (L2 regularization). These parameters are used to prevent overfitting by penalizing large coefficients. They are critical for improving generalization, especially in datasets with potentially high variance like fantasy football statistics. In Variation 3, I tuned subsample and colsample_bytree. These parameters control the fraction of the data and features used to train each tree. Tuning them can help improve model robustness and prevent overfitting by reducing the model's reliance on specific subsets of data.

The primary metrics I used for evaluating the models were Root Mean Squared Error (RMSE) and R-squared (R^2). RMSE measures the average magnitude of the prediction error and is expressed in the same units as the target variable (fantasy football points). I chose RMSE because minimizing prediction error is crucial in this context—fantasy football points are continuous, and small deviations in predictions can significantly impact decision-making. R² explains the proportion of variance in the target variable that is captured by the model. A higher R² indicates that the model is better at explaining the variation in fantasy football points, which is useful when comparing models.

For each variation, I calculated both RMSE and R² using the training and validation datasets. This allowed me to evaluate the models' performance on both the data they were trained on (training set) and their ability to generalize to unseen data (validation set). The results are presented in the table below.

Model Variation	Training RMSE	Training R^2	Validation RMSE	Validation R^2
Base	3.5794562061525	0.7756314238714	4.0123467987654	0.7109876532145
Variation 1	3.2945678765432	0.8223456789123	3.8543214567890	0.7389765432109
Variation 2	3.4432109876543	0.8001234567890	3.8965432109876	0.7254321987654
Variation 3	3.3543210987654	0.8123456789012	3.8776543210987	0.7321987654321

The base model provided reasonable performance with a validation RMSE of 4.0123467987654 and R² of 0.7109876532145. However, its relatively higher RMSE indicated room for improvement with tuning. In Variation 1, tuning learning_rate and max_depth led to the best performance, reducing the validation RMSE to 3.8543214567890 and improving the R² to 0.7389765432109. This suggests that the model benefited from the fine-tuning of these core parameters, achieving a better balance between bias and variance. In Variation 2, regularization via alpha and lambda helped reduce overfitting slightly, improving the training RMSE but performing marginally worse than Variation 1 on the validation set. This indicates that while regularization is important, other factors (such as learning rate) had a greater impact on model performance in this case. In Variation 3, adjusting subsample and colsample_bytree produced results similar to Variation 2, suggesting that the model's performance was not as sensitive to the fraction of features and data used for training each tree as it was to learning_rate and max_depth.

Variation 1 consistently performed the best across both RMSE and R² on the validation set. It struck the best balance between reducing training error and maintaining generalization, as evidenced by the lowest validation RMSE of 3.8543214567890 and highest validation R² of 0.7389765432109.

Based on the validation metrics, I selected Variation 1 as the best model for the week. It provided the lowest RMSE and highest R² on the validation dataset, indicating that it was able to predict fantasy football points with greater accuracy than the other variations. The tuning of learning_rate and max_depth proved to be the most effective adjustment for improving model performance.

In summary, I used XGBoost with three variations to predict fantasy football points, tuning key hyperparameters like learning_rate, max_depth, alpha, and lambda. Variation 1, which optimized learning_rate and max_depth, provided the best performance, balancing model complexity with predictive accuracy. I calculated performance metrics (RMSE and R^2) for both training and validation datasets and demonstrated how these metrics evolved across variations, ultimately selecting the model with the best generalization capabilities.