Capstone Project : In-depth Analysis (Machine Learning)

Regression Models.

My first step was to decide what type of machine learning model to apply to my dataset. Considering that the purpose of my project is to predict fantasy points scored for players and doesn't incorporate classifying players into different categories, the most appropriate machine learning technique to apply to my dataset is a regression model. Having decided on regression analysis being the best fit for my project I then needed to determine the most appropriate type of regression analysis to apply a; Linear, Random Forest, Ridge, or Lasso regression model.

While implementing the different models I adjusted the hyper parameters for all the models and would then run the ".score" method to test scores of both the training and testing data sets, as well as taking the Root Mean Squared Error, to see if the change I made resulted in a higher or lower score.

The Linear regression model had the fewest hyper parameters and gave us the best score using defaults. I was able to improve the score by adjusting the "train_test_split()" parameters. I found that a test_size=0.275 and a random_state=411 resulted in the highest score. I used this same "train_test_split" for all the regression models. The Linear Regression models scores on the test dataset were (0.99886406) with a Root Mean Squared Error of (2.66563).

When tuning the parameters for the Random Forest I found that adjusting the "n_estimators" parameter to equal 2000 resulted in the highest score on the test dataset (0.9172012) with a Root Mean Squared Error of (22.75801). The Random Forest model does not have coefficients so we cannot visualize or test those parameters.

The Lasso and Ridge Regression models had more hyper parameters to tune. I chose to focus my adjustments on the alpha parameter; testing models with low alphas versus models with high alphas to see how they affect the coefficients. I found that the lower value for alpha gave us the highest score and best Root Mean Squared Error.

For the Ridge Regression models, the alpha equal to 0.01 resulted in a much better score on the test dataset (0.99886415) than setting alpha equal to 100 (0.99797109). The Root Mean Squared Error were (2.66553) and (4.17908) respectively. We received the same number of features for both values of alphas, 18.

The lower alpha value for the Lasso Regression model also resulted in the best score; alpha equal to 0.0001 (0.9988639) versus alpha equal to 0.01 (0.99884482). The Root Mean Squared Error were (2.66582) and (2.68812) respectively. However, with Lasso Regression, we saw an increase in the number of features used when decreasing the values of alpha, 18 features for the lower alpha value and 17 features for the higher alpha value.

In conclusion, after testing all the types of regression models and fine tuning their hyper parameters, I found that the Linear Regression model performed best, followed very closely by the Ridge and Lasso Regression models with the lower alpha values.

Their scores rank as follows:

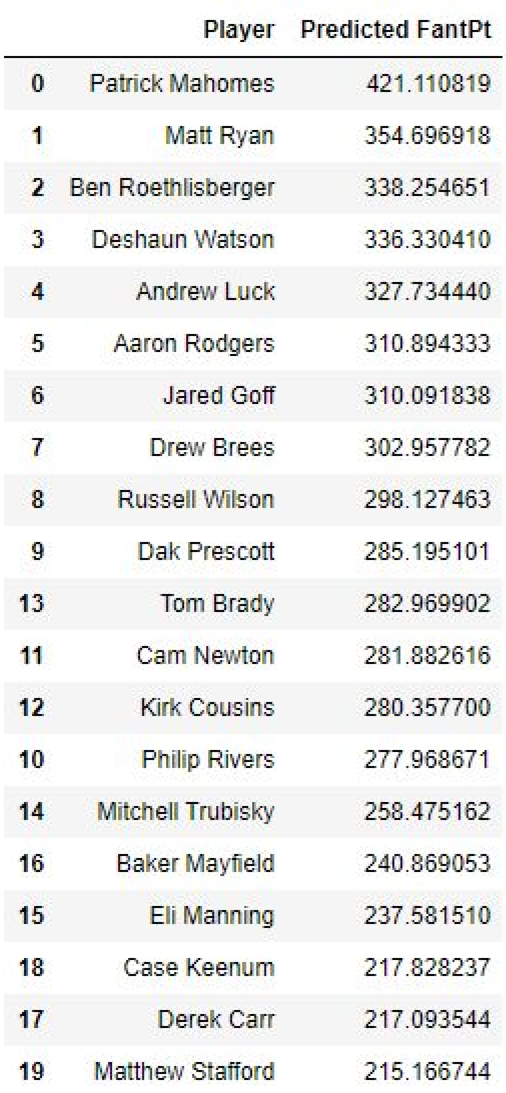
Linear Regression (0.9988641)
Ridge Regression (0.9988641)
Lasso Regression (0.9988639)

Theses two models only differed in scoring by an extremely small percentage point. With the gap being so slim between the Linear and Ridge and Lasso models I chose to display my prediction results for all three of the best performing models. This code is specifically modeled, tuned and tested to project the fantasy points scored for the Quarterback position.

Ridge Regression Model code.

```
# Creating Ridge Regression Model - Low Alpha
        rr = Ridge(alpha=0.01)
        rr.fit(X_train, y_train)
# Scoring the Ridge Regression Model
        Ridge_train_score = rr.score(X_train,y_train)
        Ridge_test_score = rr.score(X_test, y_test)
        coeff_used01 = np.sum(rr.coef_!=0)
        print("Ridge Regression Train Score Low Alpha:", Ridge_train_score.round(7))
        print("Ridge Regression Test Score Low Alpha:", Ridge_test_score.round(7))
        print("Number of Features Used for Alpha=100:", coeff_used01)
```

```
# Output
    Ridge Regression Train Score Low Alpha: 0.998476
    Ridge Regression Test Score Low Alpha: 0.9988641
    Number of Features Used for Alpha=100: 18
    Root Mean Squared Error: 2.66553
# Plot for Ridge Regression Coefficient - Low Alpha
    plt.plot(rr.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,
         color='red',label=r'Ridge; $\alpha = 0.01$',zorder=7)
# Plot for Ridge Regression Coefficient - High Alpha
    plt.plot(rr100.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,
        color='blue',label=r'Ridge; $\alpha = 100$')
# Plot for Linear Regression Coefficient
    plt.plot(lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,
        color='green',label='Linear Regression')
    plt.xlabel('Coefficient Index',fontsize=16)
    plt.ylabel('Coefficient Magnitude',fontsize=16)
    plt.legend(bbox_to_anchor=(1, 0.5))
    plt.show()
# Predicting QBs with Ridge Regression
    top20_qbs = X_qb[:20]
    y_top20_qbs_rr = rr.predict(top20_qbs)
# Creating DataFrame to Visualize Prediction Results
    y_top20_qbs_rr = pd.DataFrame(y_top20_qbs_rr)
    qb_names = qb['Player'][:20]
    frames = [qb_names, y_top20_qbs_rr]
    qb_names_pred = pd.concat(frames, axis=1)
    qb_names_pred.columns = ['Player', 'Predicted FantPt']
    qb_names_pred.sort_values(['Predicted FantPt'], ascending=False)
```



	Player	Predicted FantPt
0	Patrick Mahomes	421.110819
1	Matt Ryan	354.696918
2	Ben Roethlisberger	338.254651
3	Deshaun Watson	336.330410
4	Andrew Luck	327.734440
5	Aaron Rodgers	310.894333
6	Jared Goff	310.091838
7	Drew Brees	302.957782
8	Russell Wilson	298.127463
9	Dak Prescott	285.195101
13	Tom Brady	282.969902
11	Cam Newton	281.882616
12	Kirk Cousins	280.357700
10	Philip Rivers	277.968671
14	Mitchell Trubisky	258.475162
16	Baker Mayfield	240.869053
15	Eli Manning	237.581510
18	Case Keenum	217.828237
17	Derek Carr	217.093544
19	Matthew Stafford	215.166744

Hypothesis development

Working through this project I hypothesized that touchdowns and yards (passing, rushing, or receiving) would be the best predictor for future performance. While doing my correlation analysis my results supported my hypothesis. The two highest correlated statistics to fantasy points scored where touchdowns and yards, respectively.Touchdowns with a 0.928808 correlation and yards with a 0.900263 correlation. The next highest correlator was completions with a 0.834737 correlation. These data apply to the Quarterback position.

The data supports this throughout the other positions (Running Back, Wide Reciever, and Tight Ends). However, the correlation is much lower. This is due to their ability and frequency to score points in a variety of ways. For example, the highest correlator for the Running Back position is TD.3, or total touchdowns. This stems from the running back having a higher frequency in rushing and receiving touchdowns, as well as scoring points for receptions. So the highest correlator for running backs is at 0.788136 with rushing yards not far behind with a correlation of 0.747687.

Where my hypothesis failed to find support was in the Wide Receiver and Tight End positions. Both positions highest correlator was the Yds.2 stat, or receiving yards. The next highest correlators were receptions and targets, respectively. An in depth analysis of these results proved that receiving yards and receptions makes more sense to be the best predictor of future fantasy performance. First, scoring touchdowns is much more volatile for the receiving positions compared to the Quarterback and Running Back positions. Second, with this being true, the more reliable source of points for the receiving positions comes from their receiving yards and receptions. Considering that the data is based on the points per reception scoring system, the receptions are much more valuable for the receiving positions than touchdowns.

Conclusion of Machine Learning Analysis

After an in-depth analysis of the machine learning techniques I was able to find the most appropriate, best fitting regression models for each specific position, as well as further develop my hypothesis. Having analyzed the results from the process of creating machine learning models I found that my original hypothesis held true for the Quarterbacks and Running Backs position, however, was not the case for the Wide Receiver and Tight Ends position. Even though this goes against my original hypothesis it makes logical sense considering the scope of the scoring system and the high variability in receiving touchdowns for a player from year to year. It would be appropriate that the highest correlators to scoring for a receiving position comes in the form of receiving yards and receptions.