Sami Hakkarainen

MATH-0165

December 2023

An Analysis and Walkthrough of *Football Analytics with Python & R*

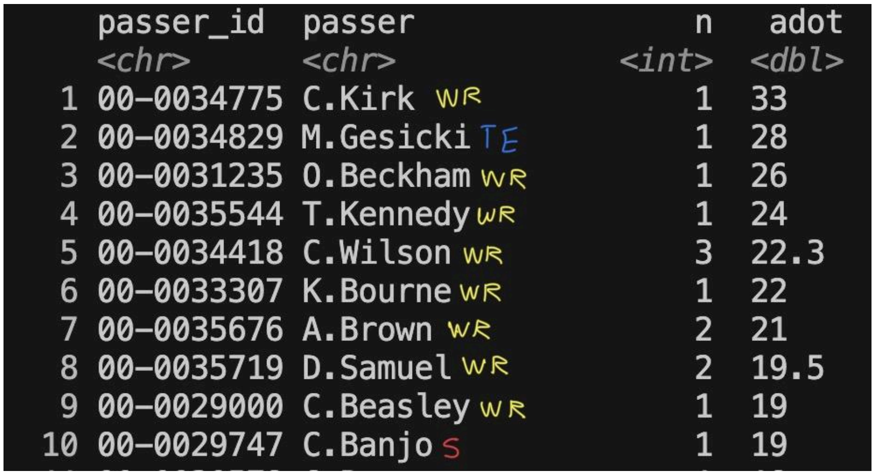
When learning about any subject, having real world examples along the way is helpful. This was prevalent throughout the entirety of MATH-0165. Whether it was the first day of class and our interest for probability was sparked with the birthday problem, or it was a story to go along with a named distribution, there was always an element of tying real life examples to the concepts being learned in class. The book *Football Analytics with Python and R,* written by Eric Eager and Richard Erickson, is a prime example of this idea. The authors take ideas of data science - learning how to use Python and R, stability of statistics, and linear regressions - and apply these ideas to play-by-play NFL statistics. As an avid sports fan, I cannot think of a better way to be introduced to these concepts; it continually sparks my interest and allows me to develop an inquisitive approach to applying what I've learned. In this document I will outline what I've learned and detail the examples that were given from the book.

Throughout the book, there is typically one case study given per chapter. These case studies are in R and Python, but I focused on R while carrying these tutorials myself. I made this decision predominantly because I had never used R before and I wanted to get experience with the language.

Learning R and Analyzing Data

The first chapter of the book is simply getting versed on the basics of R, and the package to be used throughout the rest of the book: NflFastR. This R package offers a repository of NFL play-by-play data dating back to 1999, and provides a set of functions for users to effectively scrape this data. Each play has some 300 metrics associated with it - such as play type, home team, rusher ID, etc. - which allows users to perform all kinds of operations with the data. Some notable methods of data manipulation introduced in this chapter are filtering and summarizing, as well as ensuring that data produced is an accurate representation of the question being asked.

One example of this “accurate representation” can be ranking NFL quarterbacks by average depth of target, or “*What quarterback’s throw deep balls the most?*”. Say you isolate all passing plays for a season and group by passer_id and passer name. Then, sort the results in descending order of the mean of a player's passing yards for the entirety of a season. This produces the following result:



```
   passer_id  passer               n  adot
   <chr>      <chr>            <int> <dbl>
 1 00-0034775 C.Kirk WR            1  33
 2 00-0034829 M.Gesicki TE         1  28
 3 00-0031235 O.Beckham WR         1  26
 4 00-0035544 T.Kennedy WR         1  24
 5 00-0034418 C.Wilson WR          3  22.3
 6 00-0033307 K.Bourne WR          1  22
 7 00-0035676 A.Brown WR           2  21
 8 00-0035719 D.Samuel WR          2  19.5
 9 00-0029000 C.Beasley WR         1  19
10 00-0029747 C.Banjo S            1  19
```

Figure 1: *Passer’s position was added to right have their name to provide context

Of the top 10 NFL passers of “average depth of target”, 8 are wide receivers, 1 is a tight end, and 1 is a safety. None of these players are quarterbacks. Now, observe the “n” column on the right. “n” indicates the number of passes thrown by a given player. This can be attributed to trick plays, as sometimes a team will have a non-quarterback throw a pass to trick the defense. Thus, in order to filter out these anomalies and answer the question of “*What quarterback’s throw deep balls the most?*”, one way to do this could be to filter the results such that a passer must attempt at least 50 passes in order to be considered. It is imperative to display several different metrics when observing data, and to ensure that it all looks up to par before making any general assumptions. In figure 1, if someone did know the positions of the named passers, they would have no way of knowing they are non-quarterbacks.

Stable vs. Unstable Statistics

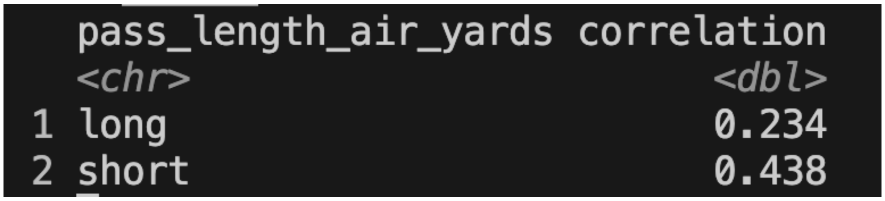
Eager and Erickson define stability as “how much of a certain statistic is ‘actually useful’, i.e. transferable to other areas, versus what can be attributed to the circumstance of the situation” (19). Stability of statistics is important because it shows how much of a statistic can be attributed to *variance*, and how much of it can be transferable to other settings. This is at the core of statistics and analytics, because you can have the fanciest algorithms and software out there, but it doesn’t mean anything if this system is built upon unstable statistics that have no correlation to future output.

The case study given in this chapter makes the hypothesis “*Throwing deep passes is more valuable than short passes, but it’s difficult to say whether or not a quarterback is good at deep passes*.” (22). The goal of this chapter is to test the latter half of this hypothesis that states the difficulty of identifying a good “deep ball” quarterback, or in other words, the stability of deep passes. One technique provided to identify the stability of a statistic is **lagging**. In the context of NFL data, this involves the following steps:

1. Create dataframes. Construct two dataframes with similar timelines, and increment the season of the “lagged dataframe” by 1. The purpose of this is so that when merged together, the lagged dataframe will compare the previous season’s statistics to that of the other dataframe.
2. Rename focus metrics. For example, in this case study, “ypa”, or yards per attempt, is being analyzed. So within the lagged dataframe, we will change the name of the “ypa” metric to “ypa_last” to represent the “ypa” from a player’s previous season.
3. Merge the two dataframes together. Now, because names of key metrics were changed, the merged data frame has both “ypa” and “ypa_last” to represent the “yards per attempt” statistic for the current year as well as the previous year.

Now that both metrics are contained within a single dataframe, we can identify the correlation between these two variables. As best stated by Eager and Erickson, “A correlation coefficient ranges from -1 to 1. A value of 0 corresponds to no correlation, whereas a value of 1 corresponds to a perfect positive correlation” (20). Luckily, R provides functions to swiftly calculate this datapoint, so there’s no need for obtaining variance and covariance for each aggregate (although I now possess the tools for that computation after a semester of probability).

Previously in the case study, passes of a length greater than 20 yards were given the tag “long” and those shorter than 20 yards were tagged “short”. By partitioning these two sets of passes and comparing the “ypa” to “ypa_last”, we were able to separately calculate correlation values for short vs. long passes, further



```
  pass_length_air_yards correlation
  <chr>                       <dbl>
1 long                        0.234
2 short                       0.438
```

Figure 2

Short passes have a correlation almost double that of long passes. Because short passes are more correlated year to year (figure 2), we can conclusively say that short passes are a more stable metric than long passes. This confirms our hypothesis that stated *“it’s difficult to say whether or not a quarterback is good at deep passes”*.

Simple Linear regressions

At its core, a linear regression can be thought of as a “line of best fit” that minimizes the overall distance between each individual data point in an *xy*-plane. The most key estimates within linear modeling are the slope and intercept coefficients of the line of best fit, and the R-squared value, which represents how accurate the model is to the actual data. An R-squared value of 1.0 means that a linear model can correctly represent the data 100% of the time.

When analyzing data in the NFL, there are countless variables at play. The more we can isolate these variables, the better assessments we can make about the data itself. As stated by Eager and Erickson, “Through regression, you can *normalize*, or *control for*, variables that have been shown to affect a player's production." (55). This chapter focuses on normalizing "yards to go" as a metric in rushing plays. Yards to go is defined as the amount of yards to a first down or touchdown a certain play is. For example, plays of "1st and 10" have 10 yards to go, plays of "3rd and 3" have 3 yards to go, etc.. These are discrete values, ranging from 1 to 99. This is a simple linear regression because only one variable is being analyzed.

*Note: 99 yards would be the greatest possible amount of yards to go on a football field. However, this is highly unlikely, because it would require the "line to gain" to be 99 yards ahead of the line of scrimmage, which is highly unlikely. More than 99% of plays have a "yards to go" value of 25 or less.

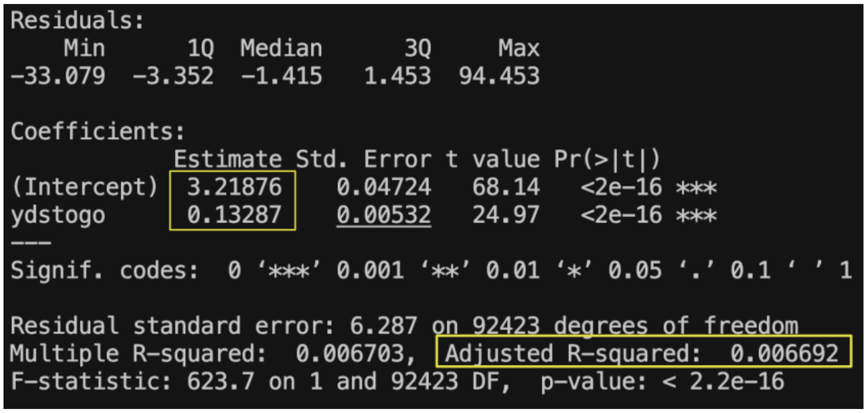
The goal of this case study is to analyze a metric known as RYOE, or rushing yards over expected. There are more advanced ways to calculate expected rushing yards than the techniques used in this chapter, but using RYOE serves as a good example to learn about linear regressions. The case study begins by extracting the play by play data from 2016 to 2022 and isolating rushing plays. In R, a linear regression model is produced by using the following syntax:

lm(rushing_yards ~ 1 + ydstogo, data = pbp_r_run)

Formula 3

, where pbp_r_run is the collection of rushing plays. This syntax can be read in plain english as "Rushing yards are predicted by an intercept (1) and a "slope parameter" for yards to go (ydstogo). Both rushing_yards and ydstogo are field descriptions within NflfastR's play by play library.

Lots of useful information is provided with this function call. Most notably, the slope and intercept of the residual line, and the R-squared value. In R, there is a package to carry out linear modeling, and by inserting the formula above (formula 3) into this library function, along with all NFL plays of type “rush”, we are given the following output:

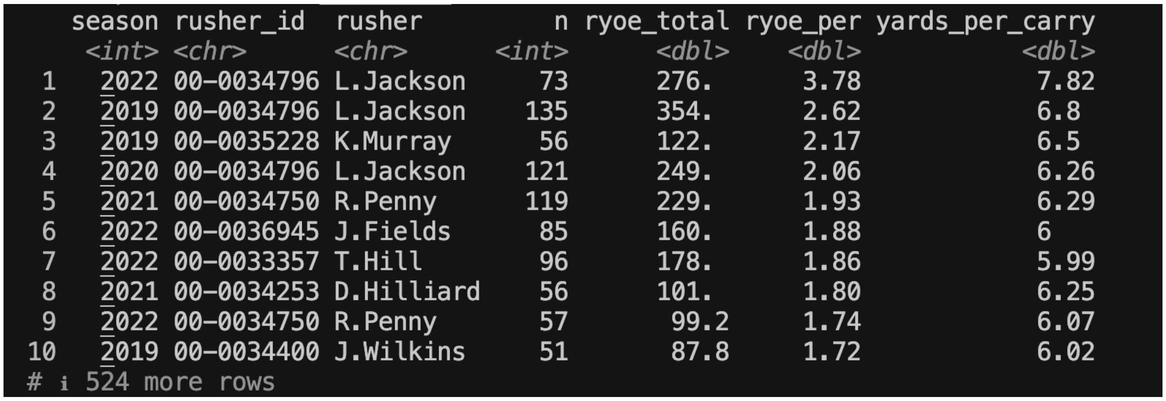


```
Residuals:
    Min      1Q  Median      3Q     Max 
-33.079  -3.352  -1.415   1.453  94.453 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.21876    0.04724   68.14   <2e-16 ***
ydstogo      0.13287    0.00532   24.97   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.287 on 92423 degrees of freedom
Multiple R-squared:  0.006703, Adjusted R-squared:  0.006692 
F-statistic: 623.7 on 1 and 92423 DF,  p-value: < 2.2e-16
```

Figure 4

Granted, an R-squared value of 0.0067 means that the linear model is not very accurate at predicting data. But this is expected because of the unpredictability with rushing plays in the NFL. And as mentioned earlier, there are better ways to calculate expected rushing yards, but this example is intended for learning purposes. These results of the intercept and ydstogo can be interpreted as the following: the expected rushing yards on any given play = 3.22 + (0.133) * (ydstogo).

Alas, by using this “line of best fit” with the play by play data with the “residual” function to get a “RYOE” metric for each play. The residual function does this by subtracting the “expected value” at a certain amount of yards to go by the actual passing yards recorded on that play. By adding these residuals to the play by play data structure, we can produce a new “field” that represents RYOE per play. By grouping data by season and player, we can then assess who had the best season performance when analyzing RYOE per play.

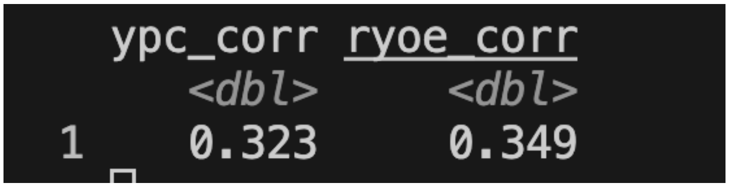


```
   season rusher_id  rusher         n ryoe_total ryoe_per yards_per_carry
    <int> <chr>      <chr>      <int>      <dbl>    <dbl>           <dbl>
 1   2022 00-0034796 L.Jackson     73      276.      3.78            7.82
 2   2019 00-0034796 L.Jackson    135      354.      2.62            6.8
 3   2019 00-0035228 K.Murray      56      122.      2.17            6.5
 4   2020 00-0034796 L.Jackson    121      249.      2.06            6.26
 5   2021 00-0034750 R.Penny      119      229.      1.93            6.29
 6   2022 00-0036945 J.Fields      85      160.      1.88            6
 7   2022 00-0033357 T.Hill        96      178.      1.86            5.99
 8   2021 00-0034253 D.Hilliard    56      101.      1.80            6.25
 9   2022 00-0034750 R.Penny       57       99.2     1.74            6.07
10   2019 00-0034400 J.Wilkins     51       87.8     1.72            6.02
# ℹ 524 more rows
```

Figure 5

Interestingly enough, the top 5 only has one running back. Lamar Jackson and Kyler Murray are both quarterbacks, and topped the list with over 2 rushing yards over expected per play. Also noteworthy is that Rashaad Penny, a running back who appears twice in the top 10, is not regarded as one of the top backs in the league. This begs the question; is RYOE a reliable metric when analyzing player production? One way to answer this question is to reuse the "lagging" technique covered in Chapter 2 to look at how the RYOE metric from one year translates to the next year. By analyzing the correlation from one year to the next, we can make a general assumption about the stability of the RYOE metric.

Alongside RYOE between years, the example in the book compared "yards per carry" over different years. This is a very intuitive metric, and what that does not require any predictions or regressions, so it serves as a good comparison for who usable the RYOE metric actually is. The correlation values are as follows:



```
   ypc_corr ryoe_corr
      <dbl>     <dbl>
1     0.323     0.349
```

Figure 6

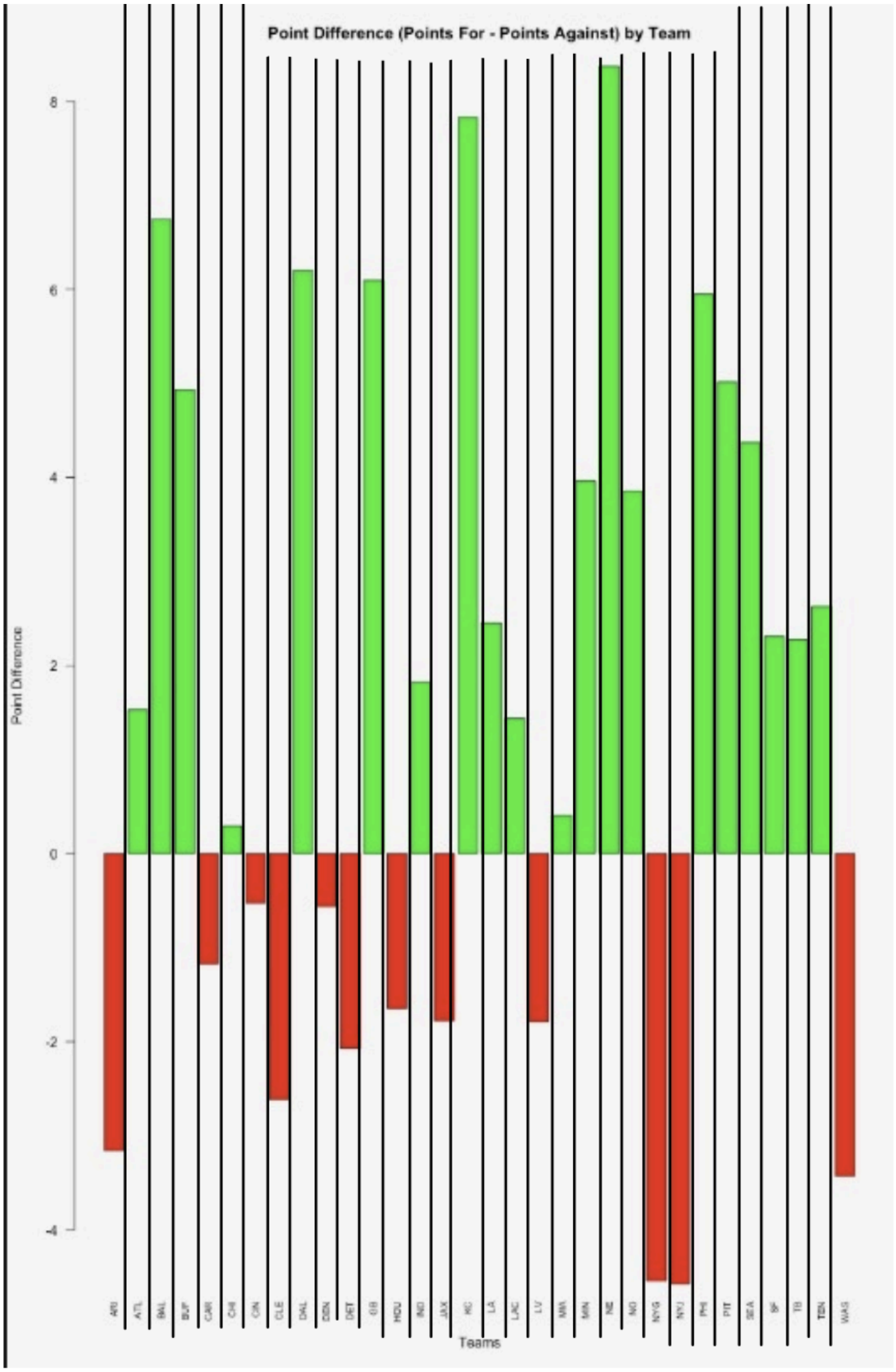
The correlation of rushing yards over expected is higher than that of yards per carry, so RYOE can be looked at as “slightly more stable” year to year as a statistic. Because yards per carry is generally thought of as a stable stat, we can conclude that rushing yards over expected is stable as well.

Conclusion

There is more content in this book in which I intend to further explore. Chapter 6, namely, is titled *Using Data Science for Sports Betting: Poisson Regression and Passing Touchdowns*, and having just learned about the Poisson distribution in MATH-0165, I feel I’ll be well versed to tackle that chapter and gain a deep understanding of the concept. As with the material I covered, I thought it was just challenging enough to learn new concepts, while at the same time being fun in an exploratory way where I could tweak certain metrics and find data for stuff that I was just inherently curious in (I included a personal project at the end of this document). I'm glad I chose this book to analyze because it is of great interest to me. Sometimes it didn’t even feel like I was doing work for a class project when working through this book, and that is the best way to do it when learning about topics that can be grueling. I look forward to maybe pursuing this in my future career!

Appendix

As mentioned in my conclusion paragraph, I ran a little personal project. I wanted a visualization of what teams have performed best at home in terms of point differential. So I was able to create a bar graph of all 32 teams over the last 5 years. Take a look below to see if your favorite team is positive or negative in point differential over the last 5 years! (You may need to zoom in. Teams are listed at the bottom).


Point Difference (Points For - Points Against) by Team
Point Difference
8
6
4
2
0
-2
-4
ARI
ATL
BAL
BUF
CAR
CHI
CIN
CLE
DAL
DEN
DET
GB
HOU
IND
JAX
KC
LA
LAC
LV
MIA
MIN
NE
NO
NYG
NYJ
PHI
PIT
SEA
SF
TB
TEN
WAS
Teams