Data Mining Project Write-Up

Data Summary

Our report is based on the dataset "NFL Combine- 2010 to 2023 Performance Data" sourced from *kaggle.com*. We immediately uploaded it into R as a dataframe, where we first cleaned the data before analyzing.

The original dataset has 15 fields: 'Year' of the combine assessment, 'Player', 'Pos' as position of the player, 'School', 'Height', 'Weight' in pounds, 'X40yd' time, 'Vertical' jump height in inches to test lower body explosiveness, 'Bench' repetitions, 'Broad.Jump' distance in inches, '3XCone' time, 'Shuttle' time, 'Round' picked, 'Pick' number, and 'Drafted' as a Boolean variable. After cleaning, we created a new column, 'height_total_in', to reformat the original height column to easily interpret as solely inches. And we changed the 'Drafted' field to be scored 1 (drafted) or 0 (not drafted) during that year.

Our main descriptive goal was to determine what variables about the players (height and weight) influenced specific results in the combine evaluation, and to understand the differences in results that might have prevented a player from being drafted. Our prescriptive goal throughout analyzing this data is to see if we can pull specific metrics that help players get drafted so players that weren't drafted can view that score/result as a goal to work towards if it's a weak spot in their skillset. Our predictive goal with the combine data is to perform linear regression to predict 3 Cone Drill times based on a player's Shuttle and 40yd time. We then picked a position of interest to us– wide receivers– and ran another linear regression model to see if we could predict how early they would get picked in a draft based on all the combine results.

Cleaning the Data

After downloading the data and uploading into R, we examined the dataset by looking at each field and observation to determine what data we need to draw from. We noticed immediately that many of the observations had missing values, and the height column was formatted as "feet-inches", rather than just a standard number measurement.

The first step towards creating our TIDY data frame was to rename the 'Broad.Jump' field, so when we ran SQL queries we wouldn't come across any errors. We renamed it to BroadJump by coding, "names(nfl_combine)[names(nfl_combine)=='Broad.Jump'] <- 'BroadJump'". The next step was to remove all the missing observations– " nfl_A <- na.omit(nfl_combine) ", which immediately created a dataframe that resulted in everyone that got drafted while keeping all the rows and deleting rows with null columns. We decided to remove the Round and Pick fields from nfl_A, as it wasn't necessary for our descriptive goal by coding " nfl_A$Round <- NULL " and " nfl_A$Pick <- NULL ".

We then created another dataframe dedicated to those not drafted, by creating a SQL query to simply get drafted observations observed as 'False', " nfl_B<-sqldf("SELECT * FROM nfl_combine WHERE Drafted = 'False'")" (SQL Query 1). nfl_B had missing observations in the Round and Pick field, so those columns immediately got deleted in the final dataframe. Finally, for nfl_B, we removed all missing values to return only players with every stat filled out. We decided to remove all missing values from both data frames because we wanted to focus on players that had results from every category, making our descriptive goals easier to comprehend. Using the code, " nfl_B<-na.omit(nfl_B) ", it's successfully cleaned. And we decided against inputting the mean or median into those missing values because it wouldn't show us accurate information about the players.

Next, we decided to create a new height column using inches, to easily compare and visualize as we continue to analyze those drafted and those who weren't. To accomplish this, we utilized the dyplr package and coded, " nfl_A <- nfl_A %>% separate(Height, into = c("feet", "inches"), sep = "-", convert = TRUE) %>% mutate(height_total_in = feet*12 + inches) ". The new column was named height_total_in, we did this to both nfl_A and nfl_B.

We feature engineered group positions to better gauge understanding position standings when creating visualizations. To do this we coded, " nfl_combine <- nfl_combine %>% mutate(PositionGroup = case_when(Pos %in% c("DE", "DL", "DT", "EDGE") ~ "DLINE",
Pos %in% c("LB", "ILB", "OLB") ~ "LB", Pos %in% c("CB", "S", "DB") ~ "DB",
Pos %in% c("RB", "FB") ~ "RB", Pos %in% c("OL", "OG","OT","C") ~ "OLINE",
TRUE ~ Pos)) ", to just recategorize the position players were in into a broader grouping: defensive line, linebacker, defensive back, running backs, and offensive line. Leaving alone quarterback, longsnappers, wide receivers, and tight ends.

After cleaning both datasets, we joined the newly cleaned nfl_A and nfl_B into nfl_C by using a set operation from the sqldf package, "nfl_C <- sqldf(' SELECT * FROM nfl_A UNION SELECT * FROM nfl_B ')" (SQL Query 2). When creating our descriptive visualizations, side by side comparisons could be drawn between drafted and undrafted since they were now joined in a new data frame.

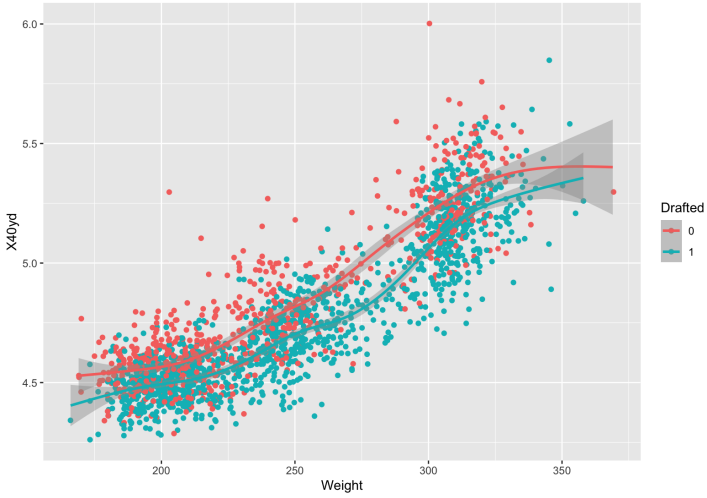
In our final step of cleaning, we altered the Drafted field to be filled with the binary codes of 1 and 0, rather than True and False. We did this by coding "nfl_C$Drafted[nfl_C$Drafted == 'True'] <- 1 " and "nfl_C$Drafted[nfl_C$Drafted == 'False'] <- 0 ". After this step, we were left with three datasets. nfl_A represents the results of the combine from those drafted. nfl_B represents the results of the combine from those not drafted. And finally, nfl_C is a joined dataset of nfl_A and nfl_B. All the datasets had the original columns but Round and Pick, and the addition of height_total_in.

Descriptive Goals with Visualizations

Visualization 1: Relationship of Weight and 40-Yard Times (Jittered Scatterplot)

The scatterplot shows relationships between player weight and 40 yard dash time, colored by whether they were drafted or not. Overall, heavier players run slower times, displaying an upward trend for both groups (drafted and not drafted). Drafted players (1) show faster times in the 200-250 weight range, with denser teal plots dominating the 4.5 range.

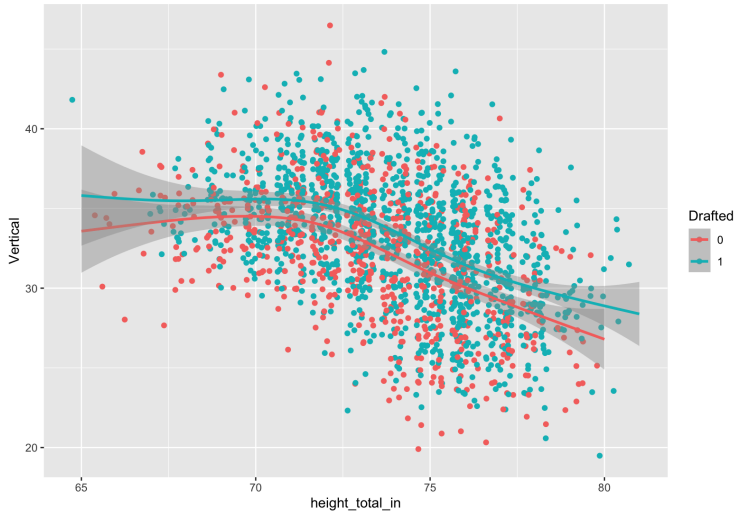
Code : ggplot(nfl_C, aes(x=Weight, y=X40yd,color=Drafted)) + geom_jitter()+ geom_smooth()



Visualization 2: Height v Vertical Jump (Jittered Scatterplot)

This scatterplot shows the player height and vertical jump performance, where athletes around 70-75 inches jumped highest compared to the 75-80 inch range. The trendline shows a slight decline in performance as height increases.

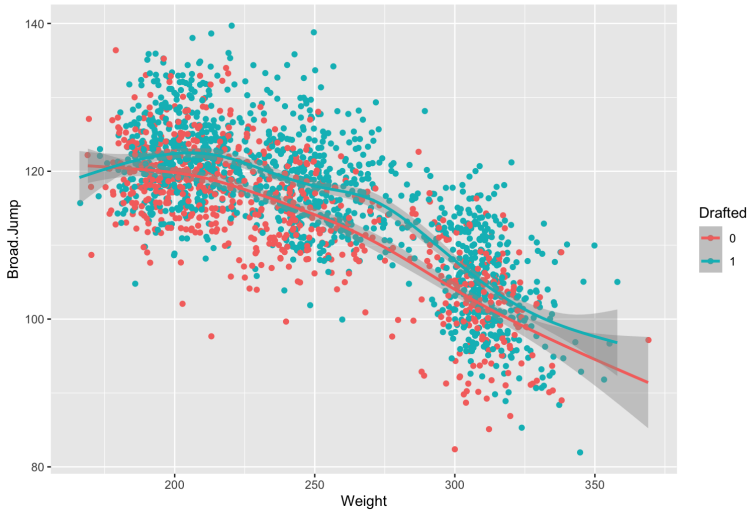
Code: ggplot(nfl_C, aes(x=height_total_in, y=Vertical,color=Drafted))+
  geom_jitter()+
  geom_smooth()



Visualization 3: Weight v Broad Jump Performance (Jittered Scatterplot)

The geom_smooth() allows us to see the clear negative relationship whereas as weight increases the broad jump performance declines. Drafted players (1) seem to perform better with its concentration of teal plots within the 120-140 range.

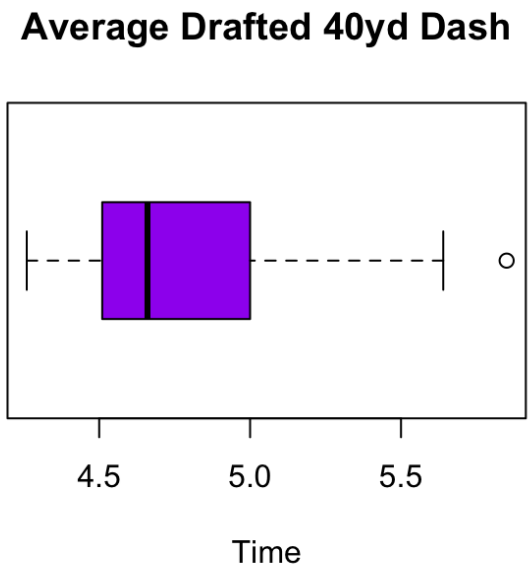
Code: ggplot(nfl_C, aes(x=Weight, y=BroadJump,color=Drafted))+
  geom_jitter() + geom_smooth()



Visualization 4: Average Drafted 40yd

The boxplot allows us to see the majority of drafted players ran between a 4.45 and 4.95. When we ran the summary of the 40yd column, we were able to confirm on average drafted players ran a 4.7. We later used this to compare times with undrafted players.

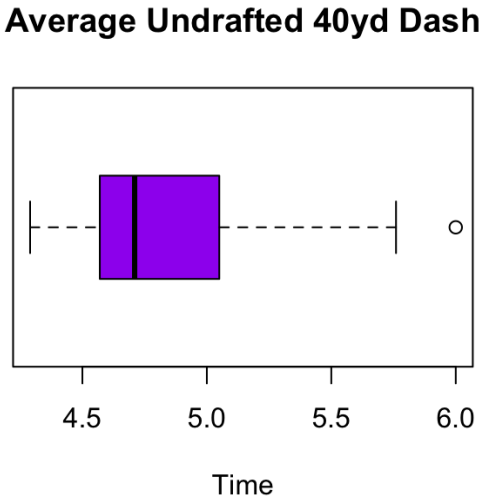
Code: boxplot(nfl_A$X40yd, main="Average Drafted 40yd Dash", xlab="Time", ylab="", col="purple", horizontal = TRUE)
summary(nfl_A$X40yd)



Visualization 5: Average Undrafted 40yd

We did the same with undrafted players, the boxplot shows similar results as drafted players. When we ran the summary command to get a closer look at the measures, it showed that the average was higher at 4.8. This tells us that 40yd isn't a main factor in determining drafts.

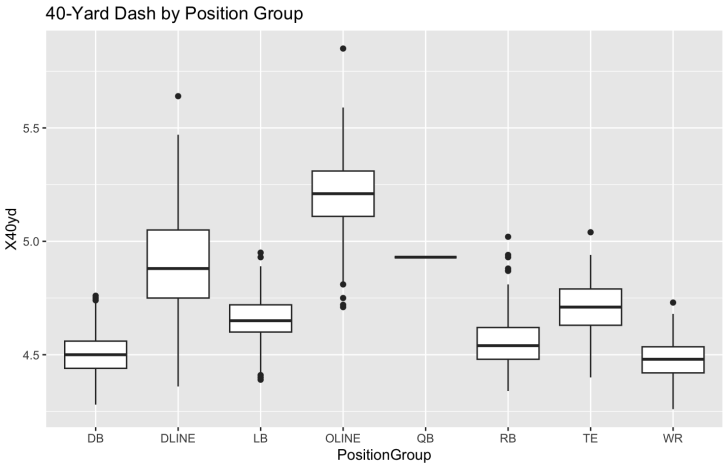
Code: boxplot(nfl_B$X40yd, main="Average Undrafted 40yd Dash", xlab="Time", ylab="", col="purple", horizontal = TRUE)
summary(nfl_B$X40yd)



Visualization 6: 40Yd Across NFL Positions (Boxplot)

The boxplot shows measures of centrality for each position. DBs and WRs show the fastest times with a median of 4.5, which reflects the need for speed in their positions. OLINE and DLINE seem to run the slowest reflecting their large physiques. We want to disclaim that for all boxplots using nfl_A, only one QB was left after cleaning, because as years increased drafted QBs weren't tested in the combine, causing missing observations.

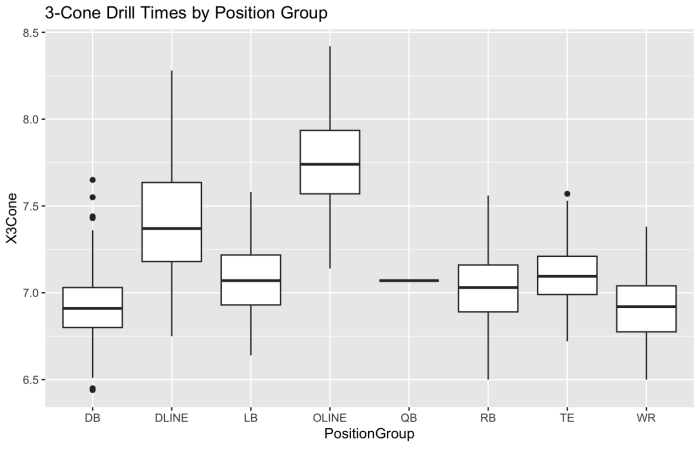
Code: ggplot(nfl_A, aes(PositionGroup, X40yd)) +
geom_boxplot() +
labs(title = "40-Yard Dash by Position Group")



Visualization 7: 3 Cone Drill Times Across NFL Positions (Boxplot)

The boxplot compares the 3 Cone times across different positions, with DB resulting in the fastest times with a median of about 6.8, which reflects their skill of being an agile and flexible player.

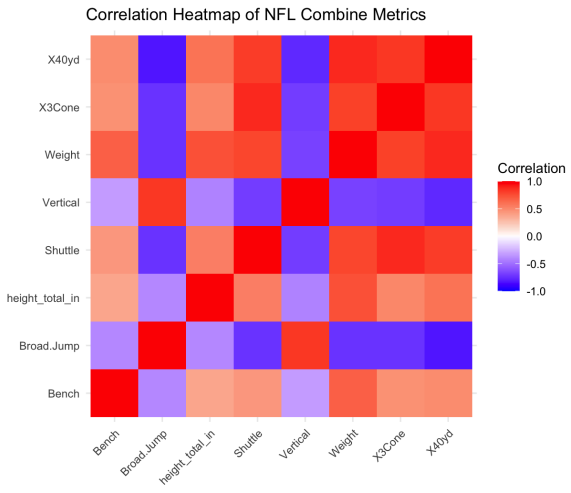
Code: ggplot(nfl_A, aes(PositionGroup, X3Cone)) +
geom_boxplot() +
labs(title = "3-Cone Drill Times by Position Group")



Visualization 8: Relationships Among NFL Combine Metrics (Heatmap)

Red in the heatmap indicates a strong positive correlation, purple indicates a strong negative correlation, and white indicates no correlation. Several patterns emerge, showing X40yd, Shuttle, and X3Cone are highly correlated due to speed. Weight shows strong negative correlations with explosiveness drills (Vertical, BroadJump).

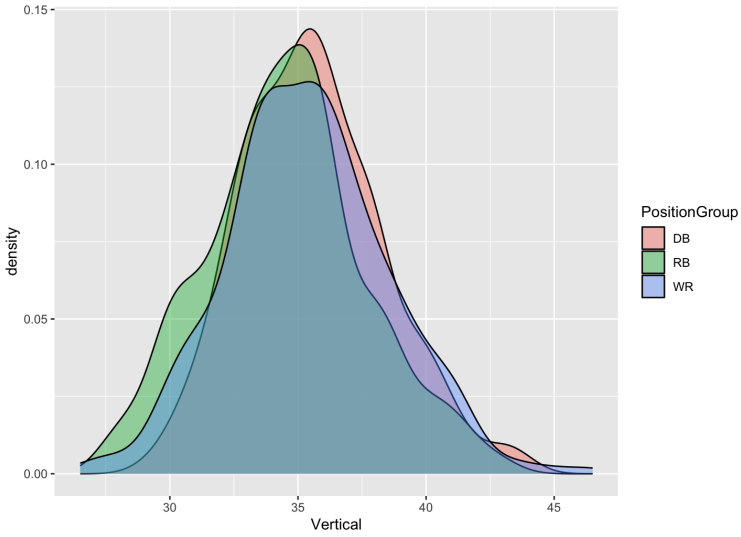
Code: ggplot(cor_df, aes(x = Var1, y = Var2, fill = Correlation)) + geom_tile() + scale_fill_gradient2(limits = c(-1, 1), midpoint = 0, low = "blue", mid = "white", high = "red") + coord_fixed() + theme_minimal() + theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)) + labs(title = "Correlation Heatmap of NFL Combine Metrics", x = NULL, y = NULL)



Visualization 9: Vertical Distribution for DBs, RBs, and WRs (Density Plot)

The three groups chosen are known to be explosive positions, as the vertical tests explosiveness. The density plot shows similar distributions across all three, with peaks at around 35 inches reflecting the explosiveness required for the positions. With DB showing a slightly higher concentration peaking above 35, consistent with their need for covering longer distances. We subsetted nfl_C and filter only the DBs, RBs, and WRs to visualize. (SQL Query 3)

Code: nfl_subset1 <- sqldf("SELECT * FROM nfl_C WHERE PositionGroup IN ('WR', 'DB', 'RB')")
ggplot(nfl_subset1, aes(Vertical, fill =PositionGroup)) + geom_density(alpha = 0.4)



SQL Queries

Query 4: How many lost observations?

After cleaning our data and running the visualizations, we noticed there was only one drafted quarterback due to the fact that consistently drafted quarterbacks weren't expected to run all tests causing missing values and being cleaned out. We ran two SQL queries to figure out how many observations were lost by counting players in each PositionGroup from nfl_combine and nfl_C. This resulted in the following tables to the left (left is nfl_combine, right is nfl_C). In total we lost 2,717 observations and were reduced to only nine positions post-clean (removed kicker and punter). This is important to note because there's only six quarterbacks in nfl_B and one in nfl_C, skewing data. (results to right)

```
sqldf("SELECT PositionGroup, COUNT(*) AS Count
       FROM nfl_combine
       GROUP BY PositionGroup;")

sqldf("SELECT PositionGroup, COUNT(*) AS Count
       FROM nfl_C
       GROUP BY PositionGroup;")
```

	PositionGroup	Count
1	DB	880
2	DLINE	766
3	K	65
4	LB	530
5	LS	22
6	OLINE	755
7	P	81
8	QB	238
9	RB	468
10	TE	266
11	WR	670

	PositionGroup	Count
1	DB	372
2	DLINE	357
3	LB	235
4	LS	6
5	OLINE	403
6	QB	7
7	RB	211
8	TE	125
9	WR	308

Query 5: Summary Stats with for Speed and Agility

For our prescriptive goal, helping players get drafted, we decided to run a query that shows the minimum, average, and maximum results drafted players achieved for the 40yd, Vertical, in 3Cone. Allowing undrafted players a goal to work towards if they score lower.

```
summary_stats<-sqldf("
  SELECT
    MIN(X40yd),
    MAX(X40yd),
    AVG(X40yd),
    MIN(Vertical),
    MAX(Vertical),
    AVG(Vertical),
    MIN(X3Cone),
    MAX(X3Cone),
    AVG(X3Cone)
  FROM nfl_C;
")
```

Query 6: Average Strength Between NFL Positions

To continue our prescriptive goal with providing players stats that can help them gauge a range they should shoot for. For this query, we ran the average bench reps for each PositionGroup allowing specific players to hone in on a result catered to their position.

```
sqldf("
  SELECT PositionGroup, AVG(Bench) AS AvgBench
  FROM nfl_A
  GROUP BY PositionGroup
  ORDER BY AvgBench DESC;
")
```

Query 7: Comparing Average Metrics Across RBs, WRs, DBs

As a group, we determined that the runningbacks, wide receivers, and defensive backs are a few of the most important positions. By averaging their scores we can measure a variety of skills about them and compare them against each other. 40yd average tells us the straight speed, vertical measures the lower

```
sqldf("
  SELECT PositionGroup,
         AVG(X40yd) AS Avg40,
         AVG(Vertical) AS AvgVertical,
         AVG(Bench) AS AvgBench,
         AVG(Shuttle) AS AvgShuttle,
         AVG(X3Cone) AS AvgThreeCone
  FROM nfl_C
  WHERE PositionGroup IN ('RB', 'WR', 'DB')
  GROUP BY PositionGroup;
")
```

body explosiveness, bench reps display upper body strength, shuttle run measures sprint agility, and 3cone covers ability to change in direction. The query allows us to see that DBs and WRs tend to have faster 40yd and vertical jumps, reflecting the need for speed and explosiveness. While RBs have higher bench press, reflecting the demand in strength.

PositionGroup	Avg40	AvgVertical	AvgBench	AvgShuttle	AvgThreeCone
DB	4.521398	35.59409	15.11290	4.194704	6.939167
RB	4.594597	34.52370	19.85308	4.297393	7.080474
WR	4.513409	35.39286	14.19481	4.255519	6.950909

Query 8: Fastest OLINE and DLINE Players

We began to focus on other positions, such as OLINE and DLINE. Because they're not known for speed we created a list of the fastest players based on the 40yd. Concluding OLINE players are typically fasters. The query results with the fastest player to be undrafted, showing that there are many variables that come into play when a player is decided to be drafted or not.

```
sqldf("
      SELECT Player, PositionGroup, Drafted,X40yd
      FROM nfl_C
      WHERE PositionGroup IN ('OLINE','DLINE')
      ORDER BY PositionGroup DESC
      LIMIT 15
")
```

Query 9: Which PositionGroup is overall most explosive?

Earlier when creating Visualization #7, we made the assumption that DBs, RBs, and WRs were most explosive in order to create our density plot. This query allows us to prove whether or not our assumption is true. By averaging the vertical and broad jump scores, we can see which PositionGroup scores highest. The result of the query confirms our assumption with WR most explosive, followed by DB and RB. (results below)

```
sqldf("
  SELECT PositionGroup,
         AVG(Vertical) AS AvgVertical,
         AVG(BroadJump) AS AvgBroad
  FROM nfl_A
  GROUP BY PositionGroup
  ORDER BY AvgVertical DESC;
")
```

	PositionGroup	AvgVertical	AvgBroad
1	WR	35.98830	122.3333
2	DB	35.98367	122.5673
3	RB	35.03306	119.6942

Query 10: Creating Athletic Score for 2023

When we first began this project we initially wanted to create a score for each player based on their combine results, helping us gauge if higher results meant being drafted. That way we have a metric to base our predictions on. Although we steered away from it, and rather just changed the observations in Drafted to 1 or 0. The query, though, allowed us to make an arbitrary score for the year 2023. By combining the combine results, we created a score for each player. The player with the highest score was Julius Brents, a drafted wide receiver with a score of 219.95.

```
sqldf("
  SELECT Player, PositionGroup,Drafted,Year,
         ( (1/X40yd) * 100
           + Vertical
           + BroadJump
           + (25 - X3Cone)
         ) AS AthleticScore
  FROM nfl_combine
  WHERE Year=2023
  ORDER BY AthleticScore DESC
  LIMIT 15;
")
```

Query 11: Explosiveness Scores

After creating a general score, we decided to hone in on a specific skill, explosiveness, which we have been looking at consistently. We generated scores for this by simply adding the Vertical and Broad Jump scores. Overall throughout 2010-2023, the most explosive player was Chris Conley, a drafted wide receiver.

```
sqldf("
  SELECT Player, PositionGroup,Drafted,
         (Vertical) + BroadJump AS ExplosivenessScore
  FROM nfl_C
  ORDER BY ExplosivenessScore DESC
  LIMIT 20;
")
```

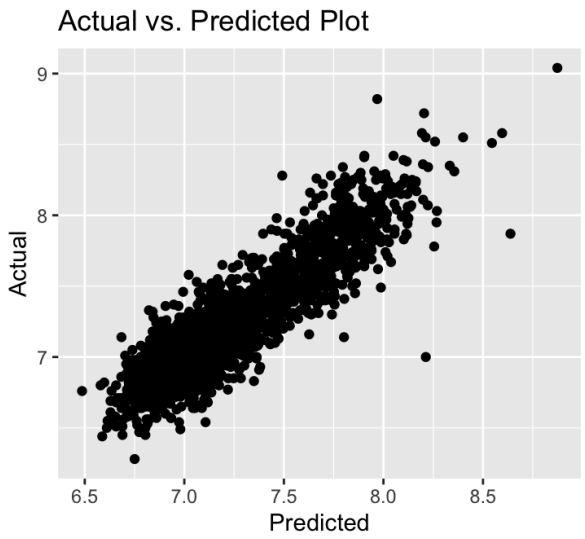
Predictive Goals with Visualizations

For our predictive analysis, we create a linear regression model predicting 3Cone time using the 40yd and Shuttle time. When we use the summary command, we're able to interpret the residuals and coefficients. The residuals tell us that the predictions are very accurate due to a very small IQR of about .22. The coefficients table tells us for every one second increase in 40yd, the player's 3Cone time increases by 0.46657 seconds. The positive coefficient reflects a moderate positive correlation between 3Cone and 40yd, so slower (higher) 40yd time means slower (higher) 3Cone time and vice versa. The shuttle coefficient tells us for every one second increase in shuttle time, 3Cone time increases by 0.91874 seconds. This also tells us 3Cone and Shuttle are strongly positively correlated, being nearly equal to 1. Both variables indicate high significance based on the p value. The residual standard error tells us our model is off by 0.179 seconds, extremely accurate for athletic timing. R squared tells us that about 81% of any variation in the 3Cone performance can be explained by the 40yd and shuttle time, showing a high predictive power.

Visualization 10: Actual v Predicted Plot - 3Cone, Shuttle, and 40yd

We created a dataframe with the Predict 3Cone values and the Actual values, then plotted them to visually check the fit of the model. As you can see in the graph, the plots fall close into a line revealing the model is accurate.

Code: ggplot(data = data.frame(fitted = model$fitted.values, residuals = nfl_C$X3Cone))+
geom_point(aes(x=fitted, y=residuals))+
labs(x="Predicted",y="Actual",title="Actual vs. Predicted Plot")



Queries 12 and 13:

We utilized a LEFT JOIN to add Round and Pick fields into nfl_C from nfl_combine, to create a new dataframe called nfl_drafted. We then created another query that filters nfl_drafted to display only wide receivers. Creating a new drafting model for WR specific players.

```
#use a query to join nfl_C with nfl_combine Round and Pick
nfl_drafted <- sqldf("
  SELECT a.*, nc.Round, nc.Pick
  FROM nfl_A a
  LEFT JOIN (SELECT Pos, Player, Round, Pick
             FROM nfl_combine) nc
  ON nc.Pos = a.Pos AND nc.Player = a.Player;
")

#select only WRs (for analysis later)
nfl_drafted_wr <- sqldf("
  SELECT *
  FROM nfl_drafted
  WHERE Pos='WR'")
```

Visualization 11: WR Picks

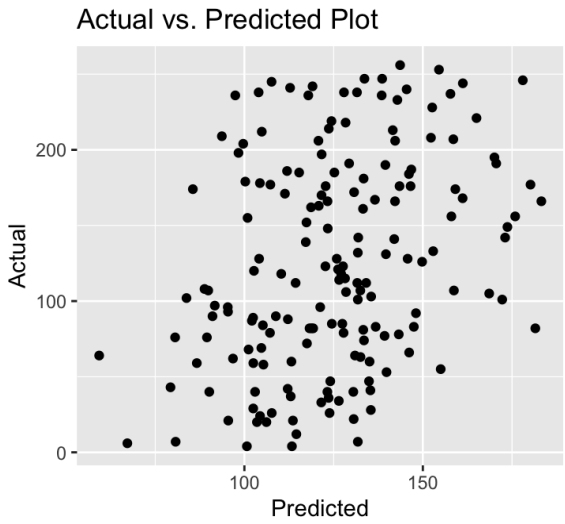
We ran another model to predict when WRs would get picked based on all their combine metrics. The model also answers which metric best predicts how early a WR gets drafted. When we run the summary command and view the coefficients table we see that 40yd is the only major significant coefficient.

```
#WR draft model - what pick overall would they be based off their metrics
draft_model <- lm(Pick ~ Weight+X40yd+Vertical+Bench+BroadJump+X3Cone+Shuttle, data=nfl_drafted_wr)

summary(draft_model)
```

We then visualize the draft_model. The points are widely scattered, telling us this model has low predictive power and accuracy as there is no clear trend. We can conclude that all the combine metrics, together, don't strongly determine the WR draft position.

Code: ggplot(data = data.frame(fitted = draft_model$fitted.values,
residuals = nfl_drafted_wr$Pick)) +
geom_point(aes(x = fitted, y = residuals)) +
labs(x = "Predicted", y = "Actual", title= "Actual vs. Predicted Plot")



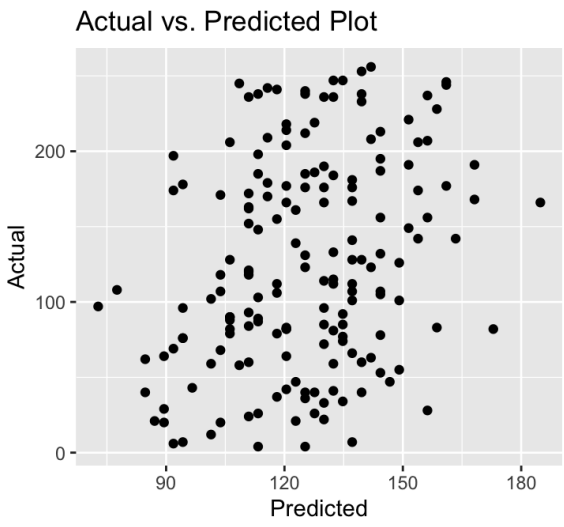
Visualization 12:

Once we determined 40yd was a main factor in drafting we ran a final prediction model. Creating a linear regression model predicting draft pick based only on the 40yd time. The residuals tell us the model shows low accuracy with about 50 picks from being drafted in either direction. The coefficient of 40yd time tells us that every 1 second increase (longer time), the draft pick will increase by about 238 picks. The residual standard error tells us our predictions are off by about 67 draft picks, which is very large and therefore a poor predictive model.

```
#ONLY LOOKING AT 40YD
draft_model40 <- lm(Pick ~ X40yd, data=nfl_drafted_wr)
summary(draft_model40)
```

Once we visualize the model, we can see that the points also don't form a clear linear pattern. Actual draft picks widely vary, telling us that 40yd alone can't be a reliable predictor to draft outcomes. There is a very weak, slight upward trend telling us faster WRs get drafted a bit earlier and vice versa.

Code:
```
ggplot(data = data.frame(fitted = draft_model40$fitted.values,
                         residuals = nfl_drafted_wr$Pick)) +
  geom_point(aes(x = fitted, y = residuals)) +
  labs(x = "Predicted", y = "Actual", title= "Actual vs. Predicted Plot")
```



Conclusion

This project has allowed us to explore the NFL Combine scores and fulfill our descriptive, prescriptive, and predictive goals. Our descriptive visualizations allowed us to identify clear patterns such as heavier positions run slower times in speed drills, with the heat map confirming by showing negative correlation with Weight and Shuttle time, for example. Our predictive model allows us to gain deeper insight on how specific combine metrics can forecast others. The 3Cone model utilized 40yd and Shuttle as predictors, resulting in a well fit model showing strong correlations between the three. In contrast, our other prediction models for draft outcomes resulted in low predictive power. Altogether, our results tell us that while combine metrics are strong indicators of athletic performance, they aren't strong predictors of draft positions. Draft decisions incorporate other qualitative factors such as team needs, medical evaluations, scheme fit, positional fit, or film performance. Future analysis could integrate college production, recent injuries, or team-by-team drafting trends to build a complete predictive model. Overall the project has confirmed wide receivers, defensive backs, and running backs are the most explosive positions in a team. Confirmed with SQL query #9, Visualization #9, and Visualizations #7 and #8.