PBDAA Final Project
Jason & Patricia

Project Info

Project Name: 40 Yard Dash Times and Draft Placement for NFL Skill Position Players

Team Members: Jason Tran & Patricia Guirao

Topic Description: This project application will analyze and compare the 40-yard dash times of Skill Position Players (Quarterbacks, Wide Receivers, Tight Ends and Running Backs) at the NFL Combine with where they were drafted in the NFL Draft

Importance: Insight from data can help determine whether faster times generally correlate with higher draft picks and how that is influenced by position (ex: how influential is 40 yard time for QB's Draft position vs Wide Receiver's vs Running Back), useful for NFL Teams and College Players

Goodness

Credibility: The results of our application will be accurate and can be trusted because the dataset is sourced directly from credible entities like the NFL and Pro Football Reference, a historical database of all NFL Stats and Information.

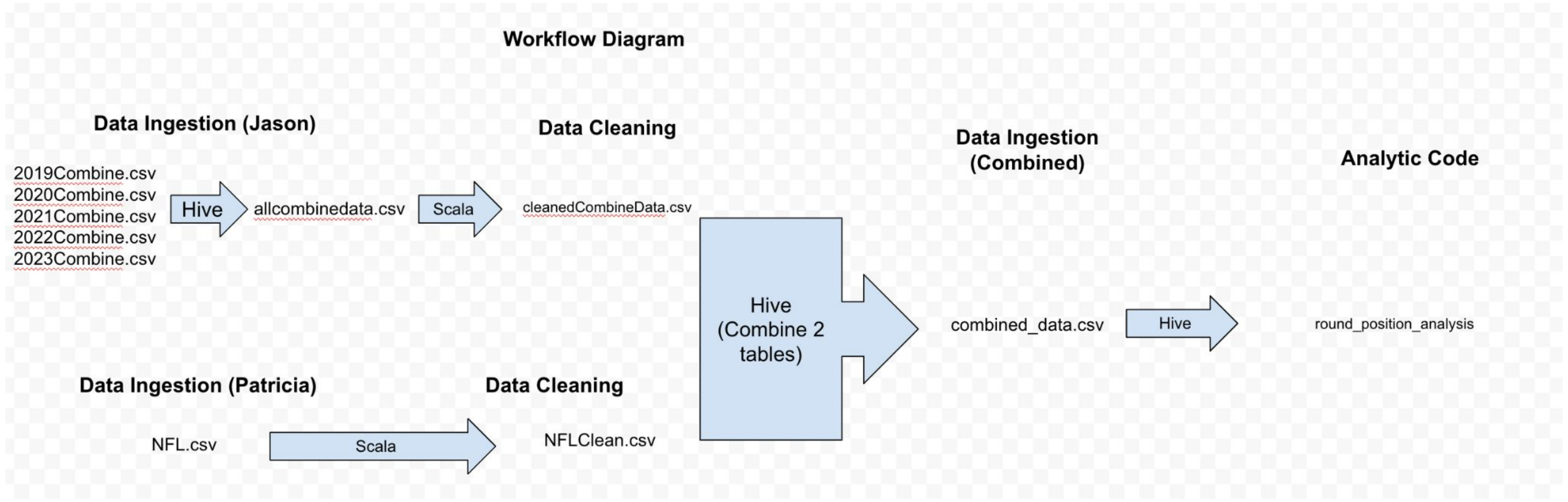
Furthermore, the NFL Combine results are to an extent objective. The 40 Yard Dash Times are laser-timed and not hand timed, ensuring objective and precise measurements. We will also be able to compare and ensure that the results of our findings will be accurate by verifying our results to the comprehensive statistical analysis from the research paper "Predictive Validity of the Physical Skills Test of the 40-Yard Dash and Draft Placement in the NFL Draft" found in the Sport Journal by Raymond Tucker and Willie Black — a very similar study.

Data Sources

Data Source 1: Pro Football Reference (https://www.pro-football-reference.com/)
Description: The data sets display all combine results of participant of the 2019 to 2023 Combine.
Schema: Player,Pos,School,College,Ht,Wt,40yd,Vertical,Bench,Broad Jump,3ConeShuttle,Drafted (tm/rnd/yr)
Size of Data: 100KB
Link: https://www.pro-football-reference.com/draft/2023-combine.htm
https://www.pro-football-reference.com/draft/2022-combine.htm
https://www.pro-football-reference.com/draft/2021-combine.htm
https://www.pro-football-reference.com/draft/2020-combine.htm
https://www.pro-football-reference.com/draft/2019-combine.htm

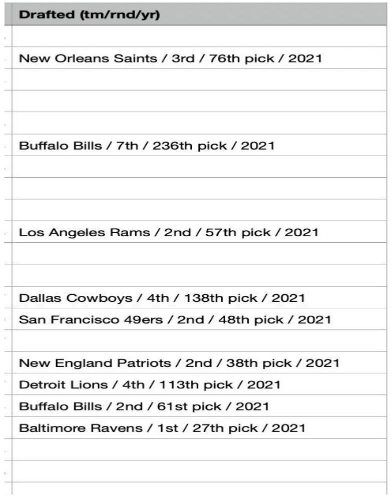
Data Source 2: Kaggle
Description: Combine Results of participants in the NFL Combine from the 2009 Combine - 2019 Combine
Schema: Year,Player,Age,School,Height,Weight,Sprint_40yd,Vertical_Jump,Bench_Press_Reps,Broad_J
ump,Agility_3cone,Shuttle,Drafted..tm.rnd.yr.,BMI,Player_Type,Position_Type,Position,Drafted
Size of Data: 500KB
Link: https://www.kaggle.com/code/redlineracer/nfl-scouting-combine/input

Diagram of Workflow

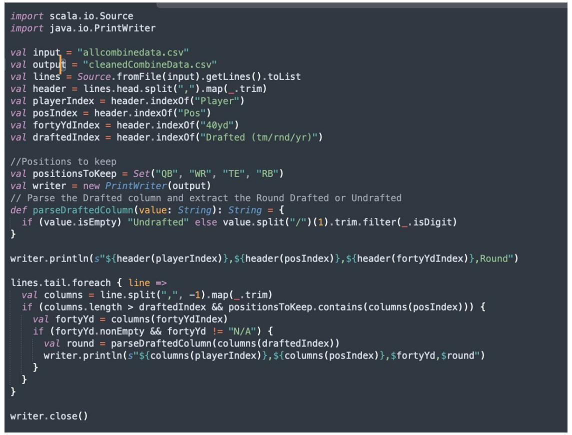


Code Challenge

Biggest Challenge: Separating the “Drafted” Column into getting just the “Round”

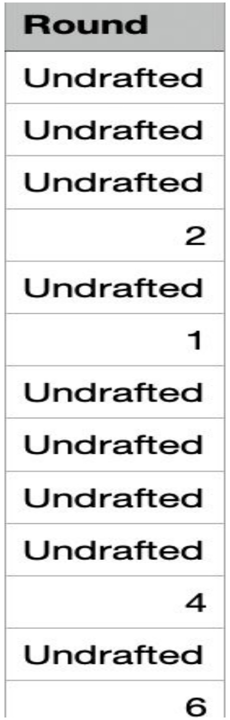


Drafted (tm/rnd/yr)
New Orleans Saints / 3rd / 76th pick / 2021
Buffalo Bills / 7th / 236th pick / 2021
Los Angeles Rams / 2nd / 57th pick / 2021
Dallas Cowboys / 4th / 138th pick / 2021
San Francisco 49ers / 2nd / 48th pick / 2021
New England Patriots / 2nd / 38th pick / 2021
Detroit Lions / 4th / 113th pick / 2021
Buffalo Bills / 2nd / 61st pick / 2021
Baltimore Ravens / 1st / 27th pick / 2021



```scala
import scala.io.Source
import java.io.PrintWriter

val input = "allcombinedata.csv"
val output = "cleanedCombineData.csv"
val lines = Source.fromFile(input).getLines().toList
val header = lines.head.split(",").map(_.trim)
val playerIndex = header.indexOf("Player")
val posIndex = header.indexOf("Pos")
val fortyYdIndex = header.indexOf("40yd")
val draftedIndex = header.indexOf("Drafted (tm/rnd/yr)")

//Positions to keep
val positionsToKeep = Set("QB", "WR", "TE", "RB")
val writer = new PrintWriter(output)
// Parse the Drafted column and extract the Round Drafted or Undrafted
def parseDraftedColumn(value: String): String = {
  if (value.isEmpty) "Undrafted" else value.split("/")(1).trim.filter(_.isDigit)
}

writer.println(s"${header(playerIndex)},${header(posIndex)},${header(fortyYdIndex)},Round")

lines.tail.foreach { line =>
  val columns = line.split(",", -1).map(_.trim)
  if (columns.length > draftedIndex && positionsToKeep.contains(columns(posIndex))) {
    val fortyYd = columns(fortyYdIndex)
    if (fortyYd.nonEmpty && fortyYd != "N/A") {
      val round = parseDraftedColumn(columns(draftedIndex))
      writer.println(s"${columns(playerIndex)},${columns(posIndex)},$fortyYd,$round")
    }
  }
}

writer.close()
```

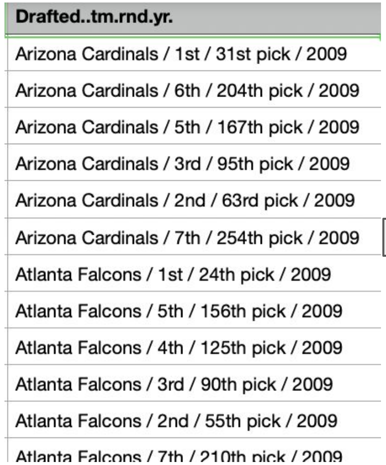
Jason Scala Cleaning Code



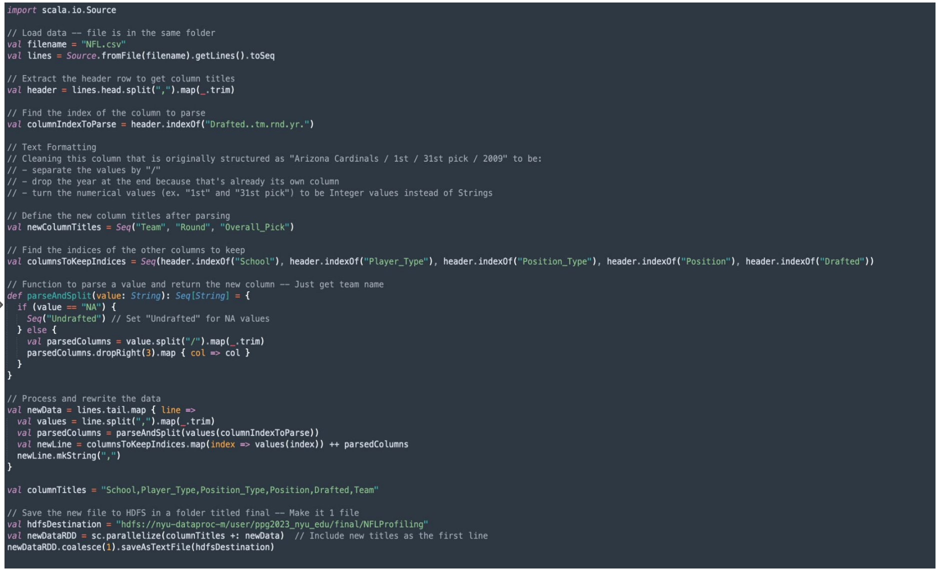
Round
Undrafted
Undrafted
Undrafted
2
Undrafted
1
Undrafted
Undrafted
Undrafted
Undrafted
4
Undrafted
6

Code Challenge

Biggest Challenge: Separating the “Drafted” Column into getting just the “Round”

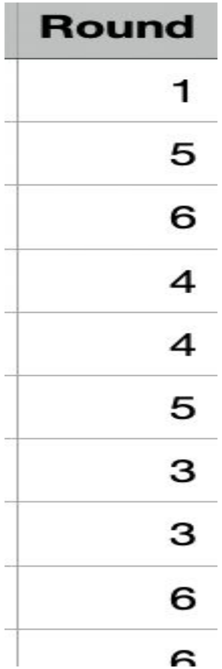


Drafted..tm.rnd.yr.
Arizona Cardinals / 1st / 31st pick / 2009
Arizona Cardinals / 6th / 204th pick / 2009
Arizona Cardinals / 5th / 167th pick / 2009
Arizona Cardinals / 3rd / 95th pick / 2009
Arizona Cardinals / 2nd / 63rd pick / 2009
Arizona Cardinals / 7th / 254th pick / 2009
Atlanta Falcons / 1st / 24th pick / 2009
Atlanta Falcons / 5th / 156th pick / 2009
Atlanta Falcons / 4th / 125th pick / 2009
Atlanta Falcons / 3rd / 90th pick / 2009
Atlanta Falcons / 2nd / 55th pick / 2009
Atlanta Falcons / 7th / 210th pick / 2009



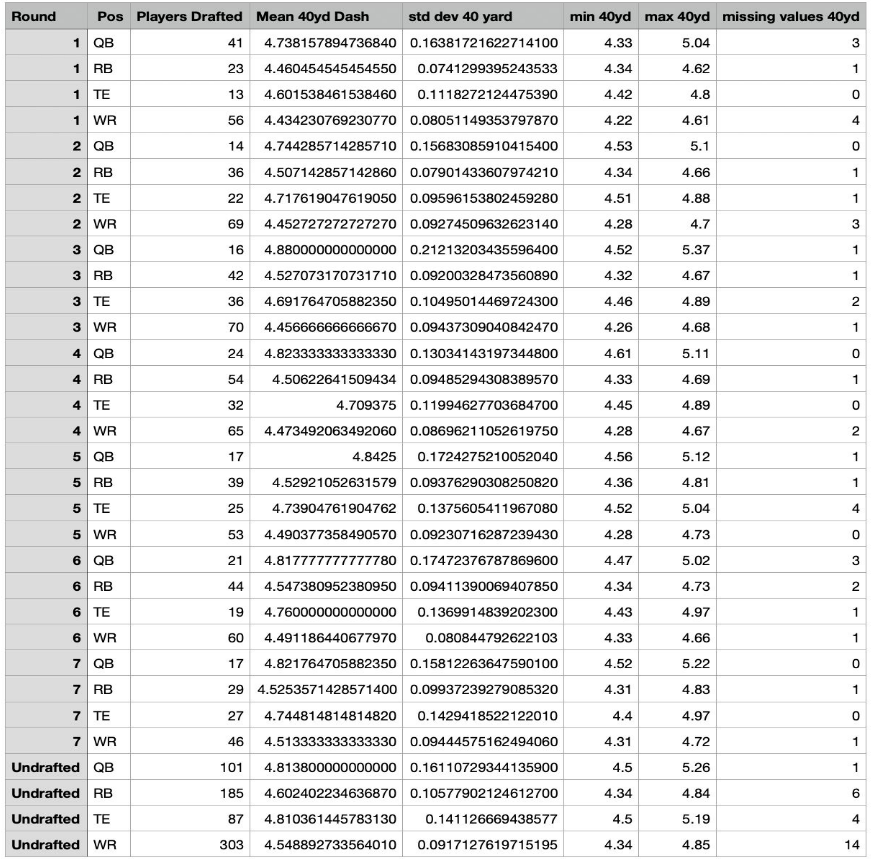
```scala
import scala.io.Source

// Load data -- file is in the same folder
val filename = "NFL.csv"
val lines = Source.fromFile(filename).getLines().toSeq

// Extract the header row to get column titles
val header = lines.head.split(",").map(_.trim)

// Find the index of the column to parse
val columnIndexToParse = header.indexOf("Drafted..tm.rnd.yr.")

// Text Formatting
// Cleaning this column that is originally structured as "Arizona Cardinals / 1st / 31st pick / 2009" to be:
// - separate the values by "/"
// - drop the year at the end because that's already its own column
// - turn the numerical values (ex. "1st" and "31st pick") to be Integer values instead of Strings

// Define the new column titles after parsing
val newColumnTitles = Seq("Team", "Round", "Overall_Pick")

// Find the indices of the other columns to keep
val columnsToKeepIndices = Seq(header.indexOf("School"), header.indexOf("Player_Type"), header.indexOf("Position_Type"), header.indexOf("Position"), header.indexOf("Drafted"))

// Function to parse a value and return the new column -- Just get team name
def parseAndSplit(value: String): Seq[String] = {
  if (value == "NA") {
    Seq("Undrafted") // Set "Undrafted" for NA values
  } else {
    val parsedColumns = value.split("/").map(_.trim)
    parsedColumns.dropRight(3).map { col => col }
  }
}

// Process and rewrite the data
val newData = lines.tail.map { line =>
  val values = line.split(",").map(_.trim)
  val parsedColumns = parseAndSplit(values(columnIndexToParse))
  val newLine = columnsToKeepIndices.map(index => values(index)) ++ parsedColumns
  newLine.mkString(",")
}

val columnTitles = "School,Player_Type,Position_Type,Position,Drafted,Team"

// Save the new file to HDFS in a folder titled final -- Make it 1 file
val hdfsDestination = "hdfs://nyu-dataproc-m/user/ppg2023_nyu_edu/final/NFLProfiling"
val newDataRDD = sc.parallelize(columnTitles +: newData)  // Include new titles as the first line
newDataRDD.coalesce(1).saveAsTextFile(hdfsDestination)
```

Patricia Scala Cleaning Code



Round
1
5
6
4
4
5
3
3
6
6

Insights



Round	Pos	Players Drafted	Mean 40yd Dash	std dev 40 yard	min 40yd	max 40yd	missing values 40yd
1	QB	41	4.738157894736840	0.16381721622714100	4.33	5.04	3
1	RB	23	4.460454545454550	0.0741299395243533	4.34	4.62	1
1	TE	13	4.601538461538460	0.1118272124475390	4.42	4.8	0
1	WR	56	4.434230769230770	0.08051149353797870	4.22	4.61	4
2	QB	14	4.744285714285710	0.15683085910415400	4.53	5.1	0
2	RB	36	4.507142857142860	0.07901433607974210	4.34	4.66	1
2	TE	22	4.717619047619050	0.09596153802459280	4.51	4.88	1
2	WR	69	4.452727272727270	0.09274509632623140	4.28	4.7	3
3	QB	16	4.880000000000000	0.21213203435596400	4.52	5.37	1
3	RB	42	4.527073170731710	0.09200328473560890	4.32	4.67	1
3	TE	36	4.691764705882350	0.10495014469724300	4.46	4.89	2
3	WR	70	4.456666666666670	0.09437309040842470	4.26	4.68	1
4	QB	24	4.823333333333330	0.13034143197344800	4.61	5.11	0
4	RB	54	4.50622641509434	0.09485294308389570	4.33	4.69	1
4	TE	32	4.709375	0.11994627703684700	4.45	4.89	0
4	WR	65	4.473492063492060	0.08696211052619750	4.28	4.67	2
5	QB	17	4.8425	0.1724275210052040	4.56	5.12	1
5	RB	39	4.52921052631579	0.09376290308250820	4.36	4.81	1
5	TE	25	4.73904761904762	0.1375605411967080	4.52	5.04	4
5	WR	53	4.490377358490570	0.09230716287239430	4.28	4.73	0
6	QB	21	4.817777777777780	0.17472376787869600	4.47	5.02	3
6	RB	44	4.547380952380950	0.09411390069407850	4.34	4.73	2
6	TE	19	4.760000000000000	0.1369914839202300	4.43	4.97	1
6	WR	60	4.491186440677970	0.080844792622103	4.33	4.66	1
7	QB	17	4.821764705882350	0.15812263647590100	4.52	5.22	0
7	RB	29	4.5253571428571400	0.09937239279085320	4.31	4.83	1
7	TE	27	4.744814814814820	0.1429418522122010	4.4	4.97	0
7	WR	46	4.513333333333330	0.09444575162494060	4.31	4.72	1
Undrafted	QB	101	4.813800000000000	0.16110729344135900	4.5	5.26	1
Undrafted	RB	185	4.602402234636870	0.10577902124612700	4.34	4.84	6
Undrafted	TE	87	4.810361445783130	0.141126669438577	4.5	5.19	4
Undrafted	WR	303	4.548892733564010	0.0917127619715195	4.34	4.85	14

Insights

Wide Receivers (WR): Mean 40-yard times range from 4.43 to 4.49 seconds. Faster speeds in early rounds (4.43 seconds in Round 1) highlight speed's importance for draft selection.

Running Backs (RB): Show a correlation between speed and draft round, with earlier round picks having faster times (mean of 4.46 seconds in Round 1).

Quarterbacks (QB) & Tight Ends (TE): Less emphasis on speed. QBs show a wider range of acceptable times (mean of 4.74 to 4.84 seconds).

Undrafted Players: Slower mean times (e.g., WRs at 4.54 seconds) suggest speed's role in affecting draft chances, especially for WRs and RBs.

Takeaway:
WRs and RBs show a clear trend where faster 40-yard dash times are favored in higher draft rounds, while QBs and TEs are evaluated with less emphasis on speed.

References

Research Papers:
https://thesportjournal.org/article/predictive-validity-of-the-physical-skills-test-of-the-40-yard-dash-and-draft-placement-in-the-nfl-draft/
https://scholarworks.calstate.edu/downloads/1z40kt64t

Dataset Sources: Pro-Football-Reference & Kaggle

Links:
https://www.pro-football-reference.com/draft/2023-combine.htm
https://www.pro-football-reference.com/draft/2022-combine.htm
https://www.pro-football-reference.com/draft/2021-combine.htm
https://www.pro-football-reference.com/draft/2020-combine.htm
https://www.pro-football-reference.com/draft/2019-combine.htm
https://www.kaggle.com/code/redlineracer/nfl-scouting-combine/input