README for the data preparation part

Group 2

December, 2nd, 2019

Abstract

This Chapter is describes the process from raw to clean data, as well as the data sources. In other words it explains the scripts 'DataCleaning2.R' and 'functionGetCleanClass2.R' and therefore also the function 'getCleanClass2'.

Description of the Goal

We want to predict wheter a college football Quarterback (QB), Running Back (RB) or Wide Receiver (WR) will be drafted into the NFL or not. For this purpose we try to combine data from different sources. This starts with game data from College Football and will be extended with further information like the NFL Combine or the Pro Day. Since we want to apply supervised learning, we would also need a 'Y', which contains the information, if a player was drafted or not.

Data Source

Most of our data was uploaded some years ago to Kaggle (https://www.kaggle.com/mhixon/college-football-statistics), but has no relevant scprips that were made of it and try to predict the NFL Draft. These Datasets contain much more information about college football, than we would need. We only use following data sets for all the years:

- player-game-statistics.csv
  - One observation in these files contains the information about one player in one game.
- player.csv
  - One observation in these files contains information about height, weight, schools etc. of one player

The best data sets (in order of lenght) about the NFL Combine and the Pro Day we found, is also one from Kaggle (https://www.kaggle.com/kbanta11/nfl-combine). Unfortunately it turned out that, compared to the College data, not even 10% of the players have accessible Combine/Pro Day data, which is why we took them out again. Otherwise too many cells would contain NA and could not be analized with all the algorithms we want to use.

The third data set we integrate, is from Pro Football reference (https://www.pro-football-reference.com/play-index/draft-finder.cgi?). It contains the information about the NFL Drafts from the last couple of years. In terms of return on investment, the most reasonable option to obtain the data, was to filter the Years 2005 to 2019, the positions QB, WR, RB and then only keeping the Rows "Year", "Rnd", "Pick", "Player", "Pos", "Tm", "College.Univ". To reproduce it, this is the way: set the named filters -> Get Results -> Share & More -> Modify & Share table -> remove all other Variables we don't need -> comma-separated (in the yellow box) -> copy paste the data into a new .txt file.

The data cleaning process

In order to clean the data fast and easy and with only a few manual steps, we build a function, that will provide us the clean data sets in the formate we want them to be. This function is called getCleanClass2 and is coded in the file 'functionGetCleanClass2.R'.

Function 'getCleanClass2'

The function getCleanClass2 needs the inputs 'draftyear' (just year number), the player-game-statistics from the two years before the draft, the player list of the two previous years (both from first source on Kaggle) and the draft dataset (from the third source). Its output is a table with the information about one single

draft year. The following explanations in plain text shall summarize the steps, if you desire more details, please see the comments in the file 'functionGetCleanClass2.R'.

getCleanClass2 will first drop all the variables, that are irrelevant for QB, RB and WR and remove observations, that don't contain any results in a game (e.g. 0's in every cell of a row). It adds a column called 'Games.Played', which allows to see, how many games were played to reach the other results, that are summed up. After that, information about the players is matched to the obtained data. Then the most important column is matched; the target value called 'Drafted', whicht is 0 if a player is not drafted and 1 if he is. Unfortunately the information about the draft is not available with the player code which is used for matching before, which means that the match has to be done by the name. This can result in some mistakes, which cannot be avoided. Then duplicates non-matchable players as well as variables that are available twice from matching are removed. After these steps, the dataframe will contain four parts in every observation:

- Col 1 - 5: Information about the Player
- Col 6: Our Y called 'Drafted'
- Col 7 - 30: The summed game statistics of the previous year
- Col 31 - 54: The summed game statistics of two years prior to the draft

The next steps separate these parts and group the two years together, in order to obtain a dataframe with 30 variables containing the game stats of both years together. This has the advantage, that Players that could only be matched to the year before the draft still can be analized.

Computing the clean data

In the script 'DataCleaning2.R' the function 'getCleanClass2' is applied to all the available years of data, to obtain a dataframe for every year. In the last part, all these dataframes are rbind-ed together and cleaned from duplicates. This overall precedure allows us to obtain all eligable players on the first hand and than only keep the latest information about players that played a senior year (those who have been drafted in their junior year will appear with their junior year).

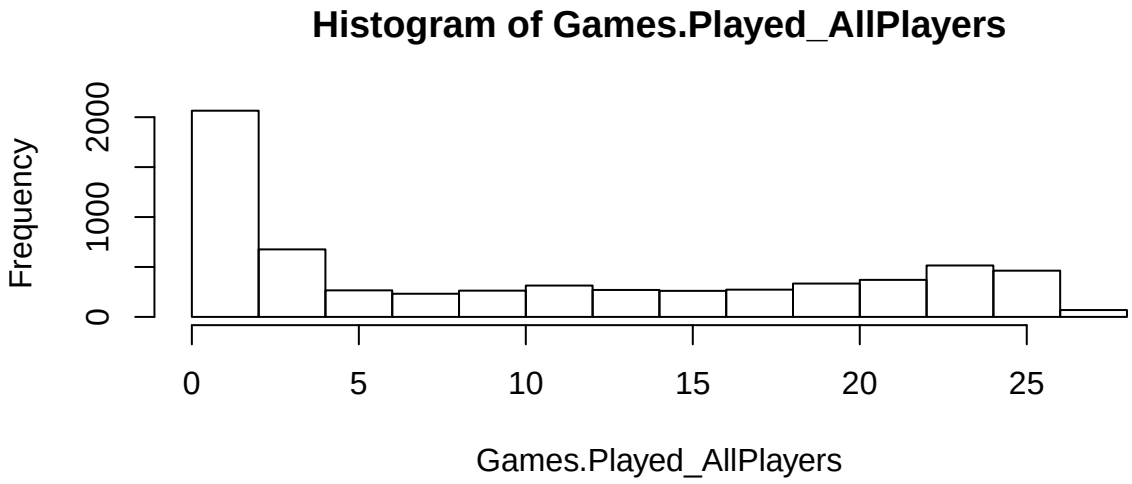
Validation of the Data and further cleaning

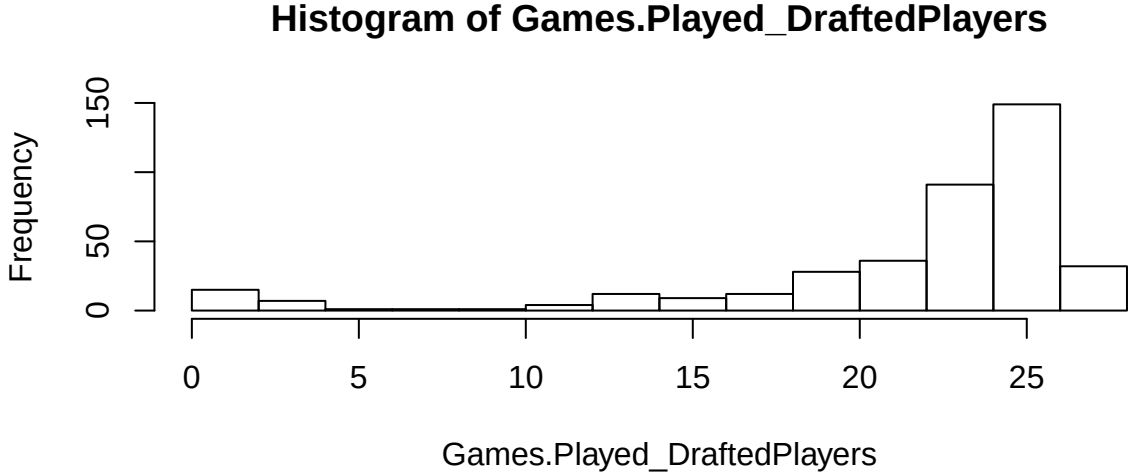
Just by having a look at the different rows (=players) can already help to see wheter the data seems to be right or not. Most of the data seems to be plausible, but for example the QB Jimmy Garoppolo has not his total performance in our data.

Table 1: Comparison of Jimmy Garoppolo's performance

	Drafted	Pass.Att	Pass.Comp	Pass.Yards	Pass.TD	Games.Played
Our Data	1	127	80	1036	10	3
Wikipedia	1	1108	706	8873	84	26

Unfortunately, the missing 23 games could not be found anywhere in the input data, which means, that there must be some errors in the data. An other way to look at this issue is by looking at the histograms of "Games.Played" of all players and compare it to the histogram of the drafted ones.





As we see in the histograms, there is a majority of players that played less than 5 games (at least according to our data) and some of them have been drafted. It seems very unlikely, that a player is drafted if he played less than 5 games in the two seasons prior to the draft. Therefore it is very likely, that all the drafted players with less than 5 played games are mistakes. As we see in the second histogram, there were no players drafted haveing played between 5 and 9 games. Looking at our business case, we decided to eliminate all players with less than 10 played games in the data. This eliminates the errors and players, that a priori don't have a chance to be drafted. The new data frame is called 'CleanClass2007to2014_3.Rdata'.

Sampling the data

Because our target value is distributed very inequally (see table below), we applied 4 different sampling methods, to approach this issue. The methods are called 'oversampling', 'undersampling', 'rose both' and 'smote'. We then cross-validate the different data in all the models, to find the best data to use for training the optimal model.

Table 2: Number of observations and Ratio of Drafted Players in the Dataframes

	Observations	Drafted_Ratio
Before filter	6372	06.24%
No Sampling	3008	12.43%
Oversampling	4022	50.00%
Undersampling	654	50.00%
Rose Both	2338	50.81%
Smote	4221	52.35%

As we see, applying the filter Games.played >= 10 already decreases the inequality of drafted vs. undrafted players by dismissing players with a chance of being drafted close to zero or have errors in the data. By sampling the data we can make the distribution close to fifty/fifty, but if this will really improve our predictions needs to be cross-validated with the models. Since the filter shall dismiss the errors, we only continue with the filtered data (one set unsampled and four sampled).

The clean data

After all the described processes the five dataframes (one unsampled and four sampled) contain following variables:

- Player.Code: A unique Number for matching the data
- Name: Name of the Player
- Class: A factor showing the college year the player was in when being in draft class with levels: JR=Junior (3.year) and SR=Senior (4.year)
- Position: A factor with Position of the Player (filtered for only QB=Quarterback, RB=Runningback, WR=Wide Receiver)
- Year: Shows the year the player was in the draft class
- Drafted: The targe which is 1 when a player was drafted and 0 when a player was not drafted
- Rush.Att: Summed rushing attempts over both seasons (mainly for RB)
- Rush.Yard: Summed rushing yards over both seasons (mainly for RB)
- Rush.TD: Summed rushing TD over both seasons (mainly for RB)
- Pass.Att: Summed passing attempts over both seasons (mainly for QB)
- Pass.Comp: Summed passing completions over both seasons (mainly for QB)
- Pass.Yard: Summed passing yards over both seasons (mainly for QB)
- Pass.TD: Summed passing TD over both seasons (mainly for QB)
- Pass.Int: Summed Inteceptions thrown over both seasons (mainly for QB)
- Pass.Conv: Summed thrown 2-pt conversion over both seasons (mainly for QB)
- Rec: Summed receptions over both seasons (mainly for WR)
- Rec.Yards: Summed reception yards over both seasons (mainly for WR)
- Rec.TD: Summed reception TD over both seasons (mainly for WR)
- Kickoff.Ret: Summed Kickoff returns over both seasons (mainly for WR/RB)
- Kickoff.Ret.Yard: Summed Kickoff return yards over both seasons (mainly for WR/RB)
- Kickoff.Ret.TD: Summed Kickoff return TD over both seasons (mainly for WR/RB)
- Punt.Ret: Summed punt returns over both seasons (mainly for WR/RB)
- Punt.Ret.Yard: Summed punt return yards over both seasons (mainly for WR/RB)
- Punt.Ret.TD: Summed punt return TD over both seasons (mainly for WR/RB)
- Off.2XP.Att: Summed 2 point conversion attempts over both seasons
- Off.2XP.Made: Summed 2 point conversions made over both seasons
- Safety: Being tackeled in the own end zone summed over both seasons(=2 pt for opponent)
- Fumble: Dropped balls summed over both seasons
- Fumble.Lost: Dropped balls recovered by opponent summed over both seasons
- Games.Played: Number of games played over both seasons (filter applied: >= 10)