FILE TRACKER

# floating files

**Folder: nba-mvp-votings-through-history**

mvp\_votings: excel spreadsheet containing NBA MVP voting [start – 2017/2018]

test\_data: excel spreadsheet containing 2018-2019 NBA MVP stats

**other files**

nba-mvp-voting: sort of cleaned version of [mvp-votings](#_floating_files)

* season and player name taken out
* contains row 639 which is only of minimums of column\_data (why is this there)

# 76\_model.mat (old)

Info: seems to be a working model or two around 76% accurate

all\_mvp\_winners: 37 x 1 vector of strings (mvp winners)

allmvpvotes: 637x28 table (columns are stats) (rows are each player who’s received a vote) (training data – up until 2017/2018 UPDATE)

all\_stats\_2019: 708x20 table (columns are stats, but contains names) (rows are player)

allstatsS1: 708x19 table (all\_stats\_2019 without column of names)

ans: 1x20 table of Alex Abrines’ date (testing only)

award\_share: 637x1 double (column of allmvpvotes for award share)

coefficients\_mvp\_votes: 20x1 double (coefficients for some model LOST)

correct: 1x1 double (# of correct predictions)

count: 1x1 double LOST

datatemp: 637x24 table (cleaned version of allmvpvotes)

doubleData: 708x19 table (double + some columns moved around version of allstatsS1)

else\_count: 1x1 double LOST

final\_1819: 708x19 table (messed up version of allstatsS1 DISREGARD)

flippedStats: 21x637 (transpose of some unknown file DISREGARD)

GPRmdl: 1x1 struct (trained ML model)

labelNames: 1x16 cell (same names as UNKNOWN)

labels: 1x20 cell (column names for tb1)

M: 1x1 double (DISREGARD)

matrixA, matrixB, matrixC: testing matrices

md1: 1x1 LinearModel (trained lin. reg model)

model1: 1x1 (another trained model)

model2: 1x1 (another trained model)

mvp\_player\_info: 637x2 table (season + player names of datatemp[MVP voting history data])

mvp\_player\_stats: 637x21 table (tb1 or FINAL TRAINING SET with award\_share)

mvp\_player\_stats\_21century: 259x21 table (spliced version of mvp\_player\_stats to only include 21st century)

mvp\_stats: 637x21 double (double version of UNKNOWN without labels)

mvp\_votes: 637x21 double (double version of mvp\_player\_stats)

names: 1x19 cell (names of statistics LOST)

nbamvpvoting1: 637x25 table (tb1 or FINAL TRAINING SET with extra columns for excess data)

newTable: 708x19 table (still of strings, but 2018-2019 data properly formatted)

playerNames1819: 708x1 table (names of players in 2018-2019)

R: 21 x 21 (correlation data, can be used to make heatmap)

tb1: 637x20 table (FINAL TRAINING SET with proper label names etc.)

# Important Functions

mvp\_percentage.m: calculates mvp\_percentage for one player using lin. reg coefficients

# Scripts

sortStats: this creates newTable, which is alphabetically sorted (708x19)

* input: allstatsS1
* input: names
* output: newTable (columns are alphabetized for consistency)

statsMVPcorr: creates 21x21 heatmap visualization of correlation data

identify\_winners: calculates the top 5 mvp\_candidates and prints them out given the output of the model

# Figures

correlation between various NBA statistics (volume + advanced)

# Attempt1

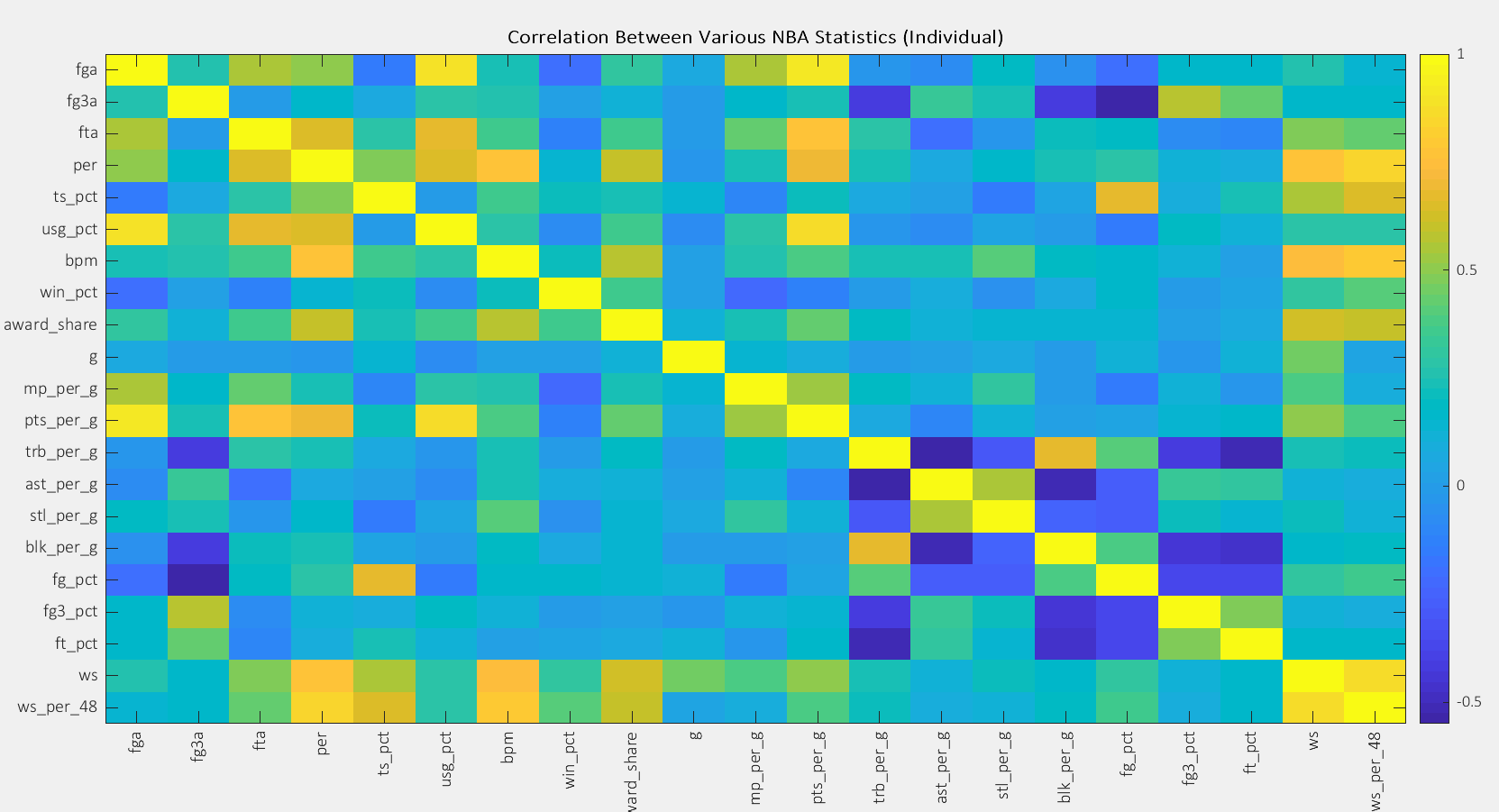
This model is for predicting the 2018-2019 NBA MVP. I started off with raw\_data from all players who’ve received MVP votes since 1980 and how many MVP votes those players received.

I formatted the data into a (#names x 21) table titled **mvp\_player\_stats**; its columns are labeled by **tdata\_labels**

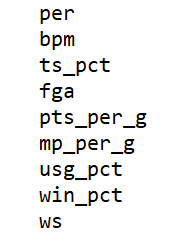


This is my training data; the input is every column except for row #9, and the response variable is the **award\_share** column

## Correlation map



Based on the map, I selected the following features.

 \*ws is actually ws\_per\_48

## Building the Model

I first used linear regression to build my model.

I created multiple models and applied those models to this year’s player data, which is a 708x19 table titled **allstats\_1819** with every column except for award\_share (which is what I’m trying to predict).

I first tested the linear regression model on the training data, to see how fitted it is. I saved this in a variable called yfit\_trainingdata\_linreg. I then attached the names and seasons to the numbers and then looked to see the predicted winner in each season. I calculated and recorded this data in (table).

I went back to the predictions and outputted it as a .csv file, which I then easily sorted to find the expected winners for the 2018-2019 NBA MVP.

I went back and created multiple other models, and repeated the above steps.