BEATING VEGAS

Predicting NBA Game Totals

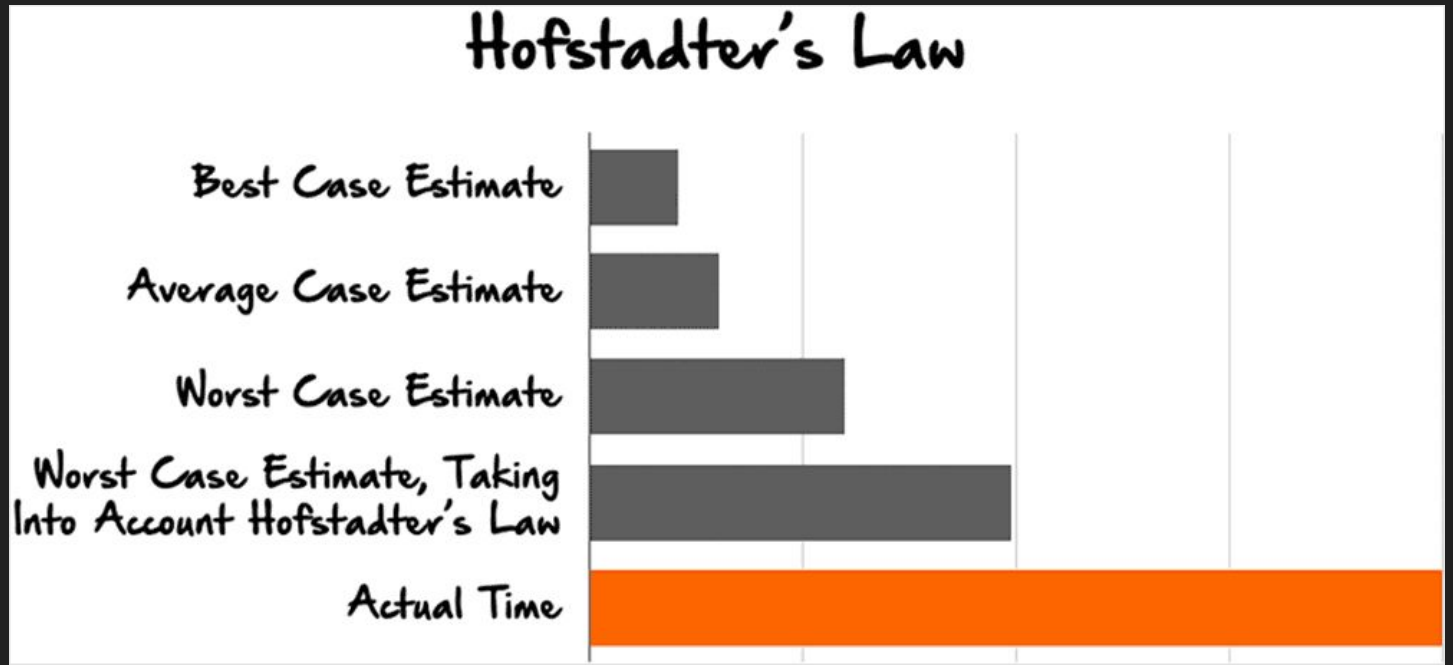
By:
Justin Wallander

Goals:

- Create a Regression Model using historical NBA data to predict NBA Game total points scored.
- Using Vegas closing line for O/U, the goal is to build a model that predicts more accurately than Vegas' lines.
- Given my intuition on what stats might impact game totals, see if there are any stats that are surprising

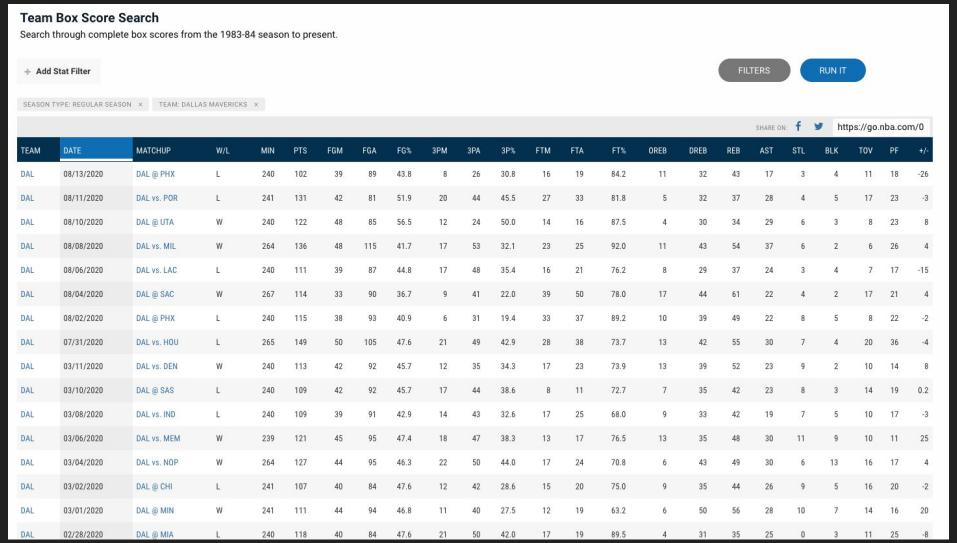
Challenges:

- Time Machine? Placing myself back in time and trying to figure out what information I will actually have available to me
- Figuring out the train-test split in order to avoid info leakage
- Hofstadter's Law:



Data:

NBA_API (from stats.nba.com)



Team Box Score Search

Search through complete box scores from the 1983-84 season to present.

+ Add Stat Filter

FILTERS RUN IT

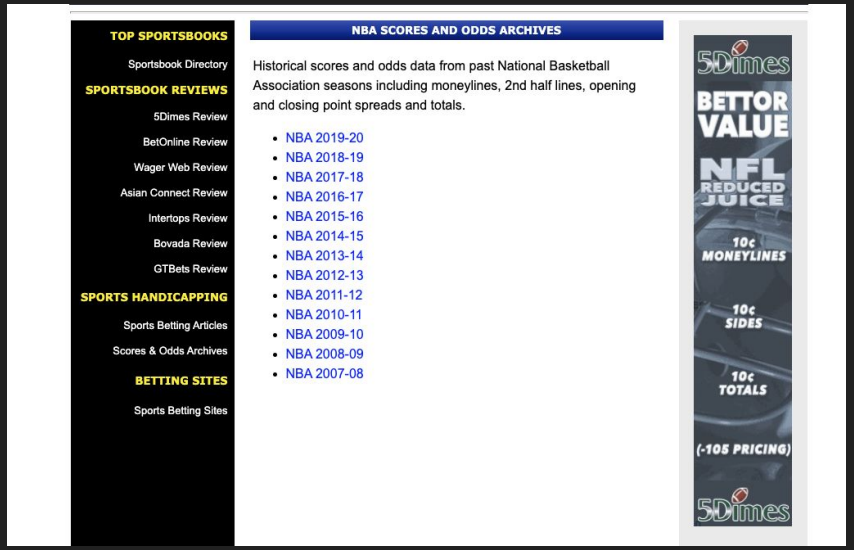
SEASON TYPE: REGULAR SEASON × TEAM: DALLAS MAVERICKS ×

SHARE ON: https://go.nba.com/0

TEAM	DATE	MATCHUP	W/L	MIN	PTS	FGM	FGA	FG%	3PM	3PA	3P%	FTM	FTA	FT%	OREB	DREB	REB	AST	STL	BLK	TOV	PF	+/-
DAL	08/13/2020	DAL @ PHX	L	240	102	39	89	43.8	8	26	30.8	16	19	84.2	11	32	43	17	3	4	11	18	-26
DAL	08/11/2020	DAL vs. POR	L	241	131	42	81	51.9	20	44	45.5	27	33	81.8	5	32	37	28	4	5	17	23	-3
DAL	08/10/2020	DAL @ UTA	W	240	122	48	85	56.5	12	24	50.0	14	16	87.5	4	30	34	29	6	3	8	23	8
DAL	08/08/2020	DAL vs. MIL	W	264	136	48	115	41.7	17	53	32.1	23	25	92.0	11	43	54	37	6	2	6	26	4
DAL	08/06/2020	DAL vs. LAC	L	240	111	39	87	44.8	17	48	35.4	16	21	76.2	8	29	37	24	3	4	7	17	-15
DAL	08/04/2020	DAL @ SAC	W	267	114	33	90	36.7	9	41	22.0	39	50	78.0	17	44	61	22	4	2	17	21	4
DAL	08/02/2020	DAL @ PHX	L	240	115	38	93	40.9	6	31	19.4	33	37	89.2	10	39	49	22	8	5	8	22	-2
DAL	07/31/2020	DAL vs. HOU	L	265	149	50	105	47.6	21	49	42.9	28	38	73.7	13	42	55	30	7	4	20	36	-4
DAL	03/11/2020	DAL vs. DEN	W	240	113	42	92	45.7	12	35	34.3	17	23	73.9	13	39	52	23	9	2	10	14	8
DAL	03/10/2020	DAL @ SAS	L	240	109	42	92	45.7	17	44	38.6	8	11	72.7	7	35	42	23	8	3	14	19	0.2
DAL	03/08/2020	DAL vs. IND	L	240	109	39	91	42.9	14	43	32.6	17	25	68.0	9	33	42	19	7	5	10	17	-3
DAL	03/06/2020	DAL vs. MEM	W	239	121	45	95	47.4	18	47	38.3	13	17	76.5	13	35	48	30	11	9	10	11	25
DAL	03/04/2020	DAL vs. NOP	W	264	127	44	95	46.3	22	50	44.0	17	24	70.8	6	43	49	30	6	13	16	17	4
DAL	03/02/2020	DAL @ CHI	L	241	107	40	84	47.6	12	42	28.6	15	20	75.0	9	35	44	26	9	5	16	20	-2
DAL	03/01/2020	DAL @ MIN	W	241	111	44	94	46.8	11	40	27.5	12	19	63.2	6	50	56	28	10	7	14	16	20
DAL	02/28/2020	DAL @ MIA	L	240	118	40	84	47.6	21	50	42.0	17	19	89.5	4	31	35	25	0	3	11	25	-8

VEGAS ODDS:

https://www.sportsbookreviewsonline.com/scoresoddsarchives/nba/nbaoddsarchives.htm



TOP SPORTSBOOKS

Sportsbook Directory

SPORTSBOOK REVIEWS

5Dimes Review

BetOnline Review

Wager Web Review

Asian Connect Review

Intertops Review

Bovada Review

GTBets Review

SPORTS HANDICAPPING

Sports Betting Articles

Scores & Odds Archives

BETTING SITES

Sports Betting Sites

NBA SCORES AND ODDS ARCHIVES

Historical scores and odds data from past National Basketball Association seasons including moneylines, 2nd half lines, opening and closing point spreads and totals.

- NBA 2019-20
- NBA 2018-19
- NBA 2017-18
- NBA 2016-17
- NBA 2015-16
- NBA 2014-15
- NBA 2013-14
- NBA 2012-13
- NBA 2011-12
- NBA 2010-11
- NBA 2009-10
- NBA 2008-09
- NBA 2007-08

Tech:


scikit learn
AWS EC2 Instance
ubuntu
XGBoost
Seaborn
pandas
NumPy
matplotlib

Data Pipeline/ Feature Engineering:

- Gather all single game data from NBA_API
- Make sure everything is uniform, clean, and sorted
- Create a running average for the season for each team:

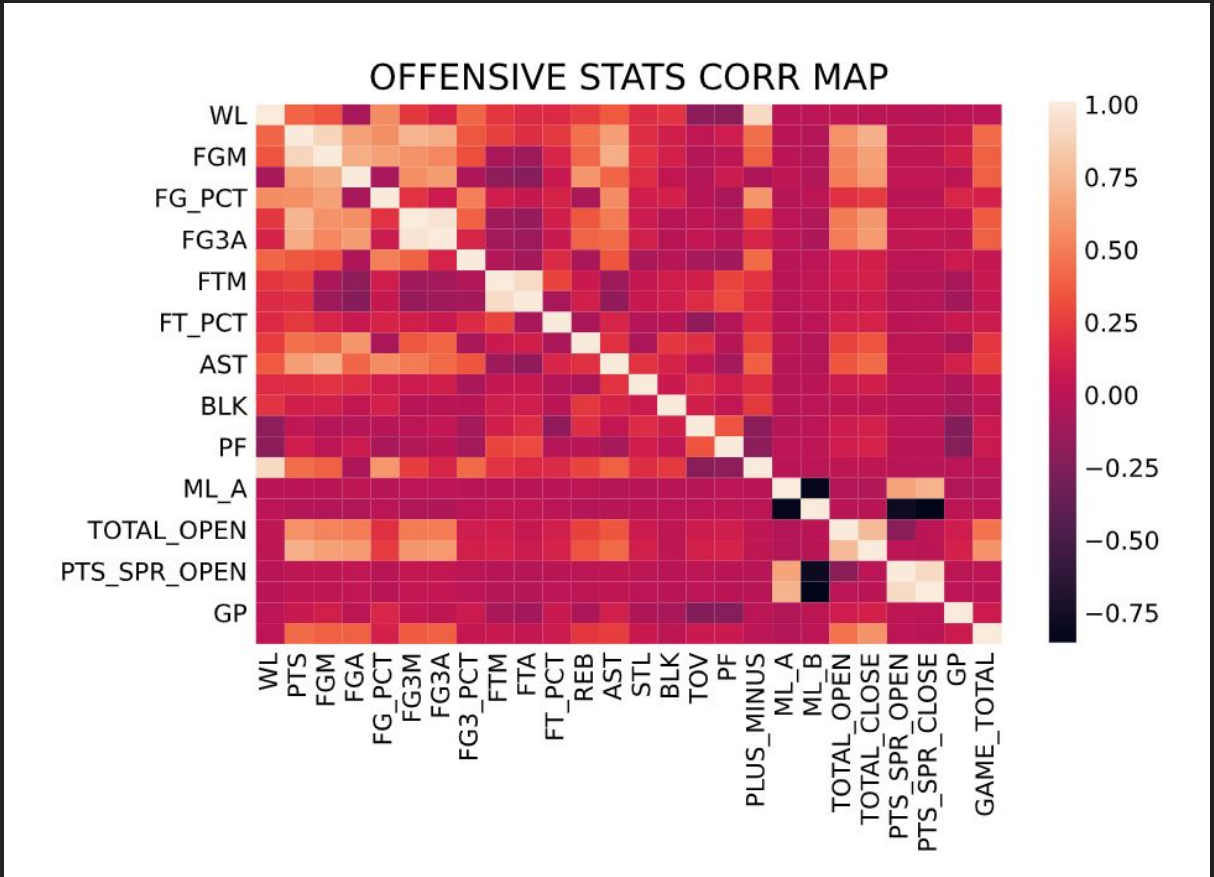
```python
#now I need to clean data types, remove columns and create the running average. I will try to do it with 1
#and then turn it into a function to loop through each team and season... function turned out ok, but
#takes for ever in the for loop
def running_col_avg(df, year, team):
    df1 = df[(df.SEASON_ID == year)&(df.TEAM_ABBREVIATION_A == team)]
    df1= df1.reset_index(drop = True)
    df1.columns = ['SEASON_ID','TEAM_ID','TEAM_ABBREVIATION','TEAM_NAME',…

    avg_cols = ['WL','PTS','FGM',…

    df1['WL']= df1['WL'].map({'W': 1, 'L' :0})
    df1['WL_OPP']=df1['WL_OPP'].map({'W': 1, 'L' :0})
    df1['GP'] = df1.index      You, 4 days ago • progress so far
    df1['GAME_TOTAL'] = df1['PTS'] + df1['PTS_OPP']

    df1[avg_cols] = df1[avg_cols].astype(float)
    counter = 1
    for idx in range(len(df1)):
        if counter != len(df1):
            df1.loc[counter, avg_cols] += df1.loc[idx, avg_cols]
            counter+=1
        df1.loc[idx, avg_cols] /= (idx +1)
    return df1
```

Data Pipeline/Feature Engineering:

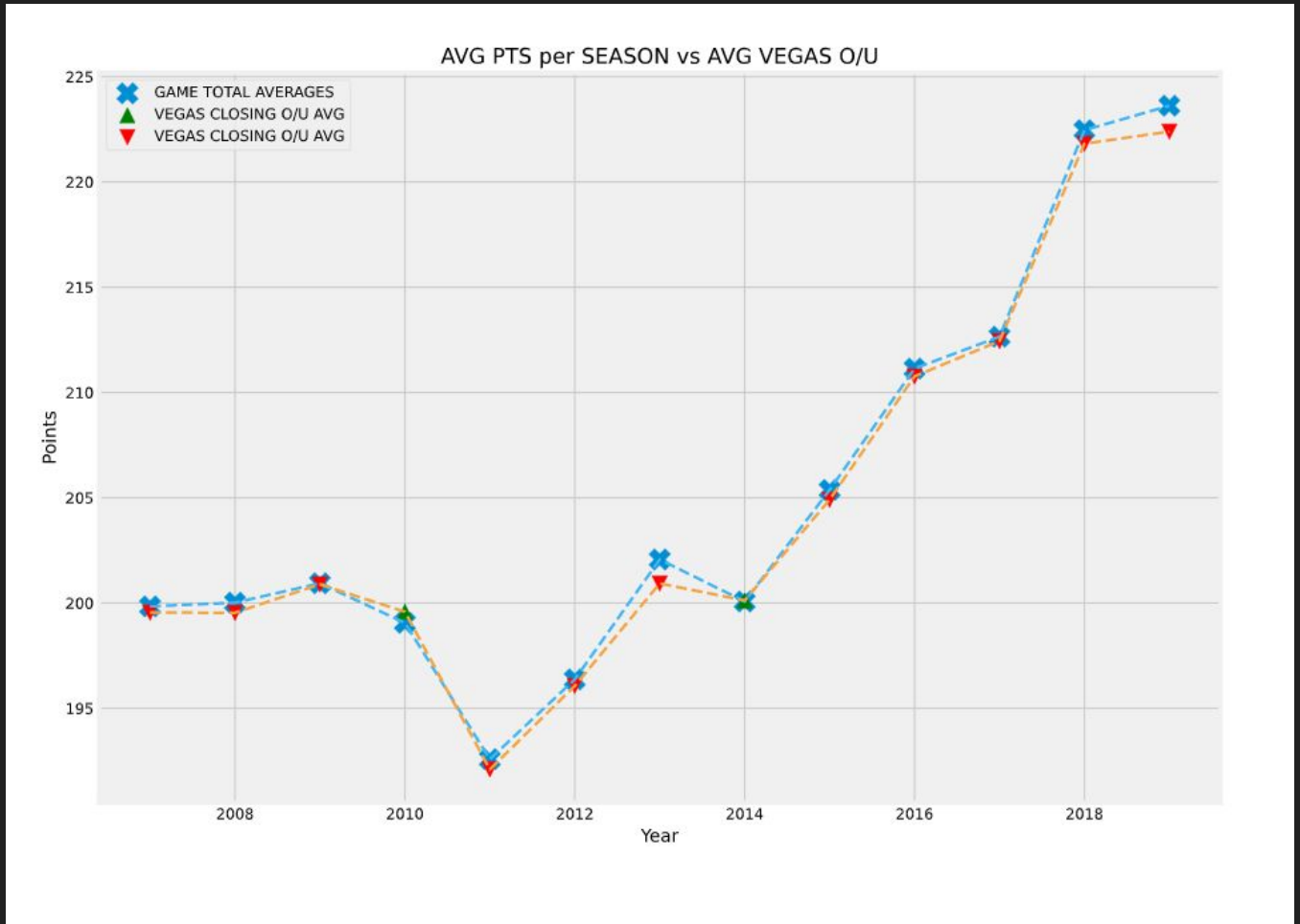
- Combine Odds data with season data
- Combine Home and away teams
- AVOID INFO LEAKAGE!!

```
#I need to shift stats columns down 1 while keeping the other columns in the same place in order
#to prevent info leakage. If I did not shift, I would be averaging in the game stats for the game
#I am trying to predict the total or      You, a few seconds ago • Uncommitted changes
shift_season_df = pd.DataFrame(columns=avg_season.columns)
for year in season_id_list:
    for team in team_abbrev_list:
        t_df = pd.DataFrame(columns=avg_season.columns)
        t_df[same_list] = avg_season[(avg_season.SEASON_ID == year) & (
                                     avg_season.TEAM_ABBREVIATION == team)][same_list]
        t_df[change_list] = avg_season[(avg_season.SEASON_ID == year) & (
                                       avg_season.TEAM_ABBREVIATION == team)][change_list].shift(periods=1)
        t_df.dropna(inplace=True)
        shift_season_df = pd.concat([shift_season_df, t_df], ignore_index=True)
```

EXPLORE:


OFFENSIVE STATS CORR MAP
WL
FGM
FG_PCT
FG3A
FTM
FT_PCT
AST
BLK
PF
ML_A
TOTAL_OPEN
PTS_SPR_OPEN
GP
WL
PTS
FGM
FGA
FG_PCT
FG3M
FG3A
FG3_PCT
FTM
FTA
FT_PCT
REB
AST
STL
BLK
TOV
PF
PLUS_MINUS
ML_A
ML_B
TOTAL_OPEN
TOTAL_CLOSE
PTS_SPR_OPEN
PTS_SPR_CLOSE
GP
GAME_TOTAL
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

EXPLORE:

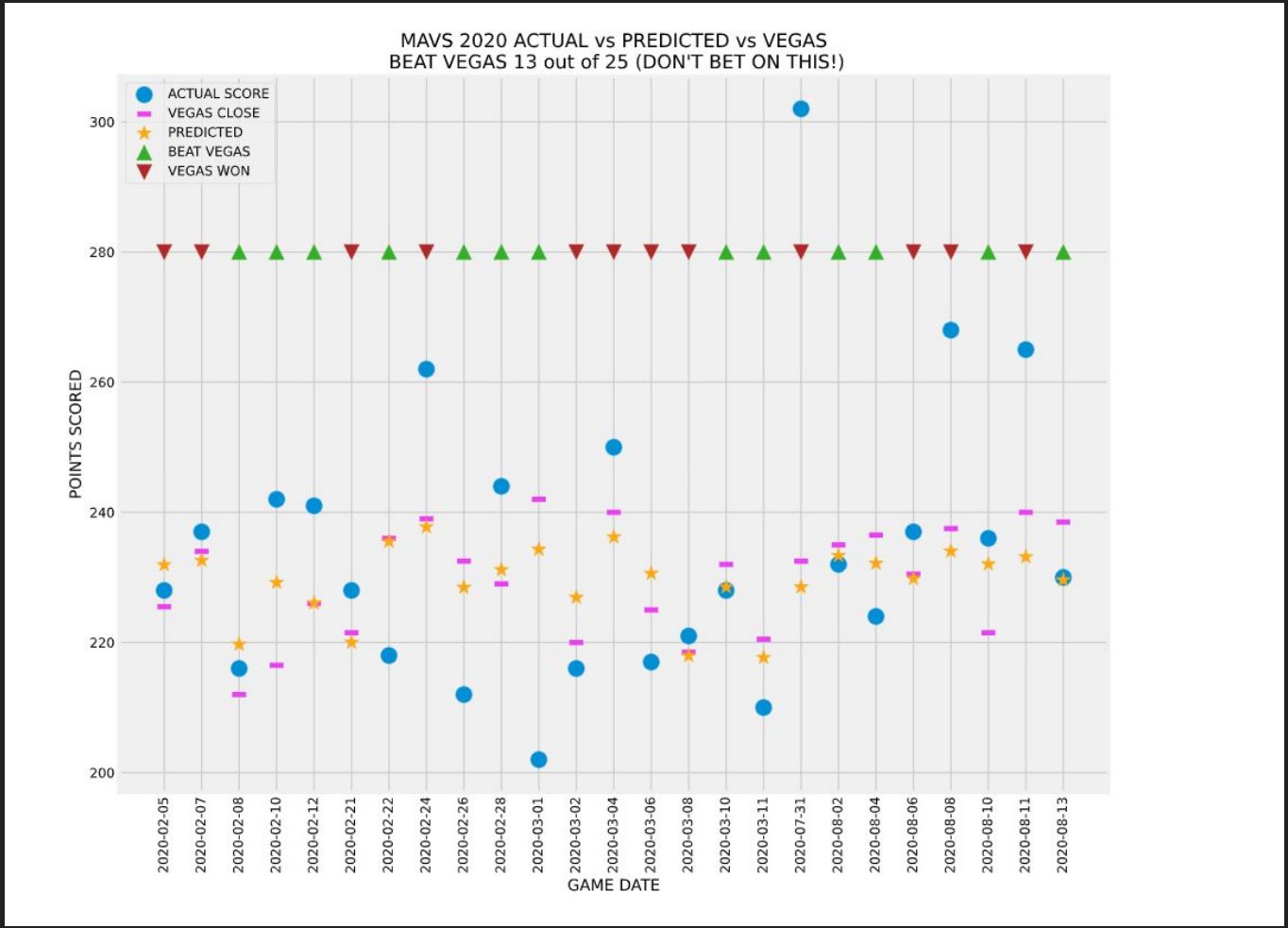

AVG PTS per SEASON vs AVG VEGAS O/U
GAME TOTAL AVERAGES
VEGAS CLOSING O/U AVG
VEGAS CLOSING O/U AVG
Points
225
220
215
210
205
200
195
2008
2010
2012
2014
2016
2018
Year

THE MODEL: XGBoost for the win?

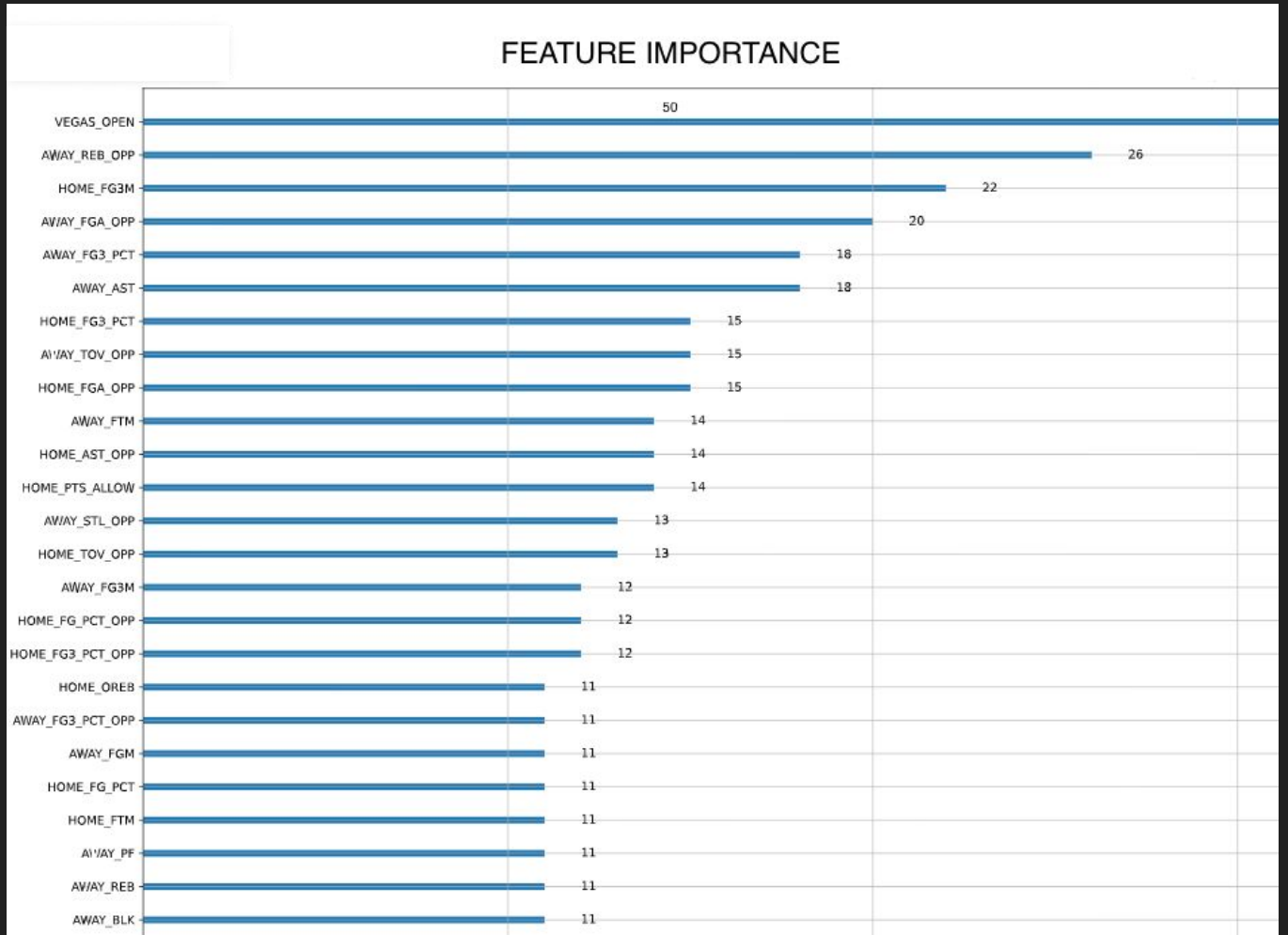
- Baseline Dummy Regressor score = 21.1072 (not too shabby)
- The first couple of runs of my XGBoost model did not beat the Dummy (disconcerting to say the least)
- After quite a bit of tinkering with my splits, feature selection, and SKLearn's RandomizedCV and GridsearchCV I was able to accomplish some decent results:
    - Features were whittled from 144 to 46
    - Vegas' RMSE on my test set = 18.56
    - My Model's RMSE on my test set= 18.75
    - HOWEVER.... When running the model specifically on 1 team (My Dallas Mavericks) I was actually able to slightly beat Vegas:
        - Vegas RMSE when predicting DAL = 21.199
        - Model RMSE when predicting DAL= 21.108

```
bst1 = xgb.XGBRegressor(
                        objective= 'reg:squarederror',
                        booster='gbtree',
                        colsample_bytree=.87,
                        learning_rate=.056,
                        max_depth=2,
                        n_estimators=199,
                        n_jobs=-1,
                        random_state=0,
                        reg_lambda=6,
                        subsample=0.61,
                        )
```

WE DID IT (sort of):



Feature Importance:



Next Steps:

- Connect to a live odds api to get current lines
- Get more granular data- advanced stats, player stats, etc
- Build a simulator to see how certain gambling strategies could play out
- ULTIMATE STEP: Add other leagues/sports into the mix

Thanks!

Email : justin.wallander@gmail.com

Github: https://github.com/justin-wallander

LinkedIn: https://www.linkedin.com/in/justin-wallander/