CAN WE BEAT THE LINE?

Group 5

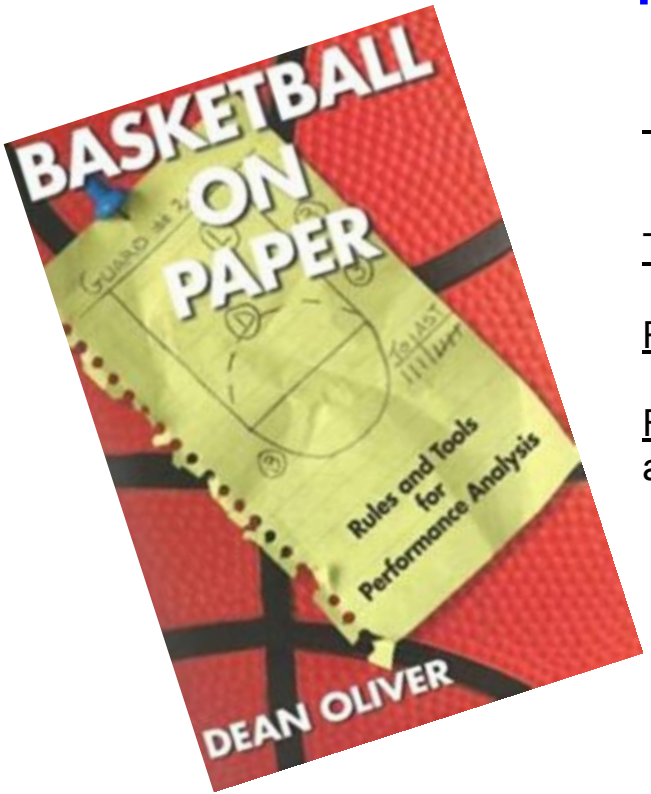
Zeshaun Subhani
Michael Bell
Christian Coleman
Sal Marino
Austin Coleman

NBA Win Predictor Model

Data Search

- Looked at many different data sources
- Found a great dataset
- There were a lot of stats to consider

The Four Factors[1]



Shooting: Effective Field Goal Percentage
2 point shots made + (3 point shots made * 0.5)/Total shots attempted

Turnovers: Turnover Percentage

Rebounding Percentage

Free Throws: measures how often a team gets to the line and the accuracy in which they make free throw attempts

1. https://www.basketball-reference.com/about/factors.html

Machine Learning

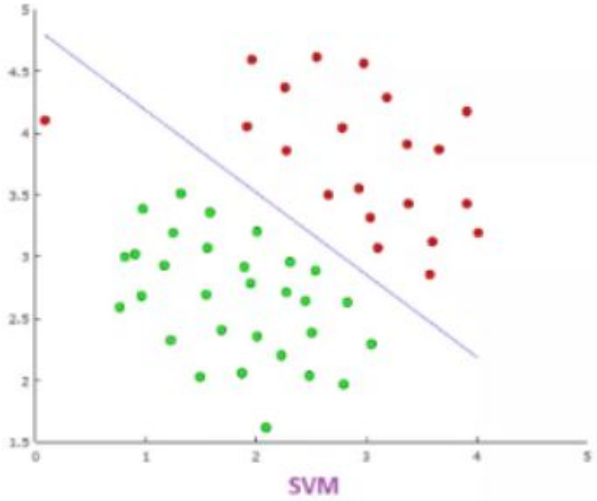
- Needed to train our model on data 2012-2018
- Models: KNN, Linear Regression, Random forest, Logistic Regression, SVM
- RESULT: Python function based on SciKitLearn use on a daily basis to update

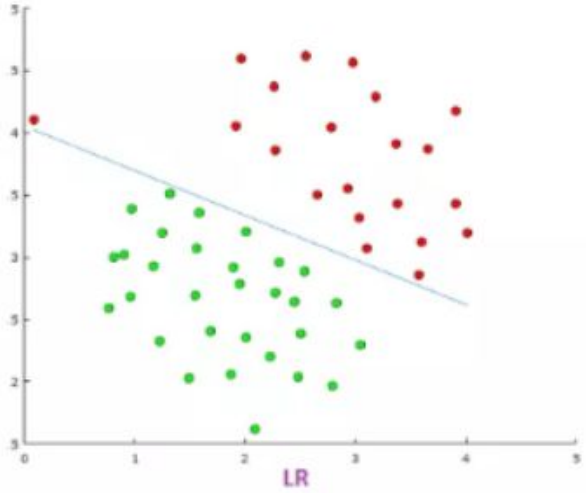
$$h_\theta = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Selection of SVM

SVM hypothesis is a discriminator function producing 1 or 0

- Logistic Regression produces probabilistic values while SVM produces 1 or 0
- LR is more sensitive to outliers than SVM





Model Result

- Determine win/loss for today
- Determine win/loss for rest of season
- Model accuracy

```
[30] # Game 1 WAS at DET
     was_stats = [.525, .541, 12.2, 13.7, 19.9, 72.0, .214, .221]
     det_stats = [.495, .514, 12.2, 12.6, 25.4, 79.3, .202, .235]
     predict_outcome('WAS', was_stats, det_stats)
```

```
[0.]
Prediction is a Loss for WAS
```

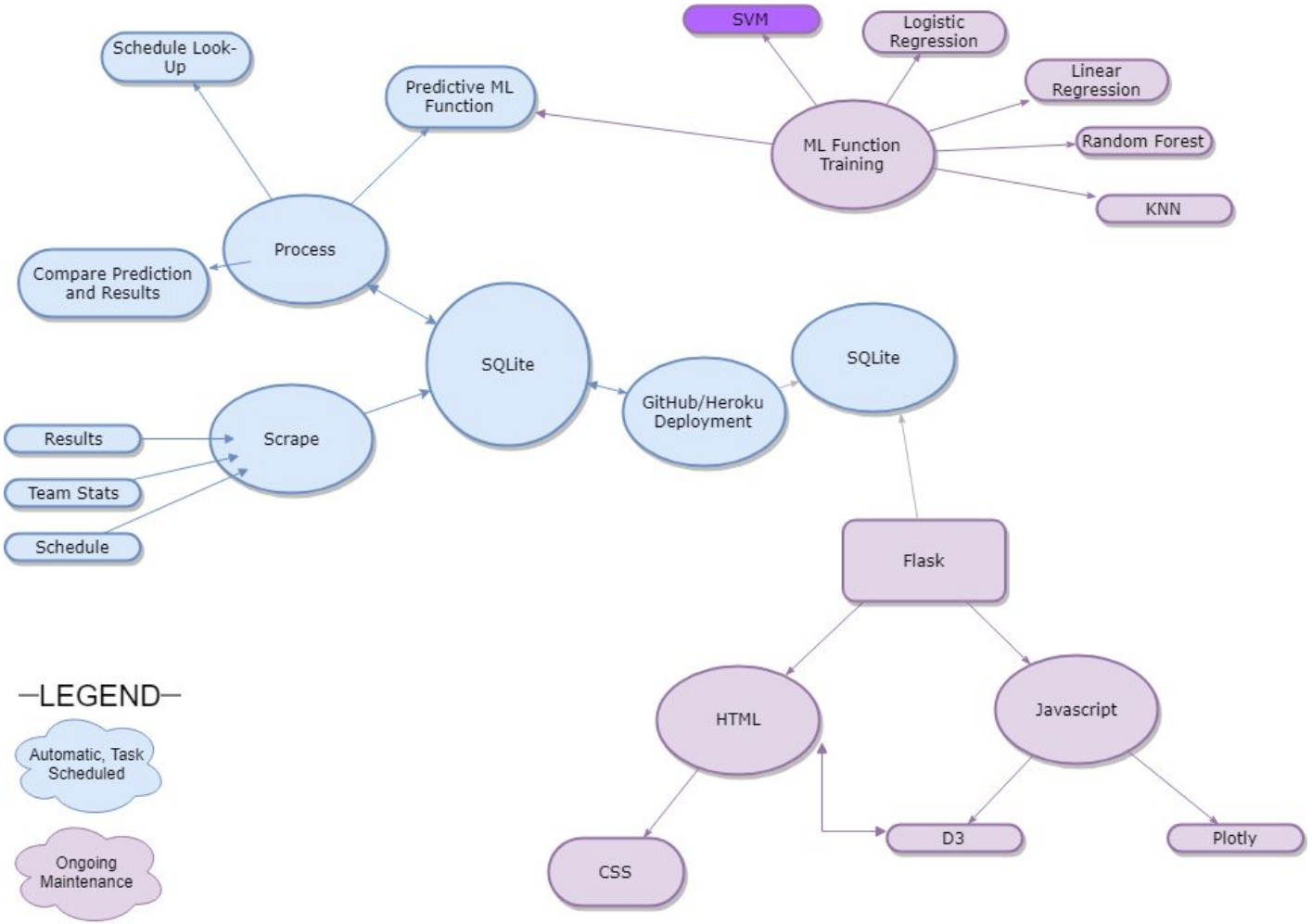
```
[31] # Game 2 Phoenix at ORL
     pho_stats = [.512, .535, 14.1, 13.2, 20.2, 74.7, .176, .227]
     orl_stats = [.506, .526, 12.1, 12.5, 20.0, 77.7, .159, .198]
     predict_outcome('PHO', pho_stats, orl_stats)
```

```
[0.]
Prediction is a Loss for PHO
```

```
[32] # Game 3 IND at ATL
     ind_stats = [.527, .501, 13.2, 14.0, 22.1, 78.1, .175, .182]
     atl_stats = [.509, .542, 15.3, 13.8, 22.9, 76.9, .201, .234]
     predict_outcome('IND', ind_stats, atl_stats)
```

```
[1.]
Prediction is a Win for IND
```

Workflow Diagram



Scraping and Scheduling

- Java intense site
- Using Xpath
- 4 factors for each team on schedule
- Update SQLite DB
- Scraped basketball-reference.com

```
46              oTOV = browser.find_by_xpath(oTOV_X).value
47              stat_list.append(oTOV)
48              dTOV = browser.find_by_xpath(dTOV_X).value
49              stat_list.append(dTOV)
50              oORB = browser.find_by_xpath(oORB_X).value
51              stat_list.append(oORB)
52              dDRB = browser.find_by_xpath(dDRB_X).value
53              stat_list.append(dDRB)
54              oFtByFga = browser.find_by_xpath(oFtByFga_X).value
55              stat_list.append(oFtByFga)
56              dFtByFga = browser.find_by_xpath(dFtByFga_X).value
57              stat_list.append(dFtByFga)
58
59              team_list.append(stat_list)
60              print(team_list)
61          x = 1
```

Team and Opponent Stats

Year/year calculations use per game statistics

	G	MP	FG	FGA	FG%	3P	3PA	3P%	2P	2PA	2P%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
Team	40	9600	1691	3717	.455	386	1194	.323	1305	2523	.517	700	993	.705	504	1435	1939	890	411	220	574	904	4468
Team/G		240.0	42.3	92.9	.455	9.7	29.9	.323	32.6	63.1	.517	17.5	24.8	.705	12.6	35.9	48.5	22.3	10.3	5.5	14.4	22.6	111.7
Lg Rank		24	9	4	17	25	22	30	6	6	15	16	9	28	1	9	3	24	1	9	19	23	16
Year/Year		-0.8%	6.0%	5.5%	+.002	-10.2%	-1.7%	-.030	11.9%	9.3%	+.012	1.0%	2.6%	-.011	0.9%	10.1%	7.6%	4.3%	13.4%	9.5%	2.6%	12.1%	3.6%
Opponent	40	9600	1569	3506	.448	405	1215	.333	1164	2291	.508	701	933	.751	409	1387	1796	958	337	197	686	919	4244
Opponent/G		240.0	39.2	87.7	.448	10.1	30.4	.333	29.1	57.3	.508	17.5	23.3	.751	10.2	34.7	44.9	24.0	8.4	4.9	17.2	23.0	106.1
Lg Rank		24	8	9	6	5	9	3	7	12	12	13	15	7	14	12	16	12	24	13	1	25	8
Year/Year		-0.8%	1.9%	4.4%	-.011	-9.8%	-0.6%	-.034	6.7%	7.3%	-.003	7.9%	10.4%	-.017	7.4%	6.3%	6.5%	2.8%	9.7%	7.1%	6.9%	7.7%	1.6%

Team Misc

	Advanced	Offense Four Factors	Defense Four Factors

NBA Win Predictor Model

- Daily update of games scheduled
- Update the predictor to account for wins and losses from schedule

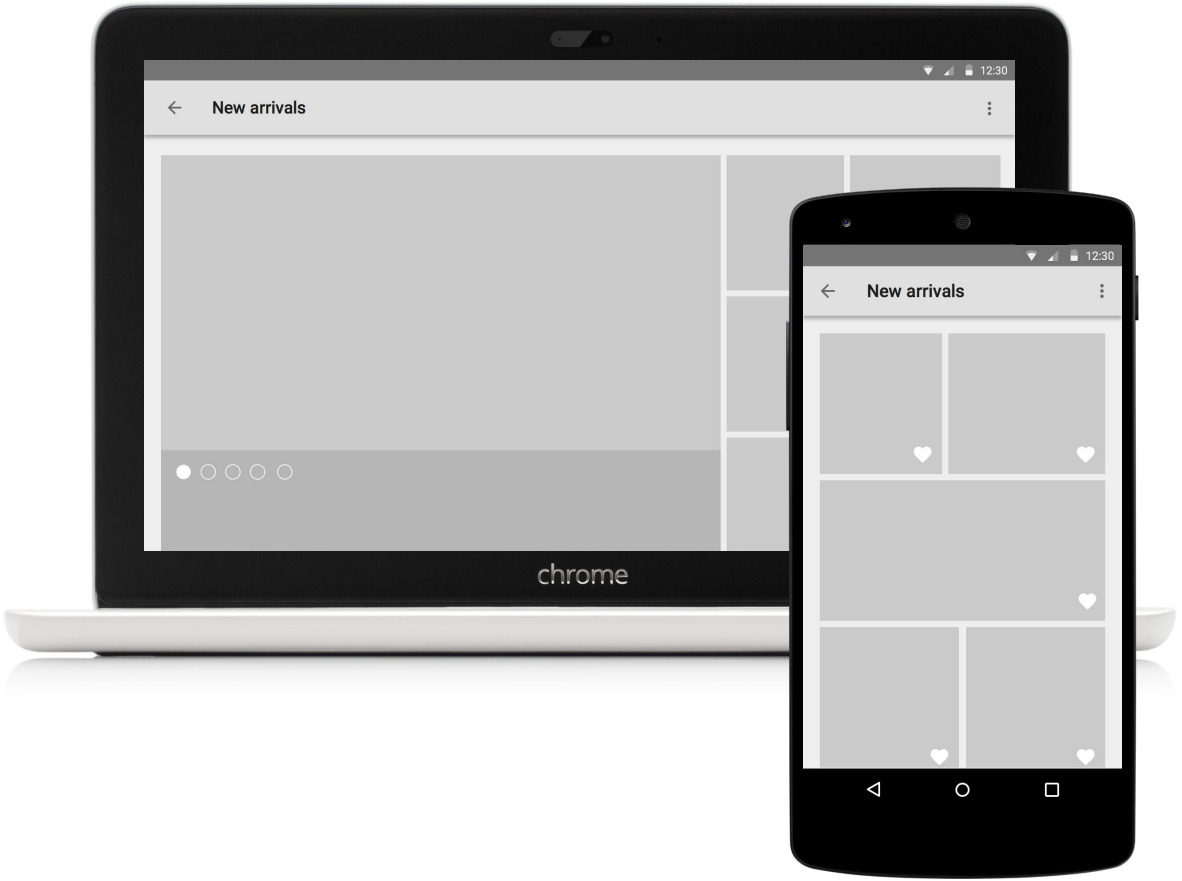
Flask / HTML

- Six routes (3 to Jsonify the SQL in Database and 3 for rendering HTML)
- Bootstrap, minimal CSS

- Java
  - D3 to loop through jsonify games to populate HTML
  - Plotly for model accuracy

GitHub / Heroku

- Need for updating daily
- Repo to view
- Allow us to collaborate



APPLICATIONS

HTML/CSS

Flask

Plotly

Pandas

Python

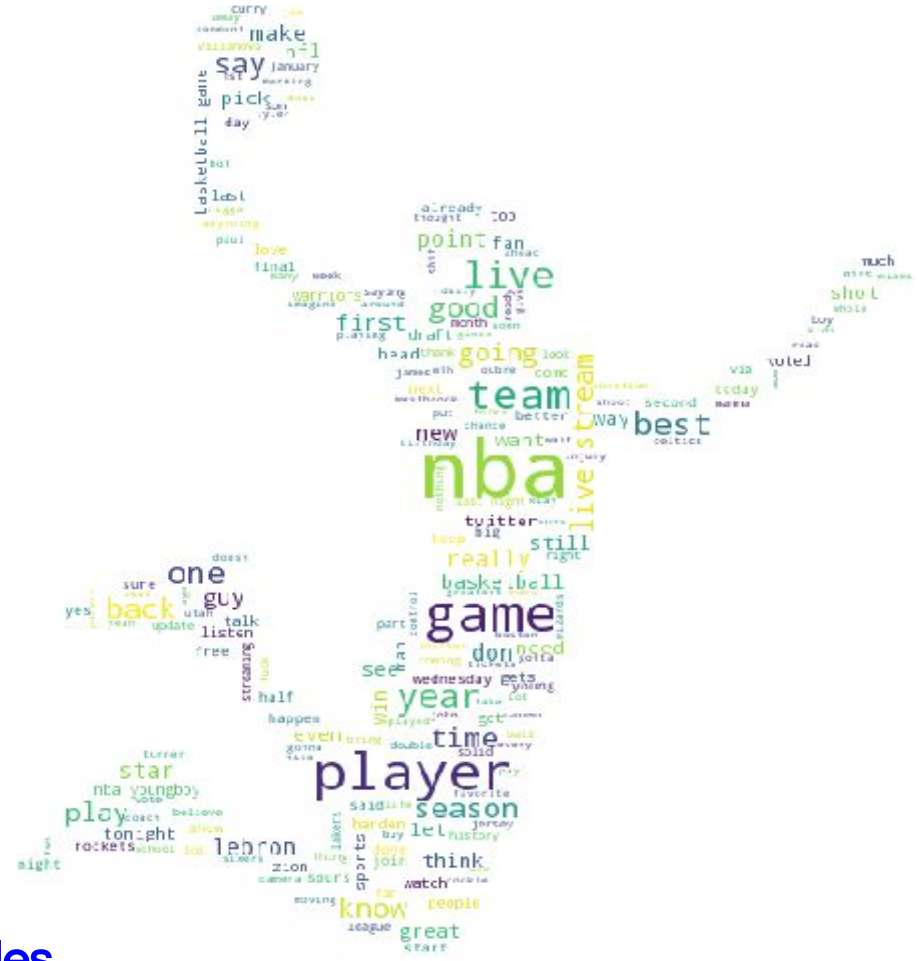
Javascript

SQLite

Tweepy

Heroku

Google Slides



DEMO

https://nba-predictions-2019.herokuapp.com/

Improvements

- Short observation time, <100 games
- Consider other factors
  - Number of rest days between games
- Incorporate other sources: Vegas lines, injury reports
  - Twitter functionality, more background stats on teams