NBA Classification

Spencer Newcomb and Jeremy Meyer

Brigham Young University

Table of contents

1. **Introduction**
   - Background Goals
   - EDA
   - Data Cleaning
2. **The Algorithms**
   - Neural Network
   - Support Vector Machine
3. **Prediction Performance**
4. **Results**
5. **Conclusions**

INTRODUCTION

Figure: Damian Lillard defending James Harden

- Principal goal of basketball is to win

1 Background

Figure: Damian Lillard defending James Harden

- Principal goal of basketball is to win
- Players score differently

1 Background

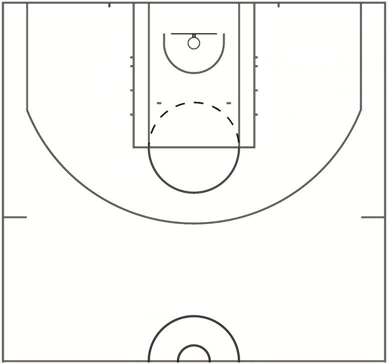
Figure: Damian Lillard defending James Harden

- Principal goal of basketball is to win
- Players score differently
- Coaches want to make the most out of each situation

1 Background

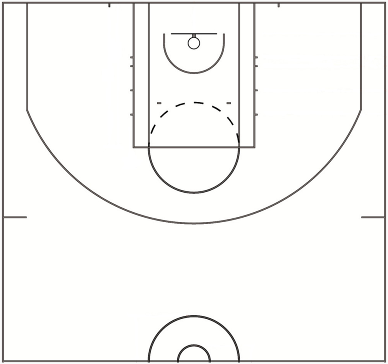
Figure: Damian Lillard defending James Harden

- Principal goal of basketball is to win
- Players score differently
- Coaches want to make the most out of each situation
- Shot data from 2014-15 NBA season to train predictive shot model



GENERAL GOAL: Predict the probability of a player making a shot in a given situation.

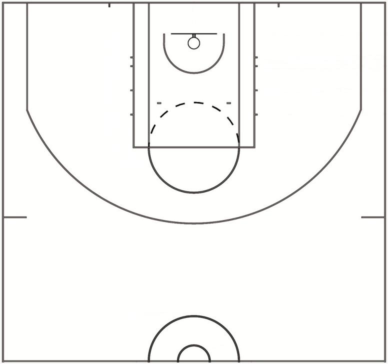
Primarily interested in the following situations:



GENERAL GOAL: Predict the probability of a player making a shot in a given situation.

Primarily interested in the following situations:

- Game-winners from varying distances and dribbling patterns

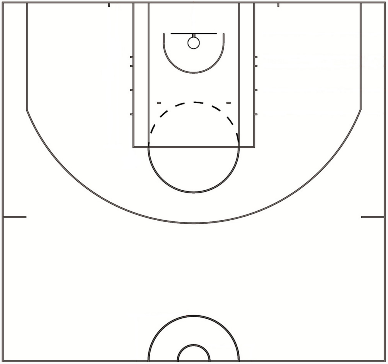


GENERAL GOAL: Predict the probability of a player making a shot in a given situation.

Primarily interested in the following situations:

- Game-winners from varying distances and dribbling patterns
- Which players are better drivers or spot shooters

1 Goals



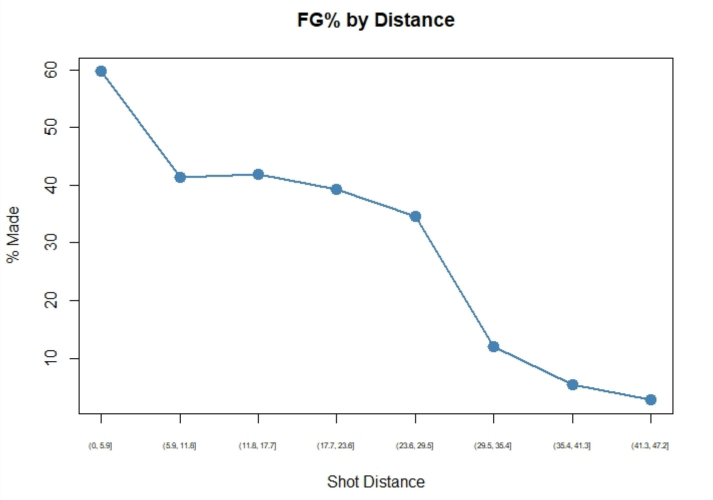
GENERAL GOAL: Predict the probability of a player making a shot in a given situation.

Primarily interested in the following situations:

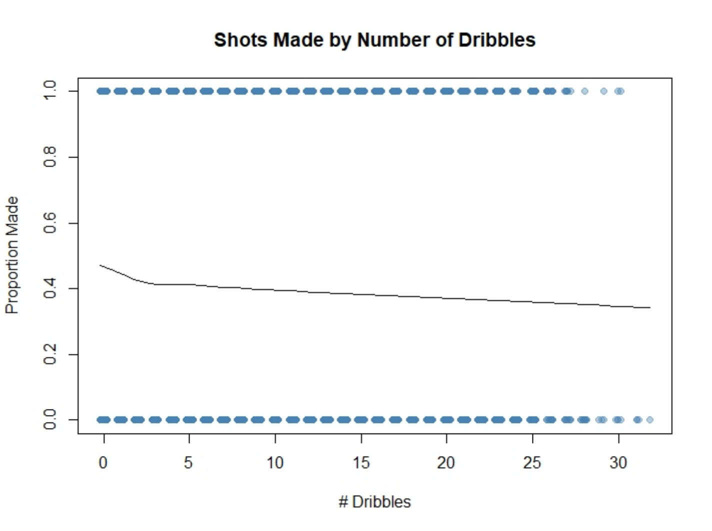
- Game-winners from varying distances and dribbling patterns
- Which players are better drivers or spot shooters
- Assess the "clutch" factor of different players

1 EDA: Distance Effect

1 EDA: Distance Effect



1 EDA: Dribbling Effect



1 EDA: Clutch Situations

	Made	Missed	avgDist	nShots
Quarters 1-3	0.457	0.543	13.429	96122
4th Quarter	0.440	0.560	14.005	28947
End Shot Clock	0.363	0.637	15.845	17414
End Game	0.313	0.687	18.976	339

1 Data Cleaning

Variables in original dataset:

- GAME_ID
- MATCHUP
- W
- LOCATION
- PERIOD
- GAME_CLOCK
- SHOT_CLOCK
- DRIBBLES
- SHOT_DIST
- SHOT_RESULT
- CLOSE_DEF_DIST
- player_name
- player_id

1 Data Cleaning

Variables in original dataset:

- GAME_ID
- MATCHUP
- W
- LOCATION
- PERIOD
- GAME_CLOCK
- SHOT_CLOCK
- DRIBBLES
- SHOT_DIST
- SHOT_RESULT
- CLOSE_DEF_DIST
- player_name
- player_id

1 Data Cleaning

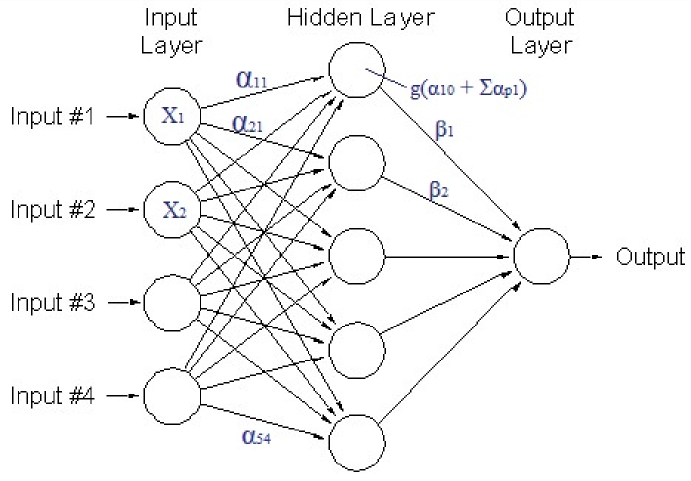
Group all data by player, and keep the following:

- LOCATION
- SEC_INTO_GAME
- SECONDS_LEFT (In Period)
- SHOT_CLOCK
- DRIBBLES
- SHOT_DIST
- SHOT_RESULT
- CLOSE_DEF_DIST

THE ALGORITHMS

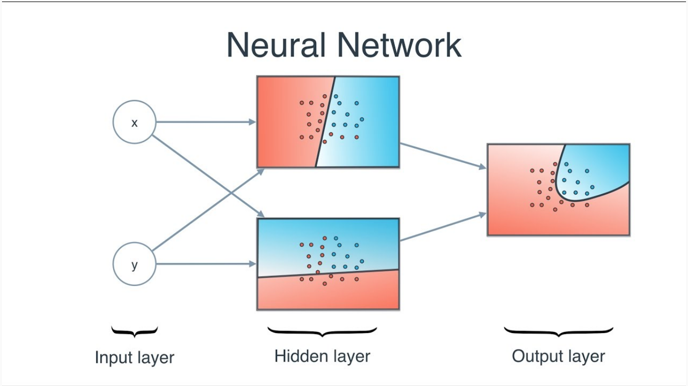
2 What is a Neural Network?

- Works with numeric inputs and outputs. Can be used for classification (make or miss).
- Idea: Transform predictors (Xs) into Zs that linearly relate to Y (response).



2 Terminology

- Node - Each "circle" located in the hidden layer. Computes weighted sum, then applies activation function (logistic).
- P denotes the number of explanatory variables
- M denotes the number of nodes in the hidden layer
- Each node and output node has a bias.
- α_{mp}: weight associated with the pth predictor onto the mth node.
  - Let **A** indicate a Mx(P+1) matrix of all α weights
- β_m: weight associated with the mth mode onto the output layer.
  - Let **B** indicate a (M+1)x1 vector of all β weights



- Allows us to capture non-linear relationships and interactions very well

2 How does it learn?

For classification, Neural Nets seek to choose weights $(\mathbf{A}, \beta)$ that minimize:

$$L(\mathbf{A}, \beta) + \lambda \mathrm{P}(\mathbf{A}, \beta) \tag{1}$$

With the cross-entropy loss function:

$$-\sum_{i=1}^{n} \sum_{k=0}^{1} I(y_i = k) \log(\pi_{ik}) \tag{2}$$

where π_{ik} is the proportion of ks in the dataset.
And weight penalty (penalty parameter λ):

$$\mathrm{P}(\mathbf{A}, \beta) = \sum_{m=0}^{M} \sum_{p=0}^{P} \alpha_{mp}^2 + \sum_{m=0}^{M} \beta_m^2 \tag{3}$$

A gradient is estimated by back propagation $\rightarrow$ Local minimum.

2 How does it learn?

2 Why Neural Networks?

Pros:

- Flexible system can capture complex non-linear patterns in NBA data
- Predictive power in large datasets
- Result is predicted probabilities $\rightarrow$ Answer research questions.

2 Why Neural Networks?

Pros:

- Flexible system can capture complex non-linear patterns in NBA data
- Predictive power in large datasets
- Result is predicted probabilities $\rightarrow$ Answer research questions.

Cons:

- Prone to overfitting
- Tons of parameters
- Very computational. We need to fit one for each player.

2 What is a Support Vector Machine (SVM)?

Big Idea:

Separate classes of response variable based on values of the different predictor variables using *hyperplanes*

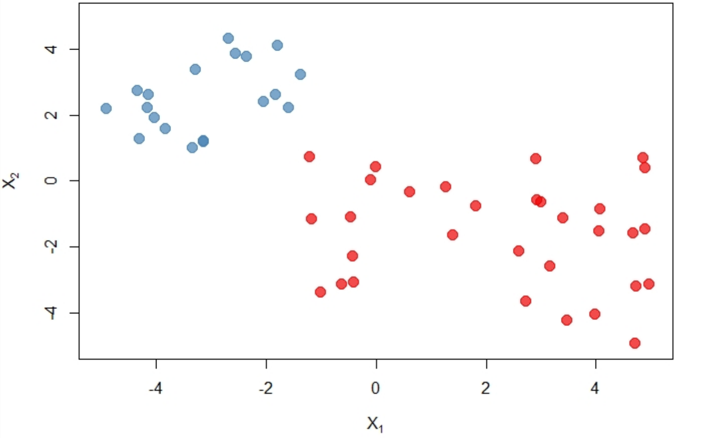
2 What is a Support Vector Machine (SVM)?

Big Idea:

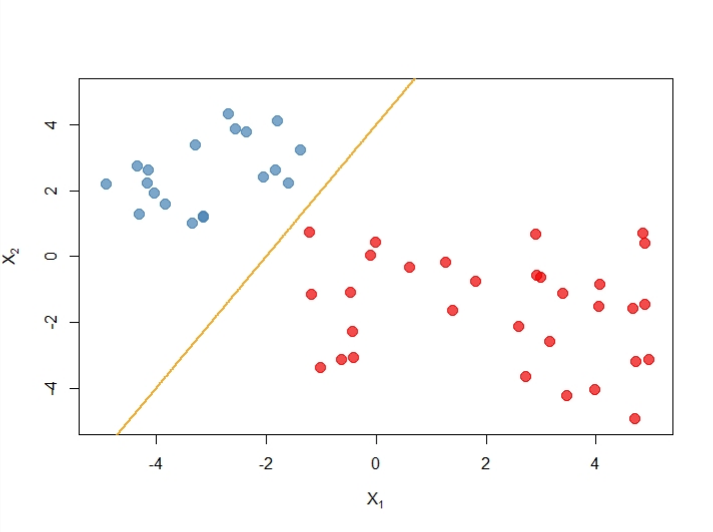
Separate classes of response variable based on values of the different predictor variables using *hyperplanes*

How do we do this in an optimal way?

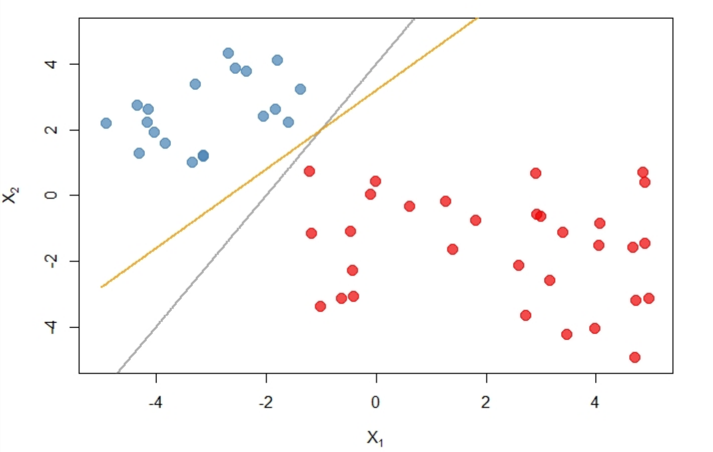
2 What is a Support Vector Machine (SVM)?



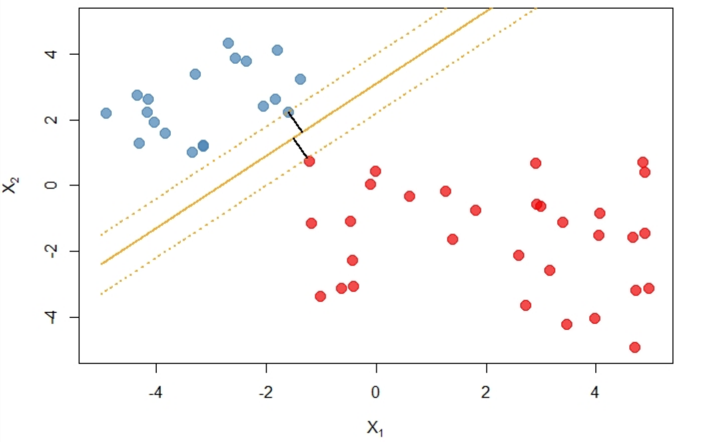
2 What is a Support Vector Machine (SVM)?



2 What is a Support Vector Machine (SVM)?



2 Finding the Best Separating Hyperplane



Let the $\boldsymbol{y}$ = set of observed responses.

Let $\mathcal{H} = \{\boldsymbol{x} : \beta_0 + \boldsymbol{x}'\boldsymbol{\beta} = 0\}$ denote a hyperplane.

This means that $f(\boldsymbol{x_0}) = \beta_0 + \boldsymbol{x_0}'\boldsymbol{\beta} > 0$ denotes an observation $\boldsymbol{x_0}$ that is above $\mathcal{H}$.

1. Set $y_i \in \{-1, 1\}$
2. Find hyperplane such that $y_i(f(\boldsymbol{x_i}) > 0 \, \forall \, i$
3. Maximize the orthogonal distance between the plane and the closest points to the plane (i.e. the margin).

Recall:

$$\text{Orthogonal Distance from } \boldsymbol{x} \text{ to } f(\boldsymbol{x}) = \frac{f(\boldsymbol{x})}{||\beta||}$$

Recall:

$$\text{Orthogonal Distance from } \boldsymbol{x} \text{ to } f(\boldsymbol{x}) = \frac{f(\boldsymbol{x})}{||\beta||}$$

Therefore, the margin, M, is defined by those points that are closest to $\mathcal{H}$:

$$M = \min_i \left[\frac{f(\boldsymbol{x}_i)}{||\beta||} \right]$$

This is what we aim to maximize. This multi-faceted optimization problem is complex, involving Lagrange dualities.

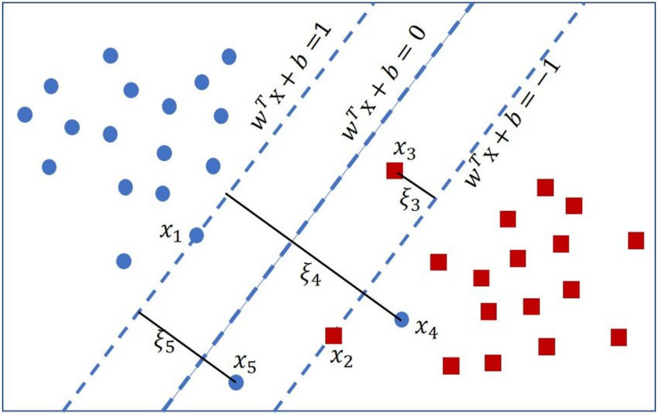
It can be shown that the resulting "maximum margin" hyperplane is then defined by

$$\hat{\beta} = \sum_{i\,:\,y_i(\beta_0 + \boldsymbol{x_i}\boldsymbol{\beta})=1} \hat{\lambda}_i y_i \boldsymbol{x_i}$$

and so

$$\hat{f}(\boldsymbol{x_0}) = \hat{\beta}_0 + \sum_{i\,:\,y_i(\beta_0 + \boldsymbol{x_i}\boldsymbol{\beta})=1} \hat{\lambda}_i y_i \boldsymbol{x_0}' \boldsymbol{x_i}$$

2 Realistic SVM



New constraint:

$$y_i(\beta_0 + \boldsymbol{x}_i' \boldsymbol{\beta}) \geq M(1 - \xi_i)$$

$$\xi_i \geq 0, \sum_{i=1}^{n} \xi_i \leq C$$

C is now a tuning parameter, determining how lenient SVM is toward observations in margin, or on wrong side of $\mathcal{H}$.

2 Realistic SVM

It can be shown that the resulting "maximum margin" hyperplane is now defined by

$$\hat{\beta} = \sum_{i\,:\,y_i(\beta_0 + \boldsymbol{x_i}\boldsymbol{\beta}) > 1 - \xi_i} \hat{\lambda}_i y_i \boldsymbol{x_i}$$

and so

$$\hat{f}(\boldsymbol{x_0}) = \hat{\beta}_0 + \sum_{i\,:\,y_i(\beta_0 + \boldsymbol{x_i}\boldsymbol{\beta}) > 1 - \xi_i} \hat{\lambda}_i y_i \boldsymbol{x_0}' \boldsymbol{x_i}$$

2 Improving the SVM

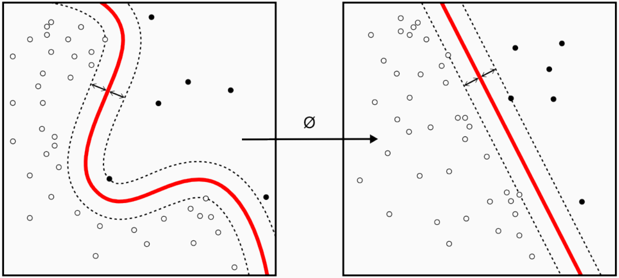
Expand to higher dimensions and project onto original space to better separate the classes.

Kernel Trick: Computationally-feasible basis function expansion

Common SVM kernels:

- Linear
- Polynomial
- Radial
- Hyperbolic Tangent

2 Kernel Basis Function Expansion



2 Why SVM?

Pros:

- Flexible
- Can retrieve (pseudo) predictive probabilities
- Suited for binary classification

2 Why SVM?

Pros:

- Flexible
- Can retrieve (pseudo) predictive probabilities
- Suited for binary classification

Cons:

- Sensitive to small changes in tuning parameters
- Prone to overfitting

Prediction Performance

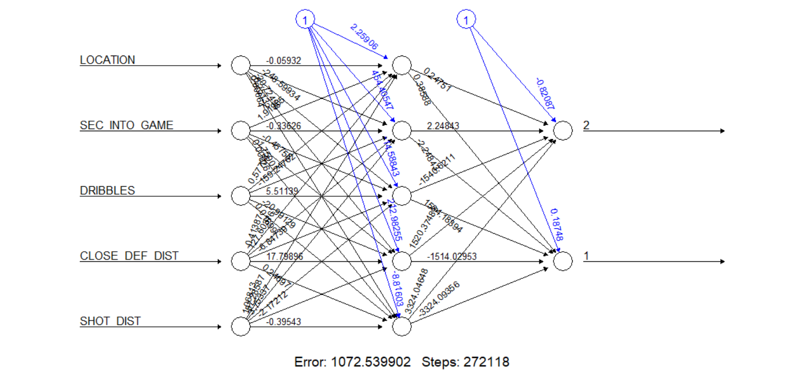
3 NN Tuning for NBA Data

Classification testing set error

# Nodes	3	5	7	10
Damian Lillard	0.412	**0.375**	0.464	0.423
James Harden	**0.423**	0.495	0.495	0.485
LaMarcus Aldridge	**0.423**	0.462	0.462	0.462
LeBron James	**0.423**	0.459	0.459	0.469
Monta Ellis	0.519	0.462	**0.365**	0.471

- We performed cross-validation to determine the number of nodes.
- Each player used a different Neural Network

3 NN Parameter Estimates



3 SVM Tuning for NBA Data

- `svm` function from `e1071` package in R
- Radial basis function kernel: $K(\boldsymbol{x_0}, \boldsymbol{x_i}) = \exp\{-\gamma||\boldsymbol{x_0} - \boldsymbol{x_i}||\}$
- Tune C and γ using cross-validation on a grid of possible values
- Assess prediction performance training on 90% of the data, and testing on remaining 10% multiple times (Monte Carlo)

Player	C	γ
Damian Lillard	4	2^{-8}
James Harden	2	2^{-8}
LaMarcus Aldridge	4	2^{-10}
LeBron James	4	2^{-11}
Monta Ellis	1	2^{-8}

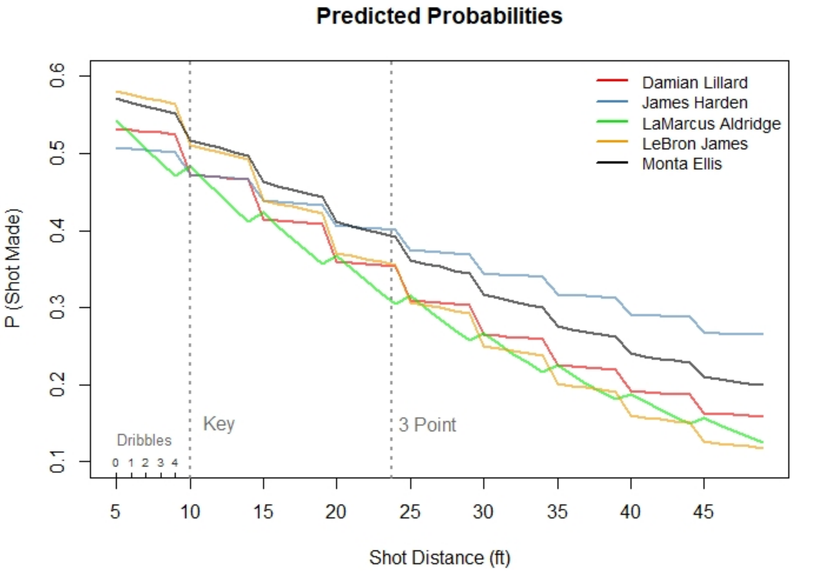
3 Comparison of Prediction Accuracies

Algorithm	Damian Lillard	James Harden	LaMarcus Aldridge	LeBron James	Monta Ellis
SVM	0.621	0.573	**0.612**	**0.635**	0.588
Neural Net	**0.625**	**0.577**	0.577	0.577	**0.635**

- SVM had the best overall accuracy (0.605 vs 0.598).
- However, accuracy doesn't tell the whole story (details in results section)
- We chose the SVM algorithm because of accuracy and sensible answers to research questions.

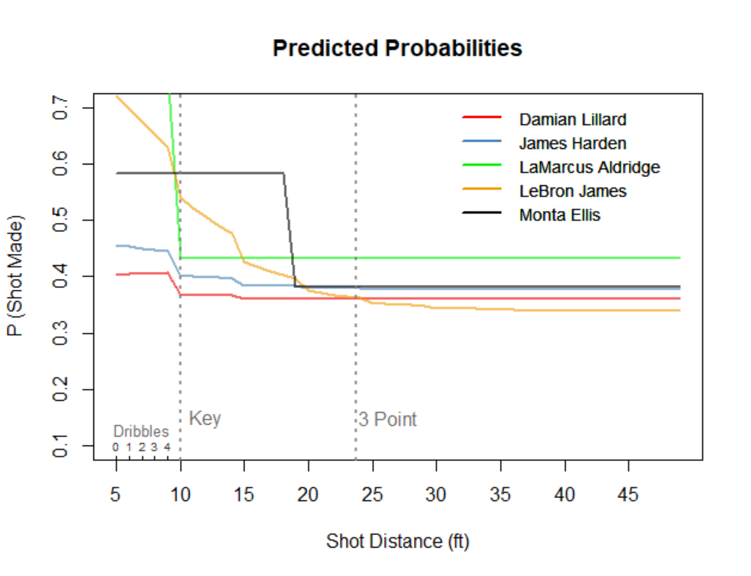
RESULTS

4 Q1: SVM



Predicted Probabilities
Damian Lillard
James Harden
LaMarcus Aldridge
LeBron James
Monta Ellis
Key
3 Point
Dribbles
0 1 2 3 4
P (Shot Made)
0.1
0.2
0.3
0.4
0.5
0.6
5
10
15
20
25
30
35
40
45
Shot Distance (ft)

4 Q1: NN



Predicted Probabilities
Damian Lillard
James Harden
LaMarcus Aldridge
LeBron James
Monta Ellis
P (Shot Made)
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Key
3 Point
Dribbles
0 1 2 3 4
5
10
15
20
25
30
35
40
45
Shot Distance (ft)

4 Q2: Shooter or Driver?

A "Shooter" is defined as:

- No Dribbles
- 23ft shot dist
- 6ft closest defender

A "Driver" is defined as:

- 5 Dribbles
- 1ft shot dist
- 2ft closest defender

Goal: Classify the tendency of each player

Problem: Almost any given player is more likely to make a layup or dunk than a three pointer

Predicted probabilities (NN)

Type	Damian Lillard	James Harden	LaMarcus Aldridge	LeBron James	Monta Ellis
Driver	0.702	0.640	0.768	0.672	0.583
Shooter	0.361	0.386	0.433	0.375	0.381

Predicted probabilities (SVM)

Type	Damian Lillard	James Harden	LaMarcus Aldridge	LeBron James	Monta Ellis
Driver	0.581	0.510	0.504	0.644	0.551
Shooter	0.353	0.421	0.344	0.341	0.356

Shooter or Driver? Possible options:

1. Naive classification - doesn't make sense
2. Compare FG% ratio to league averages
3. Use shot tendency data - which we don't have

4 Q2: Shooter or Driver?

Predicted probabilities (SVM)

Type	Damian Lillard	James Harden	LaMarcus Aldridge	LeBron James	Monta Ellis
Driver	0.581	0.510	0.504	0.644	0.551
Shooter	0.353	0.421	0.344	0.341	0.356

League average for $\frac{CloseShot\%}{Open3pt\%} = 1.532.$

Classifications (SVM)

Type	Damian Lillard	James Harden	LaMarcus Aldridge	LeBron James	Monta Ellis
Ratio	1.645	1.211	1.465	1.889	1.548
Classification	Driver	Shooter	Shooter	Driver	Driver

Note: Aldridge and Ellis could realistically be considered as either type

4 Q3: "Clutch" players

Consider the following 5 scenarios:

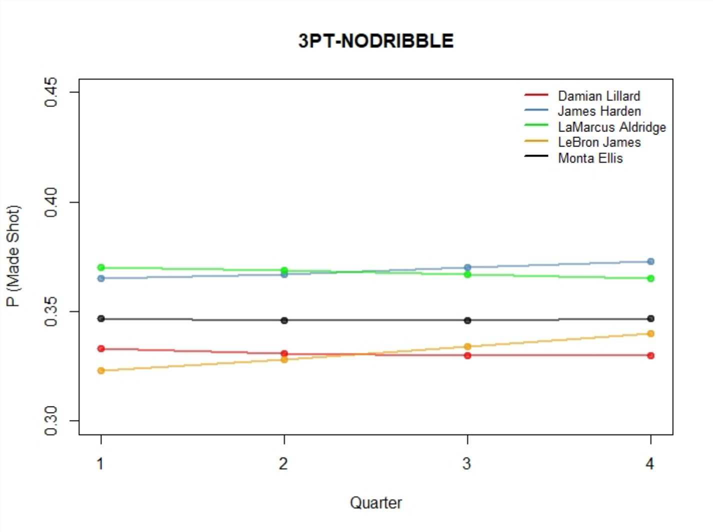
Away game, Mid 4th quarter, with 7 sec on shot clock

- **3 Pointers** (23.75ft from basket).
  - 0 Dribbles, median defender distance (3.7ft)
  - 3 dribbles, 3.7ft defender distance
- **2 pointers** (12ft from basket) and 3 dribbles.
  - Shot defender 0 ft
  - Shot defender 3.5ft
- **Dunk/Layup** 0ft from basket, 5 dribbles, 2.5 defender distance

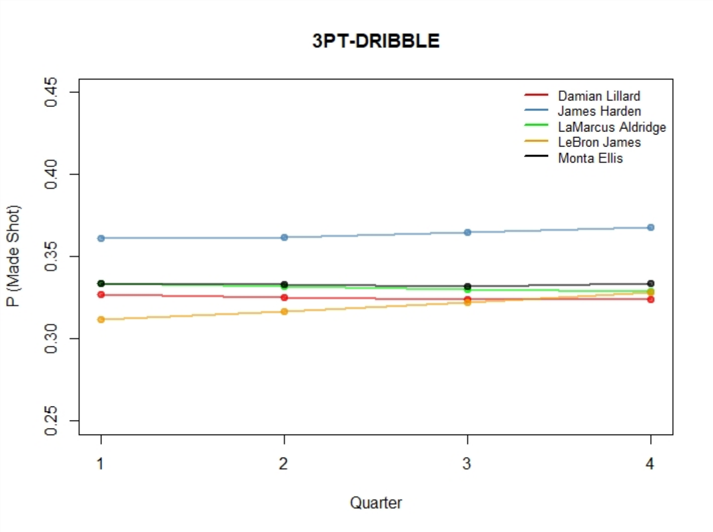
4 Q3: Average 4th Quarter Difference

Damian Lillard	James Harden	LaMarcus Aldridge	LeBron James	Monta Ellis
-0.002	0.007	-0.003	0.012	0.002

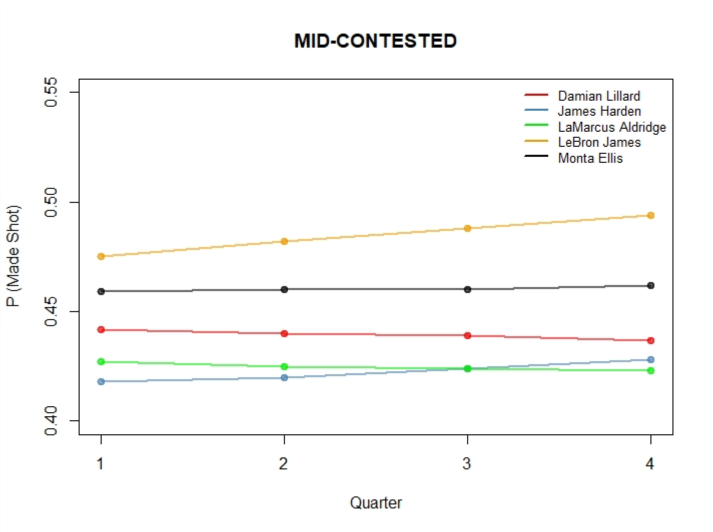
4 Q3: 3-Point Shot, No Dribbles



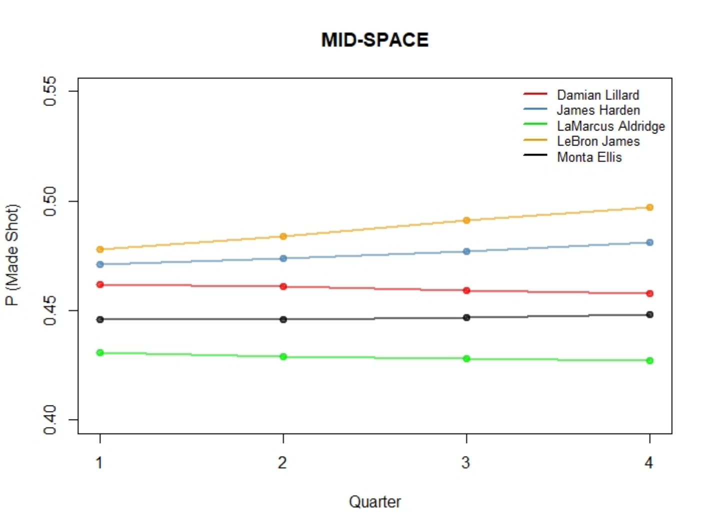
4 Q3: 3-Point Shot, 3 Dribbles



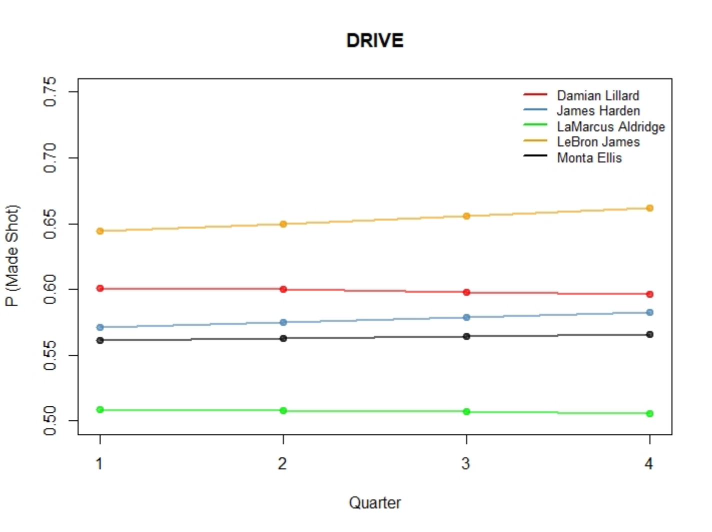
4 Q3: Mid-range Shot, Tight Defense



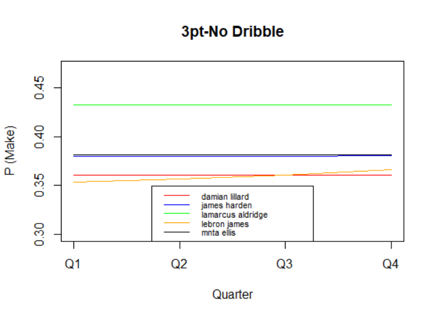
4 Q3: Mid-range Shot, Loose Defense

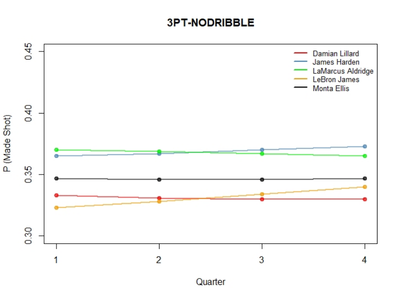


4 Q3: Drive to the Basket



4 Q3: Comparison to NN



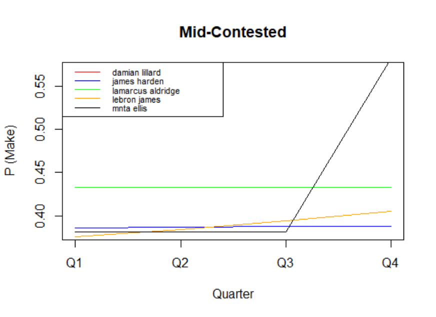


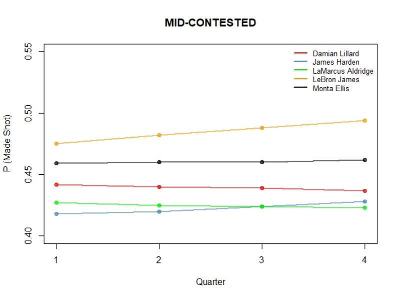
Notes:

- LaMarcus is much higher
- Damian/LeBron are relatively the same

4 Q3: Comparison to NN

However, some don’t look so similar





Notes:

- Neural Network shows some funky behavior

CONCLUSIONS

Final Algorithm Decision

- In the end, we decided on the SVM algorithm.
- Although the NN had decent accuracy, it had insensible graphs in the results section.

5 Summary of results

- With the SVM, we found that Lebron/Monta shoot best up close and James Harden shoots the best farther away in last-second situations.
- We also found that dribbles tend to have a negative impact on the chances of making a shot for all players, though Harden and Lillard are not as impacted as the others.
- Lebron James was the best "driver" and James Harden was the best "shooter".
- James Harden and Lebron James both improve in the 4th quarter, but Lebron is better overall. Lebron James is the "clutch" player.

5 Shortcomings

- Unpredictability and randomness of data (Low accuracies)
- No interpretability from the model (black box)
- No assessment of uncertainty

5 Moving Forward

More informative data

- Point differential at time of shot
- Who the defenders are
- Hot Hand / Player-specific Information