# Supervised Machine Learning Algorithms for Predicting National Basketball Association (NBA) Playoff Contention

---

# THESIS

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

## BACHELOR OF SCIENCE (Electrical Engineering)

at the

## NEW YORK UNIVERSITY
## TANDON SCHOOL OF ENGINEERING

by

**Emanuel Azcona**

**May 2017**

Approved:

Prof. Ivan Selesnick

Date

Prof. Shivendra Panwar

Date

University ID#: N12070168

Net ID: eaa349

# Vita

Emanuel Aquiles Azcona was born in Brooklyn, New York, United States, on 15 February 1994, the son of Ramona Castillo and Rafael Azcona. As a first-generation American, Emanuel was the first in his family to complete a degree at the High School for Construction Trades, Engineering, & Architecture, Ozone Park, New York, in 2012. That same year, he entered the New York University Tandon School of Engineering and received his degree of Bachelor of Science in Electrical Engineering in May, 2017.

During his stay at the Tandon School of Engineering, he served as the university's Vice President for the Institute of Electrical & Electronic Engineers (IEEE) club (2014-2015), was elected Public Relations Chair of the Society of Hispanic Professional Engineers (SHPE) (2013-2014), aided as a tutor for the NYU Higher Education Opportunity Program (2013-2017), and served as a Research Assistant through the NYU Undergraduate Summer Research Program (2014-2016).

Emanuel's research was first initiated through Professor Sundeep Rangan's Special Topics in Electrical Engineering: Introduction to Machine Learning course and continued as a part of an undergraduate senior thesis in Electrical Engineering course with Professor Ivan Selesnick. Emanuel will continue his studies as a Ph.D. student at Northwestern University, McCormick School of Engineering under the advisement of Joseph Cummings Professor Aggelos Katsaggelos in the Katsaggelos Image & Video Processing Lab.

# Acknowledgements

Placeholder.

Emanuel Azcona, New York University Tandon School of Engineering

May 8, 2017

# ABSTRACT

---

## Supervised Machine Learning Algorithms for Predicting National Basketball Association (NBA) Playoff Contention

**by**

**Emanuel Azcona**

**Advisor: Prof. Ivan Selesnick**

**Submitted in Partial Fulfillment of the Requirements**

**for the Degree of Bachelor of Science (Electrical Engineering)**

**May 2017**

Sports data analytics, data mining, and predictive analysis of sporting game outcomes are increasingly becoming more comprehensive as data collection methods continue to advance. Before data mining, sports organizations traditionally depended on human experience from coaches, scouting agents, managers, and players for predictive analysis. As the scope of data continually becomes more complex, research in machine automated precictive analysis draws a wide concern for research in event prediction of sporting matchups.

We build on work by Pedregosa et al. [7, 17] for applicable Python [19] software with pre-developed classifiers. We use 4 different classification algorithms from [7,17] for predicting what teams in the National Basketball Association (NBA) would advance to their current year's NBA Playoffs. Afterwards we use the accuracy of the classifiers and compare their results to determine the most accurate and cost-effective algorithm.

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# Chapter 1

# Introduction

The Internet and sports fanaticism have lead to the explosive growth in sports data analytics. Today, there is no mature technique to help us understand, analyze, and convert the data into predictable knowledge with 100% accuracy. In the past, sports analytics relied on the experience of experts and leaders who have coached, played, or scouted for the particular organization to make comparisons, and synthesize accurate predictions. At times, the reliability of expert experience and intuition can still not discover all the value and potential of collected data, especially the large sets of data in our control today.

As data in sports analytics continues to grow and become more boundless, a more scientific approach is desired to use the data and make more accurate predictions with it. Inherent prejudices and the fallibility of human memory mean that the brain is an inefficient tool for processing complex information, especially in the time required during sporting events. This is especially true for team sports, where they must monitor a number of players at once. One specific tool analysts are using to transform the data being gathered into accountable insights is machine learning [22].

Our studies focus on utilizing basketball data to predict the post-season status of teams in the National Basketball Association (NBA). Every season, about half of the teams in the NBA are selected to participate in the NBA Playoffs, a tournament style competition after the regular NBA season, often referred to as the NBA post-season. Our focus is to apply and analyze the results of several supervised machine learning classification algorithms in order to help us predict what teams will be chosen to participate in the NBA Playoffs and also assist us in determining the best classification algorithm to use in future cases.

## 1.1 Basketball Introduction

### 1.1.1 Brief Overview of the Game and its History

Basketball has become the most popular indoor sport played in the United States. Not long ago, it was invented by Dr. James Naismith in the year 1891 in Springfield, Massachusetts. The concept of the game was born from Naismith's school days in Ontario, Canada where he played a simple child's game known as duck-on-a-rock. The game involved attempting to knock a "duck" off the top of a large rock by tossing another rock at it [5, 9].

The National Basketball Association (NBA) was set up in 1949, and now hosts rich basketball stars before huge audiences. Many of the United States major cities' are represented by a team organization, as well as the Canadian city of Toronto. Often, some of the states with cities of dense populations will have multiple teams (1 for each of the cities) to represent them. One major example is Texas, which houses multiple teams including:

- Dallas Mavericks
- Houston Rockets
- San Antonio Spurs

Certain cities with extremely dense populations house more than 1 team. These cities include:

- New York
  - New York Knicks
  - Brooklyn Nets
- Los Angeles
  - Los Angeles Lakers
  - Los Angeles Clippers

The game of basketball is played with 2 opposing teams of 5 players on the court, and 7+ on the bench that can be swapped in to replace 1 of the 5 for their team during the game. The traditional basketball court is depicted in Figure 1.1 below and its layout plays a vital role in player positioning as well as scoring. The objective of the game is to score more points than the opposing team, while trying to prevent them from doing the same. Basketball is played in four 12-minute segments called "quarters." At the beginning of the

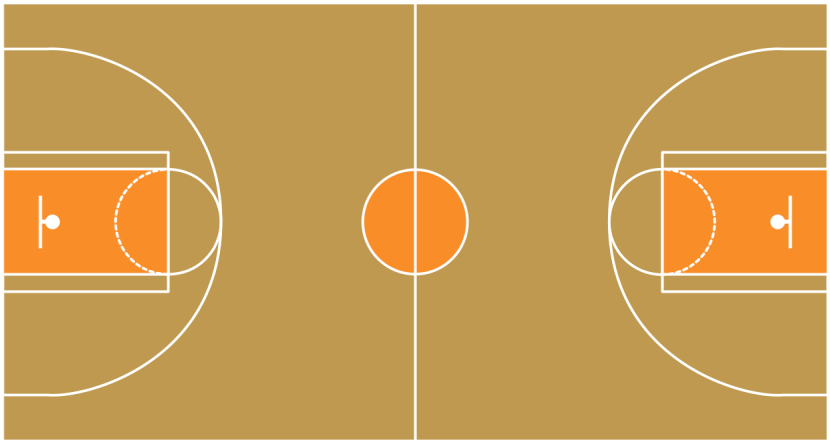

Figure 1.1: Traditional basketball court layout

game, 2 players (typically the tallest from each team, referred to as "centers") are called to the center of the court (circular area in the center of the court in Figure 1.1) and compete in a "tip-off" where the basketball is thrown into the air and the players leap and try to tip the ball to their team's side for the first possession.

### 1.1.2 Season & Conference Structure

In the regular NBA season, there are 82 total games played per team amongst the 30 teams that exist (29 teams prior to the addition of the New Orleans organization). The 82 games of the regular season are spread from October to April, 41 home and 41 away.

The NBA is divided into 2 geographical conferences:

- Western
- Eastern

and teams from opposing conferences will only play each other 2 times, meanwhile a team may play another team from their respective conference up to 3 times [16]. After the completion of the 82 games, the top 8 teams from each conference are selected to play in a tournament style competition called the NBA Playoffs. The structure of the NBA Playoffs is demonstrated in Figure 1.2.

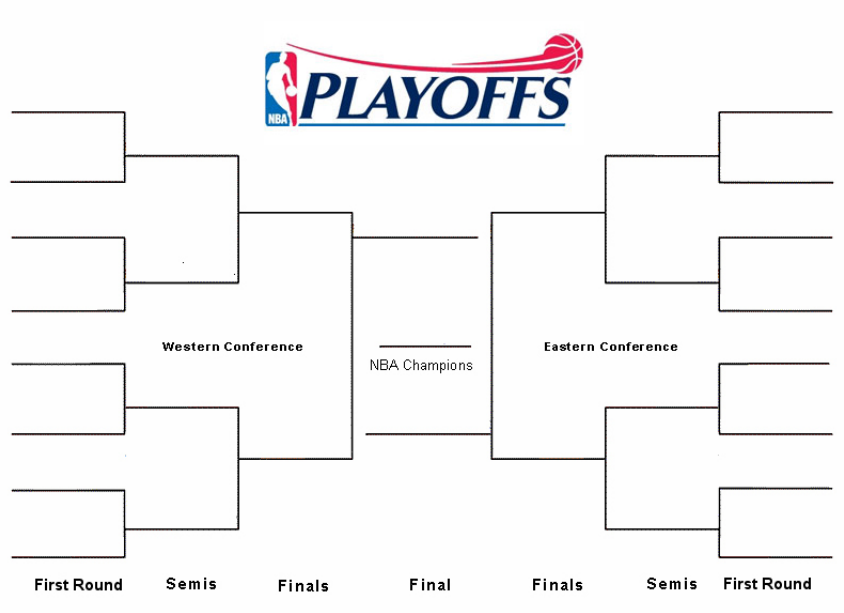


Figure 1.2: NBA Playoff bracket structure

In the NBA Playoffs, respective teams of each conference play against one another until there is a single conference champion, afterwards both conference champions (Western & Eastern) play against each other in the NBA Finals, where the winning team is crowned the NBA Champion of that NBA season.

### 1.1.3 Points (PTS)

In basketball, points are used to keep track of score in a game. Points can be accumulated in 1 of 3 ways:

- Field Goals (FG)
  - Two Pointer (2M)
  - Three Pointer (3M)
- Free Throws (FT)

In Figure 1.1, the semi-circle resembling arcs on both sides of the traditional basketball court is often referred to as the 3-point line. If players score within the confinement of the 3-point line, their contribution is worth 2 points. Shooting the ball and scoring from outside of the 3-point line boundary allows the contributor's team to earn 3 points. Free throws, which will be explained in the following subsections, earns a team 1 point for every free throw made. At the end of the game, the team with the most recorded points is declared

the winning team [16]. Wilt Chamberlain has the record for the most points scored in a single game in NBA history with 100 points [14]. Kareem Abdul-Jabbar has the record for the most cumulative career points in NBA history with $38,387$ [13]. Michael Jordan holds the record for the most points per game (PPG) average in the regular season with 30.12 PPG [15].

### 1.1.4 Field Goals (FG)

In basketball, field goal is a term used to refer to a basket scored on any shot or tap other than a free throw. Field goals can be worth 2 or 3 points depending on a player's positioning on the basketball court. "Field goal" is the official term used by the NBA in their rule book, box scores and statistics. The same terminology is used in the handbooks and rulebooks of the National Collegiate Athletic Association (NCAA) and high school basketball leagues across the world. Regarding statistics, there are 3 types of field goal statistics, Field Goals Made (FGM), Field Goals Attempts (FGA), and Field Goal Percentage (FG%). FG% is calculated such that:

$$\text{FG\%} = \frac{\text{FGM}}{\text{FGA}} \tag{1.1}$$

Wilt Chamberlain holds the record for the most recorded FGM and FGA per game in the NBA regular season with 12.1 FGM and 22.5 FGA per game [3]. Kareem Abdul-Jabbar hold the record for the most regular season FGM and FGA with 15837 FGM and 28307 FGA. DeAndre Jordan holds the record for the highest regular season FG% with 67.5% [3].

### 1.1.5 Free Throws (FT)

"Free throws," or "foul shots" are unopposed attempts to score points from a restricted area on the court. The restricted area can be seen in Figure 1.1, as the brown semi-circle region before the orange rectangle on either side of the court. Free throws are generally awarded after a foul is committed on the shooter by the opposing team. Fouls will be explained in the following subsection. There are 3 types of statistics directly associated to free throws, Free Throws Made (FTM), Free Throw Attempts (FTA), and Free Throw Percentage (FT%). FT% is determined such that:

$$\text{FT\%} = \frac{\text{FTM}}{\text{FTA}} \tag{1.2}$$

Karl Malone holds the record for the most regular season FTM and FTA with 9787 and 13188 respectively. Steve Nash holds the record for the highest regular season FT% with 90.4% [3].

### 1.1.6 Personal Fouls (PF)

Personal fouls are breaches of the rules that concern illegal personal contact with an opponent in basketball. In the NBA, players are restricted to only committing a maximum of 5 fouls per game. After 5 fouls, players "foul out" and exceed their personal foul limit, disqualifying them from participating for the remainder of the game. Types of fouls can range from:

- Charging and blocking
- Fouls on the ball-carrier
- Fouls away from the ball
- Pushing
- Holding
- Illegal use of hands
- Hand-checking (impeding opposition speed or balance of hands)
- Illegal use of elbow
- Illegal use of legs or knees
- Clear path fouls

From these types of fouls, only 2 types of events occur that would award "the fouled," opportunities to attempt free throws:

- Bonus
- Shooting Foul

Every quarter, teams are limited to the number of fouls their players can commit overall. If a team exceeds the foul limit, they enter a penalty situation, often referred to as the "bonus," where the opposition is offered 2 free throws for every foul the committing team commits again afterwards, starting with the foul that lead into the bonus.

Shooting fouls occur when the ball handler is fouled during a field goal attempt. Depending on where the ball handler is positioned, a shooting foul from the opposition could earn the ball handler 1, 2 or 3 free throw attempts. Unlike the bonus, shooting fouls typically award the ball handler immediate free throw attempts, regardless of penalty status. However,

shooting fouls are also counted when keeping track of the number of fouls committed by a team and their bonus status. Kareem Abdul-Jabbar leads the NBA in most career fouls with 4657 personal fouls.

### 1.1.7 Rebounds (REB)

Rebounds, often referred to as "boards," are a statistic awarded to players who retrieve missed field goal or free throw attempts. Players who tip in missed shots by themselves or their team members also earn rebounds. After shots are made, regardless of whether or not the basket is scored, possessions change in basketball, and often rebounding is a vital and routine part of the game, ensuring the rebounder's team possession of the ball. Rebounds can be grabbed by players from the offensive or defensive team. Often, data analysts also like to keep track of types of rebounds, defensive or offensive, to demonstrate a player's importance in ensuring more possessions of the ball for their team. Wilt Chamberlain led the NBA in rebounds in 11 different seasons and has the most career rebounds in the regular season, with $23,924$. [3].

### 1.1.8 Assists (AS)

In basketball, an assist is a statistical attribute awarded to players who pass the ball to a teammate in a way that leads to a score by field goal. In other words, the ball handler "assisted" the scorer in the making of the basket. Only the pass directly before the score may be counted as an assist and passes that lead to a shooting foul and scoring by free throws does not count in the NBA. John Stockton holds the NBA record for the most career assists per game average with 11.2 assists per game (APG). John Stockton also has the record for the most career assists with $15,806$ assists [3].

### 1.1.9 Steals (ST)

A steal occurs when a defensive player legally causes a turnover with positive aggressive action. This is often done by deflecting and controlling the ball from the ball handler, or by catching the pass or dribble of an offensive player. In order for the attempt to count as a steal, the defender must not touch the hands of the ball handler, otherwise a foul is called. Even if a deflector does not end up with the ball, steals are typically credited to the defensive players who end up with the ball. John Stockton holds the NBA record for the most career steals with $3,625$, followed by Jason Kidd with $2,684$ career steals [3].

### 1.1.10 Blocks (BK)

"Blocks," or blocked field goal attempts, are events that occur when a defensive player legally deflects a field goal attempt from an offensive player. If the defender makes contact with the scorer's hand (unless contact with the ball) a foul is called. For blocks to be legal, the block needs to occur when the shot is traveling upward or at its apex. Hakeem Olajuwon holds the NBA record for the most career blocks in the regular season with $3,830$ blocks, followed by Dikembe Mutombo with $3,289$ blocks [3].

### 1.1.11 Turnovers (TO)

"Turnovers," occur when a player causes their team to lose possession of the ball. Turnovers are typically the results of:

- ball being stolen
- making mistakes, like stepping out of bounds
- illegal screens
- double dribbles
- pass interceptions
- throwing the ball out of bounds
- three-second paint violation
- five-second back-court violation
- traveling
- shot clock violation
- offensive foul
- technical foul

Karl Malone holds the NBA record for the most career turnovers with $4,524$ [3].

## 1.2 Supervised Machine Learning

The basic idea of machine learning is to obtain the probability distribution of input data and use new data to make accurate predictions, complete a task ,or behave intelligently. In general, machine learning is about learning to "do better" in the future based on instructions and their outcomes from the past. The goal of machine learning is to automate tasks without human intervention or assistance [20].

The difference between supervised and unsupervised machine learning algorithms is the lack of learning *with* outputs in unsupervised learning. Training a machine learning task with corresponding "target" variables, or outputs, is referred to as supervised learning. With a trained model, predicting new targets for new inputs are made sufficient. If targets are expressed in the form of different classes, the task/problem is referred to as a classification problem. If the target space is continuous, it is then referred to as a regression problem. Several examples of supervised machine learning problems, but not limited to, include:

- optical character recognition: categorization of images of handwritten characters
- classification of faces in images (or indication of present faces)
- identification of email messages as spam or non-spam
- categorization of news articles based on topic
- medical diagnosis
- prediction of customer response to particular promotions/advertisements
- identification of credit card transactions which may fraudulent in nature
- weather prediction

In this study, several supervised learning approaches are analyzed in predicting NBA Playoff contention. The supervised classification algorithms analyzed include:

- Logistic Regression
  - $\ell 1$-penalized
  - $\ell 2$-penalized
  - unregularized
- Support Vector Machines (SVM)

  - Radial basis kernel
  - Polynomial kernel
  - Linear kernel
  - Sigmoid kernel
- Random Forest
- Decision Tree

### 1.2.1 Logistic Regression & Classification

Logistic regression is typically the go-to method for binary classification problems (problems with distinguishing samples among 2 different classes) borrowed by machine learning from the field of statistics. Logistic regression, at its core, involves the application of the logistic function, also called the sigmoid curve, to describe properties of population growth. The traditional "S-shaped" curve can take any real-valued number and map it to a value between 0 and 1, but never exactly at those limits. Traditionally, sigmoid function is expressed as:

$$\frac{1}{1+e^{-x}}. \tag{1.3}$$

Like linear regression, logistic regression involves the linear combination of weights, or coefficients, along with input values, $x$, to predict a target, $\hat{y}$. One example of single-feature logistic regression can be expressed as:

$$p(\mathbf{y}|\mathbf{X}) = \frac{e^{\beta_0+\beta_1\mathbf{X}}}{1+e^{\beta_0+\beta_1\mathbf{X}}}. \tag{1.4}$$

With some manipulation, we can obtain:

$$\ln\left(\frac{p(\mathbf{y}|\mathbf{X})}{1-p(\mathbf{X})}\right) = \beta_0 + \beta_1\mathbf{X}, \tag{1.5}$$

an equation of logarithmic, or "log-odds". Odds are the ratio of the probability of an event divided by the probability of the event not occurring which we can simply write as:

$$\ln(\text{odds}) = \beta_0 + \beta_1\mathbf{X} \tag{1.6}$$

With Equations 1.4 and 1.6, we can analyze the probability of an event occurring and can select our own probability thresholds for distinguishing between 2 classes given the probability that a certain sample belongs to a specific class.

As an example, we can compute the probability that a person is male or female based on their height (fictitious example). Given a height of 150cm and the model coefficients, or weight vector, we want to find out whether the person is male or female. Pretend we are given the weight vector as:

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} -100 \\ 0.6 \end{bmatrix}.$$

As previously stated, we are interested in solving the problem:

$$p(\mathbf{y} = \text{male}|\mathbf{X} = 150),$$

so we formulate a solution to our problem using Equation 1.4 and obtain:

$$\begin{aligned} p(\mathbf{y} = \text{male}|\mathbf{X} = 150) &= \frac{e^{(-100+0.6(150))}}{1 + e^{(-100+0.6(150))}} \\ &= 4.53978 \times 10^{-5}. \end{aligned}$$

The probability obtained is near zero in this case, so it is safe for our machine to assume that the 150cm tall person is not male based on our trained model.

For the purposes of our analysis, we use a Python implementation of the logistic regression model from [17]. With a simple manipulation of variables, given the target vector, $\mathbf{y}$, and the feature matrix, $\mathbf{X}$, we can determine the weight coefficient vector $\boldsymbol{\beta}$, and train our logistic regression model.

### 1.2.2 Support Vector Machine (SVM)

Support vector machines, despite their slow runtime with large data, are extremely popular and relatively simple to implement. To best explain how SVMs work, its best to start off by understanding how the Maximal-Margin Classifier works. Hyperplanes are selected in SVMs to best separate the points in the input variable space by their class. For example:

$$\beta_0 + \beta_1 X_2 + \beta_2 X_2 = 0, \tag{1.7}$$

where the weight vector is defined as:

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}, \tag{1.8}$$

determining the slope and intercept of the separating hyperplane. Using this separating hyperplane, we can make classifications by inserting the input values into the line equation, and calculating whether output lies on a certain side of the hyperplane. The distance between the closest data points and the separating hyperplane is called the margin and is calculated as the perpendicular distance from the line to those points. The best or optimal line that can separate the 2 classes is the line with the largest margin to the data. Typically, this is referred to as the Maximal-Margin Hyperplane. The data points that construct the classifier are called the support vectors, because they support the definition of the hyperplane.

A similar classifier, the Soft Margin Classifier, is more sensitive to messy data that can not be perfectly separated with a hyperplane without error. SVM relies on a "kernel trick" which uses a kernel function to compute the dot product between features as a measure of similarity. During learning for SVM, every time $f(x_i)$ and $f(x_j)$ need to be computed, we can instead compute the Kernel function,

$$K(x_i, x_j), \tag{1.9}$$

which would allow us to implicitly map data to a higher dimensional space in the hope that the data could become more easily separated in that space.

The Scikit-learn library [17] provides implementations of several SVM models with specified kernel functions. The kernel functions focused on in this experiment are:

- Linear kernel
- Radial basis kernel
- Polynomial kernel
- Sigmoid kernel

The linear kernel is easily described such that:

$$K(x_i, x_j) = x_i^\intercal x_j + c, \tag{1.10}$$

with the optional constant parameter, $c$, to serve as a kernel offset adjustment. The radial basis kernel is defined as:

$$K(x_i, x_j) = \exp\left\{-\frac{1}{2\sigma^2}\left\|x_i - x_j\right\|^2\right\}, \tag{1.11}$$

where $\sigma$ is a free parameter. A simpler form involving the free parameter, $\gamma = \frac{1}{2\sigma^2}$, can be

expressed as:

$$K(x_i, x_j) = \exp\left\{-\gamma \left\|x_i - x_j\right\|^2\right\} \tag{1.12}$$

The polynomial kernel is described as:

$$K(x_i, x_j) = \left(\alpha x_i^\intercal x_j + c\right)^d, \tag{1.13}$$

with sell-suited adjustable parameters: the slope, $\alpha$, constant term, $c$, and polynomial degree, $d$. The sigmoid kernel is defined as:

$$K(x_i, x_j) = \tanh\left(\alpha x_i^\intercal x_j + c\right), \tag{1.14}$$

where the adjustable parameters, $\alpha$, and $c$, can be easily manipulated. A common value for $\alpha$ is $\frac{1}{N}$, where $N$ is the data dimension.

### 1.2.3 Decision Tree Classifier

A classic example to explain the concept behind decision tree classification involves determining whether or not a person can play tennis based on the weather, as seen in Figure 1.3.

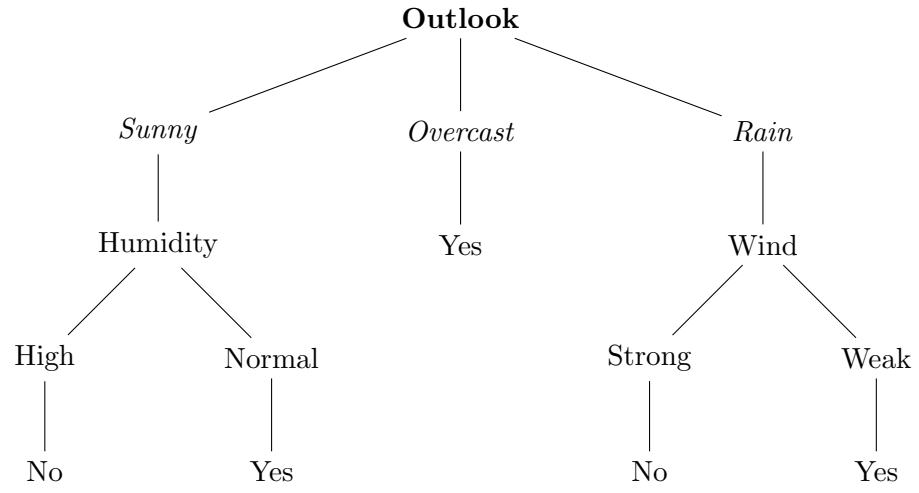


Figure 1.3: Decision tree example on weather-based tennis play outlook.

Decision Tree classification in this simple example demonstrates the feasibility and effectiveness of using these data structures in that decisions are made concretely based on simple questions (in most cases binary) rather than rely on probabilistic methods. A common problem with Decision Tree classification is an over-fitting trend due to its concrete nature and "no turning back" approach. This problem is easily solved with Random Forest

classification.

### 1.2.4 Random Forest Classifier

"Random Forests" in statistical learning have been referred to as an ensemble of classification, regression, clustering, and other algorithms. Their name was coined from their organizational characteristic in the multitudes of decision trees that are created during training and the output/predictions being the mode of the classes or mean prediction of the individual trees. Using Random Forests corrects the over-fitting habit of classical decision tree algorithms in regression and classification.

Decision tree algorithms are seldom accurate and when they are grown very deep, they tend to learn highly irregular patterns. In other words, decision tree algorithms tend to over-fit the data they are training with. This over-fitting is accompanied by low bias, and very high variance. Using Random Forests, we average multiple decision trees instead of relying on one deep instance, where each tree can be trained on different parts of the same training set.

However, Random Forests do come with a price. Random Forest models lead to increases in bias and loss of interpretation of how results are determined. With their accompanying price, Random Forests are great for their boost in performance, accuracy, and training time. An example of Random Forest regression can be seen in Figure 1.4. As seen below, the total regression output of the ensemble is the mean of the individual Decision Tree regression outputs.

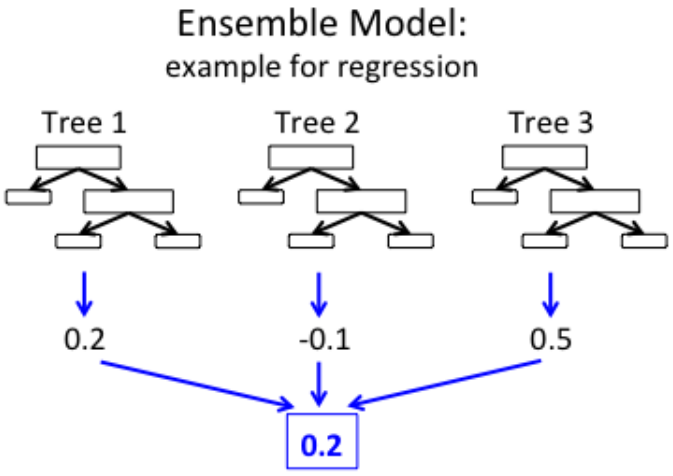


Figure 1.4: Random Forest ensemble regression model.

# Chapter 2

# Materials

The student was provided with access to computing resources in NYU Tandon School of Engineering's Bern Dibner Library Building's Multimedia Laboratory. The student had access to desktop computers with a Linux environment built-in. This resource was utilized to compute and test code on a multiprocessor computer for faster computation time, when compared to a standard laptop computer.

All the software developed was scripted in Python with the use of the following libraries:

- sys
- NumPy
- Plotly
- Pandas
- copy
- operator
- collections
- Sci-kit learn (sklearn)
- StatsModels
- _pickle
- time

Python [19] was installed using Continuum's Anaconda free software [1] installer. The downloaded packages not included in the Python version from [1] are colored like this

and can be downloaded from the web through the developers' websites or through Python package installers (pip, conda, etc.)

As for the writing of this thesis, the student conducted all development of this text using LaTeX. The dataset used for this thesis was readily available and extracted from the website: `dougstats.com` [2].

# Chapter 3

# Methodology

## 3.1 Datasets

The dataset was extracted from [2]. This website, owned by an HP engineer by the name of Doug, provides up-to-date data for players and teams in the NBA and Major League Baseball (MLB). This site contains a collection of `.txt` files for each NBA season, which were converted (saved as) comma- separated values (`.csv`) files.

## 3.2 Data Organization

A user-defined function was scripted using Python to create two ordered dictionaries with the starting season's year as the key, and the values being Pandas dataframes of the corresponding data read in from the respective `.csv` file. An example snippet of the player data from the 2001-2002 NBA season is demonstrated below in Figure 3.1.

| Player | Team | PS | GP | Min | FGM | FGA | 3M | 3A | FTM | FTA | OR | TR | AS | ST | TO | BK | PF | DQ | PTS | TC | EJ | FF | MVP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| abdul-wahad,tari | dal | SG | 24 | 441 | 55 | 147 | 1 | 2 | 24 | 33 | 41 | 84 | 24 | 20 | 27 | 10 | 56 | 1 | 135 | 0 | 0 | 0 | FALSE |
| abdur-rahim,shar | atl | PF | 77 | 2980 | 598 | 1297 | 21 | 70 | 419 | 523 | 198 | 696 | 239 | 98 | 250 | 81 | 214 | 2 | 1636 | 17 | 1 | 0 | FALSE |
| alexander,courtn | was | SG | 56 | 1325 | 223 | 474 | 5 | 18 | 98 | 121 | 43 | 148 | 86 | 35 | 60 | 7 | 113 | 0 | 549 | 0 | 0 | 1 | FALSE |
| alexander,victor | det | PF | 15 | 97 | 18 | 51 | 0 | 2 | 4 | 8 | 7 | 29 | 6 | 0 | 5 | 1 | 6 | 0 | 40 | 0 | 0 | 0 | FALSE |
| allen,malik | mia | PF | 12 | 161 | 22 | 51 | 0 | 1 | 8 | 10 | 15 | 38 | 5 | 3 | 2 | 8 | 16 | 0 | 52 | 0 | 0 | 0 | FALSE |
| allen,ray | mil | SG | 69 | 2525 | 530 | 1148 | 229 | 528 | 214 | 245 | 81 | 312 | 271 | 88 | 159 | 18 | 157 | 0 | 1503 | 2 | 0 | 0 | FALSE |
| alston,rafer | mil | PG | 50 | 600 | 66 | 191 | 27 | 71 | 18 | 29 | 10 | 72 | 143 | 32 | 40 | 2 | 42 | 0 | 177 | 1 | 0 | 0 | FALSE |
| amaechi,john | uta | C | 54 | 586 | 54 | 166 | 0 | 0 | 67 | 105 | 52 | 109 | 29 | 7 | 55 | 10 | 64 | 0 | 175 | 2 | 0 | 0 | FALSE |
| anderson,chris | den | C | 24 | 262 | 25 | 74 | 0 | 4 | 22 | 28 | 38 | 76 | 7 | 7 | 13 | 28 | 35 | 0 | 72 | 0 | 0 | 0 | FALSE |
| anderson,derek | por | SG | 70 | 1860 | 247 | 612 | 85 | 228 | 178 | 208 | 47 | 189 | 216 | 68 | 87 | 8 | 113 | 1 | 757 | 0 | 0 | 0 | FALSE |
| anderson,kenny | bos | 77 | 76 | 2430 | 312 | 716 | 9 | 33 | 98 | 132 | 57 | 275 | 403 | 141 | 119 | 10 | 210 | 4 | 731 | 9 | 1 | 1 | FALSE |
| anderson,nick | mem | SG | 15 | 219 | 21 | 76 | 13 | 48 | 5 | 9 | 1 | 33 | 14 | 6 | 14 | 6 | 17 | 0 | 60 | 0 | 0 | 0 | FALSE |
| anderson,shandon | nyk | SF | 82 | 1596 | 149 | 373 | 39 | 141 | 74 | 107 | 57 | 249 | 76 | 48 | 97 | 15 | 134 | 1 | 411 | 1 | 1 | 2 | FALSE |

Figure 3.1: Fraction of player data in `.csv` file for 2001-2002 NBA season

The dictionaries were organized in the following manner:

- Players in each NBA season
  - key: start year of season
  - value: dataframe of teams and stats
- Teams in each NBA season

  - key: start year of season
  - value: dataframe of teams and stats

Next, a cumulative dictionary of NBA seasons was created with the following sample hierarchy of data:

- NBA season start year (ex: 15, for 2015-2016)
  - Team (ex: okl, for Oklahoma City Thunder)
    - Players (ex: durant, kevin)
      - GP
      - FGM
      - etc.

Parsing through the ordered dictionaries, we also create 2 dataframes, one containing every NBA player that's ever played in any season from 2001-2015 and another with every NBA team.

## 3.3 Analysis of Total Team Statistics

Two dataframes were created, one with the total sum of integer team stats and another with the average of fractional team stats across all the NBA seasons. The integer stats include:

- Field Goals Made (FGM)
- Free Throws Made (FTM)
- Three Pointers Made (3M)
- Offensive Rebounds (OR)
- Assists (AS)
- Steals (ST)
- Turnovers (TO)
- Blocks (BK)
- Personal Fouls (PF)

- Points Made (PTS),

and the fractional stats include:

- Field Goal Percentage (FG%)
- Three Point Percentage (3PT%)
- Free Throw Percentage (FT%)
- Rebound Percentage (OR/REB) (REB%).

The fractional and total integer stats were analyzed and plotted in Figures A.1 and A.2 in Appendix A.

## 3.4 Analysis of Individual Team Statistics by NBA Season

The behavior of NBA teams throughout the different seasons in the dataset (2001-2002 through 2015-2016) was analyzed in this segment of the experiment. Before attempting to analyze individual team behavior in every NBA season, "zeroed" copies of the New Orleans teams were added to the first three seasons of our NBA dataset because the franchise did not exist until 2004-2005. All the statistics related to the New Orleans franchise for those seasons were set to 0 by default. The same integer and fractional stats were analyzed for each team in every NBA season and was analyzed in the series of Figures A.3, A.4, A.5, A.6, A.7, and A.8 in Section A.

## 3.5 Statistical Significance of Team Stats to Playoff Status

In this process, we focus on the determination of the statistical significance of particular features to NBA Playoff contention. First, we take the dictionary of all NBA teams that have existed from 2001 to 2016 and create two NumPy arrays, one with the indices of playoff teams, $i_{play}$, and the other with the indices of non-playoff teams, $i_{non}$.

Then using the arrays of indices, $i_{play}$ and $i_{non}$, a statistical significance and $p$-value test was performed using [21], as seen in Algorithm 1:

**Algorithm 1** $p$-values using dataframe, $X$, and indices arrays, $i_{play}$ & $i_{non}$

**Require:**

- $y$ be an empty dataframe
- length($i_{play}$) + length($i_{non}$) = samples($X$)

1: **for** attribute in $X$ **do**
2: $play \leftarrow X$ [attribute] [$i_{play}$]
3: $non \leftarrow X$ [attribute] [$i_{non}$]
4: statistic, $p$-value = $z$-test($play, non$)
5: $y$.appendRow( [attribute, statistic, $p$-value] )
6: **end for**
7: **return** $y$

As a result of our own analysis we obtained results detailing the statistical significance of all our features, as can be seen on Table 3.1.

Table 3.1: Statistical Significance of NBA Team Features to Playoff Contention

| **Feature** | **Statistic** | $p$**-value** |
|---|---|---|
| FG% | 8.768 | $1.813 \times 10^{-18}$ |
| AS | 8.385 | $5.089 \times 10^{-17}$ |
| PTS | 7.405 | $1.308 \times 10^{-13}$ |
| FGM | 6.609 | $3.857 \times 10^{-11}$ |
| ST | 6.408 | $1.473 \times 10^{-10}$ |
| 3PT% | 6.251 | $4.078 \times 10^{-10}$ |
| 3M | 5.574 | $2.489 \times 10^{-8}$ |
| REB% | −5.208 | $1.900 \times 10^{-7}$ |
| BK | 5.141 | $2.720 \times 10^{-7}$ |
| FTM | 4.598 | $4.258 \times 10^{-6}$ |
| MVP | 3.707 | $2.101 \times 10^{-4}$ |
| FT% | 1.201 | 0.229 |
| TO | −0.700 | 0.484 |
| PF | −0.473 | 0.635 |
| OR | −0.015 | 0.987 |

Based off the results of our statistical significance test, our analysis showed that the

following features:

- Offensive Rebounding Percentage (OR/REB = REB%)
- Personal Fouls (PF)
- Turnovers (TO)
- Offensive Rebounds (OR)
- Free Throw Percentage (FTA/FTM = %)

did not have as much statistical significance to the other features in the determination of NBA Playoff contention. Therefore, we removed these features from our feature matrix, $X$, since they probabilistically reject our null hypothesis. As a result, our analysis only used the most significant attributes (in order from most significant down), which are:

- Field Goal Percentage (FGA/FGM = FG%)
- Assists (AS)
- Points (PTS)
- Field Goals Made (FGM)
- Steals (ST)
- Three Point Percentage (3A/3M = PT%)
- Threes Made (3M)
- Blocks (BK)

## 3.6 Logistic Regression Classification

### 3.6.1 Determination of Logistic Regression Classifier Regularization Strengths, $C$, Based on Penalties

Next, we develop logistic regression classifiers using [7, 17], each with different penalties, $\ell_1$, $\ell_2$, and un-normalized. To determine the regularization strength, $C$ for each of our logistic regression classifiers we train our models over the range of every team in every NBA season all at once, measure the accuracy of our prediction and obtain the regularization strength, $C$, that gives us the greatest accuracy for each of our models. In other words we follow Algorithm 2.

**Algorithm 2** Determination of logistic regression classifier regularization strength, $\mathbf{c}_{max}$

**Require:**

- $\mathbf{c}$, vector of log-spaced test strengths
- $\mathbf{c}_{max}$, 3-element vector for storing regularization strength that grants greatest accuracy for $\ell_1, \ell_2$, and un-normalized logistic regression
- $\mathbf{a}$, empty 3-element vector for accuracy vectors of each logistic regression classifier
- $\mathbf{a}_{max}$, 3-element vector for storing maximum accuracies, each element defaulted to 0
- $\mathbf{X}$, feature matrix
- $\mathbf{y}$, classification of each sample
- $\hat{\mathbf{y}}$, empty 3-element vector, each element for storing a prediction vector
- $\mathcal{M}$, empty 3-element vector for storing logistic regression models, $\ell_1$, $\ell_2$, and un-normalized respectively

Determination of logistic regression classifier regularization (cont'd)

**for** $c$ in **c do**
  **for** i in [$\ell_1$, $\ell_2$, un-normalized] **do**
    $\mathcal{M}(i) \leftarrow (\text{penalty} = i, C = c)$
    $\mathcal{M}(i) \leftarrow \text{fit}\,(\mathbf{X}, \mathbf{y})$
    $\hat{\mathbf{y}}(i) \leftarrow \mathcal{M}(i).\text{predict}(\mathbf{X})$
    $\mathbf{a}(i) \leftarrow \frac{\hat{\mathbf{y}}==\mathbf{y}}{N}$ where $N$ is the number of samples
    **if** $\mathbf{a}_{max}(i) < \mathbf{a}(i)$ **then**
      $\mathbf{a}_{max}(i) \leftarrow \mathbf{a}(i)$
      **if** $i$ is 0 **then**
        $\mathbf{c}_{max}(i) \leftarrow c_{\ell_1}$
      **else if** $i$ is 1 **then**
        $\mathbf{c}_{max}(i) \leftarrow c_{\ell_2}$
      **else**
        $\mathbf{c}_{max}(i) \leftarrow c$
      **end if**
    **end if**
  **end for**
**end for**
**return** $\mathbf{c}_{max}$

Using Algorithm 2, we obtained a vector of the regularization strengths necessary for our respective $\ell_1$, $\ell_2$, and un-normalized logistic regression classifiers, based on training a model repeatedly over the entire dataset of teams.

### 3.6.2 Creating Attribute and Target Training Datasets Based on NBA Season

Next, we focused on creating a dictionary of NBA team data for our "progressive-training" model. Using [19], two ordered dictionaries were created, one pertaining to attribute and target training data:

**Algorithm 3** Create ordered dictionary to obtain 3-D matrix of training data for predicting targets

**Require:**

- $\mathcal{D}$, an empty ordered-dictionary
- $\mathbf{n}$, vector of NBA team names
- $\mathcal{N}$, dictionary of NBA teams, each key refers to an NBA season, unordered
- $\mathbf{y}$, vector of start years to every NBA season (i.e. 2001 for $2001 - 2002$)
- $\mathbf{X}_{tr}$, empty vector for matrices at index representing the training attributes for a certain year
- $\mathbf{y}_{tr}$, empty vector for vectors at index representing the training targets for a certain year

**for** year in $\mathbf{y}$ **do**
  currSeason $\leftarrow \mathcal{N}$(year)
  **for** attribute in [REB%, PF, TO, OR, FT%, Conference] **do**
    remove currSeason[attribute]
  **end for**
  $\mathcal{D}$(year) $\leftarrow$ currSeason
**end for**
**for** year in $\mathbf{y}$ except last year **do**
  $X_{tr} \leftarrow \mathcal{D}$(year)
  $y_{tr} \leftarrow \mathcal{D}$(year)(class)
  remove $X_{tr}$(class)
  $\mathbf{X}_{tr} \leftarrow$ append $X_{tr}$
  $\mathbf{y}_{tr} \leftarrow$ append $y_{tr}$
**end for**
**return** $\mathbf{X}_{tr}, \mathbf{y}_{tr}$

### 3.6.3 Predict NBA Playoff Contenders in Each Season & Determine Accuracy Using Logistic Regression Classifier

Next, we used our trained model to predict NBA Playoff contenders in each NBA season and compute the accuracies of our predictions using Algorithm 3 and 4.

**Algorithm 4** Obtain accuracies of logistic regression classification using obtained maximal accuracy regularization strengths, $\mathbf{c}_{max}$

**Require:**

- $\mathbf{x}_{train}$, vector of training attributes per NBA season
- $\mathbf{y}_{train}$, vector of training targets per NBA season
- $\mathbf{A}$, 3-column matrix with empty initialization (0 rows to begin with), for storing accuracies of each classifier for all NBA seasons
- $\mathbf{a}$, 3-element vector, initially empty, for temporarily storing accuracy of 1 of 3 classifiers per NBA season
- $\mathcal{M}$, 3-element vector, initially empty, for storing different logistic regression models
- $\hat{\mathbf{y}}$, 3-element vector, initially empty, for storing prediction vectors of each classifier
- $\hat{\mathbf{e}}$, 3-element vector, initially empty for storing Eastern-Conference NBA Playoff teams, in order from highest to lowest probability, for each classifier
- $\hat{\mathbf{w}}$, 3-element vector, initially empty for storing Western-Conference NBA Playoff teams, in order from highest to lowest probability, for each classifier
- $\mathbf{t}$, vector of NBA season start years
- $\mathcal{D}$, dictionary of NBA teams per season. Key/value pair is start year of season/vector of NBA teams
- $\mathbf{c}$, 3-element vector of regularization strengths for each logistic regression classifier

Obtain accuracies of logistic regression classification (cont'd)

**for** $i$ in $\mathbf{t}$ except last year **do**
  $\mathbf{X}_{test} \leftarrow \mathcal{D}(i)$
  $\mathbf{y}_{test} \leftarrow \mathbf{X}_{test}(:, 0)$
  $\mathbf{X}_{test} \leftarrow$ scale/normalize$(\mathbf{X}_{test}(:, 1:))$
  $\mathbf{w}_{test} \leftarrow$ store classes of Western-Conference teams from $\mathbf{y}_{test}$
  $\mathbf{e}_{test} \leftarrow$ store classes of Eastern-Conference teams from $\mathbf{y}_{test}$
  **for** $c, j$ in $\mathbf{c}_{max}$, $[\ell_1, \ell_2, un]$ **do**
    $\mathcal{M}(j) \leftarrow$ (penalty $= j, C = c$)
    $\mathcal{M}(j) \leftarrow$ train using $(\mathbf{x}_{train}(i), \mathbf{y}_{train}(i))$
    $\hat{\mathbf{y}}(j) \leftarrow$ predict probability of $\mathbf{X}_{test}$ and sort in order from highest to lowest probability
    $\hat{\mathbf{e}}(j) \leftarrow$ extract 8-highest probability Eastern-Conference Playoff teams from $\hat{\mathbf{y}}$
    $\hat{\mathbf{w}}(j) \leftarrow$ extract 8-highest probability Western-Conference Playoff teams from $\hat{\mathbf{y}}$
    $\mathbf{a}(j) \leftarrow$ mean$(\hat{\mathbf{w}}(j) == \mathbf{w}_{test}$ **and** $\hat{\mathbf{e}}(j) == \mathbf{e}_{test})$
  **end for**
  $\mathbf{A}(i) \leftarrow$ appendRow$(\mathbf{a})$
**end for**
**return** $\mathbf{A}$

## 3.7 Support Vector Machine (SVM) Classification

### 3.7.1 Determination of SVM Classifier Regularization Strengths, $C$, Based on Kernels

Using [7, 17], SVM classifiers were developed with the following support kernel functions:

- Linear kernel
- Radial basis kernel
- Polynomial kernel
- Sigmoid kernel

For these kernel function-supported SVM classifiers, regularization strengths were determined similar to the logistic regression regularization strengths in Section 3.6, as demonstrated in Algorithm 5.

**Algorithm 5** Determination of SVM classifier regularization strengths, $\mathbf{c}_{max}$

**Require:**

- $\mathbf{c}$, vector of log-spaced test strengths
- $\mathbf{c}_{max}$, 4-element vector for storing regularization strength that grants greatest accuracy for $rbf$, $lin$, $poly$, and $sig$ kernel function SVM classifiers
- $\mathbf{a}$, empty 4-element vector for accuracy vectors of each SVM classifier
- $\mathbf{a}_{max}$, 4-element vector for storing maximum accuracies, each element defaulted to 0
- $\mathbf{X}$, feature matrix
- $\mathbf{y}$, classification of each sample
- $\hat{\mathbf{y}}$, empty 4-element vector, each element for storing a prediction vector
- $\mathcal{M}$, empty 4-element vector for storing SVM classification models, $lin, rbf, poly$, and $sig$ respectively

**for** c in **c do**
  **for** $i$ in [$linear$, $rbf$, $poly$, $sigmoid$] **do**
    $\mathcal{M}_i \leftarrow (\text{kernel} = i, C = c)$
    $\mathcal{M}_i \leftarrow \text{fit}\,(\mathbf{X}, \mathbf{y})$
    $\hat{\mathbf{y}}_i \leftarrow \text{predict}\,(\mathbf{X})$
    $\mathbf{a}_i \leftarrow \frac{\hat{\mathbf{y}}_i == \mathbf{y}}{N}$, where $N$ is the number of samples
    **if** $\mathbf{a}_{max_i} < \mathbf{a}_i$ **then**
      $\mathbf{a}_{max_i} \leftarrow \mathbf{a}_i$
      $\mathbf{c}_{max_i} \leftarrow c$
    **end if**
  **end for**
**end for**
**return** $\mathbf{c}_{max}$

### 3.7.2 Predict NBA Playoff Contenders in Each Season & Determine Accuracy Using SVM Classifiers

Next, we used our trained model to predict NBA Playoff contenders in each NBA season and compute the accuracies of our predictions using Algorithm 3 and 6.

**Algorithm 6** Obtain accuracies of SVM classification using obtained maximal accuracy regularization strengths, $\mathbf{c}_{max}$

**Require:**

- $\mathbf{x}_{train}$, vector of training attributes per NBA season
- $\mathbf{y}_{train}$, vector of training targets per NBA season
- $\mathbf{A}$, 4-column matrix with empty initialization (0 rows to begin with), for storing accuracies of each classifier for all NBA seasons
- $\mathbf{a}$, 4-element vector, initially empty, for temporarily storing accuracy of 1 of 4 classifiers per NBA season
- $\mathcal{M}$, 4-element vector, initially empty, for storing different SVM models
- $\hat{\mathbf{y}}$, 4-element vector, initially empty, for storing prediction vectors of each classifier
- $\hat{\mathbf{e}}$, 4-element vector, initially empty for storing Eastern-Conference NBA Playoff teams, in order from highest to lowest probability, for each classifier
- $\hat{\mathbf{w}}$, 4-element vector, initially empty for storing Western-Conference NBA Playoff teams, in order from highest to lowest probability, for each classifier
- $\mathbf{t}$, vector of NBA season start years
- $\mathcal{D}$, dictionary of NBA teams per season. Key/value pair is start year of season/vector of NBA teams

Obtain accuracies of SVM classification (cont'd)

**for** $i$ in $\mathbf{t}$ except last year **do**
  $\mathbf{X}_{test} \leftarrow \mathcal{D}(i)$
  $\mathbf{y}_{test} \leftarrow \mathbf{X}_{test}(:, 0)$
  $\mathbf{X}_{test} \leftarrow$ scale/normalize$(\mathbf{X}_{test}(:, 1:))$
  $\mathbf{w}_{test} \leftarrow$ store classes of Western-Conference teams from $\mathbf{y}_{test}$
  $\mathbf{e}_{test} \leftarrow$ store classes of Eastern-Conference teams from $\mathbf{y}_{test}$
  **for** $c, j$ in $\mathbf{c}_{max}, [linear, rbf, poly, sigmoid]$ **do**
    $\mathcal{M}(j) \leftarrow (\text{kernel} = j, C = c)$
    $\mathcal{M}(j) \leftarrow$ train using $(\mathbf{x}_{train}(i), \mathbf{y}_{train}(i))$
    $\hat{\mathbf{y}}(j) \leftarrow$ predict probability of $\mathbf{X}_{test}$ and sort in order from highest to lowest probability
    $\hat{\mathbf{e}}(j) \leftarrow$ extract 8-highest probability Eastern-Conference Playoff teams from $\hat{\mathbf{y}}$
    $\hat{\mathbf{w}}(j) \leftarrow$ extract 8-highest probability Western-Conference Playoff teams from $\hat{\mathbf{y}}$
    $\mathbf{a}(j) \leftarrow \text{mean}(\hat{\mathbf{w}}(j) == \mathbf{w}_{test}$ **and** $\hat{\mathbf{e}}(j) == \mathbf{e}_{test})$
  **end for**
  $\mathbf{A}(i) \leftarrow$ appendRow$(\mathbf{a})$
**end for**
**return** $\mathbf{A}$

## 3.8 Random Forest Classification

### 3.8.1 Determining the Number of Trees for Random Forest (RF) Classifier

Unlike the logistic regression and SVM classifiers, RF classification does not depend on regularization constants, $C$. Instead, RF classification depends on a number of decision trees, $n$. We use the [7, 17] implementations of Python RF classifiers and Algorithm 3 to determine the minimal number of trees, $n$, necessary for a high accuracy prediction. This method is explained by Algorithm 7.

**Algorithm 7** Determination maximal accuracy RF classifier tree number, $n_{max}$

**Require:**

- $\mathbf{n}$, vector of log-spaced test tree numbers, $n$
- $n_{max}$, for storing tree number that grants greatest accuracy for RF classifier
- $\mathbf{a}$, empty vector for for storing accuracy of each $n$
- $a_{max}$, for storing maximum accuracies, defaulted to 0
- $\mathbf{X}$, feature matrix
- $\mathbf{y}$, classification of each sample
- $\hat{\mathbf{y}}$, prediction vector, initially empty
- $\mathcal{M}$, random forest classifier model

**for** $n$ in $\mathbf{n}$ **do**
  $\mathcal{M} \leftarrow (\text{number of trees} = n)$
  $\mathcal{M} \leftarrow \text{fit}\,(\mathbf{X}, \mathbf{y})$
  $\hat{\mathbf{y}} \leftarrow \text{predict}\,(\mathbf{X})$
  $\mathbf{a} \leftarrow \text{append}\left(\frac{\hat{\mathbf{y}}_i == \mathbf{y}}{N}\right)$, where $N$ is the number of samples
  **if** $a_{max} < \mathbf{a}[last]$ **then**
    $a_{max} \leftarrow \mathbf{a}[last]$
    $n_{max} \leftarrow n$
  **end if**
**end for**
**return** $n_{max}$

### 3.8.2 Predict NBA Playoff Contenders in Each Season & Determine Accuracy Using RF Classifiers

Using our obtained number of trees, $n$ for maximum accuracy in our RF classifier model, we used our model to predict NBA Playoff contenders in each NBA season and compute the accuracies of our prediction with the use of Algorithms 3 and 8.

**Algorithm 8** Obtain accuracies of RF classification using obtained maximal accuracy RF tree number, $n_{max}$

**Require:**

- $\mathbf{x}_{train}$, vector of training attributes per NBA season
- $\mathbf{y}_{train}$, vector of training targets per NBA season
- $\mathbf{a}$, vector with empty initialization, for storing accuracies of classifier for all NBA seasons
- $a$, for temporarily storing accuracy of RF classifier per NBA season
- $\mathcal{M}$, for storing RF model
- $\hat{\mathbf{y}}$, prediction vector, initially empty
- $\hat{\mathbf{e}}$, vector, initially empty for storing Eastern-Conference NBA Playoff teams, in order from highest to lowest probability, for RF classifier
- $\hat{\mathbf{w}}$, vector, initially empty for storing Western-Conference NBA Playoff teams, in order from highest to lowest probability, for RF classifier
- $\mathbf{t}$, vector of NBA season start years
- $\mathcal{D}$, dictionary of NBA teams per season. Key/value pair is start year of season/vector of NBA teams

**for** $i$ in $\mathbf{t}$ except last year **do**
  $\mathbf{X}_{test} \leftarrow \mathcal{D}(i)$
  $\mathbf{y}_{test} \leftarrow \mathbf{X}_{test}(:, 0)$
  $\mathbf{X}_{test} \leftarrow$ scale/normalize$(\mathbf{X}_{test}(:, 1:))$
  $\mathbf{w}_{test} \leftarrow$ store class of Western Conference teams from $\mathbf{y}_test$
  $\mathbf{e}_{test} \leftarrow$ store class of Eastern Conference teams from $\mathbf{y}_test$
  create $\mathcal{M}$ using $n_{max}$ trees
  $\mathcal{M} \leftarrow$ train using $(\mathbf{x}_{train}(i), \mathbf{y}_{train}(i))$
  $\hat{\mathbf{y}} \leftarrow$ predict probability of $\mathbf{X}_{test}$ and sort in order from highest to lowest probability
  $\hat{\mathbf{e}} \leftarrow$ extract 8-highest probabilistic Eastern Conference Playoff teams from $\hat{\mathbf{y}}$
  $\hat{\mathbf{w}} \leftarrow$ extract 8-highest probabilistic Western Conference Playoff teams from $\hat{\mathbf{y}}$
  $a \leftarrow$ mean$(\hat{\mathbf{w}} == \mathbf{w}_{test}$ **and** $\hat{\mathbf{e}} == \mathbf{e}_{test})$
  $\mathbf{a} \leftarrow$ append$(a)$
**end for**
**return** $\mathbf{a}$

## 3.9 Decision Tree Classification

Unlike the previous classification algorithms, decision tree classification does not depend on special parameters like numbers of trees and regularization. Classification in this algorithm is done with the use of a single decision tree. Using [7, 17] implementations of decision tree classifiers we can immediately obtain the the accuracy of its predictions without any special modifications or tuning.

**Algorithm 9** Obtain accuracies of decision tree classification

**Require:**

- $\mathbf{x}_{train}$, vector of training attributes per NBA season
- $\mathbf{y}_{train}$, vector of training targets per NBA season
- $\mathbf{a}$, vector with empty initialization, for storing accuracies of classifier for all NBA seasons
- $a$, for temporarily storing accuracy of decision tree classifier per NBA season
- $\mathcal{M}$, for storing decision tree model
- $\hat{\mathbf{y}}$, prediction vector, initially empty
- $\hat{\mathbf{e}}$, vector, initially empty for storing Eastern-Conference NBA Playoff teams, in order from highest to lowest probability, for decision tree classifier
- $\hat{\mathbf{w}}$, vector, initially empty for storing Western-Conference NBA Playoff teams, in order from highest to lowest probability, for decision tree classifier
- $\mathbf{t}$, vector of NBA season start years
- $\mathcal{D}$, dictionary of NBA teams per season. Key/value pair is start year of season/vector of NBA teams

Obtain accuracies of decision tree classification (cont'd)

**for** $i$ in $\mathbf{t}$ except last year **do**
  $\mathbf{X}_{test} \leftarrow \mathcal{D}(i)$
  $\mathbf{y}_{test} \leftarrow \mathbf{X}_{test}(:, 0)$
  $\mathbf{X}_{test} \leftarrow$ scale/normalize$(\mathbf{X}_{test}(:, 1:))$
  $\mathbf{w}_{test} \leftarrow$ store class of Western Conference teams from $\mathbf{y}_test$
  $\mathbf{e}_{test} \leftarrow$ store class of Eastern Conference teams from $\mathbf{y}_test$
  $\mathcal{M} \leftarrow$ initialize decision tree
  $\mathcal{M} \leftarrow$ train using $(\mathbf{x}_{train}(i), \mathbf{y}_{train}(i))$
  $\hat{\mathbf{y}} \leftarrow$ predict probability of $\mathbf{X}_{test}$ and sort in order from highest to lowest probability
  $\hat{\mathbf{e}} \leftarrow$ extract 8-highest probabilistic Eastern Conference Playoff teams from $\hat{\mathbf{y}}$
  $\hat{\mathbf{w}} \leftarrow$ extract 8-highest probabilistic Western Conference Playoff teams from $\hat{\mathbf{y}}$
  $a \leftarrow$ mean$(\hat{\mathbf{w}} == \mathbf{w}_{test}$ **and** $\hat{\mathbf{e}} == \mathbf{e}_{test})$
  $\mathbf{a} \leftarrow$ append$(a)$
**end for**
**return** $\mathbf{a}$

# Chapter 4

# Results

## 4.1 Logistic Regression Classification

In Algorithm 2, we determined the minimal regularization strength, $C$, that grants the highest accuracy for each classifier. The results of this test yielded:

Table 4.1: Accuracy of Logistic Regression Regularization Strengths (Total-Trained & Total-Predict Models)

| **Reg.**, $C$ | $\ell_1$-**Predict Acc.**, % | $\ell_2$-**Predict Acc.**, % | **Un-norm. Predict Acc.**, % |
|---|---|---|---|
| 0.01 | 46.31% | 71.81% | 71.81% |
| 0.026 | 70.47% | 72.93% | 72.93% |
| 0.0695 | 71.14% | 73.38% | 73.38% |
| 0.18 | 73.60% | 72.93% | 72.93% |
| 0.483 | 72.71% | 72.93% | 72.93% |
| 1.27 | 73.60% | 73.15% | 73.15% |
| 3.36 | 73.15% | 73.15% | 73.15% |
| 8.86 | 73.38% | 73.38% | 73.38% |
| 23.4 | 73.38% | 73.38% | 73.38% |
| 61.6 | 73.38% | 73.38% | 73.38% |
| 162.4 | 73.38% | 73.38% | 73.38% |
| 428.1 | 73.38% | 73.38% | 73.38% |
| 1128.84 | 73.38% | 73.38% | 73.38% |
| 2976.4 | 73.38% | 73.38% | 73.38% |
| 7847.6 | 73.38% | 73.38% | 73.38% |
| 20691.4 | 73.38% | 73.38% | 73.38% |
| 54555.9 | 73.38% | 73.38% | 73.38% |
| 143844.9 | 73.38% | 73.38% | 73.38% |
| 379269 | 73.38% | 73.38% | 73.38% |
| 1000000 | 73.38% | 73.38% | 73.38% |

where $\mathbf{c}_{max}$ was: $\mathbf{c}_{max} = [c_{\ell_1}, c_{\ell_2}, c_{un-norm}]$

- $c_{\ell_1} = 0.1833$
- $c_{\ell_2} = 0.0695$
- $c_{un-norm} = 0.0695$

Using $\mathbf{c}_{max}$ and Algorithm 4 the following results were obtained:

Table 4.2: Accuracy of Progressively-Trained Logistic Regression Model Over Each NBA Season

| **Train Years** | **Test Season** | $\ell_1$-**Acc.**, % | $\ell_2$-**Acc.**, % | **Un-Norm. Acc.**, % |
|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 87.5% | 81.25% | 81.25% |
| 2001-2003 | 2003-2004 | 75% | 75% | 75% |
| 2001-2004 | 2004-2005 | 81.25% | 75% | 75% |
| 2001-2005 | 2005-2006 | 68.75% | 68.75% | 68.75% |
| 2001-2006 | 2006-2007 | 75% | 81.25% | 81.25% |
| 2001-2007 | 2007-2008 | 75% | 68.75% | 68.75% |
| 2001-2008 | 2008-2009 | 75% | 75% | 75% |
| 2001-2009 | 2009-2010 | 68.75% | 68.75% | 68.75% |
| 2001-2010 | 2010-2011 | 75% | 75% | 75% |
| 2001-2011 | 2011-2012 | 81.25% | 81.25% | 81.25% |
| 2001-2012 | 2012-2013 | 81.25% | 81.25% | 81.25% |
| 2001-2013 | 2013-2014 | 75% | 81.25% | 81.25% |
| 2001-2014 | 2014-2015 | 87.5% | 81.25% | 81.25% |
| 2001-2015 | 2015-2016 | 62.5% | 62.5% | 62.5% |

Table 4.3: Training Time of Progressively-Trained Logistic Regression Model Over Each NBA Season

| **Train Years** | **Test Season** | $\ell_1$ ($\mu$sec.) | $\ell_2$ ($\mu$sec.) | **un-norm.** ($\mu$sec.) |
|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 523.09 | 298.98 | 287.06 |
| 2001-2003 | 2003-2004 | 529.05 | 307.08 | 281.81 |
| 2001-2004 | 2004-2005 | 440.12 | 324.01 | 309.94 |
| 2001-2005 | 2005-2006 | 528.81 | 354.05 | 334.74 |
| 2001-2006 | 2006-2007 | 784.16 | 928.88 | 728.13 |
| 2001-2007 | 2007-2008 | 684.98 | 685.21 | 694.99 |
| 2001-2008 | 2008-2009 | 904.80 | 839.95 | 853.06 |
| 2001-2009 | 2009-2010 | 862.36 | 970.84 | 1075.98 |
| 2001-2010 | 2010-2011 | 938.89 | 934.84 | 849.01 |
| 2001-2011 | 2011-2012 | 1012.80 | 1080.99 | 1102.92 |
| 2001-2012 | 2012-2013 | 1201.15 | 1019.24 | 1013.99 |
| 2001-2013 | 2013-2014 | 1766.92 | 1357.79 | 1486.06 |
| 2001-2014 | 2014-2015 | 1635.07 | 1430.99 | 1159.91 |
| 2001-2015 | 2015-2016 | 1309.87 | 1201.15 | 1234.05 |

Table 4.4: Prediction Time of Progressively-Trained Logistic Regression Model Over Each NBA Season

| **Train Years** | **Test Season** | $\ell_1$ ($\mu$sec.) | $\ell_2$ ($\mu$sec.) | **un-norm.** ($\mu$sec.) |
|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 84.88 | 119.92 | 62.94 |
| 2001-2003 | 2003-2004 | 81.06 | 63.90 | 114.92 |
| 2001-2004 | 2004-2005 | 77.25 | 64.85 | 63.90 |
| 2001-2005 | 2005-2006 | 79.15 | 65.80 | 64.13 |
| 2001-2006 | 2006-2007 | 225 .07 | 146.87 | 116.11 |
| 2001-2007 | 2007-2008 | 117.06 | 123.98 | 128.03 |
| 2001-2008 | 2008-2009 | 135.90 | 140.91 | 98.71 |
| 2001-2009 | 2009-2010 | 139.71 | 164.75 | 152.11 |
| 2001-2010 | 2010-2011 | 143.77 | 110.15 | 172.14 |
| 2001-2011 | 2011-2012 | 145.91 | 140.91 | 279.90 |
| 2001-2012 | 2012-2013 | 133.75 | 128.98 | 128.98 |
| 2001-2013 | 2013-2014 | 172.14 | 182.15 | 165.94 |
| 2001-2014 | 2014-2015 | 216.96 | 134.71 | 123.98 |
| 2001-2015 | 2015-2016 | 133.75 | 242.23 | 150.20 |

## 4.2 Support Vector Machine (SVM) Classification

In Algorithm 5, we determined the minimal regularization strength, $C$, that grants the highest accuracy for each classifier. The results of this test yielded:

Table 4.5: Accuracy of SVM Regularization Strengths (Total-Trained & Total-Predict Models)

| **Reg.** $C$ | *linear* **Acc.**, % | *rbf* **Acc.**, % | *poly* **Acc.**, % | *sigmoid* **Acc.**, % |
|---|---|---|---|---|
| 0.01 | 71.81% | 53.69% | 57.27% | 63.31% |
| 0.1 | 73.83% | 72.71% | 66.22% | 70.92% |
| 1 | 73.38% | 77.63% | 76.96% | 65.77% |
| 10 | 73.60% | 86.35% | 80.31% | 66.00% |
| 100 | 73.60% | 95.08% | 87.70% | 66.22% |
| 1000 | 73.38% | 100.00% | 91.05% | 66.22% |
| 10000 | 73.38% | 100.00% | 93.74% | 66.22% |
| 100000 | 74.27% | 100.00% | 95.08% | 66.22% |

where $\mathbf{c}_{max}$ was: $\mathbf{c}_{max} = [c_{linear}, c_{rbf}, c_{poly}, c_{sig}]$

- $c_{linear} = 10^5$
- $c_{rbf} = 10^3$
- $c_{poly} = 10^5$
- $c_{sig} = 10^{-1}$

Using $\mathbf{c}_{max}$ and Algorithm 6 the following results were obtained:

Table 4.6: Accuracy of Progressively-Trained SVM Model Over Each NBA Season

| **Train Years** | **Test Season** | *linear*, % | *rbf*, % | *poly*, % | *sigmoid*, % |
|---|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 81.25% | 87.50% | 56.25% | 62.50% |
| 2001-2003 | 2003-2004 | 81.25% | 75.00% | 75.00% | 62.50% |
| 2001-2004 | 2004-2005 | 75.00% | 62.50% | 75.00% | 68.75% |
| 2001-2005 | 2005-2006 | 75.00% | 68.75% | 62.50% | 75.00% |
| 2001-2006 | 2006-2007 | 75.00% | 62.50% | 62.50% | 75.00% |
| 2001-2007 | 2007-2008 | 62.50% | 75.00% | 62.50% | 75.00% |
| 2001-2008 | 2008-2009 | 68.75% | 62.50% | 62.50% | 75.00% |
| 2001-2009 | 2009-2010 | 81.25% | 62.50% | 68.75% | 68.75% |
| 2001-2010 | 2010-2011 | 62.50% | 56.25% | 50.00% | 75.00% |
| 2001-2011 | 2011-2012 | 75.00% | 75.00% | 56.25% | 81.25% |
| 2001-2012 | 2012-2013 | 68.75% | 62.50% | 62.50% | 75.00% |
| 2001-2013 | 2013-2014 | 68.75% | 50.00% | 43.75% | 68.75% |
| 2001-2014 | 2014-2015 | 75.00% | 62.50% | 68.75% | 81.25% |
| 2001-2015 | 2015-2016 | 62.50% | 56.25% | 50.00% | 75.00% |

Table 4.7: Training Time of Progressively-Trained SVM Model Over Each NBA Season

| **Train Years** | **Test Season** | *linear* (msec.) | *rbf* (msec.) | *poly* (msec.) | *sigmoid* (msec.) |
|---|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 4.190 | 0.362 | 0.292 | 0.281 |
| 2001-2003 | 2003-2004 | 15703.477 | 0.462 | 0.325 | 0.345 |
| 2001-2004 | 2004-2005 | 10724.078 | 1.120 | 0.544 | 0.527 |
| 2001-2005 | 2005-2006 | 20753.272 | 1.031 | 1.105 | 0.740 |
| 2001-2006 | 2006-2007 | 22375.329 | 2.715 | 4.632 | 1.114 |
| 2001-2007 | 2007-2008 | 51523.127 | 3.055 | 4.590 | 1.209 |
| 2001-2008 | 2008-2009 | 41662.285 | 5.967 | 12.482 | 1.695 |
| 2001-2009 | 2009-2010 | 45354.074 | 8.692 | 28.115 | 2.365 |
| 2001-2010 | 2010-2011 | 113223.861 | 10.046 | 43.595 | 2.338 |
| 2001-2011 | 2011-2012 | 95059.337 | 15.023 | 155.255 | 3.070 |
| 2001-2012 | 2012-2013 | 118900.987 | 22.739 | 879.538 | 3.593 |
| 2001-2013 | 2013-2014 | 118284.245 | 26.140 | 1585.892 | 5.030 |
| 2001-2014 | 2014-2015 | 101884.206 | 31.004 | 8849.115 | 5.510 |
| 2001-2015 | 2015-2016 | 105513.456 | 36.726 | 15585.451 | 5.559 |

Table 4.8: Prediction Time of Progressively-Trained SVM Model Over Each NBA Season

| **Train Years** | **Test Season** | *linear* (msec.) | *rbf* (msec.) | *poly* (msec.) | *sigmoid* (msec.) |
|---|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 0.129 | 0.093 | 0.086 | 0.115 |
| 2001-2003 | 2003-2004 | 0.150 | 0.097 | 0.082 | 0.112 |
| 2001-2004 | 2004-2005 | 0.148 | 0.148 | 0.096 | 0.145 |
| 2001-2005 | 2005-2006 | 0.207 | 0.125 | 0.138 | 0.178 |
| 2001-2006 | 2006-2007 | 0.174 | 0.267 | 0.156 | 0.199 |
| 2001-2007 | 2007-2008 | 0.159 | 0.140 | 0.109 | 0.184 |
| 2001-2008 | 2008-2009 | 0.174 | 0.165 | 0.163 | 0.247 |
| 2001-2009 | 2009-2010 | 0.169 | 0.270 | 0.220 | 0.354 |
| 2001-2010 | 2010-2011 | 0.190 | 0.221 | 0.183 | 0.236 |
| 2001-2011 | 2011-2012 | 0.198 | 0.221 | 0.209 | 0.394 |
| 2001-2012 | 2012-2013 | 0.358 | 0.489 | 0.236 | 0.294 |
| 2001-2013 | 2013-2014 | 0.204 | 0.273 | 0.211 | 0.390 |
| 2001-2014 | 2014-2015 | 0.210 | 0.261 | 0.190 | 0.475 |
| 2001-2015 | 2015-2016 | 0.212 | 0.292 | 0.203 | 0.351 |

## 4.3 Random Forest (RF) Classification

In Algorithm 7, we determined the minimal number of trees, $n$, that grants the highest accuracy for each classifier. The results of this test yielded:

Table 4.9: Accuracy of RF Tree Numbers, $n$, (Total-Trained & Total-Predict Models)

| **Number of Estimators (Trees),** $n$ | **Accuracy**, % |
|---|---|
| 10 | 98.66% |
| 12 | 99.33% |
| 16 | 99.78% |
| 20 | 99.78% |
| 26 | 100.00% |
| 33 | 100.00% |
| 42 | 100.00% |
| 54 | 100.00% |
| 69 | 100.00% |
| 88 | 100.00% |
| 112 | 100.00% |
| 143 | 100.00% |
| 183 | 100.00% |
| 233 | 100.00% |
| 297 | 100.00% |
| 379 | 100.00% |
| 483 | 100.00% |
| 615 | 100.00% |
| 784 | 100.00% |
| 1000 | 100.00% |

where $n_{max}$ was: $n_{max} = 26$ Using $n_{max}$ and Algorithm 8 the following results were obtained as:

Table 4.10: Accuracy, Training Time, & Prediction Time of Progressively-Trained RF Model Over Each NBA Season

| **Train Years** | **Test Season** | **RF Acc.**, % | **Train.** (msec.) | **Predict.** (msec.) |
|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 87.5% | 25.78 | 2.05 |
| 2001-2003 | 2003-2004 | 75.0% | 25.00 | 2.73 |
| 2001-2004 | 2004-2005 | 68.8% | 33.90 | 3.26 |
| 2001-2005 | 2005-2006 | 62.5% | 42.71 | 2.22 |
| 2001-2006 | 2006-2007 | 75.0% | 33.76 | 1.56 |
| 2001-2007 | 2007-2008 | 75.0% | 34.99 | 2.14 |
| 2001-2008 | 2008-2009 | 68.8% | 26.17 | 1.40 |
| 2001-2009 | 2009-2010 | 75.0% | 25.30 | 1.66 |
| 2001-2010 | 2010-2011 | 75.0% | 25.59 | 1.48 |
| 2001-2011 | 2011-2012 | 81.3% | 26.31 | 1.64 |
| 2001-2012 | 2012-2013 | 81.3% | 27.09 | 1.57 |
| 2001-2013 | 2013-2014 | 62.5% | 27.06 | 1.41 |
| 2001-2014 | 2014-2015 | 87.5% | 28.56 | 1.68 |
| 2001-2015 | 2015-2016 | 68.8% | 32.78 | 1.70 |

## 4.4 Decision Tree (DT) Classification

In Algorithm 9, the accuracy of the single DT model and the following results yielded:

Table 4.11: Accuracy, Training Time, & Prediction Time of Progressively-Trained DT Model Over Each NBA Season

| **Train Years** | **Test Season** | **DT Acc.**, % | **Train. Time** (msec.) | **Pred. Time** (msec.) |
|---|---|---|---|---|
| 2001-2002 | 2002-2003 | 87.5 | 0.55 | 0.12 |
| 2001-2003 | 2003-2004 | 75.0 | 0.58 | 0.09 |
| 2001-2004 | 2004-2005 | 75.0 | 0.52 | 0.09 |
| 2001-2005 | 2005-2006 | 62.5 | 0.7 | 0.09 |
| 2001-2006 | 2006-2007 | 81.25 | 0.93 | 0.09 |
| 2001-2007 | 2007-2008 | 81.25 | 1.39 | 0.16 |
| 2001-2008 | 2008-2009 | 68.75 | 1.41 | 0.09 |
| 2001-2009 | 2009-2010 | 56.25 | 1.61 | 0.11 |
| 2001-2010 | 2010-2011 | 56.25 | 1.53 | 0.09 |
| 2001-2011 | 2011-2012 | 68.75 | 1.98 | 0.14 |
| 2001-2012 | 2012-2013 | 68.75 | 2.23 | 0.1 |
| 2001-2013 | 2013-2014 | 62.5 | 2.61 | 0.14 |
| 2001-2014 | 2014-2015 | 62.5 | 2.61 | 0.11 |
| 2001-2015 | 2015-2016 | 50.0 | 2.9 | 0.1 |

# Chapter 5

# Conclusion

Tabulating the results of our analyses, we obtained the following chart of classifier accuracies for each NBA season:

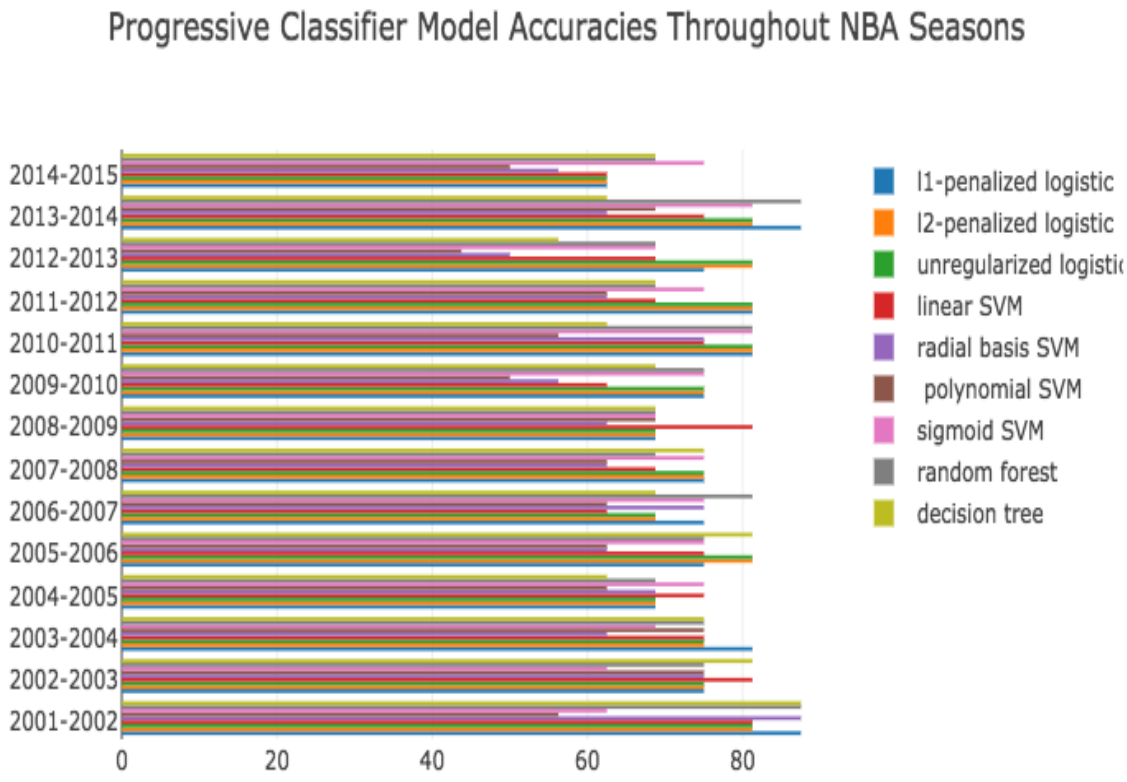


Figure 5.1: Accuracy of all classifier models throughout all NBA seasons.

In Appendix B, the results were plotted by prediction season for visibility. The mean and variance of the classifier accuracies were determined and sorted (mean: highest to lowest, variance: lowest to highest), and tabulated such that:

Table 5.1: Mean, $\mu$, ranking of classifier accuracies (greatest to least)

| **Classifier** |
|---|
| $\ell_1$-penalized logistic |
| unregularized logistic |
| $\ell_2$-penalized logistic |
| random forest |
| sigmoid SVM |
| linear SVM |
| decision tree |
| radial-basis SVM |
| polynomial SVM |

Table 5.2: Variance, $\sigma^2$, ranking of classifier accuracies (least to greatest)

| **Classifier** |
|---|
| sigmoid SVM |
| $\ell_2$-penalized logistic |
| unregularized logistic |
| linear SVM |
| random forest |
| $\ell_1$-penalized logistic |
| decision tree |
| polynomial SVM |
| radial-basis SVM |

The classification algorithm with the highest prediction accuracy with training data from $2001 - 2015$ was the radial-basis SVM classifier.

This experimental thesis was very worthy as it uncovered the many obstacles involved in obtaining, organizing, and cleaning data before processing and training. The goal of this thesis was to determine the optimal supervised learning algorithm for determining NBA Playoff contention. Based on Tables 5.1 and 5.2 a variety of choices can be made when deciding a preferred supervised learning method. Our model construction and analysis shows that optimal model selection depends on the intent of data training. If low-variance was the priority, the optimal classification algorithm to choose would be the $\ell_2$-penalized logistic classifier because of its low variance, medial mean, and extremely fast training time. If a high-mean was the priority, the optimal classification algorithm to choose would be the

un-regularized logistic classifier because of its high mean, medial variance, and extremely fast training time.

The highest achieved accuracy was a 4-way tie at 87.5% for the following classification algorithms:

- random forest
- $\ell_1$-penalized logistic
- decision tree
- radial-basis SVM

The results of this experimental thesis were encouraging. The models fitted in this project were proven to be effective, accurate, and for the most part, not very time consuming. This time-attribute was most practical in predicting outcomes because of real-world computing limitations and the immediacy of predicting outcomes in sports today. In any means, this project can be considered as a successful exploration of machine learning and data mining techniques.

## 5.1 Future Directions

This thesis investigated research at the intersection of sports analytics and supervised machine learning. The approach we proposed both produce and consume sports analytical data, with the goal of improving the accuracy of automatic predictions in sporting event outcomes. Our work opens up new directions for future research on the automatic prediction of team-based event outcomes. Some of these team-based events include:

- Other sports
  - Soccer
  - Football
  - Baseball
  - etc.
- Health
  - healthy cells vs. carcinogenic cells in ontogenetic organisms
  - natural selection
  - virus vs. antibodies

- Warfare
- Board games
- Economics

A further direction in regards to the specific work of this thesis could include the use of these same algorithms to predict which of the Playoff teams would become their respective conference champions and eventually NBA champions.

# Bibliography


[1] Anaconda Software Distribution. https://continuum.io.

[2] Doug's NBA & MLB Stats Home Page. https://dougstats.com. Last accessed December 22, 2016.

[3] Nba Stats — All Time Leaders. http://stats.nba.com/leaders/alltime/#!?SeasonType=Regular%20Season&StatCategory=FGM.

[4] Rhett Allain. Plotly Technologies Inc. Collaborative Data Science. https://plot.ly, 2015. Last accessed April 17, 2017.

[5] D. Anderson. *The Story of Basketball*. William Morrow, 1988.

[6] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is "Nearest Neighbor" Meaningful? In *In Int. Conf. on Database Theory*, pages 217–235, 1999.

[7] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API Design for Machine Learning Software: Experiences From the Scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[8] P.W.D. Charles. Project title. https://github.com/charlespwd/project-title, 2013.

[9] M. Christopher. *Great Moments in Basketball History*. Little, Brown Books for Young Readers, 2009.

[10] Pedro Domingos. A Few Useful Things to Know About Machine Learning. http://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf.

[11] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer, 6 edition.

[12] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python. http://www.scipy.org/, 2001–. [Online; accessed ¡today¿].

[13] National Basketball Association (NBA). NBA History - Career Points Leaders. http://www.espn.com/nba/history/leaders.

[14] National Basketball Association (NBA). Wilt Scores 100! http://www.nba.com/history/wilt100_moments.html.

[15] Land of Basketball. NBA All-Time Points Leaders: Career Per Game Average in the Regular Season. http://www.landofbasketball.com/all_time_leaders/points_per_game_career_season.htm.

[16] D. Oliver. *Basketball on Paper: Rules and Tools for Performance Analysis*. Potomac Books, Inc., 2004.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[18] L. E. Peterson. K-Nearest Neighbor. *Scholarpedia*, 4(2):1883, 2009. revision #136646.

[19] G. Rossum. Python Reference Manual. http://www.ncstrl.org:8900/ncstrl/servlet/search?formname=detail&id=oai%3Ancstrlh%3Aercim_cwi%3Aercim.cwi%2F%2FCS-R9525, 1995.

[20] Robert Schapire. Lecture 1, Feb 2008. Computer Science 511: Theoretical Machine Learning Lecture 1 PDF.

[21] Skipper Seabold and Josef Perktold. Statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010.

[22] W. Tubbs. How is Machine Learning Changing Sport? https://channels.theinnovationenterprise.com/articles/games-by-numbers-machine-learning-is-changing-sport, Sep 2016.

[23] Stfan van der Walt, S. Chris Colbert, and Gal Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.

# Appendix A

# Raw Data Plots

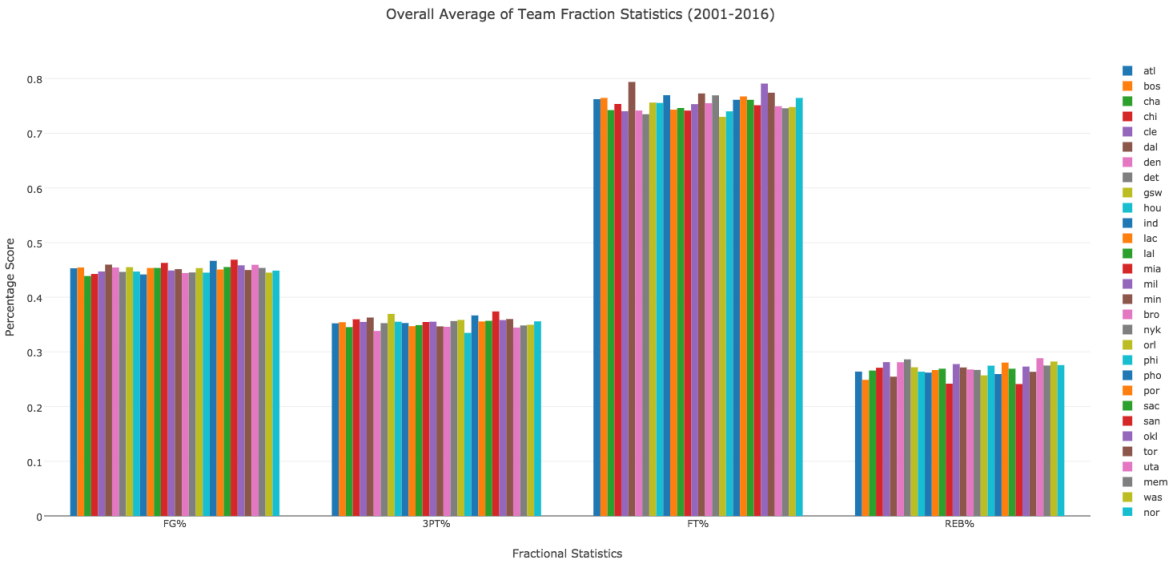


Figure A.1: Raw average of team fractional stats from NBA seasons from 2001-2016

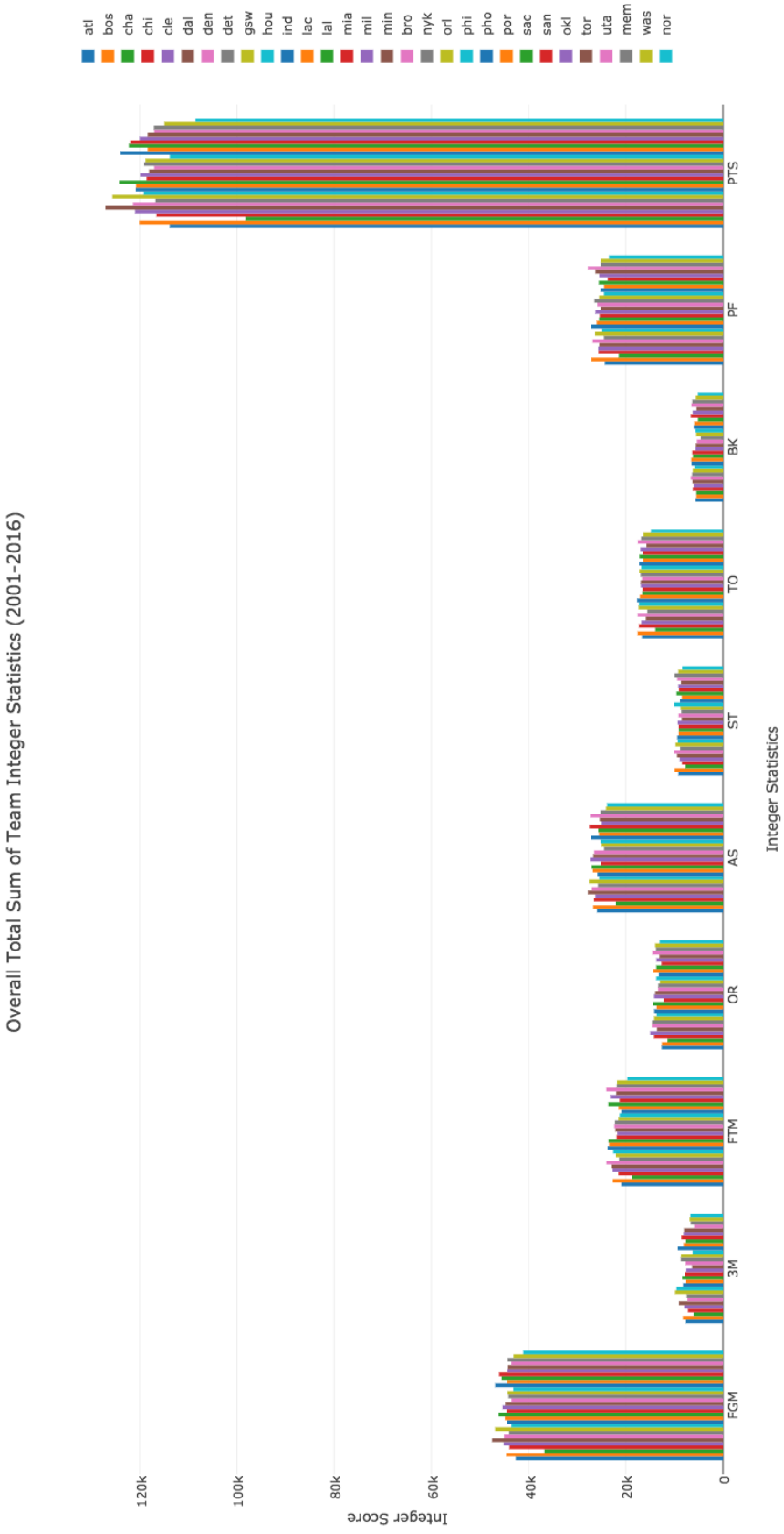


Figure A.2: Raw sum of team integer stats from NBA seasons from 2001-2016

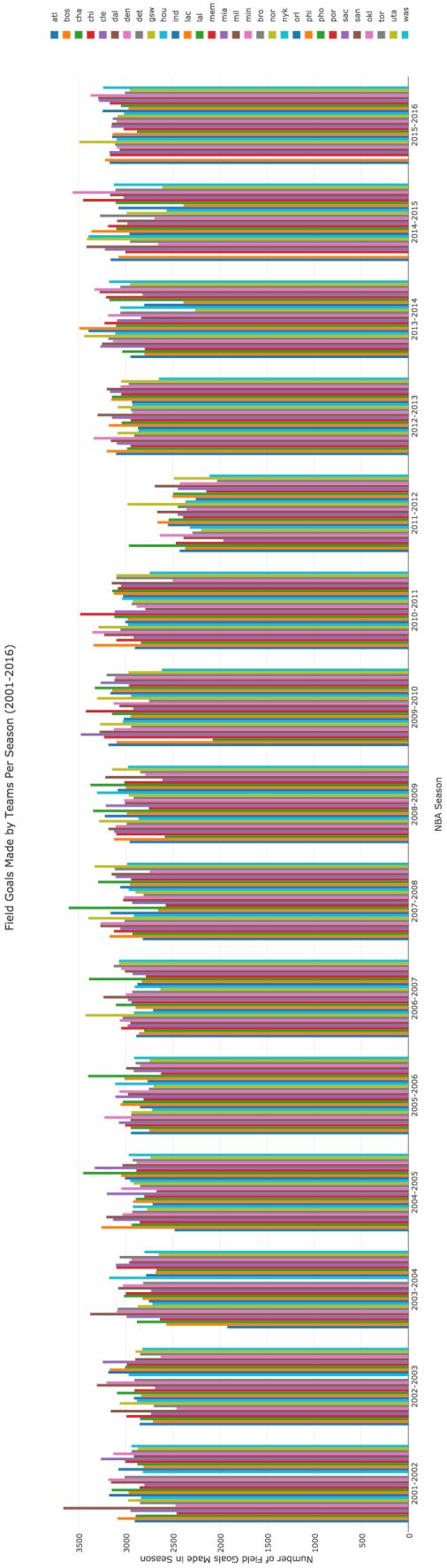


Figure A.3: Fields goals made per season by individual teams

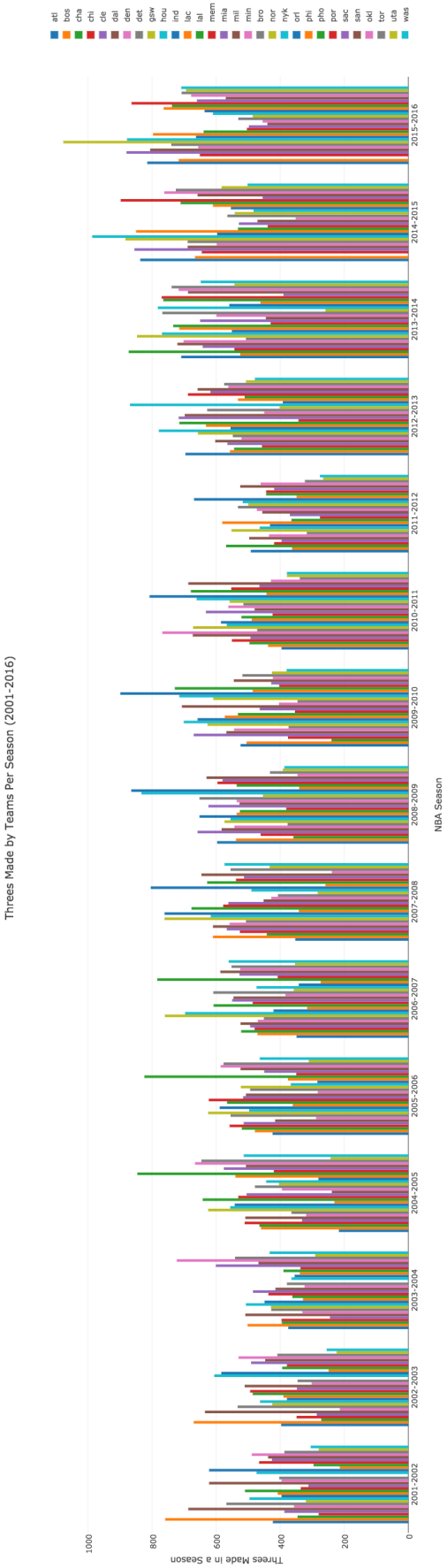


Figure A.4: Three point shots made per season by individual teams

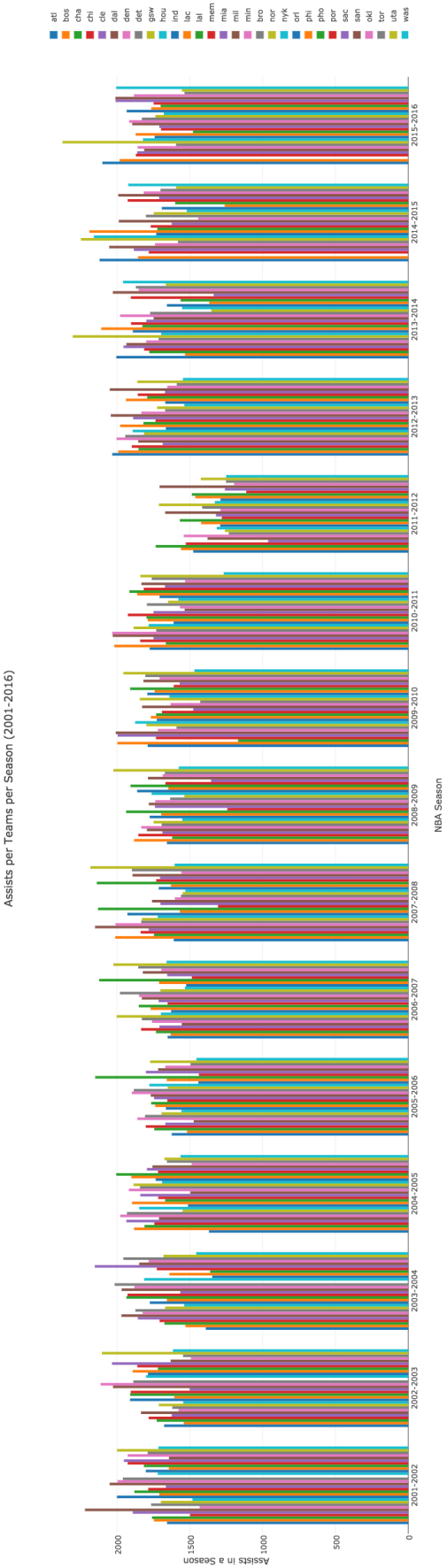


Figure A.5: Total assists made per season by individual teams

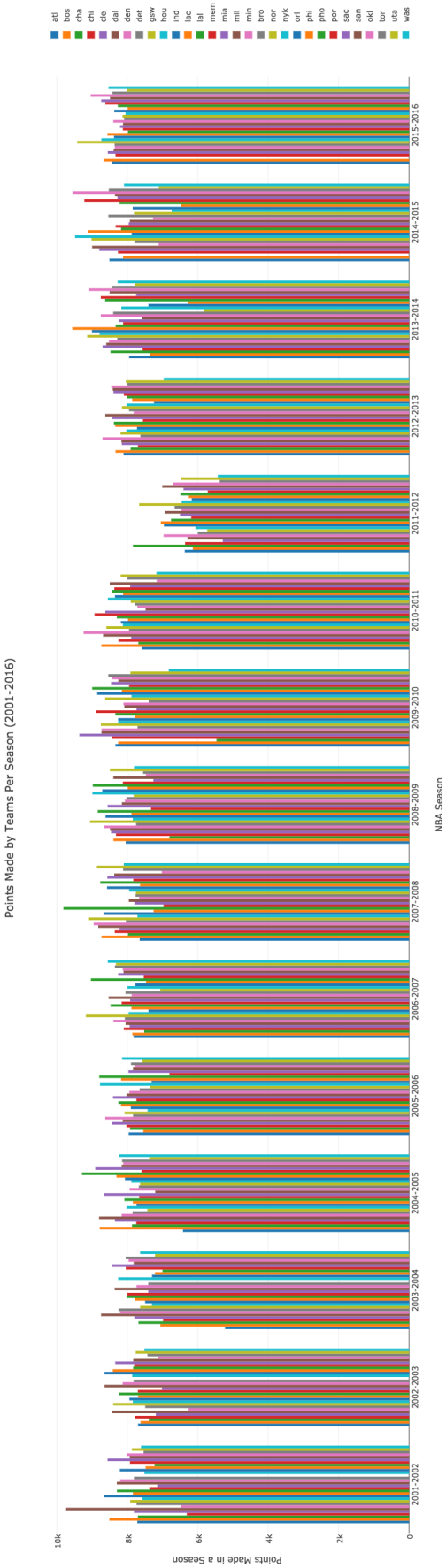


Figure A.6: Total points made per season by individual teams

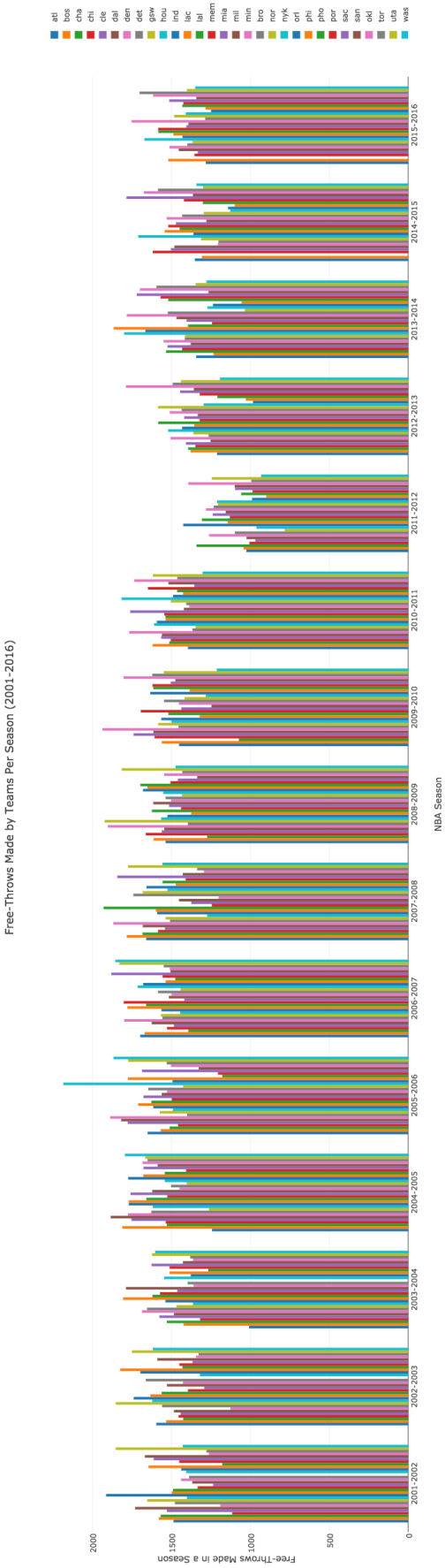


Figure A.7: Free throws made per season by individual teams

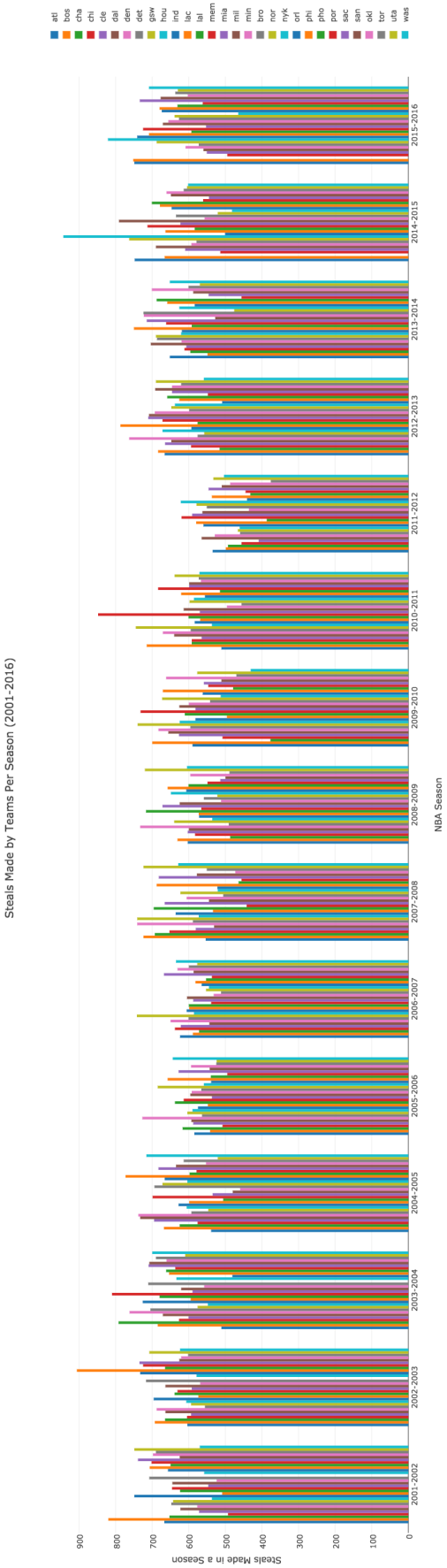


Figure A.8: Steals made per season by individual teams

# Appendix B

# All Classifier Accuracies per Season

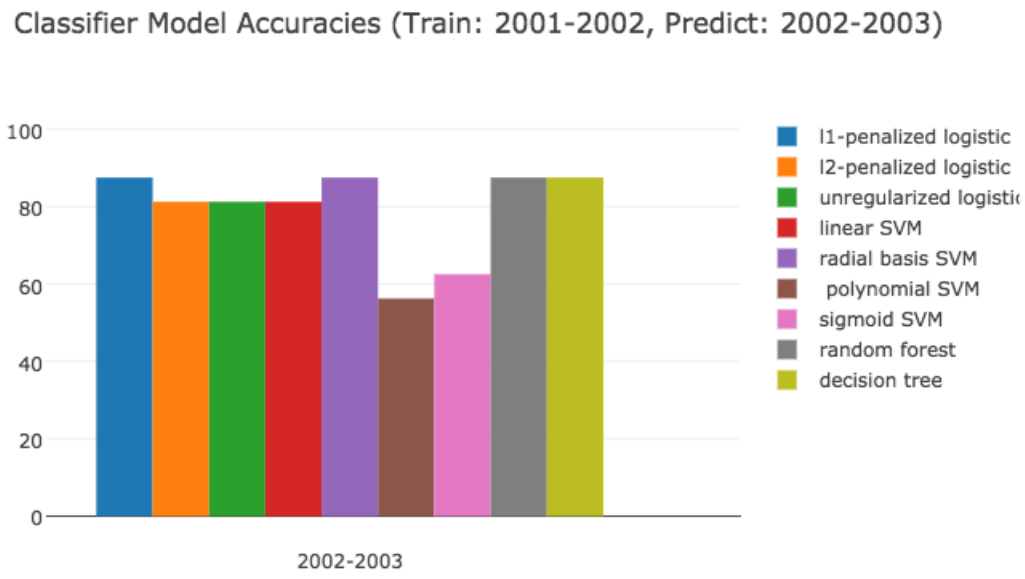


Figure B.1: Accuracies of models trained with $2001 - 2002$ data and predict $2002 - 2003$ outcome.

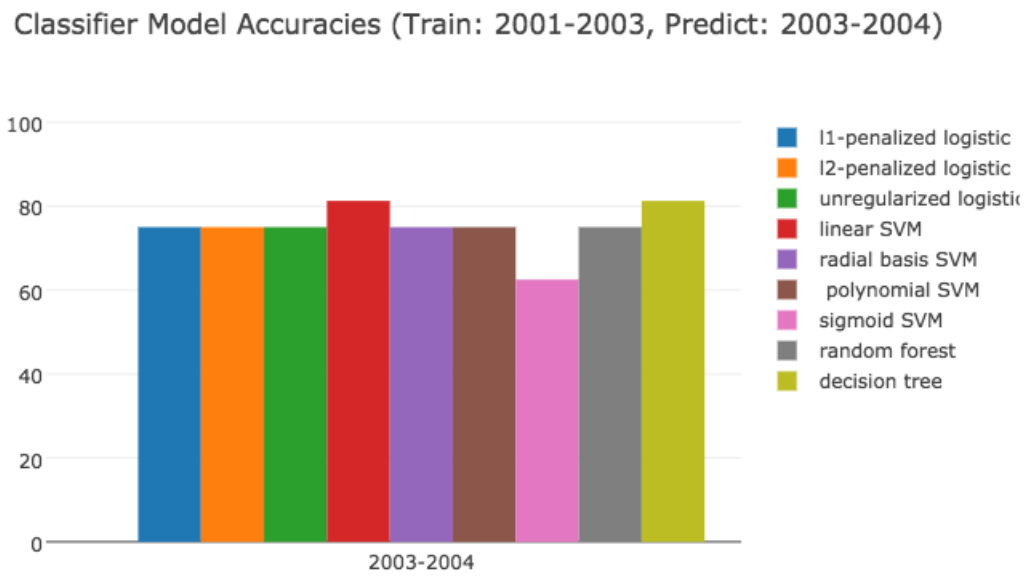


Figure B.2: Accuracies of models trained with $2001 - 2003$ data and predict $2003 - 2004$ outcome.

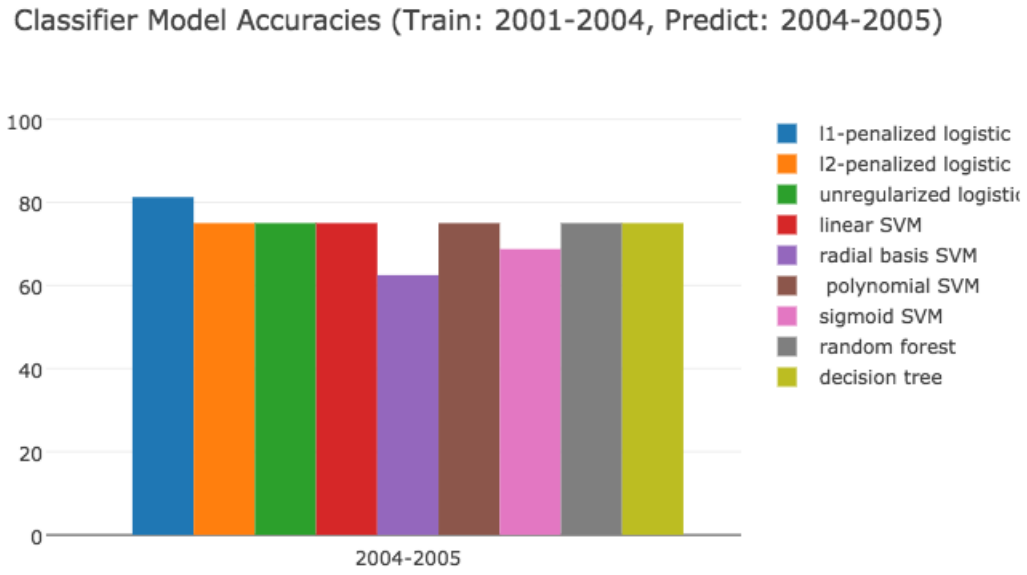


Figure B.3: Accuracies of models trained with $2001 - 2004$ data and predict $2004 - 2005$ outcome.

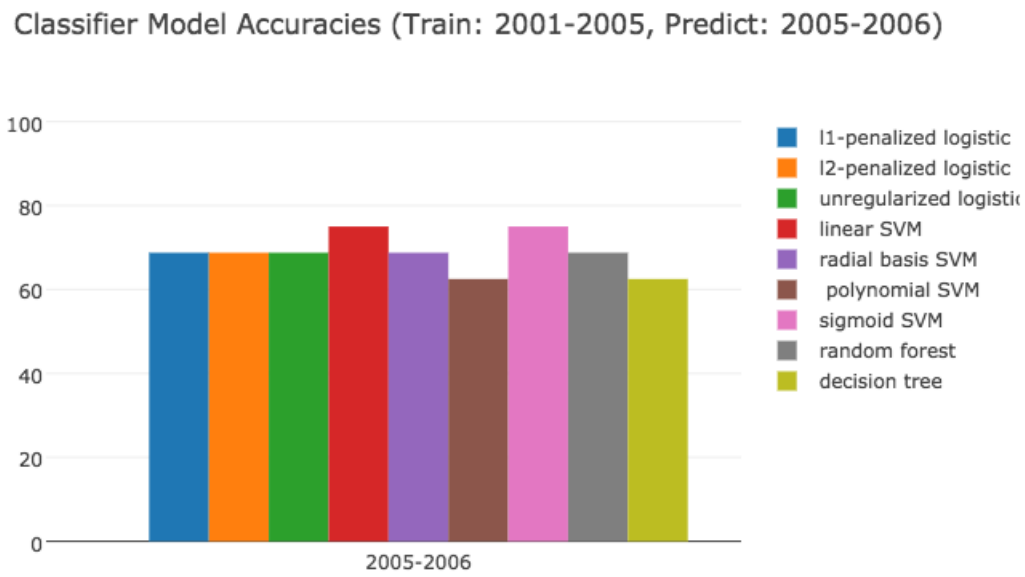


Figure B.4: Accuracies of models trained with $2001 - 2005$ data and predict $2005 - 2006$ outcome.

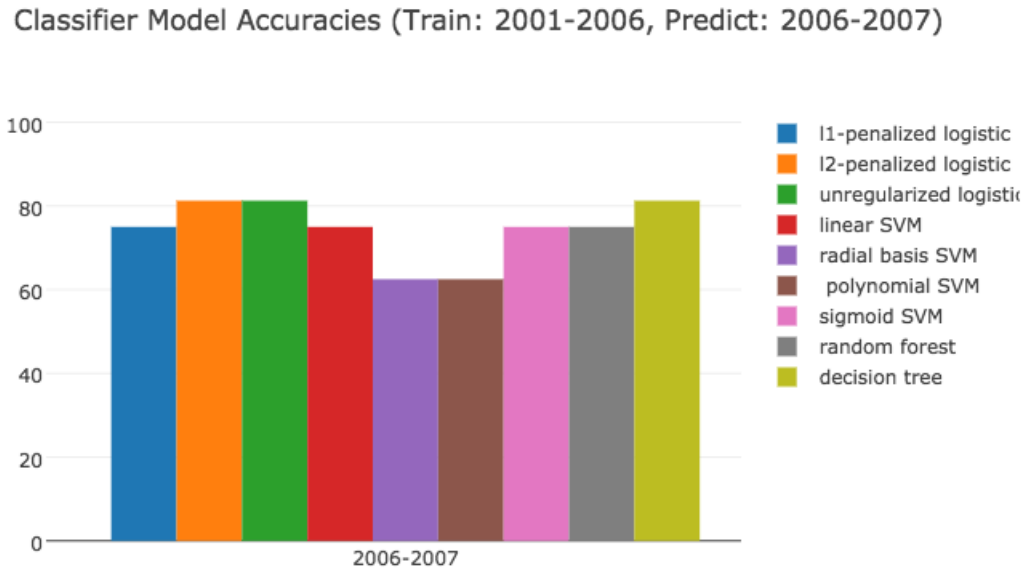


Figure B.5: Accuracies of models trained with $2001 - 2006$ data and predict $2006 - 2007$ outcome.

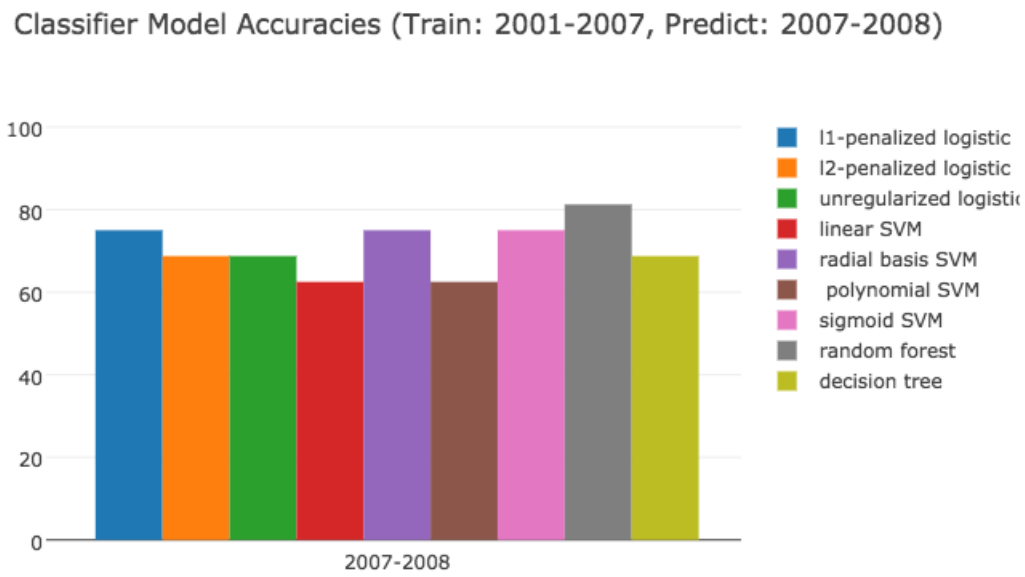


Figure B.6: Accuracies of models trained with 2001 – 2007 data and predict 2007 – 2008 outcome.

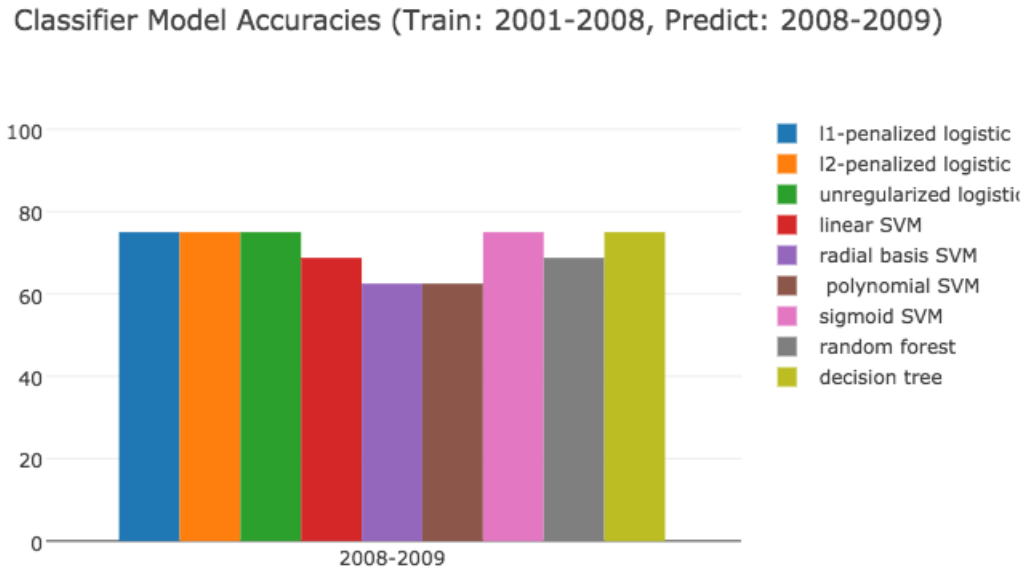


Figure B.7: Accuracies of models trained with 2001 – 2008 data and predict 2008 – 2009 outcome.

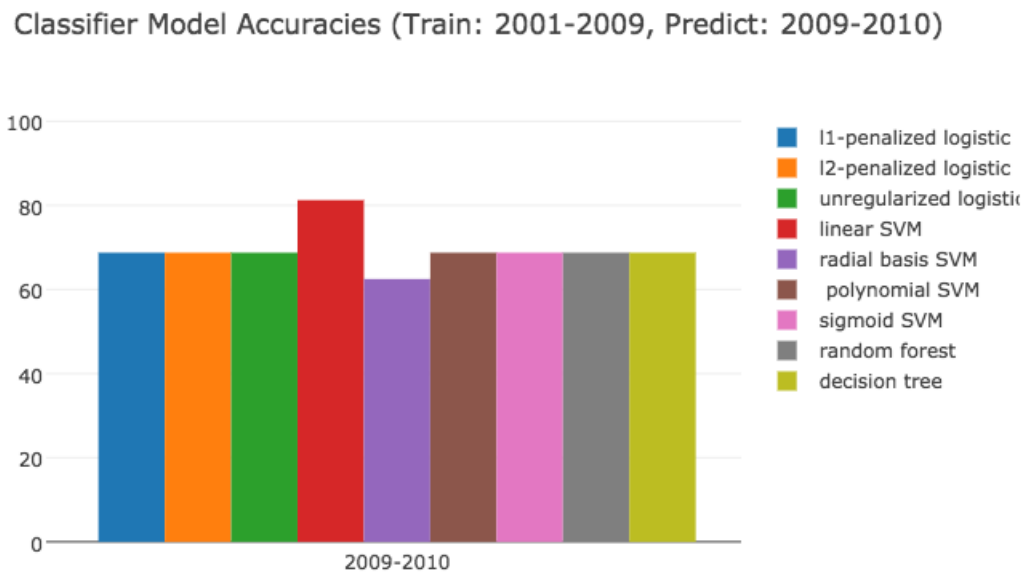


Figure B.8: Accuracies of models trained with $2001 - 2009$ data and predict $2009 - 2010$ outcome.

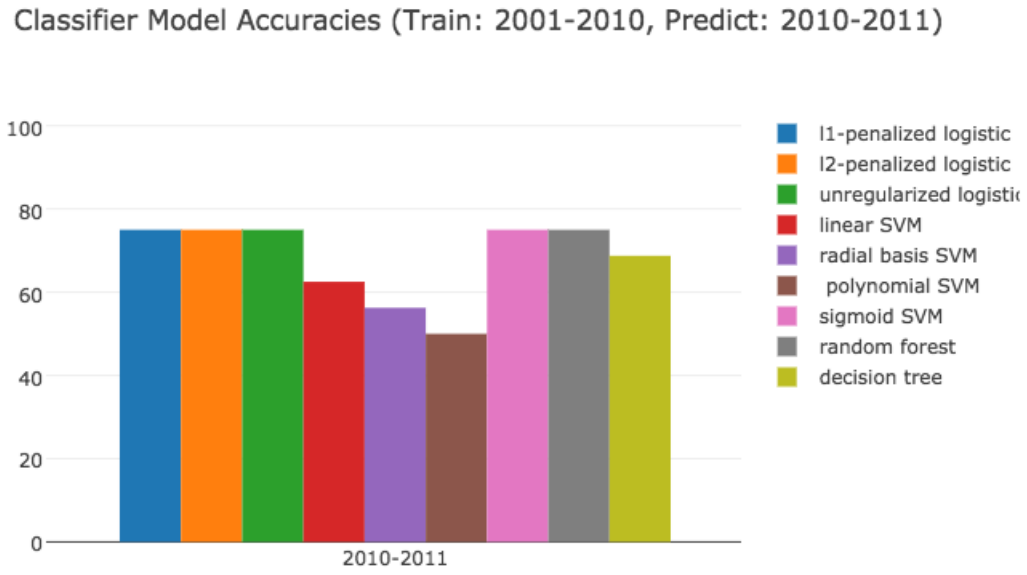


Figure B.9: Accuracies of models trained with $2001 - 2010$ data and predict $2010 - 2011$ outcome.

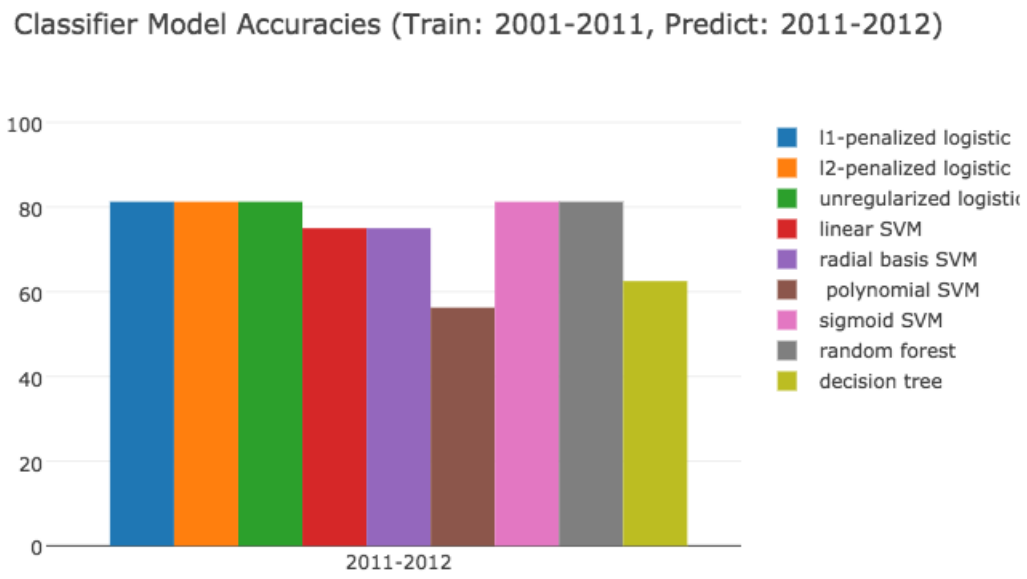


Figure B.10: Accuracies of models trained with $2001 - 2011$ data and predict $2011 - 2012$ outcome.

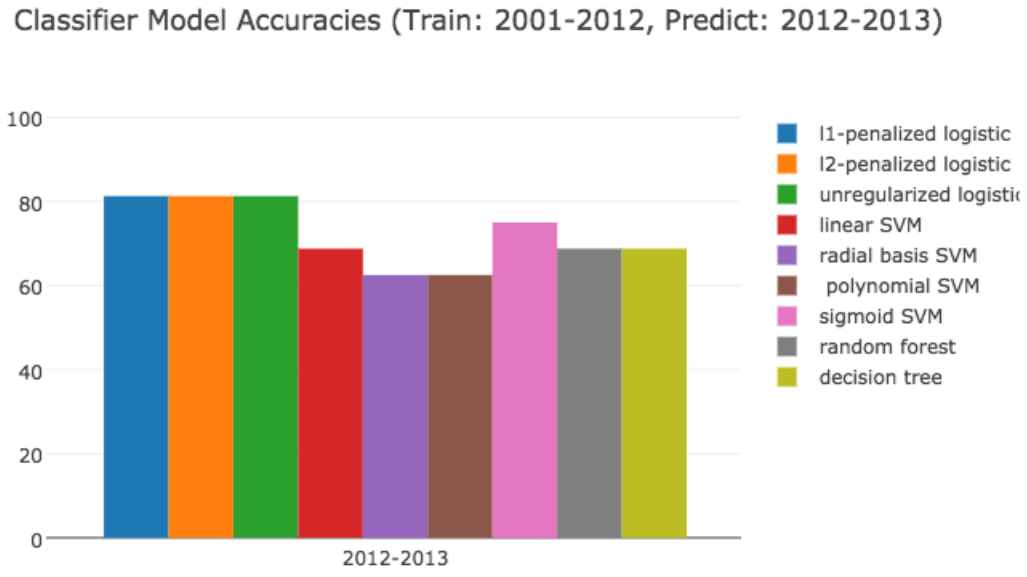


Figure B.11: Accuracies of models trained with $2001 - 2012$ data and predict $2012 - 2013$ outcome.

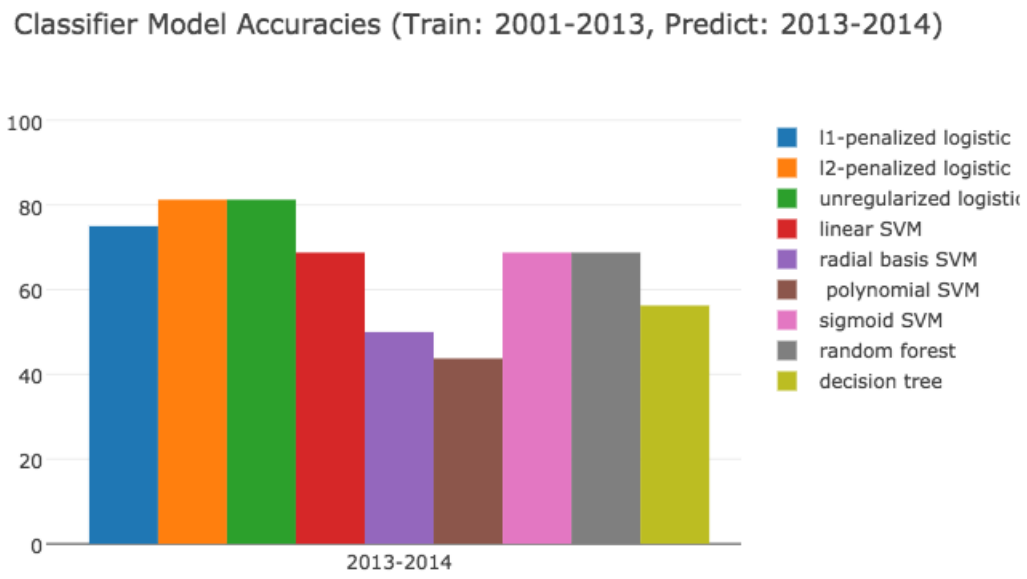


Figure B.12: Accuracies of models trained with $2001 - 2013$ data and predict $2013 - 2014$ outcome.

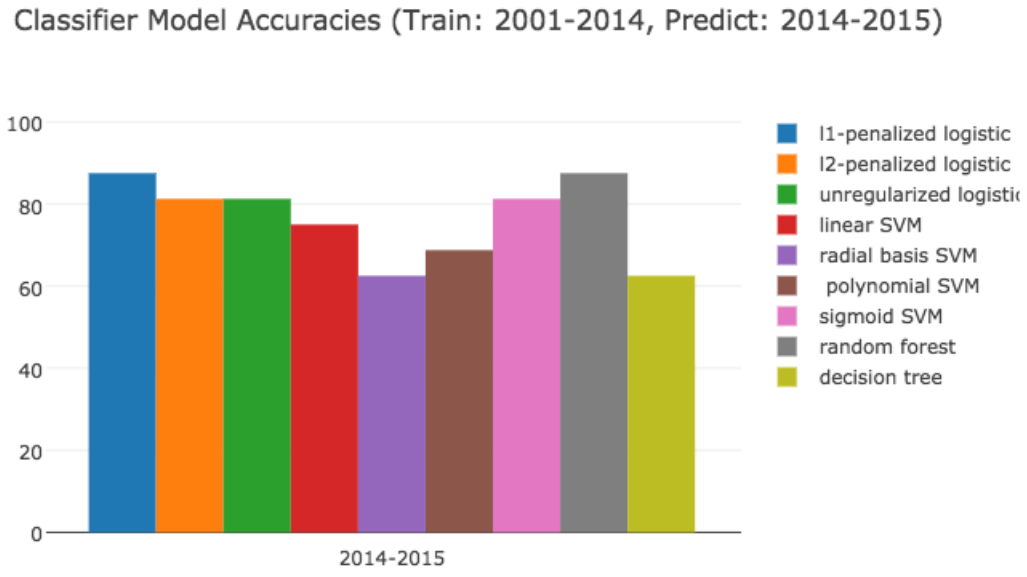


Figure B.13: Accuracies of models trained with $2001 - 2014$ data and predict $2014 - 2015$ outcome.

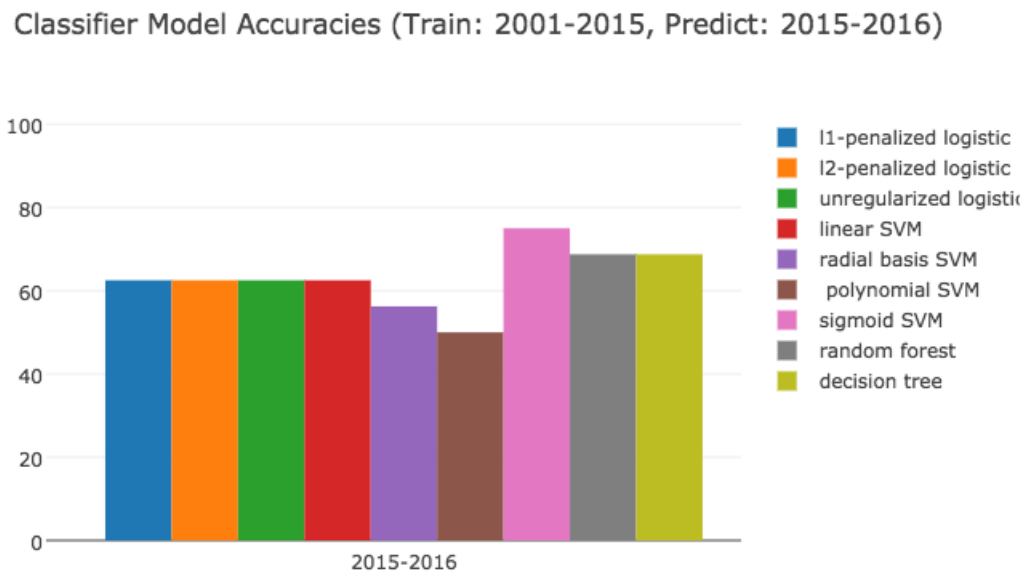


Figure B.14: Accuracies of models trained with $2001 - 2015$ data and predict $2015 - 2016$ outcome.