Milestone Grading Policy

As Speakers:

- All members **get ready** when it comes to your turn.
- **Present for 5 minutes** on introduction, solution, data, experiments, evaluation, results, and task distribution and planning**.**
- **Q&A for 2 minutes.**

As Listeners:

- Listen to the presentation and take some notes.
- Volunteer to **ask questions**.
- **(After class) Write down your NetID below. Give grades, questions, comments to each presentation on the grading forms.**
- **(On March 20 Tue) Return the grading forms to the instructor.** If you don't return it, your team will lose some points.
- You could download a print copy from the course website and print and read and grade if you missed the Presentations Day.
- *If you make detailed, constructive comments, your team will get extra points.*

As Instructor:

- Carefully read milestone papers and listen to the presentations.
- Carefully read grades, questions, comments given by students.
- Grade in scale of **0 to 15**. Members in a team will have the same grade. It takes 15% of the project and project takes 30% of the course – so, **15 = 4.5 points of 100** in the final score.
    - Grades will be given on **March 22 Thu** along with comments from students and the instructor**.**
    - *If the grade is lower than **9**, I will send an email to the team and find a time to talk about the progress at my office.*

Your NetID:

Your Name:

Milestone Presentation Schedule

Presentation & QA time	Team	Students			Title
Movie Session (2 projects)					
2:05-2:09, 2:10-2:11 **(slides attached)**	NPM	jborrero	Borrero Cordova, Juan	Senior	The Netflix Problem: Movie Clustering and Classification Based on Ratings
		bhansen4	Hansen, Brandon	Senior	
		mprosser	Prosser, Mason	Senior	
2:12-2:16, 2:17-2:18	ACC	rmackey1	Mackey, Ryan	Senior	Actor Clustering and Cast Significance on Genre and Movie Ratings
		kshin1	Shin, Kevin	Senior	
Sports Session (4 projects)					
2:19-2:23, 2:24-2:25	MLB	abrizius	Brizius, Alex	Senior	Predicting MLB Performance Based on Minor League Statistics
		mburke18	Burke, Michael	Senior	
		momalle3	O'Malley, Michael	Senior	
		jdumford	Dumford, Jacob	Graduate-Masters	
2:26-2:30, 2:31-2:32 **(slides attached)**	MML	sbanerj2	Banerjee, Sreya	Graduate-PhD.	Making March Less Mad - Predicting the NCAA Men's Basketball Tournament
		gwright3	Wright, Gabriel	Graduate-PhD.	
2:33-2:37, 2:38-2:39	EBM	nrao	Rao, Nathan	Junior	What Statistics are most Impactful? Examining Baseball's Metrics as Indicators for Success
		jspence5	Spencer, Joseph	Junior	
		rloizzo	Loizzo, Ryan	Junior	
		dchao	Chao, David	Junior	
2:40-2:44, 2:45-2:46 **(slides attached)**	POW	salpteki	Alptekin, Samuel	Junior	Predicting the Outcome of Week 1 Collegiate Football Games
		jbeiter	Beiter, Jacob	Junior	
		sberning	Berning, Samuel	Junior	
		bshadid	Shadid, Benjamin	Junior	

Life Session (4 projects)					
2:47-2:51, 2:52-2:53 **(slides attached)**	PBC	amital	Mital, Aman	Junior	Predicting Breast Cancer Diagnosis from Tumor Measurements
		anemecek	Nemecek, Andrew	Junior	
2:54-2:58, 2:59-3:00 **(slides attached)**	DPH	wbadart	Badart, William	Senior	Determining predictors of H-1B salary and approval
		lduane	Duane, Luke	Senior	
		wyu1	Yu, Wenhao	Non-Degree Student	
3:01-3:05, 3:06-3:07 **(slides attached)**	AFG	mgianni1	Giannini, Mark	Graduate-Masters	It's All Funds & Games - Predicting Kickstarter Success
		ptinsley	Tinsley, Patrick	Graduate-Masters	
		btunnell	Tunnell, Brian	Graduate-Masters	
3:08-3:12, 3:13-3:14	MPT	xwang41	Wang, Xueying	Graduate-PhD.	Misread-Proof Temporal Fact Extraction
		tzhao2	Zhao, Tong	Graduate-PhD.	

(from "Project instruction": http://www.meng-jiang.com/teaching/CSE647Spring18-Project.pdf)

The milestone will be graded as follows:

Introduction	15%	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Related Work:	10%	What other methods have addressed these or similar questions? How do these methods differ from your method?
Solution/Method:	25%	What did you do? What tools and techniques did you use? Was any innovation attempted?
Data and Experiments:	10%	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Evaluation and Results:	25%	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Writing/Presentation Quality	15%	Clarity of writing, speaking, visuals, organization, and grammar.

2:05-2:09, 2:10-2:11	NPM	jborrero	Borrero Cordova, Juan	Senior	The Netflix Problem: Movie Clustering and Classification Based on Ratings
		bhansen4	Hansen, Brandon	Senior	
		mprosser	Prosser, Mason	Senior	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

The Netflix Problem: Movie Clustering and Classification Based on Ratings

Juan Borrero, Mason Prosser, Brandon Hansen

Abstract

In this paper we introduce and explain a data science model to classify movies. The method evaluates certain features pertinent to each movie such as budget, genre, revenue, popularity, and production company to predict the respective movie's rating. The model combines several supervised machine learning algorithms including Naive-Bayes Theorem and ID3 and C4.5 Decision Trees for the classification task. Ultimately, the problem is to determine what movie attributes have the greatest correlation with the movie's rating and provide a fair prediction for released or unreleased movies based on their attributes. Of course, it is difficult to judge a movie based on a limited number of characteristics; nonetheless, the model provides a valuable critique based on objective data and an insight into potential performance of a movie. The examples used for the training data set are movies released before July 2017 and range from popular movies with large budgets and record revenues to unpopular, independent movies. Lastly, the performance of the proposed model is measured in terms of accuracy and how far the prediction is from the actual rating.

Keywords

Movie Classification, Decision Tree, Naive-Bayes, ID3, C4.5

Introduction

Throughout the past two decades, the movie industry has grown exponentially and introduced an important source of entertainment. The movie industry has become a major industry that promotes art and created a very competitive landscape for actors and directors. At the same time, the public is more demanding and the critics are harsher. All these factors lead to what we called "The Netflix Problem".

In summary, given the harsher critique and demand of the public that in many cases relies on the ratings to choose a movie, how do you predict the rating a new movie? We think that predicting movie ratings with high accuracy is something interesting and important because given the large variety of movies in streaming channels such as Netflix or Hulu, people tend to choose the movies to watch based on the ratings and customer reviews the movies have. Many online streaming services have poor rating predictions or provide rating predictions that do not give much information since the predictions do not vary much. For instance, most of the movies in Netflix have a movie rating and match rating above 90%, so the movies are not distinguishable, making the selection process difficult. Furthermore, the ratings provided on many movie rating platforms like Rotten Tomatoes, Metacritic, or IMDb may vary greatly. This is why developing a consistent data science model that can predict current and future movie ratings with a decent accuracy is useful.

Since the development of ID3 and C4.5 by Ross Quinlan, many prediction tasks use Decision Tree Algorithms to predict the labels of unknown data items. For our model, we use the same algorithms to create a decision tree with many features. The features in this case

are the attributes of the movies like genre, popularity, budget, revenue among others. Additionally, we use Naïve Bayes Theorem and plan to aggregate the results of all the algorithms together to increase the accuracy of the overall model. Regarding the technology stack, we use Python together with numpy, json, csv, and Pandas packages to construct our models. The packages Pandas, JSON, and csv serve to read, clean, and integrate the data to the models. Likewise, the numpy package helps to operate the large amount of information contained in the movies' data and build the algorithms.

Related Work

From previous work done in this area, we see methods relying on either industry business analytics like in the Piedmont system [3] or more basic exploratory data analysis. In our investigation of the subject, we are implementing machine learning methods in order to try and discover what attributes are most significant. Our general goal is prediction such as with the Piedmont system, but are treatment of prediction more closely matches that done as seen in the statistical analysis research. In the other statistics and EDA research analysis papers [1] [2] [4] [5], the focus was more on finding trends in the ratings of movies by genre, such answers would largely serve to help build accurate machine learning models that are working from sound assumptions. In our models genre will largely be considered as just another feature to be used in prediction of movie rating in conjunction with other variables.

Solution/Method

Data Cleaning

The initial data used had a number of missing entries in its original form, which gave rise to issues in its use for training different models on said data. This was resolved by either ignoring the faulty items entirely or having a placeholder value to indicate its absence. Each data item represents a unique and complex artifact, so replacing values with an average could be potentially detrimental to the data sets integrity. Depending on the model, different data organization techniques may be used in order to format attributes correctly for model training and testing.

Naive Bayes

As a basic attempt to predict a movies average rating from a set of basic features, we implemented the Naive Bayes learning algorithm in order to see if certain combinations of statistically significant attributes of a movie can be directly correlated with a movie's overall success in its rating. With this we are considering both nominal and numeric features, in which we bin numeric features for model implementation. By using the Naive Bayes algorithm we hope to find which movie features give strong indications of critical success by testing their predictive power in multiple combinations. Some features being considered for this model include primary genre, budget, runtime, popularity, and producer. In addition to binning numeric features, nominal features that have highly branching values but only a few dominant attributes within that set can be recontextualized so that extremely uncommon attributes are grouped into a new attribute of the same feature (for example, by creating a new 'Indie' category in producer).

Decision Trees (ID3, C4.5)

The use of decision trees will be a further attempt to see if basic movie features can be statistically correlated with its rating.

Similar binning techniques as the ones used for our Bayes implementation will also be used, with the binning of certain nominal features helping to combat issues with overfitting in the tree construction. Each of these methods can give good insight to the significance of some attributes over others, in the final model we will use the most accurate of the trees. The implementation of post-pruning methods will also be a viable option if in the case we want to limit specificity and improve its generalizability with subtree removal.

Data Set

The dataset we use is a Kaggle dataset and its name is: The Movies Dataset. The dataset is a CSV file and has 228MB worth of data. It consists of movies released on or before July 2017 and contains metadata on over 45 thousand movies with more than 26 million reviews from over 270 thousand users. The metrics and data points in the dataset are: ratings from 1 to 5, cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages. A link to the Dataset is found below:

- https://www.kaggle.com/rounakbanik/the-movies-dataset

Preliminary Results

The Naive Bayes model currently splits movie ratings into “Positive” and “Negative” classes, with the cutoff for a “Positive” score being above 5 on a 10-point scale. As of now, our Naive Bayes Model can predict whether or not a movie has a “Positive” score with 73.3% accuracy, which is a good start, but could stand to be further improved.

Our Decision Tree model is not yet finished, as we are still in the process of implementing functions to allow it to handle non-categorical features.

Plan

Since our initial progress was not as good as we had hoped, we plan to have weekly meetings moving forward to make sure we are all contributing the same amount of work and making solid progress. Attached at the end of the document is a Gantt chart describing both our progress and our plan moving forward.

References

[1] Miller, Thomas W. "Web and Network Data Science." Google Books. Accessed March 06, 2018. https://books.google.com/books?id=6J7fBQAAQBAJ&pg=PA130&lpg=PA130&dq=classify%2Bmovies%2Bdata%2Bscience&source=bl&ots=b8WDscztda&sig=rRfsjh4pPkI5INRUcGU5qXRS2o0&hl=en&sa=X&ved=0ahUKEwjNvZe6tdjZAhVH8IMKHTGjAtYQ6AEIiQEwCA#v=onepage&q=classify%20movies%20data%20science&f=false.

[2] "Predicting movie ratings with IMDb data and R." R-bloggers. March 02, 2014. Accessed March 07, 2018. https://www.r-bloggers.com/predicting-movie-ratings-with-imdb-data-and-r/.

[3] Piedmont Media Research. Accessed March 07, 2018. http://www.piedmontmedia.com/.

[4] "Moneyball for Movies: Market Research for Screenwriters." Creative Screenwriting. Accessed March 07, 2018. https://creativescreenwriting.com/moneyball-for-movies-screenwriters-and-market-research/.

[5] Posted by Sandipan Dey on December 16, 2017 at 1:30pm. "Data Science with Python: Exploratory Analysis with Movie-Ratings and Fraud Detection with Credit-Card

Transactions." Data Science Central. Accessed March 07, 2018. https://www.datasciencecentral.com/profiles/blogs/data-science-with-python-exploratory-analysis-with-movie-ratings.

Plan – Gantt Chart

Task	February		March		April	
	1st to 15th	15th to 28th	1st to 15th	15th to 31st	1st to 15th	15th to 30th
Defining Problem						
Collecting Data						
Data Cleaning - Incomplete/Inconsistent Data						
Data Integration						
Model Design						
- Naïve-Bayes						
- ID3						
- C4.5						
Bootstrap Aggregating						
Evaluation - Fitting Models						
Documentation						
- Milestone Paper						
- Final Paper						

Legend	
	Completed
	In Progess
	Pending

The Netflix Problem

Milestone Presentation
Mason Prosser, Juan Borrero, Brandon Hansen

Introduction

We introduce and explain a classification model to predict movie ratings. The method evaluates certain features pertinent to each movie such as budget, genre, revenue, popularity, and production company to predict the movie's rating. The model combines several supervised machine learning algorithms including Naive-Bayes Theorem and ID3 and C4.5 Decision Trees for the classification task.

Objective

Determine what movie attributes have the **greatest correlation** with the movie's rating and provide a fair **prediction** of movie rating for released or unreleased movies based on movie's attributes.

Decision Trees

26% Rotten Tomatoes

97% Rotten Tomatoes

Naive Bayes

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Data

The dataset we use comes from Kaggle and consists of movies released on or before July 2017. We do not use all the features because some are very unique leading to branching problems and little information gain.

Data Problems

- Incomplete

Many movies did not have the information for all the features.

- Inconsistent

Many entries within a column/feature did not match the type of data for the respective column.

We built a Cleaner class with the respective functions to go through the data, complete missing entries, ensure consistency, integrate data into one file and create a CSV file with the cleaned data.

budget	genres	movieId	original_lang	original_title	popularity	production_c	production_i	release_date	revenue	runtime	spoken_lang	vote_averag	vote_count
30000000	[{'id': 16, 'nai	862	en	Toy Story	21.946943	[{'name': 'Pix	[{'iso_3166_	10/30/95	373554033	81	[{'iso_639_1'	7.7	5415
65000000	[{'id': 12, 'nai	8844	en	Jumanji	17.015539	[{'name': 'Tri	[{'iso_3166_	12/15/95	262797249	104	[{'iso_639_1'	6.9	2413
0	[{'id': 10749,	15602	en	Grumpier Ol	11.7129	[{'name': 'Wo	[{'iso_3166_	12/22/95	0	101	[{'iso_639_1'	6.5	92

Current Problems and Solution

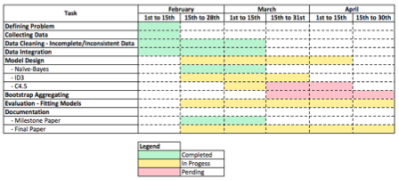
Currently our model uses **ID3 and C4.5** decision trees. The features in this case are the attributes of the movies like genre, popularity, budget, and revenue among others. Additionally, we use **Naive Bayes** model, and we plan to ensemble all the algorithms together to increase the accuracy of the overall model.

Naive Bayes → "Positive" and "Negative" classes for movie ratings,

- Cutoff: "Positive" > 5/10
- Accuracy: 73.3%, which is a good start, but could stand to be further improved.

Our Decision Tree model is not yet finished, as we are still in the process of implementing functions to allow it to handle non-categorical features.

Plan and Work Distribution



Moving forward, we plan to have weekly meetings on **Wednesdays** to keep track of the progress. Additionally, the work will be divided in equal parts among the three, especially Bootstrap aggregating, evaluation, and model design and improvement.

2:12-2:16, 2:17-2:18	ACC	rmackey1	Mackey, Ryan	Senior	Actor Clustering and Cast Significance on Genre and Movie Ratings
		kshin1	Shin, Kevin	Senior	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Project Proposal: Actor Clustering and Cast Significance on Genre

Ryan Mackey and Kevin Shin

1 INTRODUCTION

Given a data set regarding movies (rating, director, actors), can the genre and success of a movie be determined? And can meaningful clusters of people within the industry be determined?

1. Group actor/actress/directors into a clustering to see similarity in genre.
2. Categorize a movie into a genre based on who the director is and who is casted.

The methodology for determining each of these steps will go as follows:

1. Clustering: The k-medoids algorithm [1] will be used to determine groups of actors who work closely together. We will use the silhouette tactic to assist in determining the value of *k* before running the algorithm. We will retroactively examine select clusters to assign useful titles to the groupings.

2. SVM/Decision Tree: We will try both methods to determine which algorithm is better at classifying genres of movies. 75% of our data set will be used as training data and the remaining will be used as testing data. For genre determination, the model will be trained by cast names, director, and genre. For both classifications, inputs for testing will be the names of the actors and directors.

2 RELATED WORKS

3 PROBLEM DEFINITION

4 PROPOSED METHODOLOGY

Based on our SVM and/or Decision Tree, we can determine the accuracy of genre predictions and movie rating predictions. With genre, we will observe how many times the model detects the correct genre for the movies and output an accuracy percentage. With movie ratings, we will collect the variance of the outputted predicted result and use that to determine how precise our model is. Based on the accuracy and sample size, we can see whether cast and crew can be used in determining the genre or the average rating of a movie.

For the clustering problem, distances between people in the industry will be determined based on their association (movies worked on together, mutual connections, etc.) and the k-medoids algorithm will divide them into clusters as appropriate. We will try several k values, but we will inform

ourselves initially with the silhouette method. Once we are satisfied with the outcome, we will examine a selection of the clusters and consider why they are clustered together (genre, age, relationships etc.).

5 DATA AND EXPERIMENTS

5.1 Data Set

Our data will be drawn from IMDb (http://www.imdb.com/interfaces/). The information is broken up into 7 databases containing different information about the movies or actors in each. The movie unique identifier serves as the primary key for 6 of the databases, while the actor/actress/director unique identifiers serve as a foreign key in those 6 databases, and as the primary key in the 7th. Relevant data includes movie titles, top billed cast and crew on each, average IMDb ratings, and genre information.

3 datasets within the greater group of IMDb datasets are of particular interest for our study. For one, the title.basics.tsv dataset contains the information related to a given title that we will need. Information such as title name, media type, genre, and release date are all included here, tied to the primary key of the database, which is of the form ttXXXXXXX, where the X's concatenate to form an increasing integer and a unique ID for each entry. So, each unique entry refers to exactly one movie.

The second dataset of interest for our project is the name.basics.tsv file. This dataset holds information about the people working in the movie industry. Name, job, birth year and death year are included among other data. The most important information for us will be name, when it comes to time to manually infer the relationships within our clusters. The primary key in this database is of the form nmXXXXXX, where the X's concatenate into a number and increase to provide unique ID's similar to the ttXXXXXXX scheme in title.basics.tsv.

The third dataset of interest for our project is the title.principals.tsv file which includes up to 10 cast and crew members responsible for the creation of the given movie. The primary key is the ttXXXXXXX value, indicating which movie is being considered, then up to 10 nmXXXXXX values follow, indicating which people worked on the movie and what their jobs were.

5.1.1 Data Cleaning

We decided in advance that we only want to consider movies in our project. We made this determination based on the assumption that this media type will be widely distinct from other options such as TV shows and mini-series. We used the "type" field in title.basics to isolate only the IMDb entries for movies and clean out the irrelevant media types. This was done with a simple if statement in our cleaning script, and ended up narrowing our entries down from 4,000,000 to 600,000.

We narrowed the data down further by removing any entries lacking a genre. Our classifier is attempting to predict the genre based off of the actors, actresses, and directors involved in the film, so including data missing genre information would be helpful for neither training nor testing. The default value for missing data in this dataset is “\N”, so any record holding this value in the genre field was not included in our extraction. This brought the size of the dataset down from 600,000 to 560,000.

In the end, we pulled the film’s unique identifier, the title, and the genre for all movies or tvMovies that had at least one value in the genre field. We saved this output in a list of python lists.

From the title.principals tsv file, we extracted only the actors, actresses, and directors from the category feature as well as the movies correlated to these roles. We saved this information into a dictionary where the key was the tconstant for the title of the movie and the values were the actors, actresses, and directors who were a part of the production. We recognized how some movies had more crew members than others, and so to balance this out, we filled the dictionary values with placeholders ones (‘\N’). This would make it possible to fit these into a SVM.

5.1.2 Data Integration

We combined the extracted data from our title.basics tsv and title.principals tsv files into a list of lists containing the film ttID, name, genre, and actors/actresses/directors for a given movie. Only ttID’s present in both the title.basics list, and as a key in the title.principals dict were included in the finalized dataset. This reduced the size of dataset to 500,000, and represents our final reduction in numerosity. We wrote this new database into a new tsv file to consolidate our data. In order to create our training and testing set, we split up the data manually to about 75% training and 25% testing into separate files.

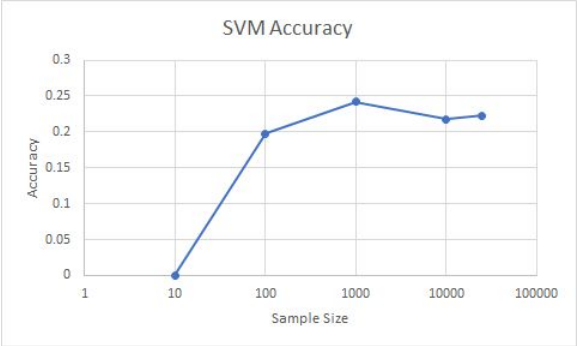
5.2 Experimental Settings

For our model, we took two approaches for our problem. We first implemented a SVM and then later implemented a decision tree. Both required us to transform our crew members’ unique IDs into floats so that our models could read it. We made a temporary solution by parsing out the ‘nm’ in the beginning of the crew IDs and then placing them in a vector space. We realize this is not as efficient and polished as utilizing OneHotEncoding or using the LabelEncoder, but we received some rudimentary and raw results. While working on this temporary resolution, we came to the conclusion that once we figure out clustering on our dataset that it will be much more feasible to determine and predict genres from our models.

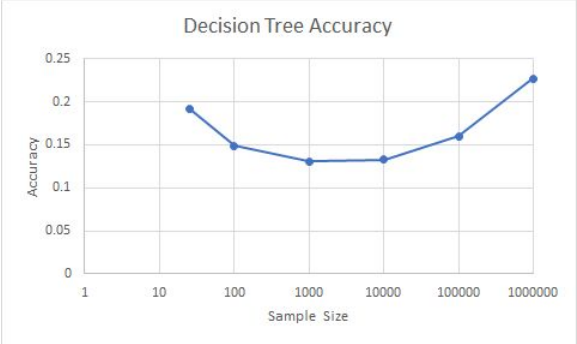
5.3 Evaluation Results

Because of time constraints, we were unable to run our models on the full dataset, but we did manage to run it on smaller sample sizes. When we tested our data with an SVM model, we ran it

on 10, 100, 1000, 10000, and 25000 samples sizes. Here are the results:



When we tested our data with a Decision Tree model, we ran it on 25, 100, 1000, 10000, 100000, and 1000000(full data set) sample sizes. Here are the results:



5.3.1 Result Discussion

So far, our results have been produced from an approximated method that will be improved after the inclusion of clustering Yet, the results we received are better than random guessing.There are 27 genres in the dataset, so random guessing should have an accuracy of around 0.04. The results from our Decision Tree and SVM are consistently above 0.1, and often exceed 0.2. The most interesting case comes with the Decision Tree with Sample Size set to 1,000,000. This number is large enough to include the entire training se, and the entire testing set, yet it output the highest accuracy of any sample size setting. This is encouraging as we move forward to refine our model.

Also, we look forward to organizing our data in such a way that makes it feasible to run an SVM model on the entire dataset. As of now we are only using 1/16 of our training set, and ¼ of our testing set. Improving these numbers will hopefully continue the upward trend observed between sample sizes of 10,000 and 25,000.

6 CONCLUSIONS

REFERENCES

[1] Hae-Sang Park, Chi-Hyuck Jun. 2008. *A simple and fast algorithm for K-medoids clustering*

2:19-2:23, 2:24-2:25	MLB	abrizius	Brizius, Alex	Senior	Predicting MLB Performance Based on Minor League Statistics
		mburke18	Burke, Michael	Senior	
		momalle3	O'Malley, Michael	Senior	
		jdumford	Dumford, Jacob	Graduate-Masters	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Predicting MLB Performance Based on MiLB Statistics

Alex Brizius
abrizius@nd.edu

Michael Burke
mburke18@nd.edu

Jacob Dumford
jdumford@nd.edu

Michael O'Malley
momalle3@nd.edu

ABSTRACT

In this paper we present a technique for predicting a successful Major League Baseball (MLB) career given the minor league statistics of a player. Our method takes data from the "Advanced" tab on fangraphs.com for the three highest Minor Leagues (A, AA, AAA) for the years 2006 through 2017[1]. Likewise, career batting statistics for MLB players from 2007 through 2017 was also exported from FanGraphs. In preliminary tests, we selected a threshold for the Wins Above Replacement (WAR) per Plate Appearance (PA) statistic for each MLB career and labeled players that exceeded the threshold as "successful" and players who did not as "unsuccessful". We then used a variety of classifiers, such as Naive Bayes and Random Forest, as well as two decision-tree approaches to predict the success of a given player in the testing data.

1. INTRODUCTION

In this paper we take a purely data-driven approach at predicting whether or not a baseball player will see success while playing Major League Baseball (MLB). Since the inception of professional baseball in the United States, baseball franchises have hired thousands of scouts and coaches to predict which players on their minor league rosters are worth adding to the major league roster. Young players get drafted out of high school or college and rather than getting put immediately onto a major league team, the team with the draft rights places that player on one of that organization's minor league teams. Then if that player excels (or at least catches the eye of the scouts) that player gets pulled up and is given a roster spot on the major league team. Some of those players continue their minor league success in the majors. These players continue to get plate appearances in the majors, whereas players who struggle will get placed back on the minor league team. Millions of dollars are spent on attempts to find better talent and improve a team's chances of winning games. In the past, the most common way to find potentially successful players was to hire scouts that either watched them play or made an assessment based on what stats they believed were the most important. With the wealth of data that the sport of baseball produces, it makes sense to use this data in a systematic way. Using data science or machine learning techniques to predict various aspects of baseball has become more and more common. It continues to be a hot topic, because the data recorded by the MLB is so vast that humans cannot efficiently analyze it all. Baseball offers another field in which computers have the opportunity to greatly outperform humans and change the way that decisions are made.

[1]Anon.Retrieved March 1, 2018 from fangraphs.com

2. RELATED WORK

Statistics have always been a huge part of baseball with the "Moneyball" story revolutionizing the game and changing the way that player selection is done. It showed that when you pay attention to the statistics rather than which players "look" a certain way, you can see great success for a much cheaper price tag. Since then, there has been quite a lot of research done in the realm of using data to predict baseball outcomes. Data is constantly used when evaluating the success of players, but when it comes to modeling success with regressions or other data science or machine learning techniques, most of it has been focused on predicting success for a team as a whole. A study published in the Athens Journal of Sports applied SVMs to several team statistics to attempt to predict the winner of the World Series[2]. They did not see groundbreaking results, but in their study a Gaussian Kernel RBF SVM offered the best prediction of a World Series victory. Another study conducted at Illinois Wesleyan University[3] conducted a study to find out which statistic is the most valuable when predicting success of a team as a whole. They concluded that best offensive stat for predicting a team's success, and therefore the statistic to look at when considering potential players, is On Base Percentage (OBS). They also considered defensive statistics and found that Walks plus Hits per Inning Pitched (WHIP) is the best defensive indicator of success. As baseball is a business, many of the studies done on baseball statistics are from the perspective of winning games with the lowest payroll possible. While our study may be useful in minimizing costs, this is not the focus of the study. Our focus is on finding successful players and reasoning that they should make

[2]https://www.athensjournals.gr/sports/2016-3-4-1-Tolbert.pdf
[3]https://www.iwu.edu/economics/PPE13/houser.pdf

the jump from the minors to the majors.

3. PROPOSED METHODOLOGY

To solve this problem, we intend to implement a few different models and test them to see which gives us the best results. With the data we have gathered, we plan to implement ID3 and CART decision trees as well as Naive Bayes, Random Forest, and AdaBoost classifiers. We will test each algorithm on our test data and choose the one that performs the best. We plan to use Python to build each of these models. The Pandas python library offers great tools when it comes to data processing and analysis. We used Pandas for the step of filtering out statistics that we chose not to use in our model as well as joining the MLB data with the MiLB data. From there, the sklearn library gives most of the functionality we need; namely, there are modules for decision trees, Naive Bayes, Random Forest, and Adaboost.

After analysis of the accuracy (and other evaluation statistics) for each model, the experimenters plan to select a subset that performs best for each league. In the end, the experimenters will likely rely on a variety of models for prediction rather than only one.

4. DATA AND EXPERIMENTS

4.1 Data Set

For this project, the advanced batting statistics for MiLB from 2006 to 2017 were gathered for each player that in a particular year had above a minimum threshold of Plate Appearances (PA); if a player appeared in multiple years' datasets, each statistic was averaged with PA's as the weight. Then, the dimensionality of the dataset was reduced heuristically. Namely, the following features are preliminarily used:

- BB% (Walks per Plate Appearance)
- K% (Strikeouts per Plate Appearance)
- BB/K (Walks per Strikeout)
- AVG (Batting Average)
- OBP (On-Base Percentage)
- SLG (Slugging Percentage)
- OPS (On-Base + Slugging Percentage)

(Note: Explaining the intricacies of each of these measurements is beyond the scope of this paper, and exact formulas for calculating these are widely available on the internet. Notably, each of these data points is relatively easily computable for anyone with even basic batting data).

Next, after combining and reducing the dataset, this numeric dataset is converted into a parallel ordinal data set. For each feature, the standard deviation and mean of each feature within a given league is computed, and each tuple's numerical value is converted to an ordinal value based on its relation to the mean and standard deviation. In other words, the numerical data is mapped to 0 (well below average), 1 (within 1 standard deviation of average), or 2 (well above average).

Finally, the label of each tuple is the success of that player in the MLB. If the player has not appeared in the MLB, then the corresponding tuple in the dataset is labeled as unsuccessful (0). Next, the mean WAR/PA is calculated for career MLB data, and if a given player's WAR/PA is above this threshold, they are labeled as successful (1); otherwise, they are labeled as unsuccessful. Moving forward, a better method for labeling "success" may be investigated.

4.2 Experimental settings

For initial experiments, five different models were trained and used against the compiled sets of data: simple entropy, Bayes, Gini, Random Forest, and AdaBoost. This allowed us to directly compare different methods and determine which is most effective across all sets of data. Additionally, this allowed us to consider the average accuracy that the models were producing from the data, providing important insight as to whether we are using the correct data on evaluating players by the proper metrics. Precision, recall, and other evaluation metrics were also produced. The main goal of these settings was not to assess the models we are using, but to assess the data we are using and how we can improve our cleaning and integration to produce more reliable results. Model is important, but the significance of our project comes from the significance behind the data.

4.3 Evaluation Results

Thus far, some trends have already began to appear; namely, the "lower" the level of the minor league in question, the easier it is to predict the success of a given player in the league. This is exemplified by the accuracy of different models all being greater than .90 for Single A, yet all below .75 for Triple A.

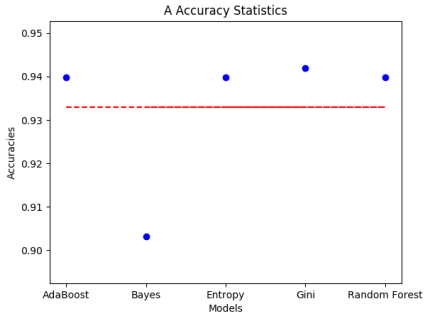


Figure 1: Accuracy for Each Method for Single A Data

For each of the minor leagues, Naive Bayes is outperformed by every other model.

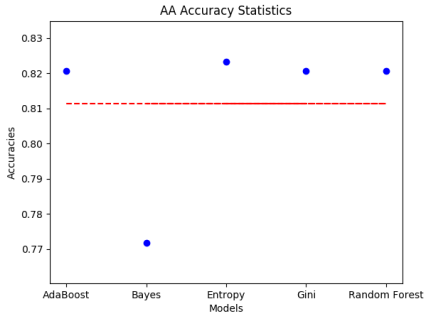


Figure 2: Accuracy for Each Method for Double A Data

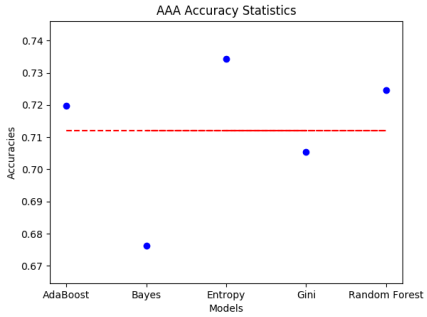


Figure 3: Accuracy for Each Method for Triple A Data

5. RESULTS

Up next for major consideration is how "success" is determined for a given MLB player; the existing method of comparing to the mean WAR/PA has given some success but instinctively seems too arbitrary. As stated before, this success of this project is determined by both the quantity of the data as well as the strength of the models. Choosing which metrics have the best impact and provide the most information is tricky because of the vast quantity of distinct measurements involved in baseball. In addition, different positions and functions as players can yield very different ideas of what it means to be successful. Developing a solution to predict overall success, considering this, is very difficult. This is where our current results come into play. Considering the data we have collected and the accuracies of the applied models, we can tailor the data to best predict success.

Additional metrics may prove necessary to increase the accuracy of our model; for example, only comparing players within the same position may help eliminate bias towards players in less defensively-intense positions. Likewise, defensive metrics, although difficult to collect, are also important representations of a player's success, and may help predict what players a MLB team may identify as worth extra investment.

6. REFERENCES

"Society for American Baseball Research | Society for American Baseball Research." Society for American Baseball Research | Society for American Baseball Research, sabr.org/.

"Baseball Encyclopedia of MLB Players." Baseball-Reference.com, www.baseball-reference.com/players.

"The Official Site of Major League Baseball." MLB.com, www.mlb.com/.

"The Official Site of Minor League Baseball | MiLB.com Homepage." MiLB.com, www.milb.com/index.jsp.

2:26-2:30, 2:31-2:32	MML	sbanerj2	Banerjee, Sreya	Graduate-PhD.	Making March Less Mad - Predicting the NCAA Men's Basketball Tournament
		gwright3	Wright, Gabriel	Graduate-PhD.	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Making March Less Mad - Predicting the NCAA Men's Basketball Tournament

Sreya Banerjee
University of Notre Dame
Notre Dame, Indiana
Sreya.Banerjee.9@nd.edu

Gabriel Wright
University of Notre Dame
Notre Dame, Indiana
Gabriel.S.Wright.142@nd.edu

Abstract

March madness is around the corner where the top 64 college basketball teams from across America participate to win the national championship. For basketball enthusiasts and computer scientists alike, in essence, this becomes a binary classification task where, with sufficient historical data, we can try to predict the outcome of tournament. In this work, we plan to implement and evaluate some common supervised machine learning approaches to predict the result of the 63 games within the tournament, based on many years worth of previous basketball tournament results. The project would involve common data science tasks like data integration, augmentation, cleaning, feature extraction, feature selection, model building and tuning. Some algorithms we wish to test are: logistic regression, Naive Bayes and support vector machine. For evaluation we plan to use the classification accuracy as a metric.

1. Introduction

Given the amount of dollars spent on gambling every year to predict the outcome of NCAA men's basketball tournament, March Madness is undoubtedly one of the biggest event in America and has generated interests of scientists worldwide. It has been an active area of research in fields as diverse as social psychology to statistics to computer science [4, 6, 7, 8, 10].

In this work, we plan to implement and evaluate some common supervised machine learning approaches to predict the result of this year's 63 games within the tournament, based on almost a decade worth of basketball results. Formally, our problem can be defined as, given two data sets: one being a data set of the outcome of NCAA college basketball games from previous years, and the other containing individual statistics for each team, is it possible for machine learning algorithms to predict the results of the games in the tournament? Essentially, the task can be formulated as a binary classification problem with two possible outcomes of win or lose for each game. A "successful" model would be one that outperforms the null model where each game is picked based on tournament seeding.

For the classification task, we plan to use the common machine learning algorithms like Naive Bayes, support vector machines (SVM) [3] and logistic regression. We chose the data set [1] where the results of basketball matches each year is tabulated as csv file. Additionally, the dataset [2] provide advanced statistics for each team in each year. We plan to use an almost a decade worth of data. However, since the results of 2018 are not present, we plan to create them ourselves. The data from 2018 would be solely used for prediction purpose.

For the tasks, we divide the data set into training (all the historical data) and testing (the 2018 tournament) components. We will use the training set to fine-tune the model through cross-validation evaluation. To select the best classification model, we plan to use the classification accuracy metric on the validation data set.

2. Related Work

The idea of using machine learning for decision making in sports is not an uncommon phenomenon [2, 4, 9]. Previous work in this regard includes detecting goals through support vector machines [1]. Quite recently, Brooks *et al*. [2] developed a data-driven strategy to rank players in soccer based solely on the pass origins and destinations during a possession with the probability of a shot. Their method correctly guesses whether a possession ends in a shot from the pass locations alone. They used the 2012-2013 La Liga season as a dataset for their model. When it comes to volleyball, Van Haaren *et al*. [9] proposed a relational-learning based pattern discovery method based on spatial and temporal cues of games. For their data set they used both the men's and women's final match from the 2014 FIVB Volleyball World Championships and were able to discover interesting strategies for the game.

[1] https://www.kaggle.com/c/march-machine-learning-mania-2017/data.

[2] https://www.sports-reference.com/

March madness is no exception. Earlier work in this regard primarily concentrated on building logistic regression based probability models using seed positions [7, 10], nearest-neighbour based approach [4] or specialized logistic regression/Markov chain models[5] for predicting the outcomes of matches. Our work primarily differs from them in the following aspects:

1. A large dataset comprising of $42,194$ matches from 2010-2017.

2. More than one attributes or features that are most predictive of the outcomes of game from knowledge/experience.

We believe having access to a large dataset and predictive attributes are crucial in designing any machine learning based model.

3. Data Acquisition and Pre-processing

In order to build our models to predict the NCAA tournament we took data from two sources: Kaggle and Sports-Reference.com.

At Sports-Reference.com, statistical data is provided for each Division 1 basketball team for a given season. We chose to use team statistics from the last eight seasons. The motivation for using more recent data was both to save time compiling the data and because the game of basketball is constantly evolving, more recent statistical data will be more predictive of current results. For example, the recent trend in professional basketball has moved away from dominant centers and has put a premium on dominant three point shooters. Trends like this may show up in team statistics.

Kaggle, the self proclaimed "Home of Data Science & Machine Learning", is an online host for data science competitions on a wide variety of topics. As part of these competitions they freely provide a large amount of data for users to work with. For the past several years Kaggle has hosted a March Madness competition, and this year is no different. Along with the competition, they have published a large amount of college basketball data for public use. Of the files they have published, we only concerned ourselves with two - "RegularSeasonCompactResults.csv" (RSCR) and "TeamSpellings.csv" (TS).

In the RSCR file, results are provided for every Division 1 basketball game dating back to 1985 (a total of 150k games). One quirk of the Kaggle data is that teams have been mapped to ID numbers (e.g. Notre Dame is team number 1323), making it incompatible with the data files from Sports-Reference. leading us to need the TS file. In TS, multiple possible spellings of each team are given with the corresponding team ID, with the intent of allowing external data to be mapped to the provided Kaggle data. We used this file to map the Kaggle data to the Sports-Reference data, resulting in a file containing results for $42,194$ games with statistics for each team that played in each game.

Basketball-Reference provides numerous attributes (>50) for each team for a given year. Before modeling we narrowed these attributes down to wins, SRS (Simple Rating System) score, true shooting percentage (TS%), effective field goal percentage (eFG%), and offensive rating (ORtg). SRS, TS%, eFG%, and ORtg are defined both for each team and for the opponents of each team (to reflect a team's relative defensive prowess).

4. Proposed Methodology

4.1. Models/Results

So far we have used a divide and conquer approach to modeling, each taking our final data set and creating a model from it. As per plan, we have successfully implemented SVM and Logistic Regression.

4.1.1 Support Vector Machine

Since our data is numeric and high-dimensional, SVM [3] was a natural choice as it has been found to be extremely efficient in high-dimensional spaces for large-scale classification problems. SVMs use a subset of training points in the decision function (called support vectors). As a consequence, it has been found to be memory efficient and has faster execution times if the data is normalized. For analysis, we assumed our data to be linearly separable and used a linear SVM kernel. We also normalized our data using min-max normalization.

A SVM model, with a set of labelled training data tries to find an optimal hyperplane for classifying new samples based on some constraints.

Given a training dataset, D = (x_i,y_i) of size m with x_i = (x_1, x_2,.., x_n), an n-dimensional feature/attribute vector and label, y_i= -1 or +1, formally the SVM classifier can be defined as a quadratic optimization problem solving the following equation:

$$min \left\|w\right\|^2 \text{ s.t } y_i(w^T x_i + b) \geq 1 \text{ for all } i \tag{1}$$

where w = (w_1, w_2, ...,w_n) is a weight vector and b is bias.

An important consideration when designing a SVM model is the parameter C that dictates the trade-off between having a wide margin and correctly classifying training data.

$$min \left\|w\right\|^2 + C\sum_{1}^{m} \xi_i \text{ s.t } y_i(w^T x_i + b) \geq (1 - \xi_i) \text{ for all } i \tag{2}$$

Needless to say, a higher value of C implies a lesser number of mis-classified training samples and is prone to overfitting.

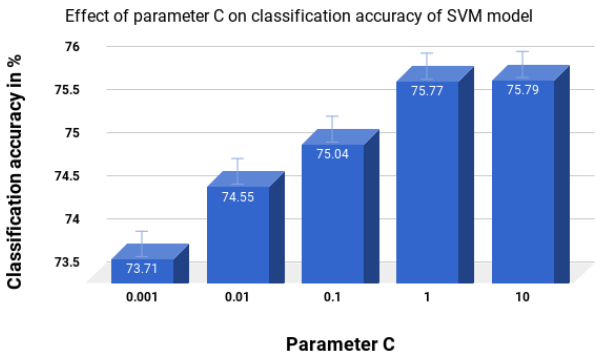


Figure 1. Results of tuning C parameter on SVM model. $C = 10$ performs relatively better than other methods in terms of classification accuracy.

Experimental settings: We performed two quantitative experiments to measure the goodness of fit for our SVM model.

The first experiment consisted of finding an optimal C value based on a fixed training and testing set; such that the model has higher prediction accuracy on unseen data without losing its generalization capability. This can be achieved in two ways: manually setting up the value of C and using grid search for parameter optimization. For the sake of completeness, we are only providing the results of tuning C manually. We plan to use grid search for our final report.

For the second method, we used k-fold cross-validation technique as a method to improve our model. Ideally, cross-validation should improve classification accuracy as it divides the data into number of folds specified (suppose, k) and trains on (k-1) sets while validating or testing the model on the other set, k times. For our experiments, we divided our data into 8 folds, both randomly and based on years available (2010-2017) and tested it against normal hold-out cross validation approach where 20% of the data was used for testing. The basis of dividing the basketball data for cross validation into sets using years available can be attributed to the fact that the chance of winning or losing a match might be correlated with team's form(or statistics) that year.

Evaluation results: Figure.1 details the result of tuning the C parameter keeping the training and testing data fixed. The results of second experiment are consolidated in Table.1 As expected, a higher value of C improves the model performance in terms of classification accuracy. However, it is still not high enough. Possibly a better parameter optimization technique like grid search would improve the result. For the second experiment, the 8-fold cross validation with randomly partitioned data performs relatively better than the other tested methods.

Method	Accuracy
with 20% hold-out training data	0.76
with 10-fold cross validation	0.76(+/- 0.02)
with 8-fold cross validation	0.76(+/- 0.01)
with cross validation on year(=8)	0.76(+/- 0.48)

Table 1. Results of cross-validation on SVM model. 8-fold cross validation performs the best in terms of classification accuracy.

4.1.2 Logistic Regression

Predicting the outcome of games can be formatted as a binary classification problem. For example, given team statistics for Team 1 and Team 2 (order arbitrarily decided), does Team 1 win the game? A very common approach to solving such a problem is a logistic regression model. Logistic regression assigns weights to each input attribute and outputs a value between 0 and 1. This output can be viewed as a probability of success relative to the target variable (in this case Team 1 winning), with any probability >0.5 being considered a "success". This is how we formatted our classification problem.

Experimental settings: For this model we did some attribute combination to make the results more intuitive. For each game in our data set, a number of our attributes (SRS, eFG%, TS%, and ORtg) are defined four times:

1. How Team 1 ranks in these categories
2. How teams who have played against Team 1 did against Team 1 in these categories
3. How Team 2 ranks in these categories
4. How teams who have played against Team 2 did against Team 2 in these categories

Items 2 and 4 in the list above can be viewed as defensive statistics for Team 1 and Team 2 respectively. Therefore, it makes sense to combine the four versions of each attribute into one attribute. This was done as follows:

$$(A_1 - A_2) - (A_3 - A_4)$$

Where the first set of parentheses represents how Team 1 does overall for a given statistic, and the second set does the same for Team 2.

Evaluation results: The logistic regression model was run ten times with 70% of the games randomly chosen to be in the training set for each iteration, and the remaining 30% placed in the testing set. The average accuracy on testing set across the 10 iterations was 0.762 which conforms to the classification accuracy achieved using SVM.

References

[1] N. Ancona, G. Cicirelli, A. Branca, and A. Distante. Goal detection in football by using support vector machines for classification. In *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, volume 1, pages 611–616. IEEE, 2001.

[2] J. Brooks, M. Kerr, and J. Guttag. Developing a data-driven player ranking in soccer using predictive model weights. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 49–55. ACM, 2016.

[3] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[4] A. Hoegh, M. Carzolio, I. Crandell, X. Hu, L. Roberts, Y. Song, and S. C. Leman. Nearest-neighbor matchup effects: accounting for team matchups for predicting march madness. *Journal of Quantitative Analysis in Sports*, 11(1):29–37, 2015.

[5] P. Kvam and J. S. Sokol. A logistic regression/markov chain model for ncaa basketball. *Naval Research Logistics (NrL)*, 53(8):788–803, 2006.

[6] S. M. McCrea and E. R. Hirt. Match madness: Probability matching in prediction of the ncaa basketball tournament. *Journal of Applied Social Psychology*, 39(12):2809–2839, 2009.

[7] N. C. Schwertman, K. L. Schenk, and B. C. Holbrook. More probability models for the ncaa regional basketball tournaments. *The American Statistician*, 50(1):34–38, 1996.

[8] T. Smith and N. C. Schwertman. Can the ncaa basketball tournament seeding be used to predict margin of victory? *The American Statistician*, 53(2):94–98, 1999.

[9] J. Van Haaren, H. Ben Shitrit, J. Davis, and P. Fua. Analyzing volleyball match data from the 2014 world championships using machine learning techniques. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 627–634. ACM, 2016.

[10] B. T. West. A simple and flexible rating method for predicting success in the ncaa basketball tournament. *Journal of Quantitative Analysis in Sports*, 2(3), 2006.

Making March Less Mad: Predicting the NCAA Tournament

Sreya Banerjee and Gabe Wright
Department of Computer Science & Engineering

1

UNIVERSITY OF NOTRE DAME

2 NCAA Basketball Tournament

- 64 top teams compete each year.
- "Bracketology" is a common practice with both fans and non-fans.
  - Warren Buffett once offered $1 Billion to anyone who could fill out a perfect bracket.
- Create a model that best predicts the tournament.
  - "Best" meaning model that picks most games correctly.
  - Aim to outperform null model based on seeding.

UNIVERSITY OF NOTRE DAME

3 Tasks

Given two data sets: one being a data set of the outcome of NCAA college basketball games from previous years, and the other containing individual statistics for each team:

- ❑ **Classification**: Is it possible for machine learning algorithms to predict the results of the 63 games in the tournament?

UNIVERSITY OF NOTRE DAME

4 Data

- ❑ Every game going back almost a decade (2010-2017), individual statistics for each team. 42, 194 games with 16 attributes.
- ❑ Dataset to be created by augmentation and cleaning from sources[1,2].

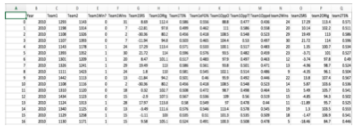


Fig: Snapshot of one of the 2 sources to be used to create dataset containing 12 attributes, some of which are empty.

Attributes:
Individual Team Stats
SRS = Simple Rating System
ORtg = Offensive Rating
TS% = True Shooting %
eFG% = effective field goal %

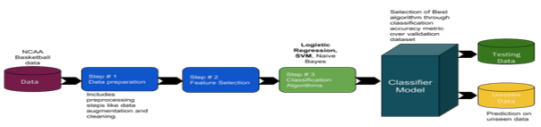
1: https://www.kaggle.com/c/march-machine-learning-mania-2017/data
2: https://www.sports-reference.com/

UNIVERSITY OF NOTRE DAME

5 Methods & Evaluation

Algorithms: Logistic regression, Naive Bayes, Support Vector Machines & Decision tree

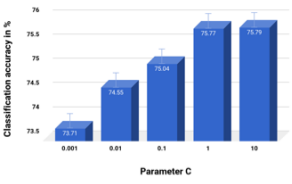
Evaluation : Classification accuracy on testing dataset.



UNIVERSITY OF NOTRE DAME

6 Support Vector Machine

Experiment 1: Effect of parameter C



Experiment 2: Effect of cross-validation

Method	Accuracy
with 20% hold-out training data	0.76
with 10-fold cross validation	0.76(+/- 0.02)
with 8-fold cross validation	0.76(+/- 0.01)
with cross validation on year(=8)	0.76(+/- 0.48)

UNIVERSITY OF NOTRE DAME

7

Logistic Regression

- Attribute combination to make the results more intuitive.
  - How Team 1 ranks in these categories
  - How teams who have played against Team 1 did against Team 1 in these categories
  - How Team 2 ranks in these categories
  - How teams who have played against Team 2 did against Team 2 in these categories
- Run 10 times with 70% randomly chosen games in the training set for each iteration, 30% placed in the testing set.
- **Average classification accuracy** on testing set across the 10 iterations was **0.762**

UNIVERSITY OF NOTRE DAME

Thank You

UNIVERSITY OF NOTRE DAME

2:33-2:37, 2:38-2:39	EBM	nrao	Rao, Nathan	Junior	What Statistics are most Impactful? Examining Baseball's Metrics as Indicators for Success
		jspence5	Spencer, Joseph	Junior	
		rloizzo	Loizzo, Ryan	Junior	
		dchao	Chao, David	Junior	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Milestone Paper: What Statistics are most Impactful?

Examining Baseball's Metrics as Indicators for Success

Nathan Rao, Joseph Spencer, Ryan Loizzo, DJ Chao

ACM Reference format:
Nathan Rao, Joseph Spencer, Ryan Loizzo, DJ Chao. 2018. **Milestone Paper: What Statistics are most Impactful?**. In *Proceedings of Data Science 40647, Notre Dame, IN USA, Spring 2018,* 3 pages.
https://doi.org/10.475/123_4

1 ABSTRACT

In this paper we present a technique for predicting Major League Baseball regular season outcomes using data compiled from previous seasons. Due to the randomness of individual games, looking at specific head to head match ups would not result in accurate predictions; therefore, using full data across multiple seasons would better indicate to us which statistics matter and how accurate they are at picking the regular season outcomes. The technique relies on a decision tree using the ID3 algorithm to classify a final group of 10 teams that we believe should be the final playoff bracket. We anticipate further cementing which statistics we will use for the decision tree and determine the most accurate threshold value for each branch to properly determine the playoff teams throughout the project. Additionally, we do not build any statistics for our model, all numbers are pulled from our reference sources. The performance of the proposed classification will be measured in terms of how accurately the final 10 teams are predicted compared to the real seasonal outcomes.

2 INTRODUCTION

Given a comprehensive database of major league baseball's raw and advanced metric statistics, how accurately can specific team and individual statistics determine the outcome for any given team's season?

The data will primarily be directed at determining the final outcome of the regular season - the final playoff bracket. However, aside from the main task of determining the playoff teams, the data may be directed to the examining the statistical strength to potentially find the "Cinderella Story" spoken about so often across all of the major sports.

We will pull complete team and individual statistics for analysis. The variety of data will give us the greatest accuracy in determining which statistics have the greatest meaning come season's end. This data will be primarily pulled from Baseball-Reference (https://www.baseball-reference.com/), one of the largest baseball statistic compilation websites on the internet.

Our analysis is based on determining the proper statistics to predicting regular season outcomes and then using those statistics to build a decision tree for the playoff teams. The regression we utilize here develops an analysis of the statistics to build the proper statistics we should use for our final decision tree.

We will evaluate our final method by applying our final algorithm to previous seasons and seeing how accurately it predicts the end of the regular season standings and postseason outcomes. Correctly predicting the teams that move on to the postseason and win the championship will deem our method and algorithm successful.

Determining if our method is successful is relatively straightforward. If the predictions are accurate, as in predicting the correct teams that move on to the postseason and win the championship, our method and algorithm will be successful.

3 DATA SET

The data set that we are examining is comprised of data that we pulled from baseball-reference.com. For the initial portion of our project, we pulled data from the 2016 MLB season. We pulled a total of 4 separate data sets: batting stats, pitching stats, fielding stats, and standings. We then combined these four data sets into one in order to examine. Going forward, we plan on pulling data from at least 10 MLB seasons for use as testing and training data.

4 METHODOLOGY

In order to identify the ideal statistics to use, we decided to use a logistic regression analysis, where each of the chosen statistics is a predictor and the response variable is Playoffs, a binary response where 1 corresponds to a team making the playoffs, and 0 refers to a team that does not make the playoffs. In order to run these regressions and analyze the results, we will utilize R.

5 PRELIMINARY RESULTS

For our preliminary results, we achieved our first goal, which was to determine an ideal subset of statistics in order to use as features for our decision tree. We have access to literally hundreds of different statistics to choose from, so it is important that we have some sort of statistical analysis in order to choose our subset of statistics.

We started by parsing all of the statistics that were available to us, and using our best judgment in order to create a relatively large list of statistics. This list included statistics that we deemed likely to be relevant in predicting a team's success. We also made sure to avoid statistics that were likely to be highly correlated with each other, such as hits and home runs, or hits and RBIs. Our initial list of statistics is as follows:

Data Science 40647, Spring 2018, Notre Dame, IN USA

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

- Runs (R)
- On Base Percentage Plus Slugging (OPS+)
- Strikeouts (SO)
- Home Runs (HR)
- Fielding Percentage (FP)
- Double Plays (DP)
- Total Zone Per 1200 Innings (RTot/yr)
- Errors (E)
- Earned Runs Average Plus (ERA+)
- Fielding Independant Pitching (FIP)
- Walks Hits per Innings Pitched (WHIP)

For our first equation, we saw the following results:

$$glm(f = Playoffs\ R + OPS + SO + HR + DP + Fld + Rtotpyr + E + ERA + FIP + WHIP)$$

Estimate	Std.	Error	t value	Pr(>|t|)
(Intercept)	4.637e+02	2.613e+02	1.775	0.09288 .
R.x	7.689e-03	1.990e-03	3.864	0.00114 **
OPS.	-5.079e-02	1.923e-02	-2.641	0.01662 *
SO.x	3.910e-04	8.474e-04	0.461	0.65007
HR.x	5.693e-03	3.671e-03	1.551	0.13831
DP	-6.972e-03	6.187e-03	-1.127	0.27462
Fld.	-4.562e+02	2.617e+02	-1.743	0.09832 .
Rtotpyr	7.073e-02	4.716e-02	1.500	0.15100
E	-6.818e-02	4.243e-02	-1.607	0.12550
ERA.	-4.418e-02	2.033e-02	-2.174	0.04333 *
FIP	-9.792e-01	3.953e-01	-2.477	0.02339 *
WHIP	-3.600e-01	2.116e+00	-0.170	0.86679

Null deviance: 6.9667 on 29 degrees of freedom
Residual deviance: 2.1759 on 18 degrees of freedom

Looking at the P values for each explanatory variable, we see that Strikeouts and WHIP are highly insignificant in predicting Playoffs. Therefore, we will remove these two variables from the equation. After removing these two, removing the next most insignificant variable (DP) causes a significant increase in residual deviance. The equation for this is shown in the methodology section. The results are as follows:

$$glm(f = Playoffs\ R + OPS + HR + DP + Fld + Rtotpyr + E + ERA + FIP)$$

Estimate	Std.	Error	t value	Pr(>|t|)
(Intercept)	4.245e+02	2.364e+02	1.795	0.087749 .
R.x	7.443e-03	1.699e-03	4.381	0.000289 ***
OPS.	-4.988e-02	1.752e-02	-2.847	0.009952 **
HR.x	6.711e-03	2.845e-03	2.359	0.028619 *
DP	-8.243e-03	5.277e-03	-1.562	0.133925
Fld.	-4.165e+02	2.366e+02	-1.760	0.093639 .
Rtotpyr	7.673e-02	4.201e-02	1.826	0.082741 .
E	-6.043e-02	3.739e-02	-1.616	0.121689
ERA.	-4.631e-02	1.781e-02	-2.601	0.017103 *
FIP	-1.049e+00	3.312e-01	-3.168	0.004841 **

Null deviance: 6.9667 on 29 degrees of freedom
Residual deviance: 2.2033 on 20 degrees of freedom

We see a slight increase in residual deviance, but this is because adding a variable to a model always decreases residual decrease, so removing one always increases it. Since the increase is small, it shows that this model is likely a better fit than the original. Therefore, our final subset of statistics and results is as follows:

- Runs (R)
- On Base Percentage Plus Slugging (OPS+)
- Home Runs (HR)
- Fielding Percentage (FP)
- Double Plays (DP)
- Total Zone Per 1200 Innings (RTot/yr)
- Errors (E)
- Earned Runs Average Plus (ERA+)
- Fielding Independant Pitching (FIP)

With these results, we can now continue to our next step and use these statistics in order to create our ID3 decision tree.

6 CHALLENGES AND PROPOSED SOLUTIONS

The analysis of Major League Baseball statistical seasons to properly predict regular season outcomes faces a few major difficulties. First, there is a non-negligible statistical difference between the two leagues, the American and National Leagues, that divide Major League Baseball. Each league has its own set of rules for batting and they are slightly different from the other league. For the National League (referenced as the "NL"), the pitcher bats in the lineup. So, whenever the spot currently designated to the pitcher comes up to bat, either the pitcher or a pinch-hitter (a sort of substitute) bats in that position. This regularly has a major effect on the batting statistics in the NL, as pitchers are typically much worse at batting that outfield players. Additionally, the inclusion of pitchers in the batting order changes the strategic makeup of NL games, as the manager must decide, as the game enters the later innings, if he should take out the pitcher for a pinch hitter to increase the chance for a positive at bat when the pitcher's spot is up to bat. All of these things can have a major effect on statistics in the NL, for pitching,

batting, even defense. The pinch hitter and defensive switch effect numbers, but the same rules apply for the American League (referenced as the "AL") as the NL so there is a negligible difference between the two leagues. For the AL, instead of the pitcher batting, the manager designates a hitter, called the designated hitter, to bat in the lineup in lieu of the pitcher. This batter only bats over the course of the game and is not in the outfield unless the manager switches players and positions with substitutes during the game. These differences are not negligible, and can greatly effect the difference between teams making and missing the postseason. However, consideration must also be given to the fact that the postseason is decided inside of the leagues - the outcome of the one league has no bearing on the postseason outlook of the other. To address this challenge, we plan on dividing the data set between AL and NL before examining stats and figuring out which are most impactful for predicting team success.

Another challenge faced is the idea of one of the four V's of data science: Volume. Baseball as a whole has amassed quite a wealth of information. Endless depths of statistics compiled since the creation of both the NL and AL can be used for anything and everything. Additionally, new and more insightful sabermetrics emerge every year to give further insight on players and teams throughout the league. This is an issue for the accuracy of our predictions as it is possible to incorporate statistics from a time period that no longer reflects the actions of the current state of Major League Baseball and would only serve to add noise to our data set. For example, more and more batters are moving towards a three true batter outcome: the home run, the walk, and the strikeout. All other outcomes of an at bat, whether it be a single, double, etc., can usually be considered, to some extent, subject to the strength of the defense. These new statistical movements for batters is in gross rebuke of typical batting ideals of old. Similarly, the pitching movements of today vary from the past. Starting pitcher in the past would pitch more often (indicating less rest between games) and pitch more when they took their place on the mound (more innings pitched per outing) than those of today. This also coincides with new ideas on how to use relief pitchers. More relievers are coming in earlier in games and for different scenarios than ever before. No longer is a team's "closer" locked in to the ninth inning of a game, racking up saves every night; now, teams are starting to put out their best or most clutch pitcher in high value scenarios, whether it is in the middle of the sixth inning or the end of the ninth. These trends all serve to effect our data set and what data is pertinent in the modern league. Accounting for these trends and using the right amount of data along with the exact right data will be the challenge and our solution.

Finally, with the sheer amount of data available to us, another challenge will be to actually find what would work best for our predictions. With the bevy of information available, processing and determining what would be best is a continual work in progress. Finding the optimal statistics and properly weighting them to build the decision tree will take time and many more trial runs than we have already completed. We plan to optimize our decision tree by continually updating and perfecting our tree to hopefully give the most accurate outcome possible, even using different data sets from different sources if needed.

7 FUTURE PLANS

The immediate future plans of the project revolve around processing and evaluating the challenges presented in section six of this paper. There are a host of potential issues, from evaluating important statistical details necessary for accurate analysis to managing our data set. First, analyzing the difference between the leagues will be particularly interesting. Many questions surrounding the leagues will be interesting to consider. Do the statistics have a measurable effect in predicting the final major league standings, and do those outcomes have an effect on the playoff considerations of the teams? We plan to further adapt our data set and our decision tree as these problems are considered. Optimizing the amount of parameters for the tree as well as continuing to find the best fit statistics themselves will always be a work in progress and a part of our analysis. Fine tuning the decision tree algorithm to fully and accurately predict the postseason teams will be important in the success of our project.

Additionally, baseball metrics can give much more insight than just the outcomes of the regular season for postseason position. Player projections, effectiveness, and trends are all popular uses for statistics today. A few ideas and applications exist for the data we have compiled that would be an interesting extension of our current project and would be interesting work after the completion of this project. First, taking our analysis for the regular season, we would like to build a postseason predictor, complete with match-up analysis and a final World Series prediction. Other predictors we would consider are Hall of Fame performances or end of year player awards like the Cy Young.

2:40-2:44, 2:45-2:46	POW	salpteki	Alptekin, Samuel	Junior	Predicting the Outcome of Week 1 Collegiate Football Games
		jbeiter	Beiter, Jacob	Junior	
		sberning	Berning, Samuel	Junior	
		bshadid	Shadid, Benjamin	Junior	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Predicting the Outcomes of Week One College Football Games

Sam Alptekin
University of Notre Dame
salpteki@nd.edu

Jacob Beiter
University of Notre Dame
jbeiter@nd.edu

Sam Berning
University of Notre Dame
sberning@nd.edu

Ben Shadid
University of Notre Dame
bshadid@nd.edu

1 INTRODUCTION

Given teams' performance and recruiting data from the previous year, can a machine accurately predict the outcome of a Week 1 college football game? During the offseason, many college football fans wonder how good their favorite team will be next season. Due to the short length of collegiate football careers, the makeup of a team changes greatly from year to year. This makes it difficult to know exactly how good a team will be. For example, in the 2016 season,the Notre Dame football team had a 4-8 record, and in the next season, they had a 10-3 record. Week 1 games generally set the tone for the rest of the season, so by predicting the outcomes of these games, we may be able to get a sense of how good a team will be.

There are a few advanced statistics that attempt to describe the strength of a college football team beyond just wins and losses. In particular S&P+ brings together many different features, breaking team offense and defense into factors such as success rate, expected points per play, and field position advantages. By looking at these more abstracted statistics for each team and how they match up with each other head-to-head, we may be able to predict games with more accuracy than comparing overall strengths.

2 RELATED WORK

Predicting the outcome of sporting events is by no means a new concept; there have been many attempts to predict the outcome of collegiate football games in the last decade. Some models try to compare matchups in the testing set to matchups in the training set, much like Naive Bayes model would [5], while others compare the statistics of the two teams in the matchup against each other to determine a winner [1] [2]. We will be following the second approach.

Whether or not the researcher is attempting to predict collegiate basketball outcomes [3], NFL performance post-college [4], or the outcomes of collegiate football matches [1] [2] [5], they all consider these tasks to be fundamentally classification problems and attempt a Decision Tree model before any other. Most include SVMs, but only a few look at Neural Networks. Interestingly, Decision Trees have been shown to perform better than SVMs when predicting collegiate football game outcomes despite their relative simplistic nature [2].

The current research in this field for the most part makes use of raw, match-related data, such as "Passes Attempted, Passes Completed, Sacks, Interceptions..." [1]. We were unable to find any research that evaluated their models on the S&P+ data that we are using. As a result, the data we are using is more explicitly indicative of a head-to-head matchup where metrics such as "rushing offense" can be compared against "rushing defense" for the two teams.

3 PROBLEM DEFINITION

The first problem is a simple classification problem: Given the performance metrics, denoted by S&P+, and recruiting ranking of two teams, can we predict which team will defeat the other?

The second problem, and the question we looked at specifically for this milestone, was whether or not the recruiting ranking was an impactful metric in predicting the outcome of a Week 1 collegiate football game.

4 PROPOSED METHODOLOGY

This project is broken into two components, each of which will have its own methodology. The first half, to be completed by the first milestone, involves making a benchmark model based on the metrics from S&P+. This model is a simple ID3 decision tree, whose features are based on combinations of the components of S&P+. For instance, the rating of a team's passing offense in a given matchup is measured by subtracting the opposing team's passing defense score from the original team's passing offense score. Since each metric of S&P+ has been normalized, these different statistics can be directly subtracted from one another.

The differences (deltas) between statistics and their counterparts are used as features in the final decision tree. Each split of this decision tree is chosen to minimize entropy, but the attribute values themselves are continuous, so each split becomes binary with the qualifier of being above or below a certain threshold.

Through this, two models are constructed. The first, which incorporates only S&P+, serves as a benchmark for other proposed models. The second incorporates additional features, such as differences in recruiting scores, to improve on this benchmark. The success of each model can be evaluated by accuracy and similar metrics.

After this milestone is reached, development of more complicated models can begin. Initially, we proposed using SVMs. Going forward, the dimensionality of our project will grow, so the flaws in the SVMs' methodology are amplified. To address this, future

Conference'17, Washington, DC, USA
 978-x-xxxx-xxxx-x/YY/MM...$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

models will either incorporate Principal Component Analysis or be translated to a different type of model (i.e. Artificial Neural Network).

5 DATA AND EXPERIMENTS

5.1 Data Set

To describe the performance of football teams in the previous year, we pulled data from the S&P+ statistical model, which has overall rankings as well as more in-depth offensive and defensive metrics that seek to define and measure what makes a football team successful. Over the off-season that separates the season statistics and the games we are predicting, a new class of football players joins the program. We scraped scores and rankings of these recruiting class from 247Sports. Finally, to find which teams were playing each other and the outcomes of the games, we got schedule data from ESPN's website.

5.2 Data Cleaning and Integration

In order to input them to a model, these data sets must be brought together and structured by game. In other words, one tuple consists of the two teams playing, the game outcome, and each statistical measure we have for each team. This portion of the work took longer than expected, due to noisy and missing data. The different data sources often had different names listed for the same team (e.g. "Florida Intl" vs "FIU"), and these issues had to be corrected manually.

Our data sources only have data for the programs at the Football Bowl Subdivision (FBS) level, the highest level in NCAA football. However, many FBS teams play their first game of the season against teams in the Football Championship Subdivision (FCS), the division below FBS. We do not have data for these teams (and they wouldn't be interesting to predict, as the FBS team nearly always wins), and so any games involving an FCS team had to be removed from the data set, giving rise to concerns of data quantity – discussed more in Conclusions.

5.3 Experimental Settings

5.3.1 Decision Tree. We used the Python module 'scikit learn' to develop our Decision Tree. It was trained on the ID3 model using Information Gain as a splitting metric. For the initial stages of this project, we chose to use data from 2015 and 2016 as training data and data from 2017 was used for the testing set, but in future iterations we will split data randomly into testing and training data.

A significant part of our project is trying to determine whether the fact that a game is the first of the season is impactful for a team's performance. One way we are measuring this is through recruiting score, a metric that determines how good a team's influx of new players are. Our first experiment shows the difference in model performance when this recruiting score metric is added to the S&P+ metrics which focus solely on in game performance.

5.4 Evaluation Results

Benchmark Model (S&P+ only)
Confusion Matrix:

Result \ Predicted	Lose	Win	Total
Lose	TN = 4	FP = 19	N = 23
Win	FN = 6	TP = 45	P = 51
Totals	N' = 10	P' = 64	74

Accuracy	0.662162162162
F1 Score	0.782608695652
Precision	0.703125
Recall	0.882352941176

Proposed Model
Confusion Matrix:

Result \ Predicted	Lose	Win	Total
Lose	TN = 3	FP = 20	N = 23
Win	FN = 2	TP = 49	P = 51
Totals	N' = 5	P' = 69	74

Accuracy	0.702702702703
F1 Score	0.816666666667
Precision	0.710144927536
Recall	0.960784313725

As shown by the metrics listed above, the proposed model, which incorporates the other major features into S&P+, appears to have an overall better success and consistency rate. Given the limited size of the training and testing datasets, however, no stronger conclusions can be made about this yet.

6 CONCLUSIONS

Using only the data from week 1 to train and test our models turned out to be more restrictive than initially thought. Out of the hundreds of games that we were originally able to scrape, data was only usable for 166 total games over 3 years. Obviously, this is not a sufficient amount of data to answer the problem discussed above, and no more games will happen in the near future, so the end goal of this project will be modified. Rather than looking only at S&P+ and its closely related datasets to train a model with a large amount of data, this project will instead focus on greatly increasing the dimensionality of the problem at hand. That is, we will incorporate as many features into our model as possible, and there are many more data sets available that aim to describe a football team's success – e.g. ESPN's FPI, FootballOutsider's FEI, Bill Connely's Returning Production, etc.

Instead of working solely as a predictor for game outcomes, this model will also be able to determine which features are strongly correlated with winning or losing. We can then test a new model with these new features against our benchmark model to see which is better for predictions of full seasons (not just week 1 games). The end goal of this project would be to evaluate the reliability of each model over the course of a season. That is, as the S&P+ statistics from last year become less relevant to the current season, can other features be used to create a more reliable long-term model instead?

Due to the high dimensionality of this new problem, using an SVM might not be a viable method for evaluation. Instead, the model will likely be translated to a model more suited for Principal Component Analysis or another dimension reduction method.

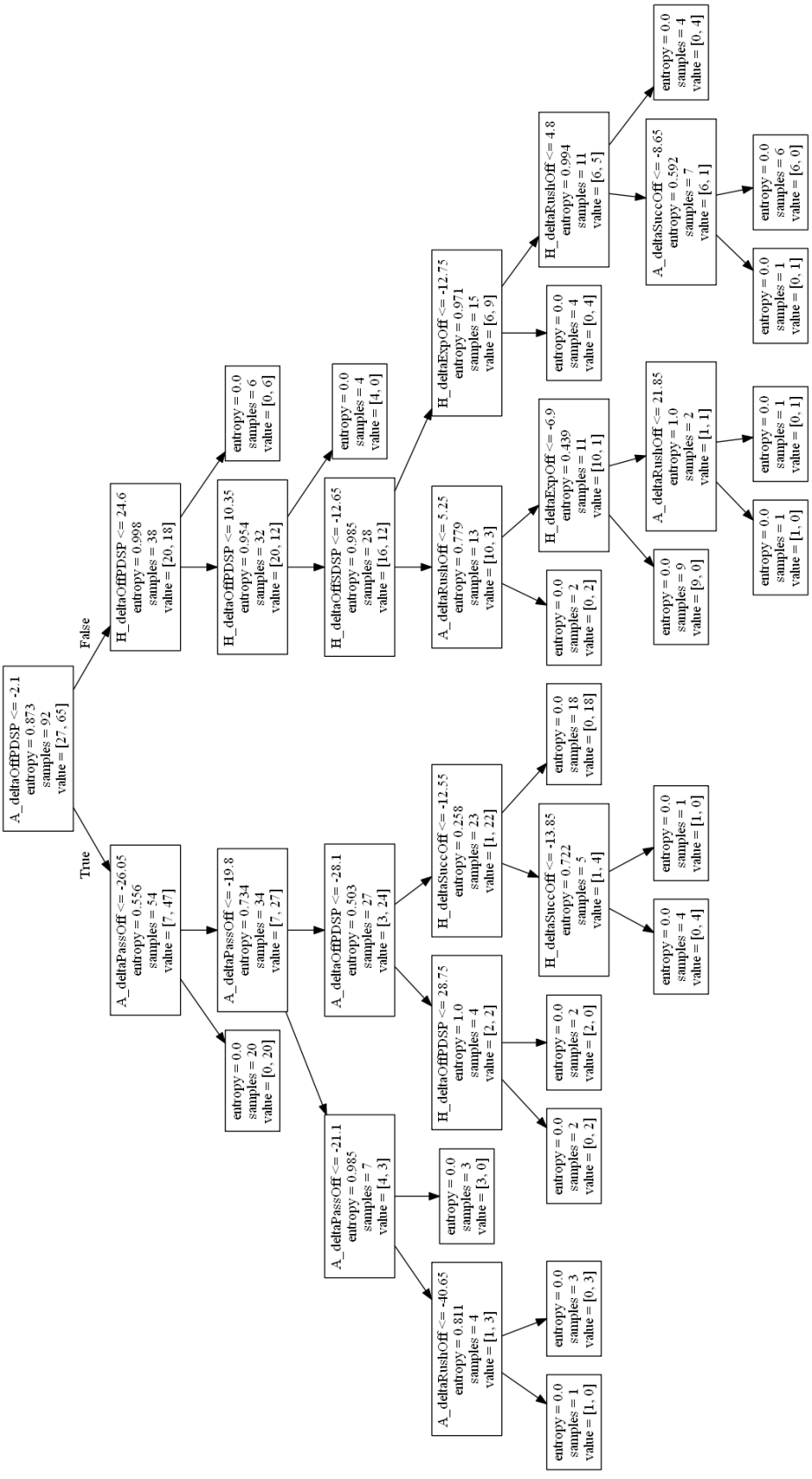


Figure 1: Constructed Decision Tree for the Benchmark Model

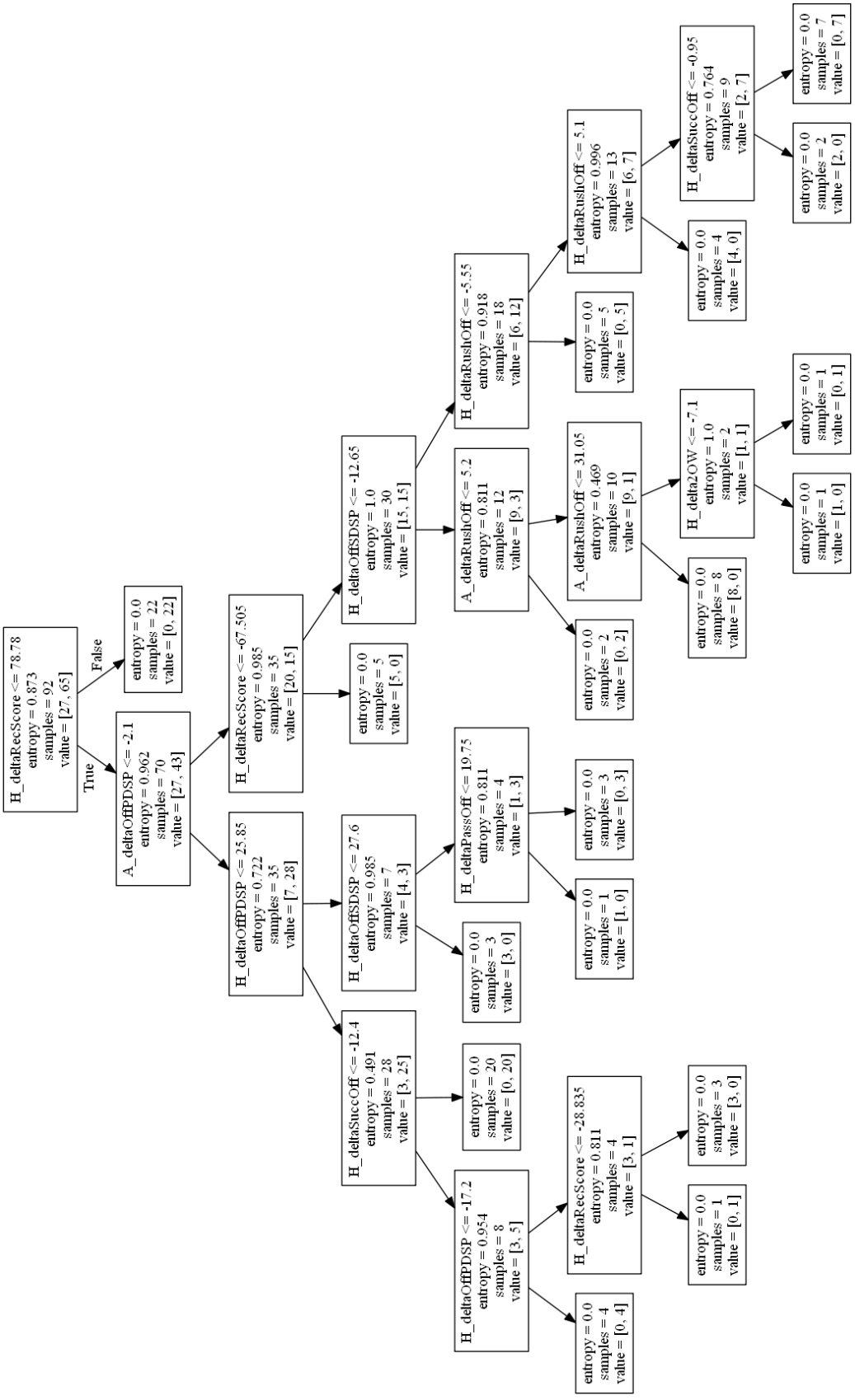


Figure 2: Constructed Decision Tree for the Proposed Model

REFERENCES

[1] Liu, Brian, and Patrick Lai. *Beating the ncaa football point spread.* (2010)

[2] Delen, Dursun, Douglas Cogdell, and Nihat Kasap. *A comparative analysis of data mining methods in predicting NCAA bowl outcomes.* International Journal of Forecasting 28.2 (2012): 543-552.

[3] Zimmermann, Albrecht, Sruthi Moorthy, and Zifan Shi. *Predicting college basketball match outcomes using machine learning techniques: some results and lessons learned.* arXiv preprint arXiv:1310.3607 (2013).

[4] Mulholland, Jason, and Shane T. Jensen. *Projecting the Draft and NFL Performance of Wide Receiver and Tight End Prospects.* CHANCE 29.4 (2016): 24-31.

[5] Leung, Carson K., and Kyle W. Joseph. *Sports data mining: predicting results for the college football games.* Procedia Computer Science 35 (2014): 710-719.

Week 1 Predictions

Sam Alptekin, Jacob Beiter, Sam Berning, Ben Shadid

Data Scraping, Cleaning, and Integration

- Scraped S&P+, recruiting score data, yearly schedules
- Cleaning
  - Teams had different names in different data sets
- Integration
  - Excel functions to combine team metrics based on schedules
- Challenges
  - FBS teams playing FCS teams
  - There are no S&P+ metrics for FCS teams

Model Development: Decision Trees

- ID3 model
- S&P+ and recruiting score
- Including recruiting score increased performance metrics

Benchmark:

Result \ Predicted	Lose	Win	Total
Lose	TN = 4	FP = 19	N = 23
Win	FN = 6	TP = 45	P = 51
Totals	N' = 10	P' = 64	74

Accuracy	0.662162162162
F1 Score	0.782608695652
Precision	0.703125
Recall	0.882352941176

Proposed:

Result \ Predicted	Lose	Win	Total
Lose	TN = 3	FP = 20	N = 23
Win	FN = 2	TP = 49	P = 51
Totals	N' = 5	P' = 69	74

Accuracy	0.702702702703
F1 Score	0.816666666667
Precision	0.71014927536
Recall	0.960784313725

Limitations of Current Models

- Range limited to 2015-2017 seasons
- Increasing dimensionality problem
  - SVMs
  - k-NN proposed model:

```
Confusion Matrix:
 [[13 10]
 [ 2 49]]
Accuracy: 0.837837837838
F1 Score: 0.890909090909
Precision: 0.830508474576
Recall: 0.960784313725
```

Future Work

- Incorporate Week 2 games
- Investigate more potential features
- Develop SVM model, incorporate dimension reduction
- Random forests

2:47-2:51, 2:52-2:53	PBC	amital	Mital, Aman	Junior	Predicting Breast Cancer Diagnosis from Tumor Measurements
		anemecek	Nemecek, Andrew	Junior	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Predicting Breast Cancer Diagnosis from Tumor Measurements

Aman Mital
University of Notre Dame
amital@nd.edu

Andrew Nemecek
University of Notre Dame
anemecek@nd.edu

ABSTRACT

In this paper, we describe initial results for our model predicting breast cancer from fine-needle aspirate measurements.

Keywords

Machine learning, data science, breast cancer, prediction, diagnosis, classification.

1. INTRODUCTION

Using data on Wisconsin breast cancer diagnosis, we plan to create a model which will classify a tumor as malignant or benign based on a series of measurements. Given the set of measurements, the model should output the class of the tumor.

We will be using a dataset provided by Kaggle from UC Irvine Machine Learning [3].

To create the model, we will divide the dataset of 570 patients into two pieces, one for training and one for evaluation. The dataset contains 30 features in addition to the class and a patient identifier, so we will first attempt to reduce the dimensionality of the data to simplify the model.

The most important metric in evaluating our model will be sensitivity i.e. measuring how many sick patients were correctly identified. Our reasoning for doing this is that it is more important to correctly identify malignant tumors than to avoid misidentifying benign tumors.

2. RELATED WORK

This problem (and this particular dataset) has been used in numerous data science and machine learning papers, initially to use machine learning to improve the accuracy of breast cancer diagnoses [4] and compare different classification models [1]. Wolberg in particular also does an excellent job explaining the meaning of the features in the dataset [4]. What the majority of the literature uses to evaluate their models is accuracy, the ratio of correct predictions to total predictions. This metric, while useful in many circumstances, may not be the best metric for a diagnostic tool, since most doctors and patients would prefer to err on the side of caution; this means that a good diagnostic tool should, if given a malignant tumor, always classify it as such. Additionally, although several papers describe how to achieve good accuracy with high optimization [1], [2], [4] and feature selection is an important component in this optimization [1], [2] the important question of how many features should be selected is hardly addressed. We attempt to resolve this issue by clustering the average Pearson correlation coefficient, as described in our section on Data Reduction.

3. MODEL

Our primary model will use linear SVM to classify each test instance. Visualization of the data shows that there is a clear dividing line in the data, which lends itself very well to SVM.

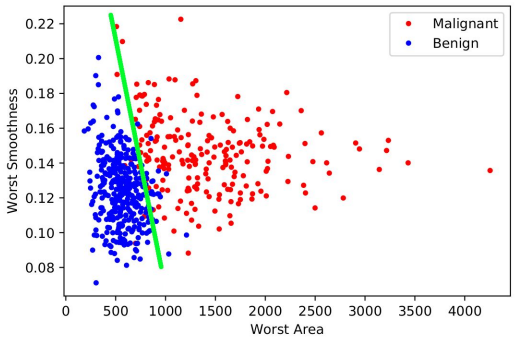


Figure 1: Two features plotted against each other, with a line demonstrating a plausible linear SVM.

We will additionally test other models, including a k-Nearest Neighbors classifier and possibly decision trees to verify which model performs the best.

4. FEATURE SELECTION

The dataset contains 30 different features, many of which are obviously correlated such as mean radius and mean area. Methods exist to select features based on univariate statistical tests (Univariate Feature Selection) and other methods can reduce the dimensionality of the data by extracting features (such as Principal Component Analysis). However, all of these rely on a manual threshold for the score/number of features to select. In order to select a good subset of the features, we will attempt to cluster the maximum absolute value of the Pearson correlation coefficient (normalized covariance) for each feature to search for a clear dividing line between relevant and irrelevant features. The reason we want to select features rather than simply use PCA is that PCA does not impact data collection. If we are able to select a subset of the features that still provides good results, this may simplify the entire diagnostic process by eliminating unnecessary measurements.

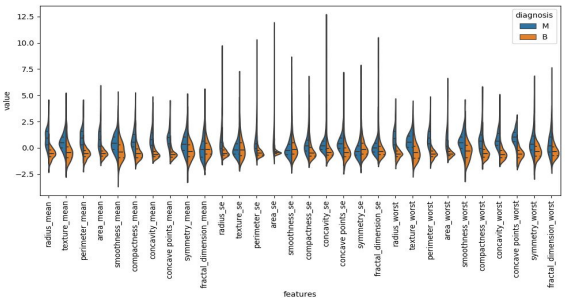


Figure 2: Distribution of each feature, separated by label

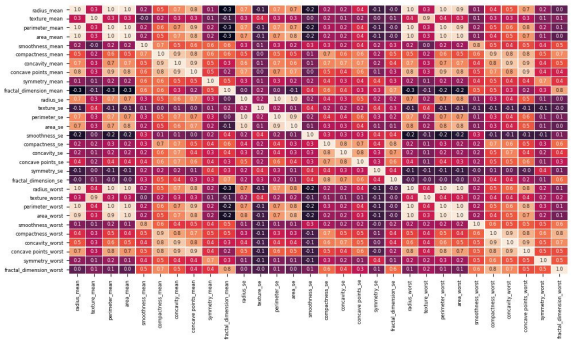


Figure 3: Heatmap of feature correlation

5. EVALUATION

As mentioned in the introduction, the primary metric for a diagnostic model should be sensitivity. Given the following parameters:

A - The number of correctly identified malignant tumors

B - The number of incorrectly identified malignant tumors

C - The number of incorrectly identified benign tumors

D - The number of correctly identified benign tumors

		Predicted	
		Malignant	Benign
Actual	Malignant	A	B
	Benign	C	D

Table 1: Confusion matrix for classifying tumors

The equation for sensitivity can be written as:

$$sensitivity = \frac{A}{A+B} \quad (1)$$

The second primary metric we will use is accuracy, written:

$$accuracy = \frac{A+D}{A+B+C+D} \quad (2)$$

For our model to be considered "good", it should maximize sensitivity, followed by accuracy.

6. PRELIMINARY RESULTS

A preliminary model was created using a linear SVM. For the initial model, the data was split by a convenience sample, with the first 500 instances of the data set being used for training and the last 69 instances being used as testing data. Multiple kernel functions (linear, rbf, sigmoid, and poly) were tested with linear giving the best results. Linear models were run with C-values of 10, 1, 0.1, and 0.01.The confusion matrices for each model can be seen in Tables 2 through 5.

From the confusion matrices, it can be seen that a C-values of 10 and 1 produce identical results. A C-value of 0.01 produced the worst results with accuracy being the lowest of the four models (.9565). The C-value of 0.1 produced the best results. Its accuracy was as high as the other models (.9710), but it increased sensitivity, resulting in a sensitivity of 1.0.

		Predicted	
		Malignant	Benign
Actual	Malignant	16	1
	Benign	1	51

Table 2: Linear SVM. C = 10.0

		Predicted	
		Malignant	Benign
Actual	Malignant	16	1
	Benign	1	51

Table 3: Linear SVM. C = 1.0

		Predicted	
		Malignant	Benign
Actual	Malignant	17	0
	Benign	2	50

Table 4: Linear SVM. C = 0.1

		Predicted	
		Malignant	Benign
Actual	Malignant	16	1
	Benign	2	50

Table 5: Linear SVM. C = 0.01

7. CHALLENGES AND PROPOSED SOLUTION

One of the challenges we are facing is selecting the best features. In our current model, all 30 features are used. We plan on utilizing LASSO feature selection and Clustering Pearson Coefficient to select the best features.

8. PLAN FOR THE FUTURE

There are three main goals we plan to accomplish in the coming months. First, we plan to reduce the number of features. Currently, all 30 of the features of the dataset are being taken into account. Second, we plan to expand the evaluation of our model. We plan on using k-fold cross validation, which will give a better idea of how successful the model is than the convenience sample holdout validation currently being used. Last, we plan on implementing other classification models, such as k-NN. Each model will be evaluated to determine which model is the most effective.

9. REFERENCES

[1] Gouda I. Salama , M.B.Abdelhalim, and Magdy Abd-elghany Zeid. Breast Cancer Diagnosis on Three Different Datasets Using Multi-Classifiers. International Journal of Computer and Information Technology, (2012) 1:1.

[2] Hussein A. Abbass. An evolutionary artificial neural networks approach for breast cancer diagnosis. Artificial Intelligence in Medicine, (2002) 25.

[3] University of California Irvine Machine Learning Repository. “Breast Cancer Wisconsin Diagnostic”. https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29.

[4] W.H. Wolberg, W.N. Street, D.M. Heisey, and O.L. Mangasarian. Computerized breast cancer diagnosis and prognosis from fine needle aspirates. Archives of Surgery, (1995) 130:511-516.

Predicting Breast Cancer Diagnosis from Tumor Measurements

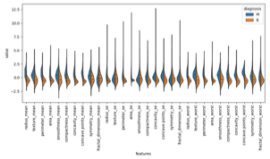
Aman Mital & Andrew Nemecek

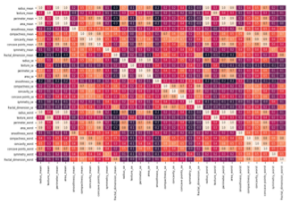
Introduction

- Primary objective: Predict the presence of breast cancer given a set of fine-needle aspirate measurements
- Secondary objective: Limit the number of features required to simplify the diagnostic process
- Data set: Wisconsin Breast Cancer Diagnostic data (WDBC)
  - Cell measurements from samples from a breast "mass"

Feature Reduction

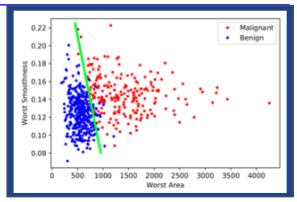
- Dataset started with 30 features
- Many features show correlation
- Want to reduce the number of features as much as possible without decreasing the effectiveness of the model





Model

- We will be using an SVM because the classes tend to be split linearly between features (see right)
- Will additionally use k-NN and possibly Decision Trees to compare
- Primary evaluation metric: Sensitivity (predicted positive / actually positive)
  - More important to find all cases of malignant tumors than to weed out false positives
- Accuracy as a secondary metric



A plot of two features against each other, with a possible SVM boundary shown in green. The features were among the "best three features" (see W.H. Wolberg, W.N. Street, D.M. Heisey, and O.L. Mangasarian. Computerized breast cancer diagnosis and prognosis from fine needle aspirates. Archives of Surgery, (1995) 130:511-516.)

Initial Results With Linear SVM

C = 10.0 **Sensitivity = 0.94** **Accuracy = 0.97**		Predicted	
		Malignant	Benign
Actual	Malignant	16	1
	Benign	1	51

C = 1.0 **Sensitivity = 0.94** **Accuracy = 0.97**		Predicted	
		Malignant	Benign
Actual	Malignant	16	1
	Benign	1	51

All > 95% accuracy!

C = 0.1 **Sensitivity = 1.0** **Accuracy = 0.97**		Predicted	
		Malignant	Benign
Actual	Malignant	17	0
	Benign	2	50

C = 0.01 **Sensitivity = 0.94** **Accuracy = 0.96**		Predicted	
		Malignant	Benign
Actual	Malignant	16	1
	Benign	2	50

Plan for the Future

- Reduce the number of features used in the model
  - Currently using all 30
  - LASSO feature selection
  - Clustering Pearson Coefficient (normalized covariance)
- Expand the evaluation of our model
  - k-fold Cross Validation
- Experiment with more models to find which works the best
  - k-NN

2:54-2:58, 2:59-3:00	DPH	wbadart	Badart, William	Senior	Determining predictors of H-1B salary and approval
		lduane	Duane, Luke	Senior	
		wyu1	Yu, Wenhao	Non-Degree Student	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Determining Predictors of H-1B Salary and Approval

Milestone Report

Wenhao Yu
University of Notre Dame
South Bend, Indiana
wyu1@nd.edu

Luke Duane
University of Notre Dame
South Bend, Indiana
lduane@nd.edu

Will Badart
University of Notre Dame
South Bend, Indiana
wbadart@nd.edu

ABSTRACT

The paper presents the initial findings of the H-1B visa program analysis project for CSE-40647/60647.

ACM Reference format:
Wenhao Yu, Luke Duane, and Will Badart. 2018. Determining Predictors of H-1B Salary and Approval. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 3 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

An H-1B visa is a visa given out to foreign workers filling a specialty occupation for an American company. The visa lasts three years, but is renewable for up up to six years. Just last year, in 2017, almost 350,000 foreign workers applied and a little under 200,000 were approved.

The H-1B lottery is a laborious and complex process for both large companies bringing in thousands of migrant employees and small ones onboarding only a couple. A tool which highlights the important features that support H-1B approvals could be a vital strategic asset for these companies. Lots of data exists in this domain, but to integrate it and perform meaningful analysis is beyond the capabilities of companies without established data science practices. We plan to produce a model that shows what features are most valuable in regards to H-1B workers' salaries and approval.

2 RELATED WORK

In April of 2017, Glassdoor published an article analyzing the salaries of H-1B immigrants and comparing them to those of domestic workers in similar roles and fields. While the report does not attempt to model H-1B workers' salaries based on other features, it offers a comprehensive statistical analysis of their pay. [1]

3 PROBLEM DEFINITION

How can we predict the approval status of a given H-1B via application? What tangential analyses provide tangible business value for companies sponsoring H-1B visas? How would the salary range change based on a given occupation?

4 PROPOSED METHODOLOGY

The sheer volume of data available to train our model necessitated that we perform a number of initial analyses before constructing the model. For these initial analyses, we chose to calculate a number of descriptive statistics over our primary data set[2] as well as a couple visualizations to quickly understand the distributions of key features. We have already identified a few outliers in the primary dataset (in particular, in the PREVAILING_WAGE feature) and cleaned our data before producing our initial findings.

After the data cleaning and description phase, we began to train our predictive models. Our baseline models are Naive Bayes model and Decision Tree model, which attempt to predict the status of an H-1B application and the salary range.

We used k-fold Cross Validation (with k=5) to make sure every available data point has been used as training and testing. Therefore, the presented accuracy of each model represents the mean of the accuracies from each cross-validation iteration.

Additionally, we will use our findings from the decision tree construction to create a random forest to predict approval status, which we expect to have the best performance. To supplement these findings and support a numerical target value such as wage, we will train a regression to model.

If time permits, we're curious to implement a neural network as a classifier, and pit it against our best performing model of those described above.

5 DATA AND EXPERIMENTS

5.1 Datasets

(1) One of the largest freely available datasets on H-1B applications comes from kaggle.com. It contains over 3 million records and tracks 10 different features per application[3]. This data covers applications roughly between 2012 and 2016.
(2) Another key dataset comes from the Foreign Labor Certification Data Center. Its data is organized by year, spanning from 2001 to 2007[4].
(3) OFLC's annual reports also provides a lot of program information and data. Although it is not raw data, it disclosures cumulative quarterly and annual releases of program to assist with external research and program evaluation[5].

[1] Glassdoor Comparison on H-1B Visa Salaries vs US Workers

[2] See 5.1 Datasets
[3] See kaggle.com/asavla/h1-visa/data
[4] See flcdatacenter.com/CaseH1B.aspx
[5] Please follow https://www.foreignlaborcert.doleta.gov/pdf/OFLC_Annual_Report_FY2016.pdf

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

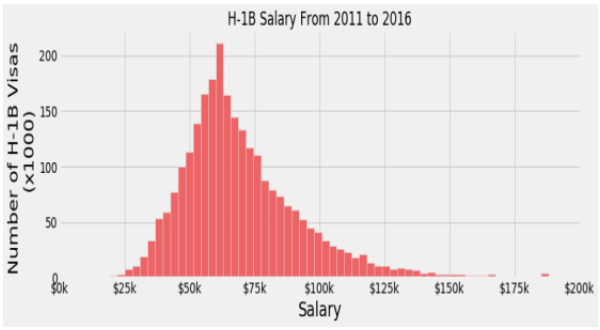


Figure 1: Salary Distribution

5.2 Data Summary

This subsection presents a preliminary description of dataset (1), the Kaggle dataset described in section 5.1 Datasets.

Figure 1 shows the salary distribution.There are 5 Outliers in the original data. The average of salary is 72,221, the median of salary is 66,602, and the standard deviation is 24,704.

Table 1 shows the frequency of each value of the CASE_STATUS feature, the column which labels whether an application was approved, according to the below caveats distributed with the dataset:

> The CASE_STATUS field denotes the status of the application after LCA processing. Certified applications are filed with USCIS for H-1B approval. CASE_STATUS: CERTIFIED does not mean the applicant got his/her H-1B visa approved, it just means that he/she is eligible to file an H-1B.

Table 1: Approval Status Classes

Class Name	Frequency
CERTIFIED	914,251
NON-CERTIFIED	134,325

Table 2: Salary Classes

Class Name	Frequency	Range
Very High	90,004	[104042,E99)
High	182,226	[79331,104042)
Middle	361,845	[59155,79331)
Low	181,648	[28963,59155)
Very Low	98,528	[12584,28963)

Table 1 shows that most H-1B applications are certified (note: this does not mean they are accepted).

Table 2 shows the frequency of each value of the WAGE feature. We have five categories and the columns show the frequency and salary range of each class.

5.3 Experimental Settings

We consider four features in our classification of each application: EMPLOYER, JOB TITLE, LOCATION, SALARY. The label is CASE_STATUS, which has two categories: CERTIFIED and NON-CERTIFIED.

About the salary level classification, we use three features :EMPLOYER, JOB TITLE, LOCATION. The label is SALARY, which has five categories: VERY HIGH, HIGH, MIDDLE, LOW, VERY LOW.

5.4 Evaluation Results

In this section, we evaluated the Decision Tree model and Naive Bayes model for approval status and salary level. The results are as follows:

Table 3: Naive Bayes Confusion Matrix for Approval

	Predicted Approved	Predicted Denied
Approved	888435	25814
Denied	98423	35901

Accuracy: 0.91409770402 Specificity: 0.26727167

Table 4: Decision Tree Confusion Matrix for Approval

	Predicted Approved	Predicted Denied
Approved	905932	8317
Denied	81756	52568

Accuracy: 0.881516343609 Specificity:0.3913522

Table 5: Naive Bayes Salary Prediction Accuracy

Class	Correct	Wrong
Very High	58282	31722
High	93270	88956
Middle	276983	84862
Low	96456	85198
Very Low	71492	27063
Total Accuracy	65.2	

Table 6: Decision Tree Salary Prediction Accuracy

Class	Correct	Wrong
Very High	67667	22337
High	137068	45158
Middle	317643	44202
Low	151558	30090
Very Low	95306	3222
Total Accuracy	84.1	

We can reasonably see that the Decision Tree model performs better than the Naive Bayes model. Besides, although the accuracy is very high, the specificity is very low. It means it is a great model to determine the person is certified, but it is not accurate to determine the person is non-certified.

6 CONCLUSIONS

REFERENCE

[1] The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011-3



Will Badart Wenhao Yu Luke Duane

Determining Predictors of **H-1B** Salary and Approval

Introduction

What is the H-1B Visa?

- A visa in the US under the INA
- Given out to foreign workers at US compan
- In 2017, 350,000 individuals applied and 200,000 were approved.

Problem

- How can we predict the approval status of a given H-1B via application? What tangential analyses provide tangible business value for companies sponsoring H-1B visas?
- How would the salary range change based on a given occupation?

Methodology

Table1: Approval Class Status

Class Name	Frequency
Certified	914,251
Non-Certified	134,325

Outlier	5
Mean	72,221
Median	66,602
S-D	24,704

Table2: Salary Class

Class Name	Frequency	Range
Very High	90,004	[104,042 , E99)
High	182,226	[79,331 , 104042)
Middle	361,845	[59,155 , 79,331)
Low	181,648	[28,963 , 59,155)
Very Low	98,528	[12,584 , 28,963)

Evaluation

1. Approval Status

Decision Tree model:
Confusion Matrix:
| TP 905,932 | FN 8,317 |
| FP 81,756 | TN 52,568 |
Accuracy: 0.914097704027
Precision: 0.917224872632
Recall: 0.990902915945
F1: 0.952641438702

Naïve Bayes model:
Confusion Matrix:
| TP 888,435 | FN 25,814 |
| FP 98,423 | TN 35,901 |
Accuracy: 0.881516343609
Precision: 0.900266299711
Recall: 0.97176480368
F1: 0.934650180132

2. Salary Level

Decision Tree model:

Class Name	Right.	Wrong
Very High	67,667	22,337
High	137,068	45,158
Middle	317,643	44,202
Low	151,558	30,090
Very Low	95,306	3,222

Accuracy: 0.84139038

Naïve Bayes model:

Class Name	Right.	Wrong
Very High	58,282	31,722
High	93,270	88,956
Middle	276,983	84,862
Low	96,450	85,198.
Very Low	71,492	27,063

Accuracy: 0.65242149

3:01-3:05, 3:06-3:07	AFG	mgianni1	Giannini, Mark	Graduate-Masters	It's All Funds & Games - Predicting Kickstarter Success
		ptinsley	Tinsley, Patrick	Graduate-Masters	
		btunnell	Tunnell, Brian	Graduate-Masters	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

It's All Funds & Games

Predicting Kickstarter Success

Mark Giannini
University of Notre Dame
mgianni1@nd.edu

Patrick Tinsley
University of Notre Dame
ptinsley@nd.edu

Brian Tunnell
University of Notre Dame
btunnell@nd.edu

1. Introduction

1.1 Project Plan

For our semester project, we decided to test our ability to predict whether or not a given Kickstarter campaign will be successful. In order to be deemed a success, the proposed campaign needs to meet or exceed the funding goal proposed by the initial author by a predefined deadline; anyone can contribute as long as the campaign is still active. In the context of our project, each instance in our data set has a unique project ID and fourteen features; these include project name, project description, keywords, financial goal in US dollars, the project deadline and the number of backers contributing to and supporting the project. Using sentiment analysis and other logistic regression techniques we have learned in previous classes, we plan to predict the binary final_status field, which indicates a successful project (1) or a failed attempt (0).

1.2 Data Sources

Initially, we planned to crawl the data from the Kickstarter website ourselves. However, upon browsing a plethora of Kaggle competitions, we found a pre-built data set that contains all our fields of interest. The supplied data has 108,129 rows, each corresponding to a project proposal submitted between May 2009 and May 2015. Each instance has the following features: Project ID, Name, Description, Funding Goal, Project Keywords, Disable Communication, Country, Currency, Deadline Date, Date Created, Date Launched, State Changed At, Launched At, Number of Backers and finally, the targeted response variable, Final Funding Status.

1.3 Proposed Evaluation

To evaluate our models predictive power, we plan on splitting our data into two sets. The first partition will be the training set, and it will be used to build and train our model. The second partition will be the testing set, and it will be used to validate the model. If we split the data 66.6%-33.3% respectively, the training set will have 72,446 rows, and the testing set will have 35,683 rows. By withholding a subset of the full data set, we have the power to test our final model on unseen data, which can be used to evaluate estimator performance; this technique also helps to avoid overfitting, producing a more generalized model capable of predicting the success of future Kickstarter projects.

, .

http://dx.doi.org/10.1145/

2. Related Work

Given the nature of Kaggle, there are several other kernels or projects that deal with predicting the success of Kickstarter campaigns. However, since the service is competition-based, other submissions are blocked. Nevertheless, many members of the data science community have tried their hand at this task. One such attempt included mostly spatial and geographical features, such as, country, state, and county; that work can be seen here. Another attempt focused on the categories of the proposed project, such as music, theatre, and art; that work can be seen here. In the latter attempt, the author found accuracy rates of 63%, 68%, and 65% for kNN, random forest, and logistic regression, respectively. For our project, we hope to surpass 80% accuracy.

3. Problem Definition

The problem of determining successful Kickstarter campaigns is inherently a classification problem; either the campaign succeeds or fails. In the context of the data, this is described in the final_status field, which contains only zeros and ones for failures and successes, respectively. Additionally, since the target variable is binary, logistic regression techniques will be used throughout the model evaluation process.

4. Proposed Methodology

To tackle this problem, we decided to apply several sklearn classifiers at the training data. The architecture for sklearn is very similar across models, which makes fitting the models very easy. Once the data has been split up into training and testing, constructing the model is as simple as calling a fit function on X_train. Listed below are the classifiers we built on the data.

- Decision Tree
- Logistic Regression
- Random Forest (Gini & Entropy)
- Naive Bayes (Bernoulli & Gaussian)

In order to evaluate the models, we looked at the following metrics as provided by the sklearn.metrics package: Confusion Matrix, Accuracy, Precision, Recall, F-1 Score, Classification Report, AUC, and Zero-One Loss. To keep our code clean, we wrote a function called evaluate_model that calculated each of these metrics given a vector of predicted labels on the X_test data and the actual labels (y_test).

5. Data & Experiments

5.1 Data Set

As stated before, the fourteen features of this data set differ in type. The name, description, and keywords fields are string objects, disable_communication is Boolean, goal is a float, country and currency are nominal, and the rest are integers (int64). For the first part of this project, we did not worry ourselves with the string objects, though they will be used later to cluster projects together. For the sake of dimensionality reduction, we removed the created_at and currency fields. The created_at column was removed since the overlap between created_at and launched_at was very strong; additionally, since the time of creation is hidden from the public, we do not wish to use it in predicting in real-world contexts. The currency column was essentially a proxy column for country, meaning it added little new information, so it was removed as well.

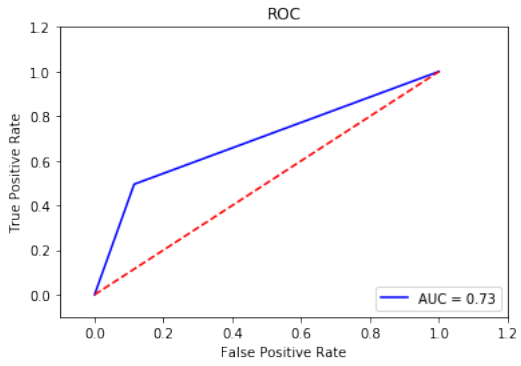
After stripping away the five columns mentioned previously, we entered the feature engineering phase. The date-related columns, originally integers representing time in the UTC format, were converted to Datetime objects using the pandas to_datetime function. From these objects, we extracted the hour, day, month, and year for the launch and deadline of the proposed campaign. After label-encoding these fields with the pandas get_dummies function, we encoded the country field as well. The new data set had 140 features, compared to the initial 14.

5.2 Experiments

5.2.1 Decision Tree

For the first model, we decided to fit a decision tree to the training data. Using a grid-search package provided by sklearn, we found that the optimal decision tree consisted of a Gini index criterion and max depth of 9. A visualization rendered by the graphviz package can be seen by running the code; the tree is rather large, and is not legible at this size. The root node of the tree separates the data based on the goal field. According to this model, projects that have lower goals are more likely to succeed, which matches up with our intuition. Below are the evaluation metrics and ROC curve for the decision tree model.

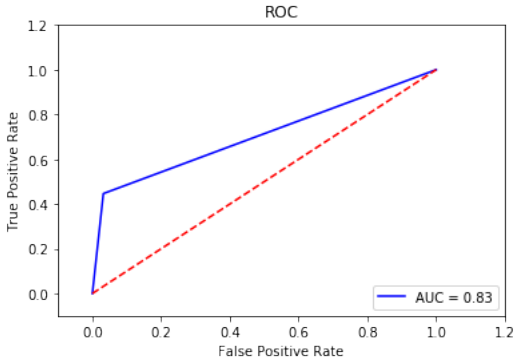
Accuracy	Precision	Recall	F-1	AUC
0.75985	0.49497	0.67057	0.56954	0.72907



5.2.2 Logistic Regression

For the second model, we decided to fit a logistic regression model. As we can see, except for the precision metric, the regression outperformed the decision tree in the previous section. Below are the evaluation metrics and ROC curve for the logistic regression model.

Accuracy	Precision	Recall	F-1	AUC
0.80063	0.44643	0.86852	0.58973	0.82787



5.2.3 Random Forest

We fit two Random Forest models using the python package sklearn and the function RandomForestClassifier. The Random Forest model is an ensemble method for classification, and can provide better predictive power than a simple Decision Tree model. Instead of relying on one decision tree for prediction, a Random Forest model constructs multiple decision trees and uses the mode of the classes for classification. The first Random Forest model was fit using Gini Impurity as the split criteria, and the second used Entropy as the split criteria. Gini Impurity measures the probability that a randomly chosen element would be incorrectly labeled if it were labeled based on the distribution of labels within the data set. The following equation is used to calculate Gini Impurity for a set of items with J classes where pi is the proportion of items with class i in the data set.

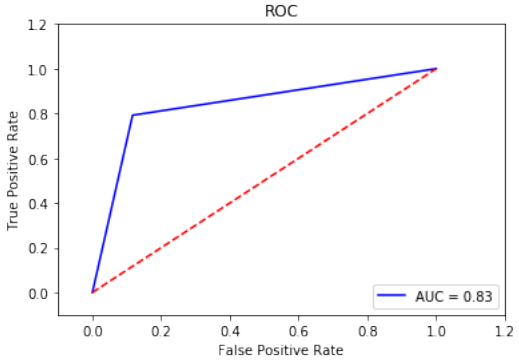
$$I_G(P) = 1 - \sum_{i=1}^{J} p_i^2 \tag{1}$$

Entropy, alternatively is used in the calculation of Information Gain, which chooses the split that results in the purest daughter nodes. Entropy is defined as the following.

$$H_T = -\sum_{i=1}^{J} p_i log_2(p_i) \tag{2}$$

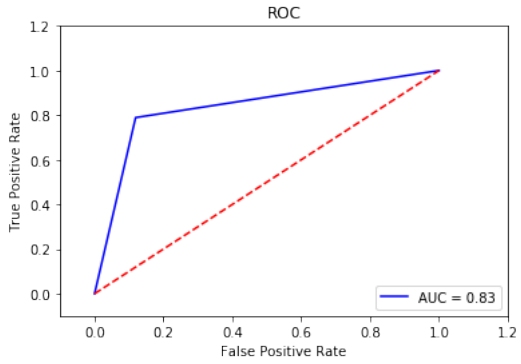
Information Gain is calculated by subtracting the weighted sum of Entropy(children) from Entropy(parent). The two Random Forest models had very similar results upon fitting them to the training data. When using Gini impurity as the split criteria, we obtained the results below.

Accuracy	Precision	Recall	F-1	AUC
0.85387	0.79193	0.76212	0.77674	0.83095



When using entropy as split criteria, we obtained similar evaluation metrics. However, this was a slightly less accurate method for creating our Random Forest, as seen in the results below.

Accuracy	Precision	Recall	F-1	AUC
0.85135	0.78905	0.75781	0.77312	0.828063

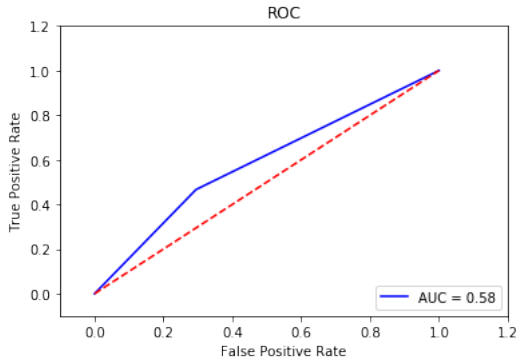


5.2.4 Naive Bayes

We also chose to fit a Naive Bayes model with a Gaussian distribution as well as a Bernoulli distribution for the numeric variables. The Naive Bayes model is a learning algorithm based on applying Bayes Theorem along with the naive assumption that all data features are independent. The GaussianNB model within the python package sklearn seemed like a good initial model to examine because of its core assumption that the features are normally distributed. Further, the BernoulliNB model was a good juxtaposition because of its ability to work well with binary or Bernoulli distributed variables. Upon testing, the Gaussian model did not perform as well as the Bernoulli model. This makes sense with the large number of binary variables within the numeric feature data set. The metrics for each of the classifiers can be seen below.

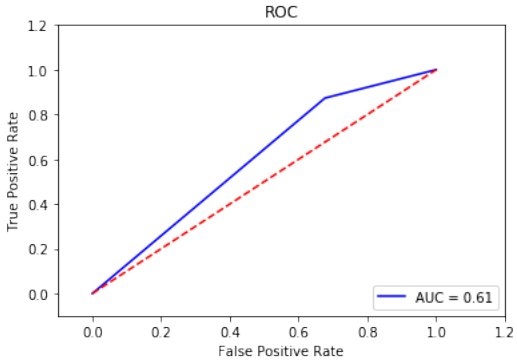
Naive Bayes - Bernoulli

Accuracy	Precision	Recall	F-1	AUC
0.62870	0.46651	0.428056	0.44645	0.58235



Naive Bayes - Gaussian

Accuracy	Precision	Recall	F-1	AUC
0.49995	0.87322	0.37894	0.52852	0.61133



For our final project, we intend to explore fitting a MultinomialNB model performs for our features requiring text classification as well as an out-of-core Naive Bayes model. This model would require pulling from each of the previously discussed models and combining each of their partial model fits into a full model capable of efficiently handling the different feature types within this set.

6. Conclusions

A. Appendix Title

Appendix, if needed.

Acknowledgments

Acknowledgments, if needed.

It's All Funds & Games
Milestone Presentation

Brian Tunnell, Mark Giannini, Patrick Tinsley

Motivation (Refresher)

- We want to be able to predict the ultimate fate of a Kickstarter campaign
  - final_status column → failure = 0, success = 1
- Data comes from Kaggle (108,129 projects)
- Train-test Split: 66.6% - 33.3% (via train_test_split function from sklearn.model_selection)
  - Training Data: 72,446 projects
  - Testing Data: 35,683 projects

Feature Engineering / Data Integration

- Here are some functions that we found very useful:
  - to_datetime (pandas library): converts unix time (1/1/1970) to datetime object
    - created_at, launched_at, state_changed_at, deadline, ...
  - get_dummies (pandas library): dummy codes categorical data
    - country, launch_hour, launch_day, launch_month, deadline_day, ...
- Expected Duration = deadline - launched_at
- Actual Duration = state_changed_at - launched_at

Data Transformation Chart (Original)

```
Columns:
['name' 'desc' 'goal' 'keywords' 'disable_communication' 'country'
 'currency' 'deadline' 'state_changed_at' 'created_at' 'launched_at'
 'backers_count' 'final_status']
Data Types:
name                    object
desc                    object
goal                   float64
keywords                object
disable_communication     bool
country                 object
currency                object
deadline                 int64
state_changed_at         int64
created_at               int64
launched_at              int64
backers_count            int64
final_status             int64
dtype: object
Shape:
(108129, 13)
```

Data Transformation Chart (Intermediate)

```
Columns:
['name' 'desc' 'goal' 'keywords' 'disable_communication' 'country'
 'backers_count' 'final_status' 'expected_duration' 'actual_duration'
 'launch_year' 'launch_month' 'launch_day' 'launch_hour' 'deadline_year'
 'deadline_month' 'deadline_day']
Data Types:
name                    object
desc                    object
goal                   float64
keywords                object
disable_communication     bool
country                 object
backers_count            int64
final_status             int64
expected_duration        int64
actual_duration          int64
launch_year              int64
launch_month             int64
launch_day               int64
launch_hour              int64
deadline_year            int64
deadline_month           int64
deadline_day             int64
dtype: object
Shape:
(108129, 17)
```

Data Transformation Chart ('Final')

```
Columns:
['name' 'desc' 'goal' 'keywords' 'disable_communication' 'backers_count'
 'final_status' 'expected_duration' 'actual_duration' 'country_AU'
 'country_CA' 'country_DE' 'country_DK' 'country_GB' 'country_IE'
 'country_NL' 'country_NO' 'country_NZ' 'country_SE' 'country_US'
 'launch_hour_0' 'launch_hour_1' 'launch_hour_2' 'launch_hour_3'
 'launch_hour_4' 'launch_hour_5' 'launch_hour_6' 'launch_hour_7'
 'launch_hour_8' 'launch_hour_9' 'launch_hour_10' 'launch_hour_11'
 'launch_hour_12' 'launch_hour_13' 'launch_hour_14' 'launch_hour_15'
 'launch_hour_16' 'launch_hour_17' 'launch_hour_18' 'launch_hour_19'
 'launch_hour_20' 'launch_hour_21' 'launch_hour_22' 'launch_hour_23'
 'launch_day_1' 'launch_day_2' 'launch_day_3' 'launch_day_4' 'launch_day_5'
 'launch_day_6' 'launch_day_7' 'launch_day_8' 'launch_day_9'
 'launch_day_10' 'launch_day_11' 'launch_day_12' 'launch_day_13'
 'launch_day_14' 'launch_day_15' 'launch_day_16' 'launch_day_17'
 'launch_day_18' 'launch_day_19' 'launch_day_20' 'launch_day_21'
 'launch_day_22' 'launch_day_23' 'launch_day_24' 'launch_day_25'
 'launch_day_26' 'launch_day_27' 'launch_day_28' 'launch_day_29'
 'launch_day_30' 'launch_day_31' 'launch_month_1' 'launch_month_2'
```

Classifier Evaluation

Model	Accuracy	Precision	Recall	F-1 Score	AUC
DT (Gini)	0.75985	0.49497	0.67057	0.56954	0.72907
LogReg	0.80063	0.44643	0.86852	0.58973	0.82787
RF (Gini)	0.85387	0.79193	0.76212	0.77674	0.83095
RF (Entropy)	0.85135	0.78905	0.75781	0.77312	0.82806
NB (Bernoulli)	0.62870	0.46651	0.42805	0.44645	0.58235
NB (Gaussian)	0.49995	0.87322	0.37894	0.52852	0.61133

Progress

- Right now, we have classifiers built for the data **without** the columns that include text
- Ultimately, we want to cluster the projects into categories based on the text columns (name, description, keywords)
- We can then use the cluster/category as another feature
  - We expect to have to use stop-words, stemming, tokenization, and tf-idf
  - Tf-idf = “term frequency - inverse document frequency” : measures how significant a word is document or collection of documents

3:08-3:12, 3:13-3:14	MPT	xwang41	Wang, Xueying	Graduate-PhD.	Misread-Proof Temporal Fact Extraction
		tzhao2	Zhao, Tong	Graduate-PhD.	

Introduction	Provide context and motivation. What questions are being addressed? Why are these questions interesting or important?
Grade (1-10):	
Related Work:	What other methods have addressed these or similar questions? How do these methods differ from your method?
Grade (1-10):	
Solution/Method:	What did you do? What tools and techniques did you use? Was any innovation attempted?
Grade (1-10):	
Data and Experiments:	What data did you use? Are your experimental methods reliable? What preprocessing was done the data?
Grade (1-10):	
Evaluation and Results:	Did you properly evaluate your experiments? Did you test for statistical significance? Do your conclusions match your results?
Grade (1-10):	
Writing/Presentation Quality	Clarity of writing, speaking, visuals, organization, and grammar.
Grade (1-10):	

Questions/constructive comments:

Misread-Proof Temporal Fact Extraction

Xueying Wang, Tong Zhao, Meng Jiang
University of Notre Dame, Notre Dame, Indiana, 46556, USA
{xwang41, tzhao2, mjiang2}@nd.edu

ABSTRACT

...

1 INTRODUCTION

Thanks to high accuracy of entity typing systems [7, 9], typed textual pattern-based methods have been a success in extracting (entity, attribute, value)-tuples (called *EAV-tuples* or *facts*) from unstructured data such as news, tweets, and scientific publications [6, 8]. Recently, temporal fact extraction has been verified as a more precise method for exploring time-related facts. TFWIN [11] platform is one of these approaches that can extract temporal facts in the form of (entity, attribute, value)-tuples along with their time conditions. An example is to find a country's presidents and their term of office represented as $(e, v, [t_s, t_e])$, like a tuple of (Mexico, Vicente Fox, [2000, 2006]).

However, the temporal information extracted by TFWIN can only be accurate to year-level while some real world scenarios may require more accurate temporal facts. Take the presidency of Mexico as an example, Vicente Fox is in office from December 2000 to November 2006, and thereafter Felipe Calderon served as the president until November 2012. The TFWIN approach in only able to extract facts as (Mexico, Vicente Fox, [2000, 2006]) and (Mexico, Felipe Calderon, [2006, 2012]), which may result in confusion in the actual president in year 2006. The dataset used by TFWIN also include temporal information on the month-level, though sparser than the year-level data. Therefore, we propose the Misread-Proof Temporal Fact Extraction (MPT) approach to extract more accurate temporal facts from unstructured data.

The problem is defined as: given massive text corpus and a specific fact type (or called attribute a), extract true temporal facts that are represented as $(e, v, [t_s, t_e])$-tuples, where the valid time of v being an attribute value of e's attribute a is from t_s to t_e, while t_s and t_e can be accurate to month.

Our main contributions in this paper are listed as follows:

- Developing the MPTF approach to extract temporal facts at a more accurate level compared to TFWIN approach.
- Experiments on a large real-world dataset demonstrate the effectiveness and efficiency of our proposed method.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 RELATED WORK

In this section, we review the relevant fields of our work, temporal fact extraction and two major approaches, TFWIN and MajVote that MPT will build upon.

Temporal fact extraction. This task is defined as extracting (entity, attribute name, attribute value)-tuples along with their time conditions from text corpora. The concept of fact is broader than relation between two entities. There are two series of existing natural language processing models: one is based on dependency parsing [10**? ? ?**], and the other is based on learning neural networks with human annotations [**? ? ?**]. These models usually work on individual sentences/paragraphs [1**? ?** , 2], and suffer from high complexity and unavailability of training data [**?**]. It is important to leverage the data amount and evaluate the trustworthiness of extracted information using the truth finding technology. Fortunately, textual patterns, such as E-A patterns [4, 5], S-O-V patterns [12], parsing patterns (by PATTY [8]), and meta patterns (by MetaPAD [6]), have been proposed to turn text data into structures in an unsupervised way – patterns can be considered as information sources. We propose to use dense temporal signals from post time and temporal tags for estimating pattern reliabilities and finding true temporal facts. Our method generates better recall with a high precision, while it is much lighter than complicated NLP models.

TFWIN. The TFWIN framework relies on World's invariant to define conflicts in the process of temporal facts extraction. The invariants are constraints on the possible number, say, *one* or *multiple*, of values/entities associated with an entity/value with/without respect to a time: they include *time-irrelevant* constraints:

- $H_{1e-to-1v}$: one entity has only *one* value on the attribute;
- $H_{1v-to-1e}$: one value is associated with only *one* entity;

and *time relevant* constraints:

- $H_{(1t)1e-to-1v}$: at a time, one entity has only *one* value;
- $H_{(1t)1v-to-1e}$: at a time, one attribute value is associated with only *one* entity.

Based on these constraints, TFWIN can be conducted through the following steps:

- Use MetaPAD to turn text data into structured data, which includes a set of meta patterns of entity types (e.g., $Country, $Person) and data types (e.g., $Month, $Year)
- For a specific attribute (e.g., country's president), generate a set of (entity, value)-pairs from the meta patterns of the corresponding entity type (e.g., $Country) and value type (e.g., $Person).
- Attach time signals to (entity, value)-pairs and get (entity, value, time)-tuples, i.e., EVT-tuples.
- Starting with a seed pattern that is of high reliability, extract a set of EVT-tuples as initial temporal facts.

- Grade and label each EVT-tuple in every meta pattern based on the constraints compared to the initial temporal facts, and update the extracted temporal facts.
- Using the scores of EVT-tuples to further access the reliability of each pattern.
- Recursively grade and label EVT-tuples based on pattern reliability and constraints, and then update pattern reliability, until all EVT-tuples are labelled or reaching maximum iterations.

MajVote. MajVote [3] is an approach that uses the weighted majority voting strategy and returns most frequent EVT-tuples.

3 APPROACH

In this section, we present the refined framework for the temporal fact discovery. Two variants of our framework are introduced.

3.1 MPT

MPT applied MajVote on TFWIN results. For each person, we choose the month with maximum occurrence time as our output.

3.2 MPT++

The TFWIN results can be divided into two subsets. First, for one country, the ending year of previous president is one year before the starting year of the current president. Second, the difference year is more than one year. For example, from TWFIN's results, Bush's office term ending at 1993, which is one year before Clinton's starting year.

- (united_states,george_h._w._bush,[1989, 1993])
- (united_states,william_jefferson_clinton,[1994, 2000])

While, Jackson's ending year is not one year before Washington's starting year.

- (united_states,andrew_jackson,[1767, 1767])
- (united_states,george_b._washington,[1789, 1792])

So, for the first case, it is more likely occur boundary problem. For example, from the ground truth, we know Bush's office term actually lasts to 1994 January. We define this kind of year as boundary years. In MPT, we consider boundary years of the two related person, and applied MajVote to find the most likely boundary time for these two people. For example, we extend Bush's ending year to 201412. And, extend Clinton's staring year to 201301.Then find the month level time form 201301 to 201412 as the boundary time for Bush and Clinton.

4 EXPERIMENTS

In this section experiments proceses are introduced, results are discussed.

4.1 Experimental Setting

Data Set. The dataset we collected rescources from English Gigaword Fourth Edition LDC2009T13 [51]. News articles here are published over the period mid-1990s to 2010. Six distinct international English newswire are involved, including Agence France Presse, Associated Press Worldstream, Central News Agency of Taiwanm Los Angeles Times/Washington Post, New York Times, and Xinhua News Angency. The size of entire dataset is 26,348MB (25.7GB), with 9.9 million articles and 4.0 billion words.

Evaluation methods. We collected grouth truth, i.e., a set of true temporal fact tuples, on country's president from Wikipedia. The ground truth has 365 $(e, v, [ts, te])$-tuples of 130 countries, which accurate to month. We evaluate the performance of our method on mining the ground truth using standard Information Retrieval metrics: precision, recall, F1 measure and AUC (Area Under the Curve). For all of the metrics, the higher scores indicate that the method has better performance.

Baseline methods. There is few existing work aims to find temporal fact from unstructured data. We compared our method with modified TFWIN results.

4.2 Experimental Results

Overall performance. The performance are shown in Table 1.

Table 1: Performance

	Precision	Recall	F1
TFWIN	0.6411	0.3837	0.4810
MPT	0.5185	0.5822	0.5485
MPT++			

5 CONCLUSIONS

Conclusions.

REFERENCES

[1] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the Web. In *IJCAI*.

[2] Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open Information Extraction: The Second Generation.. In *IJCAI*, Vol. 11. 3–10.

[3] Sally A Goldman and Manfred K Warmuth. 1995. Learning binary relations using weighted majority voting. *Machine Learning* 20, 3 (1995), 245–271.

[4] Rahul Gupta, Alon Halevy, Xuezhi Wang, Steven Euijong Whang, and Fei Wu. 2014. Biperpedia: An ontology for search applications. In *VLDB*.

[5] Alon Halevy, Natalya Noy, Sunita Sarawagi, Steven Euijong Whang, and Xiao Yu. 2016. Discovering structure in the universe of attribute names. In *WWW*. 939–949.

[6] Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M Kaplan, Timothy P Hanratty, and Jiawei Han. 2017. MetaPAD: Meta pattern discovery from massive text corpora. In *KDD*.

[7] Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label Embedding for Zero-shot Fine-grained Named Entity Typing.. In *COLING*. 171–180.

[8] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. PATTY: A taxonomy of relational patterns with semantic types. In *EMNLP*.

[9] Xiang Ren, Ahmed El-Kishky, Chi Wang, Fangbo Tao, Clare R Voss, and Jiawei Han. 2015. Clustype: Effective entity recognition and typing by relation phrase-based clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 995–1004.

[10] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *HLT-NAACL*.

[11] Xueying Wang, Qi Li, and Meng Jiang. 2018. On the Power of theWorldâĂŹs Invariants: When Truth Finding Meets Temporal Unstructured Data. In *KDD*. processing.

[12] Mohamed Yahya, Steven Whang, Rahul Gupta, and Alon Y Halevy. 2014. ReNoun: Fact extraction for nominal attributes. In *EMNLP*.