Final Report

Name: Alex Hoffer, Chongxian Chen, Jacob Smith

Group Number: 59

Project Title: Machine Learn Your Way to March Madness Glory!

Abstract

The application of machine learning to Biochemistry and Biophysics has enabled researchers in this eld to make remarkable discoveries, such as the generation of new DNA sequences. However, students of Biochemistry and Biophysics do not get the opportunity to learn machine learning. Dr. Victor Hsu of the Oregon State University Biochemistry and Biophysics department has commissioned the Stat Champs to produce an instructional module to give his students the chance to familiarize themselves with machine learning. The software product the Stats Champs have agreed to develop is a web page that allows students to train a machine learning model based on the college basketball statistics and machine learning algorithm of their choosing in order to produce a March Madness bracket. This will help students understand how machine learning algorithms produce models and how inclusion or exclusion of certain data can inuence such models. Over the course of Fall term 2016, the Stat Champs developed materials such as design documents and technology reviews in order to prepare for the engineering of the module. Then, in Winter term 2017, the Stat Champs began the software development phase of this project. In Spring term 2017, the project was finished. This report comprehensively describes the lifecycle of producing the project.

CONTENTS

1 Introduction

Biochemistry and Biophysics are two fields that are ripe with many exciting breakthroughs. Machine learning is used by these disciplines to aid their research by providing the ability to generate new sequences of DNA. Our client, Dr. Victor Hsu, is a professor in the Biochemistry and Biophysics department at OSU who recognized that his students should understand machine learning in order to prepare them for their careers. He noticed that the department does not encourage learning machine learning, and that even if they were to, machine learning is a difficult topic for people who haven't been trained in computer science. Additionally, learning machine learning through its application to these fields can be confusing, since machine learned models of DNA can be tough to interpret. Therefore, he commissioned us to produce an online instructional module where his students could grasp machine learning fundamentals in a fun and straightforward manner. To achieve this, he asked us to provide an interface where users could select from men's college basketball statistics and generate a March Madness bracket. In doing so, budding scientists would be able to familiarize themselves with basic machine learning algorithms and also witness how the inclusion or exclusion of data influences the models trained on them. Alex Hoffer developed the GUI, Chongxian Chen implemented machine learning algorithms, and Jacob Smith gathered data for use. Dr. Hsu's role in the project was supervision only. This report chronicles the lifecycle of the project through Fall, Winter, and Spring terms in 2017.

2 Original Requirements Document

2.1 Changes from Original Requirements Document

3 Original Design Document

Design Document for Machine Learning for March Madness

Alex Hoffer, Chongxian Chen, and Jacob Smith
Team Name: Stat Champs

Table of Contents

I. Introduction

Stat Champs
December 2, 2016

AFTER making reasoned decisions about which technologies to use to implement this module, we must now consider how these technologies will work together. This design document intends to explain how the three technologies each of the Stat Champs have selected to use will complete their assigned task. It will do so through the use of diagrams followed by paragraphs describing what the technology must do and why it must do it. The first three designs were produced by Alex Hoffer. These first three designs are chiefly concerned with the usability of the service, specifically the design of the graphical user interface, the instructions that will be presented to the user, and the display of the machine learning bracket that is generated. In technologies 4 and 5 Jake Smith will talk about how we will obtain and store the sports statistics for easy use by our machine learning algorithms. It is our belief that in order for the tool to be effective in educating students it must be well designed and easy to understand. Chongxian Chen will be responsible for designing and implementing machine learning algorithm. Technologies 6, 7 and 8 will be discussing technology involved in designing machine learning algorithm, namely algorithm library, statistics model and cloud server for hosting the project

II. Glossary

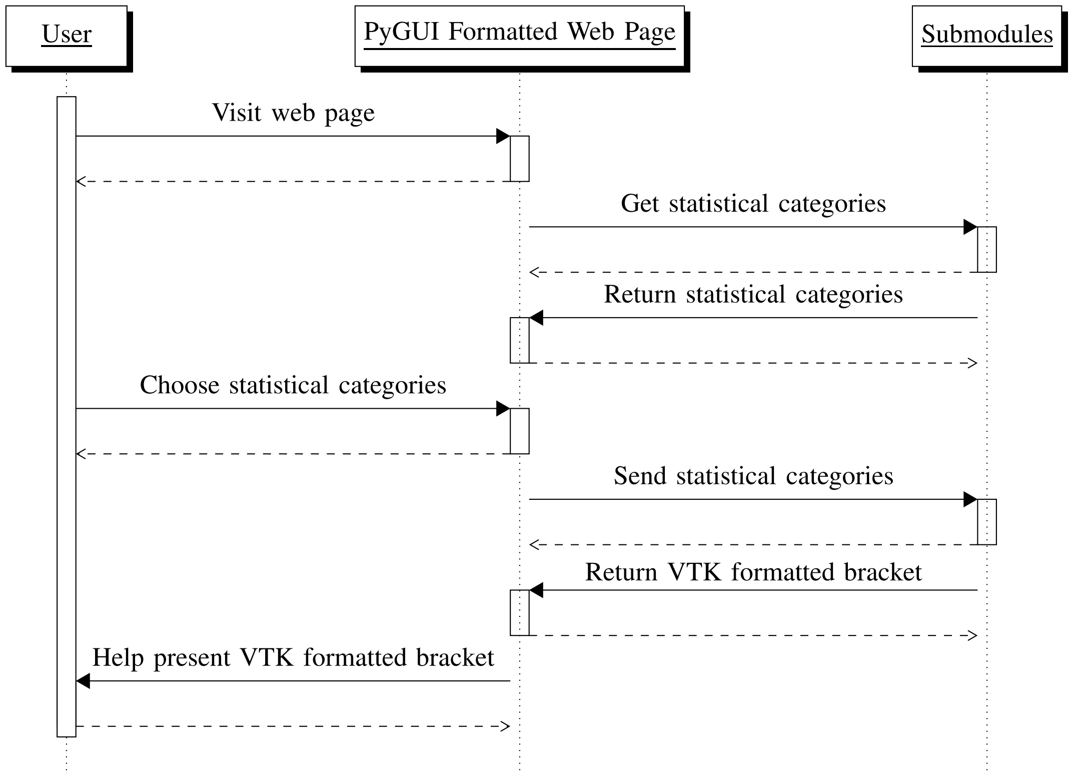
PyGUI: Graphical user interface API designed specifically for use with the Python programming language.
VTK: Data visualization API that will be used with the Python programming language.
Client-server architecture: Website format that consists of a user sending and receiving data to a server which also sends and receives data.
Webix: JavaScript API that is designed to support graphical user interfaces.
AWS: Amazon Web Service.
EC2: Amazon Elastic Compute Cloud.
SSH: Secure Shell.
RDS: Amazon Relational Database Service.
2P = 2-Point Field Goals
2P Percentage = 2-Point Field Goal Percentage
2PA = 2-Point Field Goal Attempts
3P = 3-Point Field Goals
3P Percentage = 3-Point Field Goal Percentage
3PA = 3-Point Field Goal Attempts
AST = Assists
AST Percentage = Assist percentage is an estimate of the percentage of teammate field goals a player assisted while he was on the floor.
Award Share = The formula is (award points) / (maximum number of award points).
BLK = Blocks
BLK Percentage = Block percentage is an estimate of the percentage of opponent two-point field goal attempts blocked by the player while he was on the floor.
BPM = Box Plus/Minus is a box score estimate of the points per 100 possessions that a player contributed above a league-average player, translated to an average team.
DRB = Defensive Rebounds
DRB Percentage = Defensive rebound percentage is an estimate of the percentage of available defensive rebounds a player grabbed while he was on the floor.
DRtg = Defensive Rating for players and teams it is points allowed per 100 possessions.
DWS = Defensive Win Shares
eFG Percentage = Effective Field Goal Percentage; the formula is (FG + 0.5 * 3P) / FGA. This statistic adjusts for the fact that a 3-point field goal is worth one more point than a 2-point field goal.
FG = Field Goals (includes both 2-point field goals and 3-point field goals)
FG Percentage = Field Goal Percentage; the formula is FG / FGA.
FGA = Field Goal Attempts (includes both 2-point field goal attempts and 3-point field goal attempts)
FT = Free Throws
FT Percentage = Free Throw Percentage; the formula is FT / FTA.
FTA = Free Throw Attempts
Four Factors = Dean Oliver's "Four Factors of Basketball Success
G = Games
GB = Games Behind
GmSc = Game Score: was created to give a rough measure of a player's productivity for a single game. The scale is similar

to that of points scored, (40 is an outstanding performance, 10 is an average performance, etc.)
GS = Games Started
MP = Minutes Played
MOV = Margin of Victory
ORtg = Offensive Rating for players it is points produced per 100 posessions, while for teams it is points scored per 100 possessions.
ORB = Offensive Rebounds
ORB Percentage = Offensive rebound percentage is an estimate of the percentage of available offensive rebounds a player grabbed while he was on the floor.
Pace = Pace factor is an estimate of the number of possessions per 48 minutes by a team. (Note: 40 minutes is used in the calculation for the WNBA.)
PER = Player Efficiency Rating The PER sums up all a player's positive accomplishments, subtracts the negative accomplishments, and returns a per-minute rating of a player's performance
PF = Personal Fouls
Poss = This formula estimates possessions based on both the team's statistics and their opponent's statistics, then averages them to provide a more stable estimate.
PProd = Points Produced
SOS = Strength of Schedule; a rating of strength of schedule. The rating is denominated in points above/below average, where zero is average
SRS = Simple Rating System; a rating that takes into account average point differential and strength of schedule.
STL = Steals
STL Percentage = Steal Percentage is an estimate of the percentage of opponent possessions that end with a steal by the player while he was on the floor.
Stops = Measure of individual defensive stops
TOV = Turnovers
TOV Percentage = Turnover percentage is an estimate of turnovers per 100 plays.
TRB = Total Rebounds
TRB Percentage = Total rebound percentage is an estimate of the percentage of available rebounds a player grabbed while he was on the floor.
TS Percentage = True shooting percentage is a measure of shooting efficiency that takes into account field goals, 3-point field goals, and free throws.
TSA = True Shooting Attempts
Usg Percentage = Usage percentage is an estimate of the percentage of team plays used by a player while he was on the floor.
Win Probability = The estimated probability that Team A will defeat Team B in a given matchup.

III. Designs

Design viewpoint 1: Using PyGUI for GUI of webpage

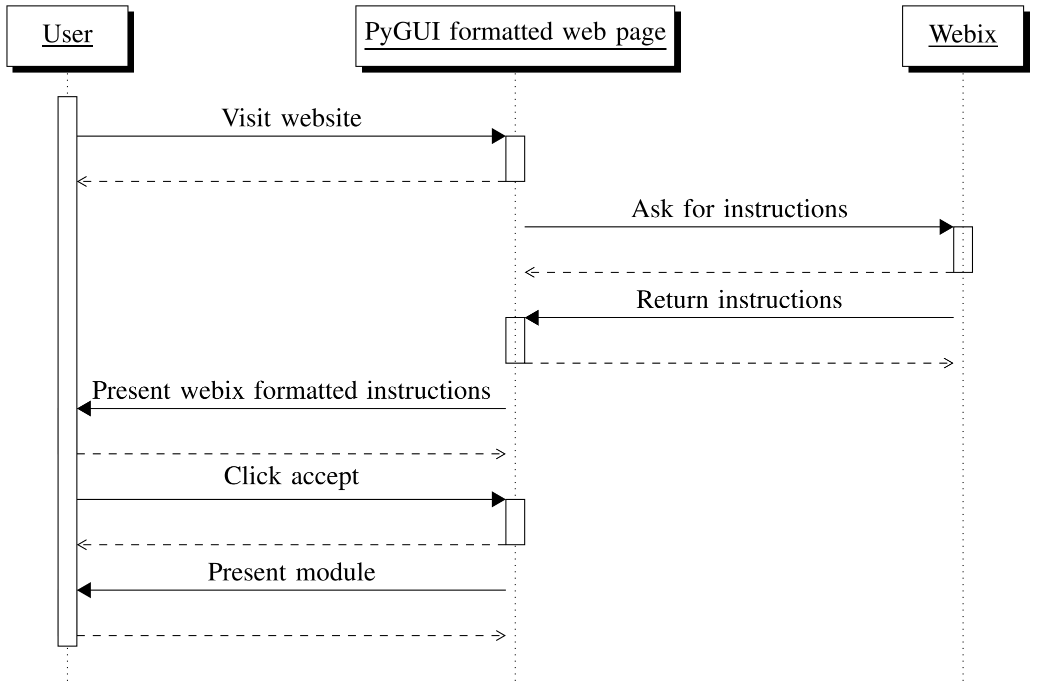
Design view 1:


User
PyGUI Formatted Web Page
Submodules
Visit web page
Get statistical categories
Return statistical categories
Choose statistical categories
Send statistical categories
Return VTK formatted bracket
Help present VTK formatted bracket

Design Rationale: The module needs a web page to operate. This web page needs to be presented in a way that is pleasing to the eye and easy to use, because our intended users are biochemists, not computer scientists. This means the page needs to be as non-esoteric as possible. To achieve this, we will be using PyGUI to format our web page. The precondition of this sequence diagram is that the user has a browser. The postcondition of this diagram are that the user is presented with a bracket that resulted from collaboration between PyGUI and VTK. The flow of events from the perspective of PyGUI is quite simplistic. While a typical client-server architecture will allow us to transfer data back and forth between our submodules, PyGUI must make this data transmission appear as convenient as possible to the user.

Design viewpoint 2: Using Webix to present instructions for the module

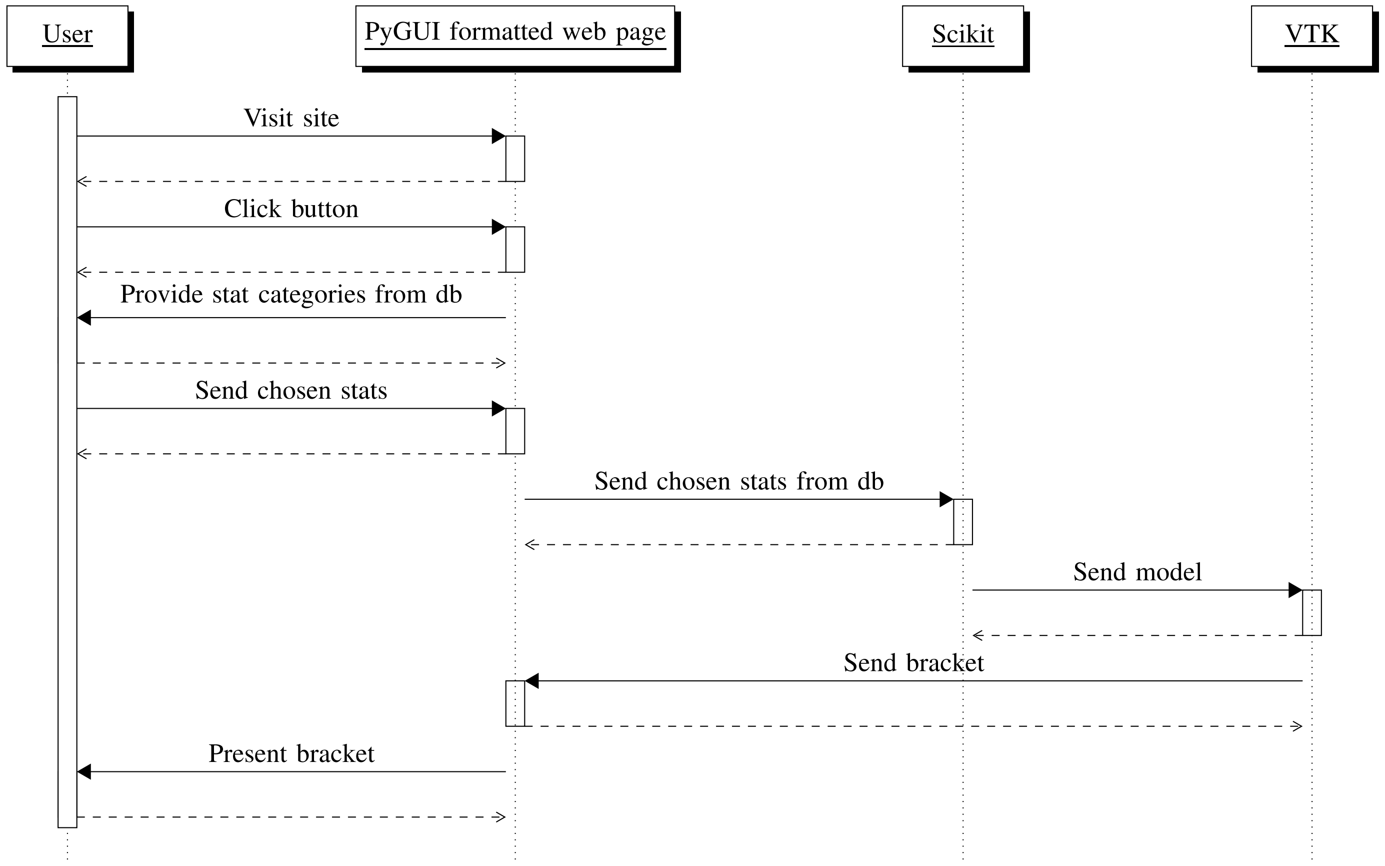
Design view 2:



Design Rationale: The module needs to present instructions to the user which informs them how to use the service. These instructions should appear seamlessly and should be visually inoffensive. Our main aim is reducing the chance of these instructions appearing unclear and thus maximizing utility. To achieve this, we will be using Webix, which will interact with our client/server code as well as with PyGUI. The precondition of this sequence diagram is that the user has a browser and has loaded the web page. The postconditions are that the user has seen the instructions (hopefully having read them) and has clicked accept to begin the service. The flow of events of the average use case are as follows: 1) the user visits the web page 2) the user is presented with instructions on properly using the module 3) the user reads the instructions 4) the user clicks accept 5) the module is presented. Note that the PyGUI formatted page in this diagram is a bit vague. PyGUI, of course, cannot handle the entire module. The web page will be calling a number of other technologies to present the module, but these technologies are not necessary when only considering how Webix will be used.

Design viewpoint 3: Generating bracket using VTK

Design view 3:



Design Rationale: The most essential function of the entire module is presenting a machine learned March Madness bracket. However, the process of machine learning is completely separate from the design of the bracket which will be displayed. Once Scikit generates a machine learned model that reflects the user's chosen statistics, it must communicate to VTK what this data is, and VTK will handle the bracket generation. In order for VTK to produce a bracket for display, a number of preconditions must occur. First, the user must have read and accepted the instructions. Then, the user must have selected data produced by a database. This data must be passed to Scikit, which generates a learned model. The postconditions of this sequence diagram is that the user will have seen the March Madness bracket that results from their chosen statistics. The flow of events for the average use case is as follows: 1) The user reads and accepts the instructions 2) The user selects statistics 3) The server sends these statistics to a machine learning submodule 4) The submodule generates a model 5) The submodule passes the model to VTK 6) VTK transforms the model into a bracket 7) A collaboration of VTK, PyGUI, and the server produce this bracket for the user to view.

Design viewpoint 4: Collection of data for the database

Design Rationale: The Machine Learning algorithms we want to work with get more accurate the more examples and stats you feed into it. Therefore, for the data collection we need to include as many possible data points as possible. We will do this by collecting stats from every major stat website and also pull more advanced data from the hoop-math which breaks down each players moves into more specialized data points. For example, hoop-math with break down a players shots from just saying he shot a 2 pointer to telling you where the shot was and how close a defender was to him at that moment. The design to grab the stats is to manually download all the csv files I can from Sports-Reference and upload them to the database then once I am caught up with the current season I will create a python script that takes the stats from all the daily games and uploads it into the database.

Design viewpoint 5: Database Design Design view 4: Couldnt get UML database design to post correctly. Design Rationale:

For designing the database I have decided to have multiple connected databases. The fist will be a running tally for all the players stats, everything from FG percentage to player efficiency and more. The second will be all the team game logs with their opponents stats for that game as well. That way we can do look ups for past matchups and analyze how both teams and players did against each other and apply that for future matchups. Then for the final database that will be connected to the game logs will be the advanced game player stats for example it will have play by play stats like who passed to who, when and who shot the ball, from where, and who was guarding him. The user will be able to choose between either all three databases in there bracket analysis or just two or even one. Each bracket should be able to bring in stats that should change the outcome of the machine learning predictions.

Design viewpoint 6: SciKit Learn Machine Learning Model Design Rationale:

The project will be implementing the prediction model using python machine learning library SciKit-Learn. More specifically, we will be using the supervised learning regression model.
The model will be starting with simple and straightforward input first, and then gradually add more complex and hard-to-predict input. The benefit of starting with straightforward input is that we can easily verify the prediction accuracy of our model. For instance, the most straightforward input could be the match results between two teams in recent years. In most situation the team that wins significantly more in the recent matches should have a higher chance of winning current match. After confirming that our model could predict well on basic inputs, we can then add more complex inputs that may also affect the result of the game like home team status, weather and rest status.
After verifying that our model?s prediction accuracy is satisfying, we will be working on enable and disable categories of inputs to meet our project?s education purpose. When enabling all or most categories of inputs, we will be expecting our model to be more accurate. With trivial category of input, our model may show very inaccurate prediction that may even contradict to recent match results. The education purpose of our project will allow users to see a difference when they choose different category of data.
When the model is basically complete, we will be testing it with actual matches, particularly the march madness event. We will be continuing modifying our algorithm to enhance the prediction accuracy with live matches.
Timeline: Training model with basic inputs: Winter Week 1 to Week 3. Training model with more categories of inputs: Winter Week 4 to Week 6. Enable and disable category of inputs: Winter Week 7 to Week 9. Testing and improving the model with March Madness: Spring Week 2 to Spring Week 5.

Design viewpoint 7: Amazon Web Service(AWS) to host the database and computing power Design Rationale:

We will be hosting our database and computing machine on Amazon Web Service. AWS is free for the first 12 months of registration with some limitation of use. After researching on AWS website, we will be able to host one linux instance free with Amazon EC2. We will also be able to hosting a free MySQL database with limited I/O from Amazon RDS. Although the I/O times are limited, but it is large enough for our project(10,000,000 I/Os). Hosting our computing power and database on the cloud is a good idea because we will be able to easily resize our computing power and database with AWS infrastructure when our project grows to a bigger size.
To start with AWS free tier for 12 months , we will be creating a group account. First we will be going to EC2 and host a Linux instance. We will have to choose a region among Amazon?s global servers. The one that is most convenient to us will be US West(Oregon). The operating system we will be using will be Ubuntu Server. Ubuntu is very popular among developers and have a large supporting community if we have trouble. Python can run seamlessly on Ubuntu terminal. For the instance type, we have only one option. T2.micro with one core cpu, one Gigabyte Ram and 8 Gigabyte Storage. The Ram is not very big so we may need to be careful with our ram usage or upgrade it in the future.
After creating the EC2 instance, we will moving forward to create the RDS database instance. We will be using relational database MySQL to store our data. The free tier also provide us with one core cpu, one Gigabyte Ram and 8 Gigabyte Storage. That should be enough for the teams in NCAA. We can also easily upgrade it when our project grows. When completing the setup of the database on AWS, we need to manually connect it to a database management tools so we can create tables and manage data. A convenient cross-platform tool we can use will be MySQL workbench. It is open-sourced, free and works very well on most platforms including Windows, Mac and Linux. We will be using SSH to connect the database on AWS to MySQL workbench. After that we can manage it like we have learned in class. Running SQL queries or create table using MySQL workbench?s graphical interface.
Timeline: Creating EC2 and RDS instance on AWS: Winter week 1 to week 2. Connecting to EC2 and RDS instance: Winter Week 3 to Week 4.

Design viewpoint 8: Statistics Model Design Rationale:

With the Statistics Libraries in Python like Py-Statiscs and a large amount of input data in our library, our statistics model is expected to give a good estimate about the weight of each factor in our prediction model. This factor will vary for different teams, but we will have a general statistics equation. After providing it with a large amount of data for each teams, the equation is expected to give respective weight for each team. And then we can supply these weight to our machine learning model to get the final result.
First we will be implementing the equation with basic inputs matching results. With only matching results, the equation may doesn?t make too much sense to estimate the weight since there is nothing to compare with. Then we should add a trivial data to the equation so we can tell the difference. We will be expecting the trivial data to have a significantly less weight than recent match results. For the trivial data, the candidates are weather, team colors, etc. We will try the team colors first since that is an easy to collect data category for the beginning of our project.
After the basic statistics model makes sense with the data category we provide, we will be adding more category of data like player info, home game status, coach info etc. With a large amount of data, we should be able to a weight of each of these categories. Then we can apply these weights in our prediction model. We will be expecting to see our model become more

accurate. With more important categories of data added, our prediction should be more confident and we will be testing it with the march madness results next Spring. With the new data, we will be testing the accuracy of our predictions and improving it.
Timeline: Implement statistics model with basic inputs: Winter Week 1 to Week 3. Implement statistics model with more categories of inputs: Winter Week 4 to Week 6. Enable and disable category of inputs and generate perspective weight: Winter Week 7 to Week 9. Testing and improving the statistics model with March Madness: Spring Week 2 to Spring Week 5.

IV. Agreement

Client

Developer

Developer

Developer

REFERENCES

[1] Kivy: Cross-platform Python Framework for NUI, *Kivy*. [Online]. Available: https://kivy.org/. [Accessed: 14-Nov-2016].

[2] D. Bolton, 5 Top Python GUI Frameworks for 2015 - Dice Insights, *Dice*, Feb-2016. [Online]. Available: http://insights.dice.com/2014/11/26/5-top-python-guis-for-2015/. [Accessed: 14-Nov-2016].

[3] JavaScript Framework and HTML5 UI Library for Web App Development-Webix, *Webix*. [Online]. Available: https://webix.com/. [Accessed: 14-Nov-2016].

[4] UKI, *Best Web Frameworks*. [Online]. Available: http://www.bestwebframeworks.com/web-framework-review/javascript/117/uki/. [Accessed: 14-Nov-2016].

[5] J. Shore, An Unconventional Review of AngularJS, *Let's Code Javascript*, 14-Jan-2015. [Online]. Available: http://www.letscodejavascript.com/v3/blog/2015/01/angular_review. [Accessed: 14-Nov-2016].

[6] F. Lungh, Notes on Tkinter Performance, *Effbot*, 14-Jul-2002. [Online]. Available: http://effbot.org/zone/tkinter-performance.htm. [Accessed: 14-Nov-2016].

[7] VTK-Enabled Applications, *VTK*. [Online]. Available: http://www.vtk.org/. [Accessed: 14-Nov-2016].

[8] Wax GUI Toolkit, *Python*. [Online]. Available: https://wiki.python.org/moin/wax. [Accessed: 14-Nov-2016].

[9] Amazon Machine Learning, *Amazon*. [Online]. Available: https://aws.amazon.com/machine-learning/. [Accessed: 14-Nov-2016].

[10] ”SciKit Learn,Machine Learning in Python, *scikit-learn*. [Online]. Available: http://scikit-learn.org/stable/. [Accessed: 14-Nov-2016].

[11] ”Pylearn2 devdocumentation, *Pylearn2*. [Online]. Available: http://deeplearning.net/software/pylearn2/. [Accessed: 14-Nov-2016].

[12] ”ONID - OSU Network ID, *Oregon State University*. [Online]. Available: http://onid.oregonstate.edu. [Accessed: 14-Nov-2016].

[13] ”Python statistics Mathematical statistics functions, *Python*. [Online]. Available: https://docs.python.org/3/library/statistics.html. [Accessed: 14-Nov-2016].

[14] ”Statistical functions (scipy.stats), *scipy.stats*. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/stats.html. [Accessed: 14-Nov-2016].

[15] ”Python Data Analysis Library, *Pandas*. [Online]. Available: http://pandas.pydata.org. [Accessed: 14-Nov-2016].

3.1 Changes from Original Design Document

4 Original Technology Review

4.1 Changes from Original Technology Review

5 Weekly Blog Posts

5.1 Alex

5.1.1 Fall Term

5.1.1.1 *Week 3*: This week we collaborated with Dr. Hsu and wrote our Project Statement, which he reviewed and signed off on. We submitted the Project Statement at 11 am on 10/14/2016. We did not really encounter any problems, except for maybe some LaTek formatting issues. I am sure the use of LaTek will become easier with more practice. Next week we will be submitting our resumes for peer review and figuring out how to proceed with this project now that we have a clear, working vision to follow.

5.1.1.2 *Week 4*: This week we finalized our project statement by revising it to reflect our instructors' suggestions. We also all produced resumes and gave them to classmates for feedback. Next week we will work on our project requirements document and attend Career Fair.

5.1.1.3 *Week 5*: This week we attended career fair and developed our project requirements document. Next week we will continue developing this document and will consult the client to get his approval.

5.1.1.4 *Week 6*: This week we revised the rough draft of our project requirements document, formatted it using LaTek, and added a Gantt chart. We submitted this requirements document to our client, but as of 3:20 pm have not heard back from him. By the end of the weekend, we hope to have the document signed and submitted. Next week, we will individually work on our tech reviews.

5.1.1.5 *Week 7*: This week we revised our project requirements document and began working on our tech review document. Next week we will finish our tech review document.

5.1.1.6 *Week 8*: This week we developed our technology review document. We ran into some issues coming up with 3 responsibilities for everybody. We also had some difficulty identifying potential technologies for each of these responsibilities. Next week we will complete our design document.

5.1.1.7 *Week 9*: This week we talked about how we would make our design document, and discussed how we would approach recording ourselves for the progress report. We are having some difficulties with design document formatting. We will probably rent out a microphone for use with a computer to record ourselves for the progress report. Next week we will turn in our design document.

5.1.1.8 *Week 10*: This week we finished our design document. We will send it to our client and get a signature as soon as possible. We faced difficulty in getting LaTex to properly generate designs like message sequence diagrams. Next week we will submit our progress report and conclude the term.

5.1.2 Winter Term

5.1.2.1 *Week 1*: This week we came back from winter break and re-calibrated. We voted on a meeting time (Tuesdays) and attended the first class. We look forward to the term.

5.1.2.2 *Week 2*: This week we had our first meeting back with our TA. Our meeting consisted of planning out the term for our team. This meant we re-established the responsibilities we set for ourselves individually last term and verbally sketched out an idea of what our Beta release would look like. My own contributions this week were setting up the web page the module will be hosted on and doing some GUI work. The web page can be found here: http://web.engr.oregonstate.edu/ hoffera/CapstoneProject/MachineLearnYourWayToMarchMadnessGlory.html. More to come on the GUI work. Chongxian has set up some machine learning algorithms we can use so next week we will have Jake provide the data to them to see how they operate. After we get a handle on these algorithms, we will begin setting up the module.

5.1.2.3 *Week 3*: This week Chongxian arranged the machine learning algorithms, Jake compiled the statistics we'd use in a .csv file, and I worked on the GUI of our web page. Next week we have class on Thursday and wee should have the algorithms being allowed to accept stats as input.

5.1.2.4 *Week 4*: This week I continued to polish the GUI for the webpage. Chongxian has selected our machine learning algorithms and Jake has helped him find examples of how to implement them. Jake also gathered a lot of basketball statistics for use in the module. We need to do the OneNote portfolio, a progress report, and a voice-over update by late February.

5.1.2.5 *Week 5*: This week we attended class, Chongxian continued developing our machine learning algorithms, Jake continued to gather data, and I continued to develop our GUI. Next week we plan to release an alpha version of our module and we need to create a OneNote, edit our documents, make a status report, and submit these to the OneNote.

5.1.2.6 *Week 6*: This week we completed our progress report, both written and presentation versions. I made our OneNote and uploaded all of our documents to it. We had a bit of a hard time filling up all of the required time for the presentation. Next week we will continue development.

5.1.2.7 *Week 7*: This week we continued coding. I re-submitted my OneNote to Dr. Winters because it didn't go through the first time. I also met with Dr. Winters to modify my OneNote a bit. We need to finish our coding to be at a beta level release.

5.1.2.8 *Week 8*: This week I waited for Jake and Chongxian to make progress. Next week we need to setup a meeting with Dr. Hsu. We also are supposed to be presenting a beta release of our project.

5.1.2.9 *Week 9*: This week we did elevator pitches. I have made a draft of our poster using a LaTex conference poster template, and I added a signature page to our progress report and sent it to our client. My two partners should have made progress on integration. Next week I need to polish the poster more and write my new progress report, which I will use the IEEEtran format for.

5.1.3 Spring Term

5.1.3.1 *Week 1*: This week I went to class to get accustomed to what the term would entail. I updated some CSS on our page. Then, I emailed our client with a link to our project. Hope to hear from him next week and hope to update our poster for Expo.

5.1.3.2 *Week 2*: In week 2, Chongxian implemented different algorithms to choose from. We all updated the poster and submitted our new draft. We also emailed our client again with our finished product and our draft for him to sign off on. We await his response. In the coming weeks, we need his sign-off on the poster and then we have Expo.

5.1.3.3 *Week 3*: This week we attended class. I went to Dr. Hsu's office to talk to him about signing off on our posters. We also pushed our code to the Github repo. Our poster final is due May 1.

5.1.3.4 *Week 4*: This week I modified our poster to match our client's suggestions, then McGrath's suggestions. Then, I submitted the poster for printing. Next week I need to do the WIRED assignment.

5.1.3.5 *Week 5*: This week I interviewed Brandon Chatham for the WIRED assignment. We are preparing for Expo.

5.1.3.6 *Week 6*: This week we got our spring term progress report ready. Expo is next week and we're preparing.

5.1.3.7 *Week 7*: This week we had Expo. Term's almost over. In the remaining three weeks we need to edit our original docs, finish 3 small writing assignments, and do a final progress report.

5.2 Chongxian

5.3 Jake

6 Final Poster

MACHINE LEARN YOUR WAY TO MARCH MADNESS GLORY!

Teaching Biochemists and Biophysicists Machine Learning

Introduction/Background

IMPORTANCE OF MACHINE LEARNING TO BIOCHEMISTRY AND BIOPHYSICS

Biochemistry and biophysics are two fields that are ripe with many exciting breakthroughs. Machine learning, a type of artificial intelligence where computer programs adapt to new data, is used by biochemists and biophysicists to do things like analyze genomic DNA sequences. Our client, a professor in the department of Biochemistry and Biophysics at OSU, recognized there was a need for his budding scientists to understand machine learning so they could be better prepared for their careers.

NEED FOR A MACHINE LEARNING INSTRUCTIONAL TOOL

Our client noticed that the Biochemistry and Biophysics curriculum at OSU did not encourage undergraduate students to learn machine learning. Even if machine learning classes were to become a cornerstone of their coursework, the content would be difficult for people without a Computer Science background. To make matters worse, teaching machine learning to these students through its application to biochemistry is particularly challenging, since biochemical results are non-definitive in that DNA sequences often do not need to be exact and can be unclear. Meanwhile, college basketball results are win-lose and therefore it is more straightforward to interpret the differences in results based on changing the inputs.

WHAT WE WERE COMMISSIONED TO DO

We were enlisted to produce an online instructional module where these students could grasp machine learning fundamentals in a clear manner. Our client wanted us to develop this module so that students could generate machine learned NCAA March Madness brackets. Since a fundamental aspect of learning machine learning is recognizing how the inclusion or exclusion of data influences resulting models, this module would satisfy the need by producing models (brackets) that were distinguishable from each other based on the college basketball statistics a user chose for training.

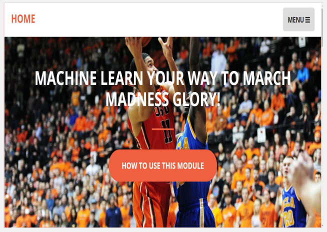
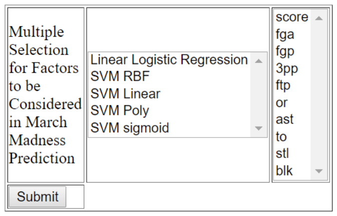


Fig. 1: Home page of website that includes project information and a link to our module.



Multiple Selection for Factors to be Considered in March Madness Prediction	Linear Logistic Regression SVM RBF SVM Linear SVM Poly SVM sigmoid	score fga fgp 3pp ftp or ast to stl blk

Submit

Fig. 2: Menu where the user chooses a machine learning algorithm and stats to generate a bracket with.

PROJECT INFORMATION

Class: CS Senior Capstone, 2016-2017

Developers:

- **Alex Hoffer** (hoffera@oregonstate.edu)
- **Jacob Smith** (smitjaco@oregonstate.edu)
- **Chongxian Chen** (chencho@oregonstate.edu)

Client: Dr. Victor Hsu, Oregon State University, Department of Biochemistry and Biophysics

PROJECT DESCRIPTION

To implement the module, we needed to complete the following five steps:

1. Develop a Graphical User Interface (GUI)
2. Aggregate/select college basketball statistics
3. Feed statistics to machine learner
4. Train a model using an algorithm of the user's choosing
5. Generate March Madness bracket that represents the model

The following headings are technical descriptions of the five steps:

1. GUI

Alex used HTML, CSS, and JavaScript to produce the GUI for our web page. HTML was used to split the page into logical sections such as Home (found in *Fig. 1*), Instructions, Module, Purpose, and About. We utilized CSS to make these sections look clean and usable. Finally, JavaScript was used to enhance the user experience by making the page interactive, such as turning certain buttons different colors upon clicking in order to notify the user of the action they had just performed.

2. AGGREGATE/SELECT STATISTICS

Jacob gathered college basketball statistics from 1985 to the current season from the website Kaggle.com in the CSV file format. Since the regular season didn't conclude until March, Jacob manually updated the database to reflect the current standings frequently until the final game was played. Then, he added stats from the tournament for future use in algorithms and analysis. We used a Python script to allow users to choose from a wide variety of stats including categories like field goals attempted per game to train a model on, as demonstrated by *Fig. 2*.

3. FEED STATISTICS TO MACHINE LEARNER

Using the Python SciKit-Learn library, Chongxian read the CSV files of the user selected statistics into Numpy arrays.

4. TRAIN A MODEL USING AN ALGORITHM

Along with their choice of statistics, users are also able to choose between different machine learning estimators such as Linear Regression and SVM Polynomial. By using a basketball ELO rating system, the supervised machine learning model is able to fit on the statistics and predict new matches. A CSV file of the match results between two teams with the probability is generated as a result. The bracket results effectively present how the users choice affects the machine learning prediction.

5. GENERATE BRACKET OF RESULTS

While a machine learned module is being generated, the user is presented with a screen that includes the command line arguments given to SciKit and informs the user on which steps are necessary to complete their request. The prediction CSV file generated from the machine learning model was then transferred into bracket form by Jake using a Python script.

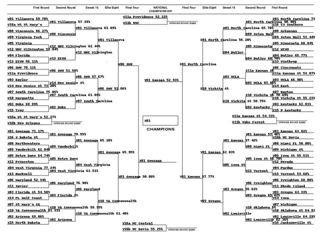


Fig. 3: A March Madness bracket predicted by the SVM RBF algorithm.

CONCLUSION

Alex Hoffer, Jacob Smith, Chongxian Chen

FEATURES PROVIDED BY THE MODULE:

- A Graphical User Interface (GUI) including a "Home" page, "Instructions" page, "About The Developers" page, and a "Purpose" page.
- The ability to select from a set of college basketball statistics.
- The ability to select from a set of popular machine learning algorithms.
- A machine learned bracket that corresponds to the specific statistics and algorithm the user requested the model to be trained on.

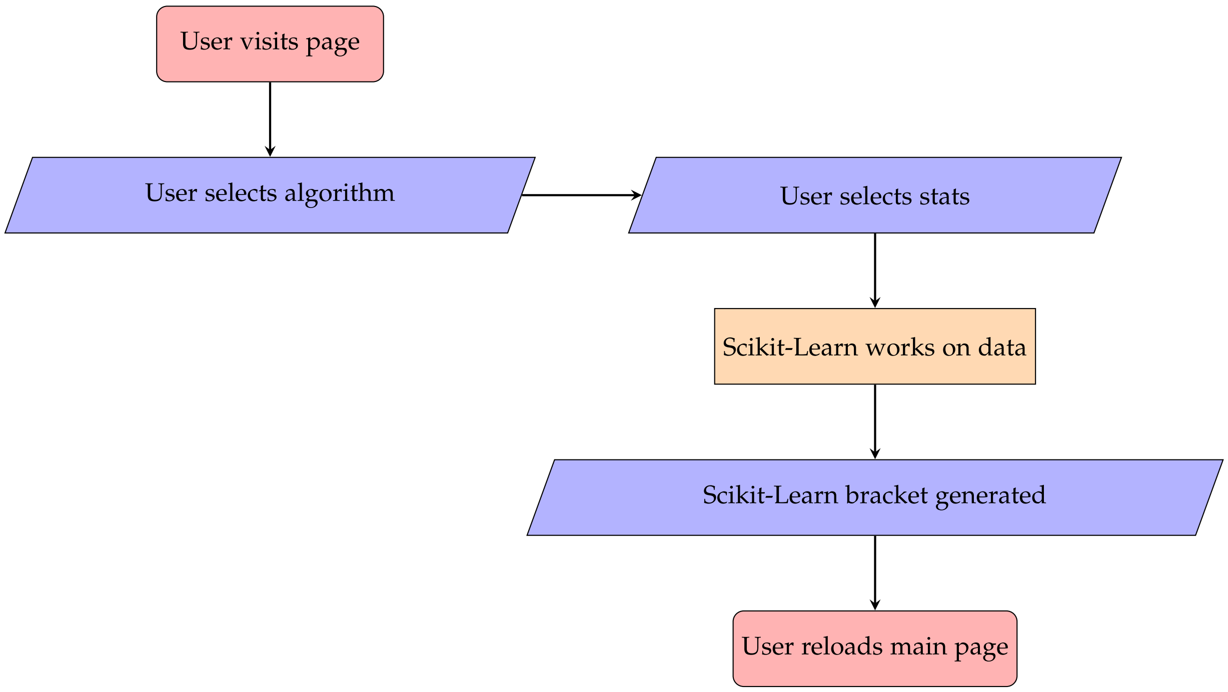
The project was completed in early April. All functional requirements as outlined by our client were completed. Future improvements to our module could include more machine learning algorithms, a wider variety of statistical categories, a more elegant looking outputted bracket, and finding a way to increase the speed at which the machine learning algorithms generate results. Additionally, the developers of such modules may wish to have prior machine learning experience, more proper modes of communication, and a more specific work schedule established before development in order to allow for time after completion to polish each component of the module separately.

Oregon State UNIVERSITY

7 Project Documentation

We were both lucky and unlucky to be assigned a project that did not make well-detailed documentation a necessary responsibility. We were lucky in the sense that this shaved off a lot of work we would have to do, and unlucky in the sense that this project did not expose us to how to properly write documentation.

7.1 How the project works



7.2 How to use the software

No installation is necessary. To use our software, all a user has to do is go to this website: http://web.engr.oregonstate.edu/ hoffera/CapstoneProject/MachineLearnYourWayToMarchMadnessGlory.html. On this site, a link to the machine learning module is posted. The module is compatible across all browsers. There are instructions on how to use the module on this site, as well as on the page where the module is hosted itself. To use the module, the user clicks on one machine learning algorithm and any number of the provided statistical categories and submits their choices. Then, a waiting period ranging from two minutes for simple machine learning algorithms to many hours for more precise algorithms is required. A message displays on the screen to inform the user they must wait. Then, the user is informed when the bracket has been generated. Finally, the user returns to the module page to see their generated bracket presented. The user can repeat this process an arbitrary number of times to generate an arbitrary number of brackets, but of course, the amount of time it takes to utilize certain machine learning algorithms is an important constraint the user must consider.

7.3 Requirements for usage

No special hardware, operating system, or runtime requirements dictate the usage of this module. As stated previously, any browser on any operating system with internet access can utilize the module.

7.4 API Documentation

We were not asked to document any API, and do not have any user guides. Such documents would not be particularly useful to any programmers because our module uses Scikit-Learn. This means that the API has already been documented for us by the fine people at Scikit-Learn. This documentation can be found here: http://scikit-learn.org/stable/documentation.html.

8 How We Learned New Technology

8.1 Alex

8.1.0.8 *Websites*:

- https://stackoverflow.com/
- https://www.w3schools.com/js/
- http://scikit-learn.org/stable/documentation.html

Stack Overflow was most valuable because it provided useful information on debugging small issues with web development. W3 Schools was second to this, because it gave me a lot of insight on how to use JavaScript to make a website *pop*. JavaScript is one of my weaker languages, so I needed it to make sure I was doing the right things. Finally, the documentation for Scikit-Learn was of course valuable for the inclusion of machine learning algorithms, but is listed last because implementation of machine learning algorithms was primarily Chongxian's responsibility.

8.1.0.9 *People*: Several people on campus were tremendously helpful in providing information that we needed. We owe a debt to our teaching assistant Xinze Guan for recommending Scikit-Learn for machine learning algorithm implementation. Our client, Dr. Victor Hsu, was also useful in this regard. We also relied on our instructor Kevin McGrath's LaTeX templates and Makefiles and Dr. Kirsten Winters' writing advice. Finally, my teammates helped me learn a lot of new and exciting technologies, with Jake demonstrating to me proper usage of LaTeX and Chongxian for exploring Scikit-Learn and elucidating a lot of its mysteries to me.

9 What We Learned

9.1 Alex

9.1.0.10 *Technical Information*: In terms of web development, I learned more JavaScript and polished my HTML/CSS abilities. I became acquainted with how Amazon Web Services works and what their limits on their free service are. I also learned a little on how to write embedded Python, how to use Scikit-Learn in Python, and how to use Python to scrape data from websites. Since these were not my primary responsibilities, though, I didn't learn them as well as I would've liked. Finally, I learned how to use LaTeX, proper usage of the IEEETran format, and the mathematical underpinnings of various machine learning algorithms.

9.1.0.11 *Non-Technical Information*: I learned how to interact with a software client. I also learned how to take an idea and move it through the necessary phases of development, including requirements, planning, and design phases all the way to implementation, maintenance, and presentation. Essentially, I learned the engineering practices necessary to develop a project from embryo to adulthood, and learned how to sell or pitch an idea.

9.1.0.12 *Project Work*: I learned that a project must run on a well-defined schedule. If each milestone of a project builds on a previous one, this is especially important. Without a rigid schedule, you fall behind sooner rather than later and the versions of your project you will release will be fraught with bugs.

9.1.0.13 *Project Management*: I learned that in a team setting, somebody has to be the team manager (whether de facto or not), otherwise milestones won't be reached and the project will be delivered late.

9.1.0.14 *Working in Teams*: I discovered certain methods for overcoming differences in work style. Some people procrastinate, others get work done early. Theres nothing wrong with either approach, but if youre in a team where there are conflicting philosophies, you need to find out early how to bridge the gap between you and your group mates.

9.1.0.15 *What I would do differently*: I would choose different responsibilities for this project. I wish I couldve done more of the machine learning aspects, rather than simply oversee Chongxians progress. I wouldve considered using something besides Scikit-Learn because I wouldve liked to have used more exotic machine learning algorithms, or at the very least neural networks, which are not supported in Scikit-Learn.

9.2 Chongxian

9.3 Jacob

Appendix A
Essential Code Listings