

Predicting March Madness

Creating a predictive data model on the NCAA men's basketball tournament



Clark Enge

Paul Hunt

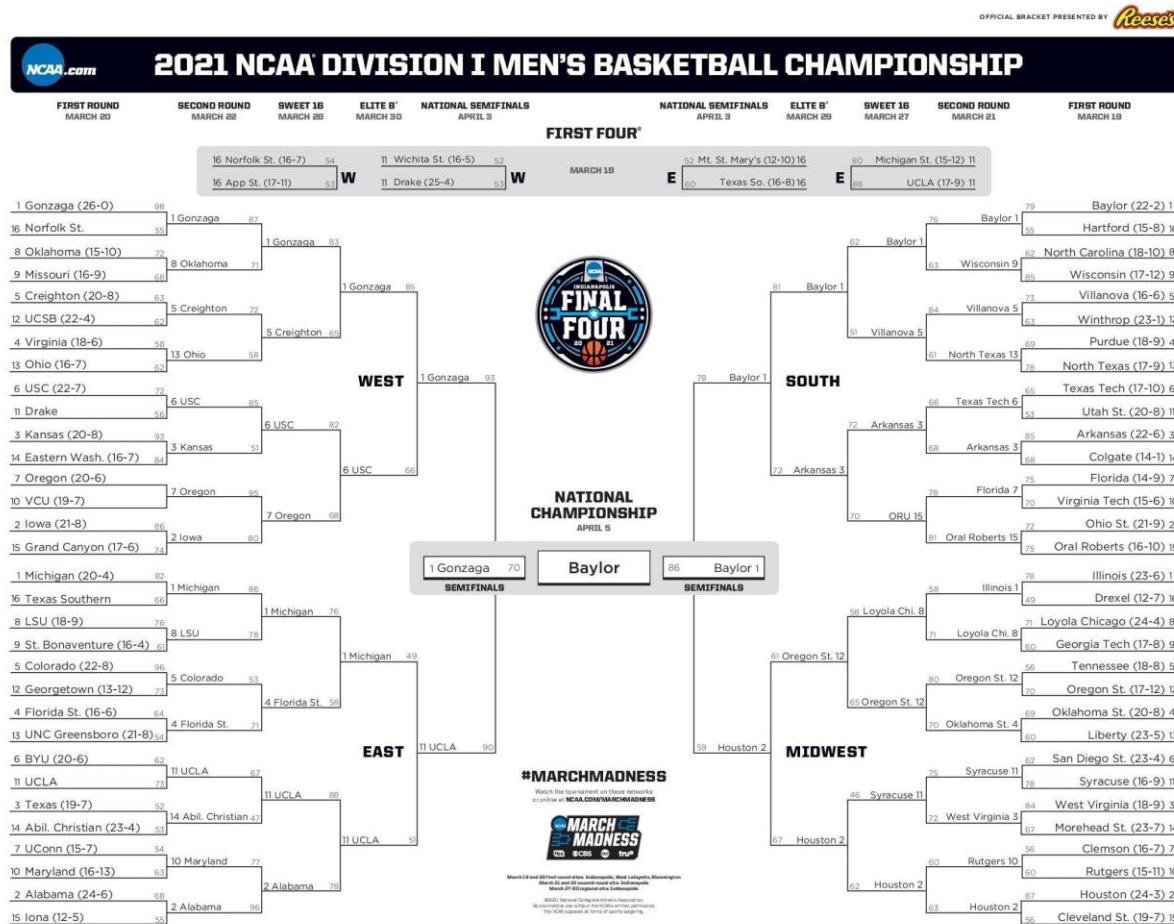
Villa Park High School

Abstract

- My problem/question for this experiment was: how can I predict the outcome of the NCAA march madness tournament using statistics? My procedures were the following: find data on the internet, manipulate this data so it would be ready to be uploaded, upload the data to a pandas dataframe, and write methods using the data frame to predict the winners of the tournament at different stages.
- My Data was the following:
- The Predicted Final Four for each Year (Team in **BOLD** was the predicted champion)
 - 2018: **Villanova**, Gonzaga, Michigan.St, Nevada
 - 2017: **Gonzaga**, Villanova, Kansas, UCLA
 - 2016: **UNC**, Texas A&M, Virginia, Kansas
- My Hypothesis was rejected as even though the algorithm successfully predicted the winner $\frac{1}{3}$ times, the predicted final four teams were highly inaccurate. However, March Madness is not called March Madness for no reason. Therefore, even though my hypothesis was rejected, I believe that this algorithm was as successful as possible in predicting the outcomes of March Madness, due to the volatility and high percentage of upsets.
-

Problem

How can I predict the winners and losers of the NCAA men's basketball Tournament, March Madness, using a predictive algorithm based on regular season statistics?



Introduction (Background Research)

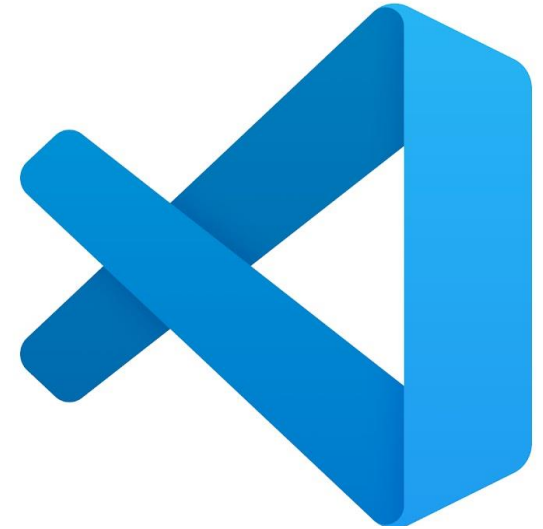
- On a whole, the research behind creating an algorithm is relatively simple; however, the research needed for a predictive model is vast. The internet was well suited to the task, and I found 20-30 years' worth of detailed NCAA men's basketball statistics. The data was all public and free for anyone to use if they knew where to look. Because my model is based on an algorithm, it can be used, in a perfect world, forever. It does not take a large server room or quantum computer to run due to its size and could be utilized using base level computers. The main issue and constraint is the number of years data that have been recorded for NCAA men's basketball. March Madness has only been around since the 1940s meaning there is less than a century of data that can be used by the algorithm. A century worth of data sounds like a ginormous quantity but in terms of machine learning, AI, and predictive data models, it is a small sample.

Hypothesis

- Because I am coding an algorithm and not testing variables in the real world. I will attempt to write pseudo code and explain. First, I am going to create a variable called 'count' that will be used for each game played, as well as rank the statistics I have in my data frames on a scale of 1-10, 10 being the most important. I will compare individual stats at a time.
- For example, let's say that points per game(PPG) is ranked as an 8. I will check the points per game of both teams in the latest season and depending on which team has the higher PPG, the count variable will either be DECREASED by 8 or increased by 8. If team A wins, the count variable will be INCREASED. If Team B wins, the count variable will be DECREASED.
- I will repeat this process for all the major stats in the latest season making sure that the total points possible for a team is a number nondivisible by 2(an odd number) This will prevent any ties. I will then check whether the number is negative or positive. If it is positive, Team A moves on as the winner. If it is negative, Team B moves on as the winner.
- I believe that this algorithm will be approximately 25% accurate across multiple years due to the extensive data and complexity of the algorithm

Materials

- A working computer with a relatively modern OS
- Wifi/Access to the internet
- VsCode
- Python 3.8 or 3.9(any version)
- Pandas(including all sub-libraries)
- PIP
- Microsoft Excel



Procedure

1. Search the internet for the NCAA men's basketball statistics.
2. Copy, cleanse, and prepare the data to be uploaded to a Pandas data frame using Microsoft Excel
3. Create a data frame in VsCode using Pandas and NumPy.
4. Write comparing methods that predict the winner of two teams
5. Iterate through all 64 teams in the tournament and predict winners for the Major 3 stages(Final 4, Last 2, and the Championship)
6. Compare the predicted winners to the actual winners for 3 separate years.

Finished Product

The screenshot displays a Jupyter Notebook titled "Untitled-1.ipynb" within the Visual Studio Code editor. The notebook contains several cells of Python code and comments, indicating a completed project.

Cell 1 (Code): This cell contains a series of calls to the `comparingMethods` module, likely for comparing different teams or strategies. The code includes comments like `#HI, This is just to confirm that my work is my own.` and `#The '#' sign allows the computer to disregard this line.` The code ends with a conditional statement: `if(points > 0): return teamA else: return teamB`.

Cell 2 (Code): This cell contains a single line of code: `>class bracket():...`.

Cell 3 (Code): This cell contains a single line of code: `>class region():...`.

Cell 4 (Code): This cell contains a series of imports and a function definition: `import pandas as pd`, `import numpy as np`, `import sys`, and `def main():`. The function `main()` includes a `rawstring` variable pointing to a file path, a `global df` declaration, and a `pd.read_excel` call. It also includes a comment: `#will create 4 different zones then compare the winner of each zone` and a `#starting with east` comment. The function ends with a list comprehension: `# order of seeds is [1,16,8,9,5,12,4,13,6,11,3,14,7,10,2,15]` and a `list_east = bracket.createlist('East')` call.

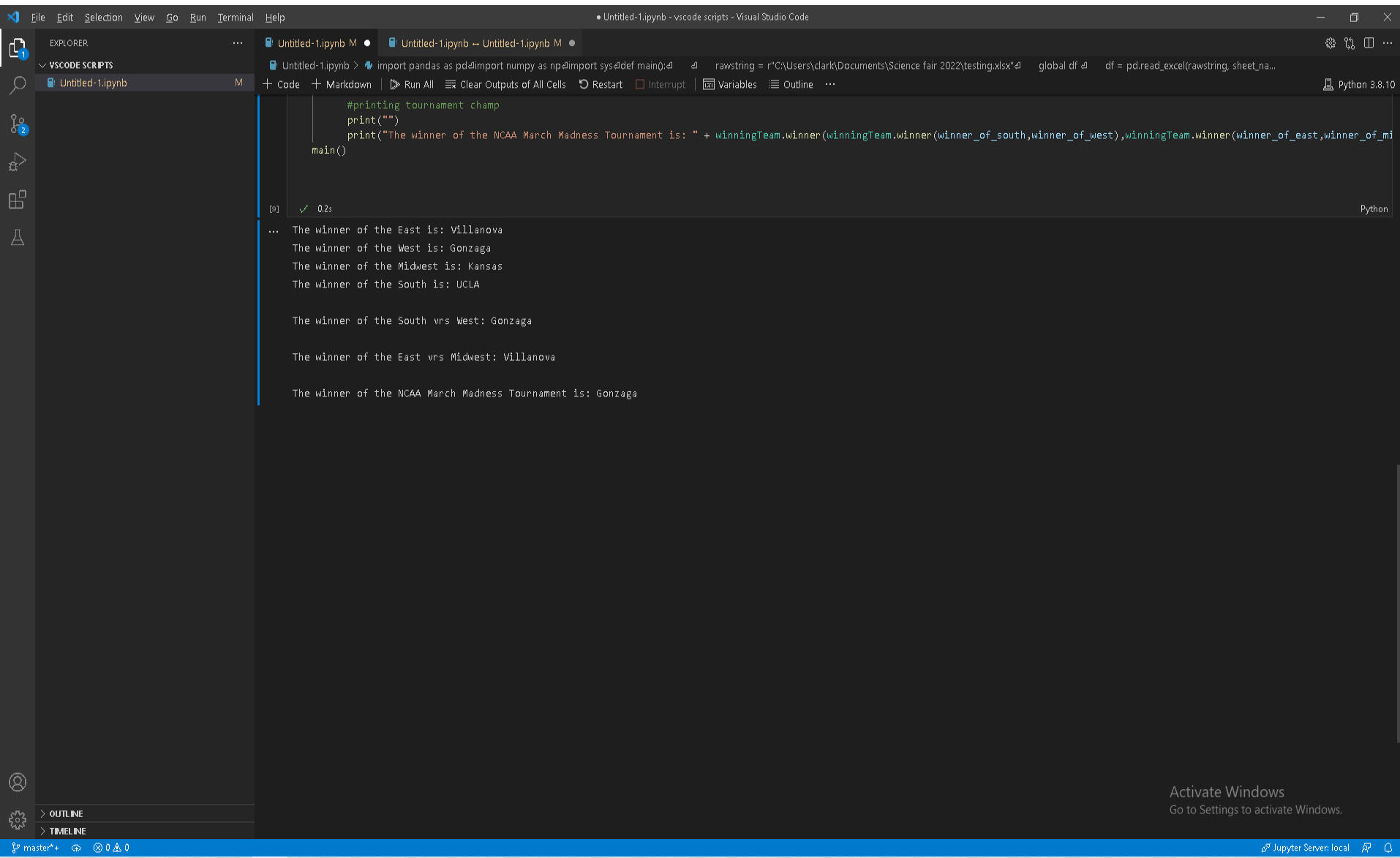
The bottom status bar shows the file is named "master*", the editor is in "Python" mode, and the Jupyter Server is running locally. The bottom right corner displays the "Activate Windows" watermark.

Results

- My Results were the following:
- The Predicted Final Four for each Year(Team in **BOLD** was the predicted champion)
 - 2018: **Villanova**, Gonzaga, Michigan.St, Nevada
 - 2017: **Gonzaga**, Villanova, Kansas, UCLA
 - 2016: **UNC**, Texas A&M, Virginia, Kansas
- The *ACTUAL* Final Four for each Year(Champion in **BOLD**)
 - 2018: **Villanova**, Michigan, Loyola Chi., Kansas
 - 2017: **N. Carolina**, Gonzaga, Oregon, South Carolina
 - 2016: **Villanova**, N. Carolina, Syracuse, Oklahoma

Results (Contin.)

Results for 2017



The screenshot shows a Jupyter Notebook in Visual Studio Code. The notebook is titled "Untitled-1.ipynb" and contains a Python script that reads an Excel file and prints the results of the 2017 NCAA March Madness Tournament. The script is as follows:

```
import pandas as pd
import numpy as np
import sys

def main():
    rawstring = r"C:\Users\clark\Documents\Science fair 2022\testing.xlsx"
    global df
    df = pd.read_excel(rawstring, sheet_name="2017")

    #printing tournament champ
    print("")
    print("The winner of the NCAA March Madness Tournament is: " + winningTeam.winner(winningTeam.winner(winner_of_south, winner_of_west), winningTeam.winner(winner_of_east, winner_of_midwest)))

    main()
```

The output of the script is as follows:

```
[0] ✓ 0.2s
... The winner of the East is: Villanova
The winner of the West is: Gonzaga
The winner of the Midwest is: Kansas
The winner of the South is: UCLA

The winner of the South vrs West: Gonzaga

The winner of the East vrs Midwest: Villanova

The winner of the NCAA March Madness Tournament is: Gonzaga
```

The bottom of the screen shows the status bar with the text "master*" and "0 0 0". The bottom right corner of the screen shows the text "Activate Windows Go to Settings to activate Windows." and "Jupyter Server: local".

Discussion

Analyzing my data, I found that, when comparing the final four teams, my algorithm was 25% accurate in 2018, 25% in 2017, and 0% accurate in 2016.

This combines for an average 16.7% accuracy in predicting the final 4. Also, when comparing the tournament winners, my algorithm was 100% accurate in 2018, 0% accurate in 2017, and 0% accurate in 2016. Meaning it was 1/3 or 33% accurate on average. Compared to other algorithms, these percentages are very low. A Machine Learning algorithm was able to model the 67 games with 70 percent accuracy.(Adrian Pierce and Lotan Weininger,

<https://docs.google.com/spreadsheets/d/1pB3rIVBcNmyPhoN5u1gP7wO6KLTe1Lz1QV-TxPVxM78/edit#gid=0>) The 70% vastly overshadows the 17% my algorithm was able to achieve. However, the Machine Learning algorithm uses hundreds if not millions of points of data and was created in years by extremely intelligent frontrunners of data science. Some possible errors or variables that could have influenced my results are corrupt data points, incorrect data, and other events that happen that influence the basketball games themselves.

Conclusion

As a whole, my results show that my hypothesis was overall rejected. This is because my algorithm was 16.7% accurate in determining the final four teams, almost 9 percent less than my hypothesis projected. I conclude that the algorithm is extremely good at finding teams that are overwhelming statistically better than the competition but fails to recognize upsets and minor variables included in these upsets.

Reflection/Application

- I learned that knowing how to write code is not the only skill needed to create programs and algorithms. I spent about 10x the time googling error messages or looking at python documentation that I did writing code. Looking back on this, I would have utilized some sort of cloud to transport my save files between my computer and laptop, as going back and forth between the two was cumbersome. My next steps would be to test the algorithm across more years, going back a few years as well as testing it every year. I would also like to deepen the comparison methods, making my algorithm more accurate. I can apply these results to everyday life by factoring in the predicted model before I create my own model to make the perfect bracket. Applying this to other studies, I find that computers are incredibly linear. They will only do what you tell them and nothing more, no guessing or speculation. Overall, this project made me appreciate computer science and the wizards behind programming all the more.

References Cited

- DISCLAIMER: Due to the extensive amount of research required for this project, the complete number of sources used would take up a ridiculous number of slides. Due to this reason, I have decided to list the main 5 sources used.
- NCAA.com, Z. P. |. (2021, February 27). *2018 NCAA tournament: Bracket, scores, stats, records*. NCAA.com. Retrieved February 25, 2022, from <https://www.ncaa.com/news/basketball-men/article/2020-05-06/2018-ncaa-tournament-bracket-scores-stats-records>
- NCAA.com. (2021, January 22). *How the field of 68 teams is picked for March madness*. NCAA.com. Retrieved February 25, 2022, from <https://www.ncaa.com/news/basketball-men/article/2018-10-19/how-field-68-teams-picked-march-madness>
- *NCAAM teams*. ESPN. (n.d.). Retrieved February 25, 2022, from <http://espn.go.com/mens-college-basketball/teams>
- *Realtimerpi.com real Time Sport rankings - a leading sports ratings and resources community on the internet*. RealTimeRPI.com Real Time Sport Rankings - A leading sports ratings and resources community on the Internet. (n.d.). Retrieved February 25, 2022, from <http://realtimerpi.com/>
- Wikimedia Foundation. (2022, February 24). *NCAA Division I men's basketball tournament*. Wikipedia. Retrieved February 25, 2022, from https://en.wikipedia.org/wiki/NCAA_Division_I_Men's_Basketball_Tournament