**Winner Winner Chicken Dinner**

Peyton Fosnough, [pfosnough@bellarmine.edu](mailto:pfosnough@bellarmine.edu)

**ABSTRACT**

The goal of this assignment was to analyze a data set with Python and complete a classification model. The data set I chose required a variety of both continuous and discrete variables that could potentially predict another variable. I used Python to clean and explore the data, create a logistic regression model, and check its accuracy.

1. **INTRODUCTION**

I came across this data set through Kaggle.com. The dataset contained values including home and away teams field goal, three point, and free throw percentage, as well as assists and rebounds, and whether or not the home team won. Using the logistic regression model, I chose to try and predict whether a bank customer bought a term deposit based on the factor given in the dataset.

1. **BACKGROUND**
   1. *Data Set Description*

My data set came from a user named Nathan Lauga and was posted on Kaggle.com. The data was taken from the NBA stats website and includes the stats from every NBA game from 2004 to December of 2020. The data set’s purpose was to collect data from both the home and away team in every single game of every season and use those team stats to determine which team would win. As someone who is heavily invested in sports and the NBA in particular, this dataset was very interesting to me.

* 1. *Machine Learning Model*

Logistic regression models are an approach for data analysis using mathematical figures. It uses multiple variables at the same time to try and predict a qualitative variable. The model takes the actual values of the continuous variables and the coded 0s and 1s of the discrete variables to make a predictive model. A classification report is then run to get an accuracy score which is calculated by taking the weighted calculation of precision and recall. The closer this accuracy score is to 1, the better the model is.

1. **EXPLORATORY ANALYSIS**

This data set contains 26,652 samples with 19 columns with both continuous and discrete data. There were 12 variables that contained null values, including PTS\_home, FG\_PCT\_home, FT\_PCT\_home, FG3\_PCT\_home, AST\_home, REB\_home, PTS\_away, FG\_PCT\_away, FT\_PCT\_away, FG3\_PCT\_away, AST\_away, and REB\_away. All of these variables had a normal distribution and therefore required the mean to be used to fill in the missing values.

**Table 1: Data Types**

|  |  |
| --- | --- |
| *Variable Name* | *Data Type* |
| PTS\_home | Float |
| FG\_PCT\_home | Float |
| FT\_PCT\_home | Float |
| FG3\_PCT\_home | Float |
| AST\_home | Float |
| REB\_home | Float |
| TEAM\_ID\_away | Int |
| PTS\_away | Float |
| FG\_PCT\_away | Float |
| FT\_PCT\_away | Float |
| FG3\_PCT\_away | Float |
| AST\_away | Float |
| REB\_away | Float |

1. **METHODS**
   1. *Data Preparation*

The original data set that I was provided with had 19 columns, however lots of these columns were not relevant to the outcome of the game and whether or not the home team would win, such as the date, season, and home/away team IDs. It was important to clean the data further before diving in to the experimental design so I removed these columns from the dataset.

* 1. *Experimental Design*

Table X: Experiment Parameters

|  |  |
| --- | --- |
| **Experiment Number** | **Parameters** |
| 1 | 60/40 split for train and test |
| 2 | 80/20 split for train and test |
| 3 | 90/10 split for train and test |

* 1. *Tools Used*

The following tools were used for this analysis: Python v6.4.8 running the Anaconda 2.2.0 environment for HP Laptop. From Python, I used several libraries including Pandas, Numpy, Matplotlib, Seaborn, and Sklearn.

Python libraries: pandas, sklearn, numpy, seaborn

* Pandas allowed me to import the data set
* Sklearn was the biggest tool used for Logistic regression. It split the data into the training and testing set. It also set up the model for logistic regression and the classification report.
* Numpy was used for filling in the missing values and cleaning the data.
* Seaborn was used to plot the variables with missing data to determine which measure of centrality was appropriate.

1. **RESULTS**
   1. *Classification Measures*

See Appendix for visual displays of the confusion matrix and classification report.

* 1. *Discussion of Results*

My model ended up being a very good predictor for whether or not the home team would win. All three classification reports came back with an accuracy score of 100%, which was very good. While my model didn’t have any false negatives, it did have a couple of false positives in each of the experiments which isn’t ideal but still shows that the model was accurate.

* 1. *Problems Encountered*

The biggest problem that I encountered came when I didn’t fully clean the dataset and used to few iterations for training the model. Not cleaning the dataset and using too few iterations caused a lot of problems when I was running my tests such as only receiving true and false negative values. When this occurred, I knew that something was wrong with my model and the dataset, and I ended up having to remove more columns that weren’t relevant to the outcome I was trying to predict.

* 1. *Limitations of Implementation*

The accuracy score indicates that this model is good and there is no reason it shouldn’t be applicable over the course of time that the dataset applies to. However, with new rules and changes in the game of basketball every year, this model, however accurate it was, may no longer be quite as accurate. For example, teams are continuing to rely more heavily on their ability to shoot three pointers accurately, so applying this model to NBA teams today may not be accurate.

* 1. *Improvements/Future Work*

I think it would be very interesting to work with a dataset that contains the most recent years’ stats and results. By doing this, I would probably be able to create a model that could possibly predict the outcome of every game throughout the season or even playoffs, which I think would be very cool.

1. **CONCLUSION**

Overall, I worked very hard on this model, and it proved to be successful. Despite not having accurate test results and low accuracy scores at first, I was able to clean the data a little bit more and create a better model. The data set contained a lot of unneeded information and variables for what I was trying to find, so I had to drop a lot of these variables as they were interfering with the test results and accuracy. Once these were dropped, I was able to run three experiments, all with 100% accuracy at different split ratios.

1. **APPENDIX**

**Confusion Matrix: Experiment 1**

**Chart, treemap chart

Description automatically generated**

**Classification Report: Experiment 1Table

Description automatically generated**

**Confusion Matrix: Experiment 2**

**Chart, treemap chart

Description automatically generated**

**Classification Report: Experiment 2** **Table

Description automatically generated**

**Confusion Matrix: Experiment 3**

**Chart, treemap chart

Description automatically generated**

**Classification Report: Experiment 3**

**Table

Description automatically generated**