Vinson Lin
vinsonlin@ucsb.edu
4835054
CS 165A
November 14, 2024

MP1 - Naive Bayes Classifier

Architecture

My code is built in three distinct sections.

1. Data file inputs and dataframe restructuring. In this part of the classifier I read in the inputs from the script and created two pandas dataframes for training data and validation/testing data. I then restructured the data and added new features in order to result in the best prediction.
2. Four functions were used to calculate different portions of the Naive Bayes Classifier
    a. probability_of_y() - this function is simple. It takes the total number of a certain class and divides it by the number of all data points. P(Y = y)
    b. calculate_mean_std() - this function calculates the mean and standard deviation of each feature for each class and returns the output into two tables. The tables are of size #_classes x #_features.
    c. gaussian_x_given_y() - calculates the probability of P(X = x | Y = y) using the gaussian formula
    d. Naive_bayes_calculation - returns the predictions of a based on a dataframe of features. It will calculate the probability of each feature given a certain class (using function c) and will add up the logarithms of each probability. It will then add the log of probability of the said class (using function a). It stores this sum in a list and a list will contain the sum for each class. It will then store the class with the highest sum in a prediction list which will be returned once each row of the data frame is finished calculating
3. Outputting the predictions. If the second data set is the validation set it will return the accuracy along with false negatives and positives. Otherwise if it is the test set it will return each prediction on a new line

Preprocessing

I added a new feature based on the past 5 win/loss which just counted the wins as an int. And another ‘ast/tov’ which was a ratio of assists over turnovers. Furthermore I created new features which modeled the difference in last 5 averages between the two teams for each stat. I then dropped all the features that were strings such as team name, season, etc. Additionally I dropped the features that were not a normal distribution such as minutes and offensive

rebounds. And finally I dropped the features with high correlation such as assists, so that the naive bayes assumption wouldn't be violated. And finally before calculation, I normalized all the features because I noticed that some of the distributions were so narrow that they would have a very small standard deviation which would disrupt later calculations. Normalizing the data allowed for each feature to be a normal distribution with standard deviation = 1, and this slightly increased accuracy.

Model Building

My Naive Bayes Classifier uses the Gaussian distribution or Normal Probability Density Function in order to calculate P(Feature = x| Class = c). This function relies on knowing the standard deviation and mean of a feature. These values are approximated using the training data.

Results

My model was able to predict the outcome of the test data with a 68.4% accuracy at a speed of 0.84. I believe the accuracy of the model is very impressive especially for the speed at which the model can calculate predictions. After completing my naive bayes classifier and before implementing any data changes my accuracy was about 65%. However with all the trial and error tweaks I made to the data set my accuracy slowly improved to 68.4%. I would say the biggest improvements were adding the new difference features and normalizing the features' values. Similarly with the speed of my model I saw major improvements after changing my initial implementation. Originally I was calculating the mean and std within my function c. However, I created a new function (function b) which calculates all the means and stds initially in order to avoid redundant calculations. This improved the speed of my classifier by several orders of magnitude.

Challenges

Overall I think my progress in implementing the Naive Bayes Classifier was very smooth and I don't think any one challenge completely discouraged me. However, there were two notable obstacles that I spent the most time and effort on.

The first was figuring out how to implement the Classifier in general. As with most large projects, I always find it daunting because I am very uncertain where to start. I feel like I always get caught up in worrying about whether or not I am implementing the code correctly, or completing the project in the most efficient way. As a result, setting up my python packages, and jupyter notebook environment proved to be a little confusing. However after looking through some videos online I was able to find valuable instruction on how to set up a proper environment and then write a basic implementation of a Naive Bayes Classifier.

The second obstacle was figuring out how to improve accuracy after making sure my Naive Bayes model was implemented correctly. This took a long time and a lot of trial and error

messing with code to use the correct features. Overall I don't necessarily think my methodology for this part of the project was the most efficient because I would create a lot of messy code with comments and such. Furthermore, the accuracy with the validation and test data was not necessarily correlated so I would say I was just trying to make the model fit for the test data and not necessarily make the best overall model.