Bisman Sodhi
bisman@ucsb.edu
337326
Architecture

My program doesn't contain classes. It only contains functions. I implemented the following functions:

Def editTrainFile("training.txt"):

This function reads the csv file. Since the txt file doesn't have names for the columns, this function adds names to the columns. It gives all the non-numerical columns like location, WindGustDir, numerical values. It also deletes the columns that are highly correlated. It returns the modified csv file

It takes 0.0250089168548584 to read the file.

Def editTestFile("testing.txt"):

This function reads the csv file. Since the txt file doesn't have names for the columns, this function adds names to the columns. It gives all the non-numerical columns like location, WindGustDir, numerical values. It also deletes the columns that are highly correlated. It returns the modified csv file

It takes 0.0029859542846679688 to read the file.

Def prior(dataset, attributeName):

The dataset parameter takes in the txt file from which it reads the data.

The attributeName parameter takes the name of the column for which it has to calculate the prior. It uses datasest.groupby("RainTomorrow").get_group(i) to get the number of yes and no. To get the probability it divides by the len(dataset). The function returns an array of 2 elements with probability of yes and no.

Def calc(x, mean, std):

Implemented the gaussian distribution pdf formula

Def llhood(dataset, attributeName, attributeData, className, classLabel):

This function takes in the dataset as the first parameter.

Second Parameter = attributeName for which we will calculate the likelihood

Third Parameter = attributeData is the number for which we will calculate the likelihood

Fourth Parameter = className is the name of the prior class

Fifth Parameter = classLabel is the value of className like P(x | RainTomorrow = Yes)

This function uses the calc function and return the likelihood

Def gnb(dataset, attribute, className):

First parameter: dataset is the text file that it reads

Second Paramenter: attribute is all the columns of the dataset except the last one

Third Parameter: ClassName is the column for which we are prdicting i.e. RainTomorrow

This function iterates over every column in the attribute and calculates the likelihood of each column using the llhood function.

It stores the likelihood of each column in an array called likelihood. It calculate the likelihood given yes and likelihood given no and return the greater of the two.

It returns and array of 1s and 0s

Def acc(inTest, myPred):

First Parameter: inTest stores the actual result

Second Parameter: myPred is the array returned from function gnb

Def main():

This function executes the code

It reads in the arguments from the command line and the calls the editTrainFile and editTestFile function to modify the txt files. Two variable are initiated X_test and Y_test. X_test contains all columns of the test file except the last column. Y_test contian the last column of the test file. Then it calls the gnb(trainingData, X_test, className="RainTomorrow").

This returns an array that predicts whether will it rain tomorrow or not for the testing dataset.

There are some other helper functions in my program, but I decided not to use them. After implementing all the functions, I call main() and it executes the program.

Preprocessing:

I deleted the columns that have high correlation from both the training and the testing dataset. I found these columns by generating a heap map. I calculate the mean and standard deviation of each column in the llhood() function, which returns the likelihood of using the gaussian normal distribution.

Model Building:

I train the Naïve Bayes classifier using the Gaussian Naïve Bayes distribution and calculating the mean standard deviation of each column. Then I plug in the data from the training set in the formula for the gaussian naïve bayes distribution. I compare the posterior of given it will rain and given it will not rain and choose the maximum of those two.

Results:

It takes 0.0250089168548584 sec to read the training.txt file. Although the time is not exactly this every time I run my program, it is about the same.

It takes 0.0029859542846679688 sec to read the testing.txt file. Although the time is not exactly this every time I run my program, it is about the same.

It takes about 452.56448101997375 second to run the classifier on the training.txt file with an accuracy of 77.01

It takes about 60.27830505371094 second to run the classifier on the testing.txt file with an accuracy of 77.91

Challenges:

Some of the challenges I faced were figuring out how to implement functions when the columns had no name. In class when we learned naïve bayes all examples had name like P(Alarm|John, !Mary) so when I was thinking of writing functions, I kept thinking in terms of column names. I used name="Location…" in the read_csv function to add the name of columns. I also spent a lot of time figuring out how to use the non-numerical data. I decided to convert it to numerical data

I haven't used python before so I googled a lot about python syntax. I mostly tried to make my model based on the formula given in lecture and discussion slides.

Weakness:

I generated a heat map to see which columns were highly correlated. Once I found them I removed those columns from the training and testing dataset. I also made a correlation function which would return the names of the column with correlation above 90%. Then I removed these columns from the training and testing dataset. I used either heatmap or the correlation function, but not both at the same time. When I used my correlation function to remove columns with correlation above 90%, then my accuracy went down compared to when I manually removed columns by looking at the heatmap. This could be due to some weakness in the model because I had expected that removing columns using the correlation function will give better results. I could look for other methods instead of correlation to remove redundant data and improve the accuracy.

I haven't accounted for outliers so removing outliers could improve the accuracy of the model.

I'm using the gaussian distribution but using a different distribution could also be helpful like the poisson distribution.