Gabriella Rich 10/06/2025

Assignment 1 Applied Machine Learning

I am using the UCI Heart Disease dataset to predict whether a patient has heart disease based on medical attributes. Early detection of heart disease is important because it helps the doctors and medical professionals come up with the correct treatment plan based off of patient attributes and decreases risk over time in preventing complications and extending life expectancy. By training machine learning models, we can predict whether a patient has heart disease or Is at risk for heart disease in order to enable enhancements in treatments and advance the overall research linked to heart disease. In this dataset there is a mixture of different types of variables. It records a patients age, gender, chest pain, blood pressure, major blood vessels, presence of blood disorders, cholesterol, fasting blood sugar, maximum heart rate, exercise relative to rest, slope, resting electrocardiographic results, exercise-induced angina, and the target. The target represents our predicted attribute where 0 indicates no heart disease while 1 indicates the presence of heart disease. This is a diverse set of variables that contains categorical, integer and real data types.

Due to the fact that this dataset has so many different variables, there is the risk of noisy and unclean data. For example, two major variables ca and thal contain missing values in our data set, so I first set out to clean this up. I wasn’t sure how to go about this because there are several different methods to handle missing values in data. I decided on imputing the data that was missing. Data imputation is the process of replacing the missing data points with estimated values based on prior data in order for the model to be able to fully analyze. I used cat_imputer = SimpleImputer(strategy="most_frequent") to replace to NaN values with the most frequent value

in that column and transform the column back into our data frame X. I repeated the processes for the other missing variable as well. I came up with this process on my own as I decided this was best practice for replacing the missing values because it preserves our data values and does not get rid of data necessary for analysis. Next, I moved to analyze the noisy data in this dataset. Since this is a relatively small data set, I decided myself on one-hot encoding the categorical variables. One-hot encoding is a technique that converts the categorical data into a numerical format into a binary format while still retaining the integrity of the data set. Pictured below is what the dataset looked before encoding versus after on hot encoding.

```
print(X)

     age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0     63    1   1       145   233    1        2      150      0      2.3
1     67    1   4       160   286    0        2      108      1      1.5
2     67    1   4       120   229    0        2      129      1      2.6
3     37    1   3       130   250    0        0      187      0      3.5
4     41    0   2       130   204    0        2      172      0      1.4
..   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
298   45    1   1       110   264    0        0      132      0      1.2
299   68    1   4       144   193    1        0      141      0      3.4
300   57    1   4       130   131    0        0      115      1      1.2
301   57    0   2       130   236    0        2      174      0      0.0
302   38    1   3       138   175    0        0      173      0      0.0

     slope   ca  thal
0        3  0.0   6.0
1        2  3.0   3.0
2        2  2.0   7.0
3        3  0.0   3.0
4        1  0.0   3.0
..     ...  ...   ...
298      2  0.0   7.0
299      2  2.0   7.0
300      2  1.0   7.0
301      2  1.0   3.0
302      1  0.0   3.0

[303 rows x 13 columns]
```

```
   age  trestbps  chol  thalach  oldpeak   ca  sex_1  cp_2  cp_3  cp_4  fbs_1  \
0   63       145   233      150      2.3  0.0    1.0   0.0   0.0   0.0    1.0
1   67       160   286      108      1.5  3.0    1.0   0.0   0.0   1.0    0.0
2   67       120   229      129      2.6  2.0    1.0   0.0   0.0   1.0    0.0
3   37       130   250      187      3.5  0.0    1.0   0.0   1.0   0.0    0.0
4   41       130   204      172      1.4  0.0    0.0   1.0   0.0   0.0    0.0

   restecg_1  restecg_2  exang_1  slope_2  slope_3  thal_6.0  thal_7.0
0        0.0        1.0      0.0      0.0      1.0       1.0       0.0
1        0.0        1.0      1.0      1.0      0.0       0.0       0.0
2        0.0        1.0      1.0      1.0      0.0       0.0       1.0
3        0.0        0.0      0.0      0.0      1.0       0.0       0.0
4        0.0        1.0      0.0      0.0      0.0       0.0       0.0
```

Now that the data was properly encoded, I could move on to standardization. I had toruble deciding on whether or not to standardize my categorical variables or conduct normalization processes on my data set. I used ChatGPT and it recommended to only standardize the continous

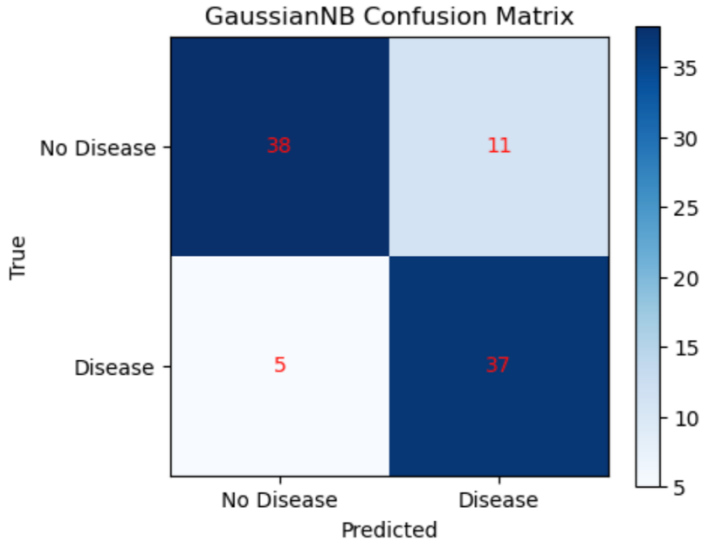
features because binary features are intuitive to the # audience receiving it (doctors/nurses/med professionals). Standardixation scaled the binary columns to have mean of 0 and variance of 1 to ensure that the variables contribute to the model that is sensitve to feature scaling. After I finished the standardization processess of my encoded data frame, it was time to build upon this knowledge and build the Naïve Bayes Model.

For this model, I used the Gaussian NB because it is used for continuous features (like in our dataset) because it assumes the conintuous features follow a normal distribution. Using Professor Luo's demo code on Gaussian NB, I was able to set the parameters to match my own code above and work on outputting some visualizations and accuracy scores of the Naïve Bayes model. I used a confusion matrix and classification report along with single sample posterior histogram to reflect the presence of heart disease (Predicted vs. Actual).

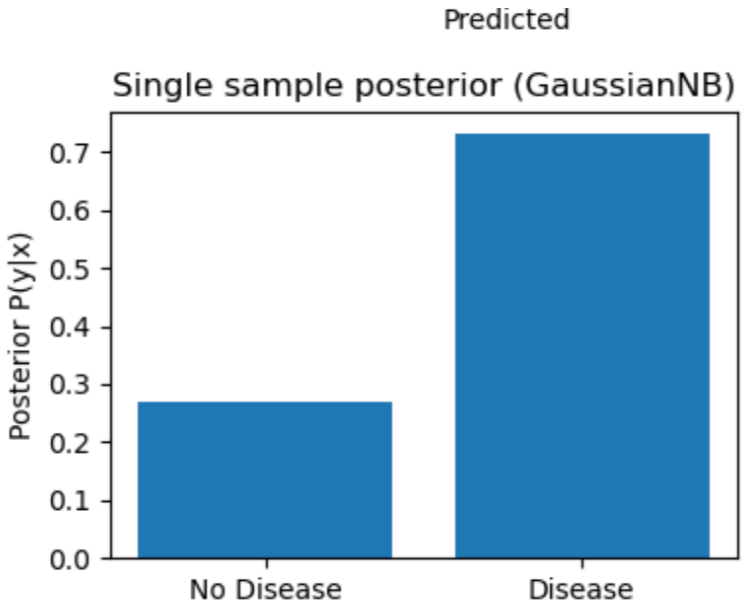
```
Accuracy: 0.8241758241758241
              precision    recall  f1-score   support

           0       0.88      0.78      0.83        49
           1       0.77      0.88      0.82        42

    accuracy                           0.82        91
   macro avg       0.83      0.83      0.82        91
weighted avg       0.83      0.82      0.82        91
```

From the classficiation report above, you can see that out of our sample, it correctly predicts the detection of heart disease 82% of the time. Preciion represents how many were corerct out of the samples, recall shows how many were correctly predicted (sensitvity), F1-score is the mean of precision and recall which balances the two and gives our overall accuracy score. I also added a confusion matrix. The confusion matrix evaluates how the model perfomed by comparing the

predicted values against the actual values of the UCI dataset to know whether or not to classify it as 0 (no heart disease) or 1 (heart disease detected).

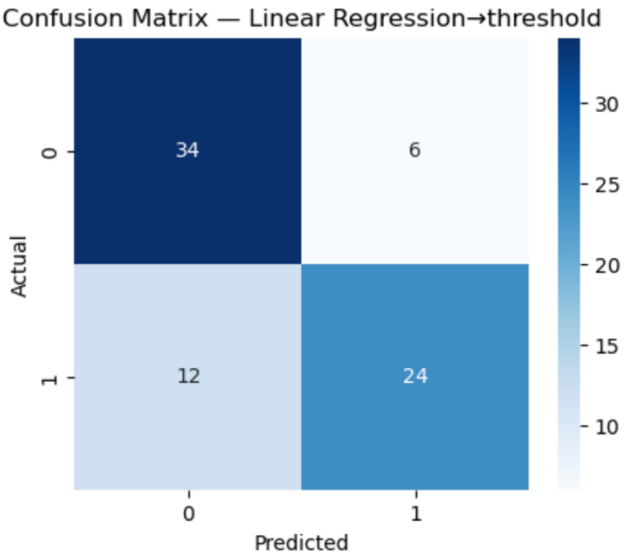


Along with this, I added in a posterior probability histogram which measures the Gausian NB model's confidence that the patient belongs to either no disease or disease class based on the patient's given data. It gives a nice visualizatiion as to why the prediction by the model was made and it also can help identify the edge cases that may have been misclassified by the model.



Following these steps, I moved on to building a linear regression model applied as a classifier with a threshold at 0.5. I used Chat GPT for this step along with the professors code to ensure that I was buidling the most accurate model to the data.

```
LinearRegression (0/1) accuracy: 0.763

Classification report (Linear Regression):
              precision    recall  f1-score   support

           0       0.74      0.85      0.79        40
           1       0.80      0.67      0.73        36

    accuracy                           0.76        76
   macro avg       0.77      0.76      0.76        76
weighted avg       0.77      0.76      0.76        76
```



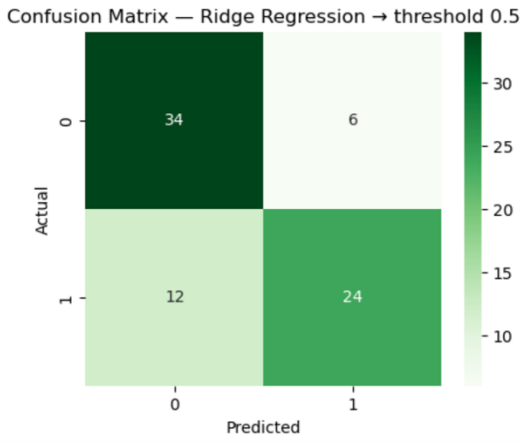
Based on the output, I was able to analyze that the linear regression model was not as accurate as my NB model, which lead me to believe that linear regression is that the right model for this data set. The model misses a significant number of heart disease patients which is critical for assessments of patients in medical scenarios. I wanted to work through how to make it the linear regression model more accurate using LASSO and Ridge techniques applied in class. I asked chat GPT to design code to generate a classification report along with a confusion matrix for both techniques to compare functionality. Lasso regression performs both variable selection and

regularization to enhance the prediction accuracy of the linear regression model and Ridge regressions adds an L2 regularization technique used to correct overfitting. As you can see below from the visualizations of these models using a confusion matrix, there wasn't much of difference using either technique which led me to believe that linear regression was not the correct model for this dataset. Overfitting does not appear to be the main issue onto why the linear regression model is underperforming so there are other factors present that are causing the linear regression model to be unsuccessful in its predictions.

Ridge Regression (0/1) accuracy: 0.763

Classification report:

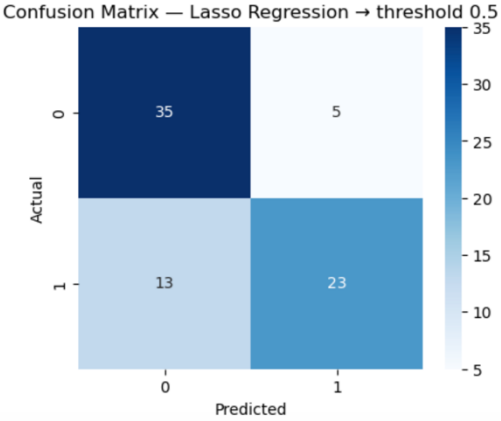
	precision	recall	f1-score	support
0	0.74	0.85	0.79	40
1	0.80	0.67	0.73	36
accuracy			0.76	76
macro avg	0.77	0.76	0.76	76
weighted avg	0.77	0.76	0.76	76



Lasso Regression (0/1) accuracy: 0.763

Classification report:

	precision	recall	f1-score	support
0	0.73	0.88	0.80	40
1	0.82	0.64	0.72	36
accuracy			0.76	76
macro avg	0.78	0.76	0.76	76
weighted avg	0.77	0.76	0.76	76



To conclude, this was a journey in creating predictive machine learning model that accurately predicts the presence of heart disease in a patient in the UCI Data Set. Prior to encoding along with standardization, I ran the Gaussian NB on the unclean data (after imputation) and was shocked that my accuracy score was 28%. Prior to this, I underestimated the effects of noisy data. I then retried several methods, worked to reclean using the methods taught in class (for example getting rid of rows and columns with missing values), and reduce noisy

data and still the model was only running at a measly 52% accuracy. I had to ask Chat GPT to work with me on debugging the model in order to achieve my results. Chat GPT was able to correct my mistakes and informed me that although I was on the right track with my thinking, I had to do more in order to clean up my data so my model could be more precise.