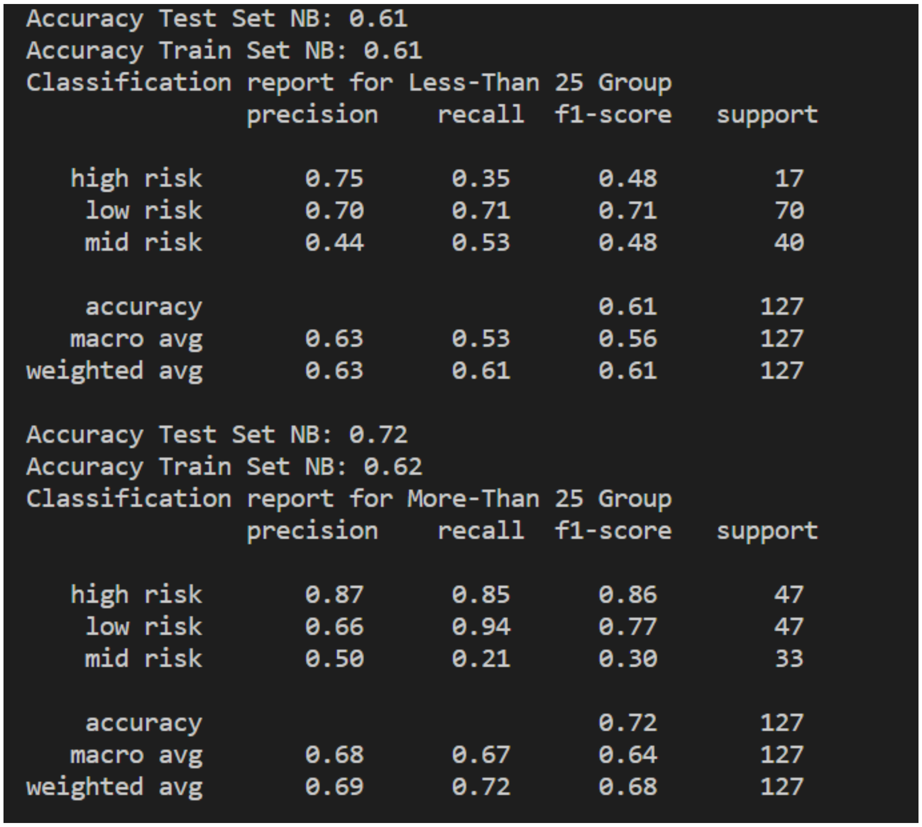
Valerie Melland
3/20/2023
Statistical Learning and Neural Networks

Homework 3: Decision Tree and Naïve Bayes Results

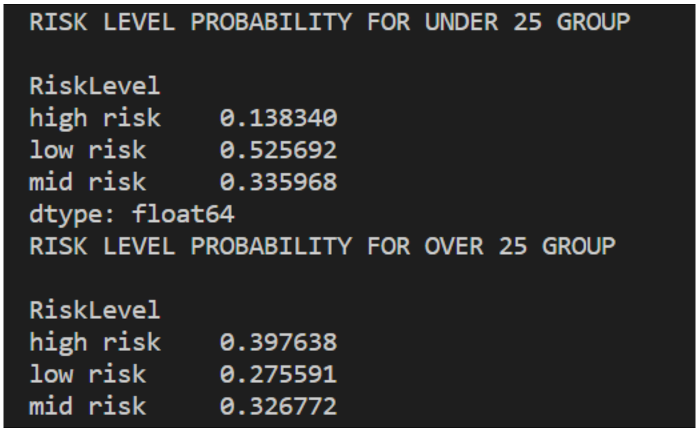
For this project, I used the Maternal Health Risk Data Set from UCI, which is found at this link: UCI Machine Learning Repository: Maternal Health Risk Data Set Data Set. The data has been collected from different hospitals, community clinics and maternal health care centers from rural Bangladesh. There are 1014 samples in the dataset and seven attributes. The attributes recorded are Age, Systolic Blood Pressure as SystolicBP (upper value of blood pressure in mmHg), Diastolic BP as DiastolicBP (lower value of blood pressure in mmHg), Blood Sugar as BS in terms of molar concentration, Body Temperature as BodyTemp, Heart Rate in beats per minute, and Risk Level. These features are all significant risk factors for maternal mortality. The Naïve Bayes model classifies the data into groups of low, mid, and high-risk based on the seven different features provided in the data set.

First, I created a NB model with the entire dataset and then I tried splitting the dataset into groups of people over 25 years old and less than 25 years old. Below is the classification reports for the data when splitting the data in two sets, one with people who less than 25 and the other is with people who are more than 25.



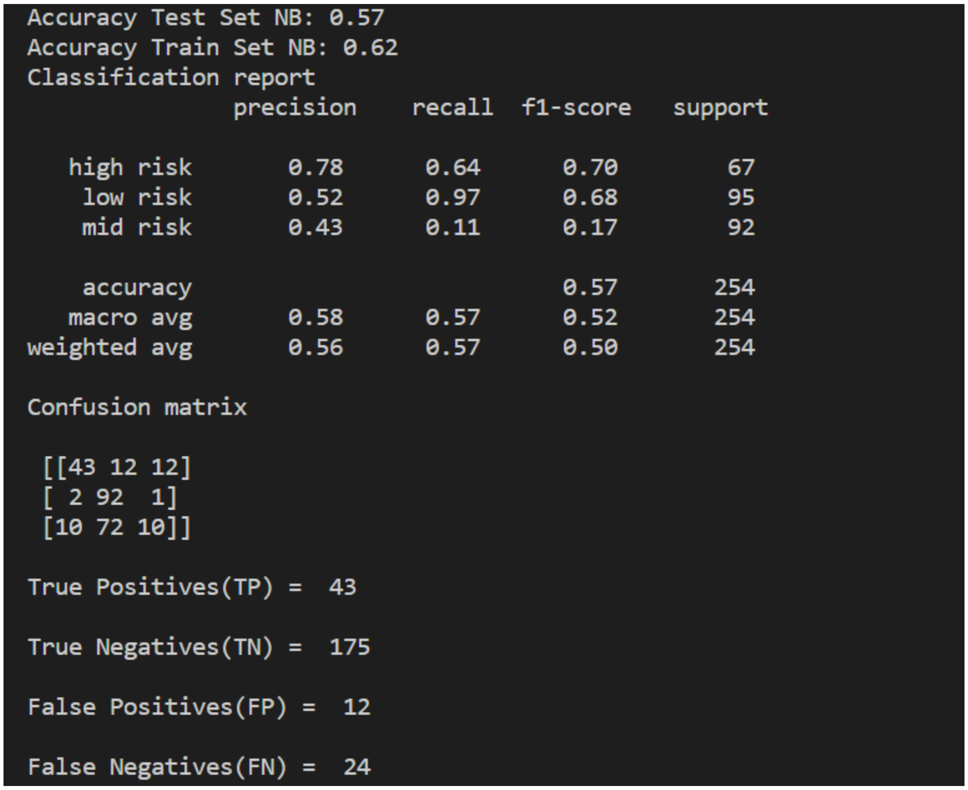
```
Accuracy Test Set NB: 0.61
Accuracy Train Set NB: 0.61
Classification report for Less-Than 25 Group
              precision    recall  f1-score   support

   high risk       0.75      0.35      0.48        17
    low risk       0.70      0.71      0.71        70
    mid risk       0.44      0.53      0.48        40

    accuracy                           0.61       127
   macro avg       0.63      0.53      0.56       127
weighted avg       0.63      0.61      0.61       127

Accuracy Test Set NB: 0.72
Accuracy Train Set NB: 0.62
Classification report for More-Than 25 Group
              precision    recall  f1-score   support

   high risk       0.87      0.85      0.86        47
    low risk       0.66      0.94      0.77        47
    mid risk       0.50      0.21      0.30        33

    accuracy                           0.72       127
   macro avg       0.68      0.67      0.64       127
weighted avg       0.69      0.72      0.68       127
```

I also included calculations for the probabilities of risk level for the data when split into the lower/more than 25 groups. You can see that there is a higher probability of people who are high risk in the over 25 group. There are more factors when classifying low and high risk, but you can see that there is a possible correlation between age and the risk level.



```
RISK LEVEL PROBABILITY FOR UNDER 25 GROUP

RiskLevel
high risk    0.138340
low risk     0.525692
mid risk     0.335968
dtype: float64
RISK LEVEL PROBABILITY FOR OVER 25 GROUP

RiskLevel
high risk    0.397638
low risk     0.275591
mid risk     0.326772
```

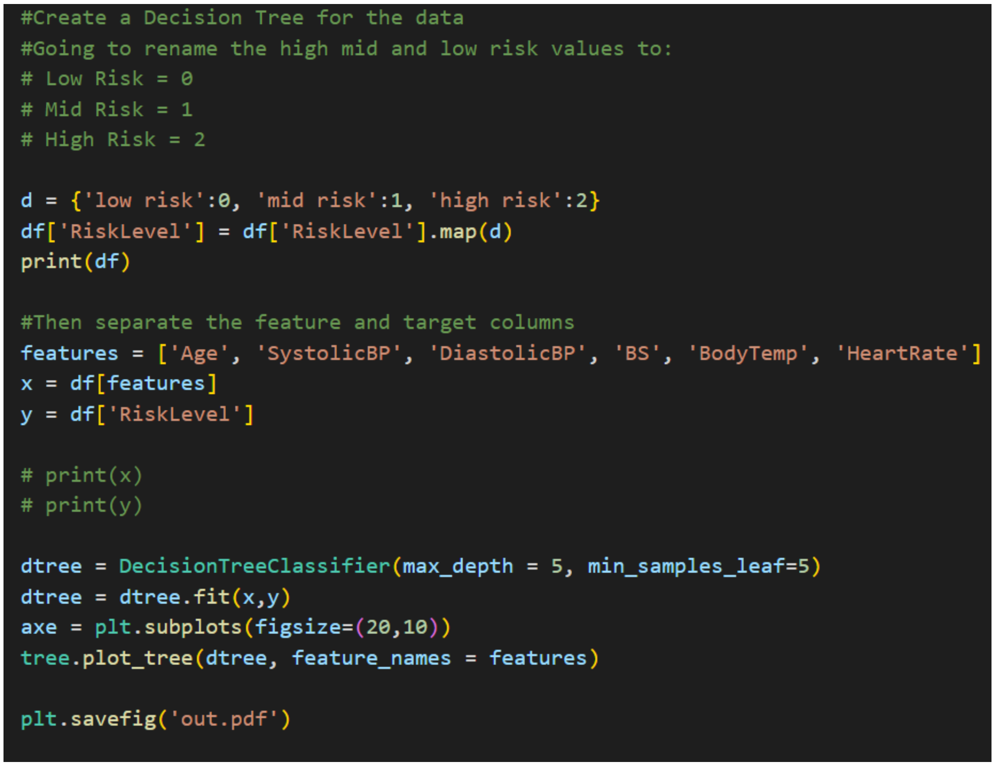
I compared the results from the NB for the complete dataset, to the NB for the dataset with ages more than 25 and less than 25 split into two datasets.



```
Accuracy Test Set NB: 0.57
Accuracy Train Set NB: 0.62
Classification report
               precision    recall  f1-score   support

   high risk       0.78      0.64      0.70        67
    low risk       0.52      0.97      0.68        95
    mid risk       0.43      0.11      0.17        92

    accuracy                           0.57       254
   macro avg       0.58      0.57      0.52       254
weighted avg       0.56      0.57      0.50       254

Confusion matrix

 [[43 12 12]
 [ 2 92  1]
 [10 72 10]]

True Positives(TP) =  43

True Negatives(TN) =  175

False Positives(FP) =  12

False Negatives(FN) =  24
```

You can see from the reports that there is a higher accuracy rating for NB when splitting the dataset into people less than and more than 25 years old. It is beneficial to get a better classification rating to split the dataset into age groups of more than 25 and less than 25.

For the decision tree, I also used a similar method and compared the decision trees using the entire dataset and the split dataset. I also tried using a train-test split method of creating a decision tree and compared that to a more straightforward method of creating the decision tree.

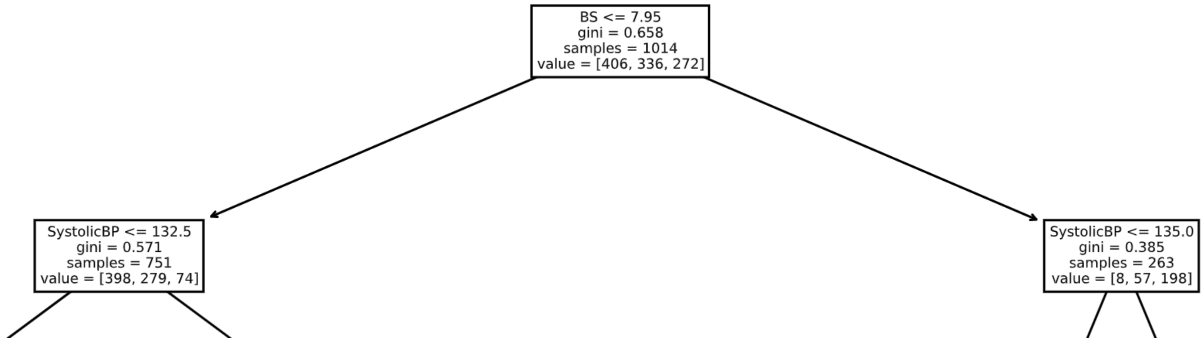


```
#Create a Decision Tree for the data
#Going to rename the high mid and low risk values to:
# Low Risk = 0
# Mid Risk = 1
# High Risk = 2

d = {'low risk':0, 'mid risk':1, 'high risk':2}
df['RiskLevel'] = df['RiskLevel'].map(d)
print(df)

#Then separate the feature and target columns
features = ['Age', 'SystolicBP', 'DiastolicBP', 'BS', 'BodyTemp', 'HeartRate']
x = df[features]
y = df['RiskLevel']

# print(x)
# print(y)

dtree = DecisionTreeClassifier(max_depth = 5, min_samples_leaf=5)
dtree = dtree.fit(x,y)
axe = plt.subplots(figsize=(20,10))
tree.plot_tree(dtree, feature_names = features)

plt.savefig('out.pdf')
```

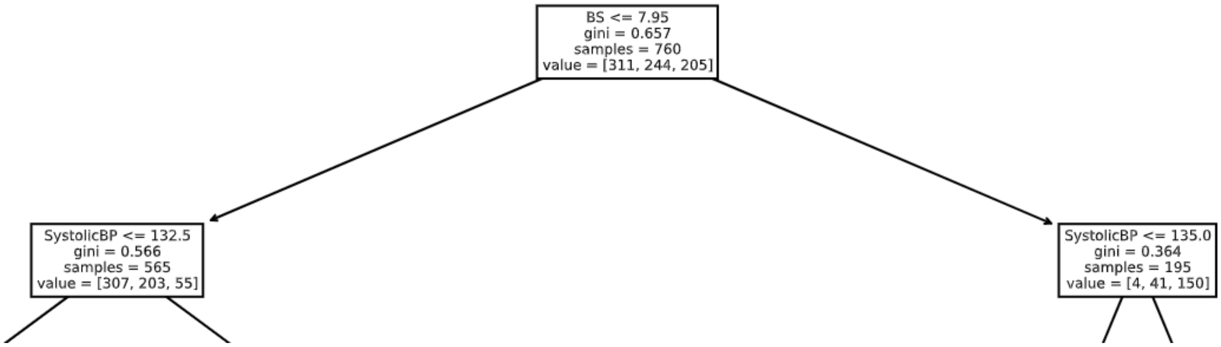
First I changed the risk levels to numeric data, with 0 representing low risk, 1 representing mid risk and 2 representing high risk. In the decision trees, this is displayed as value = [low risk, mid risk, high risk]. Which shows how many people are low risk, mid risk, and high risk. Then I split up the features and target columns into x and y variables to use for the decision tree. I set the max depth and the minimum samples per leaf to 5 to limit the size of the decision tree.

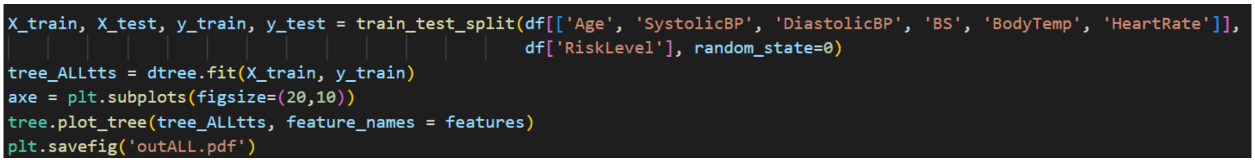
This tree I saved as a pdf called out.pdf. This I compared to the test-train-split version of the decision tree using the entire dataset. Which is the pdf file named outALL.pdf.

Both decision trees looked similar to each other, and I believe produced similar results. The tree did not seem very balanced. One side has way more samples than the other.



I’m not sure if this is a problem, but it seemed strange to me. The root of train-test-split version has less data and it seems like a more even split but it’s still not great.



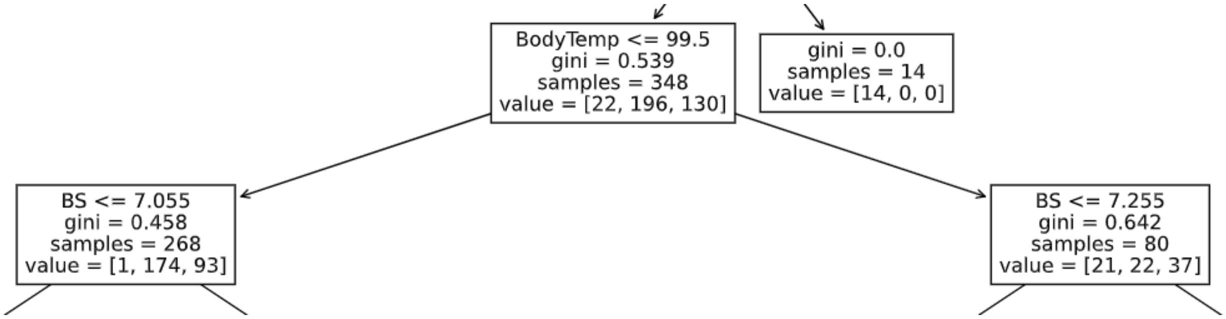


```
X_train, X_test, y_train, y_test = train_test_split(df[['Age', 'SystolicBP', 'DiastolicBP', 'BS', 'BodyTemp', 'HeartRate']],
                                                    df['RiskLevel'], random_state=0)
tree_ALLtts = dtree.fit(X_train, y_train)
axe = plt.subplots(figsize=(20,10))
tree.plot_tree(tree_ALLtts, feature_names = features)
plt.savefig('outALL.pdf')
```

This is where I create the train-test-split and the corresponding decision tree.

Then I created two more decision trees to compare to these using a train-test-split on the dataset split into people less than 25 and more than 25, these are saved in the files outlt25.pdf and outMT25.pdf. This creates a more even split of the data to use for the decision trees. Where less than 25 has 379 samples and more than 25 has 381 samples.

The largest split of the data for the less than 25 group was around the body temperature feature.



People with a body temperature lower than 99.5 has more mid and high risk people compared to the side that has body temperature higher than 99.5. But there is still a lot of people in the right side, and it is further organized using the BS and age features.

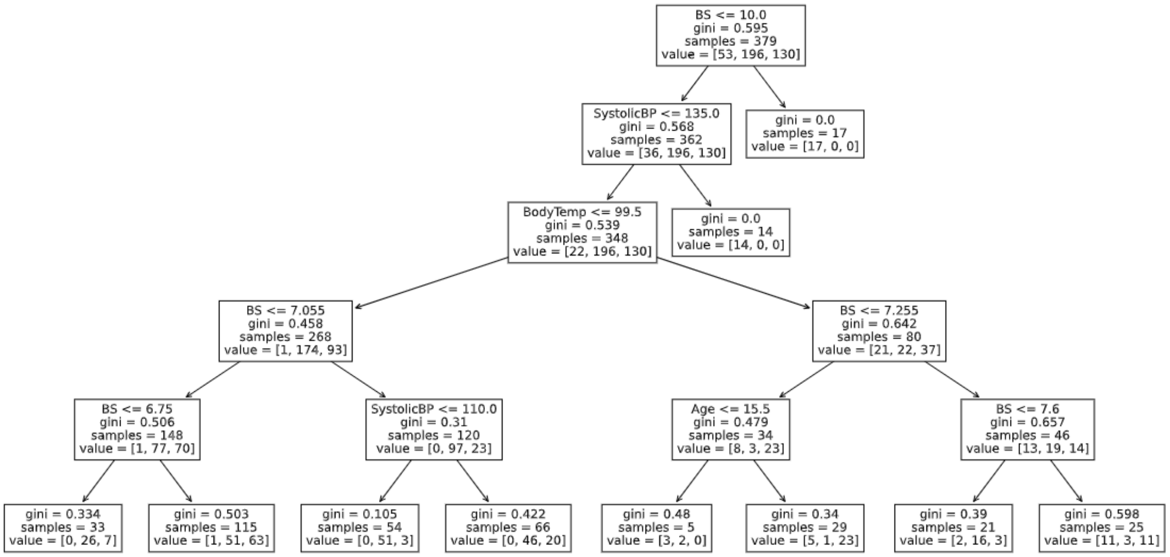


Figure 1: Decision tree using People Less Than 25

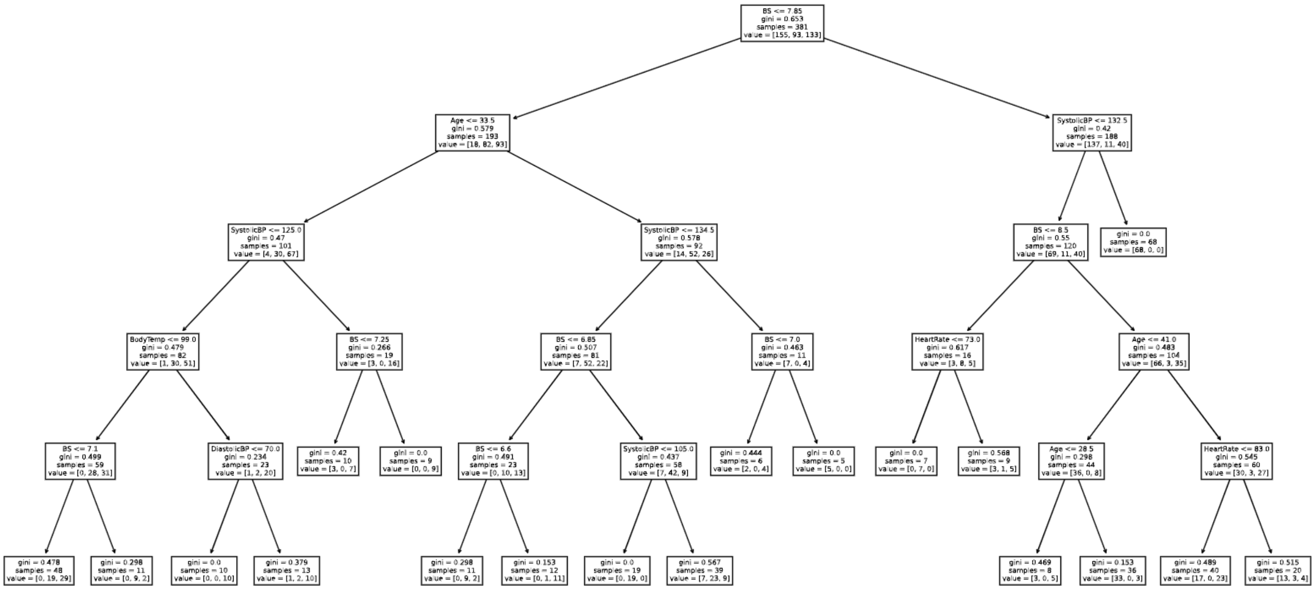


Figure 2: Decision tree using People More Than 25