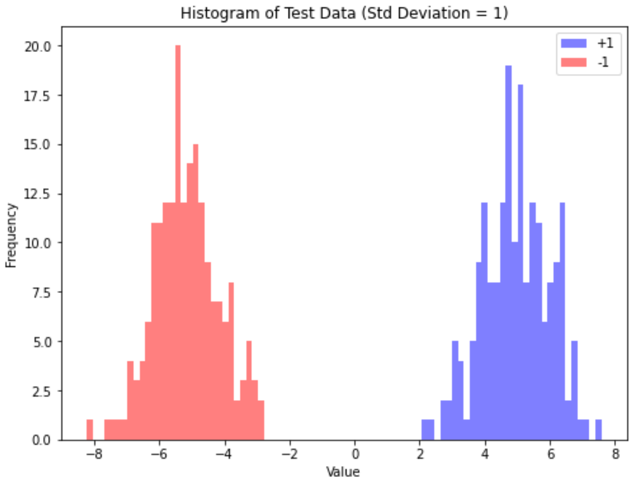
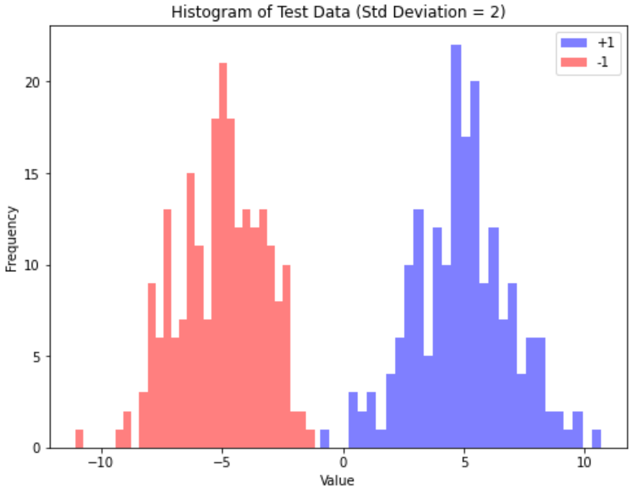
Shreeya Sampat

Lab 2 - Naïve Bayes Classification

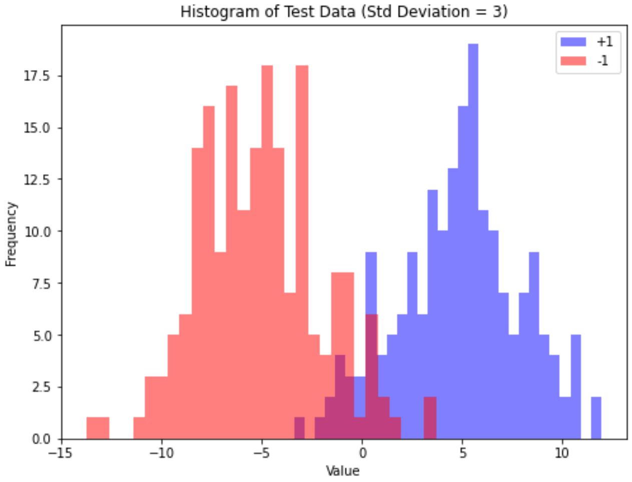
Step 6: Effect of standard deviation



Standard Deviation: 1, Misclassification Rate: 0.0



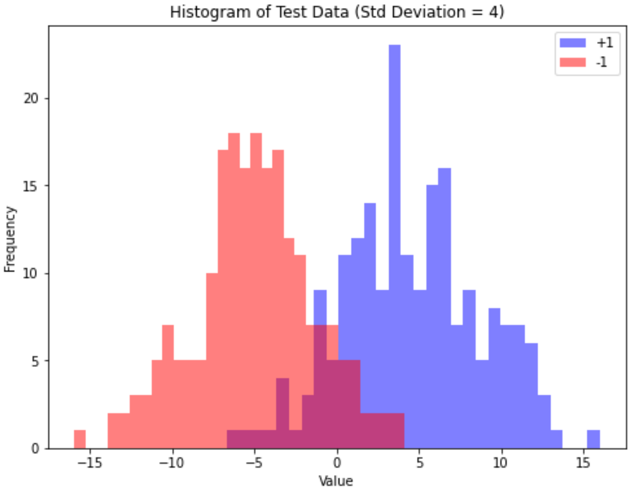
Standard Deviation: 2, Misclassification Rate: 0.0024271844660194173



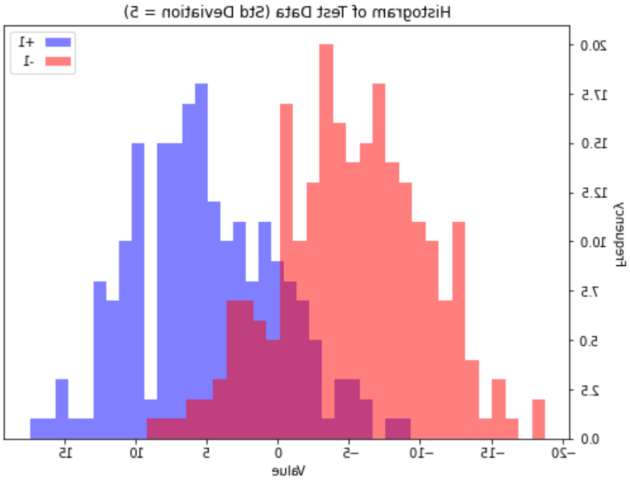
Standard Deviation: 3, Misclassification Rate: 0.06233766233766234

Shreeya Sampat

Lab 2 - Naïve Bayes Classification



Standard Deviation: 4, Misclassification Rate: 0.11407766990291263



Standard Deviation: 5, Misclassification Rate: 0.1655328798185941

As the standard deviation increases from 1 to 5:

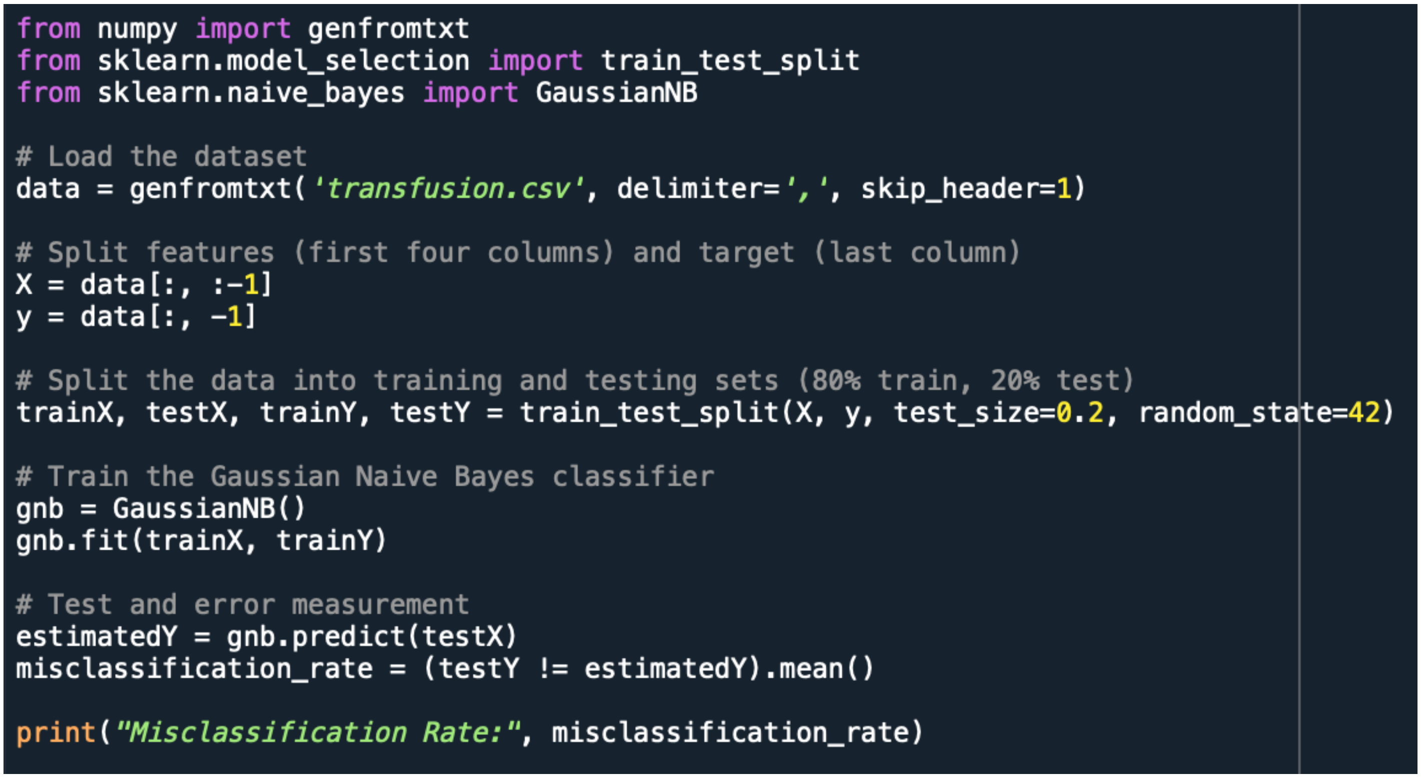
1) The misclassification rate tends to increase.
2) Histograms of test data show increased overlap between classes.
3) Higher standard deviation leads to greater variability, making classification more challenging for Gaussian Naive Bayes.

The increase in standard deviation leads to greater overlap between classes in the feature space, making it more challenging for the classifier to accurately classify instances, resulting in higher misclassification rates.

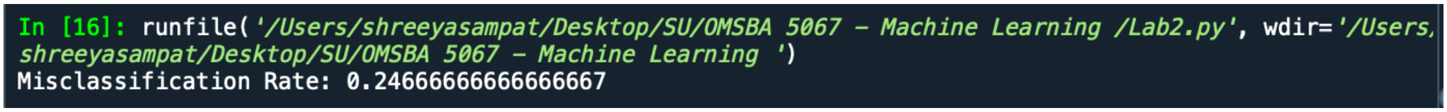
Step 7: Effect of train size

Reducing the ‘trainPortion’ from 0.8 to 0.2, 0.01, and 0.005 leads to less training data, which generally results in higher misclassification rates. Insufficient training data limits the model's ability to learn the underlying patterns effectively, leading to poorer generalization and higher misclassification rates on the test data. Adequate training data is crucial for building a robust and accurate classifier, as it allows the model to capture the true underlying distribution of the data and generalize well to unseen instances.

Step 9: Train and Test



```python
from numpy import genfromtxt
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB

# Load the dataset
data = genfromtxt('transfusion.csv', delimiter=',', skip_header=1)

# Split features (first four columns) and target (last column)
X = data[:, :-1]
y = data[:, -1]

# Split the data into training and testing sets (80% train, 20% test)
trainX, testX, trainY, testY = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the Gaussian Naive Bayes classifier
gnb = GaussianNB()
gnb.fit(trainX, trainY)

# Test and error measurement
estimatedY = gnb.predict(testX)
misclassification_rate = (testY != estimatedY).mean()

print("Misclassification Rate:", misclassification_rate)
```



```
In [16]: runfile('/Users/shreeyasampat/Desktop/SU/OMSBA 5067 - Machine Learning /Lab2.py', wdir='/Users/
shreeyasampat/Desktop/SU/OMSBA 5067 - Machine Learning ')
Misclassification Rate: 0.24666666666666667
```