Team Members: **Arnav Srivastava (sriva128) & Grayson Dufilho (gdufilho)**
GitHub Link to Repository: **Path #3 - https://github.com/ECEDataScience/miniproject-s24-grayduf.git**

Dataset Description:

We are working with the *digits* dataset from the sklearn.datasets module. This dataset is a collection of 8x8 pixel images of digits ranging from 0 to 9. We used the built-in function *train_test_split* from the sklearn library in order to split the *digits* dataset into training and test subsets. 60% of the data is allocated to the test subset while the remaining 40% is allocated to the training subset; this can be seen by one of the parameters of the *train_test_split* which states “test_size=0.6” referring to the test subset being 0.6 or 60% in size.

Method:

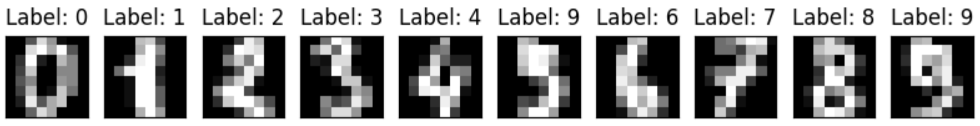
Overall, the method that we used to determine the answers to the analysis questions was to find the accuracy of the models that we were testing. This was accomplished by completing the OverallAccuracy function that took each model’s results and compared them to the actual results from the dataset. This in turn allowed us to determine the percentage of values that the model had gotten correct for each model. By looking at this “accuracy score” we were able to determine the performance of each model and precisely say which model performed better at each stage of the process. For example, the Gaussian model may have taken the original data set and failed to predict most of the correct results. This would be reflected in the accuracy score that gets returned as a decimal value as a very low value, say 0.05234. This value means that the Gaussian model has a 5.234% accuracy which shows that the model fails spectacularly at predicting which number the image is showing.

Results and Analysis:

Upon completion of the three models, we get the following output:

The overall result of the Gaussian model is 0.8007414272474513
The overall result of the K Nearest Neighbors model is 0.9545875810936052
The overall result of the MLP Classifier model is 0.9147358665430955

Original Data: [0,1,2,3,4,5,6,7,8,9] - K Nearest Neighbors Classifier Model



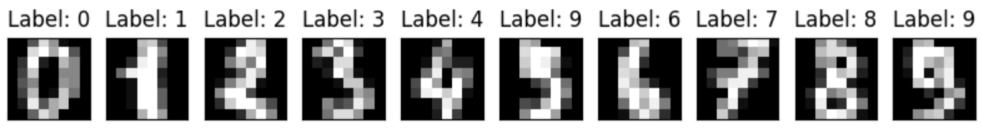
As discussed prior, the accuracy scores that are outputted can be interpreted as: 80.07% for the Gaussian model, 95.46% for the K Nearest Neighbors model, and 91.47% for the MLP Classifier model. Based on these percentages / accuracy scores, we can conclude that the Gaussian model performed the worst while the K Nearest Neighbors model performed the best.

In the next section of the project, the training data is “poisoned”. This means that the models are learning based on the training data, plus a secondary set of data comprised of random noise. This means that the new model is predicting what numbers the images are showing based on false training data. The creation of this “false” training data is achieved by generating random noise samples from a normal distribution with a standard deviation of 10, and the shape of the training data. The standard deviation is seen in the *noise_scale* variable, and the shape is kept constant to allow for the poison data to be combined with the training data correctly.

After the training data was poisoned, each model was then trained with the poisoned data and then tested to observe how accurate they would be. The results are then output in a decimal form showcasing the accuracy of each model. The accuracy scores of each model are found below:

The overall poisoned result of the Gaussian model is 0.8146431881371641
The overall poisoned result of the K Nearest Neighbors model is 0.6051899907321594
The overall poisoned result of the MLP Classifier model is 0.794253938832252

Poisioned Data: [0,1,2,3,4,5,6,7,8,9] - K Nearest Neighbors Classifier Model



For the Gaussian model, it can be seen that the accuracy actually increases slightly from 80.07% to 81.46%. This can be attributed to the fact that the Gaussian model assumes that the features (pixels of the images) are conditionally independent of the class labels. This means that with clean data, it can reasonably predict the numbers and perform decently. This also means that with noised data, it does not reasonably impact the performance of the model, and the performance of the model with the noised data depends on the noise and can be up to chance. This is seen here, as the Gaussian model increases its accuracy slightly.

Team Members: **Arnav Srivastava (sriva128) & Grayson Dufilho (gdufilho)**
GitHub Link to Repository: **Path #3 - https://github.com/ECEDataScience/miniproject-s24-grayduf.git**

For the K Nearest Neighbors model, the accuracy percentage degrades drastically once the model is trained with the poisoned data from 95.46% to 60.52%. This is because, with clean data, the KNN model makes predictions based on the majority class among the k-nearest neighbors. This is then thrown completely out the window when the noise is introduced as the noise causes the distances between data points to change and leads to misclassifications. This in turn decreases the accuracy of the model drastically to the point that it is barely passable.

The MLP Classification model has a high accuracy percentage when trained with clean data as seen with its accuracy score of 91.47%. This then degrades when trained with the noised data as the neural network that predicts the results is confused when given false data and struggles to achieve a high accuracy score, as seen with its new score of 79.43%. The tradeoff is that when trained with clean data, it achieves a high accuracy score and can predict the numbers in the images correctly almost all of the time.

The model that showed the most “robustness” after being trained was the Gaussian Naive Bayes model, as this model’s accuracy score practically stayed the same, increasing by 1.39%. The other two models did not follow suit, as the KNN model had the highest accuracy with the clean data and conversely had the lowest accuracy with the noised data, as seen by its score decreasing by 35.21%. While the MLP model had a high accuracy with the clean data, it performed about as well as the Gaussian model with the poisoned data with a decrease of 12.04%. This does not mean it was the most robust though, as being the most robust means that it changed the least, which the Gaussian model handily takes first in.

Now we attempt to denoise our poisoned data by using a denoising algorithm with the use of KernelPCA. Our denoising algorithm works in three main steps:

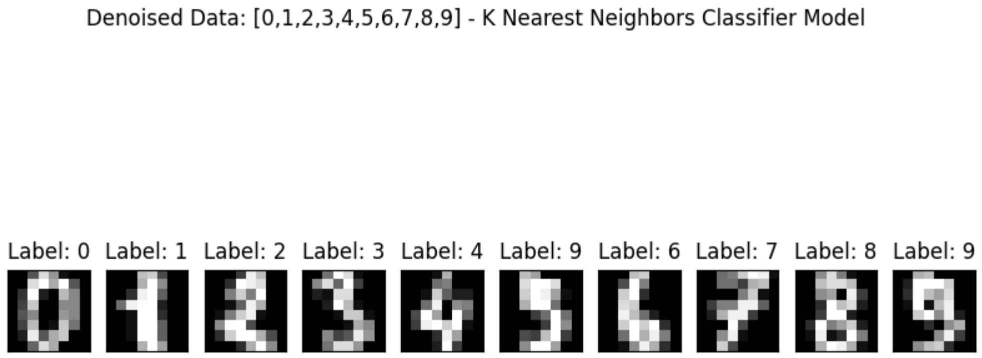
1) The *fit* method computes the necessary transformations
2) The *transform* method applies these transformations to X_train_poison_reshaped to get the reduced dimension data.
3) The *inverse_transform* method reconstructs the original data from the reduced dimension data; it discards the noisy components during reconstruction for the sake of denoising the data.

Upon running the denoising algorithm, we obtain our denoised data. Our denoised data is simply just a reconstruction of the poisoned data in the sense that the noise from earlier was detected by the algorithm and ignored upon reconstruction. So, let’s discuss how this denoising algorithm affected the accuracy scores of each model:

Poisoned Results:
The overall poisoned result of the Gaussian model is 0.8146431881371641

The overall poisoned result of the K Nearest Neighbors model is 0.6051899907321594
The overall poisoned result of the MLP Classifier model is 0.794253938832252

Denoised Results:
The overall denoised result of the Gaussian model is 0.8489341983317887
The overall denoised result of the K Nearest Neighbors model is 0.8266913809082483
The overall denoised result of the MLP Classifier model is 0.8257645968489342



As can be seen from the accuracy scores, every single model improved to some degree after the implementation of the denoising algorithm. This makes sense because the whole point of denoising is to reobtain some of the original lost accuracy. Below are the original accuracy scores and the denoised accuracy scores for the sake of comparison:

Original Results:
The overall result of the Gaussian model is 0.8007414272474513
The overall result of the K Nearest Neighbors model is 0.9545875810936052
The overall result of the MLP Classifier model is 0.9147358665430955

Denoised Results:
The overall denoised result of the Gaussian model is 0.8489341983317887
The overall denoised result of the K Nearest Neighbors model is 0.8266913809082483
The overall denoised result of the MLP Classifier model is 0.8257645968489342

As can be seen from these accuracy scores, the denoised accuracy scores are typically lower than the original accuracy scores, which makes sense because the reconstruction section of the denoising algorithm doesn't work perfectly (some noise may still remain). The Gaussian model shows contradictory results but, for reasons explained earlier, this is simply attributed to luck or pure happenstance.

Note: All of the above accuracy scores may or may not be completely correct due to differences in computers and timings. This means that the accuracy score might be off by a very small margin.