**Use Case Summary**

Medical professionals often have unique or unclear handwriting styles, making it difficult to accurately recognize and interpret handwritten characters and numbers.

Used the MNIST dataset to recognize handwritten digits in medical records because it serves as a benchmark dataset for digit classification tasks. MNIST dataset is clean, preprocessed and labeled making it easier to develop models.

The goal of this project is to create a smart system that can read and understand handwritten medical records, saving time and reducing mistakes caused by manual work. By using advanced computer algorithms like Naïve Bayes and K-Nearest Neighbours (KNN), the system will quickly and accurately turn handwritten data into digital format, making it easier for hospitals and clinics to manage large amounts of information and provide better care.

**Handwritten Digit Recognition Using MNIST Dataset**

Below are some of the data preprocessing steps that I did for the MNIST data:

After importing the csv data as pandas dataframe, I removed columns ‘Unnamed: 0’ and ‘index’. I converted the features (0, 1, …, 783) to a numpy array and named it as X\_train. Similarly created a numpy array of the labels column as y\_train.

**Exploratory analysis**

**Pixel Intensity Distribution**

To understand the pixel intensity values of all images in the train data, I first flattened the 2D numpy array into a 1D array with the .flatten() method. Then passed this flattened array to sns.histogram to visualize the distribution of the pixel intensity values. To understand the pixel intensity distribution of individual digits, I extracted label-wise data from the original data from labels 0, 1, and 8, and repeated the above steps to visualize the pixel intensity distribution for each of labels 0, 1, and 8.

Converting the intensity data to 28 x 28 Image

I used .reshape(28, 28) to transform the data into a 28x28 2D array. And used .imshow() method from matplotlib to plot the grayscale image of the reshaped array.

**Modelling**

I used the GaussNB class from last week’s sessions for building the multiclass classification model. Prior to fitting and predicting, I normalized the data (feature\_range=(0, 1)).

As part of hyperparameter tuning, I received 80% accuracy for training data and 81% for test data.

Then I used a non-Naive Bayes Classification algorithm for building another model for the normalized train and test data. I got 93% for training data and 91% for test data. So I tuned the epsilon value to 5e-3, then I got 95% for training data and 94% for test data.

I tried another classification model ‘K-Nearest Neighbours’. It took more time to predict both the train and test dataset. So I could not find the accuracy for that model.

From my modelling, I propose Non-Naive Bayes Classification for this MNIST Dataset.