50.038 Computational Data Science

2023 Jan-Apr Term 6 Project

Autonomous Triage Classification

Group: WatPandas

Name	Student ID
Kim Si Eun	1005370
Royden Yang	1005219
Samuel Chiang Wenbin	1005142

Table of Contents

Link to GitHub:

https://github.com/joenkim/Autonomous-Triage-Classification.git

Problem

Emergency Departments of hospitals have to provide 24-hour service for trauma and non-trauma surgical and medical emergencies. Triage is done by nurses to identify and prioritize patients with the most urgent needs to receive emergency service first.

However, with the lack of triage nurses on duty, flow of patients in the Accident and Emergencies Department (A&E) can bottleneck at the triage stage. Leading to longer waiting times for patients who require urgent treatment.

Patients with chest pain symptoms are an example in which triage plays a crucial role of discovering threatening cases of potential heart attack or aortic dissection. As such our solution is to use machine learning algorithms to develop a model for triaging patients with chest pain symptoms.

Datasets used

In this project, we used 3 datasets containing records of patients during triage. Two of which were csv files containing patients' details, vital measurements and conditions. The last dataset contained electrocardiogram (ECG) images of patients' ECG readings.

The first dataset we used was the "Hospital Triage and Patient History dataset", from:
https://www.kaggle.com/datasets/maalona/hospital-triage-and-patient-history-data

The second dataset used was the "Patient Priority for Clustering", from:
https://www.kaggle.com/datasets/hossamahmedaly/patient-priority-classification

The last dataset used was the "ECG image dataset", from:
https://www.kaggle.com/datasets/erhmrai/ecg-image-data

Evaluation Methodology

For all 3 datasets, we are aiming to train models that can accurately classify patients into the different priority levels of triage classes. We would thus focus mainly on the F1-Score of the models, as for triage the false negatives are of importance.

In this case false negatives would mean our model classifies a patient with life-threatening conditions, which should have a higher priority triage level, as a patient which does not require urgent treatment. Aiming for a higher F1-Score would minimise this error.

Dataset 1 : Hospital Triage and Patient History dataset

Dataset Details

This dataset originally consists of 59k rows by 972 columns worth of data, which we filtered to isolate relevant chest pain cases, resulting in 7k rows by 53 columns worth of data. The features in this dataset primarily include age, gender, blood pressure, heart rate, other vitals, chest pain presence, chief complaints, among others. Each patient is classified into 1 out of 5 ESI values; 1 representing the greatest level of emergency, whereby they need the most urgent attention, and 5 representing the least attention needed. The correlogram below focuses on drawing correlation from the symptoms detected on the patient as it depicted relatively more significant values than if we used the other features such as vitals or chief complaints.

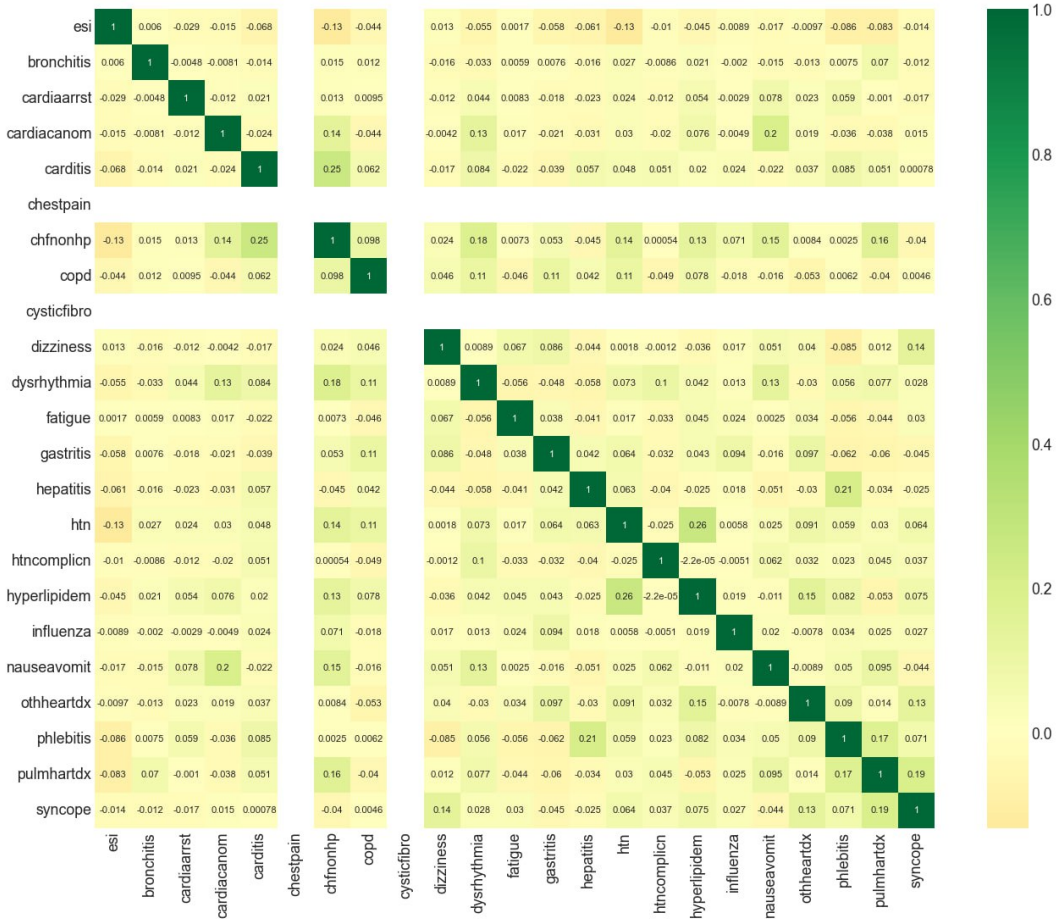


Figure 1.1 Correlogram of symptoms for Dataset 1.

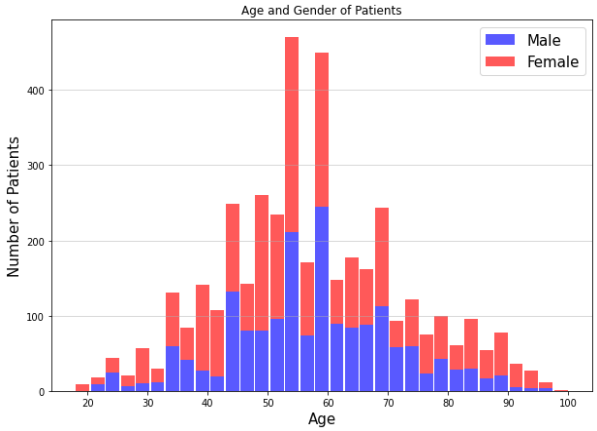


Figure 1.2 Age and Gender of Patients for Dataset 1.

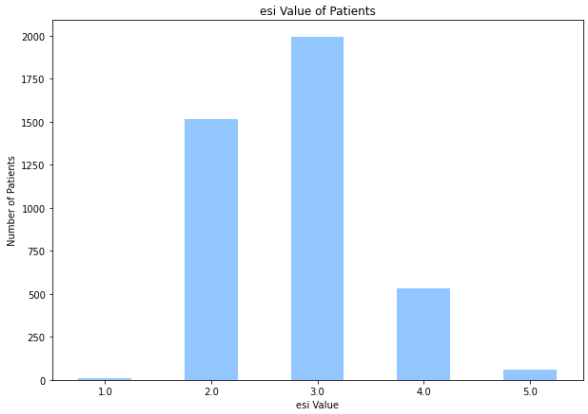


Figure 1.3 ESI value of Patients for Dataset 1

Figure 1.2 shows the age and gender distribution of patients in Dataset 1. Majority of patients within this distribution are between 40 to 70 years old, with a roughly greater number of female patients than male. Figure 1.3 shows the number of patients which were classified into each of the 5 ESI values. Majority of patients were classified at an ESI value of 3, with 2 being a close second. This suggests that most of the patients being admitted are generally in need of a greater urgency of treatment, but the number of patients who were admitted in a state of maximum emergency were minimal.

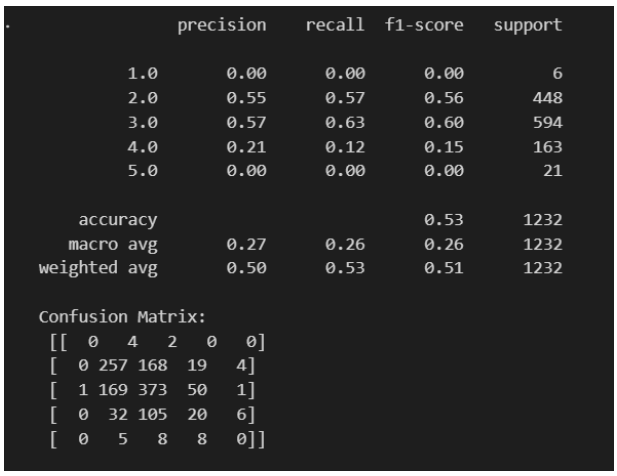
Data-preprocessing

The features in the dataset were cleaned to fill in the rows with empty values. For the symptoms and chief complaints, empty rows were filled with 0 to prevent runtime errors during training. For the vitals, it was assumed that rows without values meant that the vitals were at a normal healthy value, so they were automatically filled in with a random value between the range of usual healthy values. Outside of the three groups of features mentioned, everything else was removed since they showed virtually no relation with chest pain and were hence deemed as unnecessary to use for training.

Model and Results

We initially tried out various classification algorithms to model our dataset. We settled on Logistic Regression and Random Forest Classifier algorithms because they were able to show the highest F1-scores compared to the other algorithms as reflected in the images below (56% for ESI value 2.0 using RandomForestClassifier with n_estimators = 100, 64% for ESI value 3.0

using Logistic Regression). These models initially only accounted for the symptoms as features as a follow-up of what we talked about above.

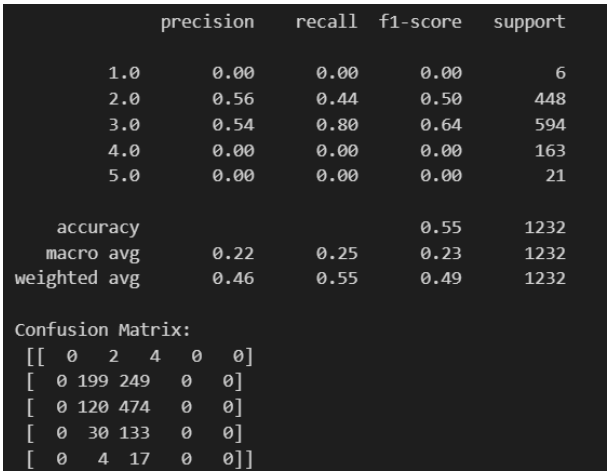


	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	6
2.0	0.55	0.57	0.56	448
3.0	0.57	0.63	0.60	594
4.0	0.21	0.12	0.15	163
5.0	0.00	0.00	0.00	21
accuracy			0.53	1232
macro avg	0.27	0.26	0.26	1232
weighted avg	0.50	0.53	0.51	1232

Confusion Matrix:

```
[[  0   4   2   0   0]
 [  0 257 168  19   4]
 [  1 169 373  50   1]
 [  0  32 105  20   6]
 [  0   5   8   8   0]]
```

Figure 1.4 RandomForestClassifier for Symptoms



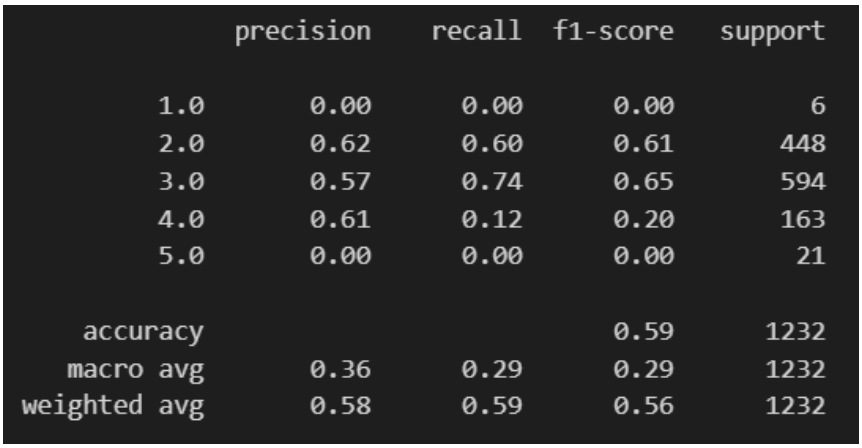
	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	6
2.0	0.56	0.44	0.50	448
3.0	0.54	0.80	0.64	594
4.0	0.00	0.00	0.00	163
5.0	0.00	0.00	0.00	21
accuracy			0.55	1232
macro avg	0.22	0.25	0.23	1232
weighted avg	0.46	0.55	0.49	1232

Confusion Matrix:

```
[[  0   2   4   0   0]
 [  0 199 249   0   0]
 [  0 120 474   0   0]
 [  0  30 133   0   0]
 [  0   4  17   0   0]]
```

Figure 1.5 LogisticRegression for Symptoms

We also used all the features of the original pre-processed dataset as part of another model, and surprisingly obtained slightly higher F1-scores with the use of RandomForestClassifier.



	precision	recall	f1-score	support
1.0	0.00	0.00	0.00	6
2.0	0.62	0.60	0.61	448
3.0	0.57	0.74	0.65	594
4.0	0.61	0.12	0.20	163
5.0	0.00	0.00	0.00	21
accuracy			0.59	1232
macro avg	0.36	0.29	0.29	1232
weighted avg	0.58	0.59	0.56	1232

Figure 1.6 RandomForestClassifier for all features

As seen from the figures above, the F1-scores obtained were not significantly high regardless of whichever features we adjusted to use for our model.

Discussion

One possibility to why our F1-scores obtained were not significant enough is due to having too many features being used for classification. Chief complaints were included in the training involving all the features of the dataset, but what the patient said when being admitted into the hospital might not have been very relevant to their actual diagnosis. Even with this consideration in mind, the model which performed slightly better was the one which used all of the features, compared to the other models which used the specific groups of features (symptoms, vitals, chief complaints). From this, we can conclude that Dataset 1 on its own is inconclusive in meeting our requirements for a solution to our problem statement.

Dataset 2 : Patient Priority for Clustering

Dataset Details

This dataset consists of 6962 rows by 17 columns of data. Each row is a patient record with values for age, gender, blood pressure, cholesterol levels, heart rate, glucose levels, and smoking status, among others. Each patient is classified into 1 among 5 colours; red, orange, yellow, green, and white. Red signifies life-threatening emergency and requires immediate attention, orange is very urgent, yellow is urgent, green is minor or minimal attention, and white means the patient can be dismissed without attention.

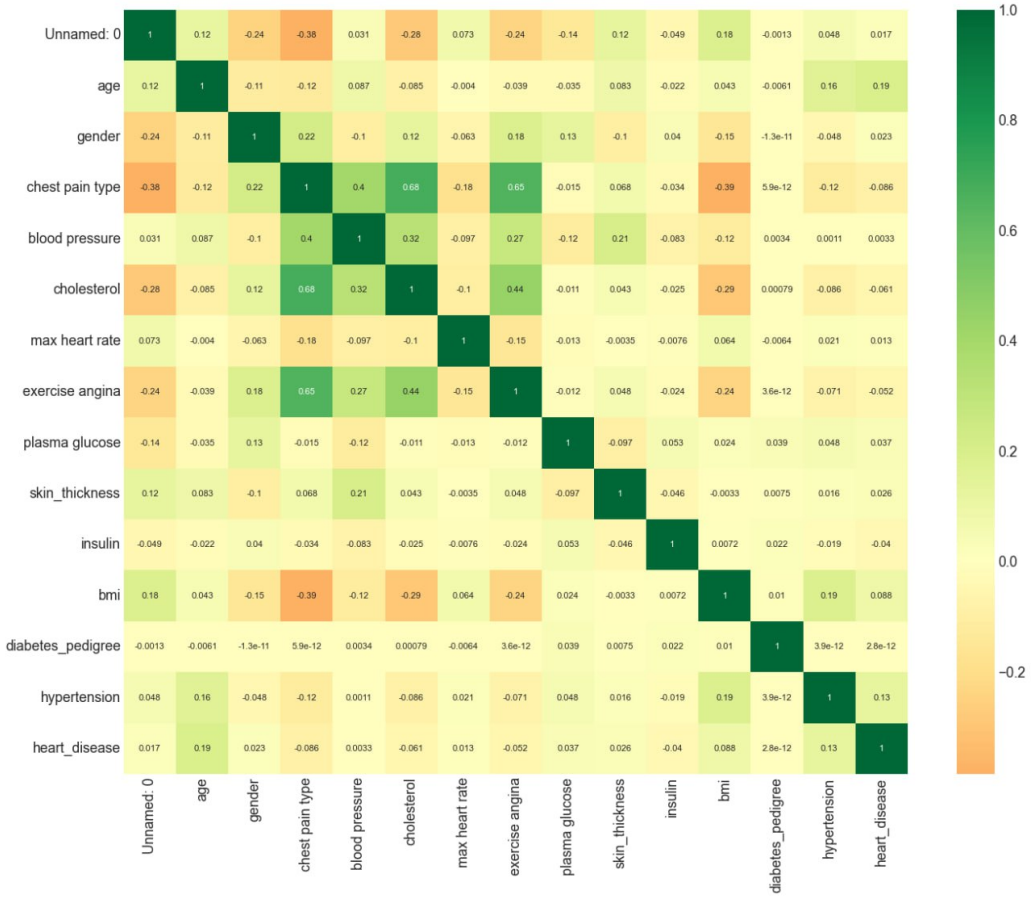


Figure 2.1 Correlogram of columns for Dataset 2.

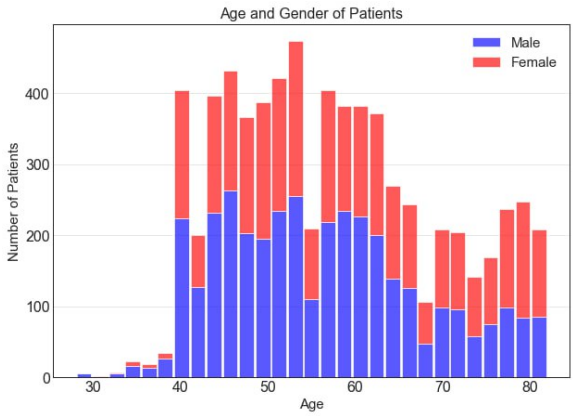


Figure 2.2 Age and gender of patients

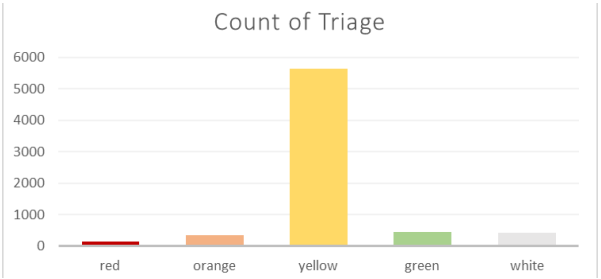


Figure 2.3 Triage count by colour

Figure 2.2 shows the age and gender distribution of patients in Dataset 2. Majority of the distribution is above 40 years old, with about equal distribution for male and female patients by age group. Figure 2.3 shows that the number of patients classified as yellow level makes up the majority of the data at over 5000 patients. This can be due to the fact that the number of urgent but not life-threatening cases is much higher than life-threatening emergencies in hospitals.

Data-preprocessing

The features in the dataset were cleaned to remove any rows with empty values. Furthermore, features we deemed unnecessary were not used in the training, such as residence type and chest pain type. One reason chest pain type was removed is that there was not enough information on how chest pain was classified for this dataset. The dataset only lists chest pain type as values of 0 to 4, without any information. Thus, not much information could be deduced from this column and therefore was not considered for training the model.

Models and Results

The initial step was experimenting with different classification algorithms learned. These include Decision Trees, K-Nearest Neighbours, Logistic Regression, Gaussian Naive Bayes, Support Vector Machine, Ada Boost, and Random Forest. While most algorithms showed generally good results with F1-Scores around 70~90%, Decision Trees seemed to show the highest F1-Score for all levels. It performed well at maximum depth of 10, reaching 78% F1-Score for red and even up to 100% F1-Score for green.

```
              precision    recall  f1-score   support

         red       0.76      0.79      0.78        33
      orange       0.81      0.85      0.83        93
      yellow       0.99      0.99      0.99      1695
       green       1.00      0.99      1.00       137
       white       0.94      0.97      0.95       131

    accuracy                           0.98      2089
   macro avg       0.90      0.92      0.91      2089
weighted avg       0.98      0.98      0.98      2089

Confusion Matrix:
 [[  26    2    3    0    2]
 [   4   79   10    0    0]
 [   2   16 1671    0    6]
 [   0    0    1  136    0]
 [   2    1    1    0  127]]
```

Figure 2.4 Classification report and confusion matrix for Dataset 2

As seen from Figure 2.4, F1-Scores as well as accuracy for all levels are quite high. However, Looking at the confusion matrix (red to white from top to bottom), we can observe that there are 2 patients of the red level who were misclassified as white. This situation should be avoided as red level is a life-threatening emergency and requires immediate attention, but white means the patient will be dismissed without attention. This can potentially harm the patient. Similarly, there are 2 patients of the white level misclassified as red. Classifying these patients as red will require additional unnecessary resources and time for doctors and nurses to attend to these patients even if they can be dismissed without treatment.

To avoid false negatives and increase our F1-Score, we attempted another approach. Red and orange levels were combined into one level called “immediate”. One reason was that this will

increase the number of datapoints of this classification, and another reason was to identify urgent patients better.

```
                precision    recall  f1-score   support

     immediate       0.81      0.93      0.87       126
        yellow       0.99      0.98      0.99      1695
         green       1.00      0.99      1.00       137
         white       0.96      0.96      0.96       131

      accuracy                           0.98      2089
     macro avg       0.94      0.97      0.95      2089
  weighted avg       0.98      0.98      0.98      2089

Confusion Matrix:
 [[ 117    7    0    2]
 [  24 1668    0    3]
 [   0    1  136    0]
 [   3    2    0  126]]
```

Figure 2.5 Classification report and confusion matrix with combined levels

This attempt showed higher overall F1-Scores, albeit still having a few false negatives.

To check if any further improvements can be made, our final attempt was combining dataset 1 and 2. Since they both classify patients into 5 levels of triage, combining the datasets will give more datapoints per level. However, one issue was that the two datasets only had 5 features in common: age, gender, blood pressure, heart rate, and glucose level. Proceeding with training gave undesirable results of F1-Scores less than 50%. For the combined dataset, a simple Neural Network of 2 layers was also used in order to improve the score. Although the Neural Network algorithm brought up the accuracy to 69%, the combined dataset still performed worse than only training on dataset 2.

```
# keras model
model = Sequential()         # a model consisting of successive layers
# input layer
model.add(Dense(n_neurons, input_dim=input_dim, activation='sigmoid'))
# output layer, with 5 neuron
model.add(Dense(5, activation='sigmoid'))
# compile the model
model.compile(loss='mean_squared_error', optimizer='sgd', metrics=['accuracy'])
```

```
results = model.evaluate(final_X_test, final_y_test)
print("test loss, test acc:", results)
```

✓ 0.4s

```
104/104 [==============================] - 0s 1ms/step - loss: 0.0869 - accuracy: 0.6949
test loss, test acc: [0.08686400204896927, 0.6948795318603516]
```

Figure 2.6 Neural Network model and result

Discussion

Training the model just on Dataset 2 showed great results with high F1-scores in general. Although there were a couple of false negatives, this algorithm can be used to get an initial grading for the patient's triage without solely relying on human classification alone. However, attempting to combine both Dataset 1 and 2 showed poorer results than on Dataset 2 alone. One reason this happened might be due to the fact that the two datasets only had 5 features in common, so the algorithm didn't have enough information to train on, even if there were more datapoints in total.

Dataset 3 : ECG image dataset

Electrocardiogram (ECG) readings are often taken by nurses during triage, especially in presence of chest pain symptoms. Providing vital information about cardiac rhythm, presence of arrhythmias, myocardial ischemia/infarction, and other abnormalities.

Dataset Details

The dataset consists of 99,190 and 24,773 ECG images in the train and test dataset respectively with 6 classifications (Figure 3.1). Each class of images had their own line colour (Figure 3.2). In the train set, 75,000 were Normal beat images, and in the test set, 18,000 were Normal beat images (Figure 3.3). Figure 3.4 shows the distribution images among the 6 classes.

N: Normal beat

S: Supraventricular premature beat

V: Premature ventricular contraction

F: Fusion of ventricular and normal beat

Q: Unclassifiable beat

M: myocardial infarction

Figure 3.1 : 6 Categories of ECG types in the dataset

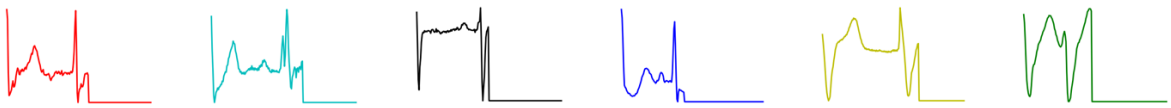


Figure 3.2 : Example of an ECG image in each of the 6 classes

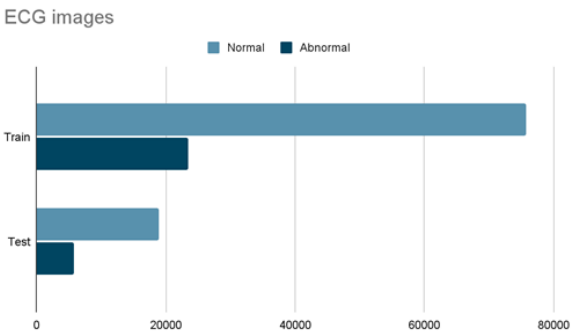


Figure 3.3 : Normal and Abnormal images count

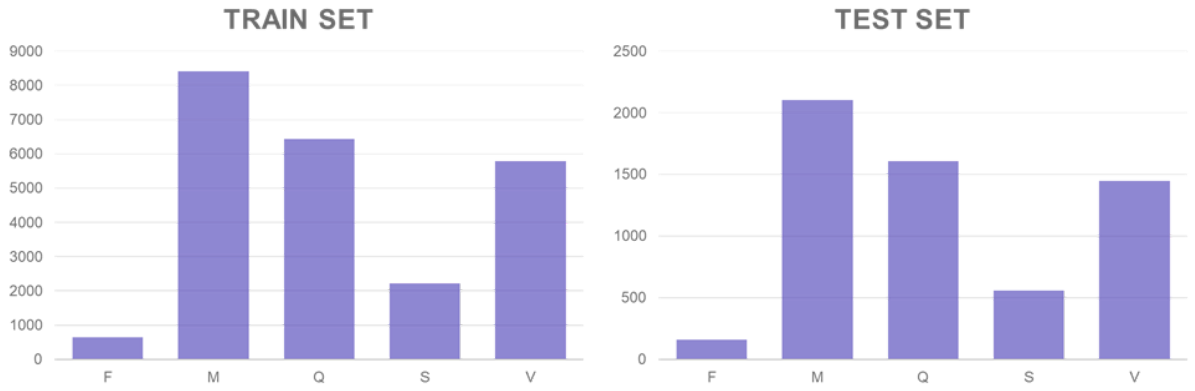


Figure 3.4 : Distribution of classes in train and test set

Data-preprocessing

The images were in 6 different fixed colours depending on their class which if left as it is, would cause the trained model to perform with 100% accuracy by just predicting based on colour. As such the images were transformed to grayscale to avoid this problem. The images were also scaled down to shape of 150,150 to improve training speed. Figure 3.5 shows the output of a batch of the transformed images.

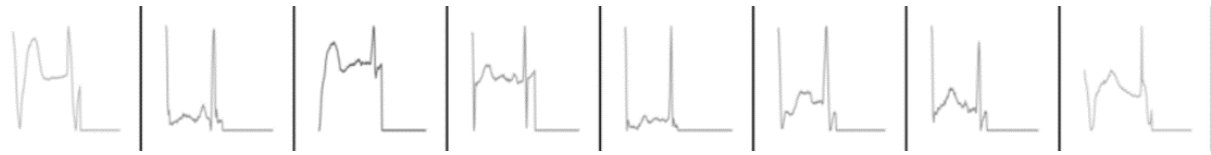


Figure 3.5 : Example of transformed images

Model

Since the model has to deal with images as inputs, a convolutional neural network is built using PyTorch. Using Conv2d layers and ReLU activation layers, followed by flattening and passing through linear layers was the main structure. Figure 3.6 shows the initial sequential block of layers for the initial model.

```
nn.Conv2d(1, 16, kernel_size = 3, padding = 1),
nn.ReLU(),
nn.Conv2d(16,32, kernel_size = 3, stride = 1, padding = 1),
nn.ReLU(),
nn.MaxPool2d(2,2),
```

```
nn.Flatten(),
nn.ReLU(),
nn.Linear(1024,6)
```

Figure 3.6 : Initial Model and its block of layers

However, training was computationally slow and GPU memory issues were encountered. As such BatchNorm2d layers were introduced to accelerate training by taking the outputs of each Conv2d layer and normalising them before passing them on as inputs into the next level. Also, Dropout layers were introduced after each linear layer to reduce the number of active nodes in the neural network to further increase training speed.

Stacking 5 of the sequential blocks of layers together we obtain our final model shown in Figure 3.7.

```
nn.Conv2d(1, 32, kernel_size = 3, padding = 1),
nn.BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True),
nn.ReLU(),
nn.Conv2d(32,64, kernel_size = 3, stride = 1, padding = 1),
nn.BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True),
nn.ReLU(),
nn.MaxPool2d(2,2),
```

```
nn.Conv2d(512, 512, kernel_size = 3, stride = 1, padding = 1),
nn.BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True),
nn.ReLU(),
nn.Conv2d(512,512, kernel_size = 3, stride = 1, padding = 1),
nn.BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True),
nn.ReLU(),
nn.MaxPool2d(2,2),


nn.Flatten(),
nn.Linear(8192,4096),
nn.ReLU(),
nn.Dropout(p=0.5, inplace=False),
nn.Linear(4096, 1024),
nn.ReLU(),
nn.Dropout(p=0.5, inplace=False),
nn.Linear(1024,6)
```

Figure 3.7 : First and Last block of layers of the model

To train the model, we used the Adam optimizer, Cross Entropy loss function and learning rate of 0.001. The model was trained with a Train-Validation Random split of 90% into train and 10% into validation, as well as in batch sizes of 8 images and for 10 epochs. In each epoch, the model was trained and evaluated with the train and validation sets. Figure 3.8 shows the validation accuracy, training and validation loss with each epoch. The validation accuracy after training was 99.78%.

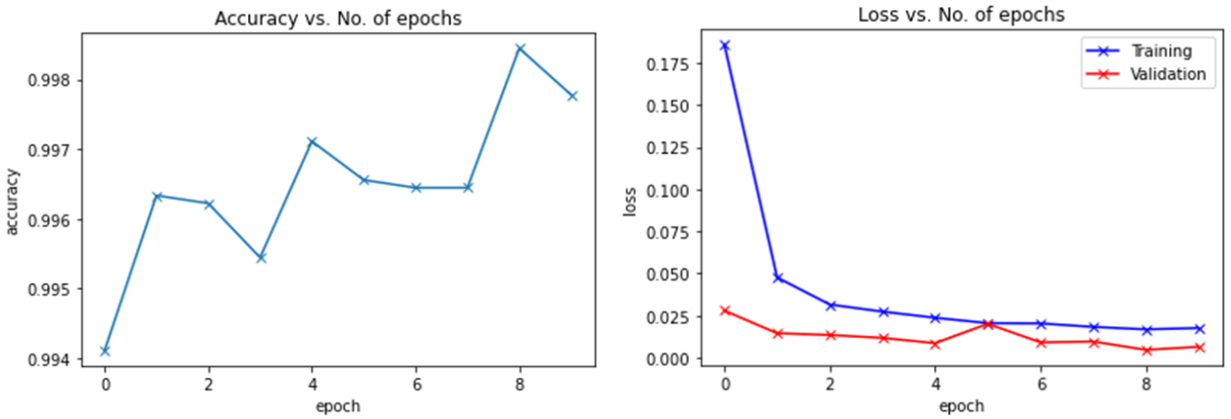


Figure 3.8 : Validation accuracy and Loss vs Epochs

Results and discussion

To evaluate the performance of the trained model, we tested the model with the 24,000 test image set. Focusing on the F1-scores, each class type were predicting well with low false negatives, with increasing order, “S” with 0.95, “V” with 0.98, “F” with 0.99, and the remaining “M”, “N” and “Q” with 1.00 F1-scores. Figure 3.9 shows the classification report of testing with the test set. Figure 3.10 shows the confusion matrix of predicted classes.

	precision	recall	f1-score	support
F	1.00	0.98	0.99	161
M	1.00	1.00	1.00	2101
N	1.00	1.00	1.00	18926
Q	1.00	1.00	1.00	1608
S	0.98	0.92	0.95	556
V	0.97	0.99	0.98	1447
accuracy			1.00	24799
macro avg	0.99	0.98	0.99	24799
weighted avg	1.00	1.00	1.00	24799

Figure 3.9 : Classification Report with F1-scores

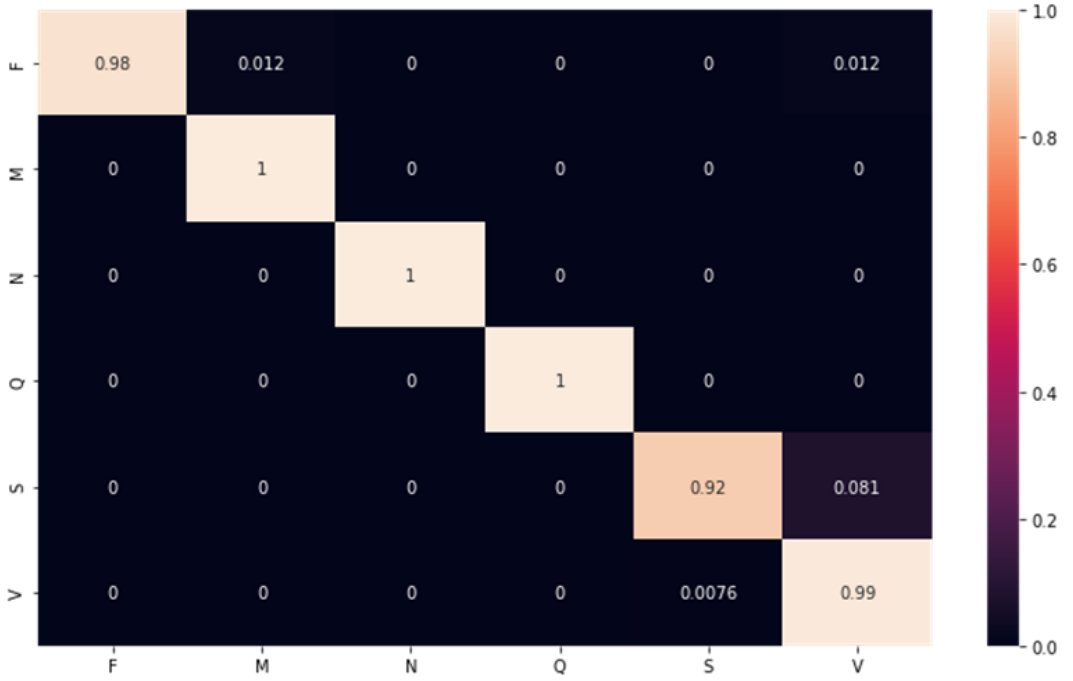


Figure 3.10 : Confusion Matrix of predicted classes

The model performed with a test accuracy of 99.758% with 60 images predicted incorrectly. Of which all were abnormal image types, observing the confusion matrix, none of these abnormal images were classified as normal, which shows that the model can distinguish binary output of normal versus abnormal images with no false negatives which is ideal and what we aim for to triage patients.

Also, looking at the incorrectly classified classes, we noticed only abnormal images of types “F”, “S” and “V” were misclassified (Figure 3.11). Which also corresponds to the 3 classes with lesser image samples in their train and test sets, possibly resulting in insufficient training data for the model to learn.

```
The incorrect classification counts are :
 F : 4 , M : 0 , N : 0 , Q : 0 , S : 45 , V : 11
```

Figure 3.11 : Incorrectly classified image types

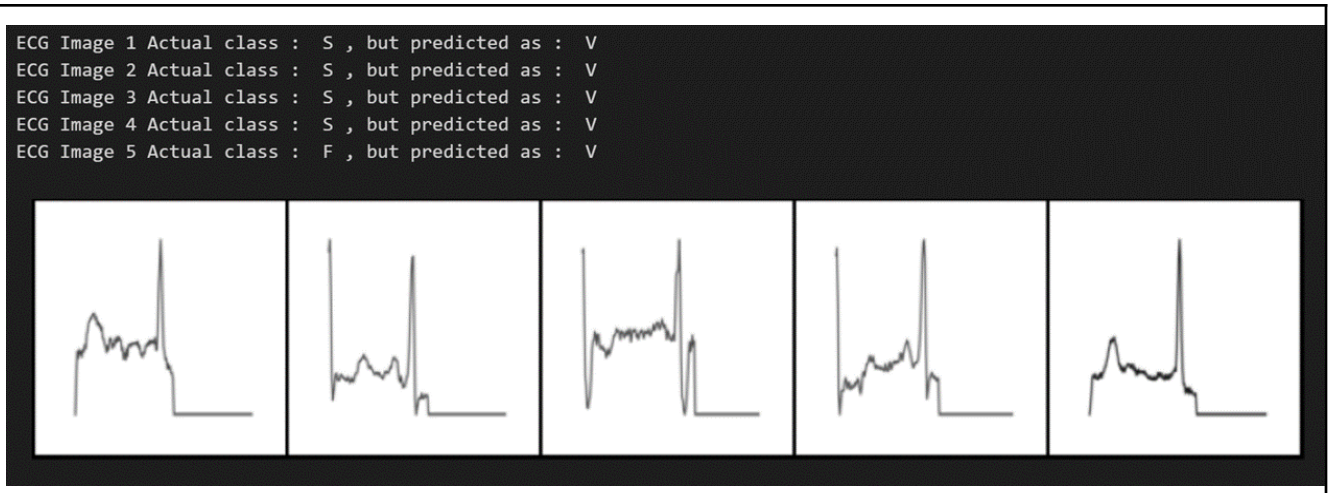


Figure 3.12 : Example of 5 misclassified images

Overall, our convolutional neural network, ECG_Classifier, was trained sufficiently well on this image dataset to accurately predict the 6 multiclass of image types. With high F1-score performance, making the model ideal for triaging patients.

Conclusion

In conclusion, the three datasets tested show greatly varying results. Dataset 1 as a standalone does not provide conclusive enough results for us to adopt as our primary solution. Dataset 2 was able to classify lower levels of yellow, green, and white very accurately, but higher levels of red and orange performed relatively less accurately. Lastly, for ECG images, the CNN model could perform sufficiently well and F1 score was high with no false negatives. If the dataset was split according to normal and abnormal cases, the model can correctly distinguish and output Normal vs Abnormal cases. Out of the three models, the CNN model for classifying ECG images performed the best and can possibly be used in conjunction with traditional triage classification procedures to determine the triage level of the patient quicker and more effectively.

Of course, there is room for improvement for our models. One improvement is to combine the datasets of records and images to get a better classification. Another improvement is to test our models on other datasets with similar features, if such datasets can be acquired.