DANA 4850 Project Report

Group Members	Student ID
Chun Ching Look	100347726
Pankaj Gaur	100349816
Sukhpreet Singh	100342947
Chandy George	100351259

The Weltel Project

Introduction

Weltel is an organisation that aims at aidin healthcare globally and facilitating the best medical care to people everywhere. From a humble start in Kenya working to cater HIV patients they have not started striving to be global.

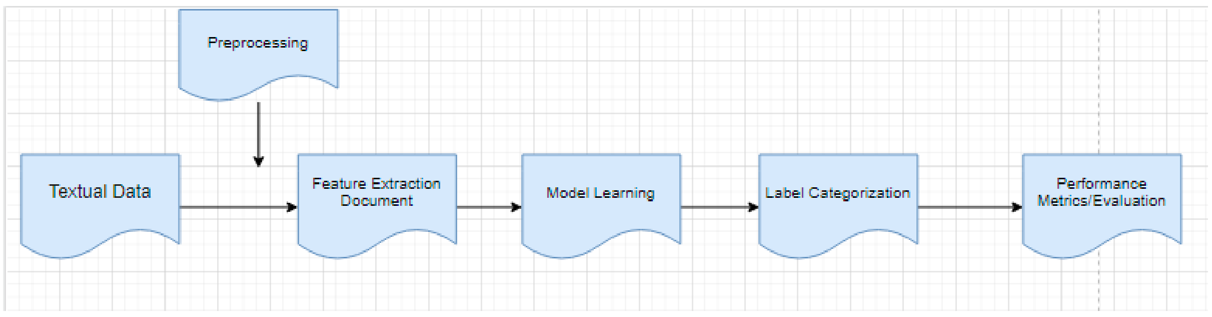
Weltel has created an online platform where patients can chat with operators and get linked with their respective practitioners(doctors). These chats are transferred to clinics/doctors and reports and created for the patient and doctor viewing.

The Project

A dataset with 3373 observations and 38 labels was given. Each observation is a conversation between a patient and an operator. Based on these texts from the conversation the patients are placed into respective diagnosis(labels).

Objectives

- Streamline texts from conversations and create keywords using NLP.
- Create a model to predict if the patients are being assigned to the right labels using ML techniques.
- Create a model that has both good accuracy and can be dynamically used.



NLP

First a list of stop words were downloaded to remove words from the text without losing its meaning and integrity.

Tools used - from nltk.corpus import stopwords.

The texts were then tokenized to form individual words and separate entities(maximum of 500 words), stemmed to reduce each of those words to its root forms and then lemetized to give more context to those words with dependence to words before and after it.

Tools used - nltk.tokenize.word_tokenize,
from nltk.stem.porter -- PorterStemmer,
from nltk.stem -- WordNetLemmatizer

A bag of words was created and the main feature words from the list were checked. And the number of times these main features were mentioned in the texts are also checked. The texts were then passed through a glove vectorizing algorithm to get proper machine readable vectors. This excludes all the abbreviations or meaningless words from the text and takes into consideration only the relevant words.

Tools used - from sklearn.feature_extraction.text -- CountVectorizer, TfidfVectorizer,
glove_file = ('https://www.kaggle.com/danielwillgeorge/glove6b100dtxt')

The data was finally divided into its 38 labels for the train and test datas, as this was a multi-label, multi-classification problem. The tokenized text was then dived to train and test sets to be fit into the model.

Machine Learning Models

We used different machine learning models like Random Forest, Naive Bayes, AdaBoost with different optimizer parameters but the accuracy was not upto the mark. For each label, the maximum accuracy obtained was 35%.

We, then, decided to use Recurrent Neural Networks. Inputs were created for the RNN as separate variables to be assigned into the model using LSTM.

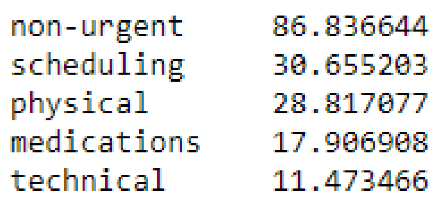
input - max length of 200
1 Embedding layer ,
LSTM layer to the embedding
And a dense layer with activation relu.

38 output layers were created to be assigned into the model for each of the labels, with a denselayer activated by sigmoid.

Optimizer -- Adam
Loss -- Binary cross-entropy
Metrics -- Accuracy

Overall model quality was checked and it gave good results. This was seen because of the unbalanced data. Overall model quality will be high because of the large number of 0s in the dataset and the model correctly predicting them. But we need to know the exact accuracy of them, model prediction the 1s. Performance matrix with the confusion matrix was taken for each of the labels.

Owing to the limited size of the dataset and not having enough true values for all the labels, we reduced the study labels to those that have more than 10 percent of true values. As the true values are more important and more frequent, those diagnosis appear in the texts.



```
non-urgent     86.836644
scheduling     30.655203
physical       28.817077
medications    17.906908
technical      11.473466
```

These were the resulting labels.

Now hyperparameter tuning was done to create a model with these labels.

The Talos package was used to do automated hypertuning to the model as required.

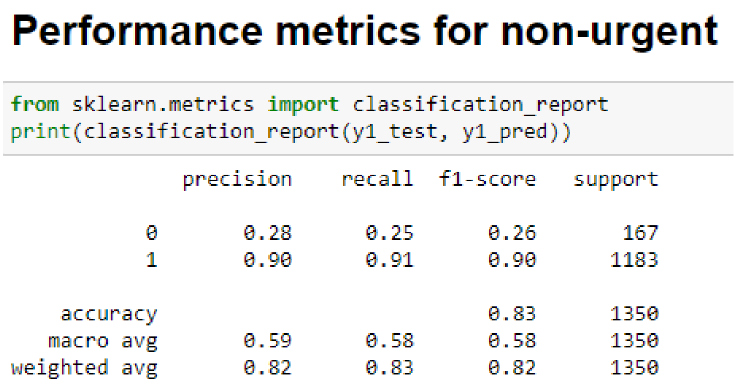
Input - max length 200
Embedding layer
LSTM layer
Dropout layer
Dense layer with activation relu
Another dropout layer

5 output layers were passed for each of the 5 labels with activation sigmoid

Optimizer -- Nadam, Adam
Loss -- binary cross entropy
Metrics -- Accuracy

The train and test data with the given parameters are scanned by talos and the best parameters for these 5 labels are given.

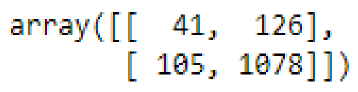
The model is then passed with these hypertuned parameters and the performance matrix is obtained for each of the 5 labels.



Performance metrics for non-urgent

```
from sklearn.metrics import classification_report
print(classification_report(y1_test, y1_pred))
```

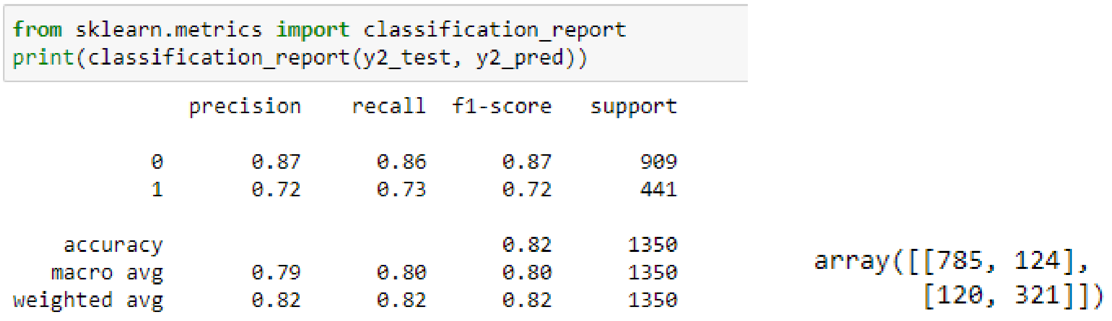
```
              precision    recall  f1-score   support

           0       0.28      0.25      0.26       167
           1       0.90      0.91      0.90      1183

    accuracy                           0.83      1350
   macro avg       0.59      0.58      0.58      1350
weighted avg       0.82      0.83      0.82      1350
```



```
array([[  41,  126],
       [ 105, 1078]])
```

Different hypertuning experiments were conducted and the model accuracy was recorded. Since our data is unbalanced, precision and recall correctly describes the output obtained from the model. From the above screenshots of classification report and confusion matrix, it can be observed that our model has a precision of .90.
This means that out of total positive predicted values (41 + 196), how many values are actual positive (41). Similarly, recall is 0.91 which means that out of total actual positive values (41 + 105), how many true values are predicted correctly (41). The number 126 in the confusion matrix depicts that there are 126 such texts which are classified as non-urgent by our model but actually do not belong to the 'non-urgent' group. Similarly, there are 105 such text messages which are actually 'non-urgent' but our model does not classify them as one.

Performance metrics for scheduling



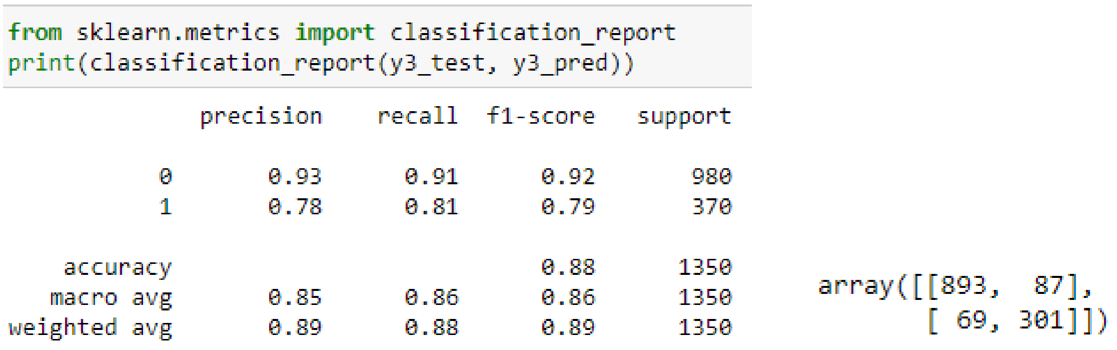
```
from sklearn.metrics import classification_report
print(classification_report(y2_test, y2_pred))
```

```
              precision    recall  f1-score   support

           0       0.87      0.86      0.87       909
           1       0.72      0.73      0.72       441

    accuracy                           0.82      1350
   macro avg       0.79      0.80      0.80      1350
weighted avg       0.82      0.82      0.82      1350
```

```
array([[785, 124],
       [120, 321]])
```

From the above classification report and confusion matrix, it can be observed that our model has a precision of .72 which means that out of total positive predicted values (785+124), how many values are actual positive (785). Similarly, recall is 0.73 which means that out of total actual positive values (785+120), how many true values are predicted correctly (785). The number 124 in the confusion matrix is the number of false positives and depicts that there are 124 such texts which are classified as ‘labelled’ by our model but actually do not belong to the ‘labelled’ group. Similarly, there are 120 such text messages which are actually as categorized as ‘labelled’ but our model does not classify them as one.

Performance metrics for Physical



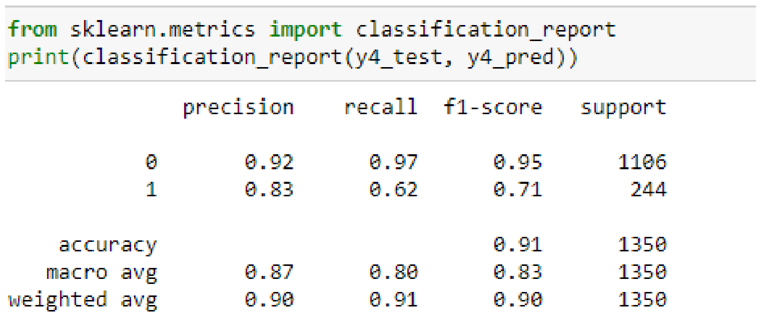
```
from sklearn.metrics import classification_report
print(classification_report(y3_test, y3_pred))
```

```
              precision    recall  f1-score   support

           0       0.93      0.91      0.92       980
           1       0.78      0.81      0.79       370

    accuracy                           0.88      1350
   macro avg       0.85      0.86      0.86      1350
weighted avg       0.89      0.88      0.89      1350
```

```
array([[893,  87],
       [ 69, 301]])
```

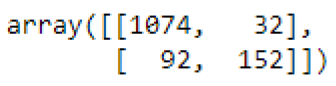
From the above classification report and confusion matrix, our model has a precision of .78 which means that out of total positive predicted values (893+87), 893 values are actual positive. Similarly, recall is 0.81 which means that out of total actual positive values (893+87), 69 are true values that are predicted correctly. The number 87 in the confusion matrix is the number of false positives and depicts that there are 87 such texts which are classified as ‘Physical’ by our model but actually do not belong to the ‘Physical’ group. Similarly, there are 69 such text messages which are actually as categorized as ‘Physical’ but our model does not classify them as one.

Performance metrics for Medications



```
from sklearn.metrics import classification_report
print(classification_report(y4_test, y4_pred))
```

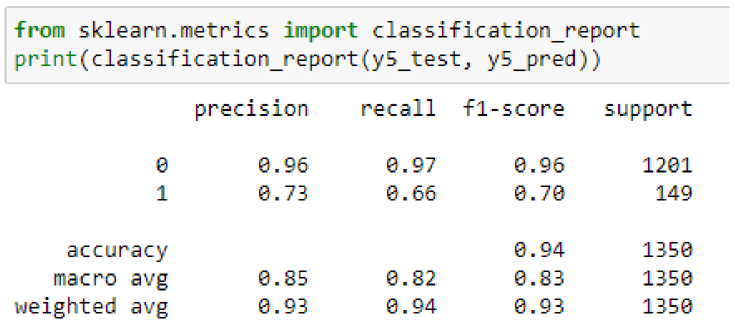
```
              precision    recall  f1-score   support

           0       0.92      0.97      0.95      1106
           1       0.83      0.62      0.71       244

    accuracy                           0.91      1350
   macro avg       0.87      0.80      0.83      1350
weighted avg       0.90      0.91      0.90      1350
```



```
array([[1074,   32],
       [  92,  152]])
```

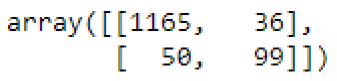
From the above classification report and confusion matrix, it can be observed that our model has a precision of .83 which means that out of total positive predicted values (1074+32), 1074 values are actual positive. Similarly, recall is 0.62 which means that out of total actual positive values (1074+32), the number of true values predicted correctly are 1074. The number 32 in the confusion matrix is the number of false positives and depicts that there are 32 such texts which are classified as ‘Medications’ by our model but actually do not belong to the ‘Medications’ group. Similarly, there are 92 such text messages which are actually as categorized as ‘Medications’ but our model does not classify them as one.

Performance metrics for Technical



```
from sklearn.metrics import classification_report
print(classification_report(y5_test, y5_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.97      0.96      1201
           1       0.73      0.66      0.70       149

    accuracy                           0.94      1350
   macro avg       0.85      0.82      0.83      1350
weighted avg       0.93      0.94      0.93      1350
```



```
array([[1165,   36],
       [  50,   99]])
```

From the above classification report and confusion matrix, it can be observed that our model has a precision of .73 which means that out of total positive predicted values (1165+36), 1165 values are actual positive. Similarly, recall is 0.66 which means that out of total actual positive values (1165+36), 1165 true values are predicted correctly. The number 36 in the confusion matrix is the number of false positives and depicts that there are 36 such texts which are classified as ‘Technical’ by our model but actually do not belong to the ‘Technical’ group. Similarly, there are 50 such text messages which are actually as categorized as ‘Technical’ but our model does not classify them as one.

Conclusion and Recommendations

Since the data is unbalanced, accuracy does not make much sense in our classification problem. So, we took precision and recall as our baseline performance metrics for our model. We could observe that because of the limited size of data, it is not reliable to say the results of this model are absolutely precise to classify text conversations. In our case we have a problem of unbalanced data, the model becomes more biased towards the majority class (which is 0's in our case) as it has a larger influence on the final loss value and our model becomes less useful. To deal with this scenario, we added weights to the losses corresponding to different classes to even out this data bias and tried different optimization methods as well.

There were few labels which were having very less sample size. For instance, the labels like Alternative, Counseling, Diet, Exercise, FF, Cultural. Service_Complaint etc. are having actual true samples less than 10% and there is no information about the relevance of these labels. Let's take an example of FF as a label in our dataset. We do not know what this means and there is no historical information about usage of these labels. Hence, we dropped such labels with low sample count and selected only 5 labels. We could be missing some important labels in this process, which might be important from Weltel's perspective.