Report on
Sentiment Dataset

1.Data Analysis and Visualization

There were only two columns of Label and Phrase in the dataset. The Feature was a phrase and the target was Label.

Number of phrases according to Label type tabulated below:

Label Type	positive	Negative
Number of Phrases	1800	1000

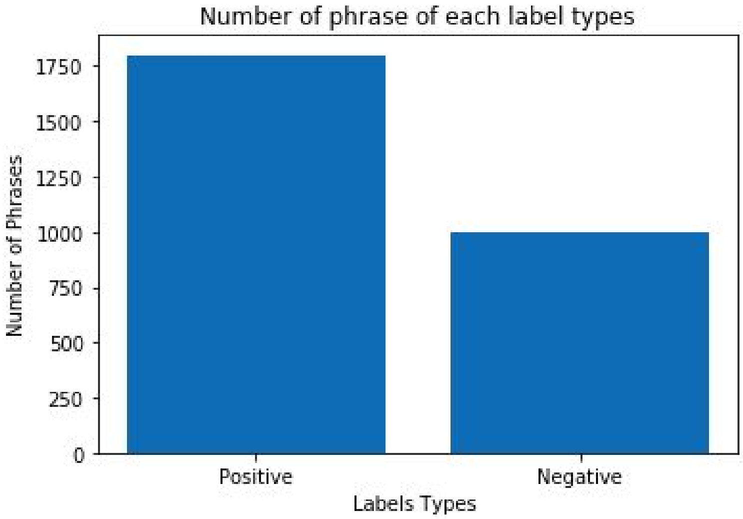


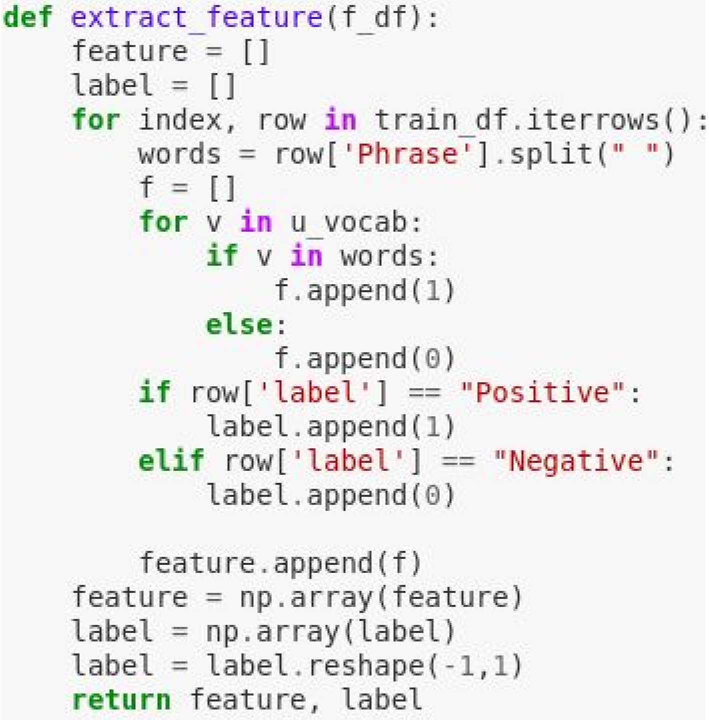
Fig:Number of phrase of positive and negative label type

From above, the table and histogram showed that a positive phrase was higher than a negative phrase in a given dataset.

The total number of data in the dataset was 2800 and 1680 data was used for training,560 data used for testing, and 560 data used for validation.

2. Feature Extraction and Normalization

The train dataset was split into x_train and y_train and the test dataset was split into x_test and y_test and validation set was split into x_val and y_val.



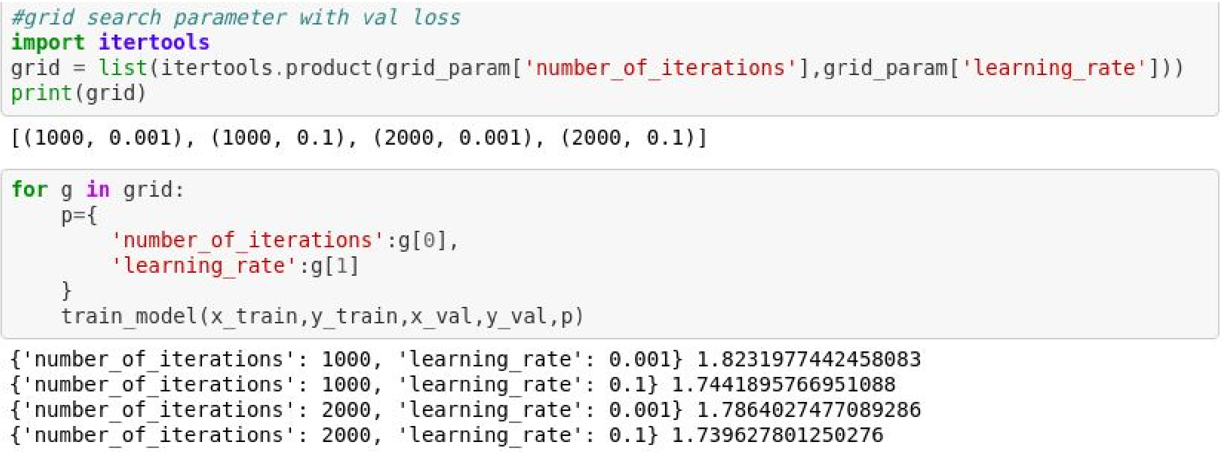
```python
def extract_feature(f_df):
    feature = []
    label = []
    for index, row in train_df.iterrows():
        words = row['Phrase'].split(" ")
        f = []
        for v in u_vocab:
            if v in words:
                f.append(1)
            else:
                f.append(0)
        if row['label'] == "Positive":
            label.append(1)
        elif row['label'] == "Negative":
            label.append(0)

        feature.append(f)
    feature = np.array(feature)
    label = np.array(label)
    label = label.reshape(-1,1)
    return feature, label
```

The above figure showed how the feature was extracted.

3.Grid search parameter with val loss

Binary cross Entropy loss was used as a model and **number_of_iterations** and **Learning_rate** used for grid search parameters.



```python
#grid search parameter with val loss
import itertools
grid = list(itertools.product(grid_param['number_of_iterations'],grid_param['learning_rate']))
print(grid)
```

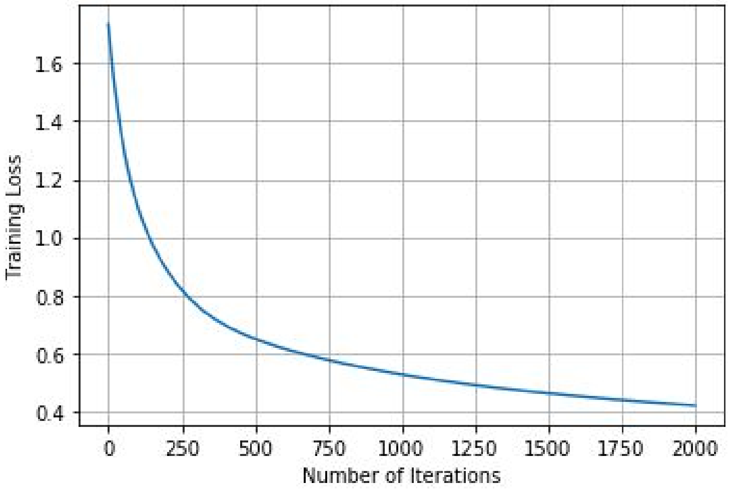
```
[(1000, 0.001), (1000, 0.1), (2000, 0.001), (2000, 0.1)]
```

```python
for g in grid:
    p={
        'number_of_iterations':g[0],
        'learning_rate':g[1]
    }
    train_model(x_train,y_train,x_val,y_val,p)
```

```
{'number_of_iterations': 1000, 'learning_rate': 0.001} 1.8231977442458083
{'number_of_iterations': 1000, 'learning_rate': 0.1} 1.7441895766951088
{'number_of_iterations': 2000, 'learning_rate': 0.001} 1.7864027477089286
{'number_of_iterations': 2000, 'learning_rate': 0.1} 1.739627801250276
```

4.Train loss plot in best model

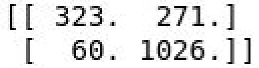
Best Hyperparameter was found in a number _of_ iteration **2000** and learning _rate **0.1** with training loss **0.4194**.



5.Model Evaluation

Model evaluated using confusion matrix ,accuracy,recall,precision,F1-score and ROC curve.

Found following Confusion matrix

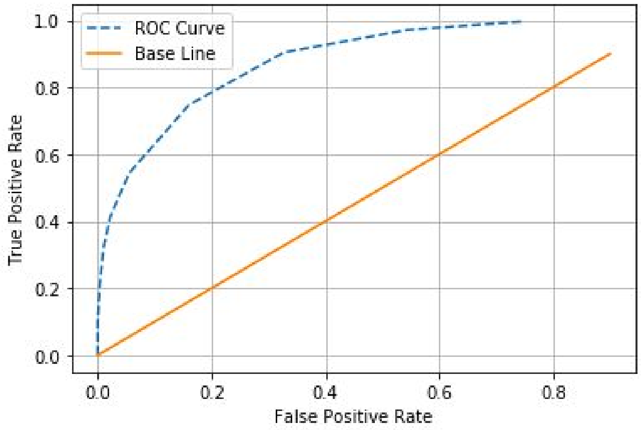


```
[[ 323.  271.]
 [  60. 1026.]]
```

Following accuracy,recall,precision,F1-score was found by evaluating model

	Value
Accuracy	0.802
Recall	0.543
Precision	0.843
F1-score	0.661

ROC Curve



6.Code link

https://github.com/rachanakafle/Linear-and-Logistic-Regression/blob/master/logistic%20regression.ipynb