**Written Assignment (60 Points)**

Written assignment tests the understanding of the student for the assignment's task. We have split the writing into sections. You will need to write 1-2 paragraphs describing the sections. Please be concise.

**In your own words, describe what the task is (20 points)**

Describe the task, how is it useful and an example.

Section 1: Sentiment Classification Dataset:

We have two datasets of a collection of hotel reviews (positive and negative), both datasets are one review per line. The reviews are not tokenized, or sentence segmented, and the words are space separated. We need to explore the distribution of the reviews (positive and negative), create training and test datasets (split the original dataset), and define functions for evaluation metrics (compute accuracy, precision, recall, and f1 score)

Section 2: Baselines:

We need to create two baselines for Random Chance and Naïve Bayes Classifier. The Random Chance predict the label according to the labels’ distribution. The Naïve Bayes Classifier using tokens in the training samples. We need preprocess the set and implement fit and predict methods. We will compare these two baselines by f1 score on same sample set.

Section 3: Logistic Regression on Features:

We need to build a logistic regression based on hand-engineered features. We need to implement feature extraction to featurize text for a classification system for sentiment analysis. We need to have a normalization formula to normalize dataset for control the scales before passing it to the classifier. We need to implement weights and logistic loss function and convert logits into prediction functions. We need to use different epochs and learning rate to improve the result.

**Describe your method for the task (10 points)**

Important details about the implementation. Feature engineering, parameter choice etc.

Section 1: Sentiment Classification Dataset:

We use util.py to load the dataset. We use a for-loop to iterate items in labels: whenever we get label == 0, we increase negative count by one, same logic for positive reviews. I use pie chart for visualization. For split the dataset, I transform the list object into DataFrame first, then shuffle indices using numpy.random.shuffle to split the dataset randomly. I use DataFrame.values.tolist() to transform DataFrame back to list object.

For evaluation metrics, we use a for-loop to compare values between predicted\_labels and true\_labels, whenever we get same values equal to 1, we increase TP (true positive) by 1. Accuracy (correct prediction / all prediction), Precision (True Positive / All Positive Prediction), Recall (True Positive / All Positive Labels), and F1 Score (2 \* Precision \* Recall / (Recall + Precision))

Section 2: Baselines:

I use NumPy library to randomly generate value set based on training distributions. I use spacy to tokenize the text and help extract features from text. I preprocess text using lemma\_ function and remove stopwords using is\_stop function.

The fit method for Naïve Bayes Classifier I need to group samples by their labels and preprocess each text. I count the words of the text for each label through a for-loop and dictionary, whenever we get a word, we update the dictionary. The predict method for Naïve Bayes Classifier I need to compute each word probability by calculate the count of that word divided by whole word count. I use LaPlace smoothing, whenever I get a new word not in the training dictionary, I add 1 to it. I use math.log to sum all probability, return result by compared positive probability and negative probability.

Section 3: Logistic Regression on Features:

I implement six features exactly showed in the table: count (positive/negative lexicon), count (1st and 2nd pronouns), add 1 if get ‘no’ or ‘!’, and log(word count of doc). Normalize torch.Tensor by iterating a 2D array and update values based on the formula. I initialize the weight using built-in library in torch, which is torch.nn.init.xavier\_uniform\_. I use built-in library in torch to implement logistic loss function, which is torch.nn.BCEWithLogitsLoss. I convert logits into prediction by using built-in library torch, which is torch.round.

**Experiment Results (10 points)**

Typically a table summarizing all the different experiment results for various parameter choices

Section 1: Sentiment Classification Dataset:

All dataset

|  |  |
| --- | --- |
| # of examples have label = 0 | # of examples have label = 1 |
| 94 | 95 |

Train Label Distribution

|  |  |
| --- | --- |
| # of examples have label = 0 | # of examples have label = 1 |
| 74 | 77 |

Dev Label Distribution

|  |  |
| --- | --- |
| # of examples have label = 0 | # of examples have label = 1 |
| 20 | 18 |

Section 2: Baselines:

Dev Set Result

|  |  |
| --- | --- |
| Random Chance F1 | Naïve Bayes F1 |
| 0.5945945945945946 | 0.9090909090909091 |

Test Set Result (Deadline)

|  |  |
| --- | --- |
| Random Chance F1 | Naïve Bayes F1 |
| 0.491228 | 0.90196 |

Section 3: Logistic Regression on Features:

With Epoch = 500 and learning rate = 0.5

Dev Set Result at Epoch 500

|  |  |
| --- | --- |
| Loss | F1 |
| 0.60735 | 0.9412 |

Dev Set Result/Logistic Regression Results

|  |  |
| --- | --- |
| Accuracy | F1-score tensor |
| 0.89473 | 0.8947 |

Test Set Result/Logistic Regression Results (Deadline)

|  |  |
| --- | --- |
| Accuracy | F1-score tensor |
| 0.84 | 0.8621 |

**Discussion (20 points)**

Key takeaway from the assignment. Why is the method good? shortcomings? how would you improve? Additional thoughts?

Section 1: Sentiment Classification Dataset:

Use pie chart to visualize the distribution of positive and negative reviews. Although there are many other kinds of charts to explore the distribution, pie chart is the most intuitive ways for distribution. Use pandas library to split the dataset into training and test set by random shuffle the indices. However, there are many other ways to split the dataset but I’m more experience with pandas rather to transform original list object into DataFrame object than try other methods. Implement the evaluation metrics based on formula helps me to further understand accuracy, precision, recall and f1 score. I believe there are some libraries can automatically compute these values.

Section 2: Baselines:

Use numpy.random.choice to generate 0 and 1’s based on distribution. Understand how to use spacy to remove stop words and tokenize sentences into words. Dictionary is quite useful when I implement fit method in Naïve Bayes Classifier. I update dictionary for counting whenever I iterate a word in the sentence. The predict method I need to create another two dictionaries for probability of each word from previous counting dictionary. The Laplace smoothing method helps us to calculate probability when the word is not in our dictionary. I use log function because sum all log is much faster than multiple all the probabilities.

Section 3: Logistic Regression on Features:

Use three built-in library in torch.tensor to initialize weights, logistic loss function, and covert logits into predictions. Try different combination of epoch and learning rate to get relatively good performance. In this assignment I only use one method to initialize weights and there are many more other built-in function to initialize weights that maybe can improve the result. On the other hand, I can implement more engineering features to improve feature extraction. For example, the count of third pronouns in the whole text, the count of comma in the whole text, and etc. The combination of epochs and learning rate can be tuned using exist library for parameter tuning.