Documentation of Assignment

The assignment aims at detecting actionable items from textual conversations particularly emails and meetings. The assignment is divided into two tasks. First one, using linguistic rules on an email dataset to detect actionable items. Second, using machine learning models to classify sentences as an action item or not.

## Task 1: NLP and Linguistic rules to identify action items

The dataset provided has about 500k emails. Emails are tokenized into sentences and few rules are framed to classify each sentence as an action item or not. Each sentence is tagged with POS information, chunks, dependency parsing and entities, Rules are framed from the tagged information to identify action items. Some rules are Precision oriented while some are Recall.

Additional information in the form of action slang such as “ASAP”, and presence of Interrogative words such as “Who, When, How” at the start of the sentence are also considered.

Rule 1 : Presence of “?” at the end of the sentence

Rule 2: Presence of Modal-Phrase in the sentence especially towards the beginning.

Rule 3: Presence of few patterns like “INTJ” followed by Verb followed by noun, Pronoun followed by verb and noun.

Rule 4: Presence of named entities such as date and time with root POS tag as Verb or Noun

**Uploaded sample output into github.**

Each sentence will have a probability at the end 1.0 action item, 0.0 for non action item.

**Tools:** NLTK, Spacy, Pandas

## Task 2: ML to identify action items

The dataset contains only action items of about 1200 in number. The problem is modeled as a classification problem. As the dataset does not contain negative samples, some random sentences about 1500, are selected from news articles. The goal of the classifier is to train the entire dataset and predict if the sentence is an action item or not. The complete dataset with negative samples has been uploaded in github directory.

Step 1: Transform each sentence into a feature vector using CountVectorizer.

Step 2: Use categorical features derived from linguistic rules above in addition to the above feature vector. These categorical features are presence of modal phrases, count of noun phrases, verb phrases and other phrases.

Step 3: Train using an ML algorithm such as NaiveBayes or Logistic Regression

Step 4: Do cross validation to report summary or results on the test data.

**Tools:** NLTK, Spacy, Pandas, Scikit-learn

**Model selection and Evaluation**

Experimented with multiple algorithms available in scikit-learn and amongst all, NaiveBayes gave the best baseline performance. Dataset is very small and intuitively also NaiveBayes, Logistic regression or Random Forest algorithms should perform better compared to other algos such as SVM or Neural Networks which require good amount of data to train.

I used accuracy and F-measure as the evaluation metrics as the problem is modeled as Classification. Used five fold cross validation for reporting results.

|  |  |  |
| --- | --- | --- |
| Method | Accuracy | F-Measure |
| BaseLine(Count Vec + MultiNB) | 0.92 | 0.93 |
| BaseLine + bigrams | 0.94 | 0.95 |
| BaseLine + bigrams + max\_feaures 1000 | 0.92 | 0.93 |
| BaseLine + bigrams + Linguistic rules | 0.94 | 0.95 |

Unigrams and Bigrams combined gave the best model and experimenting with feature selection or adding additional linguistic features did not help in increasing the performance of the model. May be with more data we can expect all these features start playing a role.

**Deep Learning**

Used Keras high level library to classify action items using neural networks. As the dataset is small, there is every chance of overfitting but the idea is how to use all modules available in Keras and different layers of neural networks like embedding, dropout, early stopping, dense layers for classification tasks. The validation accuracy achieved on the dataset is 91% which is less than that of using simple NaiveBayes.

The entire code has been uploaded in the github url provided. There are three main files for execution

python3.6 semantics/action\_item\_classifier\_NLP.py <path\_to\_enron\_email>

python3.6 ml/actionitem\_detection.py <path\_to\_labeled\_dataset>

python3.6 ml/actionitem\_detection\_deeplearning.py <path\_to\_labeled\_dataset>