Task 1 :

Intent detection on Enron email set. We define "intent" here to correspond primarily to the categories "request" and "propose".

In some cases, we also apply the positive label to some sentences from the "commit" category if they contain datetime, which makes them useful. Detecting the presence of intent in email is useful in many applications, e.g., machine mediation between human and email. The dataset contains parsed sentences from the email along with their intent (either 'yes' or 'no').

You need to build a learning model which detects whether a given sentence has intent or not. Its a 2-class classification problem.

Find the train and test dataset in **enron.zip.**
Please do not hard code any values. Be prepared to explain your choice of classifier.

Although its not required you can refer this paper for more information on the dataset : Cohen, William W., Vitor R. Carvalho, and Tom M. Mitchell. "Learning to Classify Email into``Speech Acts''." EMNLP. 2004.

Tip: Try to add feature engineering into your model. Simple baseline with logistic regression gives 71% accuracy.

SOLUTION

1. Importing the data
2. Finding the features that can be used and extracted from the data.
    a. Count characters, unique words, average sentence length, count mentions
    b. remove punctuations
    c. count capitals and average of stop words and words
3. Vectorize the features using **TFIDF** and create the train dataframe.
4. Train with **Random Forest Classifier** with target names "Yes" or "No"
5. Test the results on test data – obtained accuracy of 87.11 %

```
Accuracy =>  87.11

Random Forest Classifier results:

              precision    recall  f1-score   support

         Yes       0.86      0.92      0.89       474
          No       0.89      0.81      0.85       387

    accuracy                           0.87       861
   macro avg       0.87      0.87      0.87       861
weighted avg       0.87      0.87      0.87       861
```

Figure1: Classification report

Observe the table below from few rows of train data with features.

train_val - DataFrame

Index	label	email	char_count	word_count	sent_count	apital_char_coun	al_word_c	ed_word_c	pword_co	ue_word_c	ention_co	_wordlen	g_sentlen	que_vs_wo	words_vs_v
0	No	unfortunately we have a lot of late nights waiting on systems like we are right now	86	17	1	1	0	0	8	16	0	5.05882	17	0.941176	0.470588
1	Yes	make sense for me to attend the meeting	40	8	1	1	0	0	4	8	0	5	8	1	0.5
2	No	some content has specific hardware or softw…	113	14	1	1	0	0	3	14	0	8.07143	14	1	0.214286
3	Yes	do we have a list of attendants agenda	39	7	1	1	0	0	4	7	0	5.57143	7	1	0.571429
4	No	before i go into the depths of my answer i …	93	19	1	2	1	0	9	18	0	4.89474	19	0.947368	0.473684
5	No	the insurance policies were not available a…	145	23	1	1	0	0	10	22	0	6.30435	23	0.956522	0.434783
6	No	conference calling will be available	37	5	1	1	0	0	2	5	0	7.4	5	1	0.4
7	Yes	please give me a call and we will discuss t…	169	31	1	1	0	0	17	25	0	5.45161	31	0.806452	0.548387
8	No	the sale of the assets came about in the at…	107	21	1	1	0	0	10	18	0	5.09524	21	0.857143	0.47619
9	Yes	if your appjoy crashes right after you open…	120	21	1	44	8	0	5	21	0	5.71429	21	1	0.238095
10	Yes	if you have multiple children in multiple c…	93	17	1	2	0	0	8	16	0	5.47059	17	0.941176	0.470588
11	No	i don t have chris email address i ve left him a message	60	11	1	3	1	0	4	11	0	5.45455	11	1	0.363636
12	No	indiana donnell harvey fr florida harvey …	204	36	1	6	0	0	12	34	0	5.66667	36	0.944444	0.333333
13	No	when you interview for a job you will answe…	131	23	1	1	0	0	11	19	0	5.69565	23	0.826087	0.478261
14	Yes	hey send them again	21	4	1	1	0	0	2	4	0	5.25	4	1	0.5
15	Yes	if you re not very familiar with these the…	132	27	1	3	0	0	12	27	0	4.88889	27	1	0.444444
16	No	we leave the choice to you because we are available either night	68	12	1	1	0	0	6	12	0	5.66667	12	1	0.5
17	Yes	chris when did bp amoco get involved	38	7	1	4	1	0	2	7	0	5.42857	7	1	0.285714
18	No	i can t make it during lunch	29	6	1	1	1	0	2	6	0	4.83333	6	1	0.333333

Figure2: Train Dataframe with features

Task2:

This is an information extraction exercise from financial textual data.
Your task is to write a scalable Information Extraction algorithm to extract datapoints from the text and compute the results. The program will be evaluated on surprise test data.

From the text we need to extract the following:

- Currency

- Amount

- Rounding (up / down/ nearest) If not mentioned, take it as 'nearest' by default.

for 'Delivery Amount' and 'Return Amount'.

We have included the textual data in **isda_data.json**. Refer **sample_isda.py** for the basic implementation framework.

Input: "Rounding. The Delivery Amount and the Return Amount will be rounded to the nearest integral multiple o f EUR 100,000; provided that if an amount corresponds to the exact half of such multiple, then it will be rounded up; and provided further that, for the purpose o f the calculation of the Return Amount where a party's Credit Support Amount is, or is deemed to be, zero, the Return Amount shall not be rounded."

Output:


```
{

"delivery_currency": "EUR",
"delivery_amount": "100,000",
"delivery_rounding": "nearest",
"return_currency": "EUR",
"return_amount": "100,000",
"return_rounding": "nearest"
}
```


SOLUTION:

1. Load the textual data – isda_data.json
2. Create the train data
    a. Collect all the currency entities
    b. Collect all the rounding entities
3. Annotate all the entities in the data at their character positions, extract them.
4. Add to the train data
5. Train Named Entity Recognition
6. Test the trained model.
   E.g : Text = “I have to deliver the amount of the nearest to EUR 10,000 ”

   Extracted intenties:

```
Entities [('EUR', 'currency'), ('10,000', 'amount')]
Entities [('EUR', 'currency'), ('10,000', 'amount')]
```