1. NLP using python

February 26, 2017

pwd

```
In [2]: messages = [line.rstrip() for line in open('SMS')]

In [3]: print len(messages)

5574


In [5]: for num, message in enumerate(messages[:10]):
            print num, message
            print '/n'


0 ham        Go until jurong point, crazy.. Available only in bugis n great world l
/n
1 ham        Ok lar... Joking wif u oni...
/n
2 spam        Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. T
/n
3 ham        U dun say so early hor... U c already then say...
/n
4 ham        Nah I don't think he goes to usf, he lives around here though
/n
5 spam        FreeMsg Hey there darling it's been 3 week's now and no word back! I'
/n
6 ham        Even my brother is not like to speak with me. They treat me like aids
/n
7 ham        As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)'
/n
8 spam        WINNER!! As a valued network customer you have been selected to recei
/n
9 spam        Had your mobile 11 months or more? U R entitled to Update to the late
/n


In [6]: import pandas

In [13]: messages = pandas.read_csv('SMS', sep = '\t',names=['labels','message'])
```

```
In [14]: messages.head()
```

Out[14]:

	labels	message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

```
In [15]: messages.describe()
```

Out[15]:

	labels	message
count	5572	5572
unique	2	5169
top	ham	Sorry, I'll call later
freq	4825	30

```
In [16]: messages.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
labels     5572 non-null object
message    5572 non-null object
dtypes: object(2)
memory usage: 87.1+ KB
```

```
In [18]: messages.groupby('labels').describe()
```

Out[18]:

		message
labels		
ham	count	4825
	unique	4516
	top	Sorry, I'll call later
	freq	30
spam	count	747
	unique	653
	top	Please call our customer service representativ...
	freq	4

```
In [20]: messages['length'] = messages['message'].apply(len)
         messages.head()
```

Out[20]:

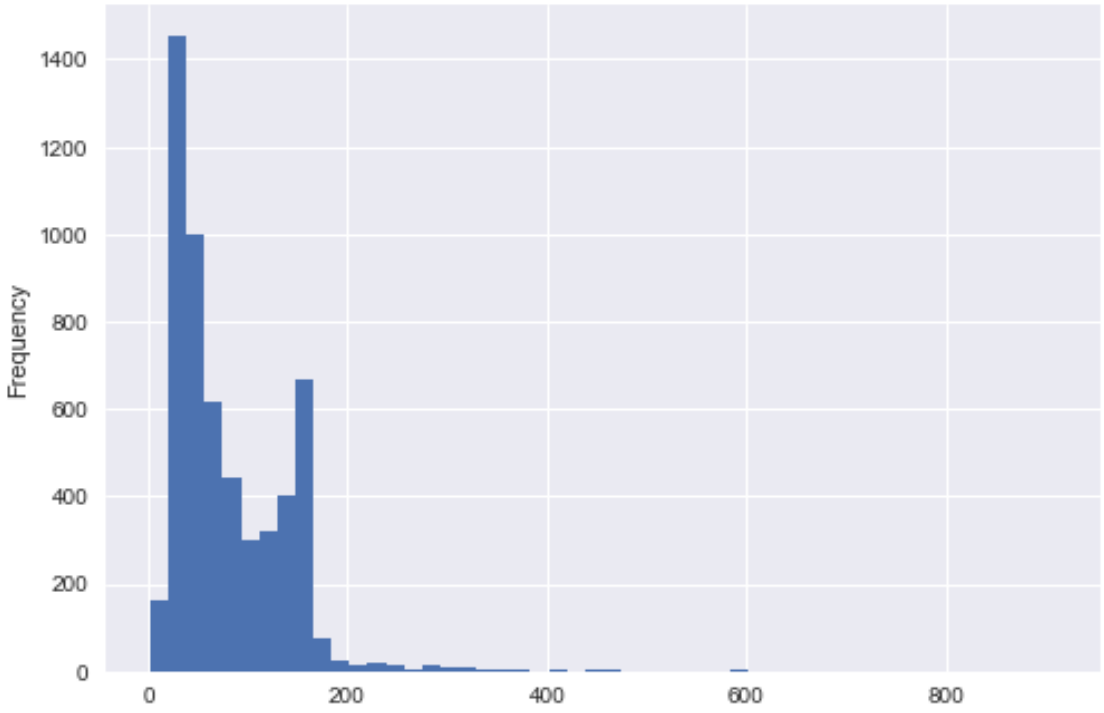
	labels	message	length
0	ham	Go until jurong point, crazy.. Available only ...	111
1	ham	Ok lar... Joking wif u oni...	29
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	155
3	ham	U dun say so early hor... U c already then say...	49
4	ham	Nah I don't think he goes to usf, he lives aro...	61

In [21]:
```
%matplotlib inline

import matplotlib.pyplot as plt
import seaborn as sns
```

In [27]:
```
messages['length'].plot(bins = 50,kind = 'hist')
```

Out[27]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x118973950>
```



In [28]:
```
messages['length'].describe()
```

Out[28]:
```
count    5572.000000
mean       80.616296
std        60.015593
min         2.000000
25%        36.000000
50%        62.000000
75%       122.000000
max       910.000000
Name: length, dtype: float64
```

In [30]:
```
messages[messages['length'] == 910]#['message'].iloc[0]
```

Out[30]:

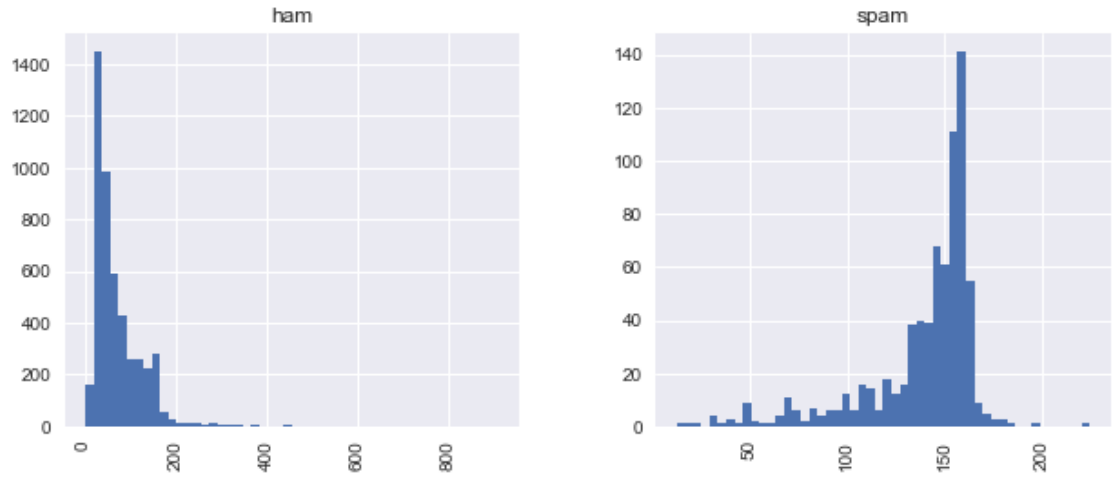
	labels	message	length
1085	ham	For me the love should start with attraction.i...	910

```
In [33]: messages[messages['length'] == 910]['message'].iloc[0]

Out[33]: "For me the love should start with attraction.i should feel that I need he

In [36]: messages.hist(column = 'length',by= 'labels',bins =50,figsize=(10,4))

Out[36]: array([<matplotlib.axes._subplots.AxesSubplot object at 0x119603b50>,
                <matplotlib.axes._subplots.AxesSubplot object at 0x1191d2490>], dty
```




```
In [37]: import string

In [38]: mess = 'Sample message! Notice: it has punctuation'

In [41]: nopunc = [char for char in mess if char not in string.punctuation]

In [43]: nopunc

Out[43]: ['S',
          'a',
          'm',
          'p',
          'l',
          'e',
          ' ',
          'm',
          'e',
          's',
          's',
          'a',
          'g',
          'e',
          ' ',
```

```
 'N',
 'o',
 't',
 'i',
 'c',
 'e',
 ' ',
 'i',
 't',
 ' ',
 'h',
 'a',
 's',
 ' ',
 'p',
 'u',
 'n',
 'c',
 'u',
 't',
 'a',
 't',
 'i',
 'o',
 'n']
```

```
In [44]: nopunc = ''.join(nopunc)
```

```
In [45]: nopunc
```

```
Out[45]: 'Sample message Notice it has puncutation'
```

```
In [46]: from nltk.corpus import stopwords
```

```
In [52]: stopwords.words('english')[:10]
```

```
Out[52]: [u'i',
          u'me',
          u'my',
          u'myself',
          u'we',
          u'our',
          u'ours',
          u'ourselves',
          u'you',
          u'your']
```

```
In [53]: len(stopwords.words('english'))
```

```
Out[53]: 153
```

```
In [54]: nopunc.split()

Out[54]: ['Sample', 'message', 'Notice', 'it', 'has', 'puncutation']

In [56]: clean_mess = [word for word in nopunc.split() if word.lower() not in stopw

In [57]: clean_mess

Out[57]: ['Sample', 'message', 'Notice', 'puncutation']

In [58]: def text_process(mess):
             # Check characters to see if they are in punctuation
             nopunc = [char for char in mess if char not in string.punctuation]

             # Join the characters again to form the string.
             nopunc = ''.join(nopunc)

             # Now just remove any stopwords
             return [word for word in nopunc.split() if word.lower() not in stopwor

In [59]: messages.head()

Out[59]:   labels                                            message  length
         0    ham  Go until jurong point, crazy.. Available only ...     111
         1    ham                      Ok lar... Joking wif u oni...      29
         2   spam  Free entry in 2 a wkly comp to win FA Cup fina...     155
         3    ham  U dun say so early hor... U c already then say...      49
         4    ham  Nah I don't think he goes to usf, he lives aro...      61

In [60]: messages['message'].head(5).apply(text_process)

Out[60]: 0    [Go, jurong, point, crazy, Available, bugis, n...
         1                        [Ok, lar, Joking, wif, u, oni]
         2    [Free, entry, 2, wkly, comp, win, FA, Cup, fin...
         3        [U, dun, say, early, hor, U, c, already, say]
         4    [Nah, dont, think, goes, usf, lives, around, t...
         Name: message, dtype: object

In [65]: from sklearn.feature_extraction.text import CountVectorizer

In [66]: bow_transformer = CountVectorizer (analyzer = text_process)

In [67]: bow_transformer.fit(messages['message'])

/Users/arnavsomani/anaconda/lib/python2.7/site-packages/ipykernel/__main__.py:9: Ur


Out[67]: CountVectorizer(analyzer=<function text_process at 0x11b4ea140>, binary=Fa
                 decode_error=u'strict', dtype=<type 'numpy.int64'>,
                 encoding=u'utf-8', input=u'content', lowercase=True, max_df=1.0,
                 max_features=None, min_df=1, ngram_range=(1, 1), preprocessor=None
                 stop_words=None, strip_accents=None,
                 token_pattern=u'(?u)\\b\\w\\w+\\b', tokenizer=None,
                 vocabulary=None)
```

```
In [69]: message4 = messages['message'][3]

In [70]: print message4

U dun say so early hor... U c already then say...

In [71]: bow4 = bow_transformer.transform([message4])

In [84]: print bow4 #bag of words

  (0, 4068)        2
  (0, 4629)        1
  (0, 5261)        1
  (0, 6204)        1
  (0, 6222)        1
  (0, 7186)        1
  (0, 9554)        2

In [75]: print bow_transformer.get_feature_names()[4073]

UIN

In [77]: messages_bow = bow_transformer.transform(messages['message'])

/Users/arnavsomani/anaconda/lib/python2.7/site-packages/ipykernel/__main__.py:9: Ur

In [78]: print 'Shape of Sparse Matrix: ', messages_bow.shape
         print 'Amount of Non-Zero occurences: ', messages_bow.nnz
         print 'sparsity: %.2f%%' % (100.0 * messages_bow.nnz / (messages_bow.shape

Shape of Sparse Matrix:  (5572, 11425)
Amount of Non-Zero occurences:  50548
sparsity: 0.08%

In [83]: from sklearn.feature_extraction.text import TfidfTransformer

         tfidf_transformer = TfidfTransformer().fit(messages_bow)
         tfidf4 = tfidf_transformer.transform(bow4)
         print tfidf4

  (0, 9554)        0.538562626293
  (0, 7186)        0.438936565338
  (0, 6222)        0.318721689295
  (0, 6204)        0.299537997237
  (0, 5261)        0.297299574059
  (0, 4629)        0.266198019061
  (0, 4068)        0.408325899334
```

```
In [85]: print tfidf_transformer.idf_[bow_transformer.vocabulary_['u']]

3.28005242674


In [86]: messages_tfidf = tfidf_transformer.transform(messages_bow)

In [87]: print messages_tfidf.shape

(5572, 11425)


In [89]: from sklearn.naive_bayes import MultinomialNB

In [90]: spam_detect_model = MultinomialNB().fit(messages_tfidf,messages['labels'])

In [93]: print 'Predicted: ', spam_detect_model.predict(tfidf4)[0]
         print 'Expected: ', messages['labels'][3]

 Predicted:  ham
Expected:  ham


In [94]: all_predictions = spam_detect_model.predict(messages_tfidf)
         print all_predictions

['ham' 'ham' 'spam' ..., 'ham' 'ham' 'ham']


In [96]: from sklearn.metrics import classification_report
         print classification_report(messages['labels'], all_predictions)

             precision    recall  f1-score   support

        ham       0.98      1.00      0.99      4825
       spam       1.00      0.85      0.92       747

avg / total       0.98      0.98      0.98      5572


In [105]: from sklearn.cross_validation import train_test_split

          msg_train, msg_test, label_train, label_test = \
          train_test_split(messages['message'], messages['labels'], test_size=0.2)

          print len(msg_train), len(msg_test), len(msg_train) + len(msg_test)

4457 1115 5572
```

```
In [99]: from sklearn.pipeline import Pipeline

         pipeline = Pipeline([
             ('bow', CountVectorizer(analyzer=text_process)),  # strings to token i
             ('tfidf', TfidfTransformer()),  # integer counts to weighted TF-IDF sc
             ('classifier', MultinomialNB()),  # train on TF-IDF vectors w/ Naive B
         ])

In [101]: pipeline.fit(msg_train,label_train)

/Users/arnavsomani/anaconda/lib/python2.7/site-packages/ipykernel/__main__.py:9: Un


Out[101]: Pipeline(steps=[('bow', CountVectorizer(analyzer=<function text_process a
                  decode_error=u'strict', dtype=<type 'numpy.int64'>,
                  encoding=u'utf-8', input=u'content', lowercase=True, max_df=1.0,
                  max_features=None, min_df=1, ngram_range=(1, 1), preprocessor=Non

In [103]: predictions = pipeline.predict(msg_test)

/Users/arnavsomani/anaconda/lib/python2.7/site-packages/ipykernel/__main__.py:9: Un


In [104]: print classification_report(predictions,label_test)

             precision    recall  f1-score   support

        ham       1.00      0.96      0.98       996
       spam       0.75      1.00      0.86       119

avg / total       0.97      0.96      0.97      1115


In [ ]:
```