Analisis de Sentimientos

In [1]:
```
import pandas as pd

filepath_dict = {'yelp':   'yelp_labelled.csv',
                 'amazon': 'amazon_cells_labelled.csv',
                 'imdb':   'imdb_labelled.csv'}

df_list = []
for source, filepath in filepath_dict.items():
    df = pd.read_csv(filepath, names=['sentence', 'label'], sep='\t')
    df['source'] = source
    df_list.append(df)

df = pd.concat(df_list)
print(df.iloc[0])
```

```
sentence    Wow ... Me encantó este lugar.
label                                    1
source                                yelp
Name: 0, dtype: object
```

Ahora hay que vectorizar las oraciones:

In [2]:
```
from sklearn.feature_extraction.text import CountVectorizer
#Prueba con frases a mano
sentences = ['A Juan le gusta el chocolate', 'Juan odia el chocolate']
#vectorizo estas frases a modo de ejemplo
vectorizer = CountVectorizer(min_df=0, lowercase=False)
vectorizer.fit(sentences)
vectorizer.vocabulary_
```

Out[2]:
```
{'Juan': 0, 'le': 4, 'gusta': 3, 'el': 2, 'chocolate': 1, 'odia': 5}
```

In [3]:
```
#transformo el vector en un array para ver cuando aparece cada palabra
vectorizer.transform(sentences).toarray()
```

Out[3]:
```
array([[1, 1, 1, 1, 1, 0],
       [1, 1, 1, 0, 0, 1]], dtype=int64)
```

In [4]:
```
#para entrenar uso los datos de yelp
from sklearn.model_selection import train_test_split
#probar entrenarlo con los 3 concatenados a ver si mejora con mas cantidad de dat
df_yelp = df[df['source'] == 'yelp']

sentences = df_yelp['sentence'].values
y = df_yelp['label'].values

sentences_train, sentences_test, y_train, y_test = train_test_split(
   sentences, y, test_size=0.25, random_state=1000)
```

In [24]:
```
#Divido mis vectores en : Entrenamiento y Test
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.externals import joblib
vectorizer = CountVectorizer()
vectorizer.fit(sentences_train)
joblib.dump(vectorizer, 'vectorizer.pkl')
X_train = vectorizer.transform(sentences_train)
X_test  = vectorizer.transform(sentences_test)
X_train
```

Out[24]:
```
<750x1986 sparse matrix of type '<class 'numpy.int64'>'
        with 7403 stored elements in Compressed Sparse Row format>
```

In [6]:
```
#Genero el clasificador, utilizo una Regresion logistica y entreno el modelo
from sklearn.linear_model import LogisticRegression
from sklearn.externals import joblib
classifier = LogisticRegression()
classifier.fit(X_train, y_train)
score = classifier.score(X_test, y_test)
#Persisto el modelo y imprimo su precisión
joblib.dump(classifier, 'modelo_entrenado.pkl')
print("Precisión:", score)
```

```
D:\Gaston\dev\lib\site-packages\sklearn\externals\joblib\__init__.py:15: Future
Warning: sklearn.externals.joblib is deprecated in 0.21 and will be removed in
0.23. Please import this functionality directly from joblib, which can be insta
lled with: pip install joblib. If this warning is raised when loading pickled m
odels, you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=FutureWarning)

Precisión: 0.788
```

In [26]:
```
#probar con cualquier frase
texto = "hoy ha sido un dia hermoso de mucho calor"
texto = [texto]

new_data = vectorizer.transform(texto)
result = classifier.predict(new_data)
valor = ""
if result[0] == 0:
    valor = "negativo 😤"
elif result[0] == 1:
    valor = "positivo 😁"

print(valor)
```

```
positivo 😁
```

In [ ]: