sa_binary_training_naive_bayes_m

May 29, 2016

0.1 Load already prepared data

```
In [7]: TEST_SIZE=0.4

In [8]: import pandas as pd

In [9]: X = pd.read_csv('../valt_sa_data/x_m.csv')
        y = pd.read_csv('../valt_sa_data/y_m.csv', header=None)[0]
```

0.2 Split data into training and test sets

Let's perform a train/test split with 60% of the data in the training set and 40% of the data in the test set. We use `random_state=0` so that every execution yields the same result.

```
In [10]: from sklearn.cross_validation import train_test_split

         X_train, X_test, y_train, y_test = train_test_split(X.as_matrix(),
                                                             y.as_matrix(),
                                                             test_size=TEST_SIZE,
                                                             random_state=0)
```

1 Train a sentiment classifier with logistic regression

We will now use logistic regression to create a sentiment classifier on the training data.

Note: This line may take a few minutes.

```
In [11]: from sklearn.naive_bayes import MultinomialNB

         clf = MultinomialNB(alpha=0.1)
         model = clf.fit(X_train, y_train)
```

2 Evaluate the trained model

We will now use the cross-validation set to evaluate our model.

In [12]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, model.predict(X_test))

print 'Confusion matrix:'
print cm

from sklearn.metrics import classification_report

print 'Classification report:'
print classification_report(y_test, model.predict(X_test))
```

```
Confusion matrix:
[[ 2113   354   293   149   644]
 [  762   337   524   291   633]
 [  472   286   876   798  1210]
 [  336   147   596  2362  3810]
 [  613   174   389  1935 19896]]
Classification report:
             precision    recall  f1-score   support

          1       0.49      0.59      0.54      3553
          2       0.26      0.13      0.18      2547
          3       0.33      0.24      0.28      3642
          4       0.43      0.33      0.37      7251
          5       0.76      0.86      0.81     23007

avg / total       0.60      0.64      0.62     40000
```