SVM

October 12, 2015

In [4]:
```
import numpy
import urllib
import scipy.optimize
import random
from sklearn import svm
from math import exp
from math import log
```

In [5]:
```
def parseData(fname):
  for l in urllib.urlopen(fname):
    yield eval(l)
```

In [6]:
```
print "Reading data..."
data = list(parseData("file:///users/YW/Dropbox/UCSD/FA2015/CSE255/HW/hw1/book_descriptions_500
print "done"
```

```
Reading data...
done
```

In [7]:
```
D_child = [d for d in data if "Children's Books" in d['categories']]
D_notchild =\
  [d for d in data if not("Children's Books" in d['categories'])][:len(D_child)]
data = D_child + D_notchild
random.seed(0)
random.shuffle(data)
```

In [8]:
```
X = [[1, "child" in s['description'],
         "magic" in s['description'],
         "funny" in s['description'],] for s in data]
```

In [9]:
```
y = ["Children's Books" in d['categories'] for d in data]
```

In [10]:
```
X_train = X[:len(X)/2]
X_test = X[len(X)/2:]
```

In [11]:
```
y_train = y[:len(X)/2]
y_test = y[len(y)/2:]
```

In [12]:
```
clf = svm.SVC(C=1000)
```

In [14]:
```
clf.fit(X_train, y_train)
```

Out[14]:
```
SVC(C=1000, cache_size=200, class_weight=None, coef0=0.0, degree=3, gamma=0.0,
  kernel='rbf', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
```

```
In [15]: train_predictions = clf.predict(X_train)
         test_predictions = clf.predict(X_test)

In [16]: print 'Training Accuracy:%.3f, Testing Accuracy:%.3f'%(1-numpy.mean(train_predictions!=y_train)

Training Accuracy:0.750, Testing Accuracy:0.738

In [18]: X = [[1, "child" in s['description'],
                  "magic" in s['description'],
                  "funny" in s['description'],
                  "kid" in s['description'],
                  "dog" in s['description'],
                  "cat" in s['description'],
                  "education" in s['description'],
                  "pat" in s['description'],
                  "grow" in s['description']] for s in data]
         y = ["Children's Books" in d['categories'] for d in data]
         X_train = X[:len(X)/2]
         X_test = X[len(X)/2:len(X)]
         y_train = y[:len(X)/2]
         y_test = y[len(y)/2:len(y)]
         clf = svm.SVC(C=1000)
         clf.fit(X_train, y_train)
         train_predictions = clf.predict(X_train)
         test_predictions = clf.predict(X_test)
         print 'Training Error:%.3f, Testing Error:%.3f'%(numpy.mean(train_predictions!=y_train), numpy
         print 'Training Accuracy:%.3f, Testing Accuracy:%.3f'%(1-numpy.mean(train_predictions!=y_train)

Training Error:0.233, Testing Error:0.250
Training Accuracy:0.767, Testing Accuracy:0.750

In [19]: X = [[1, "child" in s['description'],
                  "magic" in s['description'],
                  "funny" in s['description'],] for s in data]
         y = ["Children's Books" in d['categories'] for d in data]
         Cs = [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 100000]
         for c in Cs:
             X_train = X[:len(X)/2]
             X_valid = X[len(X)/2:3*len(X)/4]
             X_test = X[3*len(X)/4:]
             y_train = y[:len(y)/2]
             y_valid = y[len(y)/2:3*len(y)/4]
             y_test = y[3*len(y)/4:]
             clf = svm.SVC(C=c)
             clf.fit(X_train, y_train)
             train_predictions = clf.predict(X_train)
             valid_predictions = clf.predict(X_valid)
             test_predictions = clf.predict(X_test)
             print 'c=%s, Train Error:%.3f, Valid Error:%.3f, Test Error:%.3f'%\
             (c, numpy.mean(train_predictions != y_train), \
              numpy.mean(valid_predictions != y_valid), \
              numpy.mean(test_predictions != y_test))

c=0.0001, Train Error:0.492, Valid Error:0.509, Test Error:0.507
c=0.001, Train Error:0.492, Valid Error:0.509, Test Error:0.507
```

```
c=0.01, Train Error:0.252, Valid Error:0.254, Test Error:0.273
c=0.1, Train Error:0.252, Valid Error:0.254, Test Error:0.273
c=1, Train Error:0.250, Valid Error:0.251, Test Error:0.272
c=10, Train Error:0.250, Valid Error:0.251, Test Error:0.272
c=100, Train Error:0.250, Valid Error:0.251, Test Error:0.272
c=1000, Train Error:0.250, Valid Error:0.251, Test Error:0.272
c=100000, Train Error:0.250, Valid Error:0.251, Test Error:0.272
```

In [ ]:

In [ ]: