try2

July 23, 2016

In [5]:
```
import numpy as np
import pandas as pd
from copy import copy
from sklearn.ensemble import RandomForestRegressor
import csv
import matplotlib.pyplot as plt

dateparse=lambda x:pd.datetime.strptime(x,'%Y-%m-%d %H:%M:%S')
train=pd.read_csv('train.csv',parse_dates=['datetime'],date_parser=dateparse)
test=pd.read_csv('test.csv',parse_dates=['datetime'],date_parser=dateparse)


def extractFeaturesTrain(data):

    data['Hour']=data.datetime.dt.hour
    labels=data['count']
    train_years=data.datetime.dt.year
    train_months=data.datetime.dt.month
    data=data.drop(['datetime','count','casual','registered'], axis = 1)

    return np.array(data),np.array(labels),np.array(train_years),np.array(train_months),(data.c

def extractFeaturesTest(data):

    data['Hour']=data.datetime.dt.hour
    test_years=data.datetime.dt.year
    test_months=data.datetime.dt.month
    data=data.drop(['datetime'], axis = 1)
    return np.array(data),np.array(test_years),np.array(test_months)

train2=copy(train)
test2=copy(test)
test=np.array(test)
traind,labelsTrain,train_years,train_months,headers=extractFeaturesTrain(train2)
testd,test_years,test_months=extractFeaturesTest(test2)

submit=np.array((test.shape[0],2))

#train.to_csv('Remodeled Train.csv')
train=np.array(train)
print 'train is \n',traind.shape
print 'labels train are \n',labelsTrain.shape
print 'test is \n',testd.shape
```

```
def findLocations(year,month):
    locs=[]
    for i in range(0,test.shape[0]):
        if(test[i][0].year==year and test[i][0].month==month):
            locs.append(i)

    return locs

def findValidDates(year,month):
    locs=[]
    for i in range(0,train.shape[0]):
        if(train[i][0].year<=year and train[i][0].month<=month):
            locs.append(i)

    return locs

'''for i in set(test_years):
    for j in set(test_months):
        print 'Year : ',i,' month ',j:
            testLocs=findLocations(i,j)
            testSubset=testd[testLocs]

            trainLocs=findValidDates(i,j)
            trainSubset=traind[trainLocs]'''

def findLoss(gold,predicted):
    loss=0
    for i in range(gold.shape[0]):
        loss+=(np.log(predicted[i]+1) -np.log(gold[i]+1))**2

    loss=loss/gold.shape[0]
    return np.sqrt(loss)

rf=RandomForestRegressor()
split1=0.8*traind.shape[0]
trainSplit=traind[:split1,:]

testSplit=traind[split1:,:]
labelsSplitTrain=labelsTrain[:split1]
labelsSplitTest=labelsTrain[split1:]
rf.fit(trainSplit,labelsSplitTrain)
ypred=rf.predict(testSplit)
print 'trainSplit is \n',trainSplit.shape,' and testSplit is \n',testSplit.shape
print 'ypred is \n',ypred
print 'test split is \n',labelsSplitTest
print 'the loss is ',findLoss(labelsSplitTest,ypred)



rf.fit(traind,labelsTrain)
print 'testd shape is ',testd.shape
ypred2=rf.predict(testd)
```

```
        with open('submit2.csv', 'wb') as csvfile:
            resultWriter= csv.writer(csvfile)
            l=['datetime','count']
            resultWriter.writerow(l)
            for i in range(testd.shape[0]):
                #print 'test[',i,'][0] is ',test[i,0]
                l=[test[i,0],ypred2[i]]
                resultWriter.writerow(l)

        importances=rf.feature_importances_
        std=np.std([tree.feature_importances_ for tree in rf.estimators_],axis=0)
        indices=np.argsort(importances)[::-1]
        print 'Feature Ranking\n'

        for f in range(traind.shape[1]):
            print("%d. feature %d %s (%f)" % (f + 1, indices[f],headers[indices[f]], importances[indices

        fig, ax = plt.subplots()

        ax.set_title('Feature Importances')
        ax.bar(range(traind.shape[1]),importances[indices],color="b",yerr=std[indices],align='center')
        plt.xticks(range(traind.shape[1]), indices)
        ax.set_xlim([-1, traind.shape[1]])
        ax.set_xticklabels(headers[indices])
        plt.show()
```

```
train is
(10886, 9)
labels train are
(10886,)
test is
(6493, 9)
trainSplit is
(8708, 9)  and testSplit is
(2178, 9)
ypred is
[  14.5   29.    52.9 ...,  148.3  107.5   71.6]
test split is
[ 19  19  68 ..., 168 129  88]
the loss is  0.472393892624
testd shape is  (6493, 9)
Feature Ranking

1. feature 8 Hour (0.602912)
2. feature 4 temp (0.111754)
3. feature 6 humidity (0.071418)
4. feature 2 workingday (0.066232)
5. feature 5 atemp (0.043270)
6. feature 0 season (0.040731)
7. feature 7 windspeed (0.037909)
8. feature 3 weather (0.021794)
9. feature 1 holiday (0.003980)
```

```
/usr/local/lib/python2.7/dist-packages/ipykernel/__main__.py:80: DeprecationWarning: using a non-integer
/usr/local/lib/python2.7/dist-packages/ipykernel/__main__.py:82: DeprecationWarning: using a non-integer
/usr/local/lib/python2.7/dist-packages/ipykernel/__main__.py:83: DeprecationWarning: using a non-integer
/usr/local/lib/python2.7/dist-packages/ipykernel/__main__.py:84: DeprecationWarning: using a non-integer
```

```
In [7]: set(test_months)
```

```
Out[7]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}
```

```
In [22]: def getTestLocs(year,month):
             locs=[]
             print 'In testlocs year is =',year,' month is = ',month
             for i in range(0,test.shape[0]):
                 if test[i][0].year==year and test[i][0].month==month:
                     locs.append(i)
             return locs
```

```
In [24]: rf2=RandomForestRegressor()
         set(test_years)
```

```
Out[24]: {2011, 2012}
```

```
In [26]: with open('submit3.csv','wb') as csvfile:
             resultWriter=csv.writer(csvfile)
             l=['datetime','count']
             resultWriter.writerow(l)
             for i in set(test_years):
                 for j in set(test_months):
                         testLocs=getTestLocs(i,j)
                         #print 'testLoics are ',testLocs

                         testSubset1=testd[testLocs]
                         testSubset2=test[testLocs]
                         #print 'testSubset2 is ',testSubset2
                         trainLocs=np.where(train[:,0]<=min(testSubset2[:,0]))
                         trainSubset=traind[trainLocs]
                         labelsSubset=labelsTrain[trainLocs]
                         rf2.fit(trainSubset,labelsSubset)
                         ypred3=rf2.predict(testSubset1)
                         for k in range(0,testSubset2.shape[0]):
                             l=[testSubset2[k,0],ypred3[k]]
                             resultWriter.writerow(l)
```

```
In testlocs year is = 2011  month is =  1
In testlocs year is = 2011  month is =  2
In testlocs year is = 2011  month is =  3
In testlocs year is = 2011  month is =  4
In testlocs year is = 2011  month is =  5
In testlocs year is = 2011  month is =  6
In testlocs year is = 2011  month is =  7
In testlocs year is = 2011  month is =  8
In testlocs year is = 2011  month is =  9
In testlocs year is = 2011  month is =  10
In testlocs year is = 2011  month is =  11
In testlocs year is = 2011  month is =  12
```

```
In testlocs year is = 2012  month is =  1
In testlocs year is = 2012  month is =  2
In testlocs year is = 2012  month is =  3
In testlocs year is = 2012  month is =  4
In testlocs year is = 2012  month is =  5
In testlocs year is = 2012  month is =  6
In testlocs year is = 2012  month is =  7
In testlocs year is = 2012  month is =  8
In testlocs year is = 2012  month is =  9
In testlocs year is = 2012  month is =  10
In testlocs year is = 2012  month is =  11
In testlocs year is = 2012  month is =  12
```

```
In [12]: min(train[:,0])
```

```
Out[12]: Timestamp('2011-01-01 00:00:00')
```