from PDFs to CDFs 2-9

February 11, 2016

In [47]: %matplotlib inline

```python
import numpy as np  # matrices and data structures
import scipy.stats as ss  # standard statistical operations
import pandas as pd  # keeps data organized, works well with data
import matplotlib.pyplot as plt  # plot visualization

# import same data as last time:
nyw = pd.read_csv('NYC-CParkWeather.csv')
nyw = nyw.set_index('year')  # represents observations
```

In [169]: # inclass measurement data
classX = pd.DataFrame(np.array([4.5, 4.7, 4.6, 4.5, 4.1, 4.2, 4.2, 4.1, 4.2, 4.7, 4.7, 4.6, 4.7, 4.7, 4.7]))
classX.columns=['width estimate']
classX.hist(bins=8)

Out[169]: array([[<matplotlib.axes.AxesSubplot object at 0x7fc32be6af10>]], dtype=object)

```
In [172]: #X.describe()
newX = classX.loc[classX['width estimate']<4.4] #loc returns elements at the given locations
newX.hist()
newX.describe()

Out[172]:   width estimate
                count  22.000000
                mean  4.200000
                std  0.061721
                min  4.100000
               25%  4.200000
               50%  4.200000
               75%  4.200000
                max  4.300000

In [173]: nyw_m=nyw['mean']
   nyw_m.hist(bins=15)

Out[173]: <matplotlib.axes.AxesSubplot at 0x7fc32b70a9d0>
In [67]: nyw.m.plot()

Out[67]: <matplotlib.axes.AxesSubplot at 0x7fc32cac0b50>
In [148]: #import KDE
from sklearn.neighbors import KernelDensity

def kde_plot(kernel, X, color="#aaaaff", bw = 1):
    #create the estimator:
    kde_X = KernelDensity(kernel=kernel, bandwidth=bw).fit(X)

    #setup range:
    range = np.linspace(X.min()-bw*3, X.max()+bw*3, 1000)[,:np.newaxis]

    #plot:
    plt.fill(range[:,0], np.exp(kde_X.score_samples(range)), fc=color, alpha=.6)
    dots = [y-np.random.rand()*.005 for y in np.zeros(X.shape[0])] #all points, randomly jittered
    plt.plot(X[:,0], dots, '+k', color=color)

Xnyw_m = nyw_m.reshape(-1,1)
kde_plot('tophat', Xnyw_m)
plt.show()
kde_plot('gaussian', Xnyw_m, '#22aa22')
kde_plot('epanechnikov', Xnyw_m, '#ffbbbb')
In [146]: # let's see what each kernel looks like:
X = np.array([[0]])
kde_plot('tophat', X, bw=2)
kde_plot('gaussian', X, '#aaffaa', bw=2)
kde_plot('epanechnikov', X, '#ffaaaa', bw=2)
In [158]: #Pandas also has KDE built in:
    nyw_m.plot(kind='kde')

Out[158]: <matplotlib.axes.AxesSubplot at 0x7fc32be56050>

In [180]: #KDE on class measurement data.
    # let's see whether pandas' KDE matches Gaussian or Epan
classX.plot(kind='kde')
kde_plot('gaussian', classX.as_matrix(), bw=0.12, color='#44cc44')
kde_plot('epanechnikov', classX.as_matrix(), bw=0.3)
#pandas seems to be using a Gaussian kernel with bandwidth optimized to .12