assign7_1_prob_classification

May 31, 2019

In [1]:
```
import numpy as np
np.random.seed(1)

import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
def generate_data(sample_size=90, n_class=3):
    x = (np.random.normal(size=(sample_size // n_class, n_class))
         + np.linspace(-3., 3., n_class)).flatten()
    y = np.broadcast_to(np.arange(n_class),
                        (sample_size // n_class, n_class)).flatten()
    return x, y

x, y = generate_data()
```

In [3]:
```
# 各クラスのサンプル、サンプル数
n = len(x)
cs = np.unique(y)
n_class = len(cs)

indices_cs = [np.where(y==c) for c in cs]
x_cs = [x[indices_c] for indices_c in indices_cs]
n_cs = [len(x_c) for x_c in x_cs]
```

In [4]:
```
# 各クラスの計画行列
def calc_design_matrix(x, c, h=1):
    return np.exp(-(x[None] - c[:, None]) ** 2 / (2 * h ** 2))

ks = [calc_design_matrix(x_c, x) for x_c in x_cs]
```

In [5]:
```
# 各クラスの one_hot ベクター
def one_hot(indices_c):
    zeros = np.zeros(n, dtype=np.float32)
    zeros[indices_c] = 1
    return zeros

pis = [one_hot(indices_c) for indices_c in indices_cs]
```

In [6]:
```
# 最小二乗法によるパラメータ推定
l = 0.01
thetas = [np.linalg.inv(k.T.dot(k) + l * np.eye(n_c)).dot(k.T).dot(pi) for k, n_c, pi in zip(ks, n_cs
```

In [7]:
```
# 確率分布可視化用サンプル
x_vis = np.linspace(start=-5, stop=5, num=1000)
ks_vis = [calc_design_matrix(x_c, x_vis) for x_c in x_cs]
```

In [8]:
```
# 各クラスの確率分布
ls_vis = np.maximum([k_vis.dot(theta) for k_vis, theta in zip(ks_vis, thetas)], 0)
ps_vis = ls_vis / np.sum(ls_vis, 0)
```

In [9]:
```
# 可視化
plt.scatter(x_cs[0], np.zeros(n_cs[0]))
plt.scatter(x_cs[1], np.zeros(n_cs[1]))
plt.scatter(x_cs[2], np.zeros(n_cs[2]))
plt.plot(x_vis, ps_vis[0])
plt.plot(x_vis, ps_vis[1])
plt.plot(x_vis, ps_vis[2])
```

Out[9]: [<matplotlib.lines.Line2D at 0x7f0e4817e550>]