Sebastian Raschka
last modified: 03/30/2014

Problem Category

- Statistical Pattern Recognition
- Supervised Learning
- Parametric Learning
- Bayes Decision Theory
- Univariate data
- 2-class problem
- different variances
- Gaussian model (2 parameters)
- No Risk function

Given Information:

model: continuous univariate normal (Gaussian) density

$p(x|\omega_j) \sim N(\mu|\sigma^2)$

$p(x|\omega_j) \sim \frac{1}{\sqrt{2\pi\sigma^2}} \exp{ \bigg[-\frac{1}{2}\bigg(\frac{x-\mu}{\sigma}\bigg)^2 \bigg] }$

parameters:

$P(\omega_1) = P(\omega_2) = 0.5$

$\sigma_1^2 = 4, \quad \sigma_2^2 = 1$

$\mu_1 = 4, \quad \mu_2 = 10$

Bayes' Rule:

$P(\omega_j|x) = \frac{p(x|\omega_j) * P(\omega_j)}{p(x)}$

Bayes' Decision Rule:

Decide ω_1 if $P(\omega_1|x) > P(\omega_2|x)$ else decide ω_2.

$$\Rightarrow \frac{p(x|\omega_1) * P(\omega_1)}{p(x)} > \frac{p(x|\omega_2) * P(\omega_2)}{p(x)}$$

Bayes' Decision Rule:

Decide ω_1 if $P(\omega_1|x) > P(\omega_2|x)$ else decide ω_2.

$$\Rightarrow \frac{p(x|\omega_1) * P(\omega_1)}{p(x)} > \frac{p(x|\omega_2) * P(\omega_2)}{p(x)}$$

We can drop $p(x)$ since it is just a scale factor.

$\Rightarrow P(x|\omega_1) * P(\omega_1) > p(x|\omega_2) * P(\omega_2) \;\Rightarrow \frac{p(x|\omega_1)}{p(x|\omega_2)} > \frac{P(\omega_2)}{P(\omega_1)} \;\Rightarrow \frac{p(x|\omega_1)}{p(x|\omega_2)} > \frac{0.5}{0.5} \;\Rightarrow \frac{p(x|\omega_1)}{p(x|\omega_2)} > 1$ `$ \Rightarrow \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp{ \bigg[-\frac{1}{2}\bigg` `\frac{x-\mu_1}{\sigma_1}\bigg)^2 \bigg] } > \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp{ \bigg[-\frac{1}{2}\bigg( \frac{x-\mu_2}{\sigma_2}\bigg)^2 \bigg] } \quad \bigg| \quad ln`

$\Rightarrow ln(1) - ln\bigg({\sqrt{2\pi\sigma_1^2}}\bigg) - \frac{1}{2}\bigg(\frac{x-\mu_1}{\sigma_1}\bigg)^2 > ln(1) - ln\bigg({\sqrt{2\pi\sigma_2^2}}\bigg) - \frac{1}{2}\bigg(\frac{x-\mu_2}{\sigma_2}\bigg)^2 \quad \bigg| \quad \sigma_1^2 = 4, \quad \sigma_2^2 = 1, \quad \mu_1 = 4, \quad \mu_2 = 10$

$\Rightarrow -ln({\sqrt{2\pi4}}) - \frac{1}{2}\bigg(\frac{x-4}{2}\bigg)^2 > -ln({\sqrt{2\pi}}) - \frac{1}{2}(x-10)^2$

$$\Rightarrow -\frac{1}{2}\, ln(2\pi) - ln(2) - \frac{1}{8}\,(x-4)^2 > -\frac{1}{2}\, ln(2\pi) - \frac{1}{2}\,(x-10)^2 \quad \Big| \times\ 2$$

$$\Rightarrow -ln(2\pi) - 2ln(2) - \frac{1}{4}\,(x-4)^2 > -ln(2\pi) - (x-10)^2 \quad \Big| + ln(2\pi)$$

$$\Rightarrow -4ln(4) - (x-4)^2 > -4(x-10)^2$$

$$\Rightarrow -ln(4) - \frac{1}{4}\,(x-4)^2 > -(x-10)^2 \quad \Big| \times\ 4$$

$$\Rightarrow -8ln(2) - x^2 + 8x - 16 > -4x^2 + 80x - 400$$

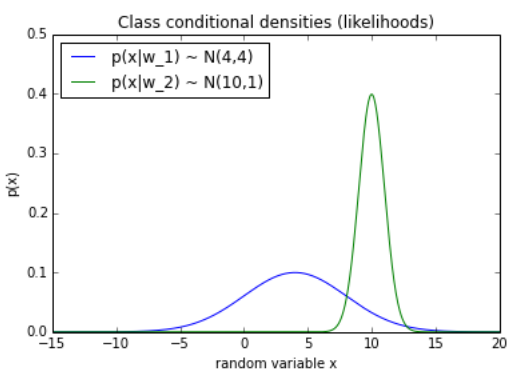
$$\Rightarrow 3x^2 - 72x + 384 - 8ln(2) > 0$$

$$\Rightarrow x < 7.775 \quad and \quad x > 16.225$$

Plotting the Class Conditional Densities and Decision Boundary

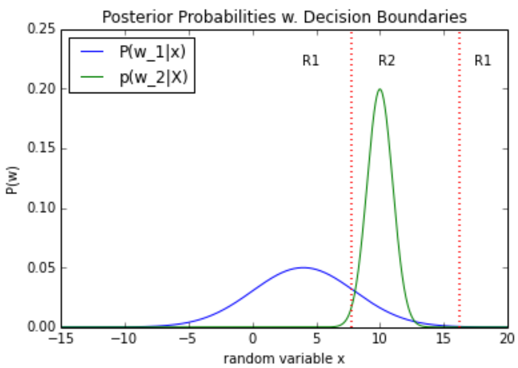
In [6]:
```
%pylab inline

import numpy as np
from matplotlib import pyplot as plt

def pdf(x, mu, sigma):
    """
    Calculates the normal distribution's probability density
    function (PDF).

    """
    term1 = 1.0 / ( math.sqrt(2*np.pi) * sigma )
    term2 = np.exp( -0.5 * ( (x-mu)/sigma )**2 )
    return term1 * term2

# generating some sample data
x = np.arange(-100, 100, 0.05)

# probability density functions
pdf1 = pdf(x, mu=4, sigma=4)
pdf2 = pdf(x, mu=10, sigma=1)

# Class conditional densities (likelihoods)
plt.plot(x, pdf1)
plt.plot(x, pdf2)
plt.title('Class conditional densities (likelihoods)')
plt.ylabel('p(x)')
plt.xlabel('random variable x')
plt.legend(['p(x|w_1) ~ N(4,4)', 'p(x|w_2) ~ N(10,1)'], loc='upper left')
plt.ylim([0,0.5])
plt.xlim([-15,20])
plt.show()
```

```
Populating the interactive namespace from numpy and matplotlib
```



In [23]:
```
def posterior(likelihood, prior):
    """
    Calculates the posterior probability (after Bayes Rule) without
    the scale factor p(x) (=evidence).

    """
    return likelihood * prior

# probability density functions
posterior1 = posterior(pdf(x, mu=4, sigma=4), 0.5)
posterior2 = posterior(pdf(x, mu=10, sigma=1), 0.5)
```

```
# Class conditional densities (likelihoods)
plt.plot(x, posterior1)
plt.plot(x, posterior2)
plt.title('Posterior Probabilities w. Decision Boundaries')
plt.ylabel('P(w)')
plt.xlabel('random variable x')
plt.legend(['P(w_1|x)', 'p(w_2|X)'], loc='upper left')
plt.ylim([0,0.25])
plt.xlim([-15,20])
plt.axvline(7.775, color='r', alpha=0.8, linestyle=':', linewidth=2)
plt.axvline(16.225, color='r', alpha=0.8, linestyle=':', linewidth=2)
plt.annotate('R2', xy=(10, 0.2), xytext=(10, 0.22))
plt.annotate('R1', xy=(4, 0.2), xytext=(4, 0.22))
plt.annotate('R1', xy=(17, 0.2), xytext=(17.5, 0.22))
plt.show()
```



In [ ]: