6.867: Exercises (Week 3)

Sept 8, 2016

Since there are no recitations this week, here's how we suggest you go through the exercises:

- 2 star exercises (4, 5, 9.1, 9.2, 10, 11) are the ones you should prioritize, to make sure you understand the underlying concepts for regression, discriminant analysis and Naïve Bayes.
- 1 star exercises (6, 12) are also very good practice, and will provide a more in-depth understanding of the material.
- And of course, feel free to take a look at all the others! They either expand on this week's material even further, or go over material covered in the previous weeks.

Contents

Solution: Don't look at the solutions until you have tried your absolute hardest to solve the problems.

1 Two stories

Which of these stories about Bayesian estimation is correct?

1. There is a random variable V (such as the distance a car will travel after I step on the brakes). We think it has a Gaussian distribution, and that the variance of that distribution is σ^2. We are not sure about the mean M of that distribution, and we model our lack of knowledge about the mean of that distribution using a Gaussian with mean μ_0 and variance σ_0^2. Our data consists of samples of that random variable, which we use to compute a new distribution, parameterized by μ_n and σ_n, on M.

2. There is an actual non-random quantity ν in the world (such as the distance a car will travel) but we don't know it. We model our lack of knowledge of ν using a Gaussian distribution with mean μ_0 and variance σ_0^2. We make measurements of that quantity that are distributed with mean ν and variance σ^2, which we use to update our "belief" about ν in the form of a Gaussian with parameters μ_n and σ_n.

Solution: Both are valid! The difference lies in whether ν is a fixed, true quantity in the world (e.g. the speed of light) or a random variable V (e.g. stopping time).

In the first case, there is true randomness, and putting a distribution over V and performing Bayesian inference is pretty natural. This 'true' randomness is called aleatoric uncertainty.

In the second case we put a distribution on ν because of our ignorance about true knowledge of V; in this scenario, we're dealing with epistemic uncertainty.

Bayesian statistics do not care about the source of uncertainty, whether it be due to true randomness or uncertainty about the true value of a parameter.

2 Bayesian astronomy

You are making observations of the position of a star, in the 2D coordinate frame of your telescope. You believe you have pointed your scope at the star, and that it generates errors with standard deviation of 10 pixels. The covariance Σ_D of the observations is $((100, 0), (0, 100)$. Your prior on the position of the star is Normal, with mean $\mu = (0, 0)$ and covariance $\Sigma = ((10, 5), (5, 10)$.

1. You make an observation of $(3, 4)$. What is your posterior on the star's position?

Solution: `(0.4327, 0.4803), ((8.9027, 4.1408), (4.1408, 8.9027))`

2. You make another observation, this time at $(2, 1)$. Now what is the star's position?

Solution: `(0.5769, 0.5769), ((8.0420, 3.4965), (3.4965, 8.0420))`

3 Car Bocce

This problem needs to be solved numerically.

You just bought a new car and are going to compete in an exciting new extreme sport called "Car Bocce."[1] The idea is to drive up to a wall and stop as close to it as possible without hitting it. The cars in this sport are specially modified so that they move forward at a fixed velocity until the driver pushes a "brake" button, at which point they brake as hard as possible until they come to rest. The driver's job is to select a distance d from the wall at which to push the brake button.

The *braking distance* B of your car is how far it will travel after you have pushed the brake button; it is stochastic, so B is a random variable. You think the braking distance is well modeled with a Gaussian distribution with $\sigma = 1$m, but you are uncertain about the mean. You model your belief about the mean with $\mu_0 = 10$ and $\sigma_0^2 = 100$.

The loss in this problem is 1000 if you hit the wall and otherwise $d - b$ where b was the actual braking distance.

(a) What is the optimal distance d^* at which to push the brake button? If you follow that strategy how likely is it that you will crash? What is your risk?

(b) You try this for a few times and (miraculously!) your car remains intact. You update your belief about the mean with the data you gathered that way and so the posterior distribution on the mean of B's distribution is $\mu_5 = 10$ and $\sigma_5^2 = 4^2$.

Now what is the optimal d^*? What is the likelihood of a crash? What is the risk?

Solution:

(a) The distribution of breaking distance B is:

$$B \sim N(\mu_0, \sigma^2 + \sigma_0^2) = N(10, 101)$$

Then the optimal distance d^* should minimize the risk as:

$$d^* = \min_d R(d) = \min_d p(b \geqslant d)1000 + \int_0^d p(b)(d - b)\,db$$

We can use mathematica to numerically solve this as:

```
R[d_,mu_,sigma_]:= (1-CDF[NormalDistribution[mu, sigma],d])*1000.0
 + NIntegrate[PDF[NormalDistribution[mu, sigma],b]*(d-b),{b,0,d}]
RStar[mu_, sigma_]:= Minimize[R[d, mu, sigma], d]
```

[1] Bocce (also known as boules or petanque) is a game in which one tries to roll balls so that they stop as close as possible to another ball. Our game also resembles a game played by kids in Israel who try to throw apricot pits close to a wall (and penny-pitching, which is the same, but in the US with pennies).

This gives a optimal stopping distance of 37.91, a risk of 23.75 and a collision probability of 0.00273.

(b) After five tries, the posterior distribution of breaking distance b is:

$$B \mid \mathcal{D} \sim N(\mu_5, \sigma^2 + \sigma_5^2) = N(10, 17)$$

And now the optimal distance d^* is the minimizer of:

$$d^* = \min_d p(b \geqslant d)1000 + \int_0^d p(b \mid \mathcal{D})(d - b)\, db$$

Performing the same calculation as before, we obtain the optimal stopping distance of 22.48, risk of 13.5348 and hitting probability of 0.001235

4 Bayesian Regression **

In this problem we will consider the standard Bayesian approach to linear regression, in which we put a Gaussian prior on the weights. Assume $x^{(i)} \in \mathbb{R}^2$, where the first feature of each $x^{(i)}$ is 1.

$$\begin{aligned} Y \mid X &\propto \text{Normal}(W^T X, 1) \\ W &\propto \text{Normal}(\mathbf{0}, \mathbf{I}) \end{aligned}$$

The figure below has some plots of $\Pr(W \mid \mathcal{D})$ and $\Pr(\mathcal{D} \mid W)$ for different values of $\mathcal{D}$. Each plot is in the space of weights, indexed by w_1 and w_2, so that the mean of $\Pr(y \mid x_2) = w_1 + w_2 x_2$.

In the densities, the smallest contour contains 10% of the probability mass, and each larger contour is the next decile. In the likelihood plots, the brighter areas have higher density.

When writing the data below we are showing the data points written as (x_2, y) pairs (since the x_1 value for each item is 1).

For each of the following quantities, indicate which plot corresponds to it, or **None** if none of them do.

(a) $\Pr(W)$

◯ A ◯ B ◯ C ◯ D ◯ E ◯ F ◯ G ◯ H √ **I** ◯ None

(b) $\Pr(\mathcal{D} = \{(1,1)\} \mid W)$

◯ A ◯ B ◯ C ◯ D ◯ E ◯ F ◯ G √ **H** ◯ I ◯ None

(c) $\Pr(\mathcal{D} = \{(-1,-1)\} \mid W)$

◯ A ◯ B ◯ C √ **D** ◯ E ◯ F ◯ G ◯ H ◯ I ◯ None

(d) $\Pr(\mathcal{D} = \{(0,-1)\} \mid W)$

√ **A** ◯ B ◯ C ◯ D ◯ E ◯ F ◯ G ◯ H ◯ I ◯ None

(e) $\Pr(W \mid \mathcal{D} = \{(1,1)\})$

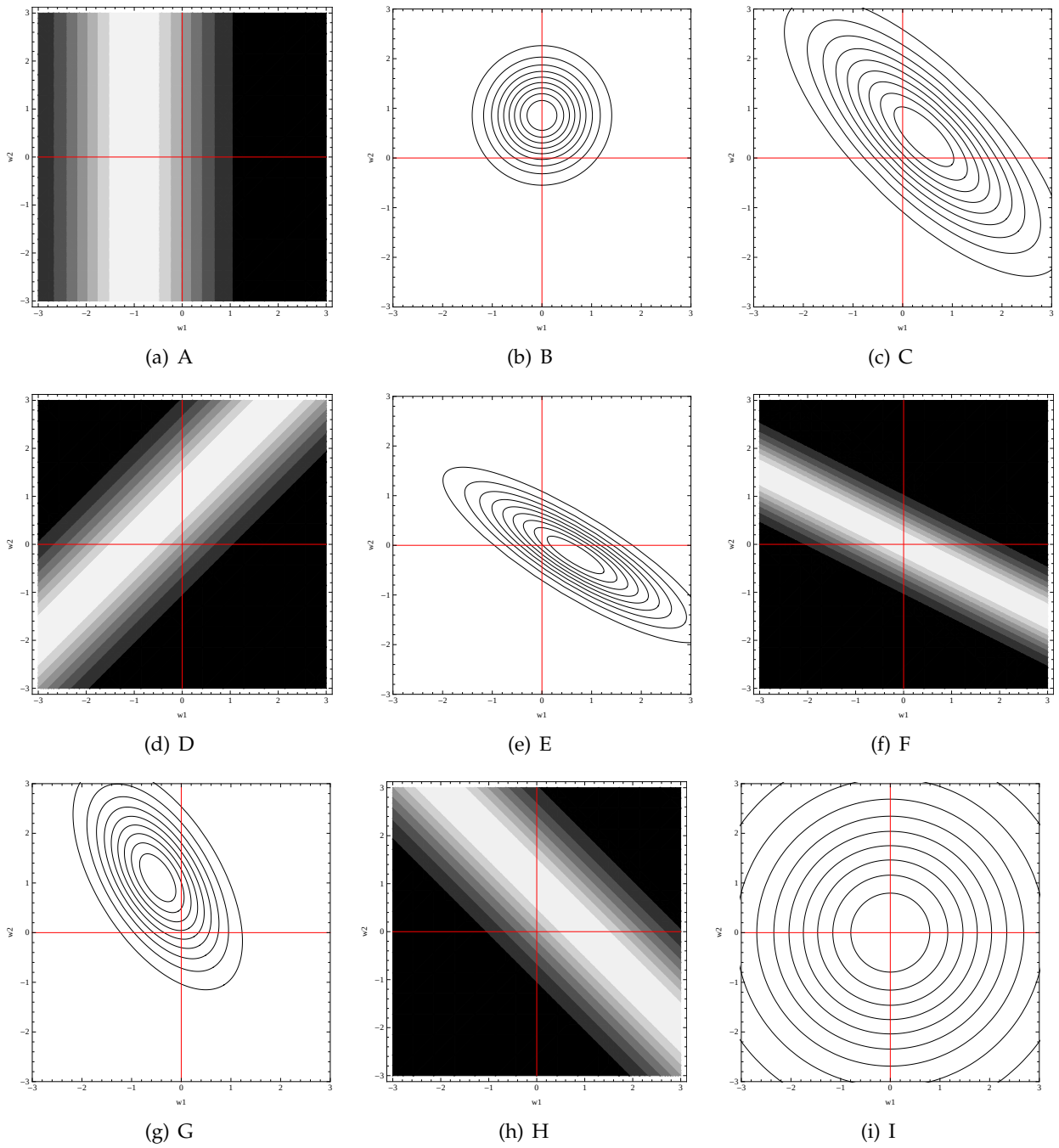
◯ A ◯ B √ **C** ◯ D ◯ E ◯ F ◯ G ◯ H ◯ I ◯ None

(f) $\Pr(W \mid \mathcal{D} = \{(1,1),(-1,-1)\})$

◯ A √ **B** ◯ C ◯ D ◯ E ◯ F ◯ G ◯ H ◯ I ◯ None

(g) $\Pr(W \mid \mathcal{D} = \{(1,1),(0,-1)\})$

◯ A ◯ B ◯ C ◯ D ◯ E ◯ F √ **G** ◯ H ◯ I ◯ None



(a) A (b) B (c) C

(d) D (e) E (f) F

(g) G (h) H (i) I

Figure 1: **Linear Regression Plots**

5 The New Normal **

1. You're an hour late for 6.867 lecture and walk in to see the following formulas written on the board; the lecture seems to have been about a Bayesian approach to linear regression. There are several symbols that seem to denote some form of variance. Pick which of the following explanations go with each one, or argue that it doesn't apply.

$$\begin{aligned} p(\theta) &= \mathcal{N}(\theta \mid \mu_0, \Sigma_0) \\ p(y \mid x, \theta) &= \mathcal{N}(\theta^T x, \sigma^2) \\ p(\theta \mid \mathcal{D}) &= \mathcal{N}(\theta \mid \mu_n, \Sigma_n) \\ \mu_n &= \Sigma_n(\Sigma_0^{-1}\mu_0 + (1/\sigma^2)\Phi^T Y) \\ \Sigma_N^{-1} &= \Sigma_0^{-1} + \frac{1}{\sigma^2}\Phi^T\Phi \\ p(y^{(n+1)} \mid x^{(n+1)}, \mathcal{D}) &= \mathcal{N}(y \mid \mu_n^T x^{(n+1)}, {x^{(n+1)}}^T \Sigma_N x^{(n+1)} + \sigma^2) \end{aligned}$$

   (a) Variance of prior prediction.

   (b) Variance of posterior prediction.

   (c) Prior variance on mean of weight distribution.

   (d) Variance of measurements.

   (e) Posterior variance on mean of weight distribution.

   Solution:

   (a) Variance of prior prediction: $x^T\Sigma_0 x + \sigma^2$.

   (b) Variance of posterior prediction: $x^T\Sigma_n x + \sigma^2$.

   (c) Prior variance on mean of weight distribution: Σ_0

   (d) Variance of measurements: σ^2

   (e) Posterior variance on mean of weight distribution: Σ_n

2. You have just discovered a time machine and want to use it to regress back in time to your first birthday. There is a big knob that seems to be freely turnable in both directions; when you turn it, there is a numeric "read-out" on the console of the time machine that varies linearly with the amount the knob is turned. Right now, the numbers on the display read 2016.75, which happens to be the current time, measured in years. You think that the amount the knob is turned correlates with how far forward or backward in time the machine goes.

   You begin to do some experiments. You find that when you arrive at a new time, you can estimate the year, with a standard deviation of about 2 years. Your best guess, initially, is that the the display is in direct correspondence with the date, but you assign a variance of 1 to the parameters of the linear dependence and you don't think the parameters are correlated.

(a) You turn the knob to 2000. What is your distribution on what year you will end up in?

(b) Once there, you realize that the year is 1015. What is your distribution on the parameters governing the relationship between the knob and the year?

(c) You turn the knob to 2010. What is your distribution on the year you will end up in?

(d) What are the numeric values of the variances listed at the end of the previous question?

Solution:

(a) Let x be the year shown on the machine, and y be the estimated year. Then the relationship between x and y is:

$$p(y \mid \theta) = \mathcal{N}(y \mid \theta_0 + \theta_1 x, 4),$$

The prior on θ is

$$p(\theta) = \mathcal{N}((\theta_0, \theta_1) \mid (0,1), I) \quad .$$

The predicted distribution of the year we will arrive at when we turn the knob to 2000 is the expected output given the posterior on the weights and an input of $(1, 2000.0)$:

$$p(y \mid 2000.0) = \mathcal{N}(y \mid (0,1)^T(1, 2000.0), (1, 2000.0)^T I(1, 2000.0) + 4)$$
$$p(y \mid 2000.0) = \mathcal{N}(y \mid 2000.0, 4000005.0)$$

Whoa! That's a big variance. Why? Uncertainty about the slope has a big lever when we're predicting a point so far away.

(b) Now, we get one observation $(x^{(1)}, y^{(1)}) = ((1, 2000.0), 1015)$. Looking at the previous part, we find

$$\mu_n = \Sigma_n \left(I(0,1) + \frac{1}{4}(1, 2000) \times 1015 \right)$$
$$\Sigma_n = \left(I + \frac{1}{4} \begin{pmatrix} 1 & 2000 \\ 2000 & 2000^2 \end{pmatrix} \right)^{-1}$$

Working this out we get:

$$\mu_n = (-0.00024, 0.507)$$
$$\Sigma_n = \frac{1}{4000005} \begin{pmatrix} 4000004 & -2000 \\ -2000 & 5 \end{pmatrix}$$

(c) Now, our distribution on what year we will go to when we turn the knob to 2010 is

$$y^{(n+1)} \sim N(y | \mu_n^T x^{(n+1)}, x^{(n+1)\top} \Sigma_n x^{(n+1)} + \sigma^2)$$
$$= N(y | 1020.075, 8.04)$$

(d) Let $x = \{1, 2010\}$, then:

Variance of prior prediction: $x^T \Sigma_0 x + \sigma^2 = 4040105$.

Variance of posterior prediction: $x^T \Sigma_n x + \sigma^2 = 8.04$.

Prior variance on mean of weight distribution: $\Sigma_0 = I$

Variance of measurements: $\sigma^2 = 4$

Posterior variance on mean of weight distribution: Σ_n (see part b).

6 Where did you put the map? *

In class, we derived the ridge regression estimator for linear regression with a prior by taking the derivative, setting it to 0, and solving. An alternative is to observe that the posterior distribution is a Gaussian and is, therefore unimodal and symmetric. Consequently, its mean is also its mode.

What is its mode? How is it related to the ridge-regression result?

Solution: Recall that the posterior distribution of the regression weights θ conditional on the data $\mathcal{D}$ follows (from problem 2):

$$\begin{aligned} p(\theta \mid \mathcal{D}) &= \mathcal{N}(\theta \mid \mu_n, \Sigma_n) \\ \mu_n &= \Sigma_n(\Sigma_0^{-1}\mu_0 + (1/\sigma^2)\Phi^T Y) \\ \Sigma_N^{-1} &= \Sigma_0^{-1} + \frac{1}{\sigma^2}\Phi^T\Phi. \end{aligned}$$

Now the ridge regression solution is

$$\hat{\theta}_{ridge} = (\lambda I + \Phi^T\Phi)^{-1}\Phi^T Y.$$

If we make the substitution, $\mu_0 = 0$, $\Sigma_0 = I/\lambda$, $\sigma^2 = 1$, the posterior mean becomes

$$\mu_n = (\lambda I + \Phi^T\Phi)^{-1}\Phi^T Y = \hat{\theta}_{ridge}$$

So using the observation from the problem, we note that μ_n is also the mode of the posterior distribution under $\mu_0 = 0, \Sigma_0 = I/\lambda$ and $\sigma^2 = 1$ (this last assumption can be made without loss of generality, since given any noise σ, we can divide the outcomes y by σ to obtain a regression problem where $\sigma^2 = 1$).

Thus returning the mean of the posterior, taking the MAP estimate, and applying ridge regression are all equivalent (up to re-parametrizations).

7 Posterior predictive distribution

We know this went by in lecture, but it was probably kind of fast. Derive the formula for the posterior predictive distribution for linear regression with a Gaussian prior on the weights.

Solution: See Bishop, 3.3.2.

8 Monotony

In the basic Gaussian parameter-estimation case (not regression), we find that after getting one data sample, the new precision matrix is the sum of the old precision matrix and the precision of the observation:

$$\Sigma_1^{-1} = \Sigma_0^{-1} + \Sigma^{-1}$$

where Σ_0 is the prior covariance on the mean of the distribution and Σ is the known covariance.

In lecture we said informally that "the variance decreases monotonically" as a the amount of data increases. We'll try to make it more formal and call it a theorem. Here are some possible theorems, some harder to prove than others. Try to prove them. Don't worry if you don't get them all. You'll probably find the *Matrix Cookbook*[2] handy.

(a) **Theorem 1** The trace of Σ_n^{-1} increases monotonically as n increases.

Solution: This follows directly from $\mathrm{Tr}(A + B) = \mathrm{Tr}(A) + \mathrm{Tr}(B)$.

(b) **Theorem 2** The minimum eigenvalue of Σ_n decreases monotonically as n increases.

Solution: Note that the minimum eigenvalue of Σ_n monotonically decreases iff the maximum eigenvalue of ${\Sigma_n}^{-1}$ increases. The maximum eigenvalue of matrix A is equal to

$$\max_{\|x\|=1} x^T A x.$$

Therefore, we only need to prove:

$$\max_{\|x\|=1} x^T \Sigma_1^{-1} x > \max_{\|x\|=1} x^T \Sigma_0^{-1} x.$$

As Σ is positive definite, Σ^{-1} is as well, and so we have $x^T \Sigma^{-1} x \geqslant 0$ for all x. Hence,

$$\max_{\|x\|=1} x^T \Sigma_1^{-1} x = \max_{\|x\|=1} \left(x^T \Sigma_0^{-1} x + x^T \Sigma^{-1} x \right) \geqslant \max_{\|x\|=1} x^T \Sigma_0^{-1} x.$$

(c) **Theorem 3** The trace of Σ_n decreases monotonically as n increases.

You can use a special case of the Woodbury identity: for invertible matrices of the correct sizes,

$$(A + B)^{-1} = A^{-1} - A^{-1}(A^{-1} + B^{-1})^{-1} A^{-1}.$$

[2] http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=3274

Solution: Applying the Woodbury identity, we have:

$$\Sigma_1 = (\Sigma_0^{-1} + \Sigma^{-1})^{-1} = \Sigma_0 - \Sigma_0(\Sigma + \Sigma_0)^{-1}\Sigma_0.$$

As both Σ and Σ_0 are positive semidefinite, $(\Sigma + \Sigma_0)^{-1}$ is also positive semidefinite. We then have that for any vector v,

$$v^\top \Sigma_0(\Sigma + \Sigma_0)^{-1}\Sigma_0 v \geqslant 0$$

by setting $v' = \Sigma_0 v$ and using the characterization of positive semidefinite matrices for $(\Sigma + \Sigma_0)^{-1}$. Hence, $\Sigma_0(\Sigma + \Sigma_0)^{-1}\Sigma_0$ is also positive semi-definite.

It then follows that $\mathrm{Tr}(\Sigma_0(\Sigma + \Sigma_0)^{-1}\Sigma_0) \geqslant 0$, and so finally

$$\mathrm{Tr}(\Sigma_1) = \mathrm{Tr}(\Sigma_0) - \mathrm{Tr}(\Sigma_0(\Sigma + \Sigma_0)^{-1}\Sigma_0) \leqslant \mathrm{Tr}(\Sigma_0).$$

9 Gaussian decision boundaries **

1. Murphy 4.21

Let $p(x|y = j) = \mathcal{N}(x|\mu_j, \sigma_j)$ where $j = 1, 2$ and $\mu_1 = 0, \sigma_1^2 = 1, \mu_2 = 1, \sigma_2^2 = 10^6$.
Let the class priors be equal, $p(y = 1) = p(y = 2) = 0.5$

(a) Find the decision region

$$R_1 = \{x : p(x|\mu_1, \sigma_1) \geqslant p(x|\mu_2, \sigma_2)\}$$

Sketch the result. Hint: draw the curves and find where they intersect by solving $p(x|\mu_1, \sigma_1) = p(x|\mu_2, \sigma_2)$.

(b) Now suppose $\sigma_2 = 1$ (and all other parameters remain the same). what is R_1 in this case?

Solution:

(a) We solve

$$\frac{1}{\sqrt{2\pi}\sigma_1}\exp\left(-\frac{1}{2\sigma_1^2}(x-\mu_1)^2\right) = \frac{1}{\sqrt{2\pi}\sigma_2}\exp\left(-\frac{1}{2\sigma_2^2}(x-\mu_2)^2\right)$$

Taking logs and cancelling common terms, we get

$$-\log\sigma_1 - \frac{1}{2\sigma_1^2}(x-\mu_1)^2 = -\log\sigma_2 - \frac{1}{2\sigma_2^2}(x-\mu_2)^2$$

To simplify the algebra we substitute $\mu_1 = 0, {\sigma_1}^2 = 1, \mu_2 = 1$:

$$-\log\sigma_2 - \frac{1}{2\sigma_2^2}(x-1)^2 = -\frac{1}{2}x^2$$

$$-x^2 + \frac{1}{\sigma_2^2}x^2 - \frac{2x}{\sigma_2^2} + \frac{1}{\sigma_2^2} + 2\log\sigma_2 = 0$$

Now let

$$a = \frac{1}{\sigma_2^2} - 1 \qquad b = -\frac{2}{\sigma_2^2} \qquad c = \frac{1}{\sigma_2^2} + 2\log\sigma_2$$

So $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a} = \pm 3.7169$.

The decision region is sketch in Figure below. Note that R_2 is discontinuous. Here is a way to solve this problem using Matlab's symbolic algebra toolbox.

```
syms x
mu1 = 0;
sigma1=1;
mu2=1;
sigma2=1000;
double(solve(normpdf(x,mu1, sigma1)-normpdf(x,mu2, sigma2)))

ans =
     -3.7169
      3.7169
```

(b) If $\sigma_2 = 1 = \sigma_1$, we know that the decision boundary is equidistant between the means:

$$x^* = (\mu_1 + \mu_2)/2 = 0.5$$

Hence $R_1 = \{x : x \leqslant x^*\}$, i.e., all points to the left of 0.5. Similarly $R_2 = \{x : x \geqslant x^*\}$, i.e., all points to the right of 0.5. In this case, R_2 is a single connected region.

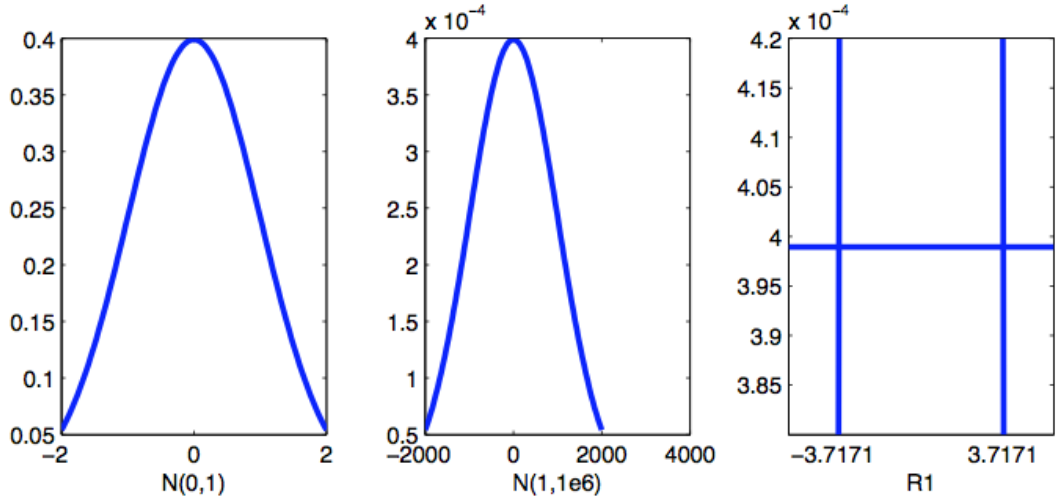


Figure: Class conditional densities $p(x|y = 1) = \mathcal{N}(0, 1)$ and $p(x|y = 2) = \mathcal{N}(1, 10^6)$ and the corresponding decision region for class 1.

2. Decision boundary types

Use these functions to investigate different possible Gaussian decision boundaries. For each of the following specifications, find an appropriate pair of Gaussians and prior class probabilities.

(a) A linear decision boundary.

(b) A linear decision boundary, with both means on the same side of the decision boundary.

(c) A parabolic decision boundary.

(d) A non-continuous decision boundary (one class represented by 2 disconnected regions).

(e) A circular decision boundary.

(f) A skewed ellipsoid decision boundary, with only one mean inside the ellipsoid.

(g) No decision boundary; the entire plane is one decision region.

Solution: Some examples are given below in Figure 9, with the specifications given in the captions.

3. In many cases, it is necessary to classify into more than two classes. A natural extension of the Gaussian mixture approach is to fit a Gaussian distribution for each class, and classify each input vector to the class with the highest posterior probability for it.

   We would like to modify the function `plot_two_gauss2d` to plot the decision boundaries between three Gaussians. Write a new `plot3gaussians` procedure and use it to plot decision boundaries on several examples.

   (a) All decision boundaries are linear.

   (b) Some decision boundaries are linear, while others are quadratic.

   (c) All decision boundaries are quadratic.

   (d) There are only two decision regions (one class never gets selected).

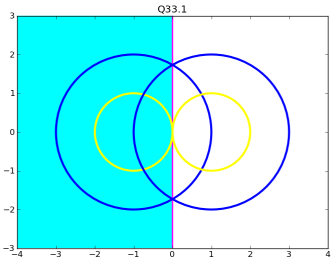
   ALSO: mark each decision region in the plots with an appropriate label.

Solution:

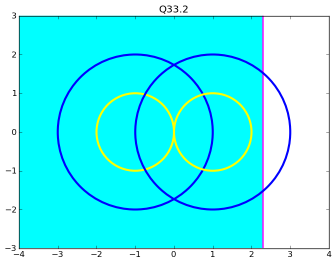
Some examples are given below in Figure 2, with the specifications given in the captions.

9.1 Python Code for plot3gaussians

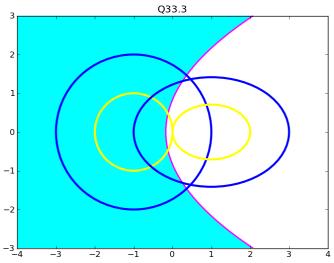
```
def plot3(mu1, C1, mu2, C2, mu3, C3, pri=array([1./3,1./3,1./3])):
    mu1 = mu1.reshape(-1,1)
    mu2 = mu2.reshape(-1,1)
    mu3 = mu3.reshape(-1,1)

    # Plot ellipses
    plotGauss2D(mu1, C1, 1, color1)
    plotGauss2D(mu1, C1, 2, color2)
    plotGauss2D(mu2, C2, 1, color1)
    plotGauss2D(mu2, C2, 2, color2)
    plotGauss2D(mu3, C3, 1, color1)
```



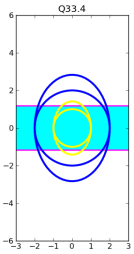
(a) $\mu_1 = [-1, 0], \Sigma_1 = \mathrm{I}$, $\mu_2 = [1, 0], \Sigma_2 = \mathrm{I}, \pi = [\frac{1}{2}, \frac{1}{2}]$



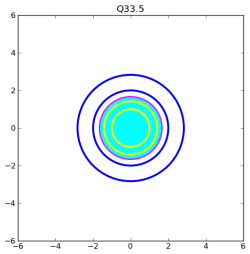
(b) $\mu_1 = [-1, 0], \Sigma_1 = \mathrm{I}$, $\mu_2 = [1, 0], \Sigma_2 = \mathrm{I}, \pi = [.99, .01]$



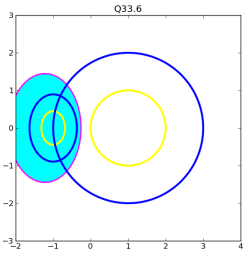
(c) $\mu_1 = [-1, 0], \Sigma_1 = \mathrm{I}$, $\mu_2 = [1, 0], \Sigma_2 = \texttt{diag}([1, .5]), \pi = [\frac{1}{2}, \frac{1}{2}]$



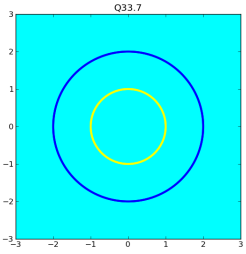
(d) $\mu_1 = [0, 0], \Sigma_1 = \mathrm{I}$, $\mu_2 = [0, 0], \Sigma_2 = \texttt{diag}([1, 2])$, $\pi = [\frac{1}{2}, \frac{1}{2}]$



(e) $\mu_1 = [0, 0], \Sigma_1 = \mathrm{I}$, $\mu_2 = [0, 0], \Sigma_2 = 2\mathrm{I}$, $\pi = [\frac{1}{2}, \frac{1}{2}]$



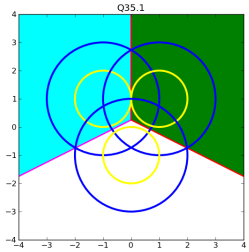
(f) $\mu_1 = [-1, 0], \Sigma_1 = \texttt{diag}([.1, .2])$, $\mu_2 = [1, 0], \Sigma_2 = \mathrm{I}$, $\pi = [\frac{1}{2}, \frac{1}{2}]$



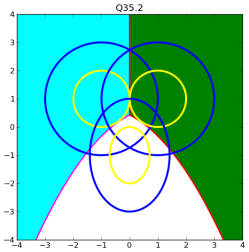
(g) $\mu_1 = [0, 0], \Sigma_1 = \mathrm{I}$, $\mu_2 = [0, 0], \Sigma_2 = \mathrm{I}$, $\pi = [.99, .01]$

Figure 2: Plots for Gaussian Decision Boundaries. Cyan = class 1, white = class 2.

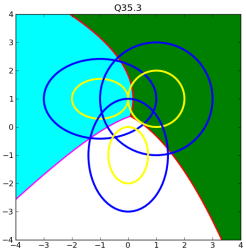
```
plotGauss2D(mu3, C3, 2, color2)

# Find axis limits
rmax = 3
xmin = min(mu1[0] - C1[0, 0] * rmax, mu2[0] - C2[0, 0] * rmax, mu3[0] - C3[0, 0] * rmax)
xmax = max(mu1[0] + C1[0, 0] * rmax, mu2[0] + C2[0, 0] * rmax, mu3[0] + C3[0, 0] * rmax)
ymin = min(mu1[1] - C1[1, 1] * rmax, mu2[1] - C2[1, 1] * rmax, mu3[1] - C3[1, 1] * rmax)
ymax = max(mu1[1] + C1[1, 1] * rmax, mu2[1] + C2[1, 1] * rmax, mu3[1] + C3[1, 1] * rmax)

# Evaluate log pdf on grid
ns = 500
```



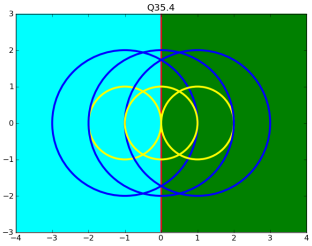
(a) $\mu_1 = [-1, 1], \Sigma_1 = \mathrm{I}$,
$\mu_2 = [1, 1], \Sigma_2 = \mathrm{I}$,
$\mu_3 = [0, -1], \Sigma_3 = \mathrm{I}$,
$\pi = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$



(b) $\mu_1 = [-1, 1], \Sigma_1 = \mathrm{I}$,
$\mu_2 = [1, 1], \Sigma_2 = \mathrm{I}$,
$\mu_3 = [0, -1], \Sigma_3 = \mathtt{diag}([.5, 1])$,
$\pi = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$



(c) $\mu_1 = [-1, 1], \Sigma_1 = \mathtt{diag}([1, .5])$,
$\mu_2 = [1, 1], \Sigma_2 = \mathrm{I}$,
$\mu_3 = [0, -1], \Sigma_3 = \mathtt{diag}([.5, 1])$,
$\pi = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$



(d) $\mu_1 = [-1, 0], \Sigma_1 = \mathrm{I}$,
$\mu_2 = [1, 0], \Sigma_2 = \mathrm{I}$,
$\mu_3 = [0, 0], \Sigma_3 = \mathrm{I}$,
$\pi = [.495, .495, .01]$

Figure 3: Plots for 3 Gaussians. Cyan = class 1, green = class 2, white = class 3.

```
xx,yy = meshgrid(linspace(xmin, xmax, ns), linspace(ymin, ymax, ns))
Xs = r_[xx.ravel().reshape(1, -1), yy.ravel().reshape(1, -1)]
V1s = gauss_logpdf(mu1, C1, Xs) + log(pri[0])
V2s = gauss_logpdf(mu2, C2, Xs) + log(pri[1])
V3s = gauss_logpdf(mu3, C3, Xs) + log(pri[2])

# Decision region is where value is greater than maximum of others (diff > 0)
v1max = V1s - maximum(V2s,V3s)
v2max = V2s - maximum(V1s,V3s)
v3max = V3s - maximum(V1s,V2s)

# Plot decision regions and boundaries
cs1 = contourf(xx, yy, v1max.reshape(ns,ns), [0,inf], colors='cyan')
cs2 = contourf(xx, yy, v2max.reshape(ns,ns), [0,inf], colors='green')
cs3 = contourf(xx, yy, v3max.reshape(ns,ns), [0,inf], colors='white')
contour(cs1, [0,0], colors='magenta', linestyles='solid', linewidths=2)
contour(cs2, [0,0], colors='red', linestyles='solid', linewidths=2)

axis([xmin, xmax, ymin, ymax])
axes().set_aspect('equal')
```

9.2 MATLAB Code for plot3gaussians

```
function plot_three_gauss2d(mu1, C1, mu2, C2, mu3, C3, pri)

if nargin < 7
    pri = [1/3 1/3 1/3];
end

% Find and set axis limits
rmax = 3;
xmin = min([mu1(1) - C1(1,1) * rmax, mu2(1) - C2(1,1) * rmax, mu3(1) - C3(1,1) * rmax]);
xmax = max([mu1(1) + C1(1,1) * rmax, mu2(1) + C2(1,1) * rmax, mu3(1) + C3(1,1) * rmax]);
ymin = min([mu1(2) - C1(2,2) * rmax, mu2(2) - C2(2,2) * rmax, mu3(1) - C3(2,2) * rmax]);
ymax = max([mu1(2) + C1(2,2) * rmax, mu2(2) + C2(2,2) * rmax, mu3(1) + C3(2,2) * rmax]);
axis equal;
axis([xmin xmax ymin ymax]);

% Evaluate log pdf on grid
ns = 500;
[xx, yy] = meshgrid(linspace(xmin, xmax, ns), linspace(ymin, ymax, ns));
Xs = [xx(:) yy(:)]';
V1s = gauss_logpdf(mu1, C1, Xs) + log(pri(1));
V2s = gauss_logpdf(mu2, C2, Xs) + log(pri(2));
V3s = gauss_logpdf(mu3, C3, Xs) + log(pri(3));

% Decision region is where value is greater than maximum of others (diff > 0)
v1max = V1s - max(V2s,V3s);
v2max = V2s - max(V1s,V3s);
v3max = V3s - max(V1s,V2s);

% Plot decision region and boundaries
hold on;
contour(xx, yy, reshape(v1max,ns,ns), [0,inf], '-m', 'Linewidth', 5);
contour(xx, yy, reshape(v2max,ns,ns), [0,inf], '-m', 'Linewidth', 5);
contour(xx, yy, reshape(v3max,ns,ns), [0,inf], '-m', 'Linewidth', 5);

% Plot ellipses
plot_gauss2d(mu1, C1, 1, 'y');
plot_gauss2d(mu1, C1, 2, 'b');
plot_gauss2d(mu2, C2, 1, 'y');
plot_gauss2d(mu2, C2, 2, 'b');
plot_gauss2d(mu3, C3, 1, 'y');
plot_gauss2d(mu3, C3, 2, 'b');
```

10 QDA with 3 classes **

Consider a three category classification problem. Let the prior probabilities be

$$P(Y=1) = P(Y=2) = P(Y=3) = \frac{1}{3}.$$

The class-conditional densities are multivariate normal densities with parameters

$$\mu_1 = [0,0]^T, \qquad \mu_2 = [1,1]^T, \qquad \mu_3 = [-1,1]^T$$

$$\Sigma_1 = \begin{bmatrix} 0.7 & 0 \\ 0 & 0.7 \end{bmatrix}, \qquad \Sigma_2 = \begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}, \qquad \Sigma_3 = \begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}$$

Classify the following points:

(a) x = [-0.5, 0.5]

(b) x = [0.5, 0.5]

Solution: We just need to pick the MAP class for each point. Here is some matlab code to compute the class posteriors.

```
mu1=[0;0];mu2=[1;1];mu3=[-1;1];
params.mu=[mu1 mu2 mu3];
params.Sigma(:,:,1)=[0.7 0;0 0.7];
params.Sigma(:,:,2)=[0.8 0.2;0.2 0.8];
params.Sigma(:,:,3)=[0.8 0.2;0.2 0.8];
params.Classprior=[1/3 1/3 1/3];
X1=[-0.5 0.5];
X2=[0.5 0.5];
post1=classify3gauss(X1,params)
post2=classify3gauss(X2,params)

function post=classify3gauss(Xtest,params)
Nclasses=length(params.Classprior)
for c=1:Nclasses
  lik(:,c) = mvnpdf(Xtest, params.mu(:, c)', params.Sigma(:,:, c));
end
classPrior = params.Classprior;
N = size(Xtest,1);
logjoint = log(lik) + repmat(log(classPrior(:)'), N, 1);
logpost = logjoint - repmat(logsumexp(logjoint,2), 1, Nclasses);
post = exp(logpost);
```

(a) You get the following results:

$$P(Y = 1 \mid \mathbf{x_1}) = 0.46, \qquad P(Y = 2 \mid \mathbf{x_1}) = 0.145, \qquad P(Y = 3 \mid \mathbf{x_1}) = 0.39$$

So class 1 has maximum posterior probability.

(b) For the second vector:

$$P(Y = 1 \mid \mathbf{x_2}) = 0.45, \qquad P(Y = 2 \mid \mathbf{x_2}) = 0.46, \qquad P(Y = 3 \mid \mathbf{x_2}) = 0.09$$

Choose class 2. (Although class 1 is very close: we might prefer the "reject" option if it was available, in this case.)

11 Naive Bayes with Mixed Features **

Consider a three-class naive Bayes classifier with one binary feature, one Gaussian feature and multinomial output:

$$y \sim \text{Mu}(y \mid \pi, 1), \qquad x_1 \mid y = c \sim \text{Ber}(x_1 \mid \theta_c), \qquad x_2 \mid y = x \sim \mathcal{N}(x_2 \mid \mu_c, \sigma_c^2)$$

where the subscript c denotes the class.

Let the parameter vectors be as follow:

$$\pi = (0.5, 0.25, 0.25), \qquad \theta = (0.5, 0.5, 0.5), \qquad \mu = (-1, 0, 1), \qquad \sigma^2 = (1, 1, 1)$$

(a) Compute $p(y \mid x_1 = 0, x_2 = 0)$ (the result should be a vector of 3 numbers that sums to 1).

(b) Compute $p(y \mid x_2 = 0)$.

(c) Compute $p(y \mid x_1 = 0)$.

(d) Explain any interesting patterns you see in your results.

Solution:

(a) We have $p(y = c, x_1, x_2) = \pi(c)\text{Ber}(x_1 \mid \theta_c)\mathcal{N}(x_2 \mid \mu_c, {\sigma_c}^2)$, where

$$\text{Ber}(x_1 \mid \theta_c) = {\theta_c}^{I(x_1=1)}(1-\theta_c)^{I(x_1=0)} = 0.5^{I(x_1=1)}0.5^{I(x_1=0)} = 0.5$$

Thus feature x_1 is irrelevant: $p(y \mid x_1, x_2) = p(y \mid x_2)$ and $p(y \mid x_1) = p(y)$.

Then,

$$\begin{aligned} p(y = c, x_2 = 0) &= \pi(c)\frac{1}{\sqrt{2\pi}\sigma_c}\exp(-\frac{1}{2{\sigma_c}^2}(x_2 - \mu_c)^2) \\ &= \pi(c)\frac{1}{\sqrt{2\pi}}\exp(-\frac{1}{2}{\mu_c}^2) \\ &\propto \pi(c)\exp(-\frac{1}{2}{\mu_c}^2) \end{aligned}$$

We get vector

$$\begin{aligned} p(y, x_2 = 0) &= [0.5, 0.25, 0.25] .* \exp([-0.5, 0, -0.5]) \\ &= [0.3033, 0.2500, 0.1516] \end{aligned}$$

Marginalizing out y, we get $p(x_2 = 0) = \sum_y p(y, x_2 = 0) = .7049$. So,

$$p(y \mid x_2 = 0) = \frac{p(y, x_2 = 0)}{p(x_2 = 0)} = [0.3033, 0.2500, 0.1516]./0.7049 = [0.4302, 0.3547, 0.2151]$$

(b) Because x_1 is uninformative, we know $p(y \mid x_2 = 0) = p(y \mid x_1 = 0, x_2 = 0)$.

(c) We also find $p(y \mid x_1) = p(y)$ for the same reason.

Below is some Matlab code to compute this, in case you are in any doubt. But as you can see from the above, you don't need Matlab to solve this simple problem!

```
prior = [0.5 0.25 0.25];
theta = [0.5 0.5 0.5];
mu = [-1 0 1];
sigma = [1 1 1];
x1 = 0; x2 = 0;

for c=1:3
  if x1==1
    lik1(c) = theta(c);
  else
    lik1(c) = 1-theta(c);
end
  lik2(c) = gausspdf(x2, mu(c), sigma(c)ËĘ2);
end
post12 = normalize(lik1 .* lik2 .* prior)
assert(approxeq(
   post12, normalize(
   [0.5 0.25 0.25] .* [0.5 0.5 0.5] .* exp([-0.5 0 -0.5]))))
post2 = normalize(lik2  .* prior)
post1 = normalize(lik1  .* prior)

The output is as follows:
post12 =
    0.4302    0.3547 0.2151
post1 =
    0.5000    0.2500 0.2500
post2 =
    0.4302    0.3547 0.2151
```

12 Probabilistic Models for Classification *

In this question we'll consider feature selection in a naive Bayes model. As in the lecture, we'll assume that the training set consists of examples $(\mathbf{x}^{(i)}, y^{(i)})$ where labels $y^{(i)} \in \{-1, +1\}$ and features $x_i \in \{0, 1\}^d$. In a regular naive Bayes model, we define

$$p(\mathbf{x}, y; \theta) = p(y) \prod_{j=1}^{d} p_j(x_j \mid y) \tag{12.1}$$

Now consider a naive Bayes model defined on a subset of all possible features. We use A to define the set of active features in the model. The set A is a subset of $\{1, 2, \ldots, d\}$. The model then takes the following form:

$$p(\mathbf{x}, y; \theta, A) = p(y) \prod_{j \in A} p_j(x_j \mid y) \prod_{j \notin A} p_j(x_j) \tag{12.2}$$

Note that for each active feature we have parameters of the form $p_j(x \mid y)$ which depend on the label y, and for inactive features we have parameters of the form $p_j(x)$ which ignore the label y.

The $p_j(x_j)$ terms are necessary to ensure that the model still correctly defines a distribution $p(\mathbf{x}, y)$ over all possible $\mathbf{x}, y$ pairs. It might be tempting to write the model as

$$p(\mathbf{x}, y; \theta, A) = p(y) \prod_{j \in A} p_j(x_j \mid y)$$

but under this definition the x_j terms for $j \notin A$ are not accounted for in the model, and the model is deficient (in fact, it can be shown that in this case $\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(\mathbf{x}, y; \theta, A) > 1$, assuming that $\mathcal{X} = \{0,1\}^d$ and $\mathcal{Y} = \{-1, +1\}$, clearly violating the laws of probability).

1. The classification function is

$$f(\mathbf{x}) = \arg\max_y p(\mathbf{x}, y; \theta, A) = \text{sign}[\log p(\mathbf{x}, +1; \theta, A) - \log p(\mathbf{x}, -1; \theta, A)]$$

Show that for any feature $j \notin A$, its value is irrelevant to the classification function. For example, for any feature vector $\mathbf{x}$, if we defined a new feature vector $\mathbf{x}'$ where $x'_j = x_j$ for $j \in A$, and $x'_j = 1 - x_j$ for $j \notin A$, then $f(\mathbf{x}) = f(\mathbf{x}')$.

Solution:

Writing out the probabilities in the classification function, we get:

$$\begin{aligned}
\Delta &:= \log p(\mathbf{x}, +1; \theta, A) - \log p(\mathbf{x}, -1; \theta, A) \\
&= \log p(y = +1) + \sum_{j \in A} \log p_j(x_j \mid y = +1) + \sum_{j \notin A} \log p_j(x_j) \\
&\quad - \log p(y = -1) - \log \sum_{j \in A} p_j(x_j \mid y = -1) - \log \sum_{j \notin A} p_j(x_j) \\
&= \log p(y = +1) + \log \sum_{j \in A} p_j(x_j \mid y = +1) - \log p(y = -1) - \log \sum_{j \in A} p_j(x_j \mid y = -1)
\end{aligned}$$

which we see does not depend on the values of the any features $j \notin A$.

2. Now let's consider a feature-selection method for these models. In this method, initially we start with A equal to the empty set; i.e., there are no active features. At each iteration we greedily choose a single feature which gives the most "improvement" in the model. We will define a precise measure of "improvement" below. Assume that maximum-likelihood estimates are used in the model:

$$\hat{p}_j(\mathbf{x} \mid y) = \frac{\sum_{i=1}^n \text{count}[x_j^{(i)} = x \text{ and } y^{(i)} = y]}{\sum_{i=1}^n \text{count}[y^{(i)} = y]}$$

and

$$\hat{p}_j(\mathbf{x}) = \frac{\sum_{i=1}^n \text{count}[x_j^{(i)} = x]}{n} \tag{12.3}$$

We measure the gain for any feature $k \notin A$ given a set of active features A as

$$\text{Gain}(k; A) = \sum_i \log p(\mathbf{x}^{(i)}, y^{(i)}; \theta', A \cup \{k\}) - \sum_i \log p(\mathbf{x}^{(i)}, y^{(i)}; \theta', A) \tag{12.4}$$

$\text{Gain}(k; A)$ simply measures how much adding the k^{th} feature improves the fit of the model to the training data, where "fit of the model" is measured as log-likelihood. At each point in the feature selection method, we choose the feature with the maximal gain. At some point, we stop adding features (for example, we might use cross-validation to choose the stopping point), so that the final model uses a relatively small subset of the features.

Show that

$$\text{Gain}(k; A) = n \sum_{y \in \{-1,+1\}} \sum_{x \in \{0,1\}} \hat{p}_k(x, y) \log \frac{\hat{p}(x \mid y)}{\hat{p}_k(x)} \tag{12.5}$$

where

$$\hat{p}_k(x, y) = \frac{1}{n} \sum_{i=1}^{n} \text{count}[x_k^{(i)} = x \text{ and } y^{(i)} = y] \tag{12.6}$$

$\text{Gain}(k; A)$ is an estimate, based on the training examples, of the mutual information between the feature x_k and the label y. Mutual information is a quantity that measures the dependence between two random variables (in this case X_k and Y), and that arises frequently in probability theory and information theory.

Solution:

$$\begin{aligned}
\text{Gain}(k; A) &= \sum_i \log p(\mathbf{x}^{(i)}, y^{(i)}; \theta', A \cup \{k\}) - \sum_i \log p(\mathbf{x}^{(i)}, y^{(i)}; \theta', A) \\
&= \sum_i \left[\log p(y^{(i)}) + \log \sum_{j \in A \cup \{k\}} p_j(x_j^{(i)} \mid y^{(i)}) + \log \sum_{j \notin A \cup \{k\}} p_j(x_j^{(i)}) \right] \\
&\quad - \sum_i \left[\log p(y^{(i)}) + \log \sum_{j \in A} p_j(x_j^{(i)} \mid y^{(i)}) + \log \sum_{j \notin A} p_j(x_j^{(i)}) \right] \\
&= \sum_i \log p_k(x_k^{(i)} | y^{(i)}) - \log p_k(x_k^{(i)}) = \sum_i \log \frac{p_k(x_k^{(i)} \mid y^{(i)})}{p_k(x_k^{(i)})}
\end{aligned}$$

where in the last step, we notice that the only difference between sums is due to feature k.

We can rewrite this in terms of the number of times $y = -1$ and $y = +1$:

$$\begin{aligned}
\text{Gain}(k; A) &= \sum_{x \in \{0,1\}} \sum_{i=1}^{n} \text{count}[x_k^{(i)} = x \text{ and } y^{(i)} = -1] \log \frac{p_k(x_k^{(i)} \mid y^{(i)} = -1)}{p_k(x_k^{(i)})} \\
&\quad + \sum_{x \in \{0,1\}} \sum_{i=1}^{n} \text{count}[x_k^{(i)} = x \text{ and } y^{(i)} = +1] \log \frac{p_k(x_k^{(i)} \mid y^{(i)} = +1)}{p_k(x_k^{(i)})} \\
&= \sum_{y \in \{-1,1\}} \sum_{x \in \{0,1\}} \sum_{i=1}^{n} \text{count}[x_k^{(i)} = x \text{ and } y^{(i)} = y] \log \frac{\hat{p}_k(x \mid y)}{\hat{p}_k(x)} \\
&= n \sum_{y \in \{-1,1\}} \sum_{x \in \{0,1\}} \hat{p}_k(x, y) \log \frac{\hat{p}_k(x \mid y)}{\hat{p}_k(x)}
\end{aligned}$$

13 Classification Questions from Bishop (note notational differences)

1. Bishop 4.9

   Consider a generative classification model for K classes defined by prior class probabilities $p(C_k) = \pi_k$ and general class-conditional densities $p(\phi|C_K)$ where ϕ is the input feature vector. Suppose we are given a training data set $\{\phi_n, t_n\}$ where $n = 1, \ldots, N$, and t_n is a binary target vector of length K that uses the 1-of-K coding scheme, so that it has components $t_{nj} = I_{jk}$ if pattern n is from class C_k. Assuming that the data points are drawn independently from this model, show that the maximum-likelihood solution for the prior probabilities is given by

   $$\pi_k = \frac{N_k}{N} \tag{13.1}$$

   where N_k is the number of data points assigned to class C_k.

   Solution:

   The likelihood function is given by

   $$p(\{\phi_n, \mathbf{t}_n\}|\{\pi_k\}) = \prod_{n=1}^{N}\prod_{k=1}^{K} (p(\phi_n|C_k)\pi_k)^{t_{nk}}$$

   and taking the logarithm, we obtain

   $$\ln p(\{\phi_n, \mathbf{t}_n\}|\{\pi_k\}) = \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \left(\ln p(\phi_n|C_k) + \ln \pi_k\right)$$

   To maximize the log likelihood with respect to π_k we need to maintain the constraint $\sum_k \pi_k = 1$. This can be done by introducing a Lagrange multiplier λ and maximizing

   $$\ln p(\{\phi_n, \mathbf{t}_n\}|\{\pi_k\}) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1\right)$$

   Setting the derivative with respect to π_k equal to zero, we obtain

   $$\sum_{n=1}^{N} \frac{t_{nk}}{\pi_k} + \lambda = 0$$

   Re-arranging then gives

   $$\lambda\pi_k = -\sum_{n=1}^{N} t_{nk} = -N_k$$

   Summing both sides over k we find that $\lambda = -N$, and using this to eliminate λ we obtain Eq. 13.1.

2. Bishop 4.10

Consider the classification model of Exercise 4.9 (above) and now suppose that the class-conditional densities are given by Gaussian distributions with a shared covariance matrix:

$$p(\phi|C_k) = \mathcal{N}(\phi|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}) \tag{13.2}$$

Show that the maximum likelihood solution for the mean of the Gaussian distribution for class C_k is given by

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk} \phi_n \tag{13.3}$$

which represents the mean of those feature vectors assigned to class C_k. Similarly, show that the maximum likelihood solution for the shared covariance matrix is given by

$$\boldsymbol{\Sigma} = \sum_{k=1}^{K} \frac{N_k}{N} \mathbf{S}_k \tag{13.4}$$

where

$$\mathbf{S}_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk} (\phi_n - \boldsymbol{\mu}_k)(\phi_n - \boldsymbol{\mu}_k)^T \tag{13.5}$$

Thus $\boldsymbol{\Sigma}$ is given by a weighted average of the covariances of the data associated with each class, in which the weighting coefficients are given by the prior probabilities of the classes.

Solution: We start with the log likelihood found in the previous problem

$$\ln p(\{\phi_n, \mathbf{t}_n\}|\{\pi_k\}) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \{\ln p(\phi_n|C_k) + \ln \pi_k\}$$

If we substitute Eq. 13.2 into this and use the definition of the multivariate Gaussian, we obtain

$$\ln p(\{\phi_n, \mathbf{t}_n\}|\{\pi_k\}) = -\frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \{\ln |\boldsymbol{\Sigma}| + (\phi_n - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1} (\phi - \boldsymbol{\mu}_k)\}$$

where we have dropped terms independent of $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}$.

Setting the derivative of the right hand side w.r.t $\boldsymbol{\mu}_k$, obtained by using the fact that $\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T \mathbf{a}) = \frac{\partial}{\partial \mathbf{x}}(\mathbf{a}^T \mathbf{x}) = \mathbf{a}$, to zero, we get

$$\sum_{n=1}^{N} t_{nk} \boldsymbol{\Sigma}^{-1} (\phi_n - \boldsymbol{\mu}_k) = 0$$

Making use of $\sum_{n=1}^{N} t_{nk} = N_k$, we can re-arrange this to obtain Eq. 13.3.

Rewriting the log likelihood as

$$\ln p(\{\phi_n, \mathbf{t}_n\}|\{\pi_k\}) = -\frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \{\ln |\boldsymbol{\Sigma}| + \mathrm{Tr}[\boldsymbol{\Sigma}^{-1} (\phi_n - \boldsymbol{\mu}_k)(\phi_n - \boldsymbol{\mu}_k)^T]\}$$

where Tr represents the trace, we can use $\frac{\partial}{\partial A}\mathrm{Tr}(AB) = B^T$ and $\frac{\partial}{\partial A}\ln|A| = \left(A^{-1}\right)^T$ to calculate the derivative w.r.t. $\mathbf{\Sigma}^{-1}$. Setting this to zero we obtain

$$\frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\left[\mathbf{\Sigma} - (\phi_n - \boldsymbol{\mu}_k)(\phi_n - \boldsymbol{\mu}_k)^T\right] = 0$$

Again, as $\sum_{n=1}^{N} t_{nk} = N_k$, we can re-arrange this to obtain Eq. 13.4, with $\mathbf{S}_k$ given by Eq. 13.5.

We do not enforce that $\mathbf{\Sigma}$ should be symmetric; the solution automatically is.

3. Bishop 4.11 - Naïve Bayes

Consider a classification problem with K classes for which the feature vector ϕ has M components each of which can take L discrete states. Let the values of the components be represented by a 1-of-L binary coding scheme. Further suppose that, conditioned on the class C_k, the M components of ϕ are independent, so that the class-conditional density factorizes with respect to the feature vector components. Show that the quantities a_k given by

$$a_k = \ln(p(\mathbf{x}|C_k)p(C_k)) \tag{13.6}$$

which appear in the argument to the softmax function describing the posterior class probabilities, are linear functions of the components of ϕ. Note that this represents an example of the naive Bayes model which is discussed in Bishop.

Solution:

The generative model for ϕ corresponding to the chosen coding scheme is given by

$$p(\phi \mid C_k) = \prod_{m=1}^{M} p(\phi_m|C_k) \qquad \text{with} \qquad p(\phi_m \mid C_k) = \prod_{l=1}^{L} \mu_{kml}^{\phi_{ml}}$$

and where in turn the $\{\mu_{kml}\}$ are the parameters of the multinomial models for ϕ.

Substituting this into Eq. 13.6 we see that

$$\begin{aligned} a_k &= \ln p(\phi \mid C_k)p(C_k) \\ &= \ln p(C_k) + \sum_{m=1}^{M} \ln p(\phi_m \mid C_k) \\ &= \ln p(C_k) + \sum_{m=1}^{M}\sum_{l=1}^{L} \phi_{ml} \ln \mu_{kml} \end{aligned}$$

which is linear in ϕ_{ml}.