Exercises
Machine Learning
HS 2016

Series 3, Nov 18, 2016 (Numerical Estimation Methods)

Machine Learning Laboratory
Dept. of Computer Science, ETH Zürich
Prof. Joachim M. Buhmann
Web https://ml2.inf.ethz.ch/courses/ml/

Email questions 1, 2, 3 to: **Andreas Hess** ahess@student.ethz.ch
Email questions 4, 5, 6 to: **Milica Petrovic** milicap@student.ethz.ch

Problem 1 (Efficient Leave-One-Out Cross Validation):

In this question, we propose a computationally efficient method to compute the error of leave-one-out cross-validation (LOOCV). Given the output vector $\boldsymbol{y} \in \mathbb{R}^n$ and the input matrix $\mathcal{X} \in \mathbb{R}^{d\times n}$, whose n columns correspond to the n data points $\boldsymbol{x}_i \in \mathbb{R}^d$, the objective of ridge regression is defined as:

$$f_\mu(\boldsymbol{w}) = \frac{1}{n}\|\mathcal{X}^T\boldsymbol{w} - \boldsymbol{y}\|^2 + \frac{\mu}{2}\|\boldsymbol{w}\|^2 \tag{1}$$

Let $\mathcal{X}_{(-i)} \in \mathbb{R}^{d\times(n-1)}$ denote the data matrix that is obtained by excluding column i form the full data matrix $\mathcal{X}$. To compute the LOOCV error, one has to compute the minimizer $\boldsymbol{w}^*_{(-i)}$, which is defined as

$$\boldsymbol{w}^*_{(-i)} = \arg\min_{\boldsymbol{w}} \left(\frac{1}{n-1}\|\mathcal{X}^T_{(-i)}\boldsymbol{w} - \boldsymbol{y}_{(-i)}\|^2 + \frac{\mu}{2}\|\boldsymbol{w}\|^2\right) \tag{2}$$

for each $i \in \{1,..,n\}$ (remember: we have n folds and n hold-out datasets in LOOCV).
Then the cross-validation error is

$$\epsilon = \frac{1}{n}\sum_i^n \left(\langle \boldsymbol{x}_i, \boldsymbol{w}^*_{(-i)}\rangle - y_i\right)^2 \tag{3}$$

We want to prove that this error can be efficiently computed as

$$\epsilon = \frac{1}{n}\sum_i^n \left(\frac{y_i - \hat{y}_i}{1 - s_i}\right)^2 \tag{4}$$

where

$$\mathcal{A} = \left(\mathcal{X}\mathcal{X}^T + \frac{(n-1)\mu}{2}\mathcal{I}\right) \tag{5}$$
$$\hat{y}_i = \boldsymbol{x}_i^T\mathcal{A}^{-1}\mathcal{X}\boldsymbol{y} \tag{6}$$
$$s_i = \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i \tag{7}$$

To prove the above result, we follow three steps:

a. Prove that

$$\boldsymbol{w}^*_{(-i)} = \mathcal{A}^{-1}\mathcal{X}\boldsymbol{y} - \mathcal{A}^{-1}\boldsymbol{x}_i\left(\frac{1}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}\right)y_i + \frac{\mathcal{A}^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T\mathcal{A}^{-1}(\mathcal{X}\boldsymbol{y})}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i} \tag{8}$$

Hint: Use Sherman-Morrison formula, i.e.

$$(\mathcal{M} - \boldsymbol{u}\boldsymbol{v}^T)^{-1} = \mathcal{M}^{-1} + \frac{\mathcal{M}^{-1}\boldsymbol{u}\boldsymbol{v}^T\mathcal{M}^{-1}}{1 - \boldsymbol{v}^T\mathcal{M}^{-1}\boldsymbol{u}} \tag{9}$$

where $\mathcal{M}$ is an invertible square matrix and $\boldsymbol{u}$, $\boldsymbol{v}$ are column vectors.

b. Derive that

$$\boldsymbol{x}_i^T \boldsymbol{w}_{(-i)}^* = \left(\frac{1}{1-s_i}\right)(\hat{y}_i - s_i y_i) \tag{10}$$

c. Show that

$$y_i - \boldsymbol{x}_i^T \boldsymbol{w}_{(-i)}^* = \left(\frac{1}{1-s_i}\right)(y_i - \hat{y}_i) \tag{11}$$

Solution 1 (Efficient Leave-One-Out Cross Validation):

The derivations includes three steps:

Step 1.

$$\begin{aligned}
\boldsymbol{w}_{(-i)}^* &= \left(\mathcal{X}_{(-i)}\mathcal{X}_{(-i)}^T + \frac{(n-1)\mu}{2}\mathcal{I}\right)^{-1} \mathcal{X}_{(-i)}\boldsymbol{y}_{(-i)} && (12)\\
&= \left(\mathcal{A} - \boldsymbol{x}_i\boldsymbol{x}_i^T\right)^{-1}(\mathcal{X}\boldsymbol{y} - \boldsymbol{x}_i y_i) && (13)\\
&\overset{\text{Eq. (9)}}{=} \mathcal{A}^{-1}(\mathcal{X}\boldsymbol{y} - \boldsymbol{x}_i y_i) + \frac{\mathcal{A}^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T\mathcal{A}^{-1}(\mathcal{X}\boldsymbol{y} - \boldsymbol{x}_i y_i)}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i} && (14)\\
&= \mathcal{A}^{-1}\mathcal{X}\boldsymbol{y} - \mathcal{A}^{-1}\boldsymbol{x}_i\left(1 + \frac{x_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}\right)y_i + \frac{\mathcal{A}^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T\mathcal{A}^{-1}(\mathcal{X}\boldsymbol{y})}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i} && (15)\\
&= \mathcal{A}^{-1}\mathcal{X}\boldsymbol{y} - \mathcal{A}^{-1}\boldsymbol{x}_i\left(\frac{1}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}\right)y_i + \frac{\mathcal{A}^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T\mathcal{A}^{-1}(\mathcal{X}\boldsymbol{y})}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i} && (16)
\end{aligned}$$

Step 2.

$$\begin{aligned}
\boldsymbol{x}_i^T\boldsymbol{w}_{(-i)}^* &\overset{\text{Eq. (16)}}{=} \boldsymbol{x}_i^T\mathcal{A}^{-1}\mathcal{X}\boldsymbol{y} + \frac{\boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T\mathcal{A}^{-1}(\mathcal{X}\boldsymbol{y})}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i} - \left(\frac{1}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}\right)\boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i y_i && (17)\\
&= \left(1 + \frac{\boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}\right)\boldsymbol{x}_i^T\mathcal{A}^{-1}\mathcal{X}\boldsymbol{y} - \left(\frac{1}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}\right)\boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i y_i && (18)\\
&= \left(\frac{1}{1 - \boldsymbol{x}_i^T\mathcal{A}^{-1}\boldsymbol{x}_i}\right)\boldsymbol{x}_i^T\mathcal{A}^{-1}(\mathcal{X}\boldsymbol{y} - \boldsymbol{x}_i y_i) && (19)\\
&= \left(\frac{1}{1-s_i}\right)(\hat{y}_i - s_i y_i) && (20)
\end{aligned}$$

Step 3.

$$\begin{aligned}
y_i - \boldsymbol{x}_i^T\boldsymbol{w}_{(-i)}^* &\overset{\text{Eq. (20)}}{=} y_i - \left(\frac{1}{1-s_i}\right)(\hat{y}_i - s_i y_i) && (21)\\
&= \left(1 + \frac{s_i}{1-s_i}\right)y_i - \left(\frac{1}{1-s_i}\right)\hat{y}_i && (22)\\
&= \left(\frac{1}{1-s_i}\right)(y_i - \hat{y}_i) && (23)
\end{aligned}$$

Problem 2 (Bootstrap):

In this question, we analyse the behaviour of a bootstrap variant, where the size of the bootstrap datasets is not assumed to be equal to the size of the original dataset. Assume we have a dataset $\mathcal{D}$, consisting of n data points (x_i, y_i).

a. Let $\mathcal{D}_c^{*1}$ be a bootstrap dataset we constructed from $\mathcal{D}$ by drawing $c \cdot n$ datapoint at random (with replacement), $c \in (0, 1]$. What is the expected fraction $u_{n,c}$ of data points that are present in $\mathcal{D}$ but not in $\mathcal{D}_c^{*1}$, i.e.,

$$u_{n,c} := \mathbf{E}\left[\frac{|\mathcal{D} - \mathcal{D}_c^{*1}|}{n}\right] \tag{24}$$

Derive an expression for $u_{n,c}$ that only depends on n and c.

b. The size n of a dataset is generally large and we therefore assume that $u_c := \lim_{n\to\infty} u_{n,c}$ is a good approximation of $u_{n,c}$. Derive a simplified expression for u_c and compute its value for $c = 1$.
Hints:

$$\lim_{n\to\infty}\left(1 - \frac{1}{n}\right)^n = \lim_{n\to\infty}\left(\frac{1}{1 + \frac{1}{n}}\right)^n \tag{25}$$

and

$$\lim_{n\to\infty}\left(f(n)^{-1}\right) = \left(\lim_{n\to\infty} f(n)\right)^{-1} \tag{26}$$

for some function f of n and $\lim_{n\to\infty} f(n) \neq 0$

c. We now consider a binary classification task, i.e., let the x_i denote some feature vector and the y_i a binary class label (e.g. 1 or 0). The labels of the data points are random, i.e., assume that there are equally many data points with labels 1 and 0 and that the labels are independent of the features of the data points. Assume we train a binary classifier $\hat{f}_c^{*1}(x)$ on the bootstrap dataset $\mathcal{D}_c^{*1}$, and that the training error is equal to 0. How do we have to choose c, such that the test error we compute on the dataset $\mathcal{D}$ is exactly half as large as the true prediction error (i.e., the prediction error on the true data source)?

Solution 2 (Bootstrap):

a. Remember that we draw $c \cdot n$ samples with replacement from $\mathcal{D}$. The probability of a data point not to be picked in the i-th trial is

$$1 - \frac{1}{n} \tag{27}$$

Hence, the probability of that data point never to be picked during $c \cdot n$ samples is

$$u_{n,c} = \left(1 - \frac{1}{n}\right)^{c\cdot n} = \left(\left(1 - \frac{1}{n}\right)^n\right)^c \tag{28}$$

b. Note

$$e = \lim_{n\to\infty}\left(1 + \frac{1}{n}\right)^n \tag{29}$$

Hence,

$$
\begin{aligned}
u_c &= \lim_{n\to\infty} u_{n,c} && (30)\\
&\overset{\text{Eq. (28)}}{=} \lim_{n\to\infty}\left(\left(1-\frac{1}{n}\right)^n\right)^c && (31)\\
&= \left(\lim_{n\to\infty}\left(1-\frac{1}{n}\right)^n\right)^c && (32)\\
&\overset{\text{Eq. (25)}}{=} \left(\lim_{n\to\infty}\left(\frac{1}{1+\frac{1}{n}}\right)^n\right)^c && (33)\\
&= \left(\lim_{n\to\infty}\frac{1}{\left(1+\frac{1}{n}\right)^n}\right)^c && (34)\\
&\overset{\text{Eq. (26)}}{=} \left(\left(\lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n\right)^{-1}\right)^c && (35)\\
&\overset{\text{Eq. (29)}}{=} \left(\frac{1}{e}\right)^c && (36)\\
&= \frac{1}{e^c} && (37)
\end{aligned}
$$

c. As we know that the generalization error will be 0.5 (remember that the labels are random), the computed test error on $\mathcal{D}$ must be $0.5 \cdot 0.5 = 0.25$. Remember that the training error is equal to 0. Solving

$$u_c \cdot 0.5 + (1-u_c)\cdot 0 = 0.25 \tag{38}$$

for u_c yields 0.5. By solving

$$0.5 = u_c = \frac{1}{e^c} \tag{39}$$

for c, we get $c = 0.693$.

Problem 3 (Practical Assignment (Numerical Estimation Methods)):

First of all, you can download the dataset and the code skeleton from the course webpage. Consider the ridge regression problem in problem 1. The question is how can we select the best regularizer μ, which minimizes the expected prediction? To address this issue, you have to follow three steps, where in each step you need to complete a brief implementation and also interpret the results.

a. In this experiment, we split the dataset into a test and training set. Then we do leave-one-out cross-validation (LOOCV) on a subset of the training set. Finally, we compare the error on the test set to the cross-validation error for different choices of the regularizer. Note that running LOOCV on the whole training set is expensive, hence we only use a random subset of the training set. To run the code, you have to fill in two blanks in the implementation of LOOCV. After running the code, explain whether the cross-validation provides a good approximation of the test error. Does it under- or overestimate the test error? Is this what you would have expected? If not, try to find a reason for the unexpected behaviour. **Hint**: change the random seed of `numpy`. Can the LOOCV error be used in order to choose the regularizer?

b. In this part, we accelerate LOOCV to run it on a larger training set. Use the result of Problem 1 to complete the implementation of the improved LOOCV in the python skeleton. Then report the execution time and the prediction error of the improved LOOCV, which is printed by the code.

c. Here we investigate the dependency of the best regularizer on the size of the training set. To this end, we run a model selection function to choose a regularizer which minimizes the LOOCV on the training set. The output of the code is a graph which shows the best regularizer for the different sample sizes. Firstly, fill in the missing lines of the model selection function. Secondly, explain how the best regularizer depends on the size of the training set.

Solution 3 (Practical Assignment (Numerical Estimation Methods)):

a. In theory, the LOOCV error should overestimate the test error. What we observe for random seed 1 is that the LOOCV error underestimates the test error. Using 4 as random seed yields the result we would expect, i.e., the LOOCV error overestimates the test error. The two errors depend on how we split our data into training/validation and test sets. Repeating the LOOCV and the computation of the errors for 100 different random seeds shows that the LOOCV error overestimates the test error on average, which is what we would expect (see Figure 1).

   Even though the error curve of the LOOCV deviates from the one of the test error, the shapes of the

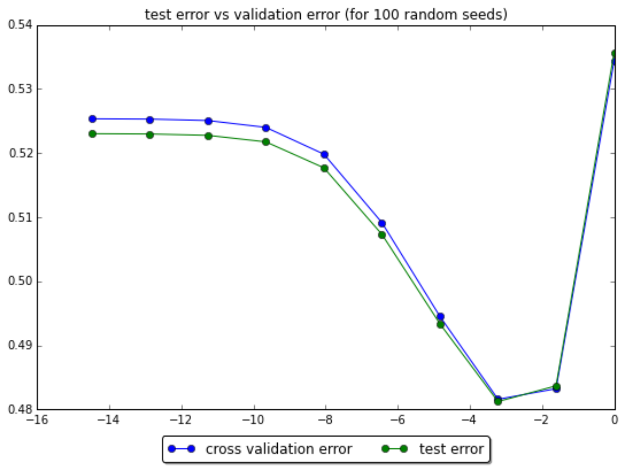


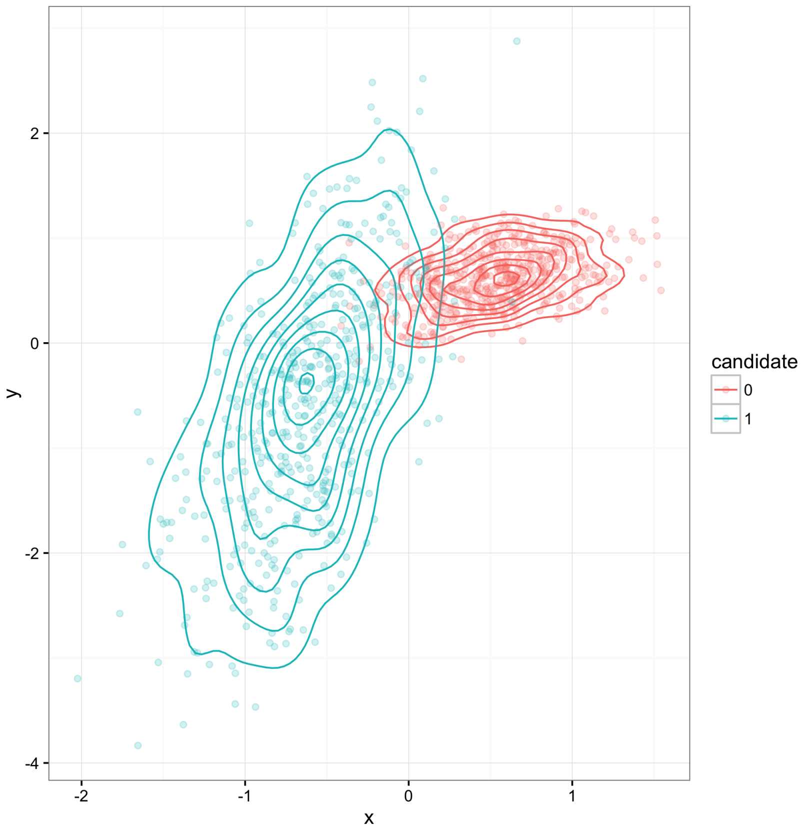
Figure 1: Ridge regression: LOOCV error vs test error for 100 different random seeds. The x-axis shows the natural logarithm of λ.

curves are very similar and the LOOCV error is a good indicator for which λ to pick.

c. The linear model we use here (due to its simplicity) is unlikely to overfit when we use a large training set. When we reduce the size of the training set, it becomes more likely that the training data does not represent the source of our data well, which can increase the variance of our prediction model. Therefore, λ should be chosen large for small training sets (in order to avoid overfitting) and can be set to smaller values as we increase the size of our training set.

Solution 4 (Decision Boundaries for Classification):

1. Here is the visualiastion; the classes appear linearly separable:



2. (a) The posterior probability of class membership is, by Bayes rule,

$$p(\mathcal{C}_a|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_a)p(\mathcal{C}_a)}{p(\mathbf{x}|\mathcal{C}_a)p(\mathcal{C}_a) + p(\mathbf{x}|\mathcal{C}_b)p(\mathcal{C}_b)} \tag{40}$$

Dividing above and below by $p(\mathbf{x}|\mathcal{C}_a)p(\mathcal{C}_a)$ we arrive at

$$p(\mathcal{C}_a|\mathbf{x}) = \frac{1}{1 + \frac{p(\mathbf{x}|\mathcal{C}_b)p(\mathcal{C}_b)}{p(\mathbf{x}|\mathcal{C}_a)p(\mathcal{C}_a)}} \tag{41}$$

and now writing

$$a = -\log\left(\frac{p(\mathbf{x}|\mathcal{C}_b)p(\mathcal{C}_b)}{p(\mathbf{x}|\mathcal{C}_a)p(\mathcal{C}_a)}\right) \tag{42}$$

we have

$$p(\mathcal{C}_a|\mathbf{x}) = \frac{1}{1 + e^{-a}} \tag{43}$$

which is the logistic function.

(b) The argument is defined in equation 42. Classification with logistic regression checks if the posterior probability for the class (that is, the value of the logistic function in equation 43) is above 0.5. This corresponds to when the sign of its argument changes from negative to positive. This argument is a logarithm, which changes sign when *its* argument goes above one. So, we see that the classification changes when $p(\mathbf{x}|\mathcal{C}_a)p(\mathcal{C}_a) > p(\mathbf{x}|\mathcal{C}_b)p(\mathcal{C}_b)$, and vice versa. This is of course, the property that we wanted.

3. (a) The decision boundary is where

$$p(\mathbf{x}|v_0)p(v_0) = p(\mathbf{x}|v_1)p(v_1) \tag{44}$$

defining the priors with $p(v_0) = p_0$ and $p(v_1) = p_1$ we have

$$\frac{1}{2\pi|\Sigma|^{-/1/2}} e^{\left(-\frac{1}{2}(\mathbf{x}-\mu_0)^T\Sigma^{-1}(\mathbf{x}-\mu_0)\right)} p_0 = \frac{1}{2\pi|\Sigma|^{-1/2}} e^{\left(-\frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma^{-1}(\mathbf{x}-\mu_1)\right)} p_1 \tag{45}$$

We can cancel the prefactors and take logs of both sides to get

$$-\frac{1}{2}\left((\mathbf{x}-\mu_0)^T\Sigma^{-1}(\mathbf{x}-\mu_0)\right) + \log p_0 = -\frac{1}{2}\left((\mathbf{x}-\mu_1)^T\Sigma^{-1}(\mathbf{x}-\mu_1)\right) + \log p_1 \tag{46}$$

Because we made the assumption that $\Sigma_0 = \Sigma_1 = \Sigma$, the terms quadratic in $\mathbf{x}$ cancel out (otherwise we obtain a *quadratic* decision boundary). Gathering terms containing $\mathbf{x}$ together, we get

$$\mathbf{x}^T\left(\Sigma^{-1}(\mu_0 - \mu_1)\right) - \frac{1}{2}\mu_0^T\Sigma^{-1}\mu_0 + \frac{1}{2}\mu_1^T\Sigma\mu_1 + \log\frac{p_0}{p_1} \tag{47}$$

We want to turn this into the form of a linear decision boundary, e.g. $w^T(x - x_0) = 0$. By inspection we can see that $w = \Sigma^{-1}(\mu_0 - \mu_1)$. Finding the form of x_0 is a little more involved. We know that the rest of this expression is $-w^T x_0$, that is

$$w^T x_0 = \frac{1}{2}\mu_0^T\Sigma^{-1}\mu_0 - \frac{1}{2}\mu_1^T\Sigma\mu_1 - \log\frac{p_0}{p_1} \tag{48}$$

The first two terms look like $a^2 - b^2$, which we can factorise as $(a-b)(a+b)$, yielding

$$w^T x_0 = \frac{1}{2}(\mu_0 - \mu_1)^T\Sigma^{-1}(\mu_0 + \mu_1) - \log\frac{p_0}{p_1} \tag{49}$$

So the first term is then $w^T\frac{1}{2}(\mu_0 + \mu_1)$. For the second term, we can imagine a prefactor for the log term which goes to 1 when multiplied by w^T - such a factor would be

$$\frac{\mu_0 - \mu_1}{(\mu_0 - \mu_1)^T\Sigma^{-1}(\mu_0 - \mu_1)} \tag{50}$$

which produces the result from lecture 8;

$$x_0 = \frac{1}{2}(\mu_0 + \mu_1) - \frac{\mu_0 - \mu_1}{(\mu_0 - \mu_1)^T\Sigma^{-1}(\mu_0 - \mu_1)}\log\frac{p_0}{p_1} \tag{51}$$

(b) As mentioned, to show the decision boundary is quadratic it is sufficient to show that when $\Sigma_0 \neq \Sigma_1$, the term quadratic in $\mathbf{x}$ doesn't cancel.

(c) We have

$$\hat{\mu} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \tag{52}$$

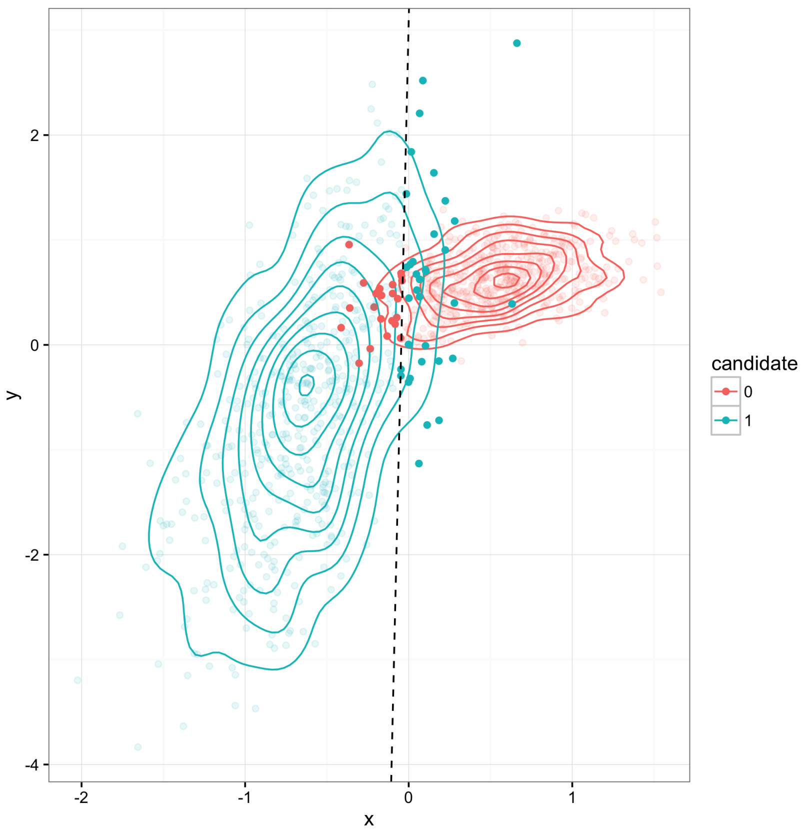
and

$$\hat{\Sigma} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \hat{\mu})(\mathbf{x}_n - \hat{\mu})^T \tag{53}$$

which produce:

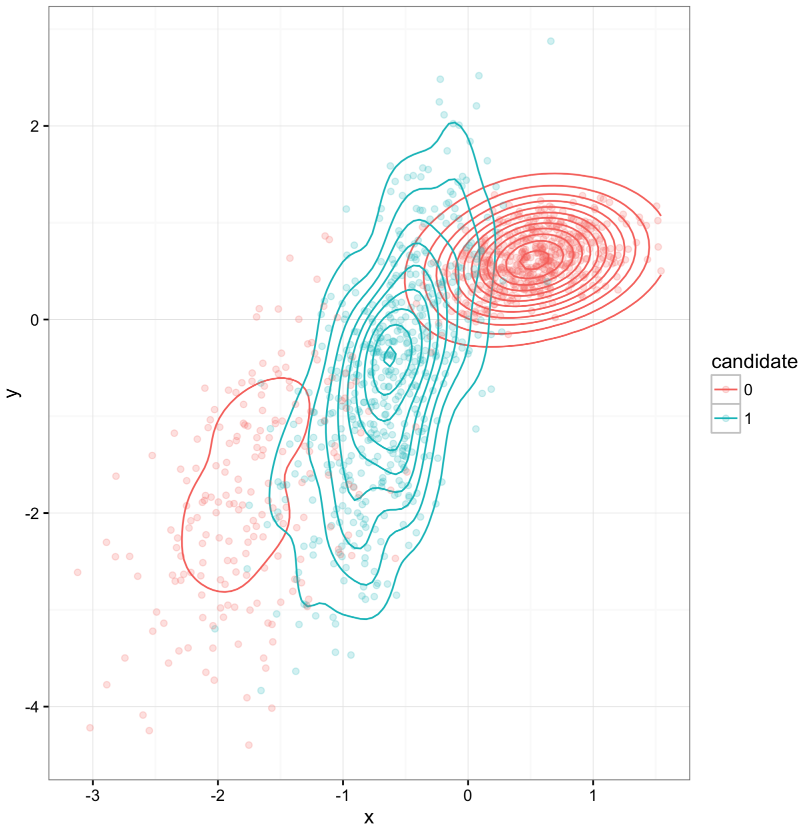
- $\hat{\mu}_0 = (0.5368697, 0.6181328)$
- $\hat{\mu}_1 = (-0.6277704 - 0.6098245)$

- $\hat{\boldsymbol{\Sigma}}_0 = \begin{pmatrix} 0.12779717 & 0.03947451 \\ 0.03947451 & 0.07781321 \end{pmatrix}$
- $\hat{\boldsymbol{\Sigma}}_1 = \begin{pmatrix} 0.1571848 & 0.2766283 \\ 0.2766283 & 1.3135227 \end{pmatrix}$



(d)

(e) Number of misclassified voters: 39



4. Now, one of the distributions appears to be *bimodal*. Alternately, there are three classes (and two of them happen to vote the same way). We can do the classification with either approach, but our decision boundaries will need to be recalculated.

5. There are various answers to this question. One example would be the multivariate Student distribution, which has support over the same domain as the multivariate normal distribution, but has more probability mass in extreme values.

Solution 5 (Perceptrons):

There are many possible solutions to this problem.

A simple one is to use an additional nonlinear feature, $x_0 x_1$. The weights can then be deduced by looking at the truth table;

x_0	x_1	$x_0 x_1$	label
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	0

Adding a constant bias of -0.5, we see that $2x_0x_1 - x_0 - x_1 - 0.5$ produces 0.5 when the label is 1, and -0.5 when the label is 0, satisfying our criterion.

Solution 6 (Relationship between Fisher's discriminant and Least Squares):

The solution is provided in Bishop's "Pattern Recognition and Machine Learning" p. 189, 190. Basically the idea is to take partial derivatives with respect to **w** and w_0 and set those to zero; from there, use our new choice of labels t_n, in order to be able to express **w** and w_0 in the desired form.

The derivation of equation (4.37) is left as an exercise in the book. The 'straight-forward' algebra that leads to equation (4.37) is as follows:

Using (4.21) and (4.34) along with the chosen target coding scheme, we can re-write the LHS of (4.33) as follows:

$\sum_{n=1}^{N}(w^Tx_n + w0 - t_n)x_n$

$= \sum_{n=1}^{N}(w^Tx_n - w^Tm - t_n)x_n$ (Using (4.34))

$= \sum_{n=1}^{N}((x_nx_n^T - x_nm^T)w - x_nt_n)$ (Using $(a^Tb)b = b(a^Tb) = b(b^Ta) = (bb^T)a$).

$= \sum_{n\in C1}((x_nx_n^T - x_nm^T)w - x_nt_n) + \sum_{n\in C2}((x_nx_n^T - x_nm^T)w - x_nt_n)$

$= (\sum_{n\in C1}(x_nx_n^T) - N_1m_1m^T)w - N_1m_1\frac{N}{N_1}$

$+(\sum_{n\in C2}(x_nx_n^T) - N_2m_2m^T)w + N_2m_2\frac{N}{N_2}$ (Using (4.21))

$= (\sum_{n\in C1}(x_nx_n^T) + \sum_{n\in C2}(x_nx_n^T) - (N_1m_1 + N_2m_2)m^T)w - N(m_1 - m_2) :: (A)$

We now use the identity:

$\sum_{i\in C_k}(x_i - m_k)(x_i - m_k)^T$

$= \sum_{i\in C_k}(x_ix_i^T - x_im_k^T - m_kx_i^T + m_km_k^T)$

$= \sum_{i\in C_k}(x_ix_i^T) - N_km_km_k^T$

Thus,

$\sum_{i\in C_k}(x_ix_i^T) = \sum_{i\in C_k}(x_i - m_k)(x_i - m_k)^T + N_km_km_k^T :: (B)$

Using (B), (4.28), we can re-write (A) as follows:

$(S_w + N_1m_1m_1^T + N_2m_2m_2^T - (N_1m_1 + N_2m_2)m^T)w - N(m_1 - m_2)$

Using (4.36), this becomes:

$(S_w + N_1m_1m_1^T + N_2m_2m_2^T - (N_1m_1 + N_2m_2)\frac{1}{N}(N_1m_1^T + N_2m_2^T))w - N(m_1 - m_2)$

Re-arranging terms, this becomes:

$(S_w + (N_1 - \frac{N_1^2}{N})m_1m_1^T - \frac{N_1N_2}{N}(m_1m_2^T + m_2m_1^T) + (N_2 - \frac{N_2^2}{N})m_2m_2^T)w - N(m_1 - m_2)$

Re-arranging terms, this becomes:

$(S_w + (\frac{(N_1+N_2)N_1 - N_1^2}{N})m_1 m_1^T - \frac{N_1 N_2}{N}(m_1 m_2^T + m_2 m_1^T) + (\frac{(N_1+N_2)N_2 - N_2^2}{N})m_2 m_2^T)w - N(m_1 - m_2)$

Re-arranging terms, this becomes:

$(S_w + \frac{N_1 N_2}{N}(m_1 m_1^T - m_1 m_2^T - m_2 m_1^T + m_2 m_2^T))w - N(m_1 - m_2)$

Using equation (4.27), this becomes:

$(S_w + \frac{N_1 N_2}{N} S_B)w - N(m_1 - m_2)$

From (4.33), this must be equal to zero. This gives us (4.37).