Statistical Machine Learning
Solutions for Exam 2020-06-17

1. i. **True**.
   ii. **False**
   iii. **False**
   iv. **False**. It is called linear if it has a linear decision boundary.
   v. **False**
   vi. **False**
   vii. **False**
   viii. **True**
   ix. **False**. The mean squared error is the sum of the bias and the variance. An increase in bias can reduce the variance.
   x. **False**

2. (a) The data are visualized in Figure 1. The numbers are used in (c).

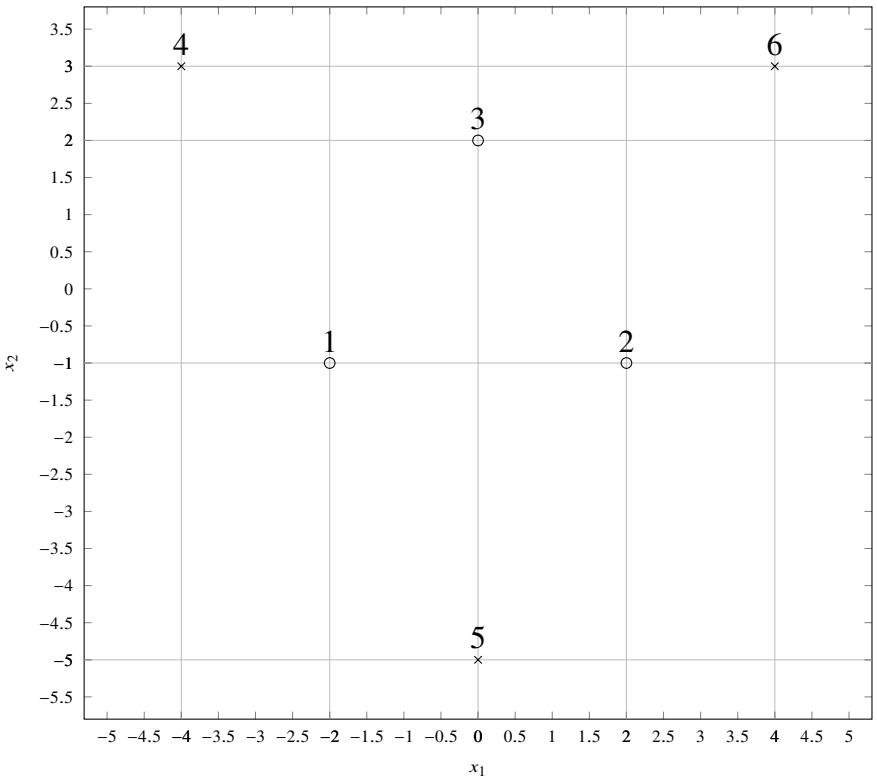

Figure 1: Visualization of the data in problem 2.

(b) iv. A 1-NN classifier give 0% training error.

v. The 3-NN classifier will classify all training points as $\circ$ and get 50% misclassification error.

ii. The means are given by $\mu_\times = (0, \frac{1}{3})$ and $\mu_\circ = (0, 0)$. The decision boundary is the straight line $x_2 = \frac{1}{6}$, where points above are classified as $\times$and points below as $\circ$. This gives a misclassification error of 33%.

vi. The classification tree can split along lines parallel too the x- or y-axis. The best we can achieve is to misclassify one point and get 17% misclassification error. This is achieved by splitting at $x_2 = 2.5$

i. Since the data is not linearly separable, logistic regression can not achieve 0% misclassification error and must achieve 33%. The decision boundary is similar to LDA.

iii. A quadratic decision boundary can separate the two classes and a QDA classifier achieves 0% misclassification error.

(c) To make the data linearly separable, we can only move point 5. If we move it above point 3 all classifiers but kNN with $k = 3$ will achieve 0% misclassification error.

Let us now focus on the 3NN classifier under the constraint that point 5 has to be above the line $x_2 = 2$. For correct classification of point 4 and 6, we need point 5 to be closer to point 4 and 6 than point 1 and 2 are, respectively. This is the intersection of two circles centered at point 4 and 6 with radius $\sqrt{20}$.

For point 5 to be correctly classified, given that it is above $x_2 = 2$, point 5 needs to be closer to point 4 and 6 than it is to point 1 and 2. This is the intersection of two half planes with boundaries at the lines passing through the midpoints between points 4 and 1 and point 6 and 2 with slope 1/2 and -1/2. (These are lines orthogonal to the line between points 4 and 1 and 2 and 6.)

The region where we can place point 5 is the shades area Figure 2. This far you can get graphically with a ruler and a compass (passare).

Now we can just pick a point inside the region. For example $(0, 4)$.

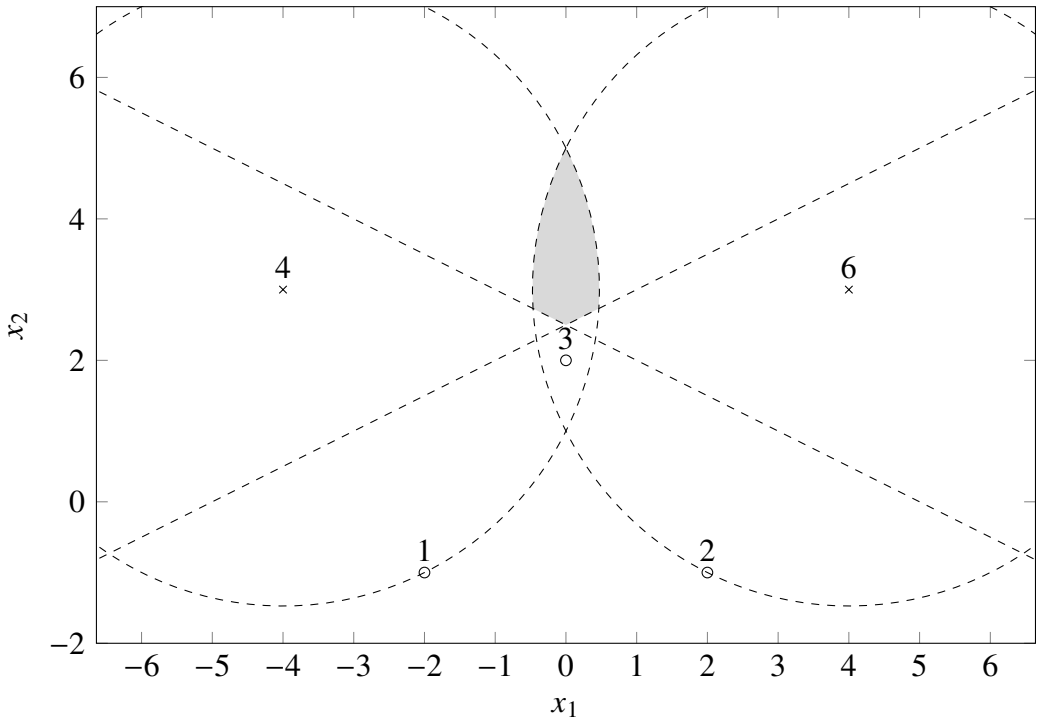

Figure 2: Area in 2 (c).

3. (a) The partitioning can be seen in Figure 3

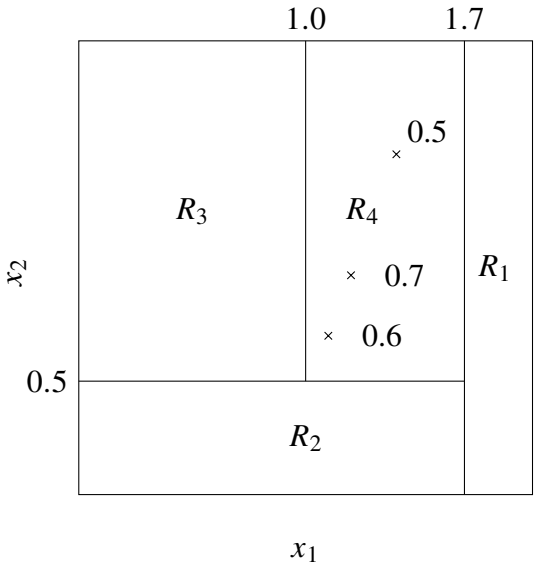

Figure 3: Partitioning of the input space for problem 3(a).

(b) All regions already have two or less data points, except for region R_4 which has three. Therefore, we need to make one additional split in that region, where one of the resulting regions will have two data points and the other region one data point.

In Figure 3, the three data points are depicted. The two possible splits are (i) to put 0.5 in one region and 0.7 and 0.6 in the other region, or (ii) to put 0.6 in one region and 0.5 and 0.7 in the other region. The MSE for these two options will be

(i) MSE = $(0.5 - 0.5)^2 + (0.7 - 0.65)^2 + (0.6 - 0.65)^2 = 2 * 0.05^2$

(ii) MSE = $(0.6 - 0.6)^2 + (0.7 - 0.6)^2 + (0.5 - 0.6)^2 = 2 * 0.1^2$

Clearly, option (i) will give a smaller MSE. This split could be realized, for example with the split $x_2 \leq 1.2$.

(c) Both boosting and bagging are meta algorithms that are applied on top of base models. In both algorithms, we create multiple copies of the base model trained on different variants of the data. In bagging, we apply multiple copies of the base model in parallel by sampling with replacement from the original dataset (bootstrapping) and learning each model using the separate datasets. The final prediction is an average of the individual models predictions.

In boosting, we apply copies of the base model sequentially but with the dataset re-weighted.

Typically, bagging reduces the variance in complex models without increasing the bias and boosting reduces the bias in high bias models without increasing the variance.

For a tree, this means we use a shallow tree in boosting and a deep tree in bagging.

4. (a) The problem is that the company did not set aside test data **before** the model had seen it. In this case, they picked the 100 most correlated input variables on **all** the data, including test data, and not only training data. To get a correct assessment of the new data error, test data must be put aside before the 100 most correlated features are selected.

   (b) With increasing λ we get less regularization and a higher model complexity. For large λ, the right of the graph in the figure, we get a complex model with lower bias and higher variance. In the left part, we get a high bias and low variance model.

   (c) For small enough λ we get $\boldsymbol{\theta} \equiv \mathbf{0}$.

5. (a) The number of parameters increase with the number of data and as such a non-parametric model.

   (b) Stack the outputs in $\mathbf{y}$, the parameters in $\boldsymbol{\theta}$ and the unmodelled errors in $\boldsymbol{\epsilon}$. For $\mathbf{X}$ we get

$$\mathbf{X} = \begin{bmatrix} \phi_{x_1}(x_1) & \phi_{x_2}(x_1) & \cdot & \phi_{x_N}(x_1) \\ \phi_{x_1}(x_2) & \phi_{x_2}(x_2) & \cdot & \phi_{x_N}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{x_1}(x_N) & \phi_{x_2}(x_N) & \cdot & \phi_{x_N}(x_N) \end{bmatrix}.$$

   (c) The parameter estimate for ridge regression is given by $\widehat{\boldsymbol{\theta}}_{RR} = \left(\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}$. The best we can do about the noise is predict $\widehat{\boldsymbol{\epsilon}} = \mathbf{0}$. We get the prediction

$$\widehat{f}(x_\star) = \mathbf{X}_\star\widehat{\boldsymbol{\theta}}_{RR} = \mathbf{X}_\star\left(\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}.$$

   (d) From (b) and the hint, we get that the prediction is formed as

$$\begin{aligned} \widehat{f}(x_\star) &= \mathbf{X}_\star\left(\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^\mathsf{T}\mathbf{y} \\ &= \mathbf{X}_\star\mathbf{X}^\mathsf{T}\left(\mathbf{X}\mathbf{X}^\mathsf{T} + \lambda\mathbf{I}\right)^{-1}\mathbf{y} \\ &= \frac{1}{D}\mathbf{X}_\star\mathbf{X}^\mathsf{T}\left(\frac{1}{D}\mathbf{X}\mathbf{X}^\mathsf{T} + \lambda\mathbf{I}\right)^{-1}\mathbf{y} \end{aligned}$$

   and $\mathbf{X}$ is given by

$$\mathbf{X} = \begin{bmatrix} \phi_{\xi_{-D2^D}}(x_1) & \phi_{\xi_{-D2^D+1}}(x_1) & \cdots & \phi_{\xi_{D2^D}}(x_1) \\ \phi_{\xi_{-D2^D}}(x_2) & \phi_{\xi_{-D2^D+1}}(x_2) & \cdots & \phi_{\xi_{D2^D}}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{\xi_{-D2^D}}(x_N) & \phi_{\xi_{-D2^D+1}}(x_N) & \cdots & \phi_{\xi_{D2^D}}(x_N) \end{bmatrix}.1$$

   Notice how $\mathbf{X}$ only show up in the product $\mathbf{X}_\star\mathbf{X}^\mathsf{T}$ and $\mathbf{X}\mathbf{X}^\mathsf{T}$. We get

$$\mathbf{X}_\star\mathbf{X}^\mathsf{T} = \begin{bmatrix} \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_\star)\phi_{\xi_i}(x_1) & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_\star)\phi_{\xi_i}(x_2) & \cdots & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_\star)\phi_i(x_N) \end{bmatrix},$$

$$\mathbf{X}\mathbf{X}^\mathsf{T} = \begin{bmatrix} \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_1)\phi_{\xi_i}(x_1) & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_1)\phi_{\xi_i}(x_2) & \cdots & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_1)\phi_{\xi_i}(x_N) \\ \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_2)\phi_{\xi_i}(x_1) & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_2)\phi_{\xi_i}(x_2) & \cdots & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_2)\phi_{\xi_i}(x_N) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_2)\phi_{\xi_i}(x_1) & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_N)\phi_{\xi_i}(x_N) & \cdots & \sum_{i=-D2^D}^{D2^D}\phi_{\xi_i}(x_N)\phi_{\xi_i}(x_N) \end{bmatrix}.$$

Let $D \to \infty$

$$\frac{1}{D}\sum_{-D2^D}^{D2^D}\phi_{\xi_i}(s)\phi_{\xi_i}(t) \to \int_{-\infty}^{\infty}\phi_\tau(s)\phi_\tau(t)d\tau = \int_{-\infty}^{\infty}e^{-(t-\tau)^2}e^{-(s-\tau)^2}d\tau$$

$$= \int_{-\infty}^{\infty}e^{-2\left(\tau+\frac{t+s}{2}\right)^2}e^{-\frac{1}{2}(t-s)^2}d\tau = \sqrt{\frac{\pi}{2}}e^{-\frac{1}{2}(t-s)^2}.$$