Midterm 2: Solutions

Introduction to Data Science

October 25, 2017

1 Supervised Learning

Supervised learning is the machine learning task of inferring a function from labeled training data. The training data consist of a set of training examples.

1.1 Decision Trees

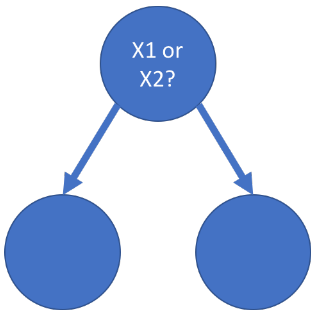
We have here some training data. The "classification" column is the target we are trying to learn. "X1" and "X2" are the features of the training data.

Instance	Classification	X1	X2
1	+	True	True
2	+	True	True
3	-	True	False
4	+	False	False
5	-	False	True
6	-	False	True

Recall that the entropy is the average information yielded by a random variable and that it is defined as:

$$H(X) = -\sum_{x \in X} P(x) \log_2 P(x) \tag{1}$$

Consider the entropy of this collection of training examples with respect to the target function "classification". We going to split the data into 2 subsets according to one of the features ("True" goes left, "False" goes right). Each subset ends up in one of the children nodes of this following binary tree.



Question 1: From this data estimate $P(+|X_1 = \text{True}, X_2 = \text{True})$, $P(+|X_1 = \text{False}, X_2 = \text{True})$, $P(+|X_1 = \text{True}, X_2 = \text{False})$, $P(+|X_1 = \text{False}, X_2 = \text{False})$, $P(X_1 = \text{True}, X_2 = \text{False}|-)$ and $P(X_1 = \text{False}|-)$.

Solution 1:

- $P(+|X_1 = \text{True}, X_2 = \text{True}) = 1$
- $P(+|X_1 = \text{False}, X_2 = \text{True}) = 0$
- $P(+|X_1 = \text{True}, X_2 = \text{False}) = 0$
- $P(+|X_1 = \text{False}, X_2 = \text{False}) = 1$
- $P(X_1 = \text{True}, X_2 = \text{False}|-) = \frac{1}{3}$
- $P(X_1 = \text{False}|-) = \frac{2}{3}$

Question 2: what feature yields a split with the most information gain from the root node to the children nodes? We define the information gain as:

$$I = H_{\text{root}}(X) - P(\text{left})H_{\text{left}}(X) - P(\text{right})H_{\text{right}}(X) - \tag{2}$$

where $P(\text{left})$ and $P(\text{right})$ are the probabilities to go left or right respectively.

Solution 2: In the root node $P(+) = P(-) = 1/2$

$$H_{\text{root}}(X) = -P(+)\log_2 P(+) - P(-)\log_2 P(-) = 1 \tag{3}$$

If we choose "X1" as the splitting variable, we have $P(+) = 2/3$ and $P(-) = 1/3$ for the left node and $P(+) = 1/3$ and $P(-) = 2/3$ for the right node:

$$H_{\text{left}}(X) = H_{\text{right}}(X) = -\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{2}{3}\log_2\left(\frac{2}{3}\right) \simeq 0.918 \tag{4}$$

We also have $P(\text{left}) = 1/2$ and $P(\text{right}) = 1/2$ therefore:

$$I = 1 - 0.9183 = 0.0817 \tag{5}$$

If we choose "X2" as the splitting variable, we have $P(+) = 1/2$ and $P(-) = 1/2$ for the left node and $P(+) = 1/2$ and $P(-) = 1/2$ for the right node:

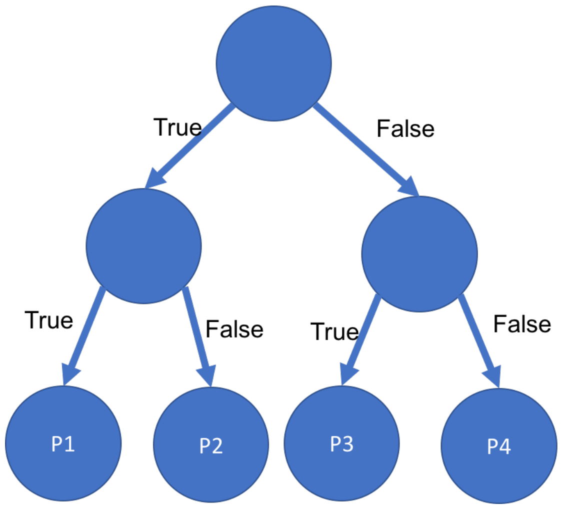
$$H_{\text{left}}(X) = H_{\text{right}}(X) = 1 \tag{6}$$

We also have $P(\text{left}) = 2/3$ and $P(\text{right}) = 1/3$ therefore

$$I = 1 - \frac{1}{3} - \frac{2}{3} = 0 \tag{7}$$

"X2" does not provide any information gain thus we prefer to split on "X1".

Question 3: Base on question 1 and question 2, provide P_1, P_2, P_3 and P_4, the probabilities to be + in the leaf node 1, 2, 3 or 4 in the following decision tree:



We are using one of the variables for the first split and the other one for the second split.

Solution 3: Because we split on "X1" on and then on "X2", we have:

- $P_1 = P(+|X_1 = \text{True}, X_2 = \text{True}) = 1$
- $P_3 = P(+|X_1 = \text{False}, X_2 = \text{True}) = 0$
- $P_2 = P(+|X_1 = \text{True}, X_2 = \text{False}) = 0$
- $P_4 = P(+|X_1 = \text{False}, X_2 = \text{False}) = 1$

1.2 Maximum likelihood estimation for Linear Regression

We are going to consider realizations of responses y and the set of variables $\{X_1, X_2, ..., X_m\}$ pair. We want to estimate y as a linear combination of variables. For each sample (i), we have

$$\widehat{y}^{(i)} = \beta_0 + \beta_1 X_1^{(i)} + \beta_2 X_2^{(i)} + ... + \beta_m X_m^{(i)} = \sum_{j=0}^{m} \beta_j X_j^{(i)} \text{ with } X_0 = 1 \tag{8}$$

We can express $\sum_{j=0}^{m} \beta_j X_j^{(i)} = \beta^T \mathbf{X}^{(i)}$ in matrix notation where $\mathbf{X}^{(i)}$ is the row (i) of the data. This model generate an error $\epsilon^{(i)}$ such that for each sample (i) we have

$$y^{(i)} = \widehat{y}^{(i)} + \epsilon^{(i)}. \tag{9}$$

We are going to assume that $\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2)$ is a random normal variable with

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(\epsilon^{(i)})^2}{2\sigma^2}\right) \tag{10}$$

Question 1: Express $p(\epsilon^{(i)}) = p(y^{(i)}|\mathbf{X}^{(i)}; \beta)$ as a function of $y^{(i)}$, $\mathbf{X}^{(i)}$ and β.

Solution 1:

$$p(y^{(i)}|\mathbf{X}^{(i)}; \beta) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(y^{(i)} - \beta^T \mathbf{X}^{(i)})^2}{2\sigma^2}\right) \tag{11}$$

Question 2: The likelihood function is defined as

$$L(\beta) = \prod_{i=1}^{n} p(y^{(i)}|\mathbf{X}^{(i)}; \beta) \tag{12}$$

where we assume that the samples are independent from each other and n is the total number of samples. $l(\beta) = \log L(\beta)$ is the log-likelihood. Prove that maximizing $L(\beta)$ with respect to β is equivalent to minimizing the mean square error $J(\beta) = \sum_{i=1}^{n} (y^{(i)} - \beta^T \mathbf{X}^{(i)})^2$.

Solution 2: Maximizing $l(\beta)$ is equivalent to maximizing $L(\beta)$.

$$\begin{aligned} l(\beta) &= \sum_{i=1}^{n} \log p(y^{(i)}|\mathbf{X}^{(i)}; \beta) \\ &= \sum_{i=1}^{n} -\frac{(y^{(i)} - \beta^T \mathbf{X}^{(i)})^2}{2\sigma^2} - \log(\sqrt{2\pi\sigma}) \\ &= -\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y^{(i)} - \beta^T \mathbf{X}^{(i)})^2 - \log(\sqrt{2\pi\sigma}) \end{aligned} \tag{13}$$

Therefore by minimizing $\sum_{i=1}^{n}(y^{(i)} - \beta^T \mathbf{X}^{(i)})^2$ we are maximizing $l(\beta)$ and consequence $L(\beta)$.

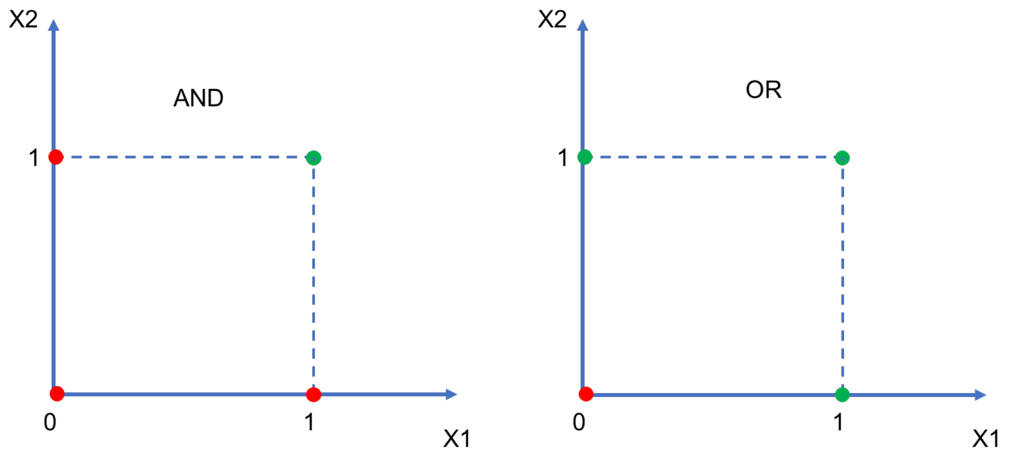
1.3 Learning XOR with a Perceptron model

We define the AND and the OR operators such that

X_1	X_2	X_1 AND X_2
1	1	1
1	0	0
0	1	0
0	0	0

X_1	X_2	X_1 OR X_2
1	1	1
1	0	1
0	1	1
0	0	0

We can represent those operators graphically where the green dot represents the outcome 1 and the red dot the outcome 0.



We consider the single layer perceptron model that attempts to classify data points. Let's consider M input variables $\{X_1, X_2, \ldots, X_M\}$ with each example outcome associated to a binary variable $Y \in \{T, F\}$. We want to learn weights $\{w_0, w_1, w_2, \ldots, w_M\}$ such that

$$w_0 + \sum_{i=1}^{M} w_i X_i > 0 \text{ if } Y = 1 \tag{14}$$

and

$$w_0 + \sum_{i=1}^{M} w_i X_i < 0 \text{ if } Y = 0 \qquad (15)$$

where w_0 is the intercept.

Question 1: In the case $M = 2$ as described above, values for w_0, w_1 and w_2 to classify correctly outcomes of the AND operator.

Solution 1: We just need a line that goes above the $(1, 0)$ and $(0, 1)$ points and below $(1, 1)$. If we have $X_2 = aX_2 + b$, we can choose $a = -1$ and $b = 1.5$ or $w_1 = w_2 = 1$ and $w_0 = -1.5$.

Question 2: In the case $M = 2$ as described above, values for w_0, w_1 and w_2 to classify correctly outcomes of the OR operator.

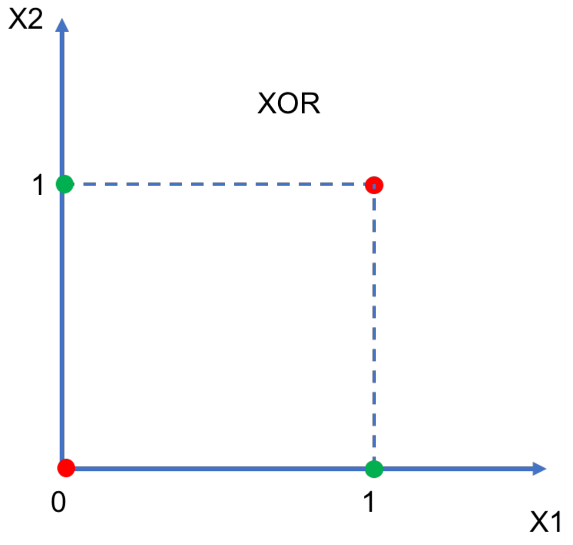
Solution 2: We just need a line that goes below the $(1, 0)$ and $(0, 1)$ points and above the $(0, 0)$. If we have $X_2 = aX_2 + b$, we can choose $a = -1$ and $b = 0.5$ or $w_1 = w_2 = 1$ and $w_0 = -0.5$.

Lets now define the XOR operator such that

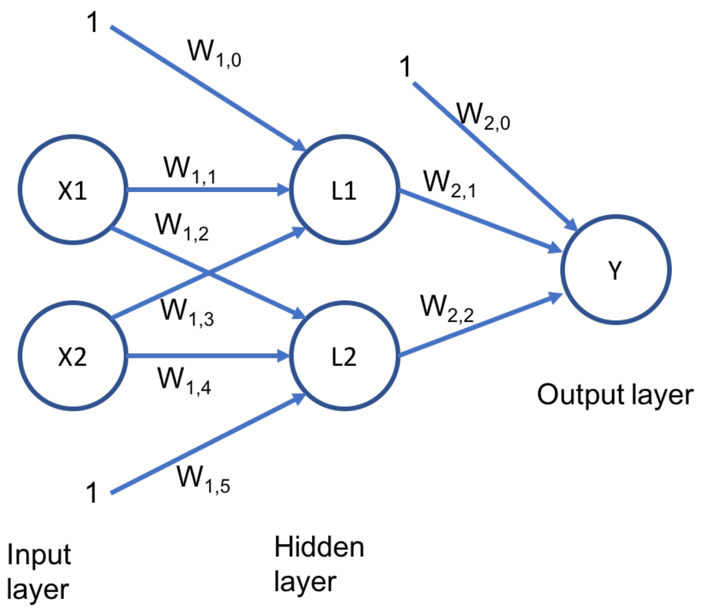
X_1	X_2	X_1 XOR X_2
1	1	0
1	0	1
0	1	1
0	0	0

Question 3: represent on a graph the outcomes of this operator

Solution 3:



Question 4: We cannot anymore classify the outcomes of this operator with simply tracing a line on the graph. We say that the outcomes of XOR are not linearly separable. To be able to learn XOR, we need induce non-linearity by introducing a so called "hidden layer" to the model:



The rules of this more complex model are quite similar. On the input layer, the output in L_1 is such that

$$w_{1,0} + \sum_{i=1}^{M} w_{1,i} X_i > 0 \text{ if } L_1 = 1 \tag{16}$$

and

$$w_{1,0} + \sum_{i=1}^{M} w_{1,i} X_i < 0 \text{ if } L_1 = 0 \tag{17}$$

where $w_{1,0}$ is the intercept. And on the hidden layer

$$w_{2,0} + \sum_{i=1}^{M} w_{2,i} L_i > 0 \text{ if } Y = 1 \tag{18}$$

and

$$w_{2,0} + \sum_{i=1}^{M} w_{2,i} L_i < 0 \text{ if } Y = 0 \tag{19}$$

Find a set $\{w_{1,0}, w_{1,1}, w_{1,2}, w_{1,3}, w_{1,4}, w_{1,5}, w_{2,0}, w_{2,1}, w_{2,2}\}$ such that this neural network can predict the outcomes of the XOR operator.

Solution 4: Let's choose $\{w_{2,0}, w_{2,1}, w_{2,2}\}$ such that the hidden to the output acts as a AND operator $w_{2,1} = w_{2,2} = 1$ and $w_{2,0} = -1.5$. Now we just need to choose the remaining weights such that $L_1 = 1$ and $L_2 = 1$ when $X_1 = 1$ and $X_2 = 0$ or $X_1 = 0$ and $X_2 = 1$ and $L_1 = 0$ and $L_2 = 1$ (or $L_1 = 1$ and $L_2 = 0$ or $L_1 = 0$ and $L_2 = 0$) when $X_1 = 0$ and $X_2 = 0$ or $X_1 = 1$ and $X_2 = 1$. We know that $Y = 1$ if $X_1 = 1$ and $X_2 = 0$ or $X_1 = 0$ and $X_2 = 1$. So in this case by symmetry

$$w_{1,0} + w_{1,1} = w_{1,0} + w_{1,3} > 0 \tag{20}$$

and

$$w_{1,5} + w_{1,2} = w_{1,5} + w_{1,4} > 0 \tag{21}$$

therefore $w_{1,1} = w_{1,3}$ and $w_{1,2} = w_{1,4}$. We know that if $Y = 0$ then L_1 can be 0 and we can have $X_1 = 1$ and $X_2 = 1$ therefore we can have

$$w_{1,0} + 2w_{1,1} < 0 \tag{22}$$

$w_{1,1} = w_{1,3} = -1$ and $w_{1,0} = 1.5$ can satisfy those constraints.

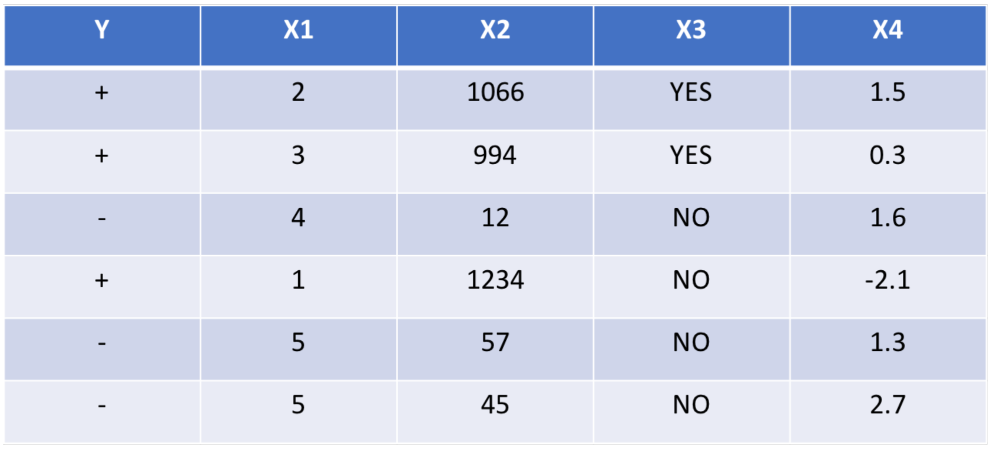
We can choose to make $L_2 = 1$ when $L_1 = 0$, therefore we can have

$$w_{1,5} + 2w_{1,2} > 0 \tag{23}$$

$w_{1,2} = w_{1,4} = 1$ and $w_{1,0} = -0.5$ can satisfy those constraints.

1.4 The k-Nearest Neighbor (k-NN) algorithm

Consider the following training data for a k-NN algorithm:



Y	X1	X2	X3	X4
+	2	1066	YES	1.5
+	3	994	YES	0.3
-	4	12	NO	1.6
+	1	1234	NO	-2.1
-	5	57	NO	1.3
-	5	45	NO	2.7

When we want to classify a new instance, we just need to compute the "distance" of this new instance from the instances of the training data and classify this new instance as its set of k nearest neighbors. To measure a distance, we are going to use here the Euclidean distance.

Question 1: Let's consider this new instance: $\{7, 1323, \text{NO}, 1.7\}$. Compute the Euclidean distance of this instance from the training data instance. You can convert YES $= 1$, NO $= 0$. What class would you predict for this instance if you were to look at the closest training data instance ($k = 1$)?

Solution 1:

1. $\sqrt{(7-2)^2 + (1323-1066)^2 + (1-0)^2 + (1.7-1.5)^2} = 257.0507$
2. $\sqrt{(7-3)^2 + (1323-994)^2 + (0-1)^2 + (1.7-0.3)^2} = 329.0288$
3. $\sqrt{(7-4)^2 + (1323-12)^2 + (0-0)^2 + (1.7-1.6)^2} = 1311.003$
4. $\sqrt{(7-1)^2 + (1323-1234)^2 + (0-0)^2 + (1.7+2.1)^2} = 89.28292$
5. $\sqrt{(7-5)^2 + (1323-57)^2 + (0-0)^2 + (1.7-1.3)^2} = 1266.002$
6. $\sqrt{(7-5)^2 + (1323-45)^2 + (0-0)^2 + (1.7-2.7)^2} = 1278.002$

The closest point is the 4th instance, therefore we would classify this new instance as $(+)$.

Question 2: You may have noticed that most of the distance comes from the variable X_2 due to the scale on which this variable lives. For k-NN it is important to rescale the variables. Recompute the Euclidean distance but this time rescale X_1, X_2 and X_4 (X_3 is more well behaved) by their standard deviation. What class would you predict for this instance if you were to look at the closest training data instance with these normalized axes?

Solution 2:

1. 3.253007
2. 2.834675
3. 2.895549
4. 4.339198
5. 2.495313
6. 2.573738

The closest point is the 5th instance, therefore we would classify this new instance as $(-)$.

Question 3: Let's now consider $k = 3$. Find the 3 nearest neighbors of the new instance described above. What class would you predict if you compare the new instance to its 3 nearest neighbors?

Solution 3: Its 3 nearest neighbors are instances $5(-)$, $6(-)$ and $2(+)$. Because there are more $(-)$ instances, we still predict $(-)$ for the new instance.

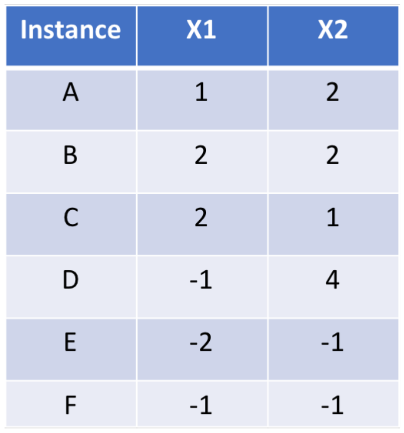
Question 4: In general do you think it is better to choose k as being odd or even? Why?

Solution 4: If we choose k as even, it is possible to have ties where the odd case alleviates this possible problem.

2 Unsupervised Learning

2.1 Clustering: the K-mean algorithm

The k-mean algorithm is a classic algorithm to cluster data together. Let's consider the following data points:



Instance	X1	X2
A	1	2
B	2	2
C	2	1
D	-1	4
E	-2	-1
F	-1	-1

We going to create 2 clusters out of those data points. Each cluster has a centroid and we are calling μ_1 the position of the first cluster centroid and μ_2 the second one. We are going to initialize those centroids to be at the point A for μ_1 and at the point B for μ_2.

Question 1: What are all the points closer to μ_1 and all the points closer to μ_2 in Manhattan distance? The Manhattan distance is the L_1 norm $(d((x_1, y_1), (x_2, y_2)) = |x_1 - x_2| + |y_1 - y_2|)$. The points closer to μ_1 are said to belong to cluster c_1 and the points closer to μ_2 are said to belong to cluster c_2. To which cluster they initially belong to?

Solution 1:

Instance	Distance from μ_1	Distance from μ_2	cluster
A	0	1	1
B	1	0	2
C	2	1	2
D	4	5	1
E	6	7	1
F	5	6	1

Question 2: We now update the position of μ_1 and μ_2 as the mean position of the clusters. For the cluster c_i, the new position μ_i is such that

$$\mu_i = \begin{pmatrix} \frac{1}{|c_i|} \sum_{j \in c_i} X_1^{(j)} \\ \frac{1}{|c_i|} \sum_{j \in c_i} X_2^{(j)} \end{pmatrix} \tag{24}$$

where $|c_i|$ is the number of instances in the cluster c_i. What is the new value for μ_1 and μ_2?

Solution 2:

$$\mu_1 = \begin{pmatrix} -0.75 \\ 1 \end{pmatrix} \text{ and } \mu_2 = \begin{pmatrix} 2 \\ 1.5 \end{pmatrix} \tag{25}$$

Question 3: Repeat the steps in question 1 and question 2 until μ_1 and μ_2 have converged to fixed points. Draw the all the points on a diagram and show the final clusters.

Solution 3: The algorithm converges after 2 iteration

$$\mu_1 = \begin{pmatrix} -4/3 \\ 2/3 \end{pmatrix} \text{ and } \mu_2 = \begin{pmatrix} 4/3 \\ 4/3 \end{pmatrix} \tag{26}$$

A, B and C belong to c_2 and D, E and F belong to c_1.

3 Learning

3.1 The Bias-variance trade off

We assume that we can relate features x to a target y through the relation

$$y = f(x) + \epsilon \tag{27}$$

where f is a deterministic function and ϵ is a random noise $\epsilon \sim \mathcal{D}(0, \sigma)$. Machine learning is about finding an estimate $\hat{f}$ of f. Note that ϵ is independent of $\hat{f}(x)$. Lets look at the Mean Square Error metric (MSE). We have

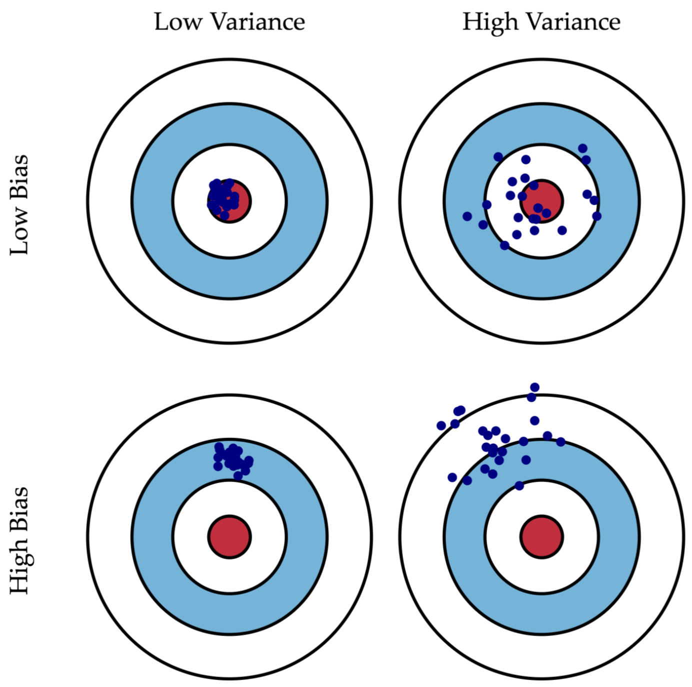
$$MSE \quad = \quad E\left[(y - \hat{f}(x))^2\right] \tag{28}$$

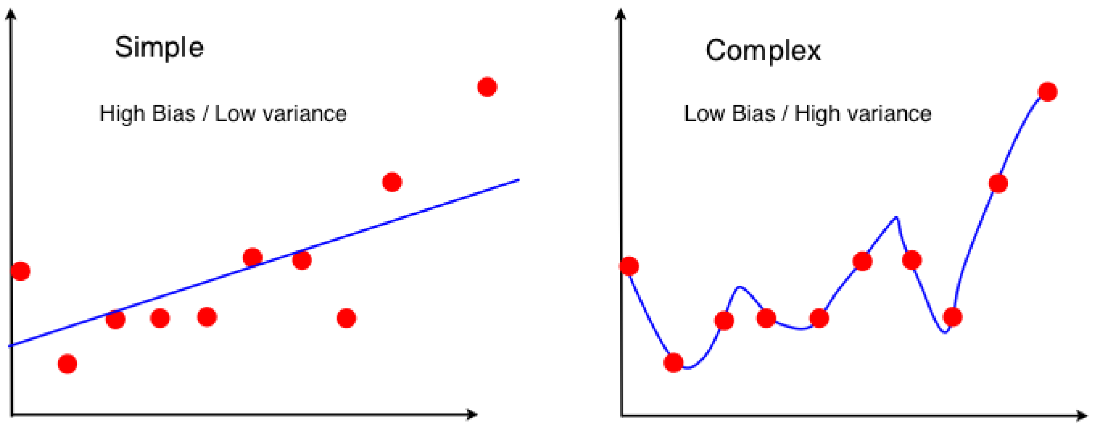
We are going to define

$$Var[\hat{f}(x)] = E\left[\hat{f}(x)^2\right] - E\left[\hat{f}(x)\right]^2 \tag{29}$$

and

$$Bias[\hat{f}(x)] = E\left[\hat{f}(x) - f(x)\right] \tag{30}$$





In machine learning, the biasvariance tradeoff (or dilemma) is the problem of simultaneously minimizing two sources of error that prevent supervised learning algorithms from generalizing beyond their training set. The bias is error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting). The variance is error from sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting) (Wikipedia).

Question 1: Prove that $MSE = \sigma^2 + Var[\hat{f}(x)] + Bias[\hat{f}(x)]^2$

Solution 1:

$$\begin{aligned} MSE &= E\left[(y - \hat{f}(x))^2\right] \\ &= E\left[y^2\right] + E\left[\hat{f}(x)^2\right] - 2E\left[y\hat{f}(x)\right] \\ &= Var[y] + E\left[y\right]^2 + Var[\hat{f}(x)] + E\left[\hat{f}(x)\right]^2 - 2E\left[y\hat{f}(x)\right] \end{aligned} \tag{31}$$

Because ϵ is independent of $\hat{f}(x)$ we have

$$\begin{aligned} E\left[y\hat{f}(x)\right] &= E\left[(f(x)+\epsilon)\hat{f}(x)\right] \\ &= E\left[f(x)\hat{f}(x)\right] + E\left[\epsilon\hat{f}(x)\right] \\ &= f(x)E\left[\hat{f}(x)\right] + E\left[\epsilon\right]E\left[\hat{f}(x)\right] \\ &= f(x)E\left[\hat{f}(x)\right] \end{aligned} \tag{32}$$

also

$$E\left[y\right] = f(x) \tag{33}$$

and

$$Var[y] = \sigma^2 \tag{34}$$

Therefore

$$\begin{aligned} MSE &= \sigma^2 + Var[\hat{f}(x)] + f(x)^2 + E\left[\hat{f}(x)\right]^2 - 2f(x)E\left[\hat{f}(x)\right] \\ &= \sigma^2 + Var[\hat{f}(x)] + \left(f(x) - E\left[\hat{f}(x)\right]\right)^2 \\ &= \sigma^2 + Var[\hat{f}(x)] + E\left[f(x) - \hat{f}(x)\right]^2 \\ &= \sigma^2 + Var[\hat{f}(x)] + Bias[\hat{f}(x)]^2 \end{aligned} \tag{35}$$

3.2 The right subset of the data

Let's assume that a bank is issuing credit cards to a set of N customers. To buy ($B = 1$) something with this credit card, the customers need first to activate the card ($A = 1$). The bank keeps track of all the people it issued a credit card to. The bank records if the people bought something ($B = 1$) or not ($B = 0$) and it records if the people activated ($A = 1$) or not ($A = 0$). The bank has some data about the customers captured by a set of features $\mathbf{X} = \{X_1, X_2, \ldots, X_m\}$.

The bank is now asking you (the company's data scientist) to build some predictive models.

Question 1: The bank wants to understand the propensity for a customer to buy something with the credit card if it sends one. How would you subset the data to build a classifier that estimates $p(B = 1|\mathbf{X})$ and what target would you choose?

Solution 1: You keep all the instances and you target the binary flag $B = \{0, 1\}$.

Question 2: It also wants to estimate the propensity for a customer to buy something with the credit card if the customer has activated it. How would you subset the data to build a classifier that estimates $p(B = 1|\mathbf{X}, A = 1)$ and what target would you choose?

Solution 2: You keep only the instances where $A = 1$ and you target the binary flag $B = \{0, 1\}$.

Question 3: It now wants to understand the probability for a customer to activate the credit card. How would you subset the data to train a classifier that estimates $p(A = 1|\mathbf{X})$ and what target would you choose?

Solution 3: You keep all the instances and you target the binary flag $A = \{0, 1\}$.

Question 4: You have build 2 supervised learning models that estimate $p(B = 1|\mathbf{X}, A = 1)$ and $p(A = 1|\mathbf{X})$ respectively. How could you use the prediction outputs of those models to estimate $p(B = 1|\mathbf{X})$ (proof required)?

Solution 4: We have

$$p(B = 1|\mathbf{X}) = p(B = 1, A = 1|\mathbf{X}) + p(B = 1, A = 0|\mathbf{X}). \tag{36}$$

We know that $p(B = 1, A = 0|\mathbf{X}) = 0$ because it is an impossible event. We have

$$\begin{aligned} p(B = 1|\mathbf{X}) &= p(B = 1, A = 1|\mathbf{X}) \\ &= p(B = 1|\mathbf{X}, A = 1)p(A = 1|\mathbf{X}) \end{aligned} \tag{37}$$

Therefore we just need to multiply the 2 predictions to estimate $p(B = 1|\mathbf{X})$.

3.3 Feature selection with LASSO

Regularization is a process to prevent overfitting. In linear regression or logistic regression, it is usually achieved by adding a controlling parameter to the loss function

$$\epsilon(\beta) = \|\mathbf{y} - \mathbf{X}\beta\|^2 + f(\beta). \tag{38}$$

The two main regularization schemes are Ridge regression

$$f(\beta) = \lambda \left\|\beta\right\|_2^2 \tag{39}$$

and Lasso regression

$$f(\beta) = \lambda \left\|\beta\right\|_1 \tag{40}$$

where λ is a free parameter. Although both Ridge and Lasso tend to reduce the magnitudes of the components of β, Lasso can zero out some less relevant features. As such Lasso can be used as a feature reduction process. Let's consider

$$\epsilon(\beta) = \left\|\mathbf{y} - \mathbf{X}\beta\right\|^2 + \lambda \left\|\beta\right\|_1 . \tag{41}$$

Remember that

$$\nabla_\beta \left\|\beta\right\|_1 = \frac{\beta}{\left\|\beta\right\|_1} = \text{sign}(\beta) \tag{42}$$

Question 1: Rederive the Normal equations such we have β as function of $\mathbf{y}$, $\mathbf{X}$, λ and sign(β)

Solution 1: We have

$$\beta = (\mathbf{X}^T\mathbf{X})^{-1} \left(\mathbf{X}^T\mathbf{y} - \lambda\text{sign}(\beta)\right) \tag{43}$$

Question 2: Let's consider the orthonormal case

$$\mathbf{X}^T\mathbf{X} = \mathbf{I}. \tag{44}$$

Write the equation for the j^{th} component β_j of the vector β.

Solution 2: We have

$$\beta_j = \mathbf{X}^T\mathbf{y}\,|_j - \lambda\text{sign}(\beta_j) \tag{45}$$

Question 3: Explain why above a certain value of λ, β_j becomes 0 and stay at 0.

Solution 3: We can see that there exist a value of λ that drives β_j to 0. For larger value of λ, sign(β_j) would change keeping β_j at 0.

3.4 The Area under the Receiver operating characteristic curve

The Area under the Receiver operating characteristic curve is (ROC AUC) is a measure of performance for binary classifiers. Consider the following predictions of a classifier and the true associated outcomes:

Truth	prediction
1	0.8
0	0.5
1	0.6
0	0.6
0	0.4
0	0.1
1	0.9
1	0.4
1	0.7

The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. Recall that the true positive rate is computed as

$$TPR = \frac{\text{Number of true positive}}{\text{Number of true positive} + \text{Number of false negative}} \tag{46}$$

and the false positive rate as

$$FPR = \frac{\text{Number of false positive}}{\text{Number of false positive} + \text{Number of true negative}} \tag{47}$$

Consider the possible thresholds being $\{0.85, 0.65, 0.55, 0.45, 0.35, 0.05\}$.

Question 1: for each of those thresholds in a descending order, compute the FPR and the TPR and plot on a graph TPR vs FPR

Solution 1:

- The FPR: $\{0, 0, 0.25, 0.5, 0.75, 1\}$
- The TPR: $\{0.2, 0.6, 0.8, 0.8, 1, 1\}$

Question 2: Estimate the area under the curve by using the trapezoidal rule of integration.

Solution 2: $AUC \simeq 0.8499$