1 Short Questions [20 pts]

Are the following statements True/False? Explain your reasoning in only 1 sentence.

1. Density estimation (using say, the kernel density estimator) can be used to perform classification.

   True: Estimate the joint density $P(Y, X)$, then use it to calculate $P(Y|X)$.

2. The correspondence between logistic regression and Gaussian Naïve Bayes (with identity class covariances) means that there is a one-to-one correspondence between the parameters of the two classifiers.

   False: Each LR model parameter corresponds to a whole set of possible GNB classifier parameters, there is no one-to-one correspondence because logistic regression is discriminative and therefore doesn't model $P(X)$, while GNB does model $P(X)$.

3. The training error of 1-NN classifier is 0.

   True: Each point is its own neighbor, so 1-NN classifier achieves perfect classification on training data.

4. As the number of data points grows to infinity, the MAP estimate approaches the MLE estimate for all possible priors. In other words, given enough data, the choice of prior is irrelevant.

   False: A simple counterexample is the prior which assigns probability 1 to a single choice of parameter θ.

5. Cross validation can be used to select the number of iterations in boosting; this procedure may help reduce overfitting.

   True: The number of iterations in boosting controls the complexity of the model, therefore, a model selection procedure like cross validation can be used to select the appropriate model complexity and reduce the possibility of overfitting.

6. The kernel density estimator is equivalent to performing kernel regression with the value $Y_i = \frac{1}{n}$ at each point X_i in the original data set.

   False: Kernel regression predicts the value of a point as the weighted average of the values at nearby points, therefore if all of the points have the same value, then kernel regression will predict a constant (in this case, $\frac{1}{n}$) for all values.

7. We learn a classifier f by boosting weak learners h. The functional form of f's decision boundary is the same as h's, but with different parameters. (e.g., if h was a linear classifier, then f is also a linear classifier).

   False: For example, the functional form of a decision stump is a single axis-aligned split of the input space, but the functional form of the boosted classifier is linear combinations of decision stumps which can form a more complex (piecewise linear) decision boundary.

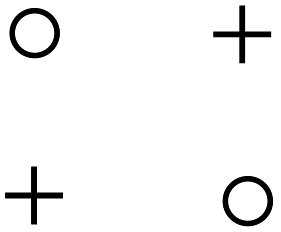
8. The depth of a learned decision tree can be larger than the number of training examples used to create the tree.

   False: Each split of the tree must correspond to at least one training example, therefore, if there are n training examples, a path in the tree can have length at most n.

   Note: There is a pathological situation in which the depth of a learned decision tree can be larger than number of training examples n - if the number of features is larger than n and there exist training examples which have same feature values but different labels. Points have been given if you answered true and provided this explanation.

For the following problems, circle the correct answers:

1. Consider the following data set:

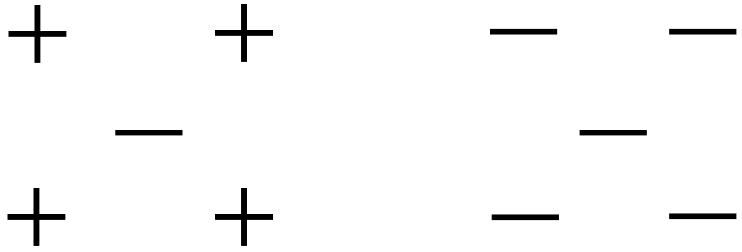


   Circle all of the classifiers that will achieve zero training error on this data set. (You may circle more than one.)

   (a) Logistic regression
   (b) SVM (quadratic kernel)
   (c) Depth-2 ID3 decision trees
   (d) 3-NN classifier

   Solution: SVM (quad kernel) and Depth-2 ID3 decision trees

2. For the following dataset, circle the classifier which has larger Leave-One-Out Cross-validation error.



   a) 1-NN

   b) 3-NN

Solution: 1-NN since 1-NN CV err: 5/10, 3-NN CV err: 1/10

6. Decision trees (10 points)

Consider the following data set with three binary input attributes (A1, A2, and A3) and one binary output, y.

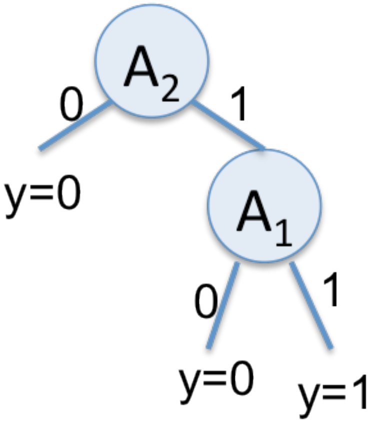
instance	A_1	A_2	A_3	y
1	1	0	0	0
2	1	0	1	0
3	0	1	0	0
4	1	1	1	1
5	1	1	0	1

Construct a decision tree to predict y given the inputs from this data using the ID3 algorithm that selects the variable at each level that maximizes the information gained.

7.1 What is the variable at the root of the tree. (5 pts)

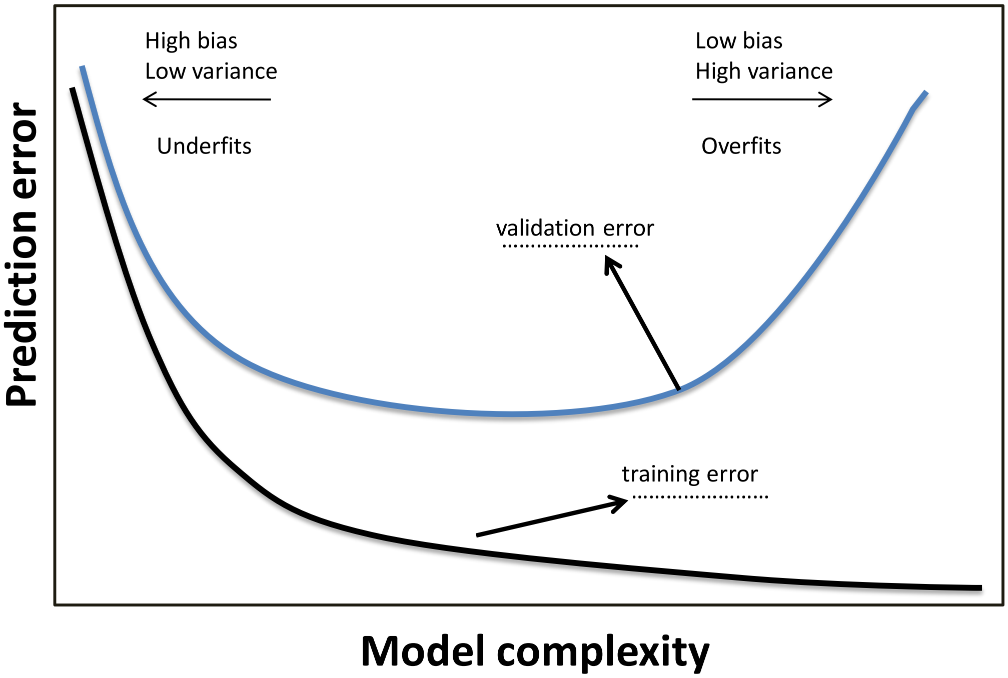
A2

7.2 Show the entire decision tree. (5 pts)



1 Training and Validation [8 Points]

The following figure depicts training and validation curves of a learner with increasing model complexity.



1. [**Points: 2 pts**] Which of the curves is more likely to be the training error and which is more likely to be the validation error? Indicate on the graph by filling the dotted lines.
2. [**Points: 4 pts**] In which regions of the graph are bias and variance low and high? Indicate clearly on the graph with four labels: "low variance", "high variance", "low bias", "high bias".
3. [**Points: 2 pts**] In which regions does the model overfit or underfit? Indicate clearly on the graph by labeling "overfit" and "underfit".

2 Bias and Variance [6 Points]

A set of data points is generated by the following process: $Y = w_0 + w_1 X + w_2 X^2 + w_3 X^3 + w_4 X^4 + \epsilon$, where X is a real-valued random variable and ϵ is a Gaussian noise variable. You use two models to fit the data:

Model 1: $Y = aX + b + \epsilon$

Model 2: $Y = w_0 + w_1 X^1 + ... + w_9 X^9 + \epsilon$

1. [**Points: 2 pts**] Model 1, when compared to Model 2 using a fixed number of training examples, has a *bias* which is:
   (a) Lower
   (b) Higher ★
   (c) The Same
2. [**Points: 2 pts**] Model 1, when compared to Model 2 using a fixed number of training examples, has a *variance* which is:
   (a) Lower ★
   (b) Higher
   (c) The Same
3. [**Points: 2 pts**] Given 10 training examples, which model is more likely to overfit the data?
   (a) Model 1
   (b) Model 2 ★

★ **SOLUTION:** Correct answers are indicated with a star next to them.

3 Experimental design [16 Points]

For each of the listed descriptions below, circle whether the experimental set up is *ok* or *problematic*. If you think it is problematic, briefly state **all** the problems with their approach:

1. [**Points: 4 pts**] A project team reports a low training error and claims their method is good.

   (a) Ok

   (b) Problematic ★

   ★ **SOLUTION:** Problematic because training error is an optimistic estimator of test error. Low training error does not tell much about the generalization performance of the model. To prove that a method is good they should report their error on independent test data.

2. [**Points: 4 pts**] A project team claimed great success after achieving 98 percent classification accuracy on a binary classification task where one class is very rare (e.g., detecting fraud transactions). Their data consisted of 50 positive examples and 5 000 negative examples.

   (a) Ok

   (b) Problematic ★

   ★ **SOLUTION:** Think of classifier which predicts everything as the majority class. The accuracy of that classifier will be 99%. Therefore 98% accuracy is not an impressive result on such an unbalanced problem.

3. [**Points: 4 pts**] A project team split their data into training and test. Using their training data and cross-validation, they chose the best parameter setting. They built a model using these parameters and their training data, and then report their error on test data.

   (a) Ok ★

   (b) Problematic

   ★ **SOLUTION:** OK.

4. [**Points: 4 pts**] A project team performed a feature selection procedure on the full data and reduced their large feature set to a smaller set. Then they split the data into test and training portions. They built their model on training data using several different model settings, and report the the best test error they achieved.

   (a) Ok

   (b) Problematic ★

   ★ **SOLUTION:** Problematic because:

   (a) Using the full data for feature selection will leak information from the test examples into the model. The feature selection should be done exclusively using training and validation data not on test data.

   (b) The best parameter setting should not be chosen based on the test error; this has the danger of overfitting to the test data. They should have used validation data and use the test data only in the final evaluation step.

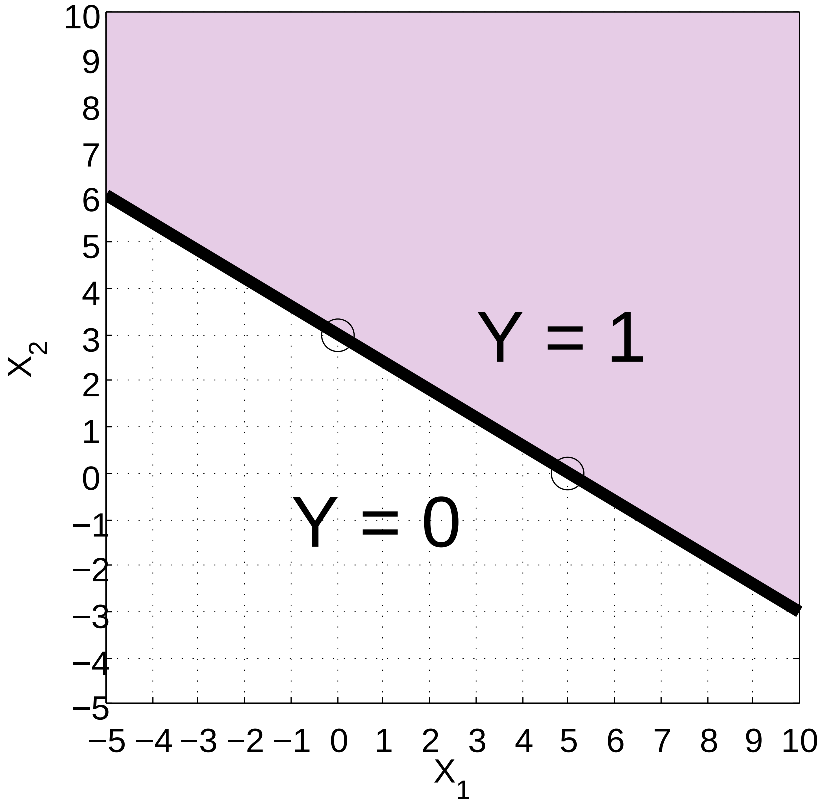
4 Logistic Regression [8 Points]

Suppose you are given the following classification task: predict the target $Y \in \{0, 1\}$ given two real valued features $X_1 \in \mathbb{R}$ and $X_2 \in \mathbb{R}$. After some training, you learn the following decision rule:

$$\textbf{Predict } Y = 1 \textbf{ iff } w_1X_1 + w_2X_2 + w_0 \geq 0 \textbf{ and } Y = 0 \textbf{ otherwise}$$

where $w_1 = 3$, $w_2 = 5$, and $w_0 = -15$.

1. [**Points: 6 pts**] Plot the decision boundary and label the region where we would predict $Y = 1$ and $Y = 0$.



★ **SOLUTION:** See above figure.

2. [**Points: 2 pts**] Suppose that we learned the above weights using logistic regression. Using this model, what would be our prediction for $P(Y = 1 \mid X_1, X_2)$? (You may want to use the sigmoid function $\sigma(x) = 1/(1 + \exp(-x))$.)

$$\mathbf{P}(Y = 1 \mid X_1, X_2) =$$

★ **SOLUTION:**

$$\mathbf{P}(Y = 1 \mid X_1, X_2) = \frac{1}{1 + \exp^{-(3X_1 + 5X_2 - 15)}}$$

5 Regression with Regularization [10 Points]

You are asked to use regularized linear regression to predict the target $Y \in \mathbb{R}$ from the eight-dimensional feature vector $X \in \mathbb{R}^8$. You define the model $Y = w^T X$ and then you recall from class the following three objective functions:

$$\min_w \sum_{i=1}^{n} \left(y_i - w^T x_i\right)^2 \tag{5.1}$$

$$\min_w \sum_{i=1}^{n} \left(y_i - w^T x_i\right)^2 \quad + \quad \lambda \sum_{j=1}^{8} w_j^2 \tag{5.2}$$

$$\min_w \sum_{i=1}^{n} \left(y_i - w^T x_i\right)^2 \quad + \quad \lambda \sum_{j=1}^{8} |w_j| \tag{5.3}$$

1. **[Points: 2 pts]** Circle regularization terms in the objective functions above.

   ★ **SOLUTION:** The regularization term in 5.2 is $\lambda \sum_{j=1}^{8} w_j^2$ and in 5.3 is $\lambda \sum_{j=1}^{8} |w_j|$.

2. **[Points: 2 pts]** For large values of λ in objective 5.2 the bias would:

   (a) increase ★
   (b) decrease
   (c) remain unaffected

3. **[Points: 2 pts]** For large values of λ in objective 5.3 the variance would:

   (a) increase
   (b) decrease ★
   (c) remain unaffected

4. **[Points: 4 pts]** The following table contains the weights learned for all three objective functions (not in any particular order):

	Column A	Column B	Column C
w_1	0.60	0.38	0.50
w_2	0.30	0.23	0.20
w_3	-0.10	-0.02	0.00
w_4	0.20	0.15	0.09
w_5	0.30	0.21	0.00
w_6	0.20	0.03	0.00
w_7	0.02	0.04	0.00
w_8	0.26	0.12	0.05

Beside each objective write the appropriate column label (A, B, or C):

- Objective 5.1: ★ **Solution:** A
- Objective 5.2: ★ **Solution:** B
- Objective 5.3: ★ **Solution:** C

6 Controlling Overfitting [6 Points]

We studied a number of methods to control overfitting for various classifiers. Below, we list several classifiers and actions that might affect their bias and variance. Indicate (by circling) how the bias and variance change in response to the action:

1. [**Points: 2 pts**] Reduce the number of leaves in a decision tree:

   ★ **SOLUTION:**

Bias	Variance
Decrease	Decrease ★
★ Increase	Increase
No Change	No Change

2. [**Points: 2 pts**] Increase k in a k-nearest neighbor classifier:

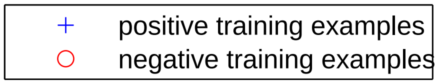
Bias	Variance
Decrease	Decrease ★
★ Increase	Increase
No Change	No Change

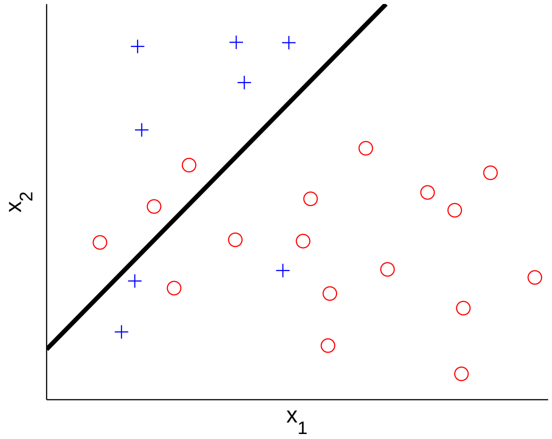
3. [**Points: 2 pts**] Increase the number of training examples in logistic regression:

Bias	Variance
Decrease	Decrease ★
Increase	Increase
★ No Change	No Change

7 Decision Boundaries [12 Points]

The following figures depict decision boundaries of classifiers obtained from three learning algorithms: decision trees, logistic regression, and nearest neighbor classification (in some order). Beside each of the three plots, write the **name** of the learning algorithm and the **number of mistakes** it makes on the training data.

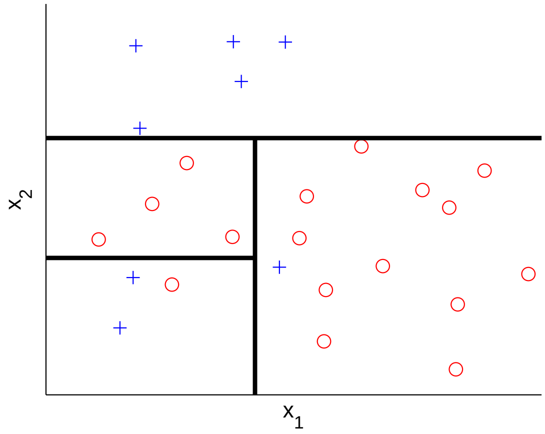




[Points: 4 pts]

Name: ★ Logistic regression

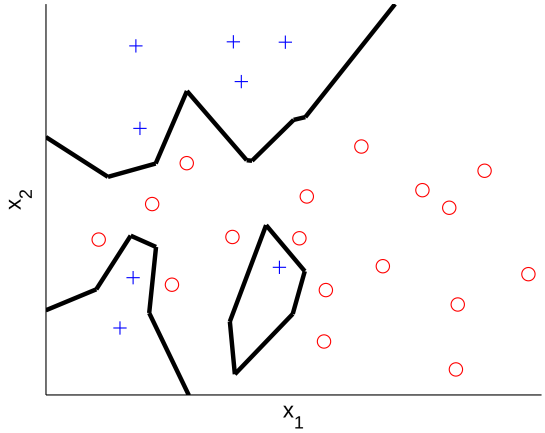
Number of mistakes: ★ 6



[Points: 4 pts]

Name: ★ Decision tree

Number of mistakes: ★ 2



[Points: 4 pts]

Name: ★ k-nearest neighbor

Number of mistakes: ★ 0

8 k-Nearest Neighbor Classifiers [6 Points]

In Fig. 1 we depict training data and a single test point for the task of classification given two continuous attributes X_1 and X_2. For each value of k, circle the label predicted by the k-nearest neighbor classifier for the depicted test point.

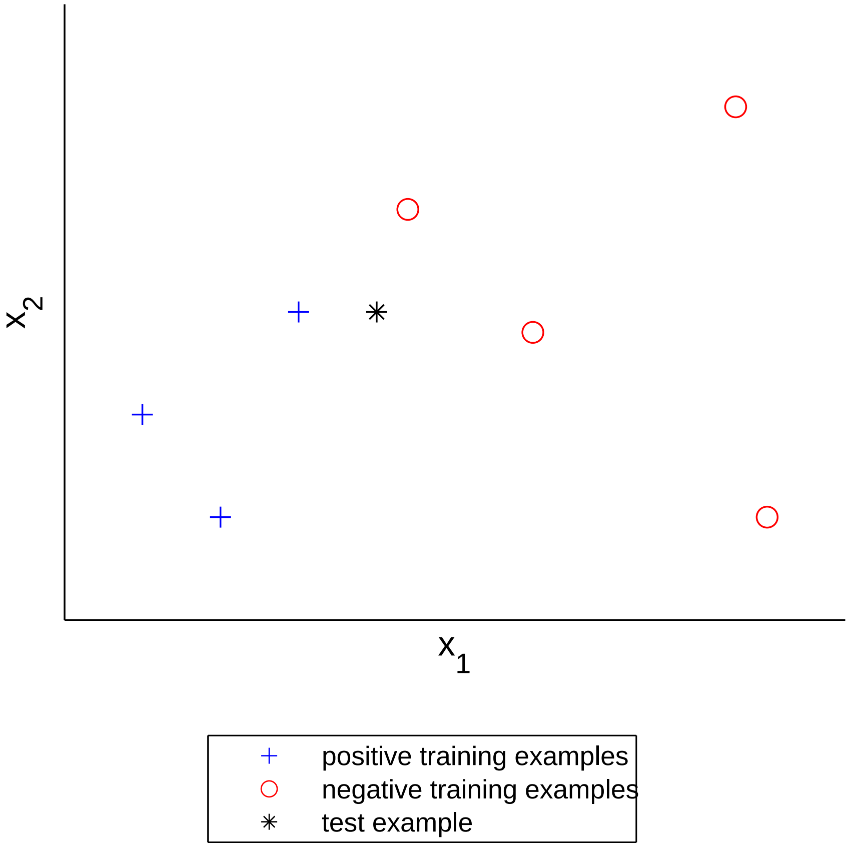


Figure 1: Nearest neighbor classification

1. **[Points: 2 pts]** Predicted label for $k = 1$:

   (a) positive ★ **(b)** negative

2. **[Points: 2 pts]** Predicted label for $k = 3$:

   (a) positive **(b)** negative ★

3. **[Points: 2 pts]** Predicted label for $k = 5$:

   (a) positive ★ **(b)** negative

9 Decision Trees [16 Points]

Suppose you are given six training points (listed in Table 1) for a classification problem with two binary attributes X_1, X_2, and three classes $Y \in \{1, 2, 3\}$. We will use a decision tree learner based on information gain.

X_1	X_2	Y
1	1	1
1	1	1
1	1	2
1	0	3
0	0	2
0	0	3

Table 1: Training data for the decision tree learner.

1. **[Points: 12 pts]** Calculate the information gain for both X_1 and X_2. You can use the approximation $\log_2 3 \approx 19/12$. Report information gains as fractions or as decimals with the precision of three decimal digits. Show your work and circle your final answers for IG(X_1) and IG(X_2).

★ **SOLUTION:** The equation for information gain, entropy, and conditional entropy are given by (respectively):

$$\begin{aligned}
\mathsf{IG}(X) &= \mathsf{H}(Y) - \mathsf{H}(Y \mid X) \\
\mathsf{H}(X) &= -\sum_x \mathbf{P}(X = x) \log_2 \mathbf{P}(X = x) \\
\mathsf{H}(Y \mid X) &= \sum_x \mathbf{P}(X = x) \sum_y \mathbf{P}(Y = y \mid X = x) \log_2 \mathbf{P}(Y = y \mid X = x)
\end{aligned}$$

Using these equations we can derive the information gain for each split. First we compute the entropy $\mathsf{H}(Y)$:

$$\begin{aligned}
\mathsf{H}(Y) &= -\sum_{y_i=1}^{n=3} \mathbf{P}(Y = y_i) \log_2 \mathbf{P}(Y = y_i) \\
&= -\sum_{y_i=1}^{n=3} \frac{1}{3} \log_2 \frac{1}{3} = \log_2 3 \approx \frac{19}{12}
\end{aligned}$$

For the X_1 split we compute the conditional entropy:

$$\begin{aligned}
\mathsf{H}(Y \mid X_1) &= -\mathbf{P}(X_1 = 0) \sum_{y_i=1}^{n=3} \mathbf{P}(Y = y_i \mid X_1 = 0) \log_2 \mathbf{P}(Y = y_i \mid X_1 = 0) \quad + \\
&\quad -\mathbf{P}(X_1 = 1) \sum_{y_i=1}^{n=3} \mathbf{P}(Y = y_i \mid X_1 = 1) \log_2 \mathbf{P}(Y = y_i \mid X_1 = 1) \\
&= -\left[\frac{2}{6}\left(\frac{0}{2}\log_2 \frac{0}{2} + \frac{1}{2}\log_2 \frac{1}{2} + \frac{1}{2}\log_2 \frac{1}{2}\right) + \frac{4}{6}\left(\frac{2}{4}\log_2 \frac{2}{4} + \frac{1}{4}\log_2 \frac{1}{4} + \frac{1}{4}\log_2 \frac{1}{4}\right)\right] \\
&= -\left(-\frac{2}{6} - 1\right) \\
&= \frac{4}{3}
\end{aligned}$$

Similarly for the X_2 split we compute the conditional entropy:

$$\begin{aligned}
\mathsf{H}(Y \mid X_2) &= -\mathbf{P}(X_2=0)\sum_{y_i=1}^{n=3}\mathbf{P}(Y=y_i \mid X_2=0)\log_2\mathbf{P}(Y=y_i \mid X_2=0) \quad + \\
&\quad -\mathbf{P}(X_2=1)\sum_{y_i=1}^{n=3}\mathbf{P}(Y=y_i \mid X_2=1)\log_2\mathbf{P}(Y=y_i \mid X_2=1) \\
&= -\left[\frac{3}{6}\left(\frac{0}{3}\log_2\frac{0}{3}+\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}\right)+\frac{3}{6}\left(\frac{2}{3}\log_2\frac{2}{3}+\frac{1}{3}\log_2\frac{1}{3}+\frac{0}{3}\log_2\frac{0}{3}\right)\right] \\
&\approx -\left(\frac{2}{3}-\frac{19}{12}\right) \\
&= \frac{11}{12}
\end{aligned}$$

The final information gain for each split is then:

$$\begin{aligned}
\mathsf{IG}(X_1) = \mathsf{H}(Y) - \mathsf{H}(Y \mid X_1) &\approx \frac{19}{12}-\frac{4}{3}=\frac{3}{12}=\frac{1}{4} \\
\mathsf{IG}(X_2) = \mathsf{H}(Y) - \mathsf{H}(Y \mid X_2) &\approx \frac{19}{12}-\frac{11}{12}=\frac{8}{12}=\frac{2}{3}
\end{aligned}$$

2. **[Points: 4 pts]** Report which attribute is used for the first split. Draw the decision tree resulting from using this split alone. Make sure to label the split attribute, which branch is which, and what the predicted label is in each leaf. How would this tree classify an example with $X_1 = 0$ and $X_2 = 1$?

★ **SOLUTION:** Since the information gain of X_2 is greater than X_1's information gain, we choose to split on X_2. See the resulted decision tree in Fig. 2. An example with $X_1 = 0$ and $X_2 = 1$ will be classified as $Y = 1$ on this tree since $X_2 = 1$.

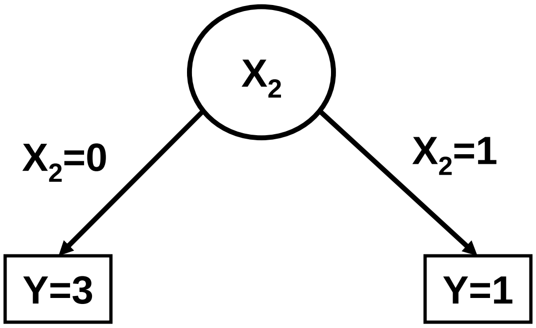


Figure 2: The decision tree for question 9.2

3 Logistic Regression [18 pts]

We consider here a discriminative approach for solving the classification problem illustrated in Figure 1.

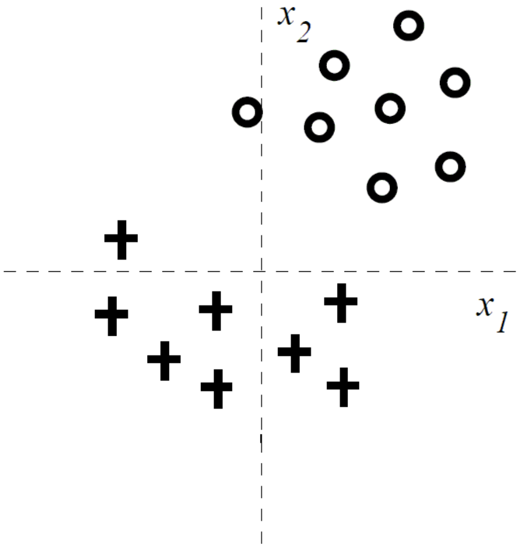


Figure 1: The 2-dimensional labeled training set, where '+' corresponds to class y=1 and 'O' corresponds to class $y = 0$.

1. We attempt to solve the binary classification task depicted in Figure 1 with the simple linear logistic regression model

$$P(y = 1|\vec{x}, \vec{w}) = g(w_0 + w_1 x_1 + w_2 x_2) = \frac{1}{1 + exp(-w_0 - w_1 x_1 - w_2 x_2)}.$$

Notice that the training data can be separated with *zero* training error with a linear separator.

Consider training *regularized* linear logistic regression models where we try to maximize

$$\sum_{i=1}^{n} \log\ (P(y_i|x_i, w_0, w_1, w_2)) - Cw_j^2$$

for very large C. The regularization penalties used in penalized conditional log-likelihood estimation are $-Cw_j^2$, where $j = \{0, 1, 2\}$. In other words, only one of the parameters is regularized in each case. Given the training data in Figure 1, how does the training error change with regularization of each parameter w_j? State whether the training error increases or stays the same (zero) for each w_j for very large C. Provide a brief justification for each of your answers.

(a) By regularizing w_2 [**2 pts**]

SOLUTION: Increases. When we regularize w_2, the resulting boundary can rely less and less on the value of x_2 and therefore becomes more vertical. For very large C, the training error increases as there is no good linear vertical separator of the training data.

(b) By regularizing w_1 [**2 pts**]

SOLUTION: Remains the same. When we regularize w_1, the resulting boundary can rely less and less on the value of x_1 and therefore becomes more horizontal and the training data can be separated with *zero* training error with a horizontal linear separator.

(c) By regularizing w_0 [**2 pts**]

SOLUTION: Increases. When we regularize w_0, then the boundary will eventually go through the origin (bias term set to zero). Based on the figure, we can *not* find a linear boundary through the origin with *zero* error. The best we can get is one error.

2. If we change the form of regularization to L1-norm (absolute value) and regularize w_1 and w_2 only (but not w_0), we get the following penalized log-likelihood

$$\sum_{i=1}^{n} \log P(y_i|x_i, w_0, w_1, w_2) - C(|w_1| + |w_2|).$$

Consider again the problem in Figure 1 and the same linear logistic regression model $P(y = 1|\vec{x}, \vec{w}) = g(w_0 + w_1x_1 + w_2x_2)$.

(a) [**3 pts**] As we increase the regularization parameter C which of the following scenarios do you expect to observe? (Choose only one) Briefly explain your choice:

( ) First w_1 will become 0, then w_2.

( ) First w_2 will become 0, then w_1.

( ) w_1 and w_2 will become zero simultaneously.

( ) None of the weights will become exactly zero, only smaller as C increases.

SOLUTION: First w_1 will become 0, then w_2.

The data can be classified with zero training error and therefore also with high log-probability by looking at the value of x_2 alone, i.e. making w_1 = 0. Initially we might prefer to have a non-zero value for w_1 but it will go to zero rather quickly as we increase regularization. Note that we pay a regularization penalty for a non-zero value of w_1 and if it does not help classification why would we pay the penalty? Also, the absolute value regularization ensures that w_1 will indeed go to *exactly* zero. As C increases further, even w_2 will eventually become zero. We pay higher and higher cost for setting w_2 to a non-zero value. Eventually this cost overwhelms the gain from the log-probability of labels that we can achieve with a non-zero w_2.

(b) [**3 pts**] For very large C, with the same L1-norm regularization for w_1 and w_2 as above, which value(s) do you expect w_0 to take? Explain briefly. (Note that the number of points from each class is the same.) (You can give a range of values for w_0 if you deem necessary).

SOLUTION: For very large C, we argued that both w_1 and w_2 will go to zero. Note that when w_1 = w_2 = 0, the log-probability of labels becomes a finite value, which is equal to n log(0.5), i.e. w_0 = 0. In other words, $P(y = 1|\vec{x}, \vec{w})$=$P(y = 0|\vec{x}, \vec{w})$=$0.5$. We expect so because the number of elements in each class is the same and so we would like to predict each one with the same probability, and w_0=0 makes $P(y = 1|\vec{x}, \vec{w})$=$0.5$.

(c) [**3 pts**] Assume that we obtain more data points from the '+' class that corresponds to y=1 so that the class labels become unbalanced. Again for very large C, with the same L1-norm regularization for w_1 and w_2 as above, which value(s) do you expect w_0 to take? Explain briefly. (You can give a range of values for w_0 if you deem necessary).

SOLUTION: For very large C, we argued that both w_1 and w_2 will go to zero. With unbalanced classes where the number of '+' labels are greater than that of 'o' labels, we want to have $P(y = 1|\vec{x}, \vec{w}) > P(y = 0|\vec{x}, \vec{w})$. For that to happen the value of w_0 should be greater than zero which makes $P(y = 1|\vec{x}, \vec{w}) > 0.5$.