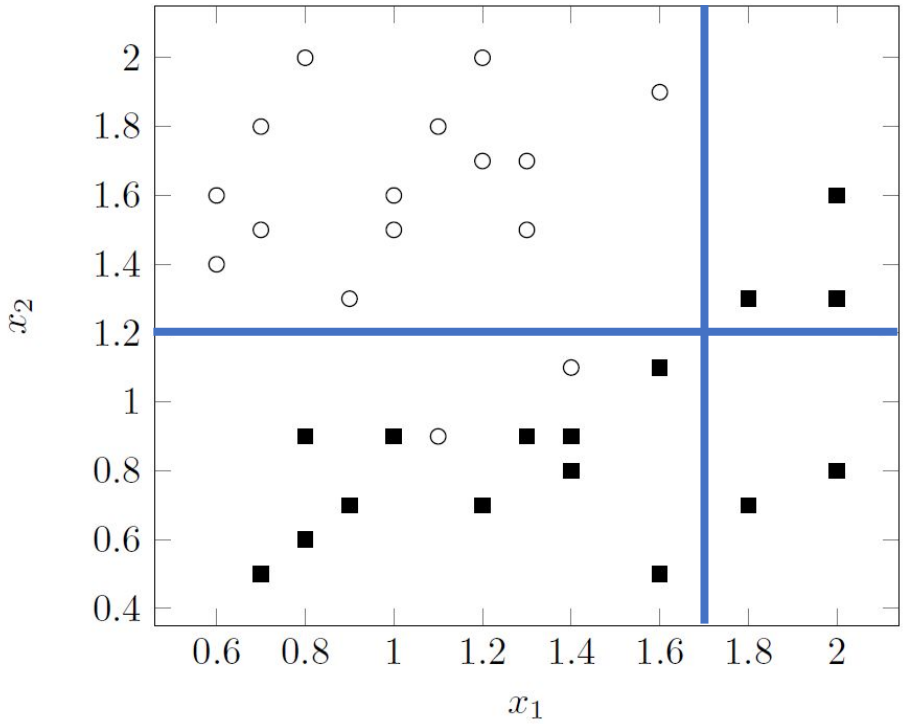
Practice Midterm Solutions

CSCI 567 TAs

Oct 2019

Decision Trees

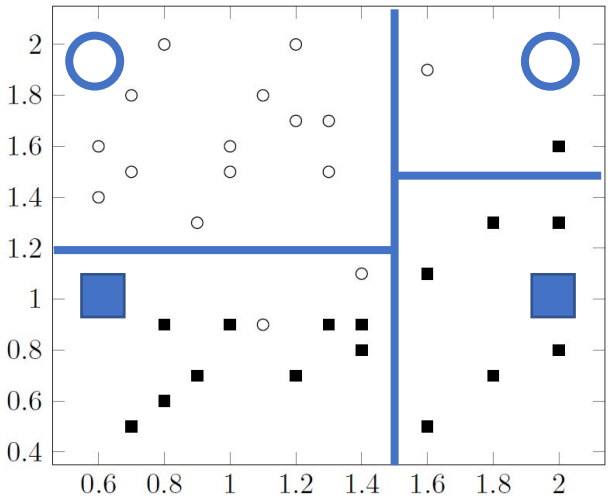
Decision Tree



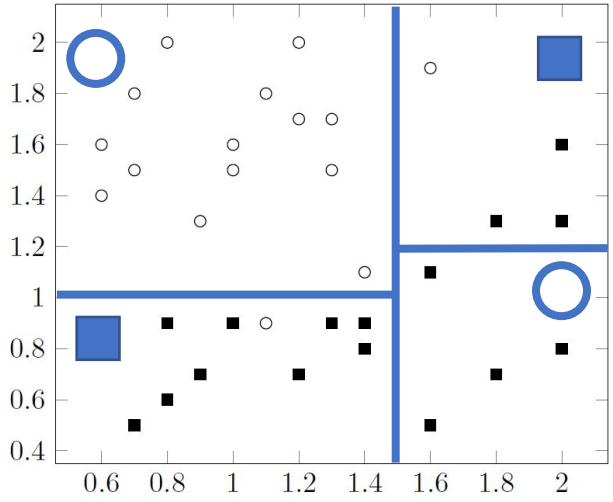
1. Suppose we use the data as training data, which of the following decision trees has the smallest training error?

Decision Tree

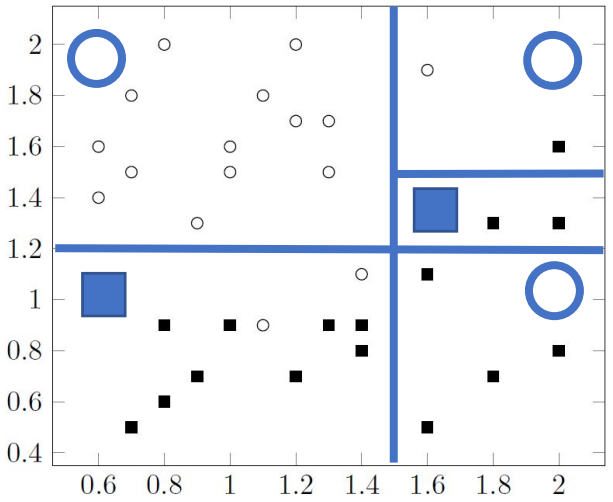
2. Suppose we use the data as validation data and apply reduced-error pruning to the decision tree above, pruning which of the following node will yield the best validation accuracy?



Prune node C



Prune node D



Prune node G

Naive Bayes

Naive Bayes

Suppose we are given the following data, where $A, B, C \in \{0, 1\}$ are random variables, and y is a binary output whose value we want to predict.

A	B	C	y
0	0	1	0
0	1	0	0
1	1	0	0
0	0	1	1
1	1	1	1
1	0	0	1
1	1	0	1

How would a naive Bayes classifier predict y if the input is $\{A = 0, B = 0, C = 1\}$

Naive Bayes

Suppose we are given the following data, where $A, B, C \in \{0, 1\}$ are random variables, and y is a binary output whose value we want to predict.

A	B	C	y
0	0	1	0
0	1	0	0
1	1	0	0
0	0	1	1
1	1	1	1
1	0	0	1
1	1	0	1

How would a naive Bayes classifier predict y if the input is $\{A = 0, B = 0, C = 1\}$

Answer: y = 1

$$P(y = 0|A = 0, B = 0, C = 1) = \frac{P(A = 0|y = 0)P(B = 0|y = 0)P(C = 1|y = 0)P(y = 0)}{P(A = 0, B = 0, C = 1)}$$

$$P(y = 1|A = 0, B = 0, C = 1) = \frac{P(A = 0|y = 1)P(B = 0|y = 1)P(C = 1|y = 1)P(y = 1)}{P(A = 0, B = 0, C = 1)}$$

Bayes' Theorem

Naive Bayes

Suppose we are given the following data, where $A, B, C \in \{0, 1\}$ are random variables, and y is a binary output whose value we want to predict.

A	B	C	y
0	0	1	0
0	1	0	0
1	1	0	0
0	0	1	1
1	1	1	1
1	0	0	1
1	1	0	1

How would a naive Bayes classifier predict y if the input is $\{A = 0, B = 0, C = 1\}$

Answer: y = 1

$$P(y = 0|A = 0, B = 0, C = 1) = \frac{P(A = 0|y = 0)P(B = 0|y = 0)P(C = 1|y = 0)P(y = 0)}{P(A = 0, B = 0, C = 1)}$$

$$P(y = 1|A = 0, B = 0, C = 1) = \frac{P(A = 0|y = 1)P(B = 0|y = 1)P(C = 1|y = 1)P(y = 1)}{P(A = 0, B = 0, C = 1)}$$

Naive Bayes

Suppose we are given the following data, where $A, B, C \in \{0, 1\}$ are random variables, and y is a binary output whose value we want to predict.

A	B	C	y
0	0	1	0
0	1	0	0
1	1	0	0
0	0	1	1
1	1	1	1
1	0	0	1
1	1	0	1

How would a naive Bayes classifier predict y if the input is $\{A = 0, B = 0, C = 1\}$

Answer: y = 1

$$P(A = 0|y = 0)P(B = 0|y = 0)P(C = 1|y = 0)P(y = 0) = \frac{2}{63} = 0.0317$$

$$P(A = 0|y = 1)P(B = 0|y = 1)P(C = 1|y = 1)P(y = 1) = \frac{1}{28} = 0.0357$$

Kernel

Kernel

Which is *not* a valid kernel function, for samples x and y and kernels $k_1(\cdot,\cdot)$ and $k_2(\cdot,\cdot)$?

(a) $k(x,y) = 5$

(b) $k(x,y) = x + y$

(c) $k(x,y) = e^{x+y}$

(d) $k(x,y) = \langle x,y\rangle^3 + (\langle x,y\rangle + 1)^2$

Kernel

Which is *not* a valid kernel function, for samples x and y and kernels $k_1(\cdot,\cdot)$ and $k_2(\cdot,\cdot)$?

(a) $k(x,y) = 5 \qquad \phi(x) = \sqrt{5}$

(b) $k(x,y) = x + y$

(c) $k(x,y) = e^{x+y} \qquad \phi(x) = e^x$

(d) $k(x,y) = \langle x,y\rangle^3 + (\langle x,y\rangle + 1)^2 \quad k_1(x,y) + k_2(x,y), k(x,y)^d$ are kernel functions

B: you can consider the x=0, y=1 and show that the kernel matrix is not positive semi-definite.
(x_1 = 2, x_2 = -1)

Kernel

Suppose we apply the kernel trick with a kernel function k to the nearest neighbor algorithm (with L2 distance in the new feature space). What is the nearest neighbor of a new data point $\boldsymbol{x}$ from a training set $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$?

(a) $\arg\min_{\boldsymbol{x}_n \in S} \; k(\boldsymbol{x}_n, \boldsymbol{x}_n) + k(\boldsymbol{x}, \boldsymbol{x}) + 2k(\boldsymbol{x}_n, \boldsymbol{x})$

(b) $\arg\min_{\boldsymbol{x}_n \in S} \; k(\boldsymbol{x}_n, \boldsymbol{x})$

(c) $\arg\min_{\boldsymbol{x}_n \in S} \; (k(\boldsymbol{x}_n, \boldsymbol{x}) - k(\boldsymbol{x}, \boldsymbol{x}_n))^2$

(d) $\arg\min_{\boldsymbol{x}_n \in S} \; k(\boldsymbol{x}_n, \boldsymbol{x}_n) - 2k(\boldsymbol{x}_n, \boldsymbol{x})$

Kernel

Suppose we apply the kernel trick with a kernel function k to the nearest neighbor algorithm (with L2 distance in the new feature space). What is the nearest neighbor of a new data point $\boldsymbol{x}$ from a training set $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$?

(a) $\arg\min_{\boldsymbol{x}_n \in S} \; k(\boldsymbol{x}_n, \boldsymbol{x}_n) + k(\boldsymbol{x}, \boldsymbol{x}) + 2k(\boldsymbol{x}_n, \boldsymbol{x})$

(b) $\arg\min_{\boldsymbol{x}_n \in S} \; k(\boldsymbol{x}_n, \boldsymbol{x})$

(c) $\arg\min_{\boldsymbol{x}_n \in S} \; (k(\boldsymbol{x}_n, \boldsymbol{x}) - k(\boldsymbol{x}, \boldsymbol{x}_n))^2$

(d) $\arg\min_{\boldsymbol{x}_n \in S} \; k(\boldsymbol{x}_n, \boldsymbol{x}_n) - 2k(\boldsymbol{x}_n, \boldsymbol{x})$

D: $\arg\min_{x_n \in S} \|\phi(x) - \phi(x_n)\|^2 = \arg\min_{x_n \in S} k(x, x) - 2k(x, x_n) + k(x_n, x_n)$

Linear Regression

Linear Regression

In this problem, we prove that if we are using the Newton's method to solve the least square optimization problem, then it only takes one step to converge. Recall that the Newton's method update the parameters as follow:

$$w^{t+1} = w^t - H^{-1}\nabla L(w^t)$$

where $H = \nabla^2 L(w^t)$ is the Hessian matrix of the loss function, i.e., $H_{ij} = \frac{\partial}{\partial w_i \partial w_j} L(w^t)$.

(1) Find the Hessian of least square loss function: $L(w) = \frac{1}{2}\sum_{n=1}^{N}(w^T x_n - y_n)^2$.

(2) Show that given any w^0, after first iteration of Newton's method, we obtain the optimal $w^* = (X^T X)^{-1} X^T y$.

Linear Regression

(1) Find the Hessian of least square loss function: $L(w) = \frac{1}{2}\sum_{n=1}^{N}(w^T x_n - y_n)^2$.
Ans:

$$\frac{\partial}{\partial w_j} L(w) = \sum_{n=1}^{N} (w^T x_n - y_n) x_{nj}$$

$$\frac{\partial^2}{\partial w_i \partial w_j} L(w) = \sum_{n=1}^{N} x_{ni} x_{nj} = (X^T X)_{ij}$$

Therefore, $H = \nabla^2 L(w) = X^T X$.

Linear Regression

(2) Show that given any w^0, after first iteration of Newton's method, we obtain the optimal $w^* = (X^T X)^{-1} X^T y$.
Ans:

$$
\begin{aligned}
w^1 &= w^0 - H^{-1} \nabla L(w^0) \\
&= w^0 - H^{-1} X^T (X w^0 - y) \\
&= w^0 - w^0 + (X^T X)^{-1} X^T y \\
&= (X^T X)^{-1} X^T y
\end{aligned}
$$

k Nearest Neighbor

True or False?

k-Nearest Neighbor (kNN) classifier is a parametric machine learning model parameterized by k.

True or False?

k-Nearest Neighbor (kNN) classifier is a parametric machine learning model parameterized by k.

False. A parametric model assumes a finite set of of parameters, e.g., w in logistic regression that linearly combines an input vector. The hyperparameter k in kNN is not a parameter in this sense.

True or False?

Given a multi-class classification dataset,

$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_N, y_n)\}$ where $\mathbf{x}_n \in \mathcal{R}^D$

Suppose that each x paire is distinct,

one can always build a kNN classifier and a decision tree that both achieve the same training error.

True or False?

Given a multi-class classification dataset,
$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_N, y_n)\}$ where $\mathbf{x}_n \in \mathcal{R}^D$
Suppose that each x paire is distinct,
one can always build a kNN classifier and a decision tree that both achieve the same training error.

True.
You can always achieve 0 classification error on the training set using 1-NN.
Similarly, you can build a DT where each leaf has at most one data point that achieves 0 classification error on the training set.

True or False?

Given two binary classification datasets A and B with the same number of points and dimensionality. Dataset A and B have the same sizes of testing set, too. The best k of a kNN classifier on test set of Dataset A is also the best k when kNN is applied to Dataset B.

True or False?

Given two binary classification datasets A and B with the same number of points and dimensionality. Dataset A and B have the same sizes of testing set, too. The best k of a kNN classifier on test set of Dataset A is also the best k when kNN is applied to Dataset B.

False. The best k does not depend on the dimensionality of the dataset; rather, it is more problem-specific.

Logistic Regression

Reconciliation of Denotations

We have introduced two types of denotation for logistic regression in this course.

- In Theory Assignment 2, we represented the two classes using 0 and 1 and minimized **cross-entropy loss**;
- in lecture notes, we represented the two classes using -1 and 1 and minimized **logistic loss**.

In the following questions, we will ask you to show that the two denotations are equivalent.

Given a dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_N, y_n)\}$ where $\mathbf{x}_n \in \mathcal{R}^D$, and $y_n \in \{-1, 1\}$. We wish apply logistic regression to $\mathcal{D}$; in other words, we wish to learn a linear combination $\mathbf{w}$ that combines features so that we can apply a sigmoid activation $\sigma(z) = \frac{1}{1+\exp -z}$ to predict the probability of the output label.

4.1 Predictive model

Please express $P(y_n|\mathbf{x}_n, \mathbf{w})$ without if-else clause.

4.1 Predictive model

Please express $P(y_n|\mathbf{x}_n, \mathbf{w})$ without if-else clause.

$$P(y_n|\mathbf{x}_n, \mathbf{w}) = \begin{cases} \sigma(\mathbf{w}^T\mathbf{x}_n), & \text{if } y_n = 1 \\ 1 - \sigma(\mathbf{w}^T\mathbf{x}_n) = \sigma(-\mathbf{w}^T\mathbf{x}_n), & \text{if } y_n = -1 \end{cases}$$
$$= \sigma(y_n\mathbf{w}^T\mathbf{x}_n).$$

This is one of the reasons that some people prefers {-1, 1} denotation.

4.2 Cross-entropy Loss

Cross-entropy loss $H(q,p) := -\sum_{n=1}^{N} \big(q_n \ln p_n + (1-q_n)\ln(1-p_n)\big)$, where $q_n, p_n \in [0,1]$. In our problem, $p_n := P(y_n = 1|\mathbf{x}_n, \mathbf{w})$. Our goal is to minimize cross-entropy in our binary classification problem. Let $q_n = \frac{y_n+1}{2}$. Please derive $\nabla_{\mathbf{w}} H(q, P(y|\mathbf{x}, \mathbf{w}))$, express it with $q_n, \mathbf{x}_n$ and $\sigma(\cdot)$, and reduce it to the simplest form.

Write down each step.
Do not jump to the end result without justifications!

Suggestions

1. Familiarize yourself with derivation of sigmoid/softmax.
2. Apply chain rule carefully.
3. Apply the useful equalities introduced in the lecture.
4. The gradient operator is linear.
5. Divide the big monster into smaller chuncks.

$$\nabla_{\mathbf{w}} H(q, P(y|x, \mathbf{w})) = \nabla_{\mathbf{w}} - \sum_{n=1}^{N} \Big(q_n \ln p_n + (1 - q_n) \ln(1 - p_n) \Big)$$

Gradient is a linear operator.

$$= - \sum_{n=1}^{N} \nabla_{\mathbf{w}} \Big(q_n \ln \sigma(\mathbf{w}^T \mathbf{x}_n) + (1 - q_n) \ln \sigma(-\mathbf{w}^T \mathbf{x}_n) \Big)$$

$$= - \sum_{n=1}^{N} \Big(q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T \mathbf{x}_n) + (1 - q_n) \nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T \mathbf{x}_n) \Big)$$

Then work on the two terms individually.

$$-\sum_{n=1}^{N}\Big(\, q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T\mathbf{x}_n) + (1-q_n)\nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T\mathbf{x}_n)\Big)$$

$$\nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T\mathbf{x}_n) = (\nabla_{z_n} \ln \sigma(z_n))(\nabla_{\mathbf{w}}\sigma(\mathbf{w}^T\mathbf{x}_n))$$

$$\text{by } \nabla_z \sigma(z) = \sigma(z)(1-\sigma(z)) \qquad = (\frac{1}{\sigma(z_n)})(\sigma(z_n)(1-\sigma(z_n))\mathbf{x}_n)$$

$$-\sum_{n=1}^{N}\Big(\, q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T\mathbf{x}_n) + (1-q_n)\nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T\mathbf{x}_n)\Big)$$

$$\begin{aligned}
\nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T\mathbf{x}_n) &= (\nabla_{z_n} \ln \sigma(z_n))(\nabla_{\mathbf{w}}\sigma(\mathbf{w}^T\mathbf{x}_n)) \\
&= \Big(\frac{1}{\sigma(z_n)}\Big)(\sigma(z_n)(1-\sigma(z_n))\mathbf{x}_n) \\
&= (1-\sigma(z_n))\mathbf{x}_n.
\end{aligned}$$

$$-\sum_{n=1}^{N}\Big(\, q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T\mathbf{x}_n) + (1-q_n)\nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T\mathbf{x}_n)\Big)$$

$$\nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T\mathbf{x}_n) = \frac{1}{\sigma(-z_n)}(\sigma(-z_n)(1-\sigma(-z_n)))(-1)\mathbf{x}_n$$

$$\text{by } \nabla_z \sigma(z) = \sigma(z)(1-\sigma(z)) \quad = -(1-\sigma(-z_n)))\mathbf{x}_n$$

$$-\sum_{n=1}^{N}\Big(\, q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T\mathbf{x}_n) + (1-q_n)\nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T\mathbf{x}_n)\Big)$$

$$\begin{aligned}
\nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T\mathbf{x}_n) &= \frac{1}{\sigma(-z_n)}(\sigma(-z_n)(1-\sigma(-z_n)))(-1)\mathbf{x}_n \\
&= -(1-\sigma(-z_n)))\mathbf{x}_n \\
\sigma(-z) = 1-\sigma(z) \qquad &= -(\sigma(z_n))\mathbf{x}_n .
\end{aligned}$$

Combining the two terms

$$-\sum_{n=1}^{N} \left(q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T \mathbf{x}_n) + (1 - q_n) \nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T \mathbf{x}_n) \right)$$

Combining the two terms

$$\left(q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T \mathbf{x}_n) + (1 - q_n) \nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T \mathbf{x}_n) \right)$$

$$\begin{aligned}
& q_n \nabla_{\mathbf{w}} \ln \sigma(\mathbf{w}^T \mathbf{x}_n) + (1 - q_n) \nabla_{\mathbf{w}} \ln \sigma(-\mathbf{w}^T \mathbf{x}_n) \\
=& q_n (1 - \sigma(z_n)) \mathbf{x}_n + (1 - q_n)(-1)\sigma(z_n) \mathbf{x}_n \\
=& \left(q_n - q_n \sigma(z_n) - (\sigma(z_n) - q_n \sigma(z_n)) \right) \mathbf{x}_n \\
=& (q_n - \sigma(z_n)) \mathbf{x}_n
\end{aligned}$$

4.2 Cross-entropy Loss

Cross-entropy loss $H(q,p) := -\sum_{n=1}^{N} \big(q_n \ln p_n + (1-q_n)\ln(1-p_n)\big)$, where $q_n, p_n \in [0,1]$. In our problem, $p_n := P(y_n = 1|\mathbf{x}_n, \mathbf{w})$. Our goal is to minimize cross-entropy in our binary classification problem. Let $q_n = \frac{y_n+1}{2}$. Please derive $\nabla_{\mathbf{w}} H(q, P(y|\mathbf{x}, \mathbf{w}))$, express it with $q_n, \mathbf{x}_n$ and $\sigma(\cdot)$, and reduce it to the simplest form.

$$\nabla_{\mathbf{w}} H(q, P(y|x, \mathbf{w})) = -\sum_{n=1}^{N} (q_n - \sigma(\mathbf{w}^T \mathbf{x}_n))\mathbf{x}_n$$

$$= \sum_{n=1}^{N} (\sigma(\mathbf{w}^T \mathbf{x}_n) - q_n)\mathbf{x}_n \quad \text{(either is fine.)}$$

4.3 Iterative Update Formula of the Weight

Let the learning rate be λ. Please express gradient descent update formula of the weight using $P(y_n|\mathbf{x}_n, \mathbf{w})$, $\mathbf{x}_n$, and y_n.

$$
\begin{aligned}
(q_n - \sigma(\mathbf{w}^T\mathbf{x}_n))\mathbf{x}_n &= \begin{cases} 1 - \sigma(\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = \ \ 1 \\ -\sigma(\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = -1 \end{cases} \\
&= \begin{cases} \sigma(-\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = \ \ 1 \\ -\sigma(\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = -1 \end{cases} \\
&= \begin{cases} (+1)\sigma((-1)\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = \ \ 1 \\ (-1)\sigma((+1)\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = -1 \end{cases}
\end{aligned}
$$

$$
\begin{aligned}
&= \begin{cases} (+1)\sigma((-1)\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = \ 1 \\ (-1)\sigma((+1)\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n, & \text{if } y_n = -1 \end{cases} \\
&= y_n\sigma(-y_n\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n \\
&= P(-y_n|\mathbf{x}_n, \mathbf{w})y_n\mathbf{x}_n
\end{aligned}
$$

$$\Rightarrow \mathbf{w}_{k+1} = \mathbf{w}_k + \lambda \sum_{n=1}^{N} P(-y_n|\mathbf{x}_n, \mathbf{w}) y_n \mathbf{x}_n$$

Meaning: weight is moving towards a place that the prediction of every data point best fits its label.

Special case λ = 1/N, the update is a mean/average in which each point is weighted by the label (-1, +1) and how much it deviates from the prediction.

In the lecture, we said that logistic regression is a soft version of perceptron because the shape of sigmoid activation function is like a smoothed version of the sign function and that the update formulae look similar.

Answer the following 2 questions.

4.4 Predictive Model

The predictive model of a perceptron take the form: $\hat{y}_n = \text{sign}(\mathbf{w}^T \mathbf{x}_n)$. It applies a sign function to predict the label from the linear combination of input features $\mathbf{x}_n$. Explain why we cannot replace the sigmoid activation with a sign function and optimize the loss using gradient descent.

4.4 Predictive Model

The predictive model of a perceptron take the form: $\hat{y}_n = \text{sign}(\mathbf{w}^T \mathbf{x}_n)$. It applies a sign function to predict the label from the linear combination of input features $\mathbf{x}_n$. Explain why we cannot replace the sigmoid activation with a sign function and optimize the loss using gradient descent.

The derivative of a sign function is zero almost everywhere. Consequently, GD never update the weight and thus fails to minimize the loss.

3.5 Perceptron Loss (4 points)

$\nabla_w \sum_{n=1}^{N} L(y_n \mathbf{w}^T \mathbf{x}_n) = \sum_{n=1}^{N} -\mathbb{I}(y_n \mathbf{w}^T \mathbf{x}_n)$, where $L(z) = \max(0, -z)$. Describe or illustrate why the derivative of the perceptron loss function takes the form of an indicator function $\mathbb{I}(\cdot).$).

3.5 Perceptron Loss (4 points)

$\nabla_w \sum_{n=1}^{N} L(y_n \mathbf{w}^T \mathbf{x}_n) = \sum_{n=1}^{N} -\mathbb{I}(y_n \mathbf{w}^T \mathbf{x}_n)$, where $L(z) = \max(0, -z)$. Describe or illustrate why the derivative of the perceptron loss function takes the form of an indicator function $\mathbb{I}(\cdot).$).

Max function is piecewise linear. If the input is less than 0, the slope (hence derivative) of it is -1; otherwise, the slope is zero. The resulting derivative is a step function, and it is thus convenient to express the derivative as an indicator function.

Neural Network

Summary for convolution layer

Input: a tensor of size $W_1 \times H_1 \times D_1$

Hyperparameters:

- K filters of size $F \times F$
- stride S
- amount of zero padding P (for one side)

Output: a tensor of size $W_2 \times H_2 \times D_2$ where

- $W_2 = (W_1 + 2P - F)/S + 1$
- $H_2 = (H_1 + 2P - F)/S + 1$
- $D_2 = K$

Neural Network

3. Suppose a convolution layer takes a $4 \times 6 \times 3$ image as input and outputs a $3 \times 4 \times 6$ tensor. Which of the following is a possible configuration of this layer? **(3 points)**

(a) Two $2 \times 4 \times 3$ filters, stride 1, no zero-padding.

(b) Two $1 \times 1 \times 3$ filters, stride 2, 1 zero-padding.

(c) Six $2 \times 4 \times 3$ filters, stride 1, no zero-padding.

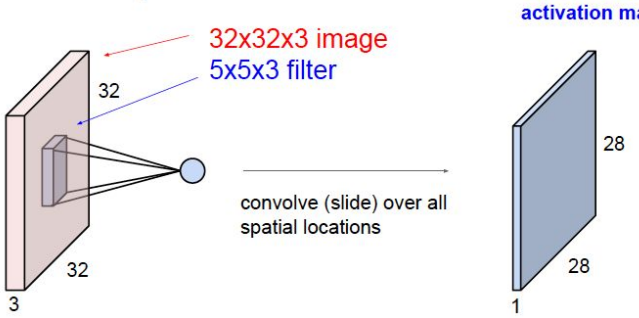
(d) Six $2 \times 4 \times 3$ filters, stride 2, 1 zero-padding.

D

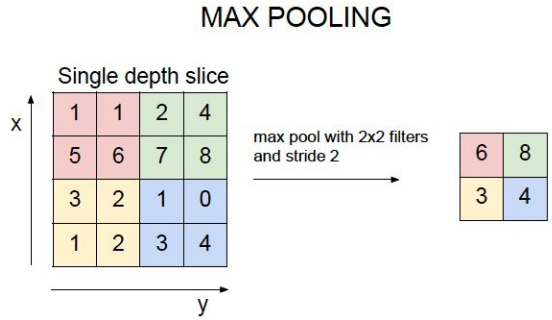
Elimination C:
(4 + 2*0 - 2)/1 + 1 = 3; (6+2*0-4)/1+1 = 5 (NOT equal to 4)

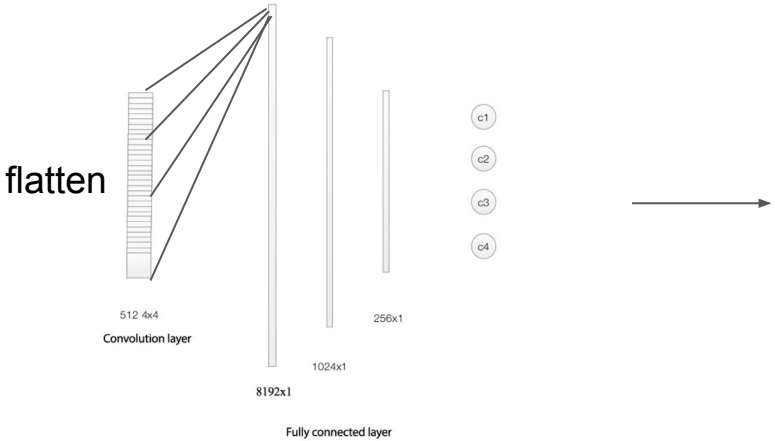
Space~O(K^2 * Cin * Cout)

Convolution Layer



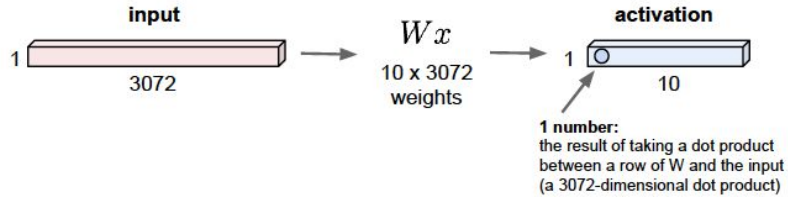
Max pooling with 2×2 filter and stride 2 is very common





Fully Connected Layer

32x32x3 image -> stretch to 3072 x 1



Neural Network

(d) How many parameters do we need to learn for the following network structure? An $8 \times 8 \times 3$ image input, followed by a convolution layer with 2 filters of size 2×2 (stride 1, no zero-padding), then another convolution layer with 4 filters of size 3×3 (stride 2, no zero-padding), and finally a max-pooling layer with a 2×2 filter (stride 1, no zero-padding). (Note: the depth of all filters are not explicitly spelled out, and we assume no bias/intercept terms are used.)

(A) 96
(B) 44
(C) 100
(D) 48

Conv1: K^2 * Cin * Cout = 2*2 * 3 * 2 = 24

Conv2: K^2 * Cin * Cout = 3*3 * 2 * 4 = 72

Pooling Layer: 0

A

Neural Network

Which of the following is wrong about neural nets?

(a) A fully connected feedforward neural net without nonlinear activation functions is the same as a linear model.
(b) Dropout technique prevents overfitting.
(c) A neural net with one hidden layer and a fixed number of neurons can represent any continuous function.
(d) A max-pooling layer has no parameters to be learned.

C

Suppose we have a Neural Network defined as below. An illustration is provided in the Figure below. Please answer the following questions.

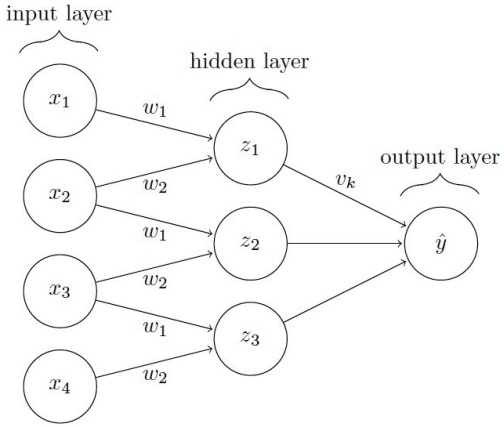


Figure 1: A neural network with one hidden layer.

Forward Propagation. The forward propagation can be expressed as:

$$\text{input layer} \qquad x_i, \tag{1}$$

$$\text{hidden layer} \qquad z_k = tanh\left(w_1 x_k + w_2 x_{k+1}\right), tanh(\alpha) = \frac{\mathrm{e}^{\alpha} - \mathrm{e}^{-\alpha}}{\mathrm{e}^{\alpha} + \mathrm{e}^{-\alpha}} \tag{2}$$

$$\text{output layer} \qquad \hat{y} = \sum_{k=1}^{3} v_k z_k \tag{3}$$

$$\text{loss function} \qquad L(y, \hat{y}) = ln(1 + exp(-y\hat{y})), \text{ where } \hat{y} \text{ is prediction, } y \text{ is ground truth} \tag{4}$$

Backpropagation Please write down $\frac{\partial L}{\partial v_k}$ and $\frac{\partial L}{\partial w_1}$ in terms of only x_k, v_k, z_k, y, and/or $\hat{y}$ using backpropagation.

Hint: $\frac{\partial \tanh(\alpha)}{\partial \alpha} = 1 - [\tanh(\alpha)]^2$.

The solution is:

$$\frac{\partial L}{\partial v_k} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial v_k}$$

$$\begin{aligned}
\frac{\partial L}{\partial \hat{y}} &= \frac{\partial}{\partial \hat{y}} \left[ln(1 + exp(-y\hat{y})) \right] \\
&= -\frac{y exp(-y\hat{y})}{1 + exp(-y\hat{y})} \\
&= -\sigma(-y\hat{y}) y \\
&= (\sigma(y\hat{y}) - 1) y
\end{aligned}$$

$$\frac{\partial \hat{y}}{\partial v_k} = \frac{\partial \sum_{k=1}^{3} v_k z_k}{\partial v_k} = z_k$$

$$\rightarrow \frac{\partial L}{\partial v_k} = (\sigma(y\hat{y}) - 1) y z_k$$

$$\frac{\partial L}{\partial w_1} = \sum_{k=1}^{3} \frac{\partial L}{\partial z_k} \frac{\partial z_k}{\partial w_1}$$

$$\begin{aligned} \frac{\partial L}{\partial z_k} &= \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_k} \\ &= (\sigma(y\hat{y}) - 1) y v_k \end{aligned}$$

$$\begin{aligned} \frac{\partial z_k}{\partial w_1} &= \frac{\partial}{\partial w_1} \tanh\left(w_1 x_k + w_2 x_{k+1}\right) \\ &= \left(1 - z_k^2\right) x_k \end{aligned}$$

$$\rightarrow \frac{\partial L}{\partial w_1} = \sum_{k=1}^{3} (\sigma(y\hat{y}) - 1) y v_k \left(1 - z_k^2\right) x_k$$