

140

HW Assignment 1 (Due by 10:30am on Sep 26)

Bhishan Poudel

1 Theory (40 points)

1. [Polynomial Curve Fitting, 20 points]

Consider the problem of fitting a dataset of N points with a polynomial of degree M , by minimizing the sum-of-squares error:

$$J(\mathbf{w}) = \frac{1}{2N} \sum_{n=1}^N (h_{\mathbf{w}}(\mathbf{x}_n) - t_n)^2 \quad (1)$$

where $h_{\mathbf{w}}(\mathbf{x}) = \sum_{j=0}^M w_j x^j$. We have shown in class that the solution to minimizing $J(\mathbf{w})$ satisfies the following set of linear equations:

$$\sum_{j=0}^M A_{ij} w_j = T_i \quad (2)$$

$$\text{where } A_{ij} = \sum_{n=1}^N x_n^{i+j} \text{ and } T_i = \sum_{n=1}^N x_n^i t_n \quad (3)$$

Derive the solution for the regularized version of polynomial curve fitting, which minimizes the objective function below:

$$J(\mathbf{w}) = \frac{1}{2N} \sum_{n=1}^N (h_{\mathbf{w}}(\mathbf{x}_n) - t_n)^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2 \quad (4)$$

2. [Probability Theory, 20 points] *Exercise 1.3, page 58 in PRML.*

Suppose that we have three coloured boxes r (red), b (blue), and g (green). Box r contains 3 apples, 4 oranges, and 3 limes, box b contains 1 apple, 1 orange, and 0 limes, and box g contains 3 apples, 3 oranges, and 4 limes. If a box is chosen at random with probabilities $p(r) = 0.2$, $p(b) = 0.2$, $p(g) = 0.6$, and a piece of fruit is removed from the box (with equal probability of selecting any of the items in the box), then what is the probability of selecting an apple? If we observe that the selected fruit is in fact an orange, what is the probability that it came from the green box?

3. [Ridge Regression (*), 20 points]

Consider the regularized linear regression objective shown below:

$$J(\mathbf{w}) = \frac{1}{2N} \sum_{n=1}^N (h(\mathbf{x}_n, \mathbf{w}) - t_n)^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

- (a) Minimizing the L2 norm of \mathbf{w} drives all parameters, including w_0 , towards 0. Are there situations in which we do not want to constrain w_0 to be small? If yes, give an example, if not show why it is useful to constrain all the weights to be small, including w_0 .
- (b) We have seen in class how to compute the weights \mathbf{w} that minimize $J(\mathbf{w})$. Assume now that we replace $\|\mathbf{w}\|$ in $J(\mathbf{w})$ with $\|\mathbf{w}_{[1:]}\|$, where $\mathbf{w}_{[1:]} = [w_1, w_2, \dots, w_M]$. Derive the solution for \mathbf{w} that minimizes this new objective function.

2 Implementation (80 points)

In this exercise, you are asked to run an experimental evaluation of linear regression on the Athens houses dataset, and on an artificial dataset with and without L2 regularization. The input data is available at <http://ace.cs.ohio.edu/~razvan/courses/ml4900/hw01.zip>. Make sure that you organize your code in folders as shown in the table below. Write code only in the Python files indicated in bold.

$w = (X^T X)^{-1} e^T e + t = \frac{\text{two } w_1, w_2, w_3, t}{\text{row 22 array}}$ $\text{RMSE} = \sqrt{\frac{\sum_{n=1}^N (h_n - t_n)^2}{N}}$ $= \frac{\text{np.sqrt(np.sum((h-t)^2))}}{N}$ $= \text{np.sqrt}((h-t)^2 \cdot \text{mean}())$	<pre>ml4900/ hw01/ code/ univariate.py multivariate.py polyfit.py train_test_line.png data/ univariate/ train.txt, test.txt multivariate/ train.txt, test.txt polyfit/ train.txt, test.txt, devel.txt</pre>
---	---

1. [Univariate Regression, 20 points]

Train a univariate linear regression model to predict house prices as a function of their floor size, based on the solution to the system with 2 linear equations discussed in class. Use the dataset from the folder `hw01/data/univariate`. Python3 skeleton code is provided in `univariate.py`. After training print the parameters and report the RMSE and the objective function values on the training and test data. Plot the training using the default blue circles and test examples using lime green triangles. On the same graph also plot the linear approximation.

2. [Multivariate Regression, 20 points]

Train a univariate linear regression model to predict house prices as a function of their floor size, number of bedrooms, and year. Use the normal equations discussed in class, and evaluate on the dataset from the folder `hw01/data/multivariate`. Python3 skeleton code is provided in `multivariate.py`. After training print the parameters and report the RMSE and the objective function values on the training and test data. Compare the test RMSE with the one from the univariate case above.

3. [Polynomial Curve Fitting, 40 points]

In this exercise, you are asked to run an experimental evaluation of a linear regression model, with and without regularization. Use the normal equations discussed in class, and evaluate on the dataset from the folder `hw01/data/polyfit`.

- (a) Select 30 values for $x \in [0, 1]$ uniformly spaced, and generate corresponding t values according to $t(x) = \sin(2\pi x) + x(x+1)/4 + \epsilon$, where $\epsilon = N(0, 0.005)$ is a

zero mean Gaussian with variance 0.005. Save and plot all the values. Done in `dataset.txt`.

- (b) Split the 30 samples (x_n, t_n) in three sets: 10 samples for training, 10 samples for validation, and 10 samples for testing. Save and plot the 3 datasets separately. Done in `train.txt`, `test.txt`, `devel.txt`.
- (c) Consider a linear regression model with polynomial basis functions, trained with the objective shown below:

$$J(\mathbf{w}) = \underbrace{\frac{1}{2N} \sum_{n=1}^N (h(x_n, \mathbf{w}) - t_n)^2}_{\text{hat MSE}} + \underbrace{\frac{\lambda}{2} \|\mathbf{w}\|^2}_{\text{L2 Regularization}}$$

$\sum_{j=1}^M w_j$
generally we don't
regularize w_0 .

Show the closed form solution (vectorized) for the weights \mathbf{w} that minimize $J(\mathbf{w})$.

- (d) Train and evaluate the linear regression model in the following scenarios:
1. Without regularization: Use the training data to infer the parameters \mathbf{w} for all values of $M \in [0, 9]$. For each order M , compute the RMSE separately for the training and test data, and plot all the values on the same graph, as shown in class.
 2. With regularization: Fixing $M = 9$, use the training data to infer the parameters \mathbf{w} , one parameter vector for each value of $\ln \lambda \in [-50, 0]$ in steps of 5. For each parameter vector (lambda value), compute the RMSE separately for the training and validation data, and plot all the values on the same graph, as shown in class. Select the regularization parameter that leads to the parameter vector that obtains the lowest RMSE on the validation data, and use it to evaluate the model on the test data. Report and compare the test RMSE with the one obtained without regularization.

3 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. Electronically submit on Blackboard a `hw01.zip` file that contains the `hw01` folder in which your code is in the 3 required files.

On a Linux system, creating the archive can be done using the command:

```
> zip -r hw01.zip hw01.
```

Please observe the following when handing in homework:

1. Structure, indent, and format your code well.
2. Use adequate comments, both block and in-line to document your code.
3. On the theory assignment, **clear and complete explanations and proofs of your results are as important as getting the right answer.**
4. Make sure your code runs correctly when used in the directory structure shown above.

HW 1

Due: Sep 19 TU

Bhishan Poudel

Q1.1 polynomial curve fitting

cost function without the regularization

is

$$J(w) = \frac{1}{2N} \sum_{n=1}^N (h_w(x_n) - t_n)^2 \quad \textcircled{1}$$

where predicted values

$$h_w(x) = \underbrace{w_0 x_0}_{=w_0} + w_1 x_1 + w_2 x_2 + \dots + w_M x_M$$

w_0, w_1, \dots, w_M
= model parameters
 w_0 = bias term or
intercept term

$$h_w(x) = \sum_{j=0}^M w_j x_j$$

minimizing the cost function w/o regularization,

$$\frac{\partial J}{\partial w} = 0 \quad \text{gives}$$

$$\sum_{j=0}^M A_{ij} w_j = T_i \quad \textcircled{2}$$

where

$$A_{ij} = \sum_{n=1}^N x_n^{i+j}$$

$$T_i = \sum_{n=1}^N x_n^i + t_n$$

\textcircled{3}

problem: minimize the function, λ is shrinkage

$$J(\mathbf{w}) = \frac{1}{2N} \sum_{n=1}^N \underbrace{\left(h_{\mathbf{w}}(\mathbf{x}_n) - t_n \right)^2}_{\frac{1}{2} \cdot \text{SSE}} + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

L2 regularizer
or
shrinkage penalty

solution: the hypothesis is

$$h_{\mathbf{w}}(\mathbf{x}_n) = \sum_{j=0}^M w_j x_n^j$$

$n = 0, \dots, N \rightarrow N$ samples x_1, x_2, \dots, x_N
 $j = 0, \dots, M \rightarrow M$ features w_0, w_1, \dots, w_M

bias $\frac{\text{trials}}{w_0}$ $\frac{\text{trials}}{w_1}$ $\frac{\text{age}}{w_2}$

$t = \text{target}$ (this)
(price)

we can write cost function as,

$$J(\mathbf{w}) = \frac{1}{2N} \sum_{n=1}^N \left(\sum_{j=0}^M w_j x_n^j - t_n \right)^2 + \frac{\lambda}{2} \sum_{j=0}^M w_j^2$$

we want to minimize the cost function $J(\mathbf{w})$ w.r.t.
weights w_i ,

$$\nabla_{w_i} J(\mathbf{w}) = 0$$

$$\frac{\partial J(w_i)}{\partial w_i} = 0$$

$$0 = \frac{1}{2N} \cdot 2 \sum_{n=1}^N \left(\sum_{j=0}^M w_j x_n^j - t_n \right) \cdot x_n^i + \frac{\lambda}{2} \cdot 2 w_i + N$$

$$0 = \sum_{n=1}^N \sum_{j=0}^M w_j x_n^{ij} - \underbrace{\sum_{n=1}^N t_n x_n^i}_{\text{constant}} + \lambda N w_i$$

$$\text{Or, } \sum_{j=0}^M \left(\sum_{n=1}^N \alpha_n^{j+n} \right) w_i + \gamma_N w_i = \sum_{n=1}^N t_n \alpha_n^i$$

$$\Rightarrow \boxed{\sum_{j=0}^M A_{ij} w_j + \gamma_N w_i = T_i} \quad \# \text{ Answer}$$

where,

$$A_{ij} = \sum_{n=1}^N \alpha_n^{i+j}$$

$$T_i = \sum_{n=1}^N t_n \alpha_n^i$$

(2)
✓

QNT2 probability Theory By Bhopal

Ex 1.3, page 58 in PRML

Red R: 3a 4o 3l

blue B: 2a 2o 0l

green G: 3a 3o 4l

$$P(R) = 0.2 \quad P(B) = 0.2 \quad P(G) = 0.6$$

- If a piece of fruit is removed from the box, then what is the prob of selecting an apple?

- If we observe that the selected fruit is an apple, what is the prob that it came from the green box?

Solution:

⑥	3a 4o 3l
---	----------------

Red

$$N_R = 10$$

2a 2o 0l

Blue

$$N_B = 2$$

3a 3o 4l

Green

$$N_G = 10$$

$$\begin{aligned} N &= N_R + N_B + N_G \\ &= 10 + 2 + 10 \\ &= 22 \end{aligned}$$

$$P(A|R) = \frac{3}{10} = 0.3 \quad P(A|B) = \frac{1}{2} = 0.5 \quad P(A|G) = \frac{3}{10} = 0.3 \quad \left. \right\} \text{from figure}$$

$$P(O|R) = \frac{4}{10} = 0.4 \quad P(O|B) = \frac{1}{2} = 0.5 \quad P(O|G) = \frac{3}{10} = 0.3$$

$$P(R) = 0.2 \quad P(B) = 0.2 \quad P(G) = 0.6 \quad] \text{given}$$

Implementation

QN 3.C

Given objective function for linear regression with polynomial basis functions is,

Cost function

$$J(\vec{w}) = \frac{1}{2N} \sum_{n=1}^N [h(x_n; \vec{w}) - t_n]^2 + \frac{\lambda}{2} \|\vec{w}\|^2 \quad \textcircled{A}$$

Now, we have to find the vectorized solution of w which minimizes cost function J .

For this,

we first write hypothesis in terms of design matrix X and weight vector w . ==

$$h = Xw \quad \textcircled{1}$$

Then, eqn \textcircled{A} can be written as,

$$J = \frac{1}{2N} (Xw - t)^T (Xw - t) + \frac{\lambda}{2} w^T w$$

We minimize this equation w.r.t. w ,

$$0 = \frac{\partial J}{\partial w}$$

$$= \frac{1}{2N} \cdot 2X^T(Xw - t) + \frac{\lambda}{2} \cdot 2w$$

more we have used matrix derivative

for formula

$$\frac{d}{d\omega} (\mathbf{x}^T \mathbf{x}) = 2\mathbf{x}$$

$$\frac{d}{d\omega} (\mathbf{A}\mathbf{x} + \mathbf{b}) = 2\mathbf{A}^T (\mathbf{A}\mathbf{x} + \mathbf{b})$$

as we have matrices

formula

then,

$$0 = \mathbf{x}^T \mathbf{x} \mathbf{w} - \mathbf{x}^T \mathbf{t} + dN M$$

$$\mathbf{x}^T \mathbf{t} = (\mathbf{x}^T \mathbf{x} + dN) \mathbf{w}$$

$$\mathbf{w} = (dN \mathbf{I} + \mathbf{x}^T \mathbf{x})^{-1} (\mathbf{x}^T \mathbf{t}) \quad ||$$

ANSWER

note that $(\mathbf{x}^T \mathbf{x})^{-1} \mathbf{x}^T$ is called moore-penrose
pseudoinverse of any shape matrix \mathbf{x} .

e.g. $\mathbf{x}^T \mathbf{x} = (4, 5, 0) (80, 4) = (4, 4) = \mathbf{I}_{4,4}$

$$\mathbf{x}^T \mathbf{t} = (4, 5, 0) (80, 1) = (4, 1)$$

$$\mathbf{w} = (4, 1) (4, 1) = (1, 1) = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

$$\mathbf{w} \cdot \mathbf{t} = [w_0 \ w_1 \ w_2 \ w_3] \mathbf{t} = x(n+1)$$

$$\mathbf{I}_{n+1} = \begin{bmatrix} 1 & & & 0 \\ & 1 & & \\ & & 1 & \\ 0 & & & 1 \end{bmatrix}$$

set $\mathbf{I}^T \mathbf{I} = \mathbf{I}$
we do not regularize bias term.

$$= \begin{bmatrix} 1 & & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

if we regularize bias term, it will shift the mean of target vector.

then adding const value to all the target DOES NOT hypothesis similarity shifted hypothesis.

I-1 (20)

I-2 (20)

I-3 (40)

HW Assignment 3 (Due by 10:30am on Oct 12)

(106/100)

(120)

1 Theory (100 points)**1. [Maximum Likelihood, 20 points]**

The Poisson distribution specifies the probability of observing k events in an interval, as follows:

$$P(k \text{ events in interval}) = e^{-\lambda} \frac{\lambda^k}{k!} \quad (1)$$

For example, k can be the number of meteors greater than 1 meter diameter that strike Earth in a year, or the number of patients arriving in an emergency room between 10 and 11 pm¹.

Suppose we observe N samples k_1, k_2, \dots, k_N from this distribution (i.e. numbers of meteors that strike Earth over a period of N years). Derive the maximum likelihood estimate of the event rate λ .

2. [Logistic Regression, 20 points]

Consider a dataset that contains the 4 examples below i.e., the truth table of the logical XOR function. Prove that no logistic regression model can perfectly classify this dataset. Do not forget the bias feature $x_0 = 1$.

x_1	x_2	t
0	0	0
0	1	1
1	0	1
1	1	0

Hint: Prove that there cannot be a vector of parameters \mathbf{w} such that $P(t = 1|\mathbf{x}, \mathbf{w}) \geq 0.5$ for all examples \mathbf{x} that are positive, and $P(t = 1|\mathbf{x}, \mathbf{w}) < 0.5$ for all examples \mathbf{x} that are negative.

3. [Logistic Regression, 20 points]

Prove that the gradient (with respect to \mathbf{w}) of the negative log-likelihood error function for logistic regression corresponds to the formula shown in lecture 4:

$$\nabla_{\mathbf{w}} E(\mathbf{w}) = \sum_{n=1}^N (h_n - t_n) \mathbf{x}_n \quad (2)$$

4. [Logistic Regression, 20 points]

In `scikit`, the objective function for logistic regression expresses the trade-off between training error and model complexity through a parameter C that is multiplied with the error term, as shown below. See the `scikit` documentation at http://scikit-learn.org/stable/modules/linear_model.html#logistic-regression.

$$E(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C * \sum_{n=1}^N \ln(e^{-t_n(\mathbf{w}^T \mathbf{x}_n)} + 1) \quad (3)$$

¹https://en.wikipedia.org/wiki/Poisson_distribution

- Show that the sum in the second term is equal with the negative log-likelihood, where $t_n = +1$ stands for positive labels and $t_n = -1$ stands for negative labels.
 - Compute the C parameter such that the objective is equivalent with the standard formulation shown on the slides in which the regularization parameter λ is multiplied with the L2 norm term.
5. [Softmax Regression, 20 points]
 Show that Logistic Regression is a special case of Softmax Regression. That is to say, if w_1 and w_2 are the parameter vectors of a Softmax Regression model for the case of two classes, then there exists a parameter vector w for Logistic Regression that results in the same classification as the Softmax Regression model.
6. [Softmax Regression (*), 20 points]
 Prove that the gradient (with respect to w_k) of the negative log-likelihood error function for regularized softmax regression corresponds to the formula shown in lecture 4, for any class $k \in [1..K]$:

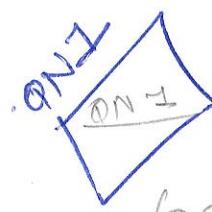
$$\nabla_{w_k} E(w) = -\frac{1}{N} \sum_{n=1}^N (\delta_k(t_n) - p(C_k | \mathbf{x}_n)) \mathbf{x}_n + \alpha w_k \quad (4)$$

2 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. On this theory assignment, **clear and complete explanations and proofs of your results are as important as getting the right answer.**

HW 03

Bhishan Poudel

maximum LikelihoodDerive MLE of event rate parameter λ)

Let x_1, x_2, \dots, x_n be iid poisson random variables with prob mass function,

pmf of poisson distn

$$p(x_i | \lambda) = \frac{e^{-\lambda} \lambda^{x_i}}{x_i!}$$

$$= p(x_i)$$

Now, the joint prob of all variables x_i , called likelihood function, is given by

likelihood

$$L(\lambda) = \prod_{i=1}^n p(x_i | \lambda)$$

$$= \prod_{i=1}^n \frac{e^{-\lambda} \lambda^{x_i}}{x_i!}$$

The log likelihood of pmf is,

Log likelihood

$$\ln L(\lambda) = \ln \prod_{i=1}^n \frac{e^{-\lambda} \lambda^{x_i}}{x_i!}$$

$$= \sum_{i=1}^n \ln \frac{e^{-\lambda} \lambda^{x_i}}{x_i!}$$

$$= \sum_{i=1}^n [-\lambda + x_i \ln \lambda - \ln x_i!]$$

$$\ln L(\lambda) = -n\lambda + n\ln \lambda - \sum_{i=1}^n \ln x_i!$$

①

To get the maximum likelihood estimate of the parameter λ , we maximize the Log Likelihood function $L(\lambda)$ w.r.t λ . [maximizing $L(\lambda)$ is same as minimizing $-n\ln L(\lambda)$]

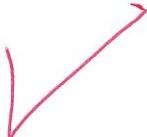
$$0 = \frac{\partial}{\partial \lambda} \ln L(\lambda)$$

$$= \frac{\partial}{\partial \lambda} [-n\lambda + n \ln \bar{x}_i - \bar{x}_i \ln \bar{x}_i]$$

$$0 = -n + \frac{\bar{x}_i}{\lambda} - 0$$

$$\lambda = \frac{\bar{x}_i}{n}$$

(25)

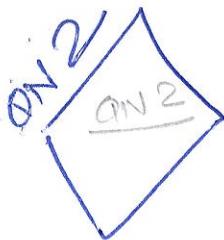


MLE of λ

$$\boxed{\lambda = \frac{\sum_{i=1}^n x_i}{n}} \rightarrow \text{number of samples}$$

Ans

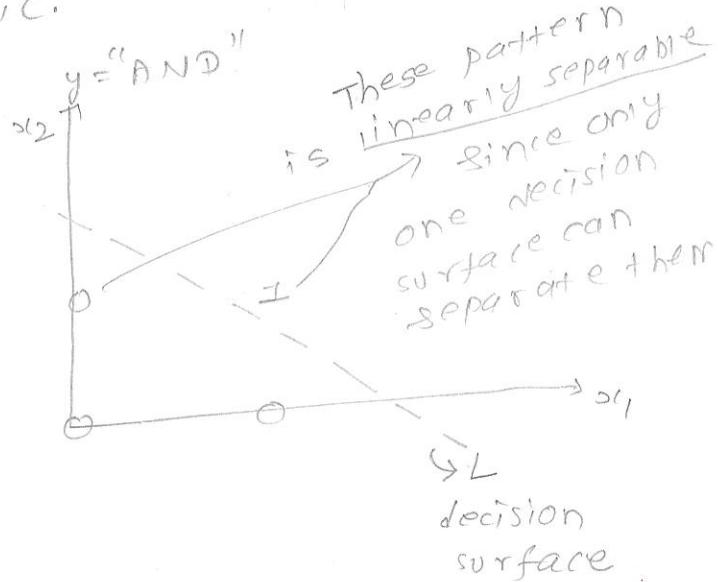
Hence, the maximum likelihood estimate of the Poisson distribution parameter λ is just the mean (or expectation) of the distribution.



XOR problem in Logistic Regression

To describe XOR problem in LR, I shall start with "AND" logic.

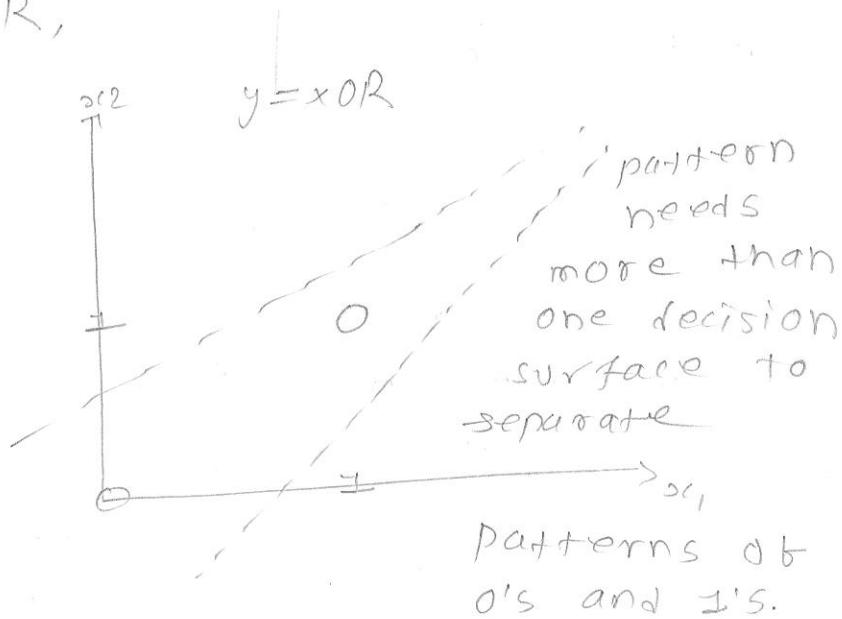
x_1	x_2	$y = x_1 \wedge x_2$
0	0	0
0	1	0
1	0	0
1	1	1



Now, look at XOR,

Truth table

x_1	x_2	$y = \text{XOR}$
0	0	0
0	1	1
1	0	1
1	1	0



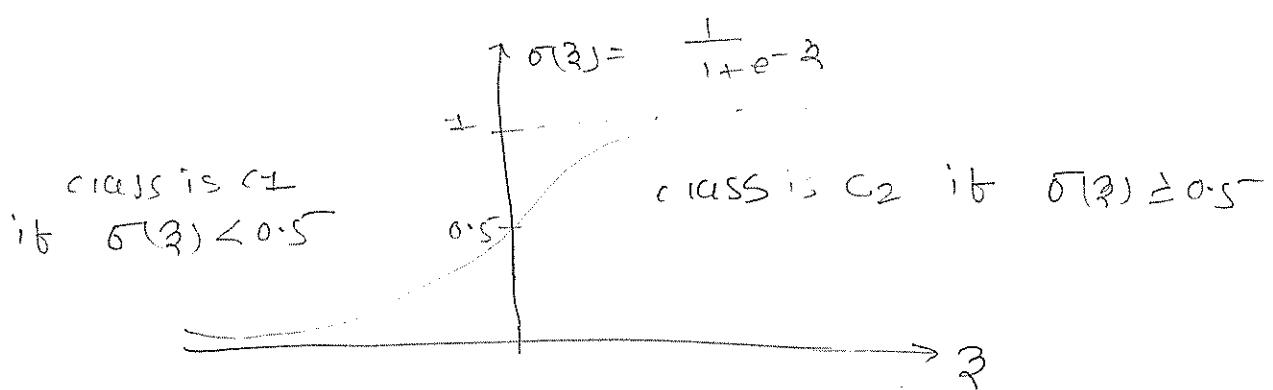
\therefore XOR logic is not linearly separable.

P.T.O.

Now, we shall show that Logistic regression is linear classifier:

The classifier used in LR is sigmoid

$$\text{function } \sigma(z) = \frac{1}{1+e^{-z}}$$



In Logistic Regression,

$$\text{pattern is } + \text{ if } \frac{1}{1+e^{-w^T x}} \geq \frac{1}{2}$$

$$0 \text{ if } \frac{1}{1+e^{-w^T x}} < \frac{1}{2}$$

let's set the classifier to separator value.

$$\frac{1}{1+e^{-w^T x}} = \frac{1}{2}$$

$$z = 1 + e^{-w^T x}$$

$$\frac{1}{z} = e^{-w^T x} = \frac{1}{e^{w^T x}}$$

$$e^{w^T x} = 1$$

$$w^T x = \ln 1 = 0$$

$$\Rightarrow \boxed{\sum_i w_i x_i = 0}$$

this means L is linear classifier.

conclusion: \Rightarrow 1) this linear classifier

2) XOR problem is non-linear and
linearly inseparable

use:

$w^T x_n > 0$ if $x_n \in +$

$w^T x_n < 0$ if $x_n \in -$

then put 4 samples into
the inequalities.

perfectly classify
dataset that
follows XOR
logic.

ANS

Better method

x_0	x_1	x_2	t
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

for linear separability, we want

$$w_0 + \sum_i w_i x_i < 0 \text{ if } t_i = 0$$

$$w_0 + \sum_i w_i x_i \geq 0 \text{ if } t_i = 1$$

Note,

$$w_0 < 0 \quad \text{--- (1)}$$

$$w_0 + w_2 \geq 0 \quad \text{--- (2)} \quad \left. \begin{array}{l} \\ \end{array} \right\} w_0 + (w_0 + w_1 + w_2) \geq 0$$

$$w_0 + w_1 \geq 0 \quad \text{--- (3)} \quad \text{but } w_0 < 0$$

$$w_0 + w_1 + w_2 < 0 \quad \text{--- (4)} \quad \text{and } w_0 + w_1 + w_2 < 0$$

\therefore it contradicts our assumptions of linear separability.

ON3
ON3

Gradient of Logistic Regression cost

(Bind & E)

for linear regression, hypothesis $h = w^T x$

for logistic regression, hypothesis $h = \sigma = \frac{1}{1+e^{-w^T x}}$

Aside:
binomial dist'n

$$p(x|n) = n! \cdot p^x \cdot (1-p)^{n-x} \\ \text{for } x \in \{0, \dots, n\} \\ n! = \frac{n!}{x!(n-x)!}$$

$$h = \sigma = \frac{1}{1+e^{-w^T x}}$$

CI write $w^T x$
as wx since they
are just matrix
dot product of two

matrices)

Binomial
Likelihood function (Bernoulli)

$$\left. \begin{array}{l} \text{prob for logistic regression} \\ p(t|x) = h^n \cdot (1-h)^{1-n} \end{array} \right\}$$

Likelihood function

$$L(w) = \prod_{n=1}^N h^n \cdot (1-h)^{1-n}$$

(Bishop eq 4.89
p. 206/223)

-ve log likelihood, E or $J = -\ln L(w)$

$$E = -\ln \prod_{n=1}^N h^n \cdot (1-h)^{1-n}$$

$$E = \sum_n E_n = \sum_{n=1}^N (-t_n \ln h_n - (1-t_n) \ln (1-h_n))$$

$$\text{Loss, } E(w) = -\frac{1}{N} \sum_n [t_n \ln h_n + (1-t_n) \ln (1-h_n)]$$

for any n th sample the cost can be written as,

LOSS

$$E = -t_n \ln h - (1-t_n) \ln (1-h) \quad \text{--- (2)}$$

-ve log
likelihood
error fn
for Vf

$$\text{here } h = \sigma = \sigma(w^T x) = \sigma(wx)$$

Before proceeding further, I would derive derivative of sigmoid function.

$$\sigma(z) = \frac{1}{1+e^{-z}} \Rightarrow \frac{d\sigma}{dz} = \frac{-1 \cdot e^{-z} \cdot (-1)}{(1+e^{-z})^2}$$

$$\begin{aligned}\frac{d\sigma}{dz} &= \frac{e^{-z}}{(1+e^{-z})^2} = \frac{1}{1+e^{-z}} \cdot \frac{(e^{-z}+1)-1}{1+e^{-z}} \\ &= \frac{1}{1+e^{-z}} \cdot \left(\frac{e^{-z}+1}{1+e^{-z}} - \frac{1}{1+e^{-z}} \right) \\ &= \frac{1}{1+e^{-z}} \cdot \left(1 - \frac{1}{1+e^{-z}} \right)\end{aligned}$$

$$\boxed{\frac{d\sigma(z)}{dz} = \sigma(1-\sigma)}$$

similarly $\boxed{\frac{d\sigma(a_2)}{dz} = \alpha \sigma(1-\sigma)}$ (here $a \neq a(z)$)

Derivative
Sigmoid
function

$$\boxed{\frac{d\sigma(a_2)}{dz} = \alpha \sigma(1-\sigma)}$$

Now, going back to the problem, let's calculate the gradient of loss function.

$$\frac{\partial E}{\partial w} = \frac{2}{n} \left[-t \ln h - (1-t) \ln(1-h) \right]$$

$$= -\frac{1}{n} \frac{\partial h}{\partial w} - \frac{(1-t)}{1-h} \cdot \frac{\partial h}{\partial w}$$

$$= -\frac{1}{n} \frac{\partial \sigma(wx)}{\partial w} + \frac{1-t}{1-h} \frac{\partial \sigma(wx)}{\partial w}$$

$$= -\frac{1}{n} \times t(1-h) + \frac{1-t}{1-h} \times h(1-h)$$

$$\therefore \frac{d\sigma(\theta)}{d\theta} = a\sigma(1-\sigma)$$

(W)

$$= -t \times \cancel{-txh} + xh - \cancel{txh}$$

✓

$$= xh - tx$$

$$\boxed{\frac{\partial E}{\partial w} = (h-t)x}$$

(here this is for a single sample, but for total loss we add up all the losses.)

$$\Rightarrow \boxed{\nabla_w E = \sum_{n=1}^N (h_n - t_n) x_n}$$

Q.E.D.

Q34
ONLY

Logistic Regression in SKLEARN

~~parts~~ the cost function for L2 penalized logistic regression is given by,

L2 Regularized cost fn
for LR
(in SKLearn)

$$E = \frac{1}{2} w^T w + C \sum_{n=1}^N \ln(1+e^{-t_n w^T x_n})$$

L2 regularizer cost for LR

where $t_n \in \{-1, 1\}$

①

we have to show that the summation

permits eval to the -ve log likelihood,

-ve log
likelihood

$$ED = -\ln p = \sum_{n=1}^N \ln(1+e^{-t_n w^T x_n}) \quad \text{--- } ②$$

Now, the posterior probabilities for class 0 and 1 are,

posterior
prob
for
LR

$$p(c_1|x) = \sigma(w^T x) = \frac{1}{1+e^{w^T x}}$$

$$p(c_0|x) = 1 - \sigma(w^T x) = \frac{e^{-w^T x}}{1+e^{-w^T x}}$$

③

The likelihood function is,

Likelihood
for
LR

$$P(t|w) = \prod_{n=1}^N h_n^{t_n} (1-h_n)^{(1-t_n)} \quad \text{--- } ④$$

11

The -ve log likelihood is,

$$\begin{aligned}-\text{un } p(\mathbf{x}|\mathbf{w}) &= -\text{un } \prod_{n=1}^N h_n^{t_n} \cdot (1-h_n)^{1-t_n} = -\sum_{n=1}^N \text{un}(h_n^{t_n}, (1-h_n)^{1-t_n}) \\&= -\sum_{n=1}^N [\text{un}(h_n^{t_n}) + \text{un}(1-h_n)^{1-t_n}]\end{aligned}$$

cost for LR

$E_D = -\text{un } p = -\sum_{n=1}^N [\underbrace{\text{un } h_n}_{\text{for class 0}} + \underbrace{(1-h_n)}_{\text{for class 1}} \text{un}(1-h_n)] \quad \textcircled{5}$

where $t_n \in \{0, 1\}$

$$h_n = \frac{1}{1+e^{-\mathbf{w}^T \mathbf{x}_n}}$$

$$1-h_n = 1 - \frac{1}{1+e^{-\mathbf{w}^T \mathbf{x}_n}}$$

$$= \frac{1+e^{-\mathbf{w}^T \mathbf{x}_n} - 1}{1+e^{-\mathbf{w}^T \mathbf{x}_n}}$$

$$= \frac{e^{-\mathbf{w}^T \mathbf{x}_n}}{1+e^{-\mathbf{w}^T \mathbf{x}_n}}$$

$$= \frac{1}{e^{\mathbf{w}^T \mathbf{x}_n} + 1}$$

$1-h_n = \frac{1}{1+e^{\mathbf{w}^T \mathbf{x}_n}}$

$$E_D = -\sum_{n=1}^N t_n p$$

$$= -\sum_{n=1}^N \left[t_n \ln \left(\frac{1}{1+e^{-w^T x_n}} \right) + (1-t_n) \ln \left(\frac{1}{1+e^{w^T x_n}} \right) \right]$$

$t_n \in \{0, 1\}$

$$= -\sum_{n=1}^N \begin{cases} \ln \frac{1}{1+e^{-w^T x_n}} & \text{if } t_n = 1 \\ \ln \frac{1}{1+e^{w^T x_n}} & \text{if } t_n = 0 \end{cases}$$

$$= -\sum_{n=1}^N t_n \ln \frac{1}{1+e^{-t_n w^T x_n}} \quad \text{if } t_n \in \{-1, 1\}$$

(F4)

$E_D = -\sum_{n=1}^N t_n \ln (1+e^{-t_n w^T x_n})$

$Q.E.D.$

$t_n \in \{-1, 1\}$

✓

QN 4b Find 'C' parameter of LR cost fn in sklearn.

Note, the L2 regularized cost function for LR

$$E = E_w + E_D$$

$$= \frac{1}{2} w^T w + (-\ln p)$$

when the t_n 's are {0, 1}

$$E = \frac{1}{2} w^T w - \sum_{n=1}^N [t_n \ln h_n + (1-t_n) \ln (1-h_n)] \quad \text{⑥}$$

$t_n \in \{0, 1\}$

when the t_n 's are {-1, 1}

$$E = \frac{1}{2} w^T w + \sum_{n=1}^N \ln (1 + e^{-t_n w^T x_n}) \quad \text{①}$$

$t_n \in \{-1, 1\}$

when t_n 's are {-1, 1} in sklearn

$$E(w) = \frac{1}{2} w^T w + C \sum_{n=1}^N \ln (1 + e^{-t_n w^T x_n}) \quad \text{②}$$

To get maximum likelihood estimate of parameter w ,

$$\hat{w} = \underset{w}{\operatorname{argmin}} \left[\frac{1}{2} w^T w + \sum_{n=1}^N \ln (1 + e^{-t_n w^T x_n}) \right]$$

$$\Rightarrow \hat{w} = \underset{w}{\operatorname{argmin}} \left[\frac{1}{2} w^T w + \frac{1}{2} \sum_{n=1}^N \ln (1 + e^{-t_n w^T x_n}) \right] \quad (\text{general form})$$

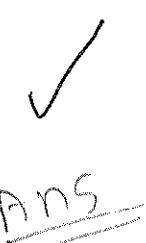
$$\text{but } \hat{w} = \underset{w}{\operatorname{argmin}} \left[\frac{1}{2} w^T w + C \sum_{n=1}^N \ln (1 + e^{-t_n w^T x_n}) \right] \quad (\text{scikit form})$$

(b)

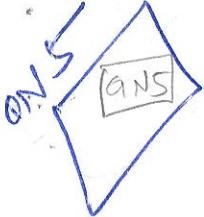
∴

$$C = \frac{1}{2}$$

Ans

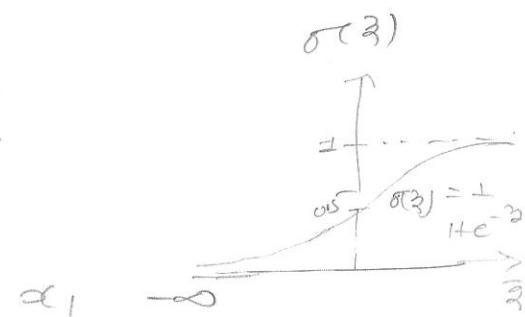
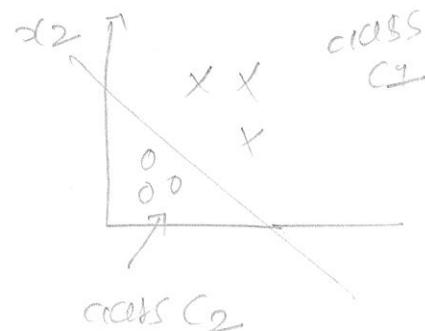


Example
To add bias term
sklearn uses
 $w^T \rightarrow w^T - C \cdot \mathbf{1}$
 E_D



softmax regression is special
case of logistic regression

FOR LR



(sigmoid fn)

(binary classification)

the posterior prob of class C1 is
posterior prob

$$p(c_1|x; w) = p(c_1|x) = \sigma(w^T x) = \frac{1}{1 + e^{-w^T x}}$$

the posterior prob of class C2 is,

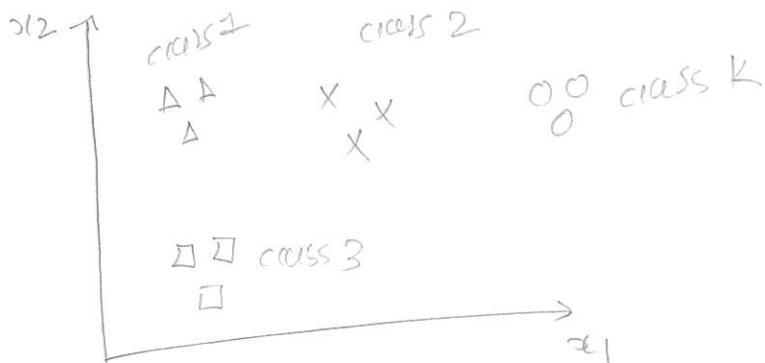
$$p(c_2|x) = 1 - \sigma(w^T x) = \frac{e^{-w^T x}}{1 + e^{-w^T x}}$$

hypothesis → the hypothesis for LR,

$$h(w) = \begin{cases} \frac{1}{1 + e^{-w^T x}} & \rightarrow \text{for class 1} \\ \frac{e^{-w^T x}}{1 + e^{-w^T x}} & \rightarrow \text{for class 2} \end{cases}$$

Again, FOR SR

(softmax Regression
or multiclass logistic regression)



total number of
samples = N

In softmax regression,
the posterior probability of
any class C_K is given by,

$$P(C_K|x, w) = \frac{e^{w_K^T x}}{\sum_{k=1}^K e^{w_k^T x}}$$

posterior
prob.

INPUT DATA X	target
x_1 some values	triangle +1
x_2	cross +2
:	rectangle +3
x_N	circle +N

[Bishop, 4.3.4]
eq 4.1104
p. 209 / Pdtb 227

note $a_K = w_K^T x$ is
called activation of
for the parameter
vector w_K

$$h = \frac{1}{\sum_{k=1}^K e^{w_k^T x}} \begin{bmatrix} e^{w_1^T x} \\ e^{w_2^T x} \\ \vdots \\ e^{w_K^T x} \end{bmatrix}$$

hyp:
plus sign
 $x w^T x$

sigmoid is -
softmax is +

Show $h(w) = h(w - \psi)$

[SL has
overparametrization
property]

$$h(w - \psi) = \frac{e^{(w_K - \psi)^T \alpha}}{\sum_{k=1}^K e^{(w_k - \psi)^T \alpha}}$$

$$= \frac{e^{w_K^T \alpha} \cdot e^{-\psi^T \alpha}}{\sum_k e^{w_k^T \alpha} \cdot e^{-\psi^T \alpha}}$$

$$= \frac{e^{w_K^T \alpha} \cdot e^{-\psi^T \alpha}}{e^{-\psi^T \alpha} \sum_k e^{w_k^T \alpha}}$$

~~$e^{-\psi^T \alpha}$~~
 \sim
does not depend on K

$$= \frac{e^{w_K^T \alpha}}{\sum_k e^{w_k^T \alpha}}$$

$$\boxed{h(w - \psi) = h(w)}$$

∴ If we change any parameter vector $w_K \rightarrow w_K - \psi$
we get same hypothesis for softmax regression.

for two-class SR,

$$h = \frac{1}{\sum_k e^{w_k^T x}} \begin{bmatrix} e^{w_1^T x} \\ e^{w_2^T x} \end{bmatrix}$$

$$= \frac{1}{e^{\vec{w}_1^T x} + e^{\vec{w}_2^T x}} \begin{bmatrix} e^{\vec{w}_1^T x} \\ e^{\vec{w}_2^T x} \end{bmatrix}$$

use overparametrization property of SR

$$h(\vec{w} - \vec{\psi}) = h(\vec{w}) \quad \text{where } \vec{\psi} = \vec{w}_2$$

$$= \frac{1}{e^{(\vec{w}_1 - \vec{w}_2)^T x} + 1} \begin{bmatrix} e^{(\vec{w}_1 - \vec{w}_2)^T x} \\ 1 \end{bmatrix}$$

write $\vec{w}_2 - \vec{w}_1 = \vec{w}$

$$= \frac{1}{1 + e^{-\vec{w}^T x}} \begin{bmatrix} e^{-\vec{w}^T x} \\ 1 \end{bmatrix}$$

$$(h)_{SR} = \begin{bmatrix} \frac{e^{-\vec{w}^T x}}{1 + e^{-\vec{w}^T x}} \\ \frac{1}{1 + e^{-\vec{w}^T x}} \end{bmatrix} = (h)_{LR} \quad \#$$

proved

This is same as h for LR

Direct method to prove softmax ($K=2$) = LR

the hypothesis for softmax regression is

$$h(w, x) = \frac{1}{\sum_{k=1}^K e^{w_k^T x}}$$

$$\begin{bmatrix} e^{w_1^T x} \\ e^{w_2^T x} \\ \vdots \\ e^{w_K^T x} \end{bmatrix}$$

when $K=2$,

$$h = \frac{1}{e^{w_1^T x} + e^{w_2^T x}} \begin{bmatrix} e^{w_1^T x} \\ e^{w_2^T x} \end{bmatrix}$$

$$(h) = \text{softmax} \begin{bmatrix} \frac{e^{w_1^T x}}{e^{w_1^T x} + e^{w_2^T x}} \times \frac{e^{-w_2^T x}}{e^{-w_2^T x}} \\ \frac{e^{w_2^T x}}{e^{w_1^T x} + e^{w_2^T x}} \times \frac{e^{-w_1^T x}}{e^{-w_1^T x}} \end{bmatrix}$$

(20)

✓

$$= \begin{bmatrix} \frac{e^{-(w_2^T - w_1^T)x}}{1 + e^{-(w_2^T - w_1^T)x}} \\ \frac{1}{1 + e^{-(w_2^T - w_1^T)x}} \end{bmatrix} = \begin{bmatrix} \frac{e^{-w_1^T x}}{1 + e^{-w_1^T x}} \\ \frac{1}{1 + e^{-w_1^T x}} \end{bmatrix}$$

where
 $w_1^T = w_2^T - w_1^T$

or,

$$(h)_{\text{softmax}} = \begin{bmatrix} \frac{e^{-w^T x}}{1 + e^{-w^T x}} \\ \vdots \\ \frac{1}{1 + e^{-w^T x}} \end{bmatrix}$$

This is (h) Logistic Regression

$\therefore h_{\text{SR}} = (h)_{\text{LR}}$ proved!

on 6
QNG

derivative of cost for softmax Regr

for softmax Regression,

prob of data x belonging to class k i.e.

posterior prob of class k given data x is,

posterior
prob
of
class
 k

$$p(k|x_n, w) = \frac{e^{w_k^T x_n}}{\sum_{k=1}^K e^{w_k^T x_n}} \quad (1)$$

BISHOP 4.3.4
p. 209 / 227

rename class k by target t_n , then the probability
that the n th example x_n belongs to target t_n is given by,

$$p(t_n|x_n, w) = \frac{e^{w_{t_n}^T x_n}}{\sum_{n=1}^N e^{w_{t_n}^T x_n}} \quad (2)$$

then likelihood

$$l(w) = \prod_{n=1}^N p(t_n|x_n, w) \quad (3)$$

The -ve log likelihood is,

$$-\ln l(w) = -\ln \prod_{n=1}^N \left(\frac{e^{w_{t_n}^T x_n}}{\sum_{n=1}^N e^{w_{t_n}^T x_n}} \right)$$

cost
for
data

$$E_0 = \frac{1}{N} \cdot (-\ln l(w)) = -\frac{1}{N} \sum_{n=1}^N \ln \left(\frac{e^{w_{t_n}^T x_n}}{\sum_{n=1}^N e^{w_{t_n}^T x_n}} \right) \quad (3)$$

Weight vector for
each N examples

here, t_n is $t_1, t_2, t_3, \dots, t_N$ (single table)
 x_n is $x_{1,n}, x_{2,n}, x_{3,n}, \dots, x_{N,n}$ (x is row vector)
 w is w_1, w_2, \dots, w_N (w is column)
 (if t_n is row w is column) should be single table

here, instead of $\vec{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{bmatrix}$ we can group

the data so that there are only K weight vectors $w = [w_1 \ w_2 \ \dots \ w_K]$, then,

$$p(c_k | x) = \frac{e^{w_k^T x}}{\sum_{k=1}^K e^{w_k^T x}} \quad (1)$$

C regrouping weight vectors for each K classes, we can rewrite cost f(x)

and,

$$E_D(w) = -\frac{1}{N} \sum_{n=1}^N \left[\sum_{k=1}^K \delta_{k(t_n)} \ln \left(\frac{e^{w_k^T x_n}}{\sum_{k=1}^K e^{w_k^T x_n}} \right) \right]$$

using w_k
c weight vector
for each K classes

$$= -\frac{1}{N} \sum_{n=1}^N \sum_{k=1}^K \delta_{k(t_n)} \left[w_k^T x_n - \ln \sum_{k=1}^K e^{w_k^T x_n} \right]$$

$$E_D = -\frac{1}{N} \sum_{n=1}^N \left[\sum_{k=1}^K \delta_{k(t_n)} w_k^T x_n - \sum_{k=1}^K \delta_{k(t_n)} \ln \left(\sum_{k=1}^K e^{w_k^T x_n} \right) \right]$$

$$\frac{\partial E_D}{\partial w_j} = -\frac{1}{N} \sum_{n=1}^N \left[\frac{\partial}{\partial w_j} \sum_{k=1}^K \delta_{k(t_n)} w_k^T x_n - \sum_{k=1}^K \delta_{k(t_n)} \ln \left(\sum_{k=1}^K e^{w_k^T x_n} \right) \right]$$

here w_j operates only when $w_j = w_k$ and summation vanishes

$$\frac{\partial E_D}{\partial w_j} = -\frac{1}{N} \sum_{n=1}^N \left[\delta_j(t_n) \alpha_n - \delta_j(t_n) \cdot \frac{1 \cdot e^{w_j^T \alpha_n}}{\sum_{k=1}^K e^{w_k^T \alpha_n}} \cdot \delta_j(t_n) \alpha_n \right]$$

$$\frac{\partial E_D}{\partial w_K} = -\frac{1}{N} \sum_{n=1}^N \left[\delta_{1K}(t_n) \alpha_n - \alpha_n \cdot \delta_{1K}(t_n) \frac{e^{w_K^T \alpha_n}}{\sum_{k=1}^K e^{w_k^T \alpha_n}} \right]$$

$\delta_{K(t_n)} \cdot \delta_{1K(t_n)}$
 $p(c_k | \alpha_n)$ (eq 1)
 we change dummy index from $j \rightarrow K$
 $p(c_k)$ is prob only when label is K
 so basically $\delta_{K(t_n)}$ is 1 for $p(c_k)$

(2) ✓

$$= -\frac{1}{N} \sum_{n=1}^N [\delta_K(t_n) \alpha_n - \alpha_n p(c_K | \alpha_n)]$$

gradient of E_D

$$\boxed{\nabla_w E_D = -\frac{1}{N} \sum_{n=1}^N (\delta_K(t_n) - p(c_K | \alpha_n)) \alpha_n}$$

(A)

$$\text{Again } \nabla_w E_w(w) = \frac{\partial}{\partial w} \frac{1}{2} w_K^T w_K = \frac{1}{2} \cdot 2w_K^T = \alpha w_K^T$$

$$\boxed{\nabla_w E_w = \alpha w_K^T}$$

(B)

~~Ans~~

$$\therefore \nabla_w E = \nabla_w E_D + \nabla_w E_w = -\frac{1}{N} \sum_{n=1}^N (\delta_K(t_n) - p(c_K | \alpha_n)) \alpha_n + \alpha w_K^T$$

$$\text{method 2} \quad E_D(w) = -\frac{1}{N} \sum_n (\frac{e^{w^T x_n}}{\sum_k e^{w^T x_k}}) \quad (\text{grouping n-means})$$

$$E_D(w) = -\frac{1}{N} \sum_{n=1}^N \sum_{k=1}^K \delta_{k(t_n)} \ln \left(\frac{e^{w^T x_n}}{\sum_{k=1}^K e^{w^T x_n}} \right) \quad (\text{grouping K-means})$$

$$\text{let, } E_n = \sum_{k=1}^K \delta_{k(t_n)} \ln \left(\frac{e^{w^T x_n}}{\sum_k e^{w^T x_n}} \right)$$

$$E_n = \sum_k \delta_{k(t_n)} w_i^T x_n - \sum_k \delta_{k(t_n)} \ln \left(\sum_k e^{w^T x_n} \right)$$

$$\frac{\partial E_n}{\partial w_j} = \delta_{j(t_n)} x_n - \delta_{j(t_n)} \cdot \frac{1}{\sum_j e^{w_j^T x_n}} \cdot e^{w_j^T x_n} \cdot \delta_{j(t_n)} x_n$$

$$= \delta_{j(t_n)} x_n - \frac{e^{w_j^T x_n}}{\sum_j e^{w_j^T x_n}} x_n$$

$$\frac{\partial E_n}{\partial w_k} = \delta_{k(t_n)} x_n - \frac{e^{w_k^T x_n}}{\sum_k e^{w_k^T x_n}} x_n$$

$$\boxed{\frac{\partial E_n}{\partial w_k} = (\delta_{k(t_n)} - p(c_k | x_n)) x_n}$$

$$\therefore \nabla_{w_k} E_D(w) = -\frac{1}{N} \sum_n [\delta_{k(t_n)} - p(c_k | x_n)] x_n \quad @$$

also,

$$\boxed{E_w(w) = \frac{\alpha}{2} w_k^T w_k}$$

$$\nabla_{w_k} E_w(w) = \frac{\partial}{\partial w_k} \frac{\alpha}{2} w_k^T w_k = \alpha w_k$$

$$\therefore \nabla E = -\frac{1}{N} \sum_{n=1}^N (\delta_{k(t_n)} - p(c_k | x_n)) x_n + \alpha w_k$$

$$\boxed{\nabla_w E = \alpha \vec{w}_k - \frac{1}{N} \sum_n (\delta_{k(t_n)} - p(c_k | x_n)) \vec{x}_n}$$

Ans

HWS

Bhishan Paudel

142
150

HW Assignment 5 (Due by 10:30am on Nov 2)

1 Theory (110 points)

1. [Properties of Linear Discriminants, 20 points]

We have proven in class that the distance between origin and the decision hyperplane $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 = 0$ is equal with $-w_0 / \|\mathbf{w}\|$. Prove that the margin between a point \mathbf{x} and the same decision hyperplane is equal with $h(\mathbf{x}) / \|\mathbf{w}\|$.

2. [Bonus, 20 points]

Prove the two properties above for the general n -dimensional case.

3. [Fisher Criterion and Least Squares, 30 points]

Show that the Fisher criterion can be written in the vectorized form shown below:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

4. [Fisher Criterion (*), 20 points]

Reference the PRML Chapter 4 material available on Blackboard under Content.

Using the definitions of the between-class and within-class covariance matrices given by (4.27) and (4.28), respectively, together with (4.34) and (4.36) and the choice of target values described in Section 4.1.5, show that the expression (4.33) that minimizes the sum-of-squares error function can be written in the form (4.37).

5. [Perceptrons, 40 points]

Consider a training set that contains the following 8 examples:

\mathbf{x}	x_1	x_2	x_3	$t(x)$
$\mathbf{x}^{(1)}$	0	0	0	+1
$\mathbf{x}^{(2)}$	0	1	0	+1
$\mathbf{x}^{(3)}$	1.5	0	-1.5	+1
$\mathbf{x}^{(4)}$	1.5	1	-1.5	+1
$\mathbf{x}^{(5)}$	1.5	0	0	-1
$\mathbf{x}^{(6)}$	1.5	1	0	-1
$\mathbf{x}^{(7)}$	0	0	-1.5	-1
$\mathbf{x}^{(8)}$	0	1	-1.5	-1

(a) Prove that the perceptron algorithm does not converge on this dataset. Do not forget to include the bias.

(b) Consider a kernel perceptron that uses a polynomial kernel $k(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x}^T \mathbf{y})^d$. What is the smallest degree d for which the kernel perceptron would converge on this dataset?

6. [Perceptrons, 10 points]

A kernel perceptron for binary classification is run for a number of epochs E on a training dataset containing N examples, resulting in the dual parameters $\alpha_1, \alpha_2, \dots, \alpha_N$. What is the total number of mistakes that are made during training?

7. [Matrix Computations, 10 points]

Let $U \in R_{k \times m}$ and $X \in R_{n \times m}$. Let u_i and x_i be the i -th columns of U and X , respectively, for $1 \leq i \leq m$. Prove that $UX^T = \sum_{i=1}^m u_i x_i^T$.

2 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. On this theory assignment, **clear and complete explanations and proofs of your results are as important as getting the right answer.**

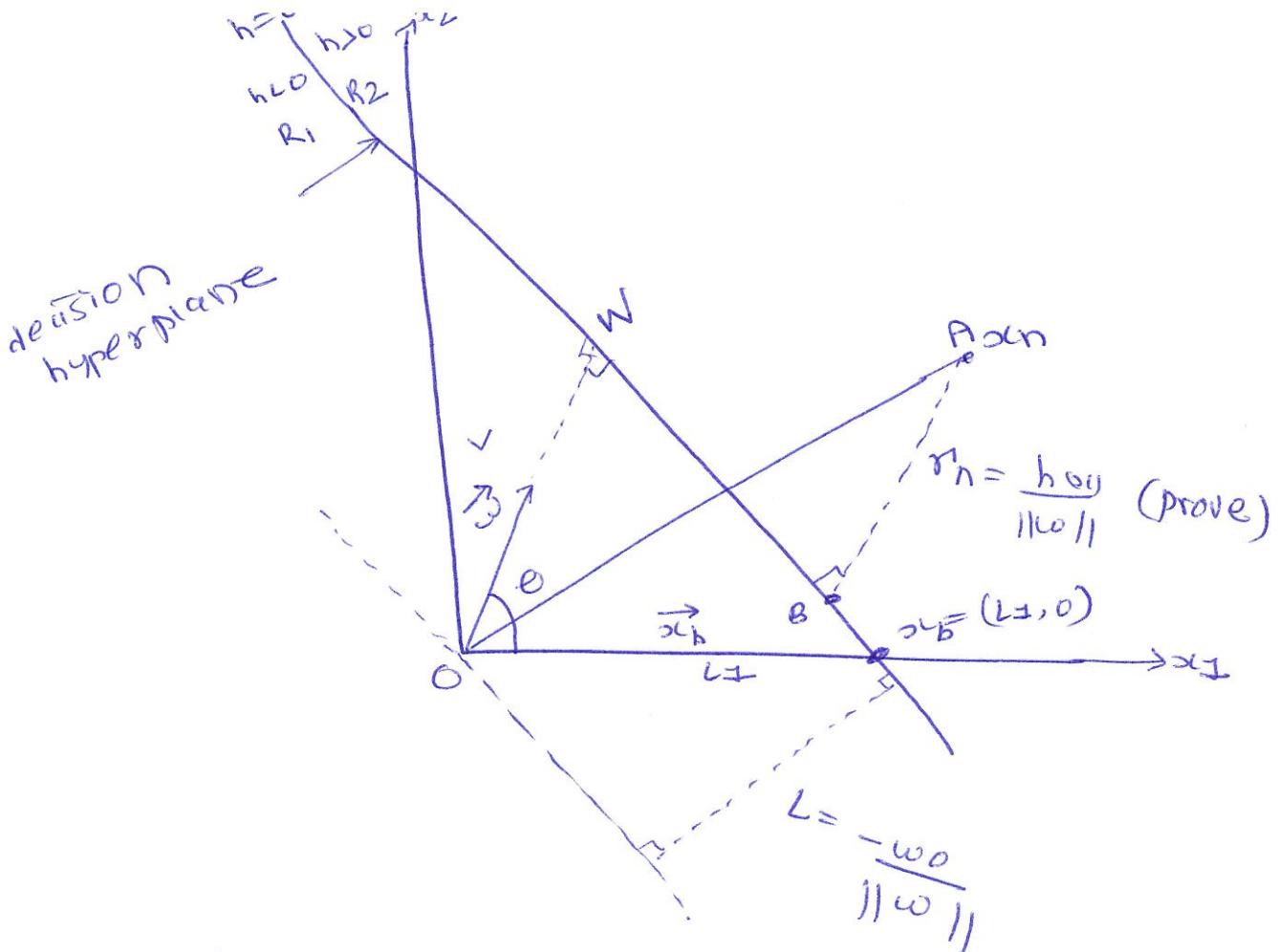
Hw 05

Bhishan Poudel

QNT

prove:

$$\text{dist betw } \underline{\text{data point}} \& \underline{\text{decision hyp.}} = \frac{\|w\|}{\|w\|}$$



(a) move $L_1 = -\frac{w_0}{\|w\|}$

Here, $\vec{w} = [w_1, w_2]$

$$\vec{x}_b = [x_1, x_2] = [L_1, 0]$$

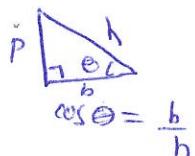
$$\vec{w} \cdot \vec{x}_b + w_0 = 0 = w_1 x_1 + w_2 x_2 + w_0$$

$$0 = w_1 L_1 + w_2 \cdot 0 + w_0$$

$$\boxed{L_1 = -\frac{w_0}{w_1}} \quad \text{--- ①}$$

again,

$$\cos \theta = \frac{\text{base}}{\text{hypotenuse}} = \frac{L}{L_1}$$



$$\cos \theta = \frac{L}{-\omega_0 / \omega_1}$$

$$\boxed{L = -\frac{\omega_0 \cos \theta}{\omega_1}} \quad \text{--- ②}$$

$$L = -\frac{w_0}{\|w\|} \cdot \frac{\vec{w} \cdot \vec{z}_b}{\|\vec{w}\| \|z_b\|} \quad ; \quad \vec{a} \cdot \vec{b} = \|a\| \|b\| \cos \theta$$

$$= -\frac{w_0}{\|w\|} \cdot \frac{\cancel{(w_1 w_2) + (\cancel{w_1}) \cancel{w_2})}}{\|\vec{w}\| \|z_b\|}$$

$$= -\frac{w_0}{\|w\|} \frac{w_1 k_1}{\|\vec{w}\| \|z_b\|}$$

$L = -\frac{w_0}{\|\vec{w}\|}$

Ans

Again, To find the coordinate of point B,

$$\vec{OB} = \vec{OA} - \vec{AB}$$

$$\vec{B} = \vec{x}_n - n_n \vec{w} \quad \rightarrow \vec{w} \text{ is unit vector perpendicular to decision hyperplane}$$

$B = x_n - n_n \frac{w}{\|w\|}$

But, point B lies in decision boundary, so,

$$\vec{w} \cdot \vec{B} + w_0 = 0$$

$$w^T (x_n - n_n \frac{w}{\|w\|}) + w_0 = 0$$

$$\text{or, } w^T x_n - n_n \frac{w^T w}{\|w\|} + w_0 = 0$$

$$\text{or, } w^T x_n - n_n / \|w\| + w_0 = 0$$

$$\therefore w^T w = \|w\|^2$$

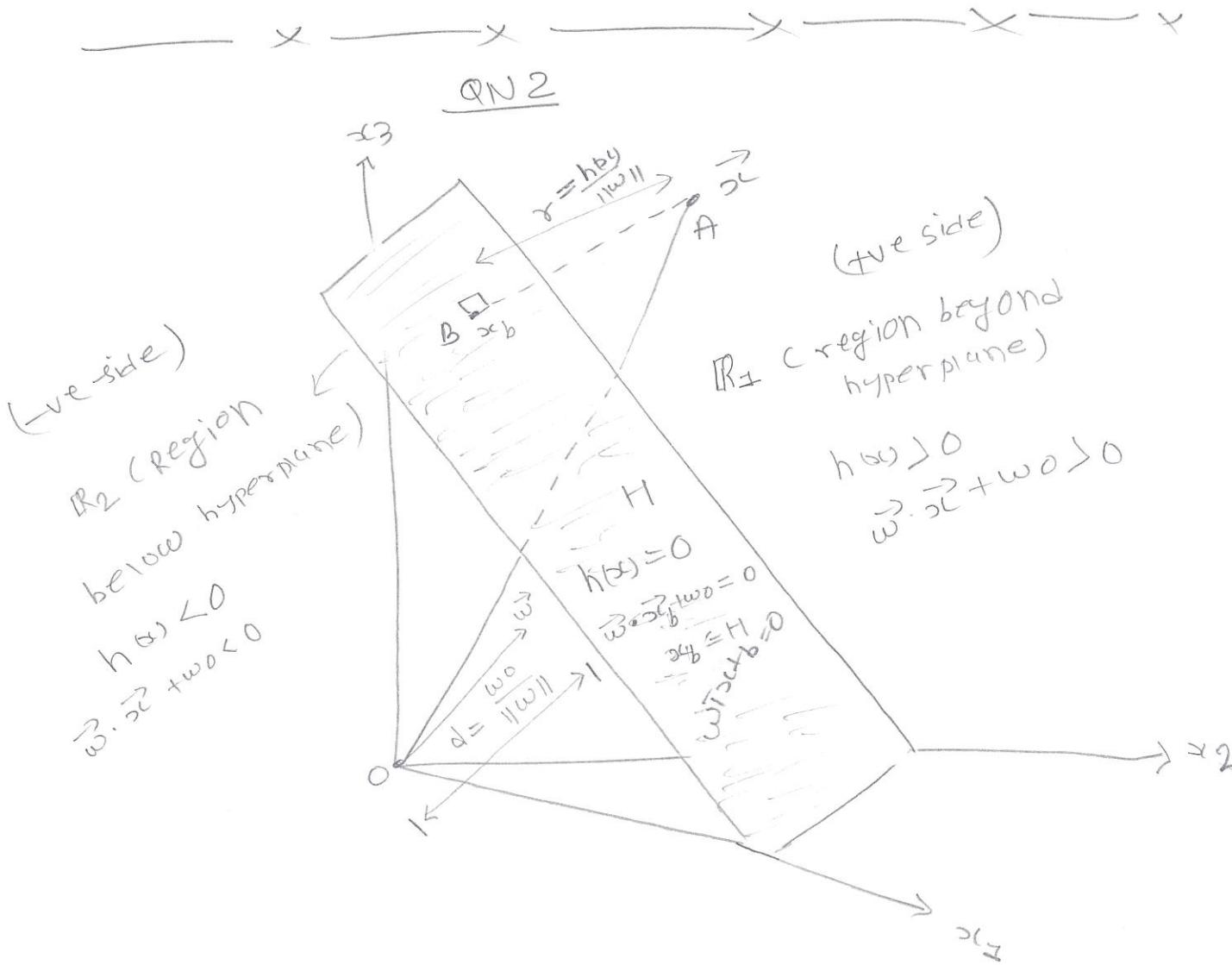
$$\text{or, } \vec{w}^T \vec{x}_n + w_0 = \gamma_n \|\vec{w}\|$$

$$\therefore \gamma_n = AB = \frac{\vec{w}^T \vec{x}_n + w_0}{\|\vec{w}\|}$$

where $h_{\vec{w}} = \vec{w}^T \vec{x} + w_0$

$$\boxed{\gamma_n = \frac{h_{\vec{w}}}{\|\vec{w}\|}} \quad \text{proved!}$$

✓ 20



QN2

For n-dimensional case,
prove:

$$\text{dist betn origin and decision hyperplane} = -\frac{w_0}{\|w\|}$$

$$\& \text{dist betn datapoint and decision hyperplane} = \frac{h(x)}{\|w\|}$$

Soln Let \vec{x} be any point on the positive side
of ^{decision} boundary (hyperplane), then,

$$\vec{x} = \vec{x}_b + r \frac{\vec{w}}{\|\vec{w}\|} \quad \text{where } \frac{\vec{w}}{\|\vec{w}\|} \text{ is unit vector}$$

$$\boxed{\vec{x}_b = \vec{x} - r \frac{\vec{w}}{\|\vec{w}\|}} \quad \text{①}$$

but, \vec{x}_b lies in boundary, we need to find r ,

$$\boxed{\vec{w} \cdot \vec{x}_b + w_0 = 0} \quad \text{②}$$

$$\text{or, } w^T (\vec{x} - r \frac{\vec{w}}{\|\vec{w}\|}) + w_0 = 0 = w^T \vec{x} - r \frac{w^T \vec{w}}{\|\vec{w}\|} + w_0 = 0$$

$$\text{if, } w^T \vec{x} + w_0 = r \|\vec{w}\| = h(x)$$

distance from datapoint to hyperplane $\Rightarrow r = \frac{w^T \vec{x} + w_0}{\|\vec{w}\|}$

DNS

Also, we have to show the distance of hyperplane from the origin,

$$\vec{OB} = -\frac{\underline{w}_0}{\|w\|} \vec{OB}$$

Here, the point x_b lies in decision hyperplane, so,

$$\vec{w} \cdot \vec{x}_b + w_0 = 0$$

$$w \cdot x_b + w_0 = 0$$

$$w \cdot x_b = -w_0$$

distance from
origin to
decision hyperplane

$$x_b = -\frac{\underline{w}_0}{\|w\|}$$

Ans

QN3

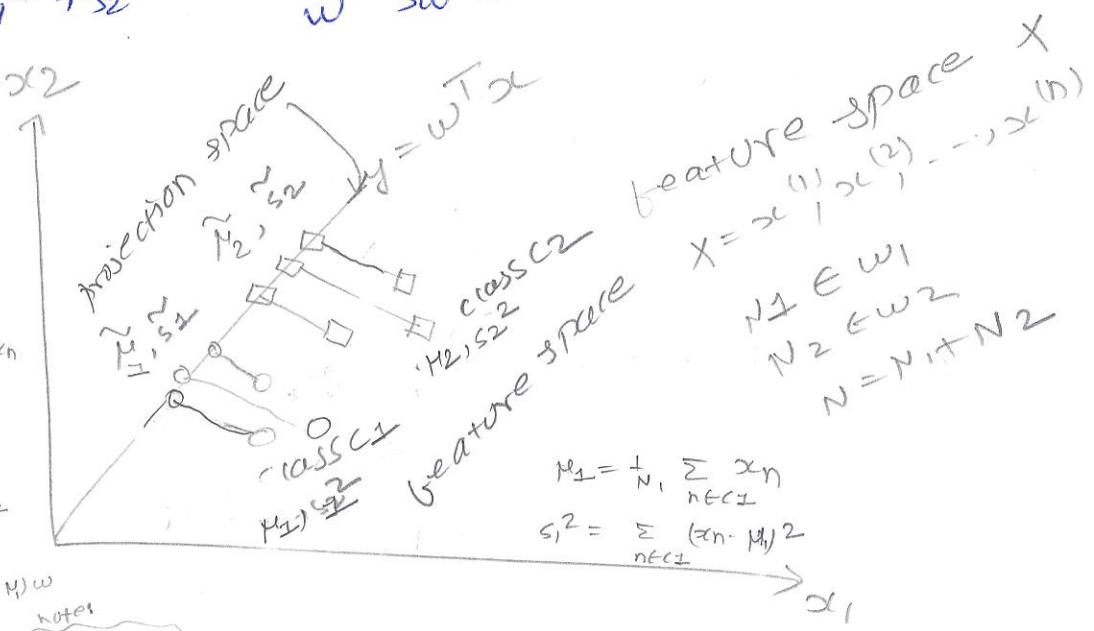
show that Fisher criterion

Bishop
(ex 4.15) can be written as,

$$J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{w^T S_B w}{w^T S_w w}$$

Solution

$$\begin{aligned}\hat{\mu}_1 &= \frac{1}{N_1} \sum_{n \in C_1} w^T x_n \\ &= w^T \mu_1 \\ s_1^2 &= \sum_{n \in C_1} (y_n - \hat{\mu}_1)^2 \\ &= \sum_{n \in C_1} (w^T x_n - w^T \mu_1)^2 \\ &= \sum_{n \in C_1} w^T (x_n - \mu_1)^\top (x_n - \mu_1) w \\ \hat{\mu}_2 &= w^T \mu_2 \\ (\text{big } S)\end{aligned}$$



In feature space X , we have input data,

$$x = (x^{(1)}, x^{(2)}, \dots, x^{(n)})$$

feature space X ,

mean

$$\mu_i = \frac{1}{N_i} \sum_{x \in C_i} x$$

projection space Y , mean $\tilde{\mu}_i = \frac{1}{N_i} \sum_{y \in C_i} y$

$$= \frac{1}{N_i} \sum_{x \in C_i} w^T x$$

$$\tilde{\mu}_i = w^T \mu_i$$

Also,

Variance,
(feature space)

Variance
(projection
space)

$$\sigma_i^2 = \frac{1}{N_i} \sum_{x \in \mathcal{E}_i} (\alpha - \mu_i)^2$$

$$\tilde{\sigma}_i^2 = \frac{1}{N_i} \sum_{y \in \mathcal{E}_i} (y - \tilde{\mu}_i)^2$$

Variance

Also, scatter in feature space,

$$S_i^2 = \sum_{x \in \mathcal{E}_i} (\alpha - \mu_i)^2 = \sum_{x \in \mathcal{E}_i} (\alpha - \mu_i)(\alpha - \mu_i)^T$$

Scatter in projection space,

$$\tilde{S}_i^2 = \sum_{y \in \mathcal{E}_i} (y - \tilde{\mu}_i)^2$$

Scatter

Also,

within-class scatter

$$S_w = S_1 + S_2$$

Also,

$$\tilde{S}_w = \tilde{S}_1 + \tilde{S}_2$$

Now, we define Fisher's criterion,

$$J(w) = \frac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

Fisher's
linear
discriminant

NOW,
 $(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = (\bar{w}^T \mu_1 - \bar{w}^T \mu_2)^2$
 $= \bar{w}^T (\mu_1 - \mu_2) (\mu_1 - \mu_2)^T w$

$$(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = \bar{w}^T S_B w \quad \textcircled{A}$$

where, $S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$

is between-class scatter.

Also, $\tilde{s}_i^2 = \sum_{y \in \mathcal{C}_i} (y - \tilde{\mu}_i)^2$
 $= \sum_{x \in \mathcal{C}_i} (\bar{w}^T x - \bar{w}^T \mu_i)^2$

$$= \sum_{x \in \mathcal{C}_i} \bar{w}^T (x - \mu_i) (\bar{x} - \bar{\mu}_i)^T w$$

$$\tilde{s}_i^2 = \bar{w}^T S_i w$$

where, feature-space scatter

$$S_i = \sum_{x \in \mathcal{C}_i} (x - \mu_i)(x - \mu_i)^T$$

$$\Rightarrow \tilde{s}_1^2 = \bar{w}^T S_1 w$$

$$\tilde{s}_2^2 = \bar{w}^T S_2 w$$

$$\tilde{s}_1^2 + \tilde{s}_2^2 = \bar{w}^T (S_1 + S_2) w = \bar{w}^T S_w w$$

where $S_w = S_1 + S_2$ is within-class scatter.

$$\Rightarrow \boxed{\xi_1^2 + \xi_2^2 = \omega^T S_w \omega} \quad \text{②}$$

Using equations ① & ②, we have,

$$J(\omega) = \frac{|\tilde{\mu}_1 - \tilde{\mu}_2|^2}{\xi_1^2 + \xi_2^2}$$

$$\boxed{J(\omega) = \frac{\omega^T S_B \omega}{\omega^T S_w \omega}}$$

Q.E.D.

30

(Q4)

In Bishop Book, (P. 190 or 208)

Bishop use equations: 4.27, 4.28, 4.34, 4.36, 4.33
 ex 4.6
 P.

Derive: $\left(S_w + \frac{N_1 N_2}{N} S_B \right) w = N (m_1 - m_2)$

4.37

Here, given, SSE is given by

$$E(w) = \frac{1}{2} \sum_{n=1}^N (w^T x_n + w_0 - t_n)^2$$

— (4.31) P. 190/208

$$\frac{\partial E}{\partial w} = 0 = \sum_{n=1}^N (w^T x_n + w_0 - t_n) x_n \quad — (4.33)$$

(4.34)
 $w_0 = -w^T m$

$$t_n = \begin{cases} \frac{N_1}{N} & \text{if } x_n \in C_1 \\ -\frac{N_2}{N} & \text{if } x_n \in C_2 \end{cases}$$

so that $\sum_{n=1}^N t_n = N_1 \frac{N}{N} - N_2 \frac{N}{N}$

mean of the total dataset (population mean)

(4.36)
 $m = \frac{1}{N} \sum_{n=1}^N x_n = \frac{1}{N} N_1 m_1 + N_2 m_2$

$$\sum_{n=1}^N t_n = 0$$

where, $m_1 = \frac{1}{N_1} \sum_{n \in C_1} x_n$ (sample mean of class 1)
 (mean of x_n belonging to class C_1)

(4.22)
 P. 205
 $m_2 = \frac{1}{N_2} \sum_{n \in C_2} x_n$

$$\Rightarrow \sum_{n \in C_1} x_n = N_1 m_1$$

$$\Rightarrow \sum_{n \in C_2} x_n = N_2 m_2$$

we define within-class scatter as,

$$eq \text{ 4.28} \\ p. 207 \boxed{S_w = \sum_{n \in C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n \in C_2} (x_n - m_2)(x_n - m_2)^T}$$

and, between-class scatter as,

$$eq \text{ 4.29} \\ p. 208 \boxed{S_B = (m_2 - m_1)(m_2 - m_1)^T}$$

we have to prove,

~~to prove~~

$$(S_w + \frac{N \cdot N_2}{N} S_B) w = N(m_1 - m_2) \quad (eq \text{ 4.37})$$

$$\Rightarrow \frac{N(m_1 - m_2)}{w} = S_w + \frac{N \cdot N_2 S_B}{N} \quad \square$$

we start from gradient zero equation (4.33),

$$0 = \sum_{n=1}^N (w^T x_n - w^T m - t_n) x_n$$

$$= \sum_{n=1}^N (w^T x_n - w^T m - t_n) x_n$$

$$= \sum_{n=1}^N w^T (x_n x_n^T - m x_n) - \sum_{n=1}^N t_n x_n$$

$$= \sum_{n \in C_1} (x_n x_n^T - x_n m^T) w - \sum_{n \in C_1} t_n x_n$$

use $t_n = \begin{cases} N_1 & \text{for } C_1 \\ -\frac{N_2}{N_2} & \text{for } C_2 \end{cases}$

$$+ \sum_{n \in C_2} (x_n x_n^T - x_n m^T) w - \sum_{n \in C_2} t_n x_n$$

$m_2 = \frac{1}{N_2} \sum_{n \in C_2} x_n \Rightarrow \sum_{n \in C_2} x_n = N_2 m_2$

$$0 = \left(\sum_{n \in C_1} x_n x_n^T - N_1 m_1 m_1^T \right) w - \frac{N_1}{N} m_1 \\ \left(\sum_{n \in C_2} x_n x_n^T - N_2 m_2 m_2^T \right) w + \frac{N_2}{N} m_2$$

$$0 = \left(\sum_{n \in C_1} x_n x_n^T + \sum_{n \in C_2} x_n x_n^T - (N_1 m_1 + N_2 m_2) m^T \right) w - N(m_1 - m_2)$$

$$\frac{N(m_1 - m_2)}{w} = \underbrace{\sum_{n \in C_1} x_n x_n^T + \sum_{n \in C_2} x_n x_n^T}_{\text{write in terms of } S_w} - (N_1 m_1 + N_2 m_2) m^T \quad \textcircled{2}$$

$$m = \frac{N_1 m_1 + N_2 m_2}{N}$$

NOW, we expand the first term in S_w ,

$$\begin{aligned} \sum_{n \in C_1} (x_n - m_1) (x_n - m_1)^T &= \sum_{n \in C_1} (x_n x_n^T - x_n m_1^T - m_1 x_n^T + m_1 m_1^T) \\ &= \sum_{n \in C_1} x_n x_n^T - m_1^T \sum_{n \in C_1} x_n - m_1 \sum_{n \in C_1} x_n^T + \sum_{n \in C_1} m_1 m_1^T \\ &= \sum_{n \in C_1} x_n x_n^T - N_1 m_1 m_1^T - N_1 m_1 m_1^T + N_1 m_1 m_1^T \end{aligned}$$

$$\boxed{\sum_{n \in C_1} (x_n - m_1) (x_n - m_1)^T = \sum_{n \in C_1} x_n x_n^T - N_1 m_1 m_1^T}$$

$$\text{So, } S_w = \sum_{n \in C_1} (x_n - m_1) (x_n - m_1)^T + \sum_{n \in C_2} (x_n - m_2) (x_n - m_2)^T$$

$$\boxed{S_w = \sum_{n \in C_1} x_n x_n^T + \sum_{n \in C_2} x_n x_n^T - N_1 m_1 m_1^T - N_2 m_2 m_2^T}$$

$$\Rightarrow \sum_{n \in C_1} x_n x_n^T + \sum_{n \in C_2} x_n x_n^T = S_w + N_1 m_1 m_1^T + N_2 m_2 m_2^T$$

then eqn ② becomes

$$\frac{N(m_1 - m_2)}{w} = sw + \underbrace{N_1 m_1 m_1^T}_{\text{1}} + \underbrace{N_2 m_2 m_2^T}_{\text{2}} - (N_1 m_1 + N_2 m_2) \cdot \underbrace{(N_1 m_1 + N_2 m_2)}_{N}^T$$

$$= sw + m_1 m_1^T \left(N_1 - \frac{N_1^2}{N} \right) + m_2 m_2^T \left(N_2 - \frac{N_2^2}{N} \right)$$

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

$$N = N_1 + N_2$$

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

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

$$= sw + \frac{N_1 N_2}{N} (m_1 m_1^T + m_2 m_2^T)$$

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

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

$$\frac{N(m_1 - m_2)}{w} = sw + \frac{N_1 N_2}{N} s_B$$

$$\begin{aligned} s_B &= (m_2 - m_1)(m_2 - m_1)^T \\ &= m_2 m_2^T - m_2 m_1^T - m_1 m_2^T + m_1 m_1^T \\ &= m_1 m_1^T + m_2 m_2^T - m_1 m_2^T - m_2 m_1^T \end{aligned}$$

$$\therefore \boxed{\left(sw + \frac{N_1 N_2}{N} s_B \right) w = N(m_1 - m_2)}$$

Q.E.D.

✓

20

ANS

Given training set,

x	x_1	x_2	x_3	$t(x)$
$x^{(1)}$	0	0	0	+1
$x^{(2)}$	0	1	0	+1
$x^{(3)}$	1.5	0	-1.5	+1
$x^{(4)}$	1.5	1	-1.5	+1
$\bar{x}^{(5)}$	1.5	0	0	-1
$x^{(6)}$	1.5	1	0	-1
$x^{(7)}$	0	0	-1.5	-1
$x^{(8)}$	0	1	-1.5	-1

a) prove that the perceptron algorithm does not converge on this dataset. (Note: include bias term)

b) consider a kernel perceptron that uses a polynomial kernel $K(x,y) = (1+x^T y)^d$. What is the smallest degree d for which the kernel perceptron would converge on this dataset?

(50)

for a given example, example belongs to class + if

$$w_0 + \vec{w} \cdot \vec{x}_n \geq 0$$

and, example belongs to class - if

$$w_0 + \vec{w} \cdot \vec{x}_n < 0$$

now, $x^{(1)} = [w_1, w_2, w_3]$ $t^{(1)} = +1$

$$w_0 + w_1 \cdot 0 + w_2 \cdot 0 + w_3 \cdot 0 \geq 0$$

$$w_0 \geq 0$$

similarly,

w

we {

$$\begin{aligned} ① \quad w_0 + 0 + 0 + 0 &\geq 0 \quad \text{--- (1)} \\ ② \quad w_0 + 0 + w_2 + 0 &\geq 0 \quad \text{--- (2)} \\ ③ \quad w_0 + 1.5w_1 + 0 - 1.5w_3 &\geq 0 \quad \text{--- (3)} \\ ④ \quad w_0 + 1.5w_1 + w_2 - 1.5w_3 &\geq 0 \quad \text{--- (4)} \\ ⑤ \quad w_0 + 1.5w_1 + 0 + 0 &< 0 \quad \text{--- (5)} \\ ⑥ \quad w_0 + 1.5w_1 + w_2 + 0 &< 0 \quad \text{--- (6)} \\ ⑦ \quad w_0 + 0 + 0 - 1.5w_3 &< 0 \quad \text{--- (7)} \\ ⑧ \quad w_0 + 0 + w_2 - 1.5w_3 &< 0 \quad \text{--- (8)} \end{aligned}$$

(4) - (3) $\Rightarrow w_2 \geq 0$
 (6) - (5) $\Rightarrow w_2 \leq 0$

✓

here, (1) - (3) gives $w_2 \geq 0$

but (6) - (5) gives $w_2 \leq 0$

this is
contradictory

\Rightarrow
perceptron
fails.

(5b) Here, the given dataset is not linearly separable in x -space.

We shall propose a perceptron that uses polynomial kernel;

The polynomial kernel is,

$$K(\alpha_i, \alpha_j) = (\alpha_i^T \alpha_j + b)^d$$

for $n = b = 1$

$$K(\alpha_i, \alpha_j) = (1 + \alpha_i^T \alpha_j)^d$$

a = scaling factor

b = bias term

d = degree of polynomial

α = augmented dataset
of shape N rows
 $m+1$ cols

α' = one example of dataset
with $m+1$ features

when $d=1$: the kernel is linear classifier, but
our dataset is NON linear, so it can not
classify correctly. 12

When $d=2$: when degree is 2, quadratic kernel
can classify linearly non-separable
dataset. and find a vector to
separate the data set.

when $d \geq 2$: higher degree polynomial kernels can
fit the linearly non-separable dataset
good, but, they may overfit the
test dataset.
may
overfit

∴ smallest degree $d=2$ ANS

Qn 6

For a kernel perceptron,

from lecture note, (lecture 06)

the algorithm is

$$1. f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} = \sum_n \alpha_n t_n \mathbf{x}^T \mathbf{x} = \sum_n \alpha_n t_n K(\mathbf{x}, \mathbf{x})$$

kernel

2. initialized: $\alpha_n = 0$

for e in E epochs

3. for $n=1 \dots N$

$$h_n = \text{sgn}(f(\mathbf{x}_n))$$

if $h_n \neq t_n$ then

$$\alpha_n = \alpha_n + 1$$

Now, from step 3, if we run perceptron

E number of epochs,

then total number of mistakes made

so far if,

$$M = \sum_{n=1}^N \alpha_n$$

Ans

/D



Q6B

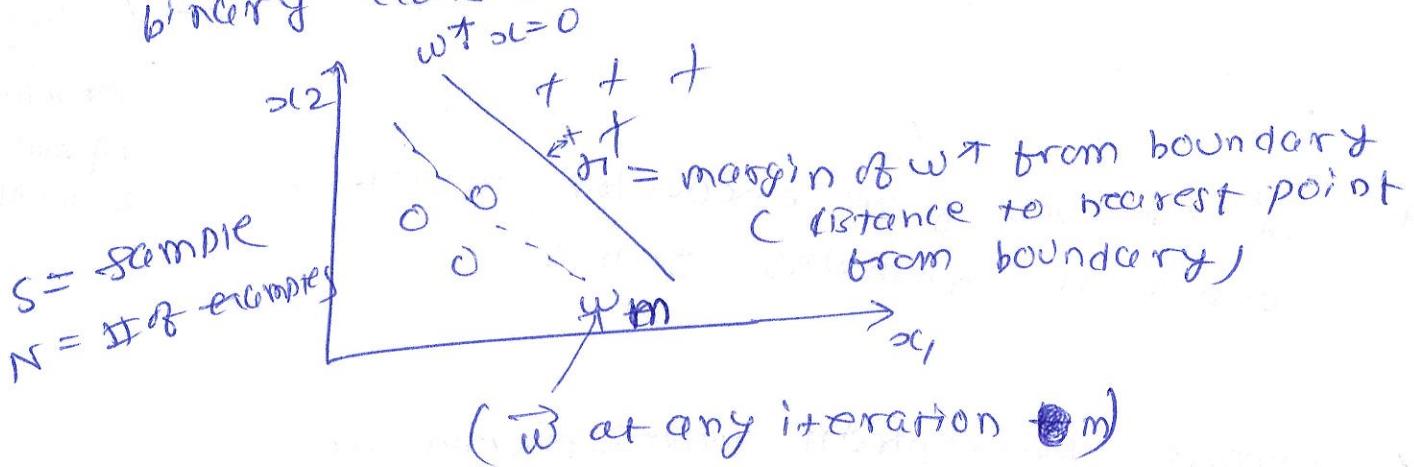
Method 2 (Extra practice) A Kernel perceptron for binary classification

- maximum error bound

classification is run for a number of epochs E on a training dataset containing N examples, resulting in the dual parameters $\alpha_1, \alpha_2, \dots, \alpha_N$.
What is the total number of mistakes that are made during training?

Soln consider a kernel perceptron for

binary classification.



Let there is a unit vector \vec{w}^T that can separate N training examples into two classes with margin γ_1 .

Let $R = \text{norm of maximum value of } \vec{x}^T$ in dataset

$$\|\vec{R}\| = \max_{x \in S} \|\vec{x}\|$$

perceptron algorithm:

1. initialize w to zero. $w_1 = [0 \ 0 \ 0 \ 0]$

2. for e in epochs

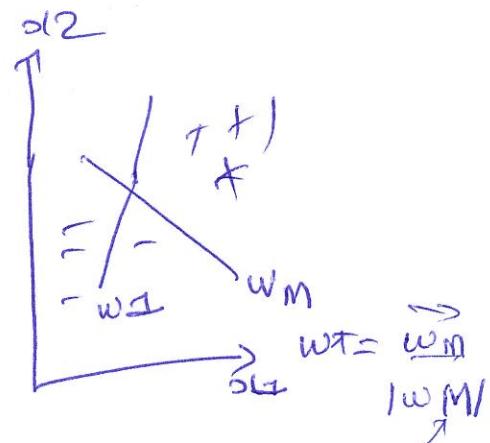
for $n = 1$ to N

if $t_n(w \cdot x_n) < 0$

$$w = w + t_n x_n$$

if converged w :

break



$wt = \frac{w_M}{\|w\|}$
for total
 M mistake
 $1 \leq m \leq M$

* \vec{w} changes if we encounter mistake, otherwise remains same.

* suppose perceptron makes a mistake at iteration m , then,

$$\vec{w}_m \cdot \vec{w}^* = (\vec{w}_{m-1} + t \vec{x}) \cdot \vec{w}^*$$

$$= \vec{w}_{m-1} \cdot \vec{w}^* + t (\vec{x} \cdot \vec{w}^*)$$

$$w_m \cdot w^* \geq \vec{w}_{m-1} \cdot \vec{w}^* + \gamma$$

from definition of γ .
each data point x
is $\geq \gamma$ from the
boundary w^*

$$\therefore \boxed{w_m \cdot w^T \geq w_{m-1} \cdot w^T + \gamma} \quad \text{--- (A)}$$

Also,

$$\|w_m\|^2 = \|w_{m-1} + t\vec{x}\|^2$$

$$= \|w_{m-1}\|^2 + 2t(w_{m-1} \cdot \vec{x}) + \|t\vec{x}\|^2$$

④ maximum is a
iteration

⑤ Here we have mistake at iteration

m , so the previous iteration

has hypothesis

$$t(w_{m-1} \cdot \vec{x}) < 0$$

$$\boxed{\|w_m\|^2 \leq \|w_{m-1}\|^2 + R^2} \quad \text{(from ④ & ⑤)}$$

Here, ~~④~~ inequalities ④ & ⑤ hold true
for all the iterations m ,

$$w_m \cdot w^T \geq w_{m-1} \cdot w^T + \gamma$$

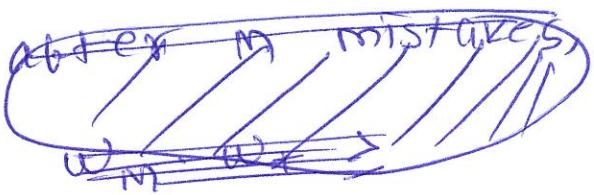
$$\|w_m\|^2 \leq \|w_{m-1}\|^2 + R^2$$

$$\|w_m\|^2 - \|w_{m-1}\|^2 \leq R^2$$

difference Δ

C If there is a mistake then square of weight vectors is less than R^2),

(for each mistake $w^2 - w_{m-1}^2$ cannot be larger than R^2)



for all iterations,

$$\vec{w}_m \cdot \vec{w}^\top \geq w_{m-1} + \gamma$$

after M mistakes,

$$\boxed{\vec{w}_M \cdot \vec{w}^\top \geq r_M} \quad \text{--- } \textcircled{C}$$

also, for all iterations,

$$\|\vec{w}_m\|^2 \leq \|\vec{w}_{m-1}\|^2 + R^2$$

after M mistakes

$$\boxed{\begin{aligned} \|\vec{w}_M\|^2 &\leq R^2 M \\ \|\vec{w}_M\| &\leq R \sqrt{M} \end{aligned}} \quad \text{--- } \textcircled{D}$$

so, from \textcircled{C} & \textcircled{D} ,

$$r_M \leq \vec{w}_M \cdot \vec{w}^\top$$

$$\leq \|\vec{w}_M\| \|\vec{w}^\top\|$$

$$\leq \|\vec{w}_M\|$$

$$r_M \leq R \sqrt{M}$$

$$\pi \sqrt{M} \leq R$$

$$\pi^2 M \leq R^2$$

$$\therefore \vec{a} \cdot \vec{b} = ab \cos \theta$$

~~cosine is less than~~

or equal to 1

\vec{w}^\top is unit vector of \vec{w}_M

$$\|\vec{w}^\top\| = 1$$

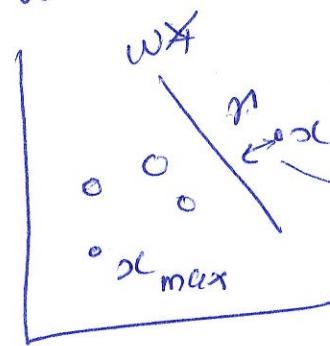
$$\boxed{M \leq \left(\frac{R}{\pi}\right)^2}$$

Ans

Hence,

let the data (training) sample S is,

$$\begin{array}{ll} \alpha_1 & t_1 \\ \alpha_2 & t_2 \\ \alpha_3 & t_3 \\ \vdots & \vdots \\ \alpha_N & t_N \end{array}$$



nearest point
to boundary
 $w \cdot \vec{\alpha} = 0$

example,

$$\|\alpha_{\max}\| = R$$

$$\begin{array}{lll} \alpha_1: 0 0 1 2 3 & t_1 = + \\ \alpha_1: 5 0 3 2 6 & t_2 = - \\ \alpha_N: 10 20 30 40 & t_N = + \end{array}$$

then, if we use Perceptron to train our model, the total number of mistakes (M)

made is,

$$M \leq \left(\frac{R}{\gamma_1}\right)^2$$

QNT

Let $V \in R_{k \times m}$

$X \in R^{n \times m}$

$v_i, x_i = i^{\text{th}} \text{ column of } V \text{ and } X \text{ respectively}$
 $1 \leq i \leq m$.

prove $VX^T = \sum_{i=1}^m v_i x_i^T$

SOLUTION:

$$V = V_{k,m} = \begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1m} \\ v_{21} & v_{22} & \cdots & v_{2m} \\ \vdots & & & \\ v_{k1} & v_{k2} & \cdots & v_{km} \end{pmatrix}_{k,m}$$

$$X = X_{n,m} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{pmatrix}_{n,m}$$

\uparrow
column vector

$$V = X^T = \begin{pmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ x_{12} & x_{22} & \cdots & x_{n2} \\ \vdots & \vdots & & \vdots \\ x_{1m} & x_{2m} & \cdots & x_{nm} \end{pmatrix} = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_m^T \end{pmatrix}_{m,1}$$

From definition of matrix multiplication,

$$\text{if } U = U_{km} \\ \text{and } V = V_{mn}$$

then,

$$(UV)_{pq} = \sum_{i=1}^m U_{pi} V_{iq} \quad \text{if } 1 \leq p \leq k \text{ and } 1 \leq q \leq n$$

$1 \leq p \leq k$
 $1 \leq q \leq n$

where $X = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix}_{n \times m}$

$n \text{ rows}$
 $m \text{ cols}$

$$x_1 = \begin{pmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{m1} \end{pmatrix}_{1 \times m}$$

$$= \left(\begin{array}{ccc} u_{11} & u_{12} & u_{1m} \\ u_{21} & u_{22} & u_{2m} \\ \vdots & \vdots & \vdots \\ u_{k1} & u_{k2} & u_{km} \end{array} \right)_{k \times m} \downarrow \left(\begin{array}{ccc} x_{11} & x_{21} & \dots & x_{m1} \\ x_{12} & x_{22} & \dots & x_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1m} & x_{2m} & \dots & x_{mm} \end{array} \right)_{m \times n}$$

$$= \left(\begin{array}{c} u_{11}x_{11} + u_{12}x_{12} + u_{1m}x_{1m} \\ u_{21}x_{21} + u_{22}x_{22} + u_{2m}x_{2m} \\ \vdots \\ u_{k1}x_{k1} + u_{k2}x_{k2} + u_{km}x_{km} \end{array} \right)_{k \times 1} + \left(\begin{array}{c} u_{11}x_{21} + u_{12}x_{22} + u_{1m}x_{2m} \\ u_{21}x_{21} + u_{22}x_{22} + u_{2m}x_{2m} \\ \vdots \\ u_{k1}x_{k1} + u_{k2}x_{k2} + u_{km}x_{km} \end{array} \right)_{k \times 1} + \left(\begin{array}{c} u_{11}x_{m1} + u_{12}x_{m2} + u_{1m}x_{mm} \\ u_{21}x_{m1} + u_{22}x_{m2} + u_{2m}x_{mm} \\ \vdots \\ u_{k1}x_{m1} + u_{k2}x_{m2} + u_{km}x_{mm} \end{array} \right)_{k \times 1}$$

$$= \left(\begin{array}{ccc} u_{11}x_{11} & u_{11}x_{21} & u_{11}x_{m1} \\ u_{21}x_{11} & u_{21}x_{21} & u_{21}x_{m1} \\ \vdots & \vdots & \vdots \\ u_{k1}x_{11} & u_{k1}x_{21} & u_{k1}x_{m1} \end{array} \right)_{k \times m} + \left(\begin{array}{ccc} u_{12}x_{12} & u_{12}x_{22} & u_{12}x_{m2} \\ u_{22}x_{12} & u_{22}x_{22} & u_{22}x_{m2} \\ \vdots & \vdots & \vdots \\ u_{k2}x_{12} & u_{k2}x_{22} & u_{k2}x_{m2} \end{array} \right)_{k \times m} + \left(\begin{array}{ccc} u_{1m}x_{1m} & u_{1m}x_{2m} & u_{1m}x_{mm} \\ u_{2m}x_{1m} & u_{2m}x_{2m} & u_{2m}x_{mm} \\ \vdots & \vdots & \vdots \\ u_{km}x_{1m} & u_{km}x_{2m} & u_{km}x_{mm} \end{array} \right)_{k \times m}$$

$$= U_1 X_1^T + U_2 X_2^T + U_m X_m^T$$

10 ✓

similarly,

$$U_1 X_1^T = \begin{pmatrix} U_{11} \\ U_{21} \\ \vdots \\ U_{K1} \end{pmatrix} \begin{pmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{n1} \end{pmatrix}^T$$

$$= \begin{pmatrix} \rightarrow \\ U_{11} \\ U_{21} \\ \vdots \\ U_{K1} \end{pmatrix} \downarrow (x_{11} \ x_{21} \ \dots \ x_{n1})_{1, n}$$

$$U_1 X_1^T = \begin{pmatrix} U_{11} x_{11} & U_{11} x_{21} & U_{11} x_{n1} \\ U_{21} x_{11} & U_{21} x_{21} & U_{21} x_{n1} \\ \vdots & \vdots & \vdots \\ U_{K1} x_{11} & U_{K1} x_{21} & U_{K1} x_{n1} \end{pmatrix}_{1, n} \quad \text{--- (a)}$$

similarly,

$$U_2 X_2^T = \begin{pmatrix} U_{12} x_{12} & U_{12} x_{22} & U_{12} x_{n2} \\ U_{22} x_{12} & U_{22} x_{22} & U_{22} x_{n2} \\ \vdots & \vdots & \vdots \\ U_{K2} x_{12} & U_{K2} x_{22} & U_{K2} x_{n2} \end{pmatrix}_{1, n} \quad \text{--- (b)}$$

and,

$$U_m X_m^T = \begin{pmatrix} U_{1m} x_{1m} & U_{1m} x_{2m} & U_{1m} x_{nm} \\ U_{2m} x_{1m} & U_{2m} x_{2m} & U_{2m} x_{nm} \\ \vdots & \vdots & \vdots \\ U_{Km} x_{1m} & U_{Km} x_{2m} & U_{Km} x_{nm} \end{pmatrix} \quad \text{--- (c)}$$

$$\therefore U X^T = U_1 X_1^T + U_2 X_2^T + \dots + U_m X_m^T$$

$$\boxed{U X^T = \sum_{i=1}^m U_i X_i^T} \quad \text{Q.E.D.}$$



HW Assignment 7 (Due by 10:30am on Nov 30)

1 Theory (140 points)

1. [Normalized Kernels, 50 + 25 points]

Let $K : X \times X \rightarrow R$ be a kernel function defined over a sample space X .

- (a) Prove that the function below (a *normalized kernel*) is a valid kernel.

$$\frac{K(x, y)}{\sqrt{K(x, x)K(y, y)}}$$

- (b) Why it would not make sense to normalize a Gaussian kernel?

- (c) [Bonus] Which types of input data would benefit from normalizing the kernel function? Explain why, and provide real world examples.

2. [Support Vector Machines, 50 points]

Prove that the sum of slacks $\sum \xi_n$ from the objective function of the SVM formulation with soft margin is an upper bound on the number of misclassified training examples.

3. [Max Margin Hyperplanes, 20 points]

Consider the constrained optimization SVM problem for the separable case shown on slide 12. Show that, if the 1 on the right-hand side of the inequality constraint is replaced by some arbitrary constant $\gamma > 0$, the resulting maximum margin hyperplane is unchanged.

4. [Kernel Techniques, 20 + 20 points]

(a) Show that if $k_1(\mathbf{x}, \mathbf{y})$ and $k_2(\mathbf{x}, \mathbf{y})$ are valid kernel functions, then $k(\mathbf{x}, \mathbf{y}) = k_1(\mathbf{x}, \mathbf{y})k_2(\mathbf{x}, \mathbf{y})$ is also a valid kernel.

(b) (*) Show that if A is a symmetric positive semidefinite matrix, then $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T A \mathbf{y}$ is a valid kernel. Hint: Inderjit Dhillon's Linear Algebra Background describes some useful properties of symmetric positive semidefinite matrices.

5. [Positive Definite Matrices (*), 20 points]

Show that a diagonal matrix \mathbf{W} whose elements satisfy $0 < W_{ii} < 1$ is positive definite. Show that the sum of two positive definite matrices is itself positive definite.

6. [Large Margin Perceptron (*), 30 points]

Let \mathbf{u} be a current vector of parameters and \mathbf{x} and \mathbf{y} two training examples such that $\mathbf{u}^T(\mathbf{x} - \mathbf{y}) < 1$. Use the technique of Lagrange multipliers to find a new vector of parameters \mathbf{w} as the solution to the convex optimization problem below:

minimize:

$$J(\mathbf{w}) = \frac{1}{2}\|\mathbf{w} - \mathbf{u}\|^2$$

subject to:

$$\mathbf{w}^T(\mathbf{x} - \mathbf{y}) \geq 1$$

2 Text Classification (100 points)

Train and test the SVM algorithm on the *Spam vs. Non-spam* and *Atheism vs. Religion* classification problems, using the datasets provided for the previous assignment. Use a linear kernel, with the cost parameter $C = 5$. Report and compare the accuracy of the trained SVM models with the perceptron and average perceptron accuracies from the previous assignment.

3 Digit Recognition (200 points)

In this exercise, you are asked to run an experimental evaluation of SVMs and the perceptron algorithm, with and without kernels, on the problem of classifying images representing digits.

1. The UCI Machine Learning Repository at www.ics.uci.edu/~mlearn maintains datasets for a wide variety of machine learning problems. For this assignment, you are supposed to work with the Optical Recognition of Handwritten Digits Data Set. The webpage for this dataset is at:

<http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits>

The actual dataset is located at:

<http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/>

Read the description of the dataset. Download the training set `optdigits.tra` and the test set `optdigits.tes`. Use the first 1000 examples in `optdigits.tra` for *development* and the rest of 2823 examples for *training*. Use all 1797 examples in `optdigits.tes` for *testing*. Scale all the features between [0, 1], as discussed in class, using the min and max computed over the training examples. Create training files for each of the 10 digits, setting the class to 1 for instances of that digit, and to -1 for instances of other digits, i.e. *one-vs-rest* scenario.

2. Train first the linear perceptron, with the number of epochs set to $T \in \{1, 2, \dots, 20\}$. After training each linear perceptron, normalize the learned weight vector. Select for T the value that obtains the best overall accuracy on the development data, and use this value for the remaining perceptron experiments.

Run experiments with the linear and kernel perceptron algorithms. For the kernel perceptron, experiment with polynomial kernels $k(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x}^T \mathbf{y})^d$ with degrees $d \in \{2, 3, 4, 5, 6\}$, and with Gaussian kernels $k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2 / 2\sigma^2)$ with the width $\sigma \in \{0.1, 0.5, 2, 5, 10\}$. For each hyper-parameter value, you will have trained 10 models, one for each digit. In order to compute the label for a test or development example, you will run the 10 trained models and output the label that obtains the highest score. Compute the accuracy on the development data and identify the hyper-parameter value that obtains the best accuracy. Use the tuned hyper-parameter (d for poly-kernel, σ for Gaussian) to compute the overall performance on the test data.

For each of the three perceptrons (linear, poly kernel and Gaussian kernel) report the total training time, the overall accuracy, and the number of support vectors. Show and compare the corresponding 4 confusion matrices. Which digit seems to be the hardest to classify? Which perceptron / kernel combination achieves the best performance? Which algorithms are slower at training time, and why?

- Run the same experiments using SVMs instead of perceptrons, i.e. linear SVMs and SVMs with polynomial and Gaussian kernels. Use the same tuning scenarios for the hyper-parameters of the polynomial and Gaussian kernels. Use $C = 1$ in all SVM experiments. Report the same types of results and analysis as above, and compare with the perceptron results.

4 Tools

You are free to use MATLAB, R, or packages written in C++/Java/Python such as SVM-LIGHT (C), LIBSVM (C++, Java), or SCIKIT-LEARN (Python) to complete the implementation part of this assignment. Their web sites contain plenty of documentation on how to use them. If you use SCIKIT-LEARN, the following functionality from the `sklearn.svm` will be useful:

- `SVC()`: This is the main class used for SVM classification models. Its implementation is based on LIBSVM. Make sure that you properly map the SVM hyper-parameters to the parameters in the constructor of this class. For example, the `gamma` parameter in the constructor corresponds to our $1/2\sigma^2$ coefficient in the Gaussian kernel. The formulas for the kernels implemented by SVC are described in this User Guide.
- `decision_function(x)`: Once the classifier is trained, this will compute the distance between a sample `x` and the decision hyperplane. This is the quantity that you can use to determine the highest scoring class when training the 10 *one – vs – rest* classifiers: once a classifier is trained for all 10 digits, given a sample `x` you compute this quantity for all 10 classifiers and select the class that corresponds to the classifier with largest decision function value.
- `fit()`: This is the function used to train the classifier.
- `predict(x)`: This is used to calculate the (binary) label for sample `x`.

LIBSVM, and therefore SCIKIT-LEARN too, already implement the *one-vs-rest* classification scheme. In this scheme, you can directly use the training dataset with the 10 original labels, and SCIKIT-LEARN will train the 10 binary classifiers for you. You can use this capability for this assignment, however bonus points will be given if you train the 10 binary classifiers directly, as described for the perceptron algorithm above, by creating a binary training dataset for each class.

5 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. Electronically submit on Blackboard a `hw07.zip` file that contains the `hw07` folder in which you place the code and the datasets. Make sure you include a `README.txt` file explaining how the code is supposed to be used to replicate the results included in the report. The screen output produced when running the code should be redirected to (saved into) an `output.txt` file.

On a Linux system, creating the archive can be done using the command:

```
> zip -r hw07.zip hw07
```

Please observe the following when handing in homework:

1. Structure, indent, and format your code well.
2. Use adequate comments, both block and in-line to document your code.
3. **Do not submit third-party ML packages on Blackboard!** Just explain in the REAMDE file how you use external packages.
4. Make sure your code runs correctly when used in the directory structure shown above.
5. **Type and nicely format the project report**, including discussion points, tables, graphs etc. so that it is presentable and easy to read.
6. Working code and/or correct answers is only one part of the assignment. The project report, including discussion of the specific issues which the assignment asks about, is also a very important part of the assignment. Take the time and space to make an adequate and clear project report. On the non-programming learning-theory assignment, clear and complete explanations and proofs of your results are as important as getting the right answer.

HW 07

Bhishan Poudel

63/50

Lecture

Normalized Kernel is a valid kernel.

Defn: Let $K(x,y)$ is a valid kernel, then, there exist a feature map $\phi(\cdot)$ in some Hilbert space that,

$$K(x,y) = \phi(x) \cdot \phi(y)$$

$$\text{Now, } \hat{K}(x,y) = \frac{K(x,y)}{\sqrt{K(x,x), K(y,y)}}$$

$$= \frac{\phi(x) \cdot \phi(y)}{\|\phi(x)\| \|\phi(y)\|} = \hat{\phi}(x) \cdot \hat{\phi}(y)$$

where $\hat{\phi}(x) = \frac{\phi(x)}{\|\phi(x)\|}$

since $K(x,y)$ is kernel $\Rightarrow \phi(x) \cdot \phi(y) \geq 0$

also, denominator $\Rightarrow \|\phi(x)\| \|\phi(y)\| \geq 0$

$\therefore \hat{K}(x,y)$ is non-negative. } $\therefore \hat{K}(x,y)$
 also $\hat{K}(x,y)$ is symmetric. } is a
 valid kernel.

Q17b Gaussian function,

Gaussian fn

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\|x-\mu\|^2}{2\sigma^2}}$$

(normalized gaussian function when $y=\mu$)

The gaussian kernel is,

Gaussian Kernel

$$K(x,y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}} = e^{-\gamma \|x-y\|^2}$$

where $\gamma = \frac{1}{2\sigma^2}$

$$K(0,0) = e^0 = 1$$

$K(0,0) = 1$ for gaussian kernel.

Since gaussian kernel is already normalized ($K(0,0)=1$), it would make no sense to normalize gaussian kernel.

3C some of the kernels used in practice

linear kernel

$$K(x,y) = \alpha^T y + c = \langle \alpha, y \rangle + c = \vec{\alpha} \cdot \vec{y} + c$$

Ans

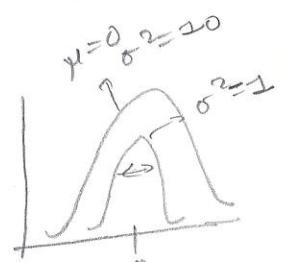
polynomial kernel

$$K(x,y) = (\alpha^T y + c)^d$$

α = slope
 d = degree
 c = const

Gaussian Kernel

$$K(x,y) = e^{-\frac{1}{2\sigma^2} \|x-y\|^2} = e^{-\pi/\sigma^2 \|x-y\|^2}$$



$$FWHM = 2\sqrt{2\ln 2} \cdot \sigma$$

(σ or γ determines the width of the gaussian curve.)

overestimate \rightarrow behave like linear kernel

underestimate \rightarrow lack regularization and high sensitive to noise

tuning of $\sigma \rightarrow$ very important, otherwise very low performance

exponential kernel $K(x,y) = e^{-\gamma \|x-y\|}$

sigmoid kernel $K(x,y) = \tanh(\alpha^T y + c)$

Gaussian kernels are already normalized and we need to normalize polynomial kernels.

cosine normalization

- Reduces dimension of data by 1 (since it projects all the data inside a unit radius sphere)
- Good for high dimensional input data
e.g. optdigits with 64 features
- NOT good for low dimensional data
e.g. Iris data with only two four features.

(ON2)
GO
GO

prove! sum of slack $\geq \epsilon_n$ is an upper bound on the number of misclassified training examples.

solution:

Input data

X

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_N \end{bmatrix}$$

Target

t

$$\begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_i \\ \vdots \\ t_N \end{bmatrix}$$

$$t \in \{-1, 1\}$$

$$x_i \in \mathbb{R}^d$$

$$x_i \in \{x_{i1}, x_{i2}, \dots, x_{id}\} \quad d = \text{dimension of features.}$$

$$\text{hypothesis} \quad h_i = w^T x_i + b$$

$$t_i \in \{-1, 1\}$$

$$\text{algorithm: } \underset{\substack{\text{constraint} \\ \rightarrow}}{t_i (w^T x_i + b)} \geq 1 - \epsilon_i$$

$$\underset{\substack{\text{minimization} \\ \rightarrow}}{\text{objective}} \quad J(w, b) = \frac{1}{2} \|w\|^2$$

$$\text{minimize } J(w, b) = \frac{1}{2} \|w\|^2$$

$$\text{with constraint } t_i (w^T x_i + b) \geq 1 - \epsilon_i$$

consider

$$t_i = +1$$

(correct classification)

$$w^T x_i + b > 0$$

$$t_i (w^T x_i + b) > 0 \quad \because t_i = +1$$

~~constraint~~ $t_i (w^T x_i + b) \geq 1 - \xi_i$; $\xi_i \geq 0$

Aside:

for perceptron

$$h_i = \text{sgn}(w^T x_i + b)$$

$$h_i = t_i (w^T x_i + b)$$

and if

if $t_i h_i$ we update w and b

~~constraint~~

$$+ve \geq 1 - \xi_i$$

since constraint want $\xi_i \geq 0$

~~constraint~~ ξ_i cannot be greater than 1
 $0 \leq \xi_i \leq 1$

$$t_i (w^T x_i + b) \geq 1 \text{ if } \xi_i = 0$$

(hard margin)

consider $t_i = +1$ (correct classification)

$$w^T x_i + b < 0$$

$$t_i (w^T x_i + b) < 0 \quad \because t_i = +1$$

~~constraint~~:

$$t_i (w^T x_i + b) \geq 1 - \xi_i$$

$$+ve \geq 1 - \xi_i$$

$$\xi_i \geq 1 - (+ve)$$

$$0 \leq \xi_i \leq 1$$

$$\text{and } t_i (w^T x_i + b) \geq 1 \text{ if } \xi_i = 0$$

\Rightarrow for correct classification

$$0 \leq \xi_i \leq 1$$

consider $t_i = +1$

(misclassification)

$$w^T x_i + b < 0$$

$$t_i (w^T x_i + b) < 0 \quad \because t_i = +1$$

~~constraint~~:

$$t_i (w^T x_i + b) \geq 1 - \xi_i$$

$$-ve \geq 1 - \xi_i$$

$$\xi_i \geq 1 + (-ve)$$

$$\xi_i > 1$$

$t_i = -1$ (misclassification)

$$w^T x_i + b > 0$$

$$t_i (w^T x_i + b) < 0 \quad \because t_i = -1$$

~~constraint~~:

$$t_i (w^T x_i + b) \geq 1 - \xi_i$$

$$-ve \geq 1 - \xi_i$$

$$\xi_i \geq 1 + (+ve)$$

$$\xi_i > 1$$

\Rightarrow for misclassification

$$\xi_i > 1$$

Now, for correct classification

$$e_{ii} \geq 0$$

$$\sum e_{ii} \geq 0$$

i: correct

for misclassified

$$e_{ii} > 1$$

$$\sum e_{ii} > \sum 1$$

i: misclassified i: misclassified

$$\sum e_{ii} > \# \text{ of misclassified}$$

$$\therefore \sum_{i \in \text{correct}} e_{ii} + \sum_{i \in \text{misclassified}} e_{ii} > \# \text{ of misclassified}$$

$$\therefore \boxed{\sum_{i=1}^N e_{ii} > \# \text{ of misclassified}}$$

Q.E.D.

This shows that sum of slack (e_{ii}) over all the N training examples is greater than total number of misclassified examples, hence, sum of slacks is an upper bound to the number of misclassified examples.

Aside: the constrained optimization of SVM algorithms

$t_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$ is called Hard margin SVM, and,

$t_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - e_{ii}$ is called Soft margin SVM.

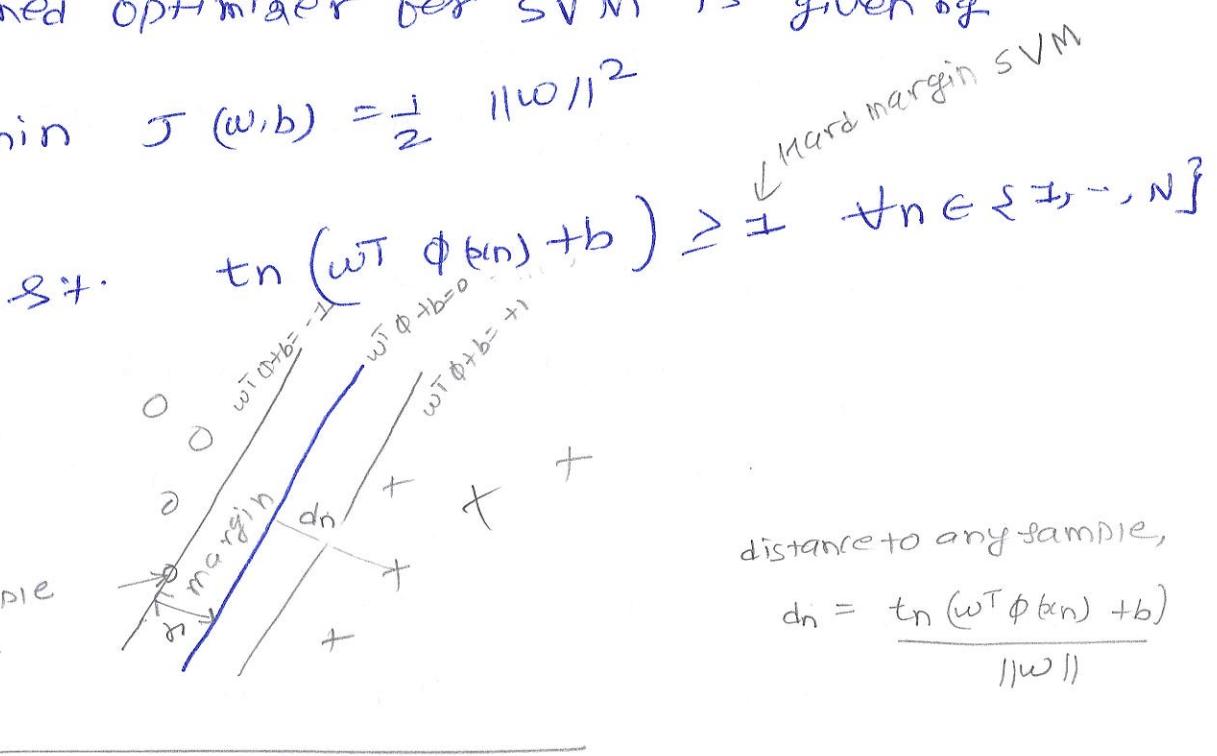
They both are non-trivial algorithm (unlike vanilla perceptron) and use Lagrange multipliers to solve the optimization.

QN3

max margin hyperplane

for the linearly separable dataset, the constrained optimiser for SVM is given by

$$\min J(w, b) = \frac{1}{2} \|w\|^2$$



from slides,
(lecture 07, page 5)

$$\text{margin} = \min_n \frac{t_n (\mathbf{w}^T \phi(\mathbf{x}_n) + b)}{\|\mathbf{w}\|}$$

we find parameters w and b that maximizes margin,

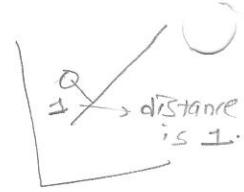
$$w^*, b^* = \underset{w, b}{\operatorname{argmax}} \cdot \min_n t_n (\mathbf{w}^T \phi(\mathbf{x}_n) + b) / \|w\|$$

Since w^*, b^* are obtained from argmax, they do not depend on rescaling of w, b .

So, without loss of generality, we can say that the nearest distance to the sample from hyperplane is 1.

i.e. for closest point(s),

$$t_n(\omega^T \phi(x_n) + b) = 1$$



then, our optimizing constraint becomes,

$$t_n(\omega^T \phi(x_n) + b) \geq 1 \quad \forall n \in \{1, \dots, N\}$$

[Aside: previously we used

$$t_n(\omega^T \phi + b) = 1$$

$$t_n h_n = 1$$

(for perceptron)

$$\begin{cases} \omega^T \phi(x_n) + b > 0 & \text{if this is +} \\ \omega^T \phi(x_n) + b < 0 & \text{if this is -} \\ h_n = \text{sgn}(\omega^T \phi(x_n) + b) \end{cases}$$

now, with inclusion of this constraint, the new optimization problem for SVM becomes,

$$\left\{ \begin{array}{l} \text{minimize} \\ \sigma(\omega | b) = \frac{1}{2} \|\omega\|^2 \end{array} \right.$$

$$\text{s.t. } t_n(\omega^T \phi(x_n) + b) \geq 1 \quad \forall n \in \{1, \dots, N\}$$

here we have
use the margin
 $\gamma = 1$.

we have to show that if $\gamma_1 = 0$, then
also the decision hyperplane is unchanged.

Solution :

The distance to the n th sample
from the decision hyperplane is,

$$d_n = \frac{t_n(\omega^T \phi(t_n) + b)}{\|\omega\|} \quad \} \rightarrow ①$$

If we rescale the weight vector to ± 1 ,
 $\|\omega\| = 1$, then,

$$d_n = t_n(\omega^T \phi(t_n) + b) \quad \} \rightarrow ②$$

then, maximizing margin algorithm becomes,

$$\max_{\omega, b} \min_n d_n \quad \} \rightarrow ③$$

$$\text{s.t. } \|\omega\| = 1$$

Let γ_1 be the maximum margin to ^{closest} training example,

$$\forall n \quad t_n(\omega^T \phi(t_n) + b) \geq \gamma_1$$

$$\Rightarrow \quad t_n\left(\frac{\omega^T \phi}{n} + \frac{b}{n}\right) \geq 1$$

$$\Rightarrow \quad t_n(N^T \phi + B) \geq 1 \quad \} \rightarrow ④$$

where $v = \frac{w}{\|w\|}$, $B = \frac{b}{\|w\|}$, γ = margin to the closest example.

then, $v = \frac{w}{\|w\|}$

$$\|v\|^2 = \left\| \frac{w}{\|w\|} \right\|^2 = \frac{1}{\|w\|^2} \|w\|^2$$

$$\boxed{\|v\|^2 = \frac{1}{\|w\|^2}} \quad \text{since } \|w\| = 1$$

Hence, our original optimization problem,

$$\max_{w, b} \min_n d_n = \min_n (w^T \phi(x_n) + b)$$

reduces to maximizing $\frac{1}{\|w\|^2}$

i.e. minimizing $\|v\|^2$

i.e. minimizing $\frac{1}{2} \|v\|^2$

which we can write as,

$$\min \frac{1}{2} \|v\|^2$$

$$\text{s.t. } \min_n (v^T \phi(x_n) + b) \geq 1.$$

(Eqn ④)

which is same as

$$\min \mathcal{J}(w, b) = \frac{1}{2} \|w\|^2$$

$$\text{s.t. } \min_n (w^T \phi(x_n) + b) \geq 1 \quad \text{in Eqn ④}$$

A \rightarrow D

~~given~~ \Rightarrow product of two kernels is a kernel.

Let ϕ_1, ϕ_2 are the feature map of kernels k_1, k_2

then,

$$\begin{aligned} K_1(x_1, x_2) \cdot K_2(b_1, b_2) &= (\phi_1(x_1) \quad \phi_1(x_2)) \quad (\phi_2(b_1) \quad \phi_2(b_2)) \\ &= \left(\sum_{i=1}^{\infty} f_i(x_1) f_i(b_2) \right) \left(\sum_{j=1}^{\infty} g_j(b_1) g_j(x_2) \right) \\ &= \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} f_i(x_1) f_i(b_2) g_j(b_1) g_j(x_2) \\ &= \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} (f_i(x_1) g_j(x_1)) \quad (f_i(b_2) g_j(b_2)) \\ &= \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} h_{ij}(x_1) h_{ij}(b_2) \end{aligned}$$

where $h_{ij}(b)$ is a feature vector in feature space ϕ_3 such that

$$h_{ij}(x) = f_i(x) g_j(x)$$

$$\therefore K_1(x_1, x_2) \cdot K_2(b_1, b_2) = \phi_3(x_1)^T \phi_3(x_2)$$

is a valid kernel.

~~X~~ ~~any b~~ $x^T A y$ is valid kernel if A is sym PSD matrix.

join: Given $A \rightarrow$ symmetric positive semidefinite

$A^T A$ is also sym PSD (from
inherit Shallow's reference)

Now,

$$\text{Let, } K(x,y) = x^T A^T A y$$

$$= (Ax)^T (Ay) \quad \because (AB)^T = B^T A^T \\ = \phi(x)^T \phi(y)$$

where we define feature map
 $\phi(z) = Az$

$\therefore K(x,y)$ is a valid kernel.

~~ANSWER~~ Diagonal matrix with positive elements
is positive definite matrix.

~~20/20~~
SOLN: Let $w = w_{n,n} \in \mathbb{R}^{n \times n}$ is a diagonal matrix then,

$$w = \begin{bmatrix} w_{11} & & & \\ & w_{22} & & \\ & & \ddots & \\ & & & w_{nn} \end{bmatrix}_{n,n}$$

Let \vec{x} be a n -dimensional column vector,

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{n,1}$$

then, the inner product of diagonal matrix w associated with \vec{x} is,

$$\vec{x}^T w \vec{x} = [x_1 \ x_2 \ \dots \ x_n]_{1,n} \begin{bmatrix} w_{11} & & & \\ & w_{22} & & \\ & & \ddots & \\ & & & w_{nn} \end{bmatrix}_{n,n} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{n,1}$$

$$= \sum_{j=1}^n w_j x_j^2$$

where $w_j = w_{jj}$
(diagonal elements)

$$x^T W x = \sum_{j=1}^n w_j x_j^2 = \text{positive number}$$

here all the diagonal elements w_j are positive
 $(0 < w_{ii} \leq +\infty)$.
 then the RHS quantity is positive
 and hence W is a positive definite matrix.

After: $W = \text{diag}(w_{11}, w_{22}, \dots, w_{nn})$, $w_{ii} \geq 0$
 let $\sqrt{W} = \text{diag}(\sqrt{w_{11}}, \sqrt{w_{22}}, \dots, \sqrt{w_{nn}})$

$$\begin{aligned} x^T W x &= (x^T \sqrt{W}^T)(\sqrt{W} x) \\ &= (\sqrt{W} x)^T (\sqrt{W} x) \\ &= \|\sqrt{W} x\|^2 \geq 0 \end{aligned}$$

$\therefore W$ is PSD.

~~ANSWER~~ sum of two PSD matrices is PSD matrix.

Let W is a positive ~~semi-definite~~ matrix
 $\Rightarrow \alpha^T W \alpha \geq 0 \quad \forall \alpha \in \mathbb{R}^n$

Let V is a positive semi-def matrix
 $\Rightarrow \alpha^T V \alpha \geq 0$

now, $\alpha^T W \alpha + \alpha^T V \alpha \geq 0$

or, $\alpha^T (W+V) \alpha \geq 0$ (distributive property of matrix)

this means that sum of two matrices $W+V$ is also positive semi-definite.

proved

Ques $\omega = u + \lambda(b-y)$

\rightarrow s.t. $J(\omega) = \frac{1}{2} \|\omega - u\|^2$

$$= \frac{1}{2} \|\lambda(b-y)\|^2$$

constraint: $\omega^T(b-y) = u^T(b-y) + \lambda \|b-y\|^2 \geq 1$

for, $\lambda \|b-y\|^2 \geq 1 - u^T(b-y)$

$$\lambda \geq \frac{1 - u^T(b-y)}{\|b-y\|^2}$$

Third, optimization problem:

$$\min_{\lambda} J(\lambda) = \frac{1}{2} \lambda^2 \|b-y\|^2$$

s.t. $\lambda \geq \frac{1 - u^T(b-y)}{\|b-y\|^2}$

Again, $J(\lambda)$ will be minimum when λ will be minimum.
 But $\min \lambda = \frac{1 - u^T(b-y)}{\|b-y\|^2}$

so, $\lambda = \frac{1 - u^T(b-y)}{\|b-y\|^2}$

then, the ω is given by
 (eliminate λ)

$$\boxed{\omega = u + \frac{1 - u^T(b-y)}{\|b-y\|^2} (b-y)}$$

Ans

Lag6

Lagrange multipliers

Here, the optimization problem is,

minimize w $J(w) = \frac{1}{2} \|w - u\|^2$

s.t. $w^T (x-y) \geq 1$

$$w^T x - w^T y \geq 1$$

$$1 - w^T x + w^T y \leq 0$$

primal Lagrangian

$$L_p(w, \alpha, y) = \frac{1}{2} (w-u)^2 + \alpha (1 - w^T x + w^T y)$$

$\alpha \geq 0$ is Lagrange multiplier

Set gradients to zero

$$\frac{\partial L_p}{\partial w} = w - u - \alpha x + \alpha y \Rightarrow w = u + \alpha x - \alpha y = u + \alpha(x-y)$$

$$\frac{\partial L_p}{\partial \alpha} = 1 - w^T x + w^T y \Rightarrow 1 = w(x-y) \Rightarrow \alpha = \frac{1}{w}$$

$$\text{then, } w = u + \frac{\alpha}{\alpha} = u + \frac{1}{w} \Rightarrow w^2 = uw + 1 \Rightarrow w^2 - uw - 1 = 0$$

$$w^* = \frac{u \pm \sqrt{u^2 + 4}}{2}$$

This is
not
solution
look
this

In gradient descent algorithm we initialize $\vec{u} = 0$

and $\alpha > 0$ and find the optimum value of w^* .

Bhishan Poudel

HW Assignment 8 (Due by 10:30am on Dec 7)

1 Theory (150 points)

1. [Kernel Nearest Neighbor, 50 points]

The nearest-neighbour classifier 1-NN assigns a new input vector \mathbf{x} to the same class as that of the nearest input vector \mathbf{x}_n from the training set, where in the simplest case, the distance is defined by the Euclidean metric $\|\mathbf{x} - \mathbf{x}_n\|^2$. By expressing this rule in terms of scalar products and then making use of kernel substitution, formulate the nearest-neighbour classifier for a general nonlinear kernel.

2. [Distance-Weighted Nearest Neighbor, 50 points]

We have seen how to use kernels to formulate a distance-weighted nearest neighbor algorithm, when the labels are binary. Formulate a kernel-based, distance-weighted nearest neighbor that works for K classes, where $K \geq 2$.

3. [Naive Bayes, 50 points]

The Naive Bayes algorithm for text categorization presented in class treats all sections of a document equally, ignoring the fact that words in the title are often more important than words in the text in determining the document category. Describe how you would modify the Naive Bayes algorithm for text categorization to reflect the constraint that words in the title are K times more important than the other words in the document for deciding the category, where K is an input parameter (include pseudocode).

4. [Logistic Regression (*), 50 points]

Assume that a binary feature x_i is equal to 1 for all training examples \mathbf{x} belonging to a particular class C_k , and zero otherwise (i.e. x_i perfectly separates examples from class C_k from all other examples). Show that in this case the magnitude of the ML solution for \mathbf{w}_k goes to infinity, thus motivating the use of a prior over the parameters (Hint: use the fact that the gradient on slide 24 must vanish at the solution).

2 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. On this theory assignment, **clear and complete explanations and proofs of your results are as important as getting the right answer.**

HW08

Bishan Poudel

K-Nearest Neighbor (Kernel based classifier)

① 1 nearest neighbor

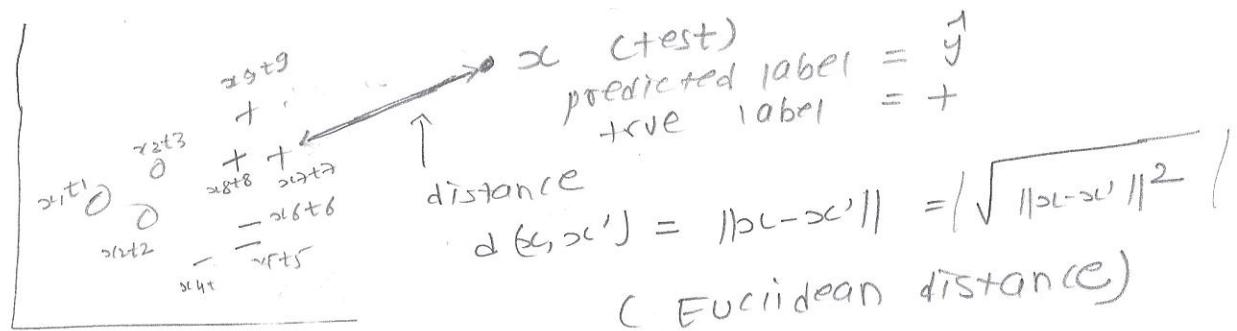
training dataset : $(x_1, t_1), (x_2, t_2), \dots, (x_n, t_n)$

test dataset : x

required output: \hat{y} (label of x or class of x)

t_1, t_2, \dots, t_n are classes from c_1, c_2, \dots, c_m

for example {circle plus minus }
 $c_1 \quad c_2 \quad c_m$



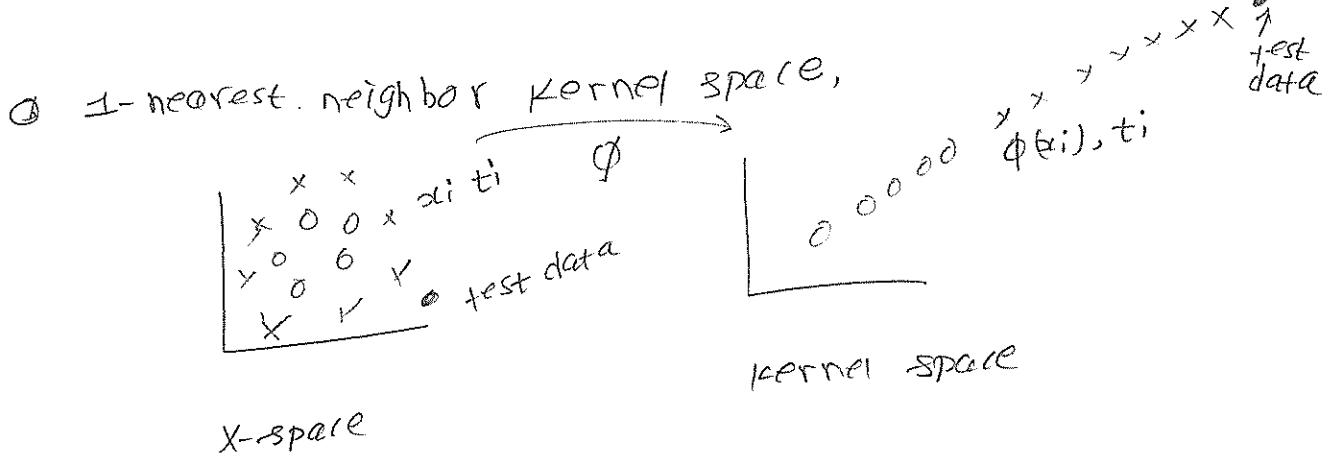
How to find \hat{y} ?

⇒ For 1-nearest neighbor, calculate the distance (maybe Euclidean, cosine, edit, etc) from test point \vec{x} to all the data points $x_i, 1 \leq i \leq n$ then find the minimum distance data point x' .

Assign label of x' as the predicted label to \vec{x} .

$$\hat{y} = \text{class of } x'$$

= class of x_i such that $d(x_i)$ is minimum



Here, when calculating distance, we use kernel method.

X-space: $d(x, x') = \|x - x'\| = \sqrt{\|x - x'\|^2} = \sqrt{(x - x')^T (x - x')}$

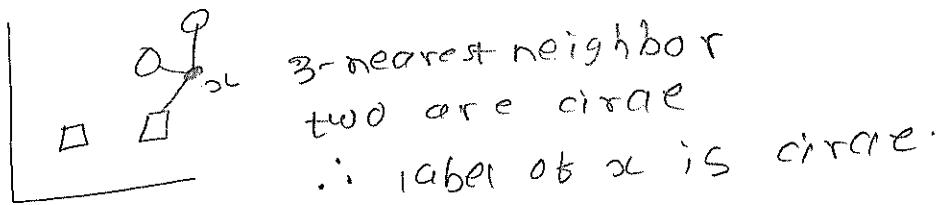
(Euclidean distance)

Kernel-Space: $d(\phi(x), \phi(x')) = \sqrt{(\phi(x) - \phi(x'))^T (\phi(x) - \phi(x'))}$
 $= \sqrt{K(x, x')}$

→ we compute the distance of test \vec{x} to the all the training examples x_i ; using kernel method then choose the minimum distance training point x' and then,

$$\text{label of } x = \text{label of } x'$$

⑥ K-nearest neighbor kernel method.



Kernel \Rightarrow distances are calculated using
kernel trick

$$d(x_i, x_j) = \sqrt{k_{\phi}(x_i, x_j)}$$

Kernel can be polynomial kernel,
gaussian kernel and so on.

Let x_1, x_2, \dots, x_K are nearest of x

then,

$$y(x) = \underset{t \in T}{\operatorname{argmax}} \sum_{i=1}^K \delta_t(t_i)$$

Hence, $t_i = \begin{cases} 0, & \text{circle square} \\ 1, & \text{square} \end{cases}$

$K=3$

for x_1 and x_2 $t=t_i=0$ number = 2

for x_3 $t=t_i=1$ number = 1

$$\operatorname{argmax} = 0$$

Summary

input $(x_1, t_1), (x_2, t_2), \dots, (x_n, t_n)$ $t_n \in \{t_1, t_2, \dots, t_k\}$

test \sim , $t = ?$

map input from x -space to kernel-space

x -space input $\phi(x_1, t_1) \quad \phi(x_2, t_2) \quad \dots \quad \phi(x_n, t_n)$

kernel space test $\phi(x), t = ?$

compute similarity $K(\phi(x))$ for all x_i , align

(compute distance $= \sqrt{K(\phi(x))}$ is redundant since
distance \leq similarity)

then predicted class of x ,

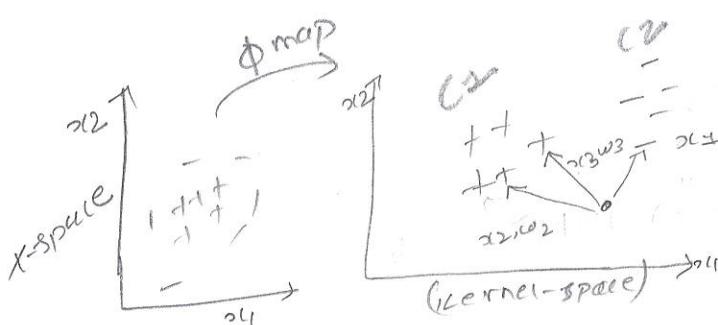
$$y(x) = \arg \max_{t \in T} \sum_{i=1}^k \delta_t(t_i)$$

[PN2] DISTANCE-weighted Nearest Neighbor for classification

@ Kernel space

@ distance-weighted KNN binary classification

ϕ map



$$t_i \in \{-1, +1\}$$

3-nearest neighbor

contribution from + : $w_2 + w_3$

contribution from - : w_1

suppose $w_2 + w_3 > w_1$

then label of $x = +$

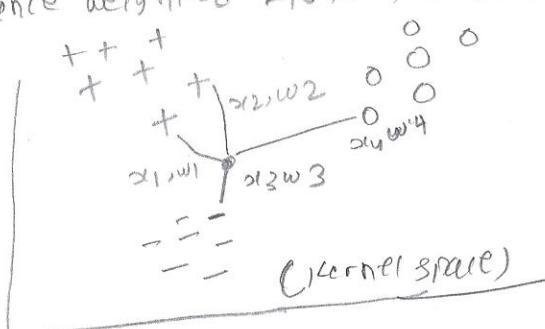
- x' is test example
- x_1, x_2, \dots, x_k are the nearest points to x' , calculated using Euclidean distance in kernel space.
- $w_k(x_i)$ is the measure of similarity between point x' and x_i in kernel space.

$$\text{Weighted sum} = \sum_{i=1}^k w_k(x_i) \delta_t(t_i)$$

$$y = \text{sign} \left(\sum_{i=1}^k w_k(x_i) \delta_t(t_i) \right)$$

- (b) Kernel-based distance weighted KNN for multiclass classification

Note: nearer points have more contribution



4-nearest neighbor

contribution from class c1 = +
= $w_1 + w_2$

contribution from class c2 = -
= w_3

contribution from class c3 = 0
= w_4

suppose $w_1 + w_2 > w_3 \quad \therefore y = +$
and $w_1 + w_2 > w_4 \quad \Rightarrow$

$$y = \operatorname{argmax}_{t \in T} \sum_{i=1}^k w_k(x_i) \delta_t(t_i)$$

Ans

Q3 Naïve Bias

① simple text classification using naïve bias

- each document is an example vector
- a document x has n words w_1, w_2, \dots, w_n
- from all the documents $\mathcal{D} = \{x_1, x_2, \dots, x_m\}$
- we create vocabulary file (this is feature vector for whole dataset \mathcal{D})

→ vocabulary $V = \{w_1, w_2, \dots, w_{|V|}\}$, there are $|V|$ number of words in vocabulary

→ each document x belongs to one of the categories

$C_K = \{c_1, c_2, \dots, c_K\}$

for example $\mathcal{D} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \text{line one} \\ \text{line two} \\ \text{line three} \end{bmatrix}$

input data target (category label)

news
sports
religion

We want \Rightarrow given a document x find its category

$p(c_k|x)$ probability of document x belonging to category c_k (e.g. news)

Naïve Bias gives \Rightarrow

① $p(c_k)$	probability of each categories (priors)
② $p(w_i c_k)$	conditional probability of data x w.r.t. category c_k

then we use Bayes theorem

to get $p(c^*|x)$.

$$p(c^*|x) = \frac{p(x|c^*) p(c^*)}{p(x)}$$

$$= p(c^*) \prod_{i=1}^n p(w_i|c^*)$$

where c^* corresponds to the $p(c^*|x)$ that has maximum probability

$$c^* = \arg \max_{c_k} p(c_k|x)$$

NB for toy classification when all words have equal importance

D: documents x_1, x_2, \dots, x_n

C $\{1, 2, \dots, K\}$: classes

$\{w_1, w_2, \dots, w_N\}$

below threshold words

V: vocabulary (unique words in D) \rightarrow top words

DK: subset examples having class C K

n $_K$: # of words in DK

1. for each category C K :

2. $D_K = \text{subset with category } C_K$

$$P(C_K) = \frac{|D_K|}{|D|} \quad (\text{prior})$$

3.

4.

$n_K = \# \text{ words in } D_K$

for each word $w_i \in V$:

let $n_{wi} = \# \text{ of } w_i \text{ in } D_K$

6.

$$\text{set } P(w_i | C_K) = \frac{n_{wi} + 1}{n_K + N} \quad \begin{array}{l} \text{Laplace} \\ \text{smoothing} \end{array}$$

7.

8. return priors $P(C_K)$
and cond prob $p(w_i | C_K)$

Then we find the posterior probability for each class C K ,

$$P(C_K|x) = \frac{P(x|C_K) \cdot P(C_K)}{P(x)}$$

$$\propto p(C_K) P(x)$$

$$\propto p(C_K) \cdot \prod_{i=1}^N p(w_i | C_K) \quad \begin{array}{l} \text{wi are words in} \\ \text{test document x} \end{array}$$

$$\propto \ln [p(C_K) \cdot \prod_i p(w_i | C_K)]$$

$$\propto \ln p(C_K) + \sum_i \ln (p(w_i | C_K))$$

(to prevent underflow)

Testing

$$c^* = \arg \max_{C_K} p(C_K|x)$$

$$c^* = \arg \max_{C_K} p(C_K) \prod_i p(w_i | C_K)$$

$$\propto \arg \max_{C_K} [\ln p(C_K) + \sum_i \ln (p(w_i | C_K))]$$

Example of Naive Bayes

train	doc	words	class	total
	1	chinese Beijing chinese	$c_1 = c$	
	2	chinese chinese shanghai	c_1	
	3	chinese macao	c_1	
	4	TOKYO JAPAN chinese	$c_2 \}$	$n_2 = 3$ words $d_2 = 1$ example $n_2 + n_1 = 3 + 6 = 9$ $\exists = \text{total words}$ 6 unique words
test	1	chinese Chinese Chinese TOKYO Japan (\rightarrow)	?	$ V = 6$

vocabulary: $\{ w_1^{\text{chinese}}, w_2^{\text{Beijing}}, w_3^{\text{shanghai}}, w_4^{\text{macao}}, w_5^{\text{TOKYO}}, w_6^{\text{JAPAN}} \} \quad |V| = 6$

word frequency:

	w_1^{chinese}	w_2^{Beijing}	w_3^{shanghai}	w_4^{macao}	w_5^{TOKYO}	w_6^{JAPAN}
document freq in c_1 :	5	1	1	1	0	1
document freq in c_2 :	1	0	1	1	1	1
Laplace smoothing:	1	1	2	2	1	2

$\frac{n_{ik} + 1}{n_k + |V|}$

category $c_1 = C$

• prior $P(c_1) = |D_1| / |D| = 3/4$ (3 documents belongs to class c_1)

• number of words in class, $n_1 = 8$

• conditional probabilities ($P(w_i | c_1) = \frac{n_{ik} + 1}{n_k + |V|}$)

$$\text{chinese } P(w_1 | c_1) = \frac{\cancel{5} + 1}{\cancel{8} + 6} = \frac{6}{14} = \frac{3}{7} \quad \text{in } P(w_1 | c_1) =$$

$$\text{Beijing } P(w_2 | c_1) = \frac{\cancel{1} + 1}{\cancel{8} + 6} = \frac{2}{14} = \frac{1}{7} \quad \text{in } P(w_2 | c_1) =$$

$$\text{shanghai } P(w_3 | c_1) = \frac{\cancel{1} + 1}{\cancel{8} + 6} = \frac{2}{14} = \frac{1}{7} \quad \text{in } P(w_3 | c_1) =$$

$$\text{macao } P(w_4 | c_1) = \frac{\cancel{1} + 1}{\cancel{8} + 6} = \frac{2}{14} = \frac{1}{7} \quad \text{in } P(w_4 | c_1) =$$

$$\text{TOKYO } P(w_5 | c_1) = \frac{\cancel{0} + 1}{\cancel{8} + 6} = \frac{1}{14} = \frac{1}{14} \quad \text{in } P(w_5 | c_1) =$$

$$\text{JAPAN } P(w_6 | c_1) = \frac{\cancel{0} + 1}{\cancel{8} + 6} = \frac{1}{14} = \frac{1}{14} \quad \text{in } P(w_6 | c_1) =$$

category $c_2 = j$

• prior $P(c_2) = |D_2| / |D| = 1/4$
(1 document belongs to class c_2)

• $n_2 = 3$

• conditional probabilities

$$P(w_1 | c_2) = \frac{\cancel{1} + 1}{\cancel{3} + 6} = \frac{2}{9}$$

$$P(w_2 | c_2) = \frac{0 + 1}{\cancel{3} + 6} = \frac{1}{9}$$

$$P(w_3 | c_2) = \frac{1}{9}$$

$$P(w_4 | c_2) = \frac{1}{9}$$

$$P(w_5 | c_2) = \frac{2}{9}$$

$$P(w_6 | c_2) = \frac{2}{9}$$

③ choosing class

$$P(c_1 | D) \propto P(c_1) \prod_{i \in c_1} P(w_i | c_1) = \frac{3}{4} \cdot \left(\frac{2}{7}\right)^3 \cdot \frac{1}{14} \cdot \frac{1}{14} = 0.0003$$

$$P(c_2 | D) \propto P(c_2) \prod_{i \in c_2} P(w_i | c_2) = \frac{1}{4} \cdot \left(\frac{1}{7}\right)^3 \cdot \frac{2}{14} \cdot \frac{2}{14} = 0.0001$$

$$c^* = \operatorname{argmax}_{c \in \mathcal{C}} \prod_{i=1}^n P(w_i | c_i) = \operatorname{argmax}_{c \in \mathcal{C}} \{ 0.0003, 0.0001 \} = \text{Label 0 or 0.0003} = c_1$$

P.T.O.

⑥ give more importance to title of doc

① for each category C_k :

find class prior $P(C_k) = |D_{C_k}| / |D|$ (where D_{C_k} is subset of D with class C_k)

② # loop through title words

③ let n_k = number of title words in D_{C_k} all examples

for each title word $w_i \in V_1$:

let n_{ki} = number of words in title of all D_{C_k} examples

find conditional prob of each title words

$$P_1(w_i | C_k) = \frac{n_{ki} + 1}{n_k + N_1} \times K \leftarrow \begin{array}{l} \text{multiplication factor} \\ \text{for title words} \end{array}$$

(4)

loop through body words

let n_k = number of body words

for each body word $w_i \in V_2$:

let n_{ki} = number of words in body of all D_{C_k} examples (for example C_k is class 1)

find conditional prob of each body words

$$P_2(w_i | C_k) = \frac{n_{ki} + 1}{n_k + |V_2|}$$

\Rightarrow output $P(C_k)$, $P_1(w_i | C_k)$, $P_2(w_i | C_k)$

choosing class:

we again suppose title words and body words are independent,

$$P(C_k | \alpha) \propto \underbrace{P_1(\alpha | C_k) P(C_k)}_{\text{for title}} \cdot \underbrace{P_2(\alpha | C_k) P(C_k)}_{\text{for body}}$$