Feng Wei

wei8

ECE 8560 Takehome#2

More Statistical PR

1. Assessing Classification Results from Takehome#1

We have know that this is a $C = 3$ and $d = 4$ problem. And in takehome#1 we use Bayesian classifier solved this problem. Now we know the true class is 3-1-2-3-2-1 and this sequence repeats throughout S_T. So we use this sequence to check our takehome#1 results, and compute the P_{error}. The confusion Matrix and P_{error} is showed below.

$$P(error) = P(\bar{x}\ is\ assigned\ to\ the\ wrong\ class)$$

$$P(error) = \Sigma\Sigma P(C_j|C_i) * P(C_i)\ \ where\ i \neq j;\ i,\ j\ from\ 1\ to\ 3$$

$$P(error) = \frac{320 + 153 + 583 + 28 + 212 + 40}{5000 + 5000 + 5000} = 8.9\%$$

$Confusion\ Matrix$

Class\Assigned	C_1	C_2	C_3	$Total$
C_1	4527	320	153	5000
C_2	583	4389	28	5000
C_3	212	40	4748	5000

2. Separating Hyperplanes

In this part,we implement a (simple) Ho-Kashyap hyperplane classifier for the given training set. Consider the classes pairwise. We generate the 3 hyperplanes from the respective subsets of H. In each case, provide the exact parameters of the respective like bellow

$\omega_{12} = [0.00835883 \ -0.00464676 \ 0.00095206 \ -0.01810467 \ -0.83225869]$
$\omega_{13} = [1.354e-03 \ -3.981e-03 \ -5.304e-02 \ 1.727e-04 \ -3.429e-01]$
$\omega_{23} = [-0.01891765 \ -0.00719361 \ -0.32457742 \ 0.06494742 \ -2.57168678]$
The Ho-Kashyap method is a Linear Discriminant Functions
In linear Discriminant Functions we want to use a ω to get a function like:

$$g(\bar{x}) = \bar{\omega}\bar{x} - \bar{\omega_0} = f(x) = \begin{cases} > 0 & class_1 \\ else & class_2 \end{cases}$$

After convert $\overline{x}$ from d dimension to d+1 dimension;

$$Ya > 0$$

Minimize the following criterion function, restricting to positive b

$$J_{hk}(a, b) = ||Ya - b||^2$$

As usual, take partial derivatives . After some math procedures we have :

$$a = (Y^T Y)^{-1} Y^T b$$

$$e^k = Ya^k - b^k$$

$$b^{k+1} = b^k + \alpha(e^k + |e^k|)$$

Iteration until $k > k_{max}$ or $b^{k+1} = b^k$
In this case we set $k_{max} = 100$.
After we run the iteration we know these three class is not linearly separable.

- Hyperplane classifier performance on S_T , as a confusion matrix and estimate of P_{error}

$$\bar{x} = (x_1, x_2, x_3, x_4, 1)^T$$
$$a = \omega_{12} * \bar{x}$$
$$b = \omega_{13} * \bar{x}$$
$$c = \omega_{23} * \bar{x}$$

$$Class = \begin{cases} Class_1 & a > 0 \ and \ b > 0 \\ Class_2 & c > 0 \\ Class_3 & else \end{cases}$$

$$P(error) = P(\bar{x} \ is \ assigned \ to \ the \ wrong \ class)$$
$$P(error) = \Sigma\Sigma P(C_j|C_i) * P(C_i) \ \ where \ i \neq j; \ i, \ j \ from \ 1 \ to \ 3$$
$$P(error) = \frac{631 + 556 + 1119 + 23 + 497 + 13}{5000 + 5000 + 5000} = 18.9\%$$

$Confusion \ Matrix$

Class\Assigned	C_1	C_2	C_3	$Total$
C_1	3813	631	556	5000
C_2	1119	3858	23	5000
C_3	497	13	4490	5000

3. $K-NNR$ Strategies

$K-NNR$ classifier results on S_T using H. In order to improve the computation efficiency we use $K-D\ tree$ to store H, that will make the compute more efficient but the $error$ will increase at the same time.

$$P(error) = P(\bar{x}\ is\ assigned\ to\ the\ wrong\ class)$$

$$P(error) = \Sigma\Sigma P(C_j|C_i) * P(C_i) \quad where\ i \neq j;\ i,\ j\ from\ 1\ to\ 3$$

$$P(error) = \begin{cases} 13.4\% & k=1 \\ 11.1\% & k=3 \\ 10.5\% & k=5 \end{cases}$$

$k=1$ $Confusion\ Matrix$

Class\Assigned	C_1	C_2	C_3	$Total$
C_1	4090	634	276	5000
C_2	643	4270	87	5000
C_3	292	68	4640	5000

$k=3$ $Confusion\ Matrix$

Class\Assigned	C_1	C_2	C_3	$Total$
C_1	4329	465	206	5000
C_2	625	4326	49	5000
C_3	272	44	4684	5000

$k=5$ $Confusion\ Matrix$

Class\Assigned	C_1	C_2	C_3	$Total$
C_1	4392	414	194	5000
C_2	628	4320	52	5000
C_3	255	42	4703	5000

4. PCA results (feature derivation assessment on S_T)

We choose the two "best" features that derived from the given four dimensions data.
The features is like bellow:
Before we use PCA, we should normalize the data, we choose $Z-score$ to do normalization
For $Class_1$
The proportion of variance of each of the principal components.

$$[0.26413761\ 0.25621136\ 0.24194866\ 0.23770236]$$

The weight vector for projecting a data point into PCA space for two "best" features

$$[0.64573123\ 0.37947763\ 0.54927163\ 0.37057872]$$

$$[-0.24091028\ 0.61117142\ 0.32117736\ -0.68211204]$$

For $Class_2$
The proportion of variance of each of the principal components.

$$[0.25865031\ 0.25017416\ 0.24652209\ 0.24465344]$$

The weight vector for projecting a data point into PCA space for two "best" features

$$[-0.38235534\ 0.5794042\ \ 0.54400402\ 0.471333]$$

$$[0.74435967\ -0.25969645\ 0.38182738\ 0.48238397]$$

For $Class_3$
The proportion of variance of each of the principal components.

$$[0.25865031\ 0.25017416\ 0.24652209\ 0.24465344]$$

The weight vector for projecting a data point into PCA space for two "best" features

$$[-0.66325964\ 0.5608821\ -0.15300357\ -0.47126196]$$

$$[-0.07712786\ -0.35779359\ 0.74410786\ -0.55887256]$$

After we choose these two feature we treat these data as two dimensions data, use what We have did in assign one to classify the test data.

$$P(error) = P(\bar{x}\ is\ assigned\ to\ the\ wrong\ class)$$

$$P(error) = \Sigma\Sigma P(C_j|C_i) * P(C_i) \quad where\ i \neq j;\ i,\ j\ from\ 1\ to\ 3$$

$$P(error) = \frac{1388+394+595+91+5424+349}{5000+5000+5000} = 22.4\%$$

$Confusion\ Matrix$

Class\Assigned	C_1	C_2	C_3	$Total$
C_1	3218	1388	394	5000
C_2	595	4314	91	5000
C_3	542	349	4109	5000

5. Comparison and analysis of Parts $1-4$

$Method$	P_{error}	$Compexity$
Bayesian classifier	8.9%	/
Separating Hyperplanes	18.9%	/
K-NN (5-NN)	10.5%	Use K-D tree
PCA	22.4%	Reduce dimension

From the table we know that Bayesian classifier has the lowest P_{error}. And PCA is much more efficient. Both have some advantages and disadvantages.