BMEG3105 Lecture 10 (9/10) Scribing

1155176975

1. Logistic function

Problem of KNN: 1. Need to store all data.

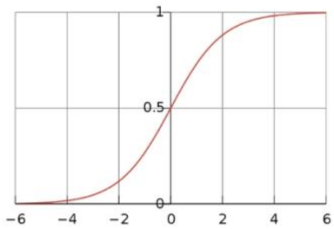
2. Need to calculate the distance matrix.

3. Slow predicting.

If we have a formula: $w_h H + w_w W + w_0 \geq 0.5$

$$\frac{1}{1+e^{-(w_h H + w_w W + w_0)}} \geq 0.5$$

(If wh, Ww, and w0 are large)



$$\frac{1}{1+e^{-t}} \geq 0.5$$

Training: fit the training data (how to choose wh, Ww and w0)

Make $\frac{1}{1+e^{-(w_h H + w_w W + w_0)}} \geq 0.5$ **correct for the training data**

$$Y^{output} = \frac{1}{1+e^{-(w_h H + w_w W + w_0)}}$$

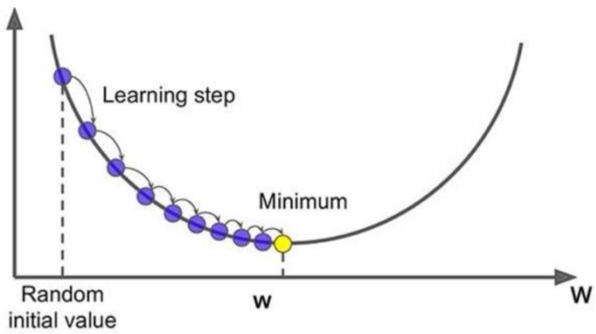
- ✧ $(Y^{output}-Y)^2$ should be as small as possible ->use calculus
- ✧ Y: the true label we have for training data
- ✧ It is the loss function we would like to minimise

<u>Gradient descent algorithm</u>

$L = \sum_{P_1}^{P_4}(Y^{output} - Y)^2$ **is a function of** ws

For each w**, we want to find a value to make the function value smallest**

$$\sum_{P_1}^{P_4}(Y^{output} - Y)^2$$



Step1. Initialise wh and Ww and W0 at any random value

Step2. For P1 to P4, calculate the output Y^{output}.

Step3. Update weights by

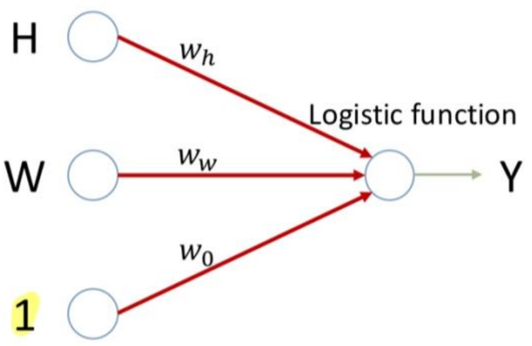
$$w_i = w_i + \Delta w_i$$

$$\Delta w_i = 2 * \alpha(Y - Y^{output}) \frac{\partial Y^{output}}{\partial w_i}$$

α **is a small constant**

Step4. Repeat the above step until no more to update.

Simplest neural network



Input nodes Output nodes

From Logistic regression to Neural Network

Advantages	Fast prediction.
	Successful in real-life problems
	High tolerance to noisy data
Disadvantages	Long training time
	Poor interpretability

✧ Alphafold: the most successful deep learning application.

2. Binary classification evaluation

Which clustering method is better? Which classification should we trust?

➔ Quantitative values are needed to summarise the performance of different methods.

✧ Confusion matrix

		Predicted class	
		Class=Yes	Class=No
Actual class	Class=Yes	a(TP)	b(FN)
	Class=No	c(FP)	d(TN)

Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

-However, there may be misleading for imbalanced data.

	Predicted class	
Actual class	Class=Yes	Class=No
Class=Yes	4949(TP)	0(FN)
Class=No	51(FP)	0(TN)

Imbalanced classes

$$Accuracy = \frac{TP + TN}{TP + TN + FP + TN} = \frac{4949}{4949 + 51} = 0.99$$

Maybe misleading for imbalanced data

-Although we have **precision** (how many are correct among the predicted positive samples), **recall** (how many actual positive samples are predicted to be positive) and **F1 score** (the weight average of precision and recall) -> Still maybe misleading for imbalanced data.

	Predicted class	
Actual class	Class=Yes	Class=No
Class=Yes	4949(TP)	0(FN)
Class=No	51(FP)	0(TN)

$$Precision = \frac{a}{a + c} = \frac{4949}{4949 + 51} = 0.99 \qquad Recall = \frac{a}{a + b} = 1 \qquad F1\ score = \frac{2 * precision * recall}{presicion + recall} = 0.995$$

	Predicted class	
Actual class	Class=Yes	Class=No
Class=Yes	4949(TP)	0(FN)
Class=No	51(FP)	0(TN)

*Look at the confusion matrix directly if you know it is an imbalanced dataset

$$Balanced\ accuracy = 0.5 * \left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) = 0.5$$

- Value is not absolute. Context matters.

3. Cross-validation

- A technique for assessing how the result of a machine learning analysis will generalise to an independent data set -> measure the performance of models.

For one round cross-validation: Partitioning a set of data into complementary subsets.

Performing the analysis on one subset (training set).

Validating the analysis on the other subset (testing set).

n-fold cross-validation (train multiple times, leave out a disjoint subset of data each time for validation. Average the validation set accuracies)

Process:

- Randomly partition data into n disjoint subsets
- For i = 1 to n
  - Validation Data = i-th subset
  - h <- classifier trained on all data except for Validation Data
  - Accuracy(i) = accuracy of h on Validation Data
- Final Accuracy = mean of the n recorded accuracies

Examples: 5-fold cross-validation

10 data points:

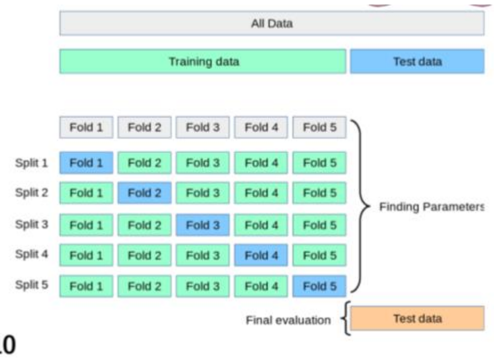
- P1-P10

5-fold

- P1-2, P3-4, P5-6, P7-8, P9-10
- The grouping can be random

Procedure

- P1-2's results based on the model from P3-10
- ...
- P9-10's results based on the model from P1-8
- Averaging



Leave-one-out cross-validation (a special case of n-fold cross-validation, where n=N)

Process:

- Partition data into N disjoint subsets, each containing one data point
- For i = 1 to N
  - Validation Data = i-th subset
  - h <- classifier trained on all data except for Validation Data
  - Accuracy(i) = accuracy of h on Validation Data
- Final Accuracy = mean of the N recorded accuracies

4. Multi-class classification

✧ No need to change the algorithm for KNN but for logistic regression, we need to

1. Build a logistic regression for each class

2. When predicting, assign class with the highest value

3. When training, train 3*N parameters, where N=number of classes we have.

4. Considering each class as a binary classification problem.

Example:

$$Macro - average = \frac{0.9 + 0.95 + \cdots + 0.7 + 0.2}{6} = 0.73$$

$$Micro - average = \frac{0.9 * 150 + \cdots + 0.2 * 10}{150 + \cdots + 10} = 0.85$$

The low-performance of small classes will show up in Macro-average

Class	Accuracy	Cells
1	0.9	150
2	0.95	50
3	0.85	100
4	0.8	40
5	0.7	20
6	0.2	10