Chapter 5

Evaluation of classifiers

In machine learning, there are several classification algorithms and, given a certain problem, more than one may be applicable. So there is a need to examine how we can assess how good a selected algorithm is. Also, we need a method to compare the performance of two or more different classification algorithms. These methods help us choose the right algorithm in a practical situation.

5.1 Methods of evaluation

5.1.1 Need for multiple validation sets

When we apply a classification algorithm in a practical situation, we always do a validation test. We keep a small sample of examples as validation set and the remaining set as the training set. The classifier developed using the training set is applied to the examples in the validation set. Based on the performance on the validation set, the accuracy of the classifier is assessed. But, the performance measure obtained by a single validation set alone does not give a true picture of the performance of a classifier. Also these measures alone cannot be meaningfully used to compare two algorithms. This requires us to have different validation sets.

Cross-validation in general, and k-fold cross-validation in particular, are two common method for generating multiple training-validation sets from a given dataset.

5.1.2 Statistical distribution of errors

We use a classification algorithm on a dataset and generate a classifier. If we do the training once, we have one classifier and one validation error. To average over randomness (in training data, initial weights, etc.), we use the same algorithm and generate multiple classifiers. We test these classifiers on multiple validation sets and record a sample of validation errors. We base our evaluation of the classification algorithm on the *statistical distribution of these validation errors*. We can use this distribution for assessing the *expected error* rate of the classification algorithm for that problem, or compare it with the error rate distribution of some other classification algorithm.

A detailed discussion of these ideas is beyond the scope of these notes.

5.1.3 No-free lunch theorem

Whatever conclusion we draw from our analysis is conditioned on the dataset we are given. We are not comparing classification algorithms in a domain-independent way but on some particular application. We are not saying anything about the expected error-rate of a learning algorithm, or comparing one learning algorithm with another algorithm, in general. Any result we have is only true for the particular application. There is no such thing as the "best" learning algorithm. For any

learning algorithm, there is a dataset where it is very accurate and another dataset where it is very poor. This is called the *No Free Lunch Theorem.*[1]

5.1.4 Other factors

Classification algorithms can be compared based not only on error rates but also on several other criteria like the following:

- risks when errors are generalized using loss functions
- training time and space complexity,
- testing time and space complexity,
- interpretability, namely, whether the method allows knowledge extraction which can be checked and validated by experts, and
- easy programmability.

5.2 Cross-validation

To test the performance of a classifier, we need to have a number of training/validation set pairs from a dataset X. To get them, if the sample X is large enough, we can randomly divide it then divide each part randomly into two and use one half for training and the other half for validation. Unfortunately, datasets are never large enough to do this. So, we use the same data split differently; this is called *cross-validation.*

Cross-validation is a technique to evaluate predictive models by partitioning the original sample into a training set to train the model, and a test set to evaluate it.

The *holdout method* is the simplest kind of cross validation. The data set is separated into two sets, called the training set and the testing set. The algorithm fits a function using the training set only. Then the function is used to predict the output values for the data in the testing set (it has never seen these output values before). The errors it makes are used to evaluate the model.

5.3 K-fold cross-validation

In K-fold cross-validation, the dataset X is divided randomly into K equal-sized parts, X_i, $i = 1, \ldots, K$. To generate each pair, we keep one of the K parts out as the validation set V_i, and combine the remaining $K-1$ parts to form the training set T_i. Doing this K times, each time leaving out another one of the K parts out, we get K pairs (V_i, T_i):

$$
\begin{aligned}
V_1 &= X_1, \quad T_1 = X_2 \cup X_3 \cup \ldots \cup X_K \\
V_2 &= X_2, \quad T_2 = X_1 \cup X_3 \cup \ldots \cup X_K \\
&\ldots \\
V_K &= X_K, \quad T_K = X_1 \cup X_2 \cup \ldots \cup X_{K-1}
\end{aligned}
$$

Remarks

1. There are two problems with this: First, to keep the training set large, we allow validation sets that are small. Second, the training sets overlap considerably, namely, any two training sets share $K-2$ parts.

[1] "We have dubbed the associated results NFL theorems because they demonstrate that if an algorithm performs well on a certain class of problems then it necessarily pays for that with degraded performance on the set of all remaining problems."(David Wolpert and William Macready in [7])

2. K is typically 10 or 30. As K increases, the percentage of training instances increases and we get more robust estimators, but the validation set becomes smaller. Furthermore, there is the cost of training the classifier K times, which increases as K is increased.

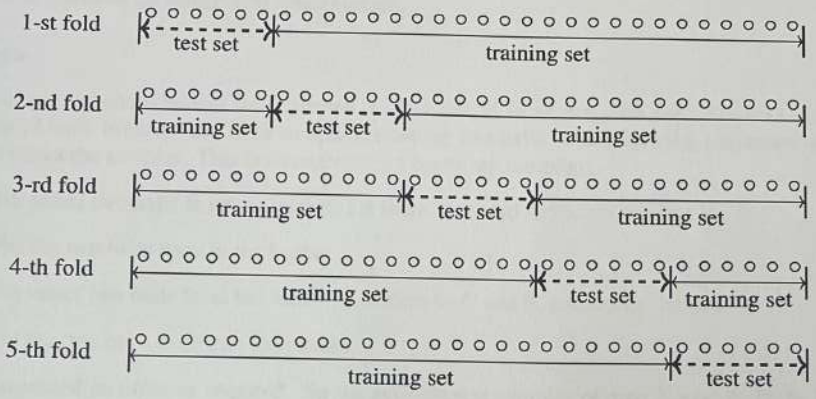

Figure 5.1: One iteration in a 5-fold cross-validation

Leave-one-out cross-validation

An extreme case of K-fold cross-validation is *leave-one-out* where given a dataset of N instances, only one instance is left out as the validation set and training uses the remaining $N-1$ instances. We then get N separate pairs by leaving out a different instance at each iteration. This is typically used in applications such as medical diagnosis, where labeled data is hard to find.

5.3.1 5×2 cross-validation

In this method, the dataset X is divided into two equal parts $X_1^{(1)}$ and $X_1^{(2)}$. We take as the training set and $X_1^{(2)}$ as the validation set. We then swap the two sets and take $X_1^{(2)}$ as the training set and $X_1^{(1)}$ as the validation set. This is the first fold. the process id repeated four more times to get ten pairs of training sets and validation sets.

$$
\begin{aligned}
T_1 &= X_1^{(1)}, \quad V_1 = X_1^{(2)} \\
T_2 &= X_1^{(2)}, \quad V_2 = X_1^{(1)} \\
T_3 &= X_2^{(1)}, \quad V_3 = X_2^{(2)} \\
T_4 &= X_2^{(2)}, \quad V_4 = X_2^{(1)} \\
&\vdots \\
T_9 &= X_5^{(1)}, \quad V_3 = X_5^{(2)} \\
T_{10} &= X_5^{(2)}, \quad V_{10} = X_5^{(1)}
\end{aligned}
$$

It has been shown that after five folds, the validation error rates become too dependent and do not add new information. On the other hand, if there are fewer than five folds, we get fewer data (fewer than ten) and will not have a large enough sample to fit a distribution and test our hypothesis.

5.3.2 Bootstrapping

Bootstrapping in statistics

In statistics, the term "bootstrap sampling", the "bootstrap" or "bootstrapping" for short, refers to process of "random sampling with replacement".

Example

For example, let there be five balls labeled A, B, C, D, E in an urn. We wish to select different samples of balls from the urn each sample containing two balls. The following procedure may be used to select the samples. This is an example for bootstrap sampling.

1. We select two balls from the basket. Let them be A and E. Record the labels.
2. Put the two balls back in the basket.
3. We select two balls from the basket. Let them be C and E. Record the labels.
4. Put the two balls back into the basket.

This is repeated as often as required. So we get different samples of size 2, say, A, E; B, E; etc. These samples are obtained by sampling with replacement, that is, by bootstrapping.

Bootstrapping in machine learning

In machine learning, bootstrapping is the process of computing performance measures using several randomly selected training and test datasets which are selected through a precess of sampling with replacement, that is, through bootstrapping. Sample datasets are selected multiple times.

The bootstrap procedure will create one or more new training datasets some of which are repeated. The corresponding test datasets are then constructed from the set of examples that were not selected for the respective training datasets.

5.4 Measuring error

5.4.1 True positive, false positive, etc.

Definitions

Consider a binary classification model derived from a two-class dataset. Let the class labels be c and $\neg c$. Let x be a test instance.

1. **True positive**

   Let the true class label of x be c. If the model predicts the class label of x as c, then we say that the classification of x is *true positive*.

2. **False negative**

   Let the true class label of x be c. If the model predicts the class label of x as $\neg c$, then we say that the classification of x is *false negative*.

3. **True negative**

   Let the true class label of x be $\neg c$. If the model predicts the class label of x as $\neg c$, then we say that the classification of x is *true negative*.

4. **False positive**

   Let the true class label of x be $\neg c$. If the model predicts the class label of x as c, then we say that the classification of x is *false positive*.

	Actual label of x is c	Actual label of x is $\neg c$
Predicted label of x is c	True positive	False positive
Predicted label of x is $\neg c$	False negative	True negative

5.4.2 Confusion matrix

A confusion matrix is used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. A *confusion matrix* is a table that categorizes predictions according to whether they match the actual value.

Two-class datasets

For a two-class dataset, a confusion matrix is a table with two rows and two columns that reports the number of false positives, false negatives, true positives, and true negatives.

Assume that a classifier is applied to a two-class test dataset for which the true values are known. Let TP denote the number of true positives in the predicted values, TN the number of true negatives, etc. Then the confusion matrix of the predicted values can be represented as follows:

	Actual condition is true	Actual condition is false
Predicted condition is true	TP	FP
Predicted condition is false	FN	FN

Table 5.1: Confusion matrix for two-class dataset

Multiclass datasets

Confusion matrices can be constructed for multiclass datasets also.

Example

If a classification system has been trained to distinguish between cats, dogs and rabbits, a confusion matrix will summarize the results of testing the algorithm for further inspection. Assuming a sample of 27 animals - 8 cats, 6 dogs, and 13 rabbits, the resulting confusion matrix could look like the table below: This confusion matrix shows that, for example, of the 8 actual cats, the system predicted that

	Actual "cat"	Actual "dog"	Actual "rabbit"
Predicted "cat"	5	2	0
Predicted "dog"	3	3	2
Predicted " rabbit"	0	1	11

three were dogs, and of the six dogs, it predicted that one was a rabbit and two were cats.

5.4.3 Precision and recall

In machine learning, precision and recall are two measures used to assess the quality of results produced by a binary classifier. They are formally defined as follows.

Definitions

Let a binary classifier classify a collection of test data. Let

$$\begin{aligned} \text{TP} &= \text{Number of true positives} \\ \text{TN} &= \text{Number of true negatives} \\ \text{FP} &= \text{Number of false positives} \\ \text{FN} &= \text{Number of false negatives} \end{aligned}$$

The *precision* P is defined as

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

The *recall* R is defined as

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Problem 1

Suppose a computer program for recognizing dogs in photographs identifies eight dogs in a picture containing 12 dogs and some cats. Of the eight dogs identified, five actually are dogs while the rest are cats. Compute the precision and recall of the computer program.

Solution

We have:

$$\begin{aligned} \text{TP} &= 5 \\ \text{FP} &= 3 \\ \text{FN} &= 7 \end{aligned}$$

The *precision* P is

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{5}{5+3} = \frac{5}{8}$$

The *recall* R is

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{5}{5+7} = \frac{5}{12}$$

Problem 2

Let there be 10 balls (6 white and 4 red balls) in a box and let it be required to pick up the red balls from them. Suppose we pick up 7 balls as the red balls of which only 2 are actually red balls. What are the values of precision and recall in picking red ball?

Solution

Obviously we have:

$$\begin{aligned} \text{TP} &= 2 \\ \text{FP} &= 7 - 2 = 5 \\ \text{FN} &= 4 - 2 = 2 \end{aligned}$$

The *precision* P is

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{2}{2+5} = \frac{2}{7}$$

The *recall* R is

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{2}{2+2} = \frac{1}{2}$$

Problem 3

Assume the following: A database contains 80 records on a particular topic of which 55 are relevant to a certain investigation. A search was conducted on that topic and 50 records were retrieved. Of the 50 records retrieved, 40 were relevant. Construct the confusion matrix for the search and calculate the precision and recall scores for the search.

Solution

Each record may be assigned a class label "relevant" or "not relevant". All the 80 records were tested for relevance. The test classified 50 records as "relevant". But only 40 of them were actually relevant. Hence we have the following confusion matrix for the search:

	Actual "relevant"	Actual "not relevant"
Predicted "relevant"	40	10
Predicted "not relevant"	15	25

Table 5.2: Example for confusion matrix

$$\begin{aligned} \text{TP} &= 40 \\ \text{FP} &= 10 \\ \text{FN} &= 15 \end{aligned}$$

The *precision* P is

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{40}{40 + 10} = \frac{4}{5}$$

The *recall* R is

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{40}{40 + 15} = \frac{40}{55}$$

5.4.4 Other measures of performance

Using the data in the confusion matrix of a classifier of two-class dataset, several measures of performance have been defined. A few of them are listed below.

1. Accuracy = $\dfrac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$
2. Error rate = 1− Accuracy
3. Sensitivity = $\dfrac{\text{TP}}{\text{TP} + \text{FN}}$
4. Specificity = $\dfrac{\text{TN}}{\text{TN} + \text{FP}}$
5. F-measure = $\dfrac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}$

5.5 Receiver Operating Characteristic (ROC)

The acronym ROC stands for Receiver Operating Characteristic, a terminology coming from signal detection theory. The ROC curve was first developed by electrical engineers and radar engineers during World War II for detecting enemy objects in battlefields. They are now increasingly used in machine learning and data mining research.

TPR and FPR

Let a binary classifier classify a collection of test data. Let, as before,

TP = Number of true positives
TN = Number of true negatives
FP = Number of false positives
FN = Number of false negatives

Now we introduce the following terminology:

$$\begin{aligned}\text{TPR} &= \text{True Positive Rate}\\ &= \frac{\text{TP}}{\text{TP} + \text{FN}}\\ &= \text{Fraction of positive examples correctly classified}\\ &= \text{Sensitivity}\\ \text{FPR} &= \text{False Positive Rate}\\ &= \frac{\text{FP}}{\text{FP} + \text{TN}}\\ &= \text{Fraction of negative examples incorrectly classified}\\ &= 1 - \text{Specificity}\end{aligned}$$

ROC space

We plot the values of FPR along the horizontal axis (that is , x-axis) and the values of TPR along the vertical axis (that is, y-axis) in a plane. For each classifier, there is a unique point in this plane with coordinates (FPR, TPR). The ROC space is the part of the plane whose points correspond to (FPR, TPR). Each prediction result or instance of a confusion matrix represents one point in the ROC space.

The position of the point (FPR, TPR) in the ROC space gives an indication of the performance of the classifier. For example, let us consider some special points in the space.

Special points in ROC space

1. **The left bottom corner point $(0, 0)$: Always negative prediction**

   A classifier which produces this point in the ROC space never classifies an example as positive, neither rightly nor wrongly, because for this point TP = 0 and FP = 0. It always makes negative predictions. All positive instances are wrongly predicted and all negative instances are correctly predicted. It commits no false positive errors.

2. **The right top corner point $(1, 1)$: Always positive prediction**

   A classifier which produces this point in the ROC space always classifies an example as positive because for this point FN = 0 and TN = 0. All positive instances are correctly predicted and all negative instances are wrongly predicted. It commits no false negative errors.

3. **The left top corner point $(0, 1)$: Perfect prediction**

   A classifier which produces this point in the ROC space may be thought as a perfect classifier. It produces no false positives and no false negatives.

4. **Points along the diagonal: Random performance**

   Consider a classifier where the class labels are randomly guessed, say by flipping a coin. Then, the corresponding points in the ROC space will be lying very near the diagonal line joining the points $(0, 0)$ and $(1, 1)$.

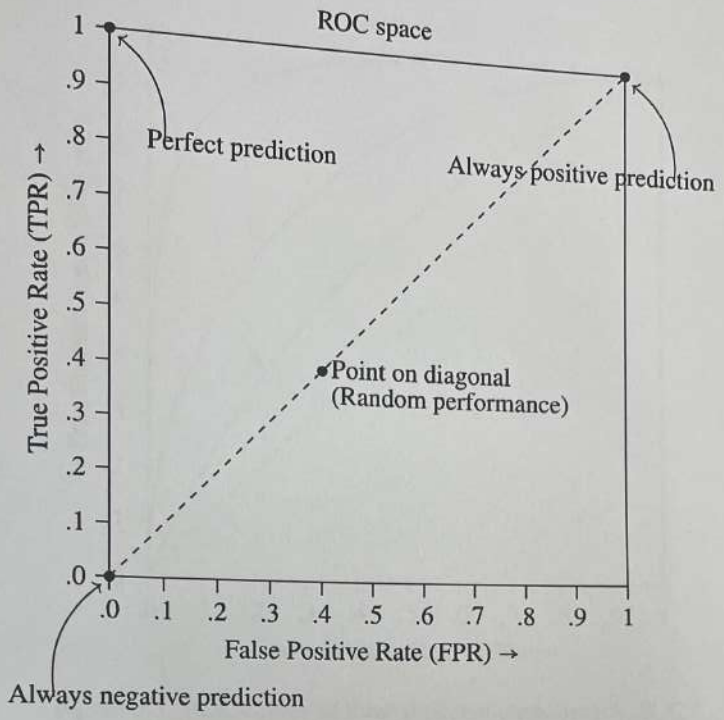

Figure 5.2: The ROC space and some special points in the space

ROC curve

In the case of certain classification algorithms, the classifier may depend on a parameter. Different values of the parameter will give different classifiers and these in turn give different values to TPR and FPR. The ROC curve is the curve obtained by plotting in the ROC space the points (TPR , FPR) obtained by assigning all possible values to the parameter in the classifier.

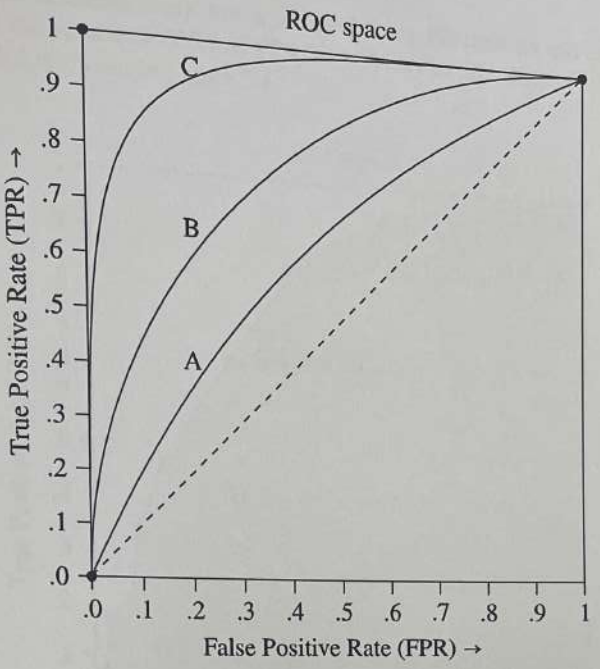

Figure 5.3: ROC curves of three different classifiers A, B, C

The closer the ROC curve is to the top left corner $(0, 1)$ of the ROC space, the better the accuracy of the classifier. Among the three classifiers A, B, C with ROC curves as shown in Figure 5.3, the classifier C is closest to the top left corner of the ROC space. Hence, among the three, it gives the best accuracy in predictions.

Example

Cut-off value of BMI	Breast cancer		Normal persons		TPR	FPR
	TP	FN	FP	TN		
18	100	0	200	0	1.00	1.000
20	100	0	198	2	1.00	0.990
22	99	1	177	23	0.99	0.885
24	95	5	117	83	0.95	0.585
26	85	15	80	120	0.85	0.400
28	66	34	53	147	0.66	0.265
30	47	53	27	173	0.47	0.135
32	34	66	17	183	0.34	0.085
34	21	79	14	186	0.21	0.070
36	17	83	6	194	0.17	0.030
38	7	93	4	196	0.07	0.020
40	1	99	1	199	0.01	0.005

Table 5.3: Data on breast cancer for various values of BMI

The body mass index (BMI) of a person is defined as (weight(kg)/height(m)^2). Researchers have established a link between BMI and the risk of breast cancer among women. The higher the BMI the higher the risk of developing breast cancer. The critical threshold value of BMI may depend on several parameters like food habits, socio-cultural-economic background, life-style, etc. Table 5.3

gives real data of a breast cancer study with a sample having 100 patients and 200 normal persons.[2] The table also shows the values of TPR and FPR for various cut-off values of BMI. The ROC curve of the data in Table 5.3 is shown in Figure 5.4.

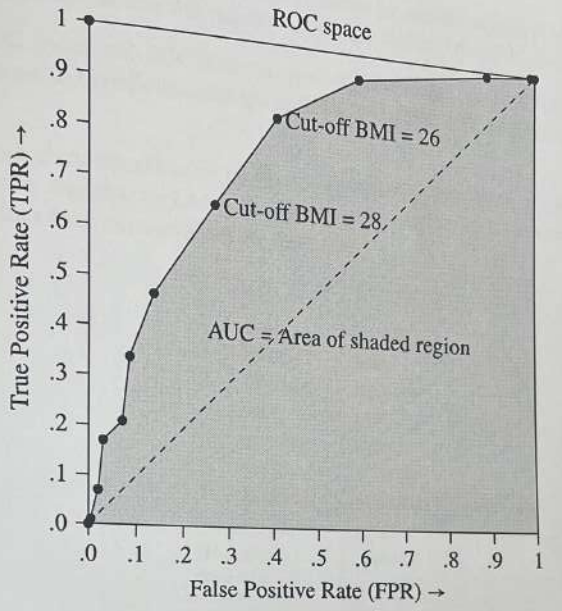

Figure 5.4: ROC curve of data in Table 5.3 showing the points closest to the perfect prediction point $(0, 1)$

Area under the ROC curve

The measure of the area under the ROC curve is denoted by the acronym AUC (see Figure 5.4). The value of AUC is a measure of the performance of a classifier. For the perfect classifier, AUC = 1.0.

5.6 Sample questions

(a) Short answer questions

1. What is cross-validation in machine learning?
2. What is meant by 5×2 cross-validation?
3. What is meant by leave-one-out cross validation?
4. What is meant by the confusion matrix of a binary classification problem.
5. Define the following terms: precision, recall, sensitivity, specificity.
6. What is ROC curve in machine learning?
7. What are true positive rates and false positive rates in machine learning?
8. What is AUC in relation to ROC curves?

[2] https://www.ncbi.nlm.nih.gov/ pmc/articles/PMC3755824/

(b) Long answer questions

1. Explain cross-validation in machine learning. Explain the different types of cross-validations.
2. What is meant by true positives etc.? What is meant by confusion matrix of a binary classification problem? Explain how this can be extended to multi-class problems.
3. What are ROC space and ROC curve in machine learning? In ROC space, which points correspond to perfect prediction, always positive prediction and always negative prediction? Why?
4. Consider a two-class classification problem of predicting whether a photograph contains a man or a woman. Suppose we have a test dataset of 10 records with expected outcomes and a set of predictions from our classification algorithm.

	Expected	Predicted
1	man	woman
2	man	man
3	woman	woman
4	man	man
5	woman	man
6	woman	woman
7	woman	woman
8	man	man
9	man	woman
10	woman	woman

   (a) Compute the confusion matrix for the data.

   (b) Compute the accuracy, precision, recall, sensitivity and specificity of the data.

5. Suppose 10000 patients get tested for flu; out of them, 9000 are actually healthy and 1000 are actually sick. For the sick people, a test was positive for 620 and negative for 380. For the healthy people, the same test was positive for 180 and negative for 8820. Construct a confusion matrix for the data and compute the accuracy, precision and recall for the data.
6. Given the following data, construct the ROC curve of the data. Compute the AUC.

Threshold	TP	TN	FP	FN
1	0	25	0	29
2	7	25	0	22
3	18	24	1	11
4	26	20	5	3
5	29	11	14	0
6	29	0	25	0
7	29	0	25	0

7. Given the following hypothetical data at various cut-off points of mid-arm circumference of mid-arm circumference to detect low birth-weight construct the ROC curve for the data.

Mid-arm circumference (cm)	Normal birth-weight TP	Low birth-weight TN
≤ 8.3	13	867
≤ 8.4	24	844
≤ 8.5	73	826
≤ 8.6	90	800
≤ 8.7	113	783
≤ 8.8	119	735
≤ 8.9	121	626
≤ 9.0	125	505
≤ 9.1	127	435
≤ 9.2 and above	130	0