K-Nearest Neighbors Algorithm for Classification

1. Introduction
2. K-Nearest Neighbors Classification
3. Example
4. Diagnostics of Classifiers - Part 1
5. N-Fold Cross-Validation

1 Introduction

2 K-Nearest Neighbors Classification

3 Example

4 Diagnostics of Classifiers - Part 1

5 N-Fold Cross-Validation

Categorical Response

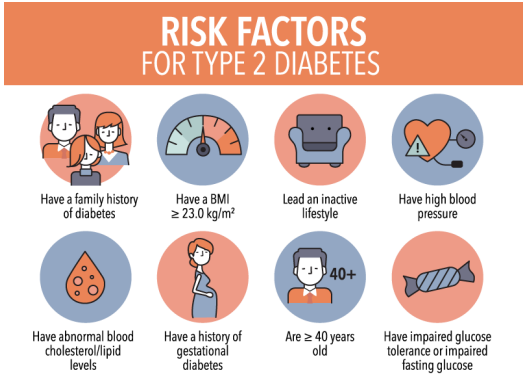
- Many learning tasks involve the following: given the value of predictors X, make a good prediction of the outcome y which we denote by $\hat{G}$.

- In some cases, the outcome y is categorical which typically represented numerically by codes. For example categorical with binary response is often represented by 0, 1.

- One approach is to treat the binary coded outcome y as a quantitative, to get the prediction. Then, assign class label to $\hat{G}$ according to whether the prediction is larger than δ which is a constant in (0,1), and is chosen case by case.

Example: Singapore's war against diabetes

- A person is either having diabetes (yes) or not (no).
- Based on X including age, gender, BMI and lifestyle choices, an online Diabetes Risk Assessment (DRA) was developed to predict whether a person is at risk to develop diabetes or not ($\widehat{G}$).



Source: `https://pss.hpb.gov.sg/DRA/GetQuestions`

1 Introduction

2 K-Nearest Neighbors Classification

3 Example

4 Diagnostics of Classifiers - Part 1

5 N-Fold Cross-Validation

Notations

- There are n training points with features x and categorical response y. Each (x_i, y_i) is the information of an observation.
- With the information x, the prediction on the category for the response is denoted as $\widehat{G}(x)$.
- For simple, **we only consider the cases of binary response** (0, 1) only. Hence, the prediction, $\widehat{G}(x)$, is either 0 or 1.

K-Nearest Neighbors Algorithm

- The k-nearest neighbors method uses training points closest in feature space to x to find

$$\frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} y_i,$$

where $\mathcal{N}_k(x)$ is the neighborhood of x defined as the set of k closest points (in terms of Euclidean distance).

- δ is a pre-chosen threshold, the average above is converted to $\widehat{G}(x)$ according to the rule

$$\widehat{G}(x) = \begin{cases} 1, & \text{if it's} > \delta, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Recall: Euclidean distance

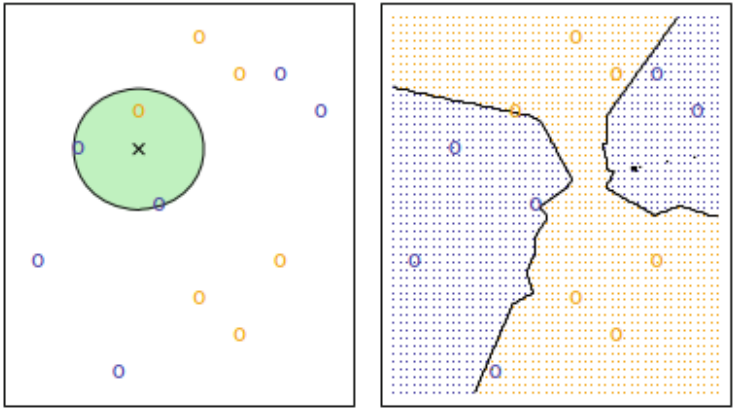
- For 2 dimensions (X_1, X_2), the Euclidean distance between point P with (x_1^P, x_2^P) and point Q with (x_1^Q, x_2^Q) is

$$\sqrt{(x_1^P - x_1^Q)^2 + (x_2^P - x_2^Q)^2}$$

- For the case of d dimensions $(X_1, X_2, ..., X_d)$, then the Euclidean distance between point P with $(x_1^P, x_2^P, ..., x_d^P)$ and point Q with $(x_1^Q, x_2^Q, ..., x_d^Q)$ is

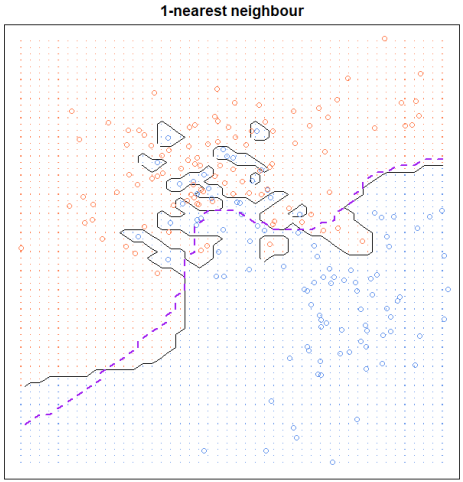
$$\sqrt{(x_1^P - x_1^Q)^2 + (x_2^P - x_2^Q)^2 + ... + (x_d^P - x_d^Q)^2}$$

K-Nearest Neighbors for Classification



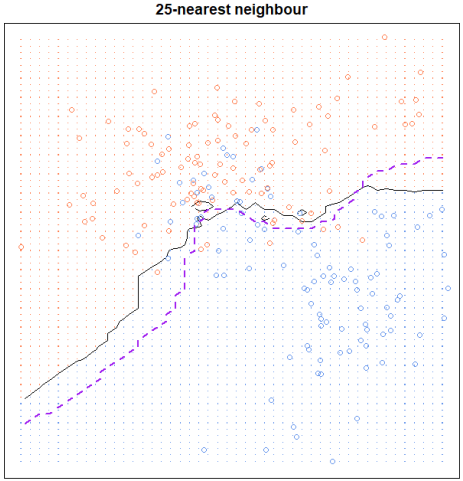
Blue$= 0$, orange$= 1$. The k-nearest neighbors classification using $k = 3$. Left: the predicted outcome $\widehat{Y}$ at the marked feature value is $1/3$, hence $\widehat{G} = 0$. Right: the k-nearest neighbors decision boundary. Source: *An Introduction to Statistical Learning*, James et al.

Small k



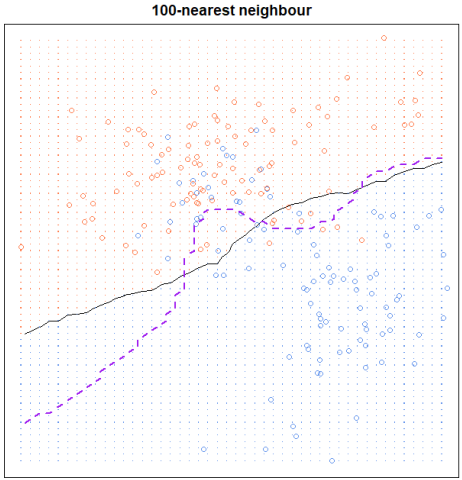
Blue$= 0$, orange$= 1$. The k-nearest neighbors classification using $k = 1$.

Larger k



Blue$= 0$, orange$= 1$. The k-nearest neighbors classification using $k = 25$.

Very large k



Blue$= 0$, orange$= 1$. The k-nearest neighbors classification using $k = 100$.

Bias-variance tradeoff

- When $k = 1$, the decision boundary is overly flexible and influenced by local features of a handful of training data points $\rightarrow$ low bias, high variance.

- When $k = 100$, the method yields more stable but less flexible decision boundaries $\rightarrow$ high bias, low variance.

1 Introduction

2 K-Nearest Neighbors Classification

3 Example

4 Diagnostics of Classifiers - Part 1

5 N-Fold Cross-Validation

Application: The Stock Market Data

We will use the `knn()` function from the 'class' package in R to perform k-nearest neighbors classification and prediction. We need the following four inputs to `knn()`:

(i) A matrix containing the predictors or features x associated with the training data

(ii) A matrix containing the predictors or features x associated with the data for which we wish to make predictions

(iii) A vector containing the class labels for the training data

(iv) A value for k, the number of nearest neighbors to be used by the classifier

Application: The Stock Market Data

- The CSV file `Smarket.csv` contains data on percentage returns for the S&P 500 stock index over 1250 days, from the beginning of 2001 until the end of 2005.

- For each date, the data contains the percentage returns for each of the five previous trading days, Lag1 through Lag5.

Extensions

A large subset of the most popular techniques in use today are variants of linear regression and k-nearest neighbors:

- Kernel methods use weights that decrease smoothly to zero with distance from the target point, rather than the effective $0/1$ weights used by k-nearest neighbors.
- In high-dimensional spaces the distance kernels are modified to emphasize some variable more than others.
- Local regression fits linear models by locally weighted least squares, rather than fitting constants locally.
- Linear models fit to a basis expansion of the original inputs allow arbitrarily complex models.
- Projection pursuit and neural network models consist of sums of non- linearly transformed linear models.

1 Introduction

2 K-Nearest Neighbors Classification

3 Example

4 Diagnostics of Classifiers - Part 1

5 N-Fold Cross-Validation

Diagnostics

- When a classifier is built, it helps to assign class labels to person, item or transaction. There is a need to evaluate its performance.

- In general, for two class labels, C and C', where C' denotes "not C", some working definitions and formulas follow:

- True Positive: Predict C, when actually C

- True Negative: Predict C', when actually C'

- False Positive: Predict C, when actually C'

- False Negative: Predict C', when actually C

Confusion Matrix

- We will study the *confusion matrix* which is a specific table layout that allows visualization of the performance of a classifier.

- In a two-class classification, confusion matrix is constructed as below.

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

Confusion Matrix

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

- TP and TN are the correct predictions.
- A good classifier should have large TP and TN; and has small numbers (ideally zero) for FP and FN.

Example: Classifying Spam Emails

- Based on an e-mail's content, e-mail providers use classification methods to decide whether the incoming e-mail messages are spam.
- Based on features such as the presence of certain keywords and images (X), classification methods assign an incoming email a label as "spam" or "non-spam" class (y).
- From a training set of emails, a classifier by k-nearest neighbor has been built.

Example: Classifying Spam Emails

		Predicted Class		
		Spam	**Non-Spam**	Total
Actual Class	**Spam**	3	8	11
	Non-Spam	2	87	89
Total		5	95	100

- A testing set has 100 emails (with their spam or non-spam label known).
- Above is the confusion matrix.

Diagnostics of Classifiers: The 5 Metrics

- There are many criteria to evaluate. For now, we consider 5 basic measurements below.

  Accuracy; True Positive Rate (TPR); False Positive Rate (FPR - Type I error); False Negative Rate (FNR - Type II error) and Precision.

- Few more measurements (such as ROC curve and AUC value) will be introduced in other chapters.

Diagnostics of Classifiers: Accuracy

- The *accuracy* (or the overall success rate) is a metric defining the rate at which a model has classified the records correctly.

- It is defined as the sum of TP and TN divided by the total number of instances:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$

- A good model should have a high accuracy score, but having a high accuracy score alone does not guarantee the model is well established.

Diagnostics of Classifiers: True Possitive Rate

- The true positive rate (TPR) shows the proportion of positive instances the classifier correctly identified:

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

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

Diagnostics of Classifiers: False Possitive Rate

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

- The false positive rate (**FPR**) shows the percentage of negatives the classifier marked as positive.
- The FPR is also called the false alarm rate or Type I error rate

$$\text{FPR} = \frac{FP}{FP + TN}$$

Diagnostics of Classifiers: False Negative Rate

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

- The false negative rate (FNR) shows the percent of positives the classifier marked as negatives.
- It is also known as the miss rate or Type II error rate.

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

Diagnostics of Classifiers: Precision

- **Precision** is the percentage of instances that are actually positive among the marked positives.

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

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

Diagnostics of Classifiers: General

- A well-performed classifier should have a high TPR (ideally 1) and a low FPR and FNR (ideally 0).

- In reality, it is rare to have TPR $= 1$, FPR $= 0$, and FNR $= 0$, but these measures are useful to compare the performance of multiple classifiers that are designed for solving the same problem.

Diagnostics of Classifiers in Practice

- Note that in general, the model that is more preferable may depend on the business situation.
- During the discovery phase of the data analytics lifecycle, the team should have learned from the business what kind of errors can be tolerated.
- Some business situations are more tolerant of Type I errors, whereas others may be more tolerant of Type II errors.

Example: Email Spam Filtering

- Consider the example of email spam filtering.
- Some people (such as busy executives) only want important email in their inbox and are tolerant of having some less important email end up in their spam folder as long as no spam is in their inbox.
- In this case, a higher false positive rate (FPR) or type I error can be tolerated.

Example: Email Spam Filtering

- Other people may not want any important or less important email to be specified as spam and are willing to have some spams in their inbox as long as no important email is sent to the spam folder.

- In this case, a higher false negative rate (FNR) or type II error can be tolerated.

Example: Medical Screening

- Another example involves medical screening during an infectious disease outbreak.
- The cost of having a person, who has the disease, to be instead diagnosed as disease-free is extremely high, since the disease may be highly contagious.
- Therefore, the false negative rate (FNR) or type II error needs to be low.
- A higher false positive rate (FPR) or type I error can be tolerated.

Example: Security Screening

- Third example involves security screening at the airport.
- The cost of a false negative in this scenario is extremely high (not detecting a bomb being brought onto a plane could result in hundreds of deaths) whilst the cost of a false positive is relatively low (a reasonably simple further inspection)
- Therefore, a higher false positive rate (FPR) or type I error can be tolerated, in order to keep the false negative rate (FNR) or type II error low.

Filtering Spam Email: Calculation

		Predicted Class		
		Spam	**Non-Spam**	Total
Actual Class	**Spam**	3	8	11
	Non-Spam	2	87	89
Total		5	95	100

- $$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% = \frac{3 + 87}{3 + 87 + 2 + 8} \times 100\% = 90\%$$
- $$\text{TPR} = \frac{TP}{TP + FN} = \frac{3}{3 + 8} \approx 0.273$$
- $$\text{FPR} = \frac{FP}{FP + TN} = \frac{2}{2 + 87} \approx 0.022$$

Filtering Spam Email: Calculation

		Predicted Class		
		Spam	**Non-Spam**	Total
Actual Class	**Spam**	3	8	11
	Non-Spam	2	87	89
Total		5	95	100

- $$\text{FNR} = \frac{FN}{TP + FN} = \frac{8}{3 + 8} \approx 0.727$$

- $$\text{Precision} = \frac{TP}{TP + FP} = \frac{3}{3 + 2} = 0.6$$

1 Introduction

2 K-Nearest Neighbors Classification

3 Example

4 Diagnostics of Classifiers - Part 1

5 N-Fold Cross-Validation

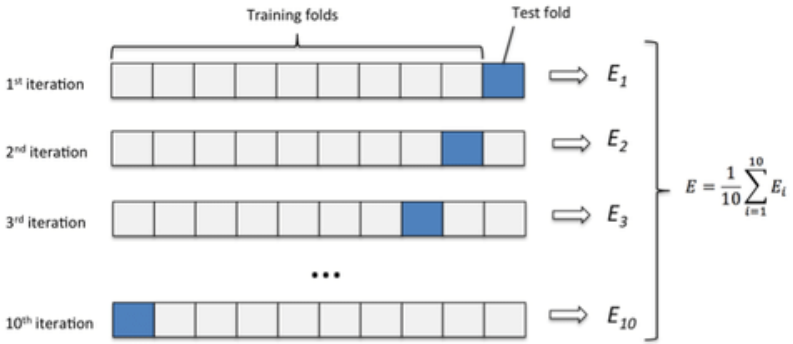
N-Fold Cross-Validation

- We have some measurements that can be used to evaluate the performance of a classifier.
- In practice, when we are presented with a data set, how should we go about estimating these performance measures?
- A common practice is to perform N-Fold Cross-Validation

What is N-Fold Cross-Validation?

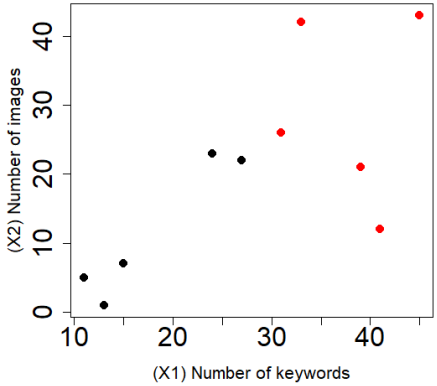
- The entire data set is randomly split into N data sets of approximately equal size.
- (N-1) of these data sets are treated as the training data set, while the remaining one is the test data set. A measure of the model error is obtained.
- This process is repeated across the various combinations of N data sets taken $(N-1)$ at a time.
- The observed N models errors are averaged across the N folds.

N-Fold Cross-Validation



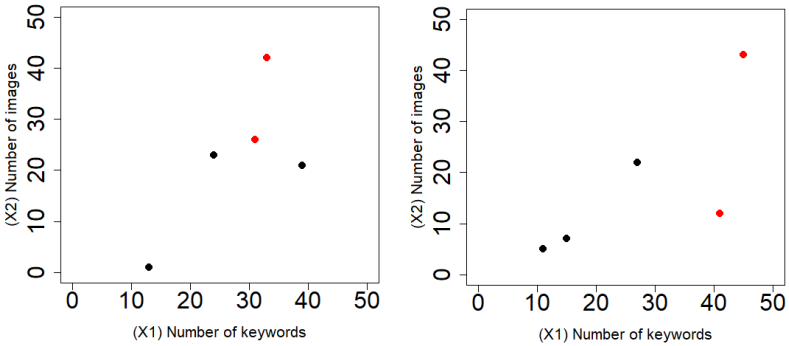
Example: Anti-Spam Techniques

- Let us illustrate N-Fold Cross-Validation with an example with the k-nearest neighbor classifier for spams, where we specify $k = 1$.
- Suppose our data set consists of 10 data points. Each point is labelled as spam (red, $y = 1$) or non-spam (black, $y = 0$)



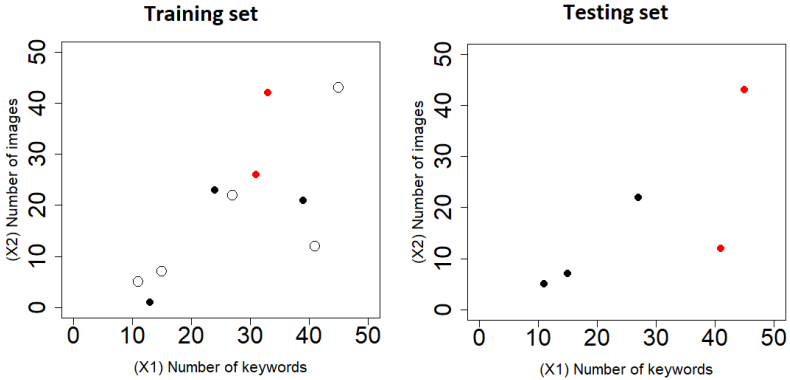
Anti-Spam Techniques: 2-fold CV

- For 2-fold cross validation, we randomly split the whole data set of 10 points into two data sets of 5 points each.

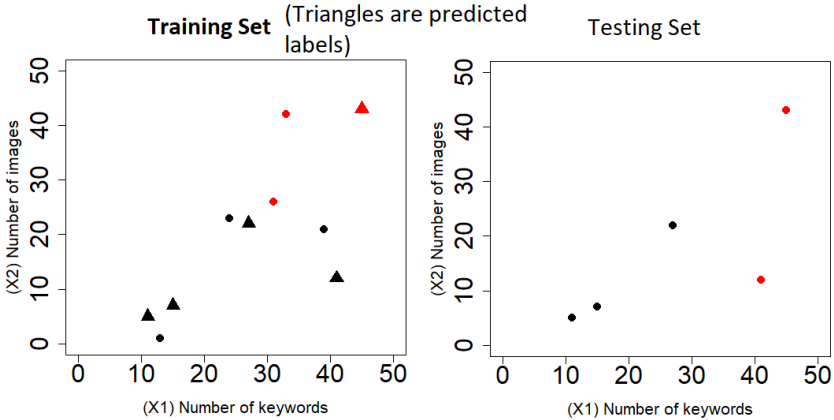


Anti-Spam Techniques: 2-fold CV

- For the first iteration, we use the first data set as the training set and the second data set as the testing set.



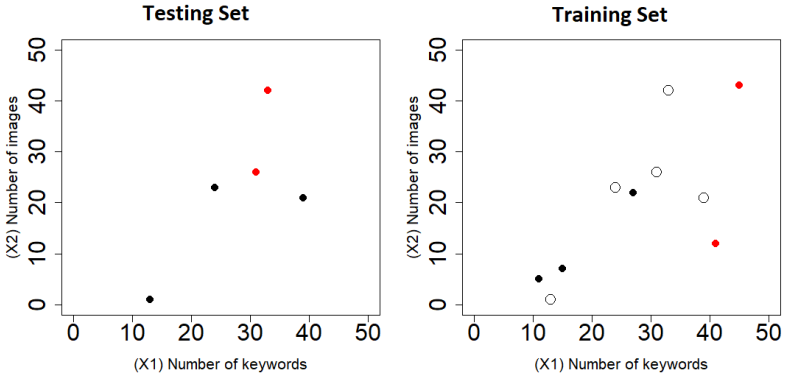
Anti-Spam Techniques: 2-fold CV



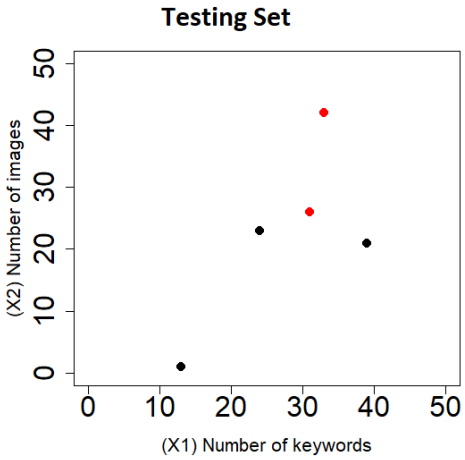
- In this iteration, we estimate the accuracy of the 1-nearest neighbor algorithm to be $\frac{4}{5}$.

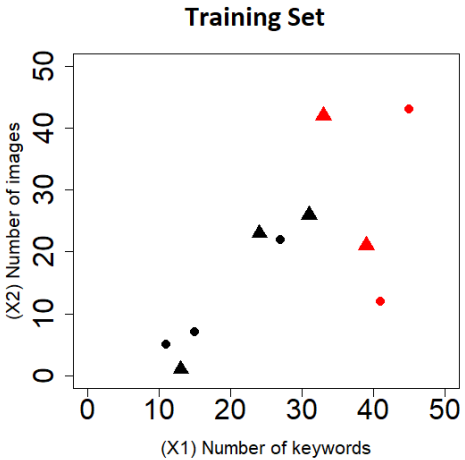
Anti-Spam Techniques: 2-fold CV

- For the second iteration, we use the second data set as the training set and the first data set as the testing set.



Anti-Spam Techniques: 2-fold CV





Anti-Spam Techniques: 2-fold CV

- In the second iteration, we estimate the accuracy of the 1-nearest neighbor algorithm to be equal to $\frac{3}{5}$

- Therefore, based on 2-fold CV, the **accuracy** of the 1-nearest neighbor algorithm is estimated to be $\left(\frac{4}{5}+\frac{3}{5}\right)/2=\frac{7}{10}$.

- One might use other criteria instead of **accuracy** (such as TPR, Precision, etc.) to evaluate the classifier.