Supervised Learning Methods: K-Nearest Neighbors Algorithm for Classification

1. Introduction
2. Linear Model for Classification
3. K-Nearest Neighbors Classification
4. Diagnostics of Classifiers
5. N-Fold Cross-Validation

1 Introduction

2 Linear Model for Classification

3 K-Nearest Neighbors Classification

4 Diagnostics of Classifiers

5 N-Fold Cross-Validation

Supervised Learning

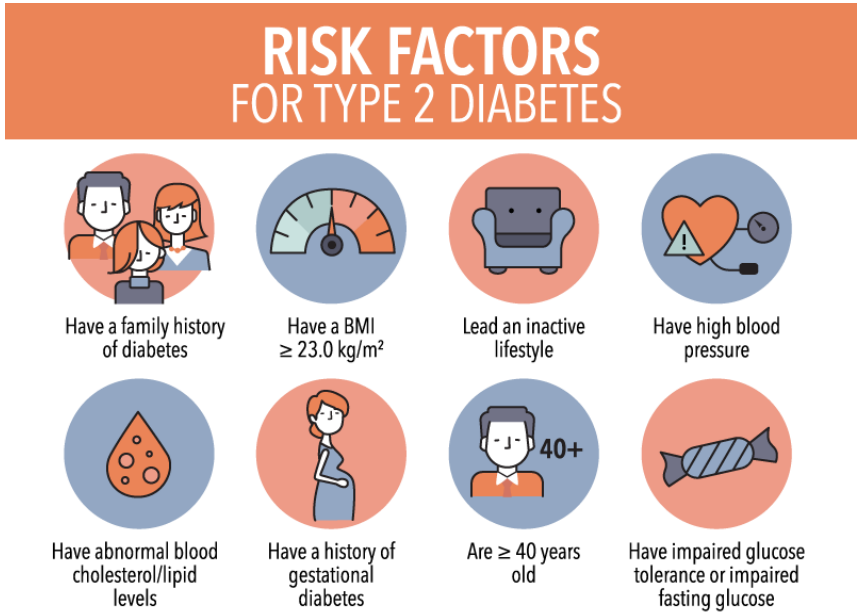
- Many learning tasks involve the following: given the value of predictors X, make a good prediction of the outcome G which we denote by $\hat{G}$.

- In some cases, the outcome G is a categorical variable which typically represented numerically by codes, for example two-class memberships are often represented by the binary codes 0 or 1.

- One approach is to treat the binary coded outcome as a quantitative outcome Y. The predictions $\hat{Y}$ will typically lie in $[0, 1]$, and we can assign to $\hat{G}$ the class label according to whether $\hat{Y} > 0.5$.

Example: Singapore's war against diabetes

- "In setting the battle scene, Health Minister Gan Kim Yong said the disease is already costing the country more than $1 billion a year. Of the more than 400,000 diabetics today, one in three do not even know they have the disease."
  - *The Straits Times*, April 13, 2016
- Based on X such as 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 (G).



Source: `https://www.gov.sg/factually/content/can-you-develop-diabetes`

1 Introduction

2 Linear Model for Classification

3 K-Nearest Neighbors Classification

4 Diagnostics of Classifiers

5 N-Fold Cross-Validation

Linear Regression in a Classification Context

- Suppose we are involved in developing the DRA to predict the binary outcome G of whether a person is at risk to develop diabetes or not based on BMI (X_1) and age (X_2).
- The response Y is coded as 0 for not at risk and 1 for at risk.
- In order to construct the prediction rules, we assume that the *training* data $\{(x_i, y_i), i = 1, ..., n\}$ is available, where $x = (x_1, x_2)$.
- The training data 'supervises' the construction of prediction rules: given value for x, how to predict the corresponding outcome y.
- Recall from Topic 3 (Linear Regression), one way is to predict via a model

$$\widehat{Y}(x) = \widehat{f}_1(x) = \widehat{\beta}_0 + \widehat{\beta}_1 x_1 + \widehat{\beta}_2 x_2.$$

Linear Regression in a Classification Context

- We obtain the *least squares* parameter estimate $(\widehat{\beta}_0, \widehat{\beta}_1, \widehat{\beta}_2)$ as the minimizer of the residual sum of squares defined using the training data.

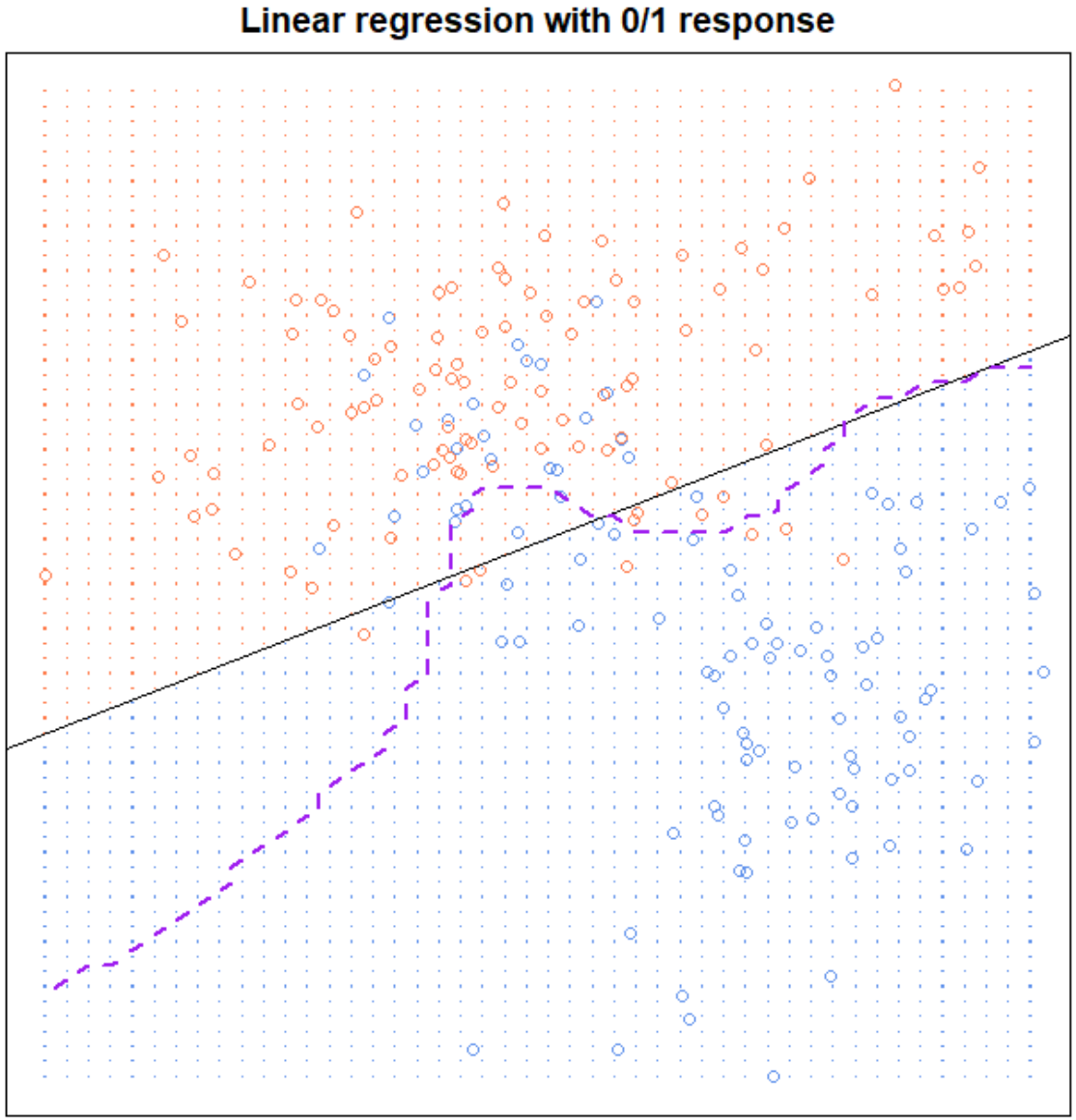
- The fitted value $\widehat{Y}(x)$ is converted to fitted class membership $\widehat{G}(x)$ according to the rule

$$\widehat{G}(x) = \begin{cases} 1, & \text{if } \widehat{Y}(x) = \widehat{\beta}_0 + \widehat{\beta}_1 x_1 + \widehat{\beta}_2 x_2 > 0.5, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

- The *decision boundary* is therefore defined by

$$0.5 = \widehat{\beta}_0 + \widehat{\beta}_1 x_1 + \widehat{\beta}_2 x_2.$$

Linear Regression in a Classification Context



Blue$= 0$, orange$= 1$. Solid line indicates linear decision boundary. Dotted purple line indicates Bayes decision boundary.

1 Introduction

2 Linear Model for Classification

3 K-Nearest Neighbors Classification

4 Diagnostics of Classifiers

5 N-Fold Cross-Validation

K-Nearest Neighbors Algorithm

- The k-nearest neighbors method use those observations in the training data closest in feature space to x to form $\widehat{Y}(x)$.

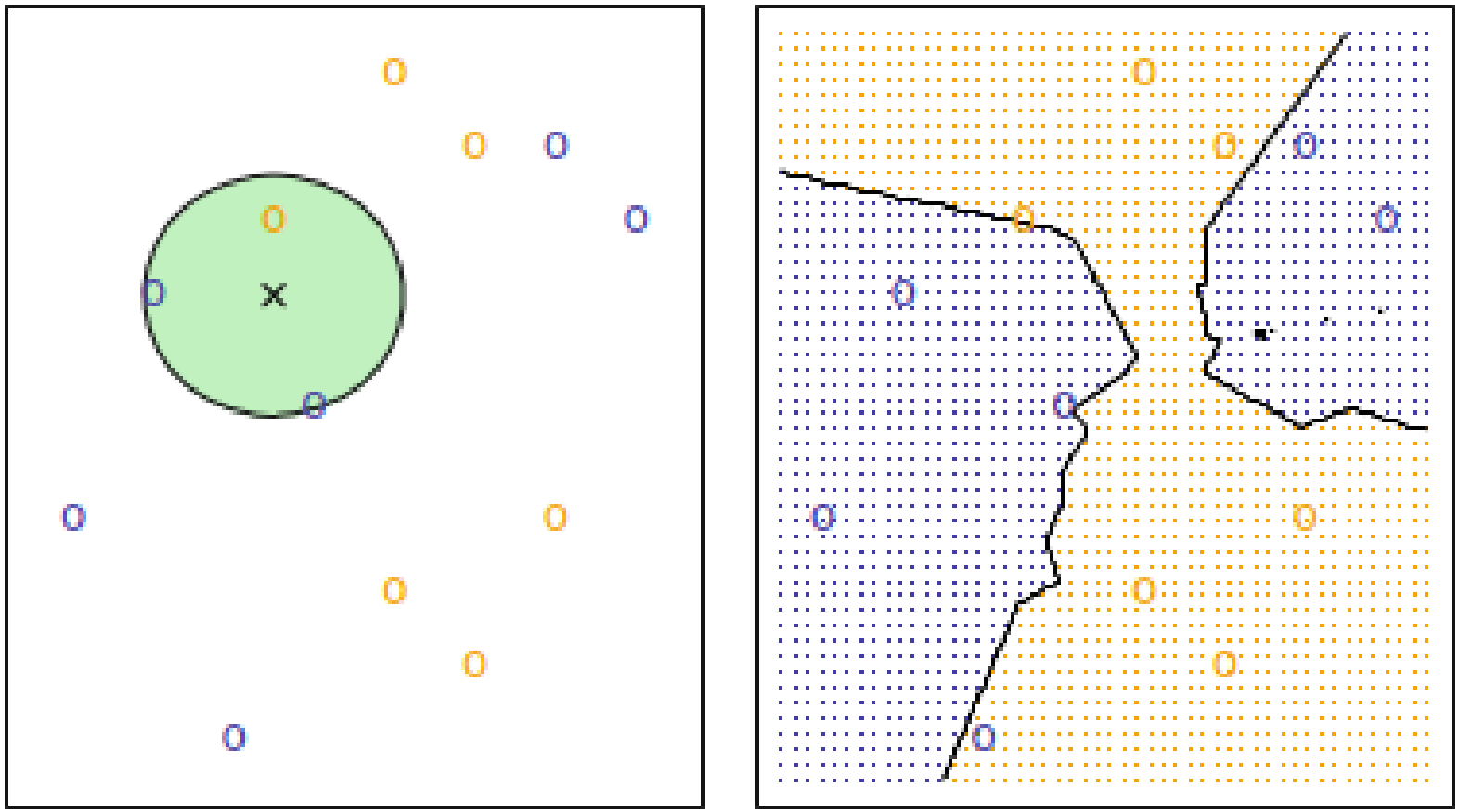
- Specifically,

$$\widehat{Y}(x) = \widehat{f}_2(x) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} y_i,$$

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

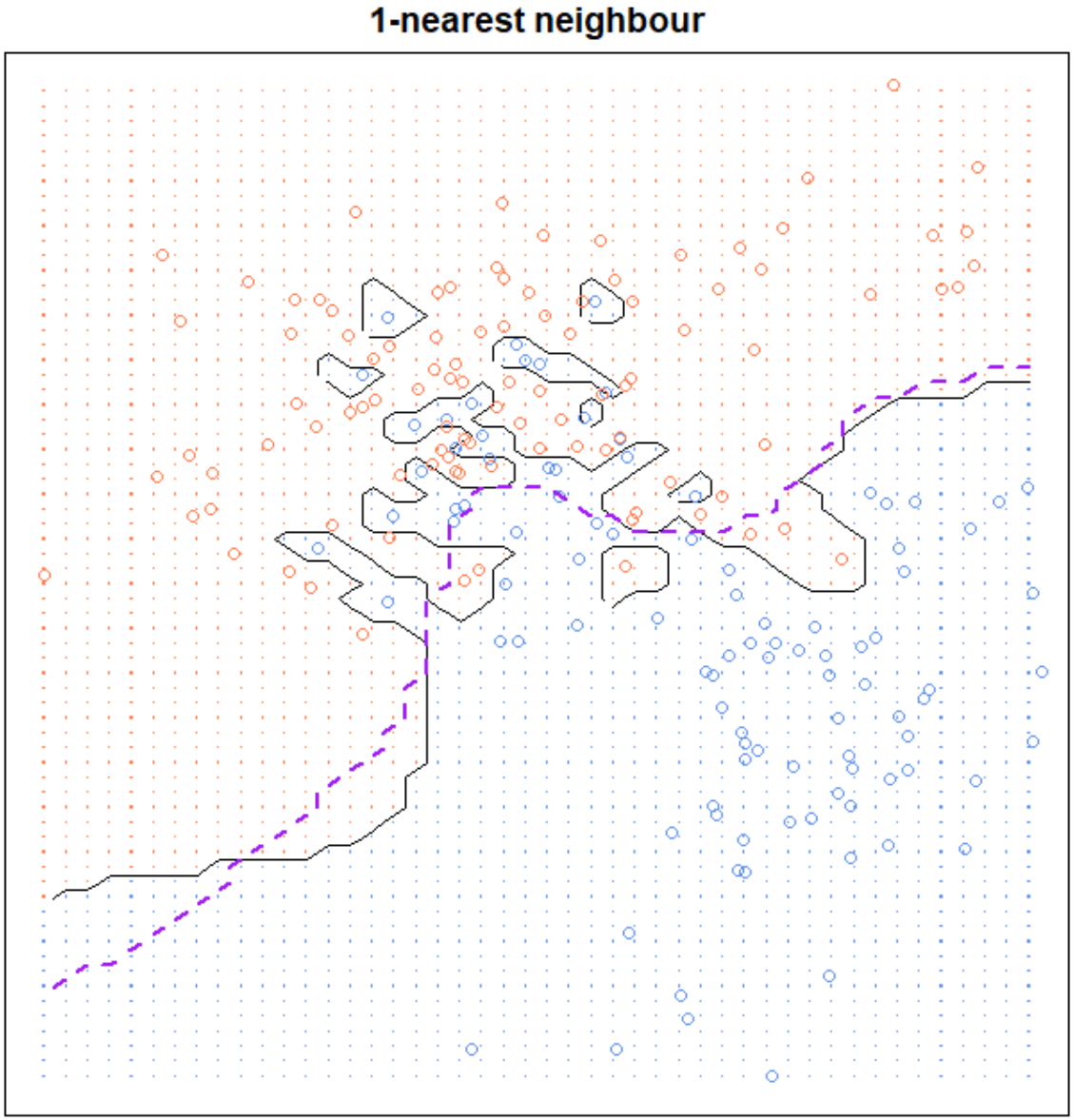
- Assign $\widehat{G}(x) = 1$ if $\widehat{Y}(x) > 0.5$.

K-Nearest Neighbors for Classification



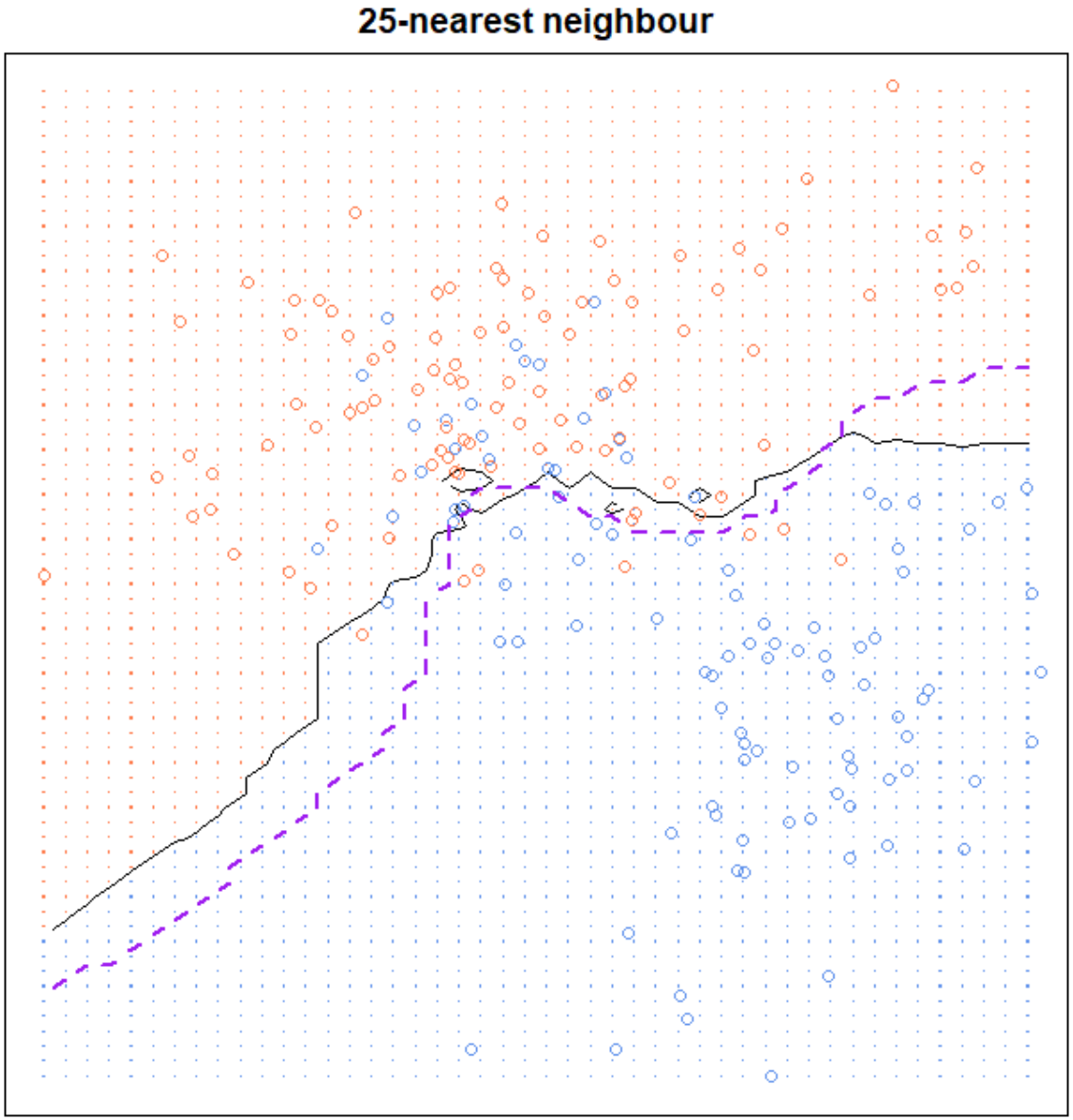
Blue$= 0$, orange$= 1$. The k-nearest neighbors classification using $k = 3$. Left: the predicted outcome $\widehat{Y}$ at the marked feature value is $1/3$, so that $\widehat{G} = 0$. Right: the k-nearest neighbors decision boundary. Source: *An Introduction to Statistical Learning*, James et al.

Small k: low bias, high variance



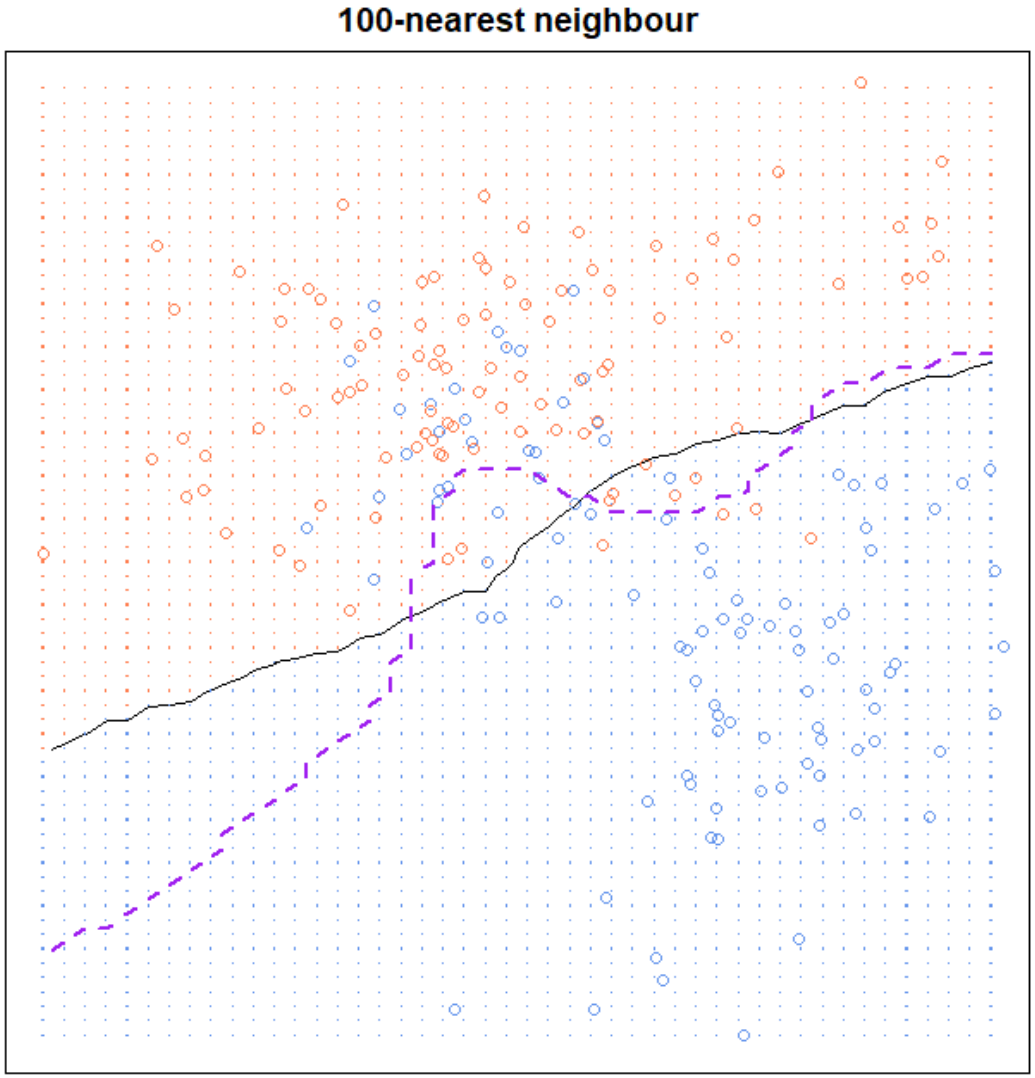
Blue$= 0$, orange$= 1$. The k-nearest neighbors classification using $k = 1$.

k-nearest neighbors classification



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

k-nearest neighbors classification



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

Bias-variance tradeoff

- The *effective* number of parameters of k-nearest neighbors is n/k, which increases with decreasing k. If $k = 1$, then there would be n partitions and we would fit one parameter (a mean) in each partition of the feature space.

- 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.

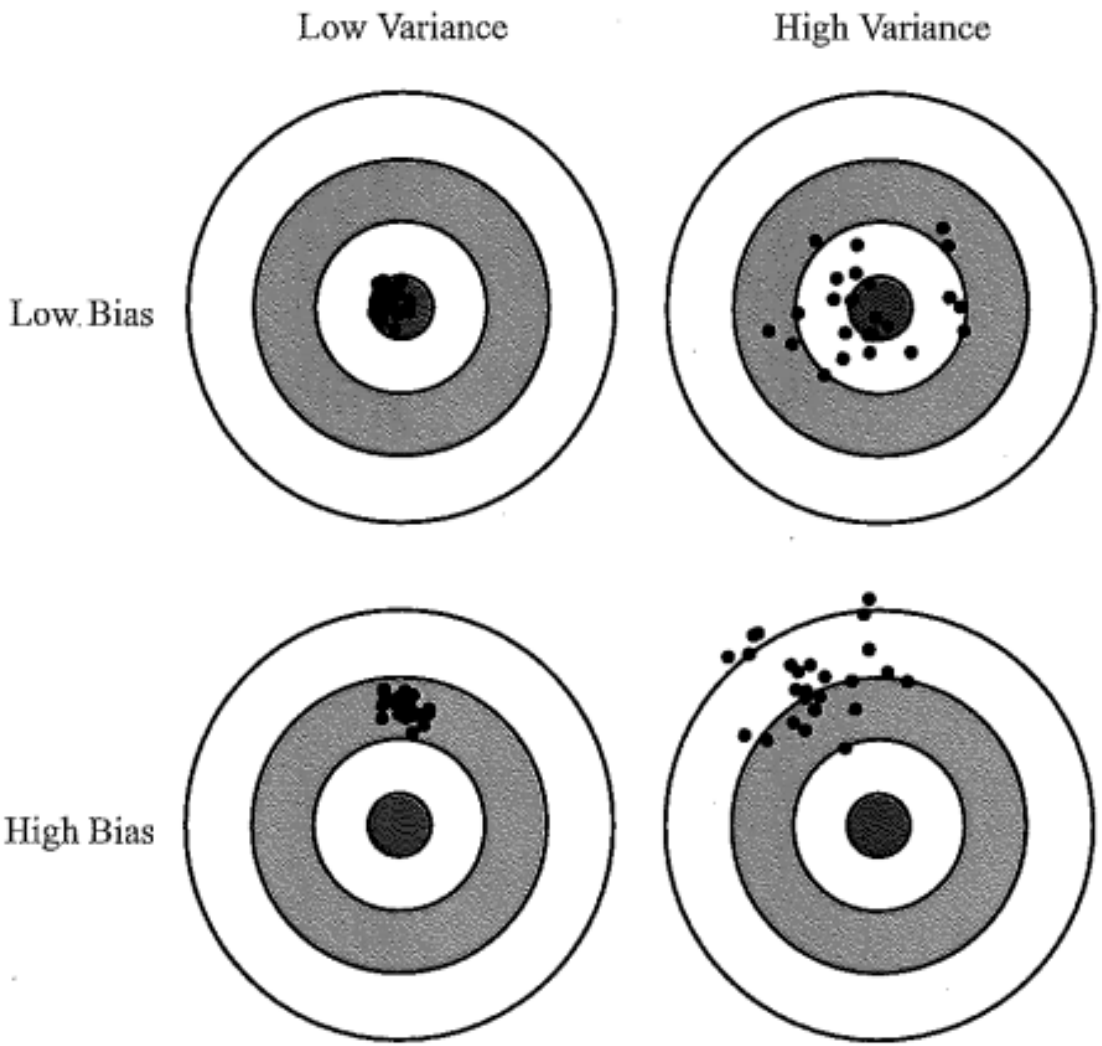
- When $k = 100$, the method yields more stable but less flexible decision boundaries $\rightarrow$ high bias, low variance.

- The training error rate

$$n^{-1} \sum_{i=1}^{n} I\{\widehat{G}(x_i) \neq y_i\},$$

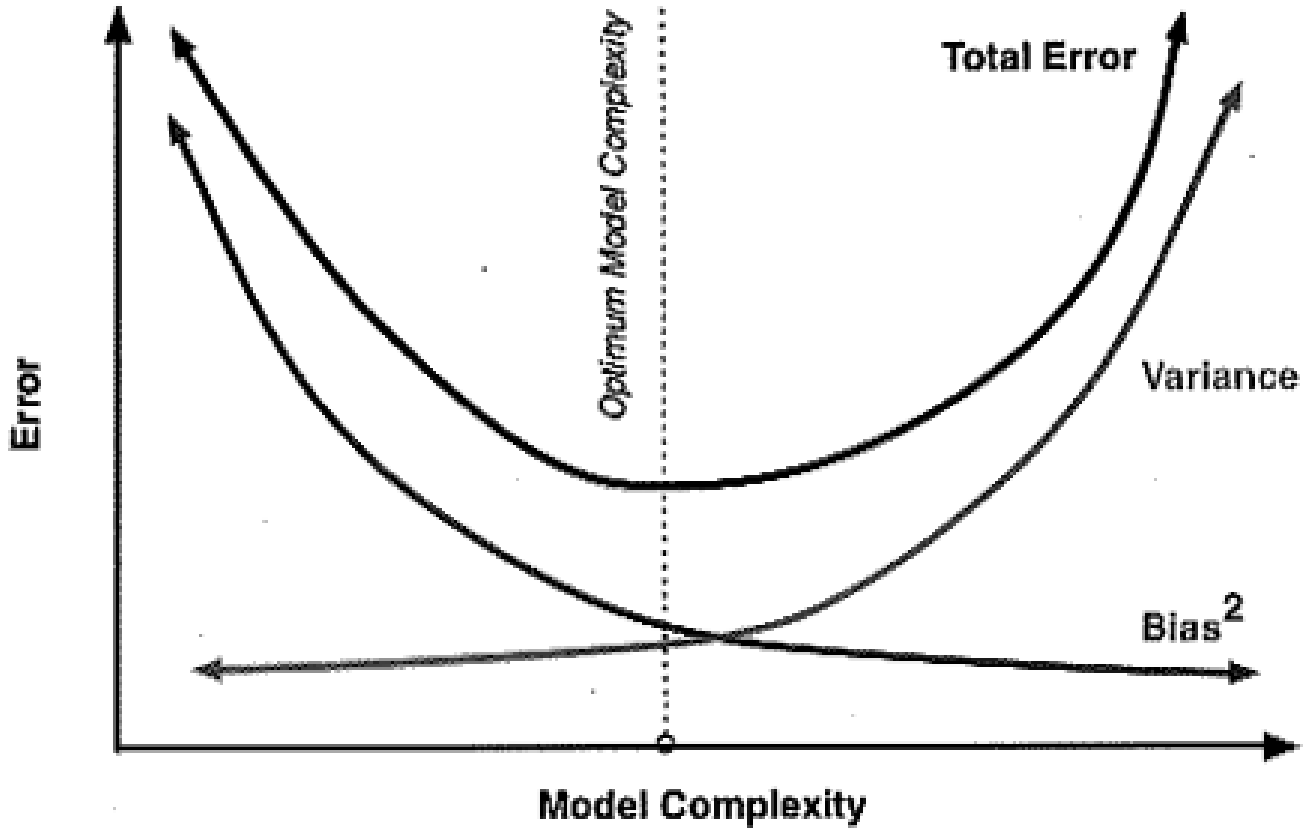
where $I(\cdot)$ is the indicator function, is not a good criterion for picking k.

Bias-variance tradeoff



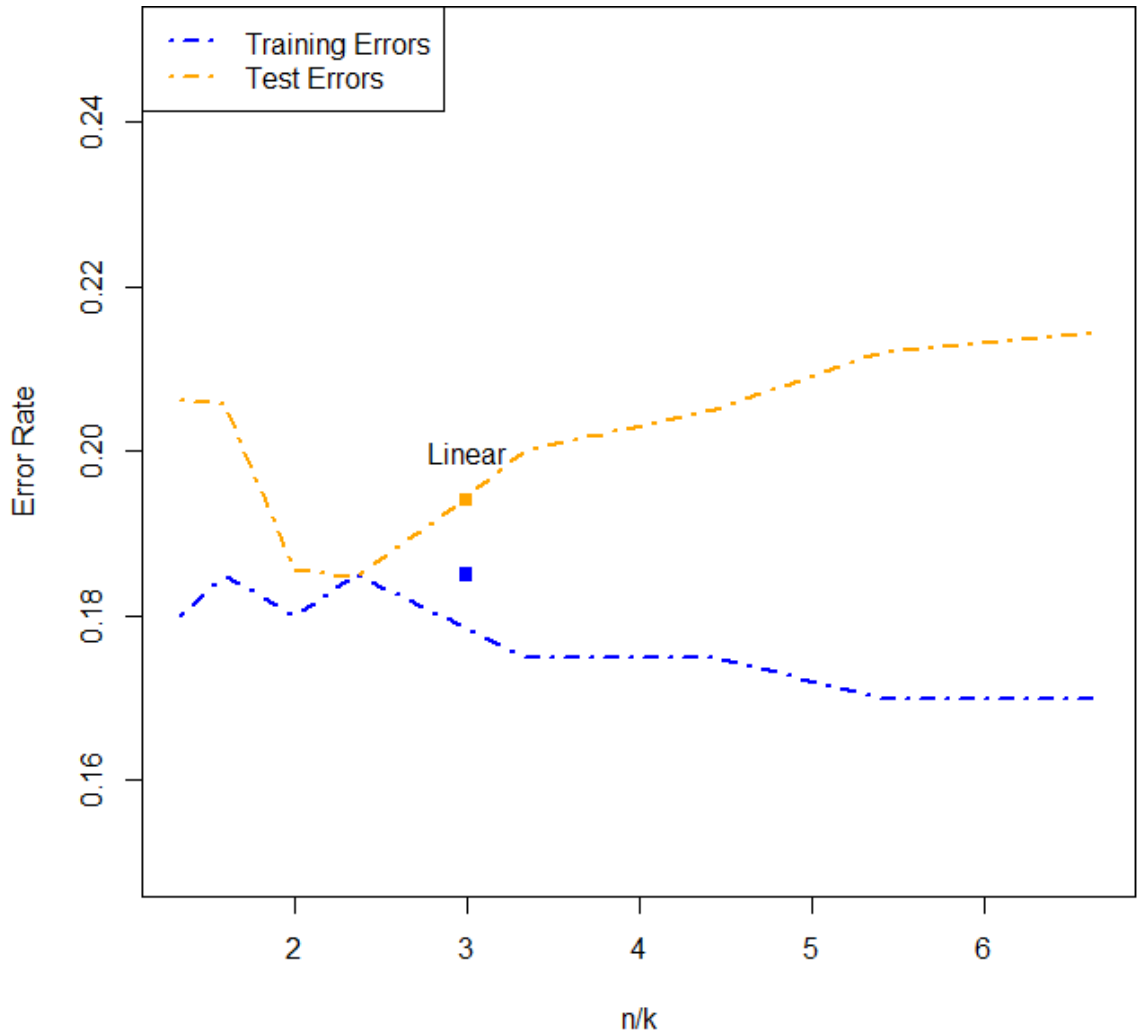
Graphical illustration of bias and variance. Source: `https://scott.fortmann-roe.com/docs/BiasVariance.html`

Bias-variance tradeoff



Bias and variance contributing to total error. Source: `https://scott.fortmann-roe.com/docs/BiasVariance.html`

Bias-variance tradeoff



The k-nearest neighbors training error rate (blue, $n = 200$ observations) and test error rate (orange, 10,000 observations), as the *effective* number of parameters n/k increases. The results for linear regression are indicated by filled squares.

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.

```
> market = read.csv("C:/Data/Smarket.csv")
> dim(market)
[1] 1250   10
> #summary(market[,2:10])
> train =(market$Year <2005)
> train.data = market[train,]
> test.data  = market[!train ,]
> dim(test.data)
[1] 252  10
```

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 Linear Model for Classification

3 K-Nearest Neighbors Classification

4 Diagnostics of Classifiers

5 N-Fold Cross-Validation

- We have studied the k-nearest neighbor algorithm as an example of a classifier.
- However, there is a need to evaluate the performance of the classifiers.
- k-nearest neighbor is often used as a classifier to assign class labels to a person, item, or transaction.
- 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, a preset threshold may be used to separate positives from negatives (e.g. we used the majority rule, $\hat{Y} < 0.5$, in the k-nearest neighbor example).

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

Confusion Matrix

- TP and TN are the correct guesses.

- A good classifier should have large TP and TN and small (ideally zero) numbers for FP and FN.

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

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 presence of certain keywords and images (X), classification methods assign a given email to the "spam" or "non-spam" class (y).
- Here the outcome y has two classes.

Example: Classifying Spam Emails

- A testing set of 100 emails (with their spam or non-spam label known).

- Example confusion matrix of a k-nearest neighbor classifier to predict if each email is spam or not

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

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.

- We will introduce more fine-grained measures better evaluate the performance of a classifier.

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

- The false positive rate (FPR) shows what percent of negatives the classifier marked as positive.

- The FPR is also called the false alarm rate or **Type I error rate**

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

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

Diagnostics of Classifiers: False Negative Rate

- The false negative rate (FNR) shows what percent of positives the classifier marked as negatives.

- It is also known as the miss rate or type II error rate.

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

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

Diagnostics of Classifiers: Precision

- **Precision** is the percentage of instances marked positive that really are positive:

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

		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 model should have a high TPR that is ideally 1 and a low FPR and FNR that are 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 models 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 spam in their inboxes as long as no important email makes it into 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 Linear Model for Classification

3 K-Nearest Neighbors Classification

4 Diagnostics of Classifiers

5 N-Fold Cross-Validation

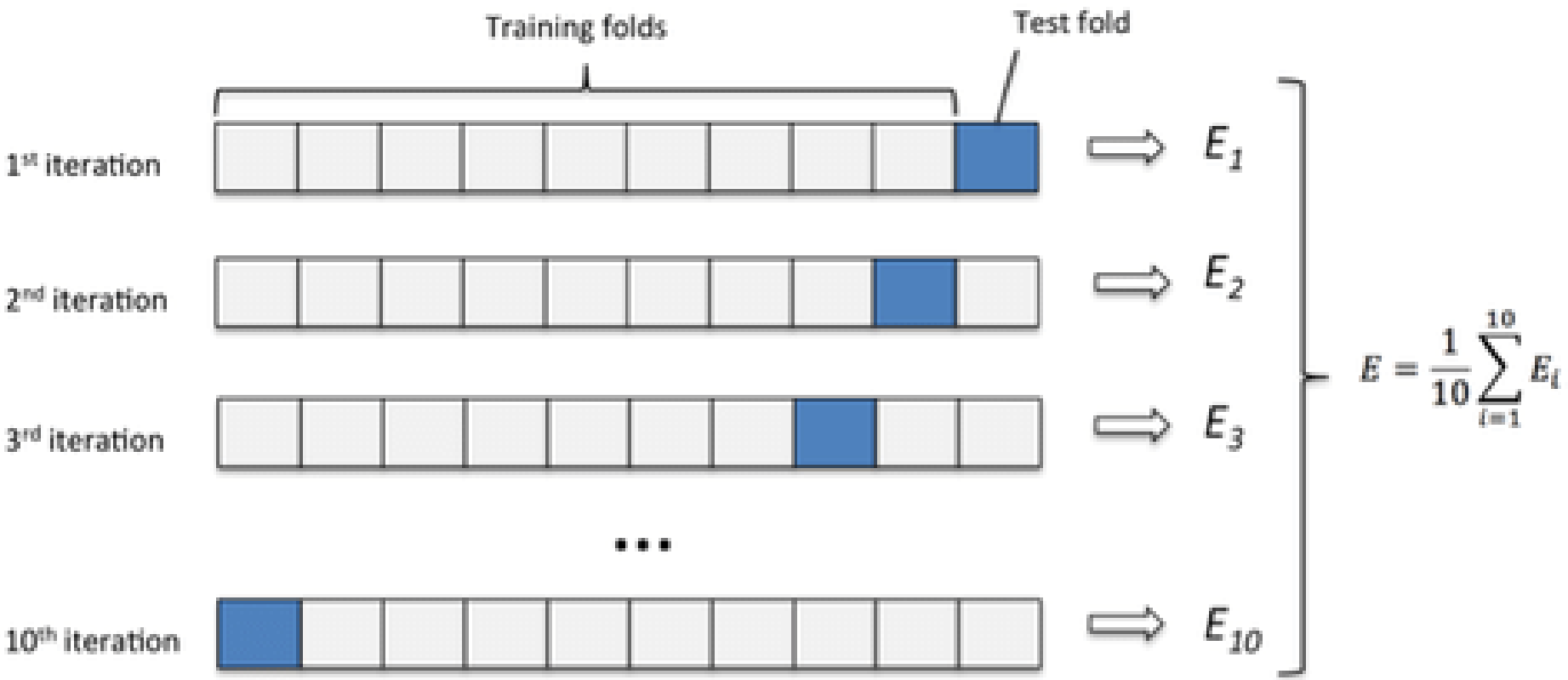
N-Fold Cross-Validation

- We have some measurements that can be used to evaluate the performance of a classifier.
- In practice, when we are presented with a dataset, 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?

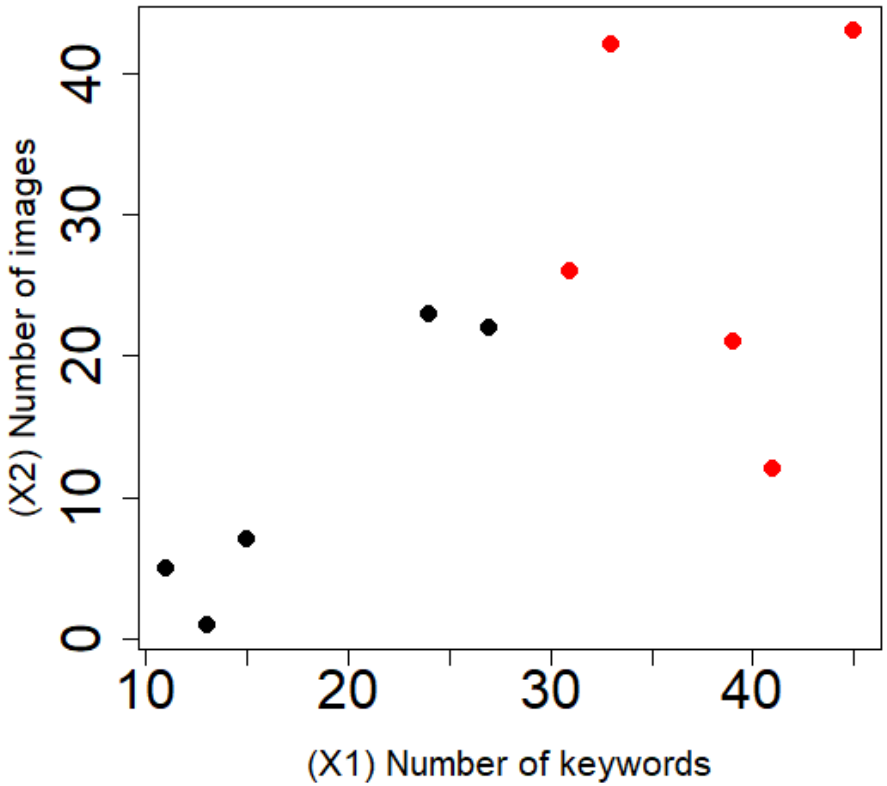
- The entire data set is randomly split into N data sets of approximately equal size.
- (N-1) of these data sets are treated as the training data set, while the remaining one is the test data set. A measure of the model error is obtained.
- This process is repeated across the various combinations of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

N-Fold Cross-Validation



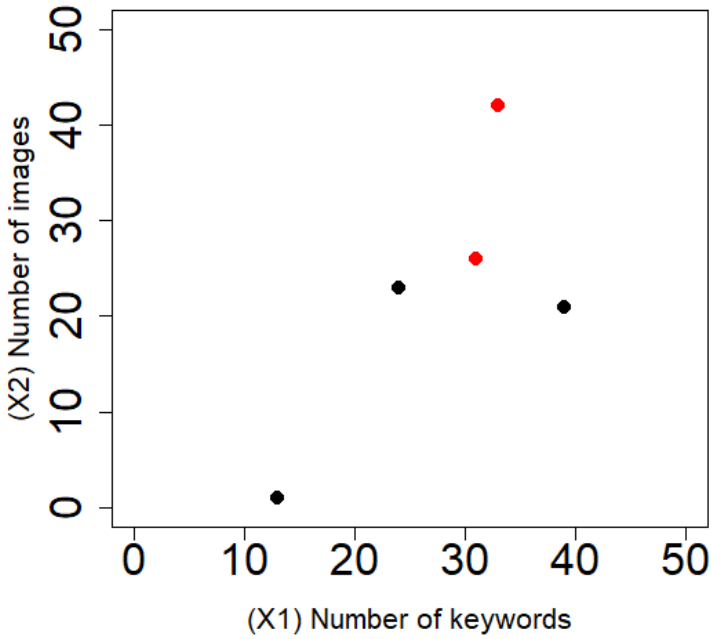
Example: Anti-Spam Techniques

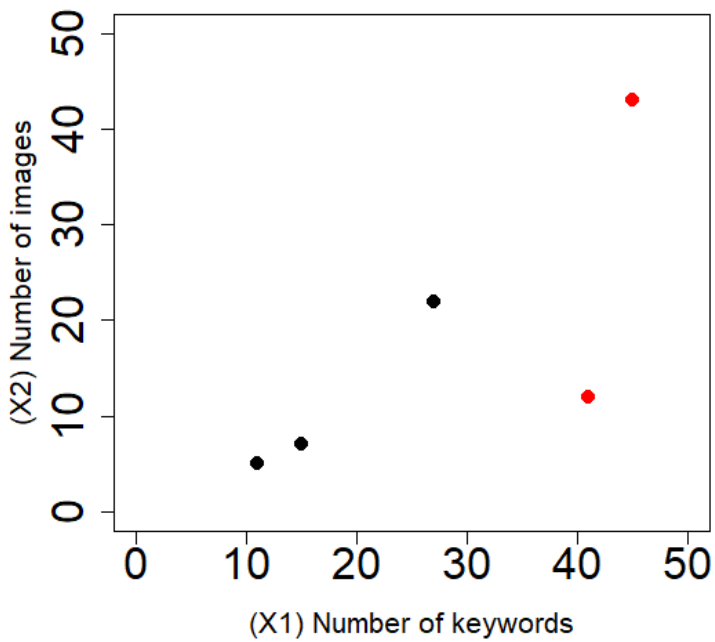
- Let us illustrate N-Fold Cross-Validation with an example with the k-nearest neighbor classifier for spams, where we specify $k = 1$.
- Suppose our data set consists of 10 data points. Each point is labelled as spam (red, $y = 1$) or non-spam (black, $y = 0$)



Anti-Spam Techniques: 2-fold CV

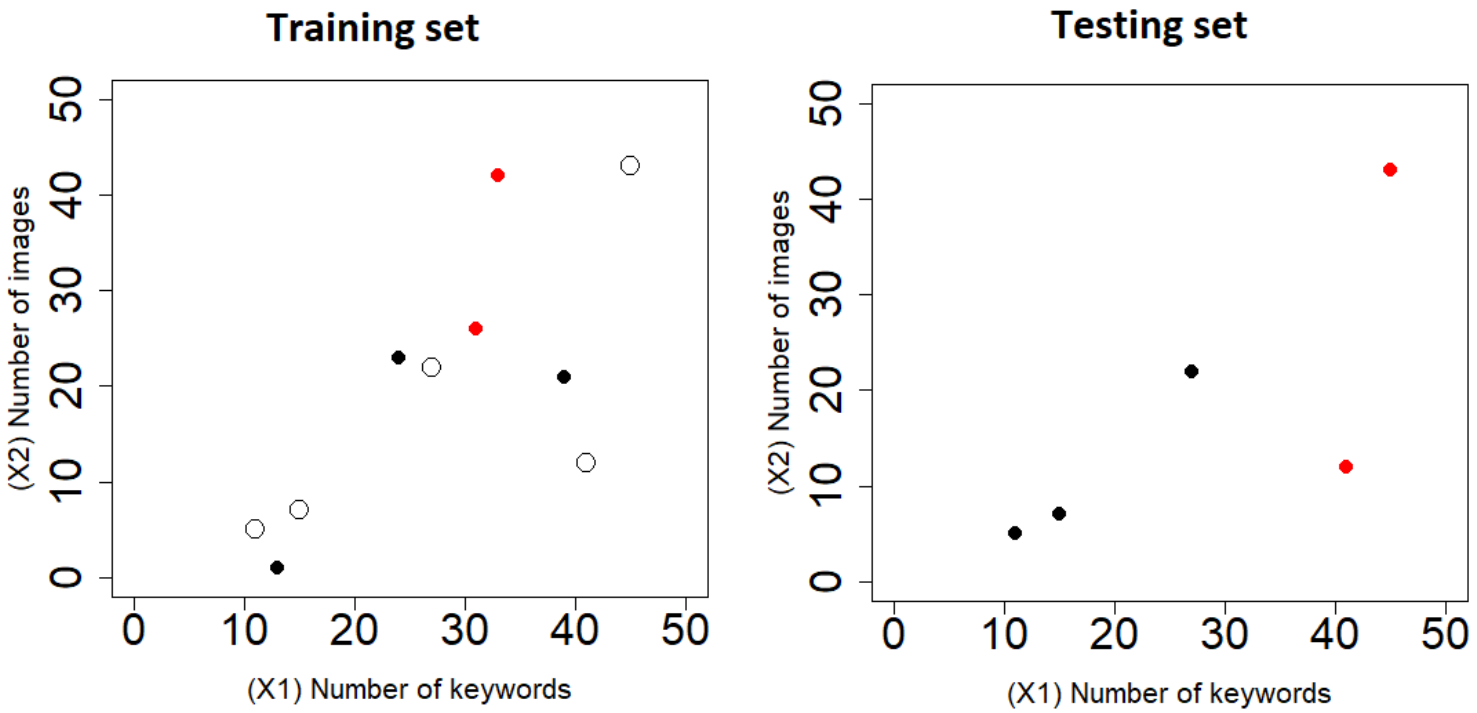
- For 2-fold cross validation, we randomly split the whole data set of 10 points into two data sets of 5 points each.



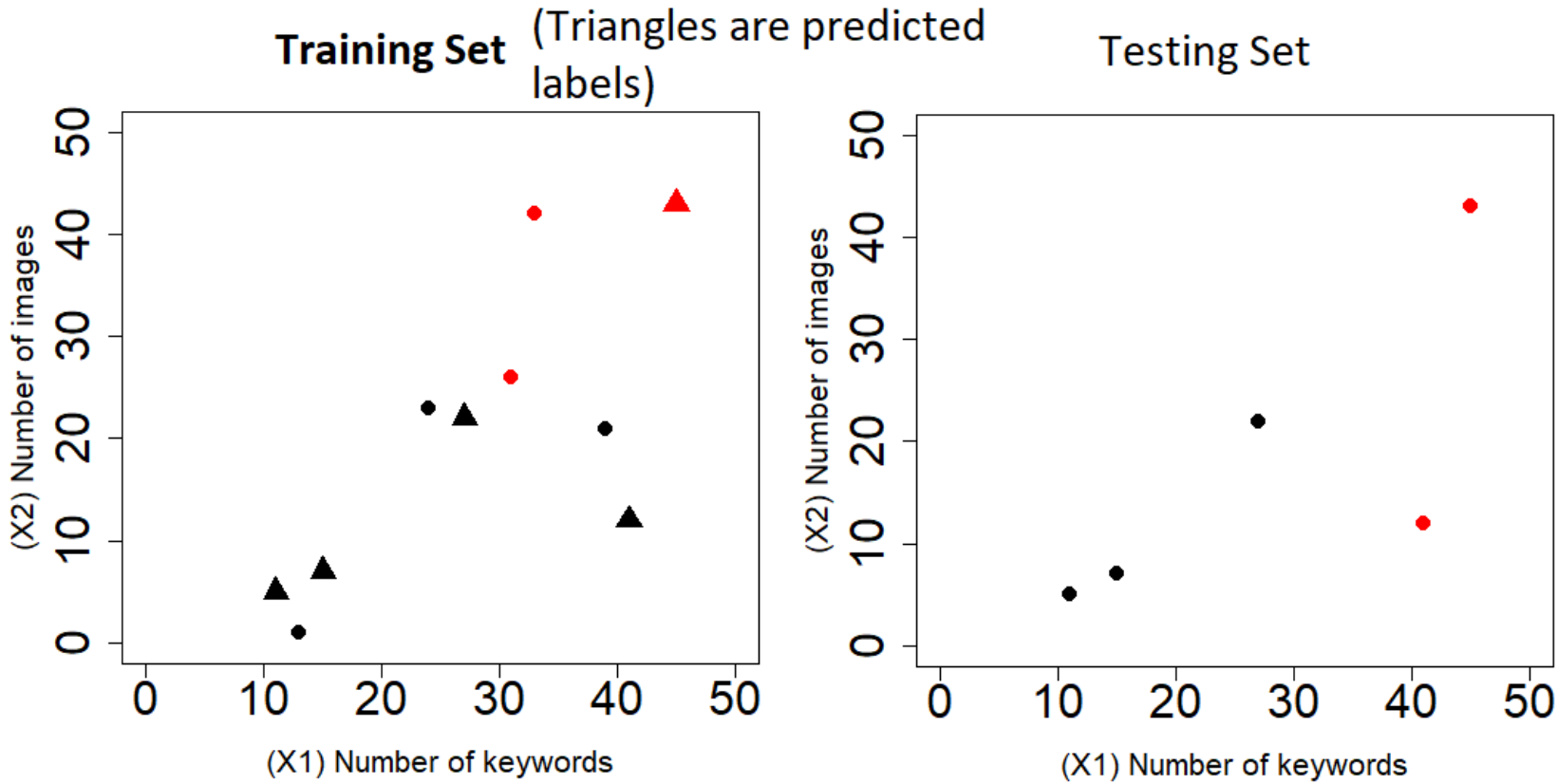


Anti-Spam Techniques: 2-fold CV

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



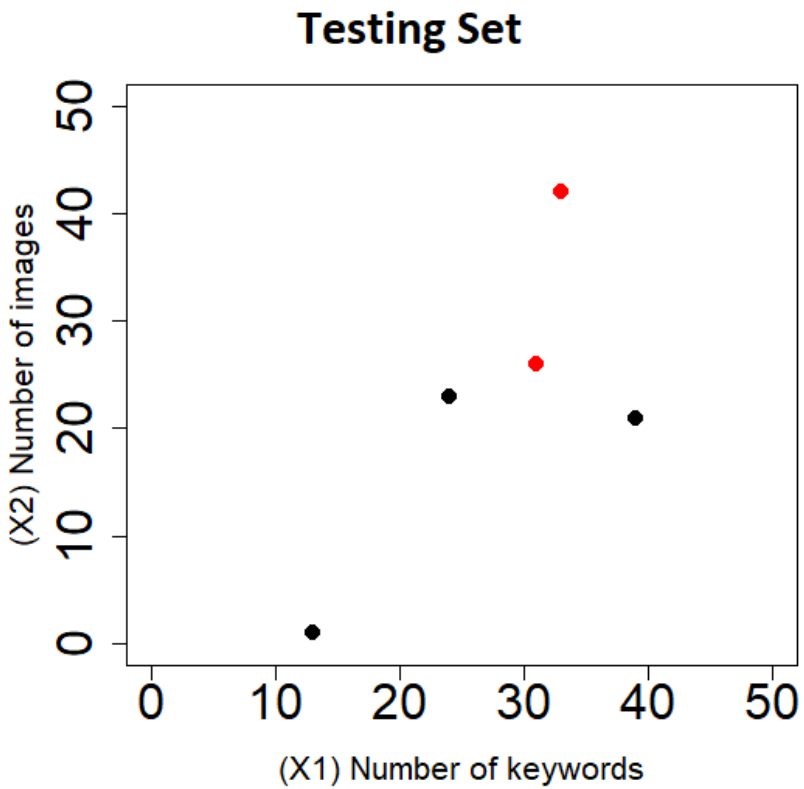
Anti-Spam Techniques: 2-fold CV

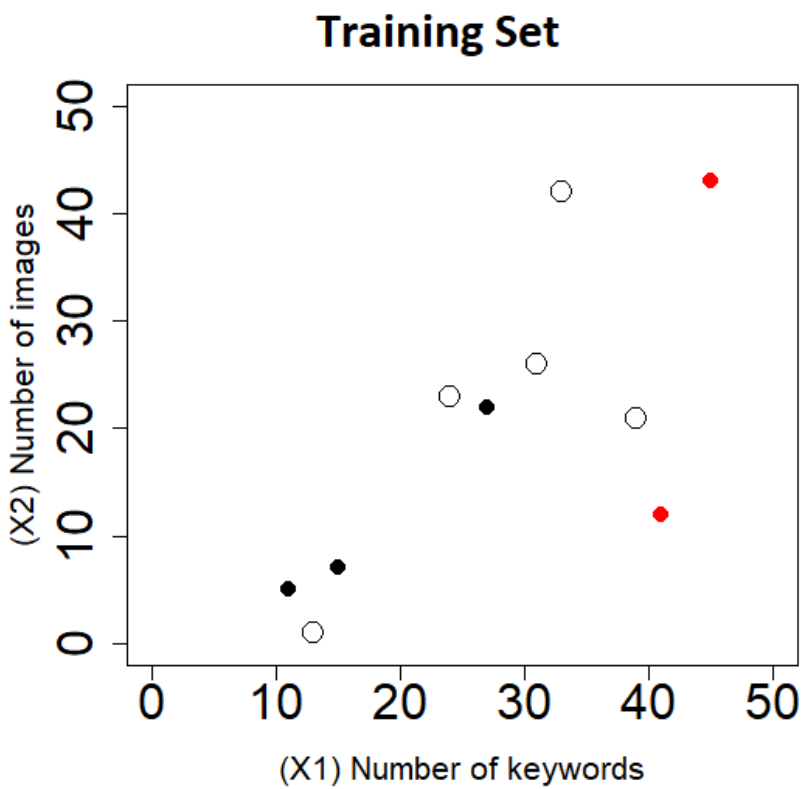


- In this iteration, we estimate the accuracy of the 1-nearest neighbor algorithm to be equal to $\frac{4}{5}$.

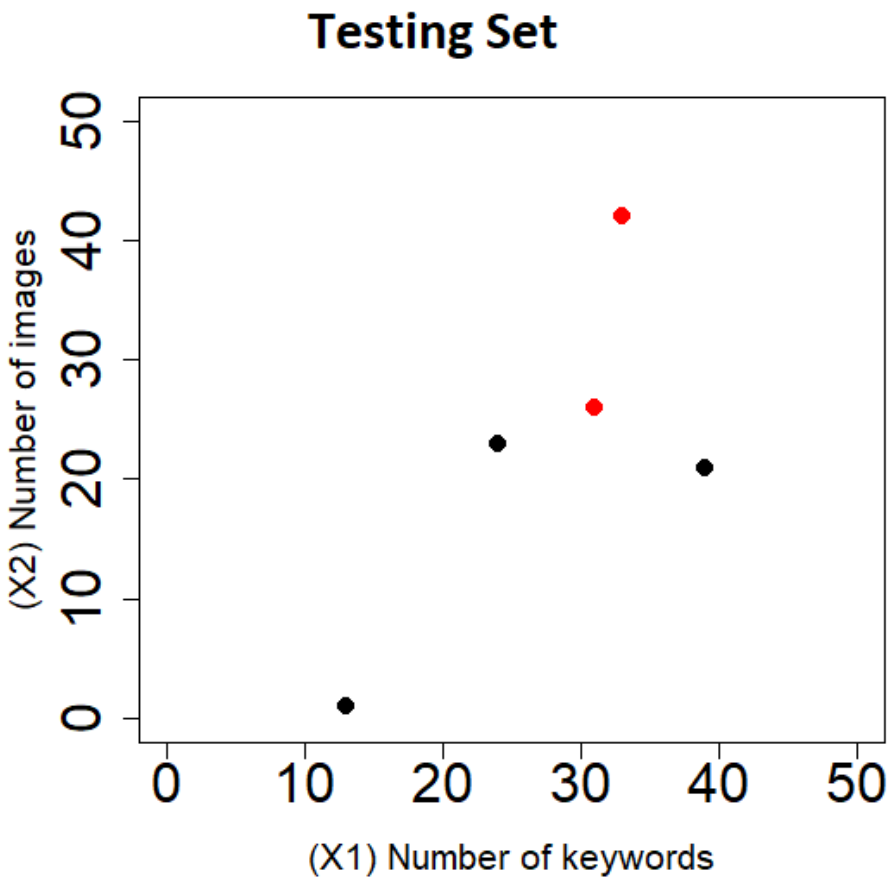
Anti-Spam Techniques: 2-fold CV

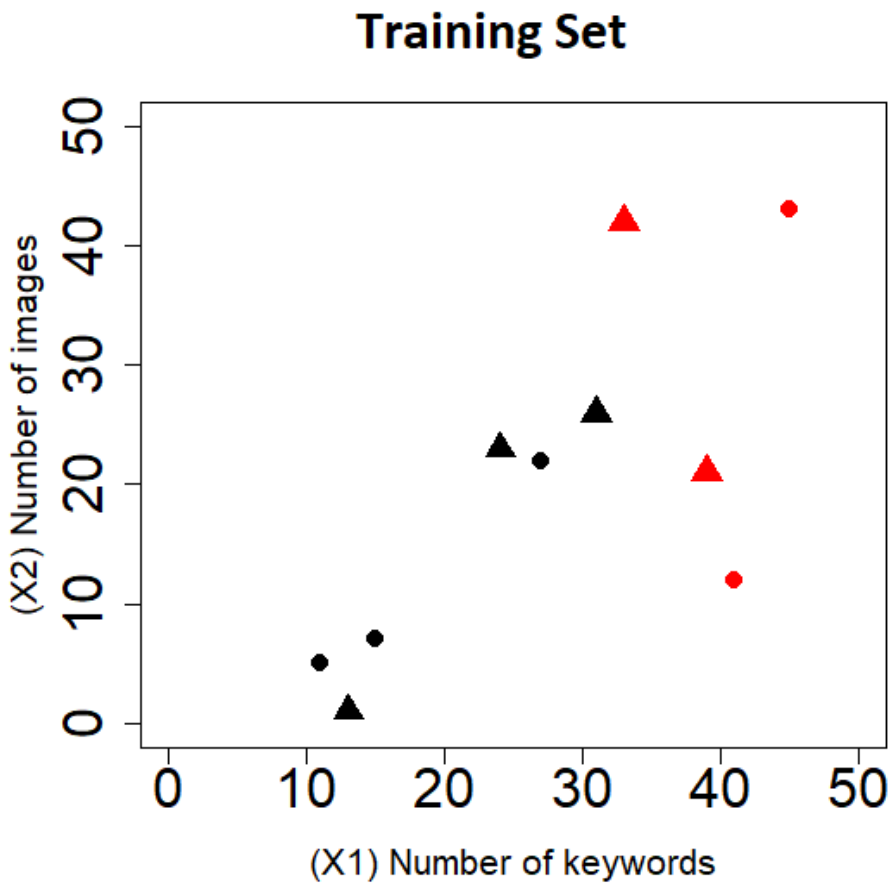
- For the second iteration, we use the second data set as the training set and the first data set as the testing set.





Anti-Spam Techniques: 2-fold CV





Anti-Spam Techniques: 2-fold CV

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

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