COMP9517: Computer Vision

Pattern Recognition

Part 2

Separability

- ***Separable classes***
  - if a discrimination hyperspace exists that separates the feature space such that only objects from one class are in each region, then the recognition task has separable classes

- ***Linearly separable***
  - if the discrimination hyperspaces are hyperplanes, it is linearly separable

Linear Classifier

- If we have training set of N observations:

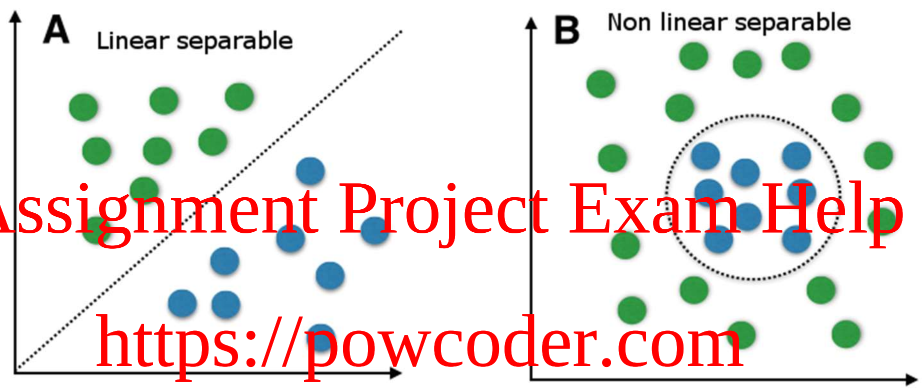
$$\{(x_i, y_i)\}, x_i \in \mathbb{R}^d, y_i \in \{-1,1\}$$

- A binary classification problem can be modeled by $f(x)$ using the data such that:

$$f(x_i) = \begin{cases} > 0 & y_i = +1 \\ < 0 & y_i = -1 \end{cases}$$

- So in this approach $y_i f(x_i) > 0$

Linear Classifier



https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/linear_classification.html

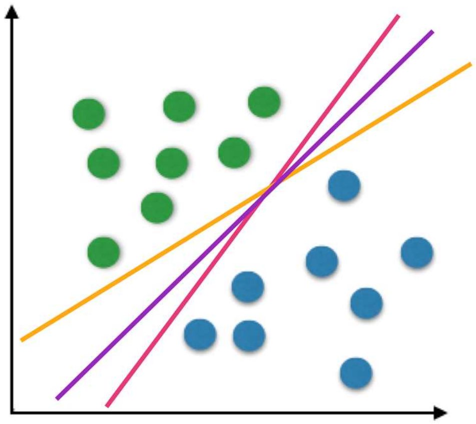
- A linear classifier has the form:

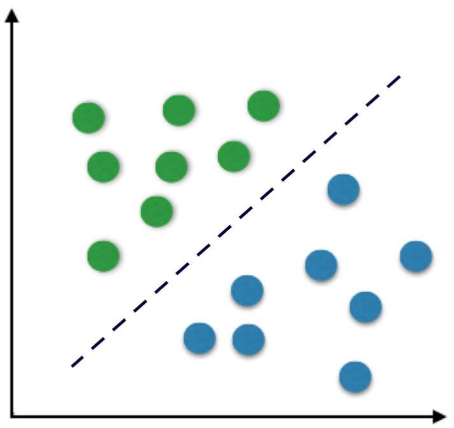
$$f(x) = W^T x + b = w_1 x_1 + w_2 x_2 + \ldots + w_d x_d + b$$

- This is equivalent to a line in 2D, a plane in 3D and a hyperplane in nD
- We use the training data to learn W and b

Linear Classifier

- Which line is the best line?
- For the purpose of generalization a line with large margin is preferred
- A maximum margin solution is the most stable model under perturbation of the input





Support Vector Machines

- We are looking for a W which satisfies:

$$W^T x + b = 0$$

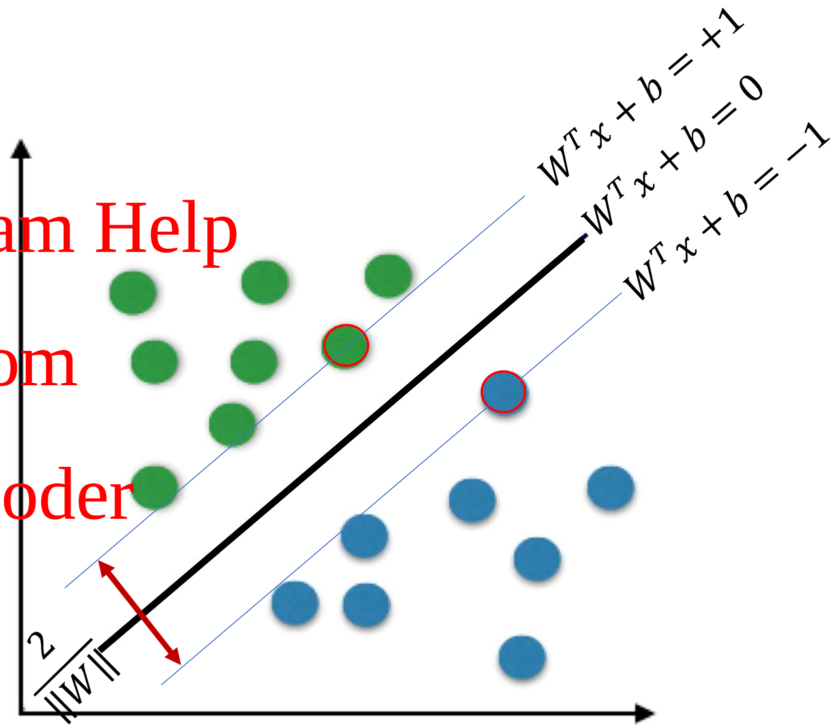
- We know that this is equivalent to:

$$c(W^T x + b) = 0$$

- This means that we have freedom to choose $||W||$
- If we choose $||W||$ such that

$$W^T x_+ + b = +1 \qquad W^T x_- + b = -1$$

- Then the margin is $\frac{2}{||W||}$ which we want to maximize



Support Vector Machines

- Support Vector Machines (SVM) can be formulated as an optimization problem:

$$\arg max_W \left(\frac{2}{\|W\|}\right) \quad subject\ to \quad y_i(W^T x_i + b) \geq 1\ for\ i = 1, \ldots, N$$

- and this is equivalent to:

$$\arg min_W \left(\frac{\|W\|^2}{2}\right) \quad subject\ to \quad y_i(W^T x_i + b) \geq 1\ for\ i = 1, \ldots, N$$

- This is a quadratic optimization problem subject to linear constraints, which has a unique minimum and can be solved with Lagrangian multipliers method

- This is called "hard margin SVM" which does not allow any misclassification, but if classes are not fully separable, we have to use "soft margin SVM" which allows some degree of misclassification

Soft Margin Support Vector Machines

- In hard margin SVM, we assume classes are linearly separable, but what if separability assumptions doesn't hold?

- Introduce "slack" variables ξ_i to allow misclassification of instances

Soft Margin Support Vector Machines

When classes were linearly separable, we had:

$$y_i(w.\mathrm{x}_i + b) \geq 1$$

But if we get some data that violate this slack value:

$$y_i(w.\mathrm{x}_i + b) \geq 1 - \xi_i \;\; and \;\; \xi_i \geq 0$$

So, the total violation:

$$total\ violation = \sum_{i=1}^{m} \xi_i$$

This is a measure of violation of margin and now, we optimize for:

$$\arg min_W \left(\frac{\|W\|^2}{2} + C \sum_{i=1}^{m} \xi_i \right) \quad subject\ to \;\; y_i(w.\mathrm{x}_i + b) \geq 1 - \xi_i \;\; and \;\; \xi_i \geq 0$$

Soft Margin Support Vector Machines

- “Soft margin” SVMs to handle noisy data
- Parameter C bounds influence of any one training instance on decision boundary
- If C goes to infinity you go back to the hard margin SVM (doesn’t tolerate error)
- Still a quadratic optimization problem

Nonlinear Support Vector Machines

- To generate non-linear decision boundaries, we can map the features into new feature space where classes are linearly separable and then apply the SVM there

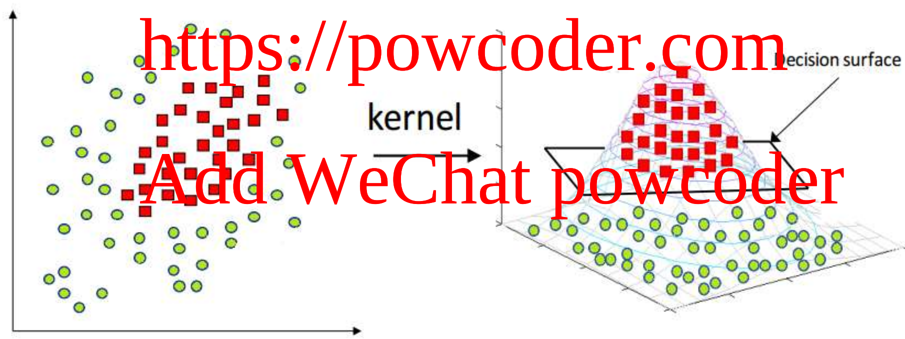


Image source: https://medium.com/analytics-vidhya/how-to-classify-non-linear-data-to-linear-data-bb2df1a6b781

- In SVM, feature mapping into high dimensional space can be done using Kernel trick which helps with the complexity of the optimization problem

Support Vector Machines

- Pros:
  - Very effective in high dimension
  - Effective when the number of features is larger than the training data size
  - Among the best algorithms when classes are separable
  - SVM algorithms can work very well when data is sparse (i.e. many values are 0)
- Cons:
  - For larger dataset, it takes longer time to process
  - Does not perform well for overlapping classes
  - Hyperparameters need to be tuned for sufficient generalization

Multiclass Classification

- If there are more than two classes in our observations, we have to build multiclass classifier
- Some methods may be directly used for multiclass classification
  - K-nearest neighbours
  - Decision trees
  - Bayesian techniques
- For those that cannot be directly applied to multiclass problems, we can transform them to binary classification by building multiple binary classifiers.
- There are two techniques:
  - One vs rest: builds one classifier for one class vs the rest and assign a test sample to the class that has the highest confidence score
  - One vs one: builds one classifier for every pair of classes and assign a test sample to the class that has the highest number of predictions

Evaluation of Error

- **Error rate**
  - error rate of classification system measures how well the system solves the problem it was designed for

- **Reject class**
  - generic class for objects that cannot be placed in any of the known classes

- **Performance**
  - Performance determined by both error and rejections made
  - Classifying all inputs into reject class means system makes no errors, but is useless!

- **Classification error**
  - The classifier makes classification error whenever it classifies input object as class C_i when true class is $C_j,\ i \neq j$, and $C_i \neq C_r$ (the reject class)

Evaluation of Error

- **Empirical error rate**
  - Empirical error rate is the number of errors made on independent test data divided by number of classifications attempted

- **Empirical reject rate**

  - is the number of rejects on independent test data divided by number of classifications attempted

- **Independent test data**
  - are sample objects with true class (labels) known, including objects from the reject class, and that were not used in designing the feature extraction and classification algorithms
- Samples used for training and testing should be representative

False Alarms and False Dismissals

- For two-class problems, the errors have a special meaning and are not symmetric

- For example, in medical diagnosis, when a person has disease versus person does not have disease:

  - If the person does NOT have the disease, but the system incorrectly says she does, then the error is a ***false alarm/false positive***

  - On the other hand, if the person DOES have the disease, but the system incorrectly says she does NOT, then the error is a ***false dismissal or false negative***
- Consequences and costs of the two errors are very different

False Alarms and False Dismissals

- There are bad consequences to both, but false negative is generally more catastrophic

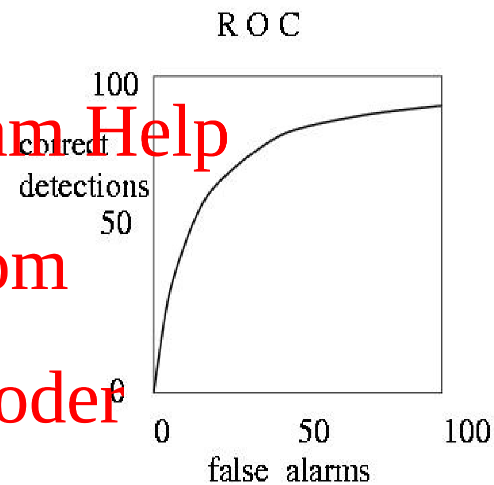
- So, we generally try to bias the system to minimize false negatives, possibly at the cost of increasing the false positives

- The ***Receiver Operator Curve (ROC)*** relates the false alarm rate to correct detection rate

- In order to increase correct detections, we may have to pay the cost of higher number of false alarms.

Receiver Operating Curve ROC

- plots correct detection rate versus false alarm rate
- generally, false alarms go up with attempts to detect higher percentages of known objects
- Area Under (RO)C - AUC



actual input object	decision	error type?
frack	frack	correct alarm (no error)
not a frack	frack	false alarm (error)
frack	not a frack	false dismissal (error)
not a frack	not a frack	correct dismissal (no error)

Confusion Matrix

- Matrix whose entry *(i, j)* records the number of times that an object truly of class *i* was classified as class *j* (*True positive)*
- used to report results of classification experiments
- diagonal entries indicate the successes
- high off-diagonal numbers indicate confusion between classes

class j output by the pattern recognition system

true object class		'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'	'R'
	'0'	97	0	0	0	0	0	1	0	0	1	1
	'1'	0	98	0	0	1	0	0	1	0	0	0
	'2'	0	0	96	1	0	1	0	1	0	0	1
	'3'	0	0	2	95	0	1	0	0	1	0	1
	'4'	0	0	0	0	98	0	0	0	0	2	0
	'5'	0	0	0	1	0	97	0	0	0	0	2
	'6'	1	0	0	0	0	1	98	0	0	0	0
	'7'	0	0	1	0	0	0	0	98	0	0	1
	'8'	0	0	0	1	0	0	1	0	96	1	1
	'9'	1	0	0	0	3	0	0	0	1	95	0

confusion may be unavoidable between some classes for example, between 9's and 4's, or between u's and j's for handprinted characters

Confusion Matrix

- Table of Confusion
  - For binary classification

		Prediction Outcome	
		P	N
Actual Value	P'	**True Positive(TP)**	False Negative (FN)
	N'	False Positive(FP)	**True Negative(TN)**

- Accuracy

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

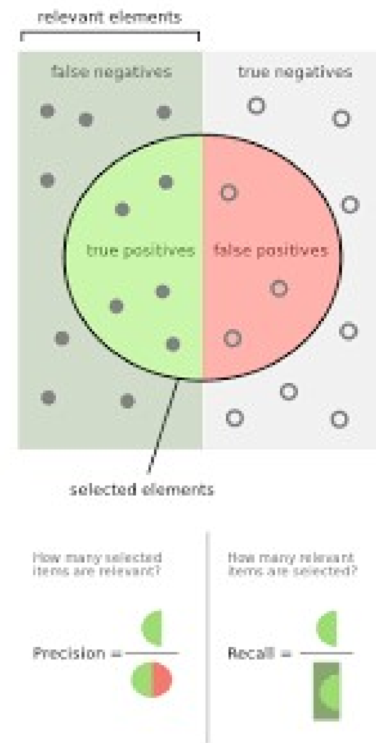
Precision versus Recall

- ***Precision/correctness***
  - is the number of relevant objects classified correctly divided by the total number of relevant objects classified

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

- ***Recall/sensitivity/completeness***
  - is the number of relevant objects classified correctly divided by total number of relevant/correct objects

$$\text{Re}\,call = \frac{TP}{TP + FN}$$



https://en.wikipedia.org/wiki/Precision_and_recall

More Terminology and Metrics

true positive (TP)
  eqv. with hit

true negative (TN)
  eqv. with correct rejection

false positive (FP)
  eqv. with false alarm, Type I error

false negative (FN)
  eqv. with miss, Type II error

sensitivity or true positive rate (TPR)
  eqv. with hit rate, recall
  $TPR = TP / P = TP / (TP + FN)$

false positive rate (FPR)
  eqv. with fall-out
  $FPR = FP / N = FP / (FP + TN)$

accuracy (ACC)
  $ACC = (TP + TN) / (P + N)$

specificity (SPC) or True Negative Rate
  $SPC = TN / N = TN / (FP + TN) = 1 - FPR$

positive predictive value (PPV)
  eqv. with precision
  $PPV = TP / (TP + FP)$

negative predictive value (NPV)
  $NPV = TN / (TN + FN)$

false discovery rate (FDR)
  $FDR = FP / (FP + TP)$

Matthews correlation coefficient (MCC)
  $MCC = (TP * TN - FP * FN)/\sqrt{PNP'N'}$

F1 score
  $F1 = 2TP^2 / (P + P')$

Regression

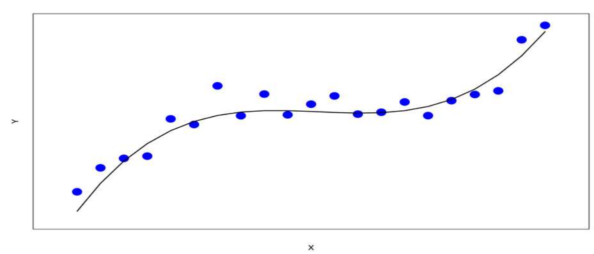
- Suppose that we have a training set of N observations:

$$\{(x_i, y_i)\}, x_i \in \mathbb{R}^d, y_i \in \mathbb{R}$$

- Similar to classification, the regression problem is to estimate $f(x)$ from the training data such that:

$$y_i = f(x_i)$$

- But here the output variable has a continuous value



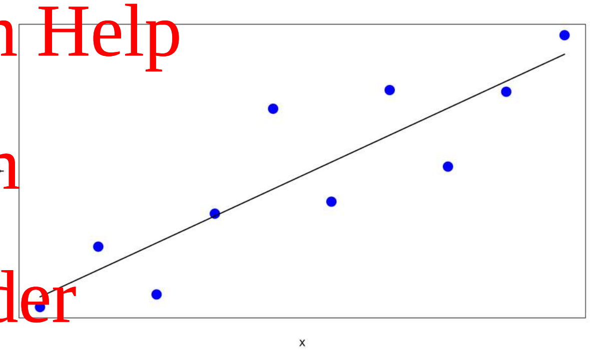
Linear Regression

- In linear regression, we assume there is a linear relationship between the output and features:

$$f(x) = w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_d x_d$$

$$X = [1, x_1, x_2, \ldots, x_d],$$

$$W = [w_0, w_1, \ldots, w_d]^T$$



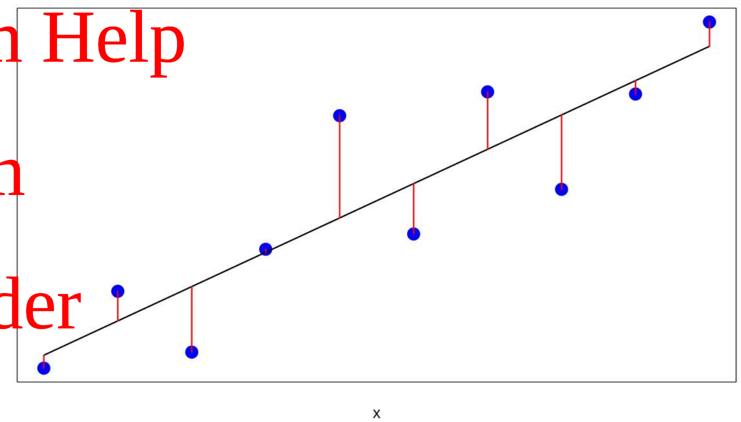
$$f(x) = XW$$

- How to find the best line?
  - The most popular estimation model is "*least squares*"

Least Squares Regression

- The idea is to minimize the residual sum of squares

$$RSS(W) = \sum_{i=1}^{N} (y_i - f(x_i))^2 = (y - XW)^T (y - XW)$$



- How to find the fit?

$$\widehat{W} = \arg min_W (RSS(W))$$

- It turns out that RSS is a quadratic function and we can differentiate RSS with respect to W

Least Squares Regression

$$\frac{\partial RSS}{\partial W} = -2X^T(y - XW)$$

$$\frac{\partial^2 RSS}{\partial W \partial W^T} = 2X^T X$$

- If we assume that X has full rank, then $X^T X$ is positive and it means we have a convex function which has a minimum so:

$$X^T(y - XW) = 0$$

$$\widehat{W} = (X^T X)^{-1} X^T y$$

Linear Regression: Example

- Assume that we have the length and width of some fishes and we want to estimate their weights from this information (features).

- Start with one feature x_1 which is easier for visualization.

$$y = w_0 + w_1 x_1$$

$$X = \begin{bmatrix} 1 & 100 \\ 1 & 102 \\ & \vdots \\ 1 & 97 \end{bmatrix}, \; W = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \;, \quad y = \begin{bmatrix} 5 \\ 4.5 \\ \vdots \\ 4.3 \end{bmatrix}$$

Length (x1)	Width (x2)	Weight (y)
100	40	5
102	35	4.5
92	33	4
83	29	3.9
87	36	3.5
95	30	3.6
87	37	3.4
104	38	4.8
101	34	4.6
97	39	4.3

Linear Regression: Example

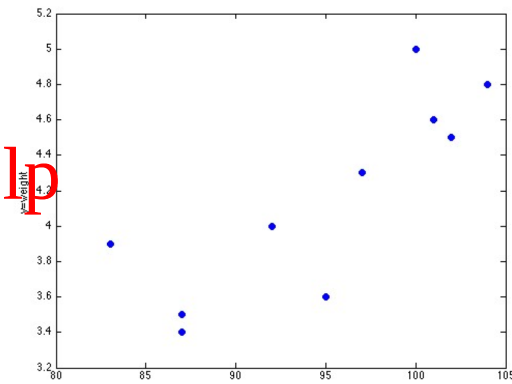
$$W = (XX^T)^{-1}X^Ty = \begin{bmatrix} -1.8 \\ 0.0635 \end{bmatrix}$$

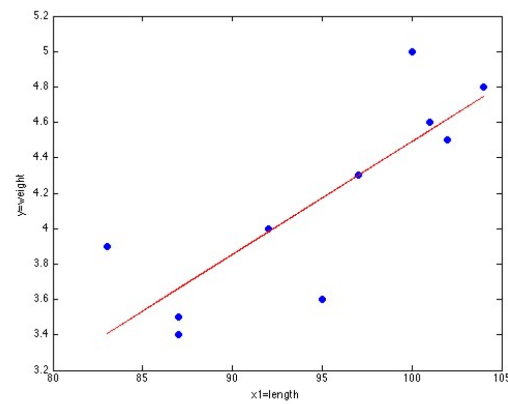
$$RSS(W) = \sum_{i=1}^{N}(y_i - f(x_i))^2 = (y - XW)^T(y - XW) = 0.9438$$

- For two features x_1, x_2 we repeat the same procedure, the only difference is in X:

$$X = \begin{bmatrix} 1 & 100 & 40 \\ 1 & 102 & 35 \\ & \vdots & \\ 1 & 97 & 39 \end{bmatrix}$$

$$W = \begin{bmatrix} -2.125 \\ 0.0591 \\ 0.0194 \end{bmatrix},\ RSS(W) = 0.9077$$





Regression Evaluation Metrics

- **Root Mean Square Error (RMSE)**
  - It represents the standard deviation of the predicted values from the observed values

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

- **Mean Absolute Error (MAE)**
  - It represents the average of the absolute differences between the predicted values and observed values

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

RMSE penalizes big differences between predicted values and observed values more heavily. Smaller values of RMSE and MAE are more desirable.

Regression Evaluation Metrics

- **R-Squared (R^2)**
  - Explains how well the selected feature(s) explain the output variable

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

  - One problem with R-squared is that by adding any extra features, its value increases even if the feature does not actually improve the model

- **Adjuster R-Squared (Adjuster R^2)**

  - Explains how well the selected feature(s) explain the output variable, adjusted for the number of features:

$$R_{adj.}^2 = 1 - \left[\frac{(1 - R^2)(N - 1)}{N - d - 1}\right]$$

where N is the number of samples and d is the number of features

Larger values of R-squares and adjuster R-squared are more desirable

Normalization

- Goal: to change the scale of numeric values to a common scale
- Commonly applied techniques:
  - **Z-score:** re-scales the data(features) such that it will have a standard normal distribution (with mean=0, variance=1). It works well for data which is normally distributed:

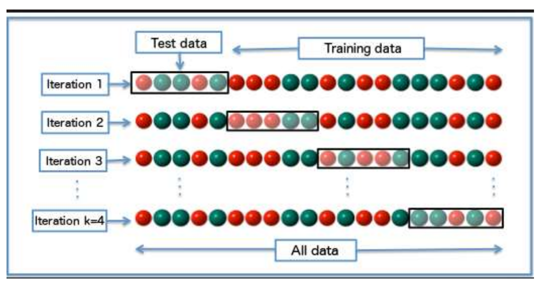
$$\frac{x-\mu}{\sigma}$$

  - **Min-max normalization:** re-scale the range of the data to [0,1]. So the minimum value will be mapped to 0 and the maximum value will be mapped to 1.

$$\frac{x-x_{min}}{x_{max}-x_{min}}$$

Cross Validation

- We want to make sure that when we are done with training our model, it will work well on out-of-sample data. To evaluate the performance of the model in terms of under-fitting/over-fitting/generalizability, we need to test it on unseen data for which we know the ground truth/labels. Cross Validation (CV) is a technique to test the effectiveness of our model.
  - **Train-test split:** in this approach, we randomly split the available data into training and test sets (usually 80:20). We train the model on the training set and then evaluate it on the test set.
  - **K-fold cross validation:** We split the data into K subsets or folds, and at each iteration we keep one fold out for cross validation and use the rest for training. We repeat this procedure K times, until all K folds have been used once as the test set. The performance of the model will be the average of the performance on K test sets.



References and Acknowledgements

- Shapiro and Stockman, Chapter 4
- Duda, Hart and Stork, Chapters 1, 2.1
- Hastie, Tibshirani & Friedman, “the elements of statistical learning”, Chapter 2, chapter 12
- More references
  - Sergios Theodoridis, Konstantinos Koutroumbas, *Pattern Recognition*, 2009
  - Ian H. Witten, Eibe Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2005
- Some diagrams are extracted from the above resources