ECS607U Data Mining

Week 5: Classification

Dr Lin Wang

School of EECS, Queen Mary University of London

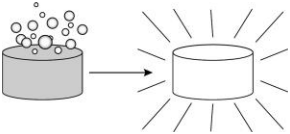
Last Lecture: Data Preprocessing

1. Data Preprocessing: An Overview

2. Data Cleaning

3. Ddimensionality Reduction

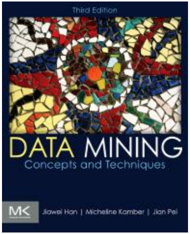
4. Data Normalization

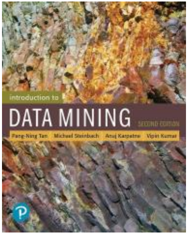


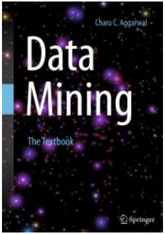
This Lecture's contents

Reading

- Chapters 8 and 10 of J. Han, M. Kamber, J. Pei, "Data Mining: Concepts and Techniques", 3rd edition, Elsevier/Morgan Kaufmann, 2012
- Chapters 3 and 7 of P.–N. Tan, M. Steinbach, A. Karpatne, V. Kumar, "Introduction to Data Mining", 2nd edition, Pearson, 2019
- Chapters 6 and 10 of C. C. Aggarwal, "Data Mining: The Textbook", Springer, 2015







Classification

Classification task

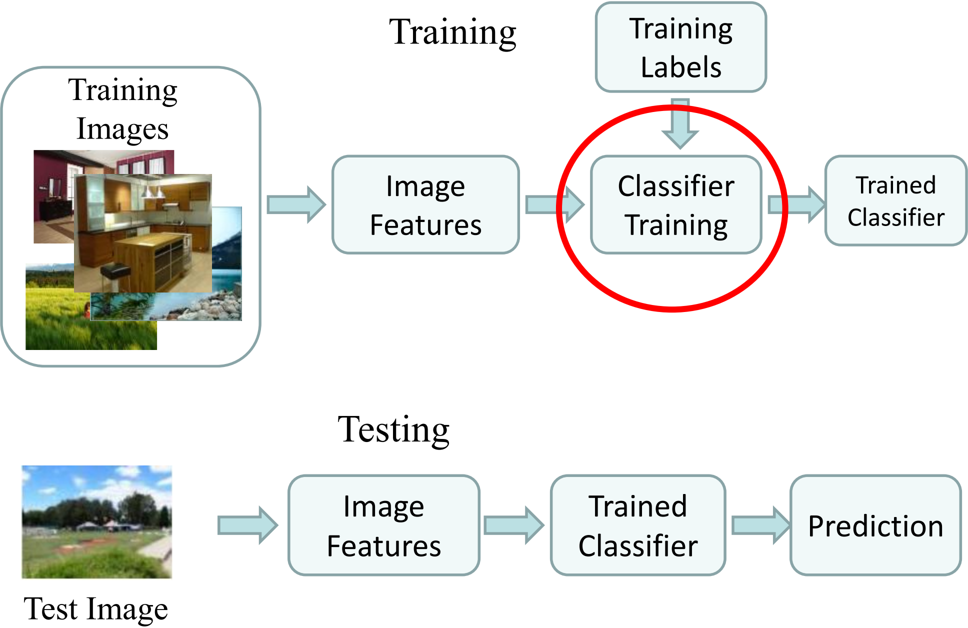
Classification

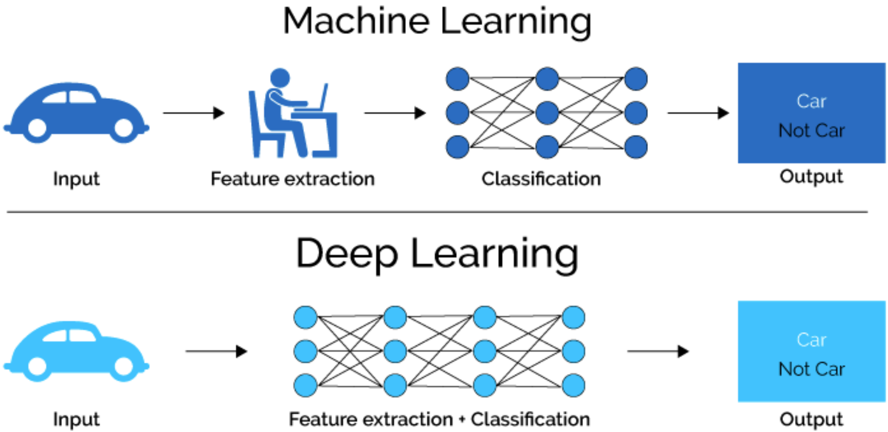
Classification is a form of data analysis that extracts models describing important data classes. Such models, called classifiers, predict categorical class labels.

Classification is a supervised machine learning method where the model tries to predict the correct label of a given input data.

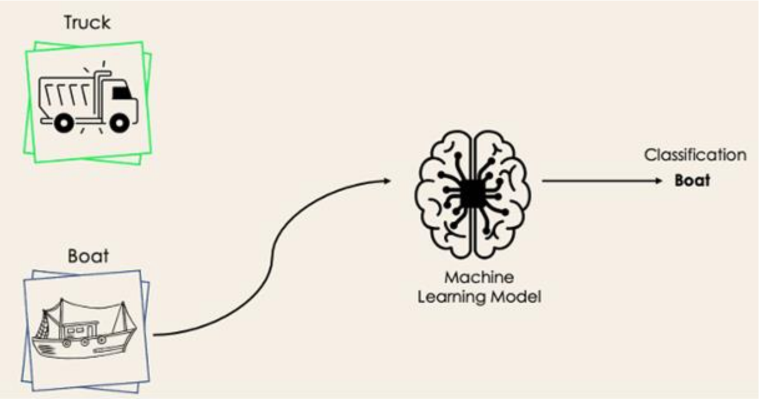
In classification, the model is fully trained using the training data, and then it is evaluated on test data before being used to perform prediction on new unseen data.

Classification example: Image categorization

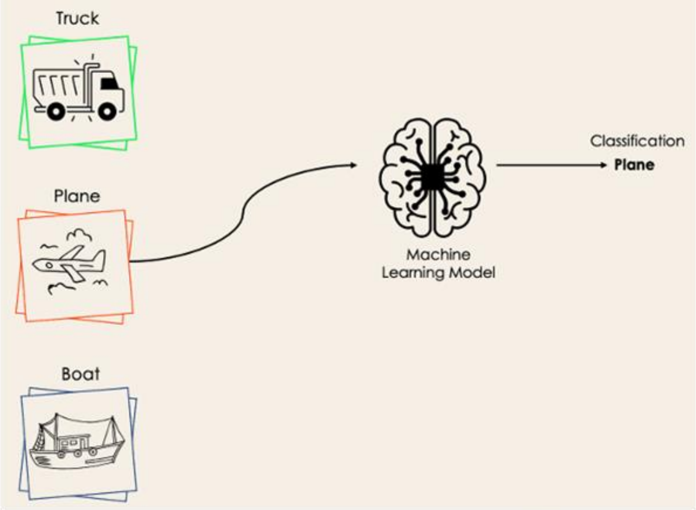



Machine Learning
Car
Not Car
Input
Feature extraction
Classification
Output
Deep Learning
Car
Not Car
Input
Feature extraction + Classification
Output

Binary vs Multiclass



Binary vs Multiclass



Applications

- Medical image analysis
- Natural language processing
- Video surveillance
- Face recognition
- Speech recognition
- Biometric identification
- Credit scoring
- Drug discovery

Classifiers

- SVM
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- K-nearest neighbor
- Etc.

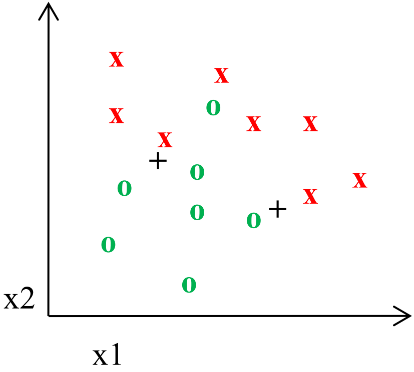
Which is the best one?

Classification

KNN classifier

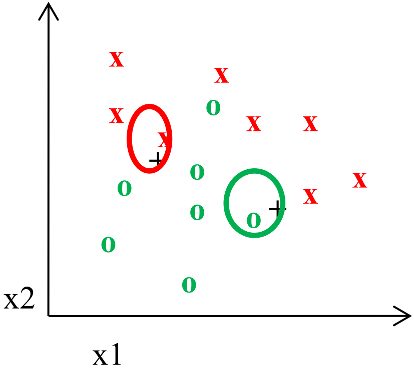
K–nearest neighbor

- Consider a dataset
- $D = (x_1, y_1), \ldots, (x_N, y_N)$



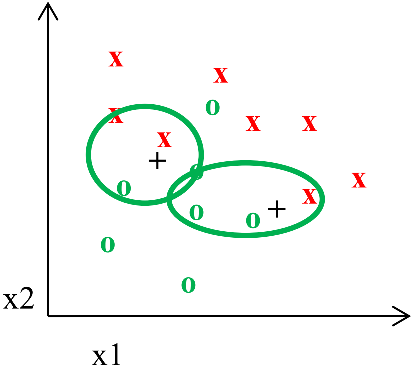
1−nearest neighbor

- Consider a dataset $D = (x_1, y_1), \ldots, (x_N, y_N)$
- A one–nearest neighbour classifier classifies a new observation x as y_i whenever the observation x_i is the closest observation to x in the dataset D



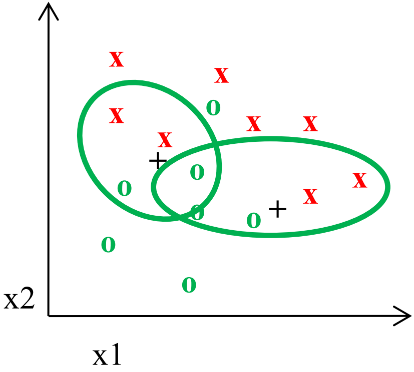
3-nearest neighbor

- A *K*-nearest neighbours classifier classifies a new observation x as *y* if the majority of the *K*-nearest neighbours of x in the dataset *D* belongs to class *y* (tie breaking is arbitrary)



5-nearest neighbor

- A *K*-nearest neighbours classifier classifies a new observation x as *y* if the majority of the *K*-nearest neighbours of x in the dataset *D* belongs to class *y* (tie breaking is arbitrary)



Using K-NN

- Simple, and a good one to try first
- Computational cost dependent on the size of the training dataset
- A hyperparameter (such as K) is any setting required by a learning algorithm to produce a classifier

Distance function

The distance function between observations is an important choice

For example, the Euclidean distance e given by

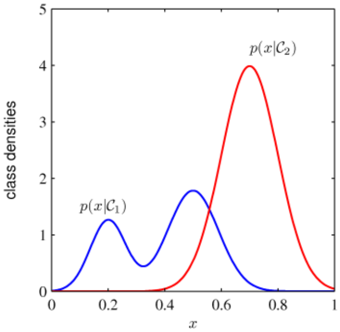
$$e(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'|| = \sqrt{\sum_{j=1}^{d}(x_j - x_j')^2}$$

Classification

Naïve Bayesian classifier

Naïve Bayesian classifier

- Naïve Bayes aims to model the distribution of inputs within a specific class or category.
- The algorithm calculates the probability of a data point belonging to each class and assigns it to the class with the highest probability.
- Based on Bayes Theorem



Bayes Theorem

Likelihood

Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)}$$

Posterior Probability

Prior Probability

From training data

- *P*(*c*|*x*) is the posterior probability of *class* (c, *target*) given *event* (x, *attributes*).
- *P*(c) is the prior probability of *class*.
- *P*(*x*|c) is the likelihood which is the probability of the *event* given *class*.
- *P*(*x*) is the prior probability of the *event*.

Naïve Bayesian classifier steps

1. Convert the given dataset into frequency tables.
2. Compute relevant prior probabilities.
3. Use Bayes theorem to calculate the posterior probability.
4. Make a decision, e.g.

$$p(c_1|x) > p(c_2|x) \Rightarrow x \in c_1$$

$$p(c_1|x) < p(c_2|x) \Rightarrow x \in c_2$$

Naïve Bayesian classifier example

- Suppose we have a dataset of weather conditions and corresponding target variable "Play".
- Task: Based this dataset we use Naïve Bayesian classifier to decide that whether we should play or not on a particular day according to the weather conditions.
- For instance, when the weather is Sunny, should we play or not?

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Naïve Bayesian classifier example

- Suppose we have a dataset of weather conditions and corresponding target variable "Play".
- Based this dataset we Naïve Bayesian classifier to decide that whether we should play or not on a particular day according to the weather conditions.
- For instance, when the weather is Sunny, should we play or not?

$$P(\text{Yes}|\text{Sunny}) = \frac{P(\text{Sunny}|\text{Yes})P(\text{Yes})}{P(\text{Sunny})}$$

$$P(\text{No}|\text{Sunny}) = \frac{P(\text{Sunny}|\text{No})P(\text{No})}{P(\text{Sunny})}$$

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Naïve Bayesian classifier example

Step 1: Convert the data set into a frequency table

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Naïve Bayesian classifier example

Step 2: Compute relevant prior probabilities

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

$$P(Yes|Sunny) = \frac{P(Sunny|Yes)P(Yes)}{P(Sunny)}$$

$$P(No|Sunny) = \frac{P(Sunny|No)P(No)}{P(Sunny)}$$

$\Rightarrow$

P(Sunny) = 5/14 = 0.36
P(Yes) = 9/14 = 0.64
P(No) = 5/14 = 0.36
P(Sunny|Yes) = 3/9 = 0.333
P(Sunny|No) = 2/5=0.4

Naïve Bayesian classifier example

Step 3: Use Naive Bayesian equation to calculate the posterior probability

P(Sunny) = 5/14 = 0.36
P(Yes) = 9/14 = 0.64
P(No) = 5/14 = 0.36
P(Sunny|Yes) = 3/9 = 0.333
P(Sunny|No) = 2/5=0.4

P(Yes|Sunny) = P(Sunny | Yes) * P(Yes) / P (Sunny) = 0.6
P(No|Sunny) = P(Sunny | No) * P(No) / P (Sunny) = 0.4

Naïve Bayesian classifier example

Step 4: Make a decision

P(Yes|Sunny) > P(No|Sunny)

Naïve Bayesian classifier example

The key is to compute the prior probabilities from the training data

Likelihood

Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)}$$

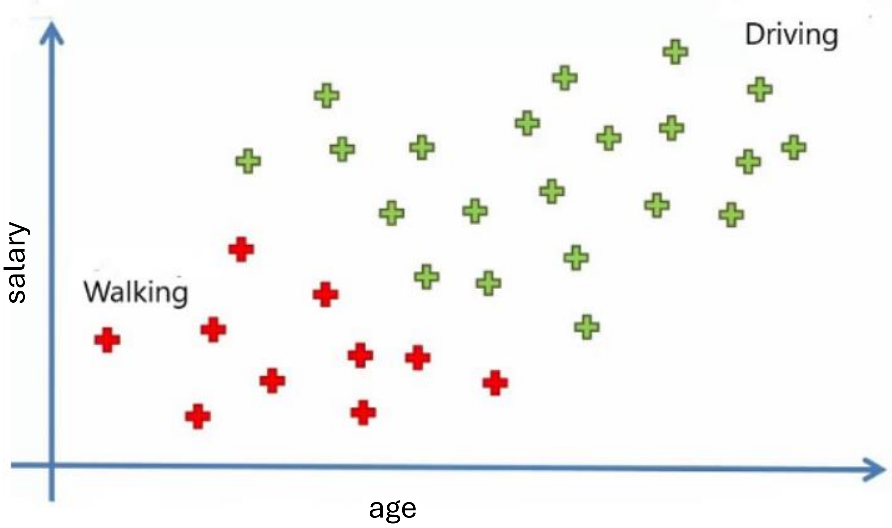
From training data

Posterior Probability

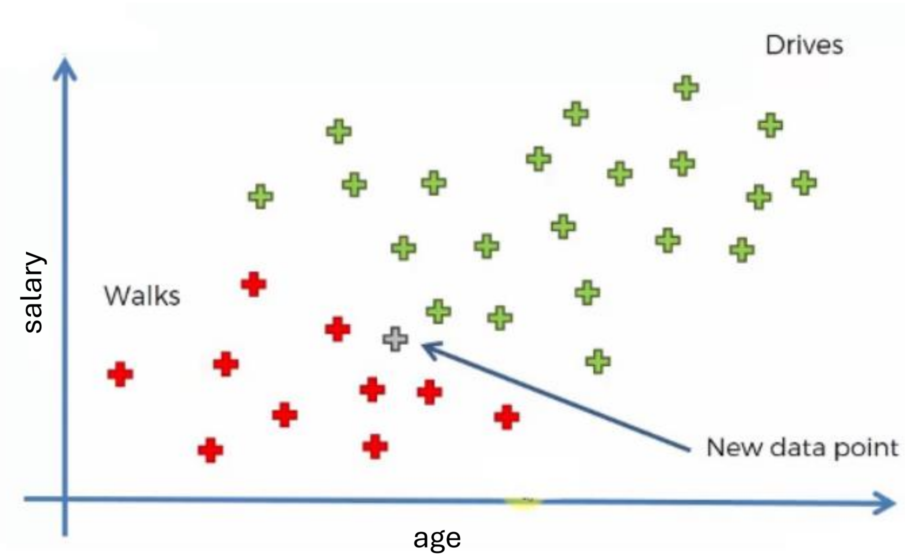
Prior Probability

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(P(c)$$

Naïve Bayesian classifier example



Naïve Bayesian classifier example



Naïve Bayesian classifier example

- The posterior probability of walking for the new data point is

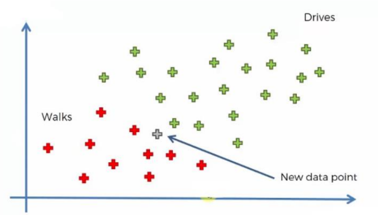
Posterior Probability — Likelihood — Prior Probability

$$P(Walks|X) = \frac{P(X|Walks) * P(Walks)}{P(X)}$$

- The posterior probability of driving for the new data point is

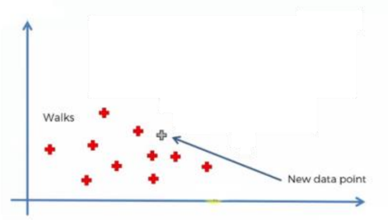
Posterior Probability — Likelihood — Prior Probability

$$P(Drives|X) = \frac{P(X|Drives) * P(Drives)}{P(X)}$$



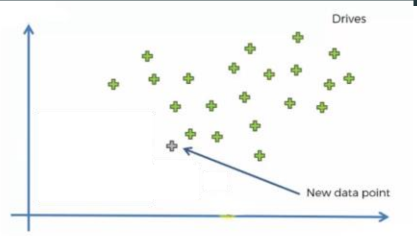
Naïve Bayesian classifier example

salary



age

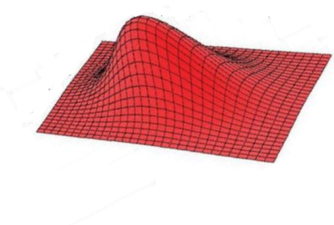
salary

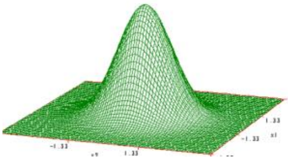


age

$$P(\boldsymbol{x}|walk) = G(\boldsymbol{x}, \boldsymbol{\mu_1}, \boldsymbol{\sigma_1})$$

$$P(\boldsymbol{x}|drive) = G(\boldsymbol{x}, \boldsymbol{\mu_2}, \boldsymbol{\sigma_2})$$





Naïve Bayesian classifier: discussion

Pros

- Easy and fast
- Work well with small amount of data
- Can incorporate prior knowledge of the attributes

Cons

- Independent assumption of attributes
- Need to estimate distribution probability density function of attributes

Classification

Model evaluation

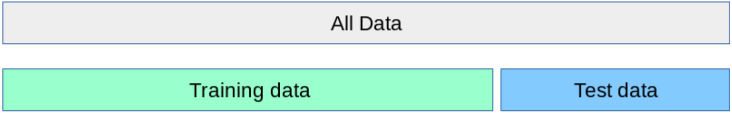
Generalisation

- A classifier (`e.g.` one-nearest neighbour) will always predict the correct class for any observation that *already exists* in the dataset D
- However, it is usually necessary to estimate generalisation: the capacity of a given classifier to assign unseen observations to correct classes
- How to estimate generalisation given a fixed D?

Training and test datasets

- A simple approach is to split an original dataset into a training dataset D and a test dataset D^0
- The test dataset usually contains at least 20% of the original pairs, which are chosen randomly

- A learning algorithm is given only the training dataset D in order to produce a classifier f
- The classifier f may be evaluated on the unseen observations in the test dataset D^0



Performance metric: accuracy

- The accuracy of a classifier f on a dataset $D = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ is the fraction of observations in D that are classified correctly by f:

$$\frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left[f(\mathbf{x}_i) = y_i\right]$$

where $\mathbb{I}[e] = 1$ if e is true, and $\mathbb{I}[e] = 0$ otherwise

- The accuracy is always between 0% and 100%
- Note the effect of class imbalance: if 99% of the observations in D belong to class y, a classifier that predicts y for every observation has 99% accuracy

Performance metric: Confusion matrix

- Confusion matrix is a table that is used in classification problems to assess where errors in the model were made.

- Binary classification: 2x2 confusion matrix (e.g. disease diagnosis)
  - True Positive: You predicted positive, and it’s true.
  - True Negative: You predicted negative, and it’s true.
  - False Positive: (Type 1 Error): You predicted positive, and it’s false.
  - False Negative: (Type 2 Error): You predicted negative, and it’s false.

		Actual class	
		P	**N**
Predicted class	**P**	**TP**	FP
	N	FN	**TN**

Performance metric: F1 score

- Binary classification: 2x2 confusion matrix (e.g. disease diagnosis)
  - Precision: proportion of positive cases that were correctly identified.

    Precision = (TP)/(TP+FN)
  - Recall: proportion of actual positive cases which are correctly identified.

    Recall = TP/(FP+TP)
  - Accuracy: proportion of cases which are correctly identified

    Accuracy = (TP+TN)/(TP+FN+FP+TN)
  - F1 score: harmonic mean of precision and recall

    F1 = 2(Precision*Recall)/(Precision+Recall)

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Performance metric: F1 score

- Multiclass classification: CxC confusion matrix **M** (C is the number of classes)
- A one–vs–all technique is used to compute the individual scores for every class in the dataset.
- The macro F1 score is calculated as the average across all classes

$$\text{recall}_i = \frac{M_{ii}}{\sum_{j=1}^{C} M_{ij}}, \quad \text{precision}_j = \frac{M_{jj}}{\sum_{i=1}^{C} M_{ij}},$$

$$F1 = \frac{1}{C} \sum_{i=1}^{C} \frac{2 \cdot \text{recall}_i \cdot \text{precision}_i}{\text{recall}_i + \text{precision}_i}.$$

		PREDICTED classification			
	Classes	a	b	c	d
ACTUAL classification	a	TN	FP	TN	TN
	b	FN	TP	FN	FN
	c	TN	FP	TN	TN
	d	TN	FP	TN	TN

Performance metric: Confusion matrix

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

- Precision: proportion of positive cases that were correctly identified.

  $\text{Precision} = (TP)/(TP+FN)$

- Recall: proportion of actual positive cases which are correctly identified.

  $\text{Recall} = TP/(FP+TP)$

- Accuracy: proportion of cases which are correctly identified

  $\text{Accuracy} = (TP+TN)/(TP+FN+FP+TN)$

- F1 score: harmonic mean of precision and recall

  $F1 = 2(\text{Precision}*\text{Recall})/(\text{Precision}+\text{Recall})$

- Marco-F1 score: average F1 score of two classes

		Actual	
		Class P	Class N
Prediction	Class P	80	40
	Class N	20	60

Performance metric: Confusion matrix

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

- Precision: proportion of positive cases that were correctly identified.

  Precision = (TP)/(TP+FN)
- Recall: proportion of actual positive cases which are correctly identified.

  Recall = TP/(FP+TP)
- Accuracy: proportion of cases which are correctly identified

  Accuracy = (TP+TN)/(TP+FN+FP+TN)
- F1 score: harmonic mean of precision and recall

  F1 = 2(Precision*Recall)/(Precision+Recall)

		Groundtruth	
		Class P	Class N
Prediction	Class P	800	4
	Class N	200	6

Overfitting and underfitting

- Suppose the training and test datasets are large and representative
- A classifier overfits if its performance is much lower on the test set than on the training set
- A classifier underfits if its performance is low both on the test set and on the training set
- Different learning algorithms produce classifiers with potentially different biases, which ultimately leads to different performances

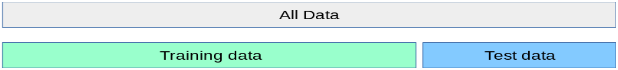
Classification

Model selection

Model selection

- Suppose $f_1, \ldots, f_T$ are classifiers trained on D
- Model selection is the process of choosing between these classifiers based on their performance
- This process applies whether the classifiers are the result of different learning algorithms or different hyperparameters for the same algorithm
- Using nearest neighbors classifier as an example, you may suppose that f_K is a K-nearest neighbours classifier trained on D, we want to find the best K value

Model selection

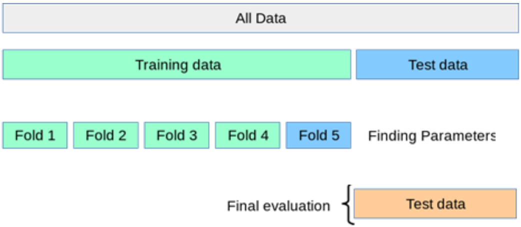


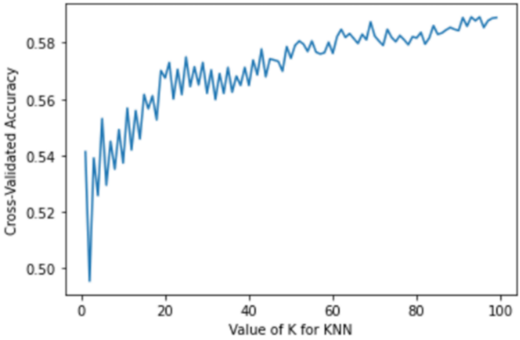
- Training data
  - A classifier f_k should not be selected based on its performance on the training set D
  - Performance on the training set is not a reliable estimate of generalisation for f_k
- Testing data
  - A classifier f_k should not be selected based on its performance on the test set D^0
  - Performance on the test set could have been obtained by sacrificing performance on other datasets, and there would be no data left to enable a reliable estimate of generalisation for f_k
  - If model selection were seen as a learning algorithm that receives a dataset D and produces a classifier f_k, this approach would have used the test set D^0 during training

Training, validation, and test datasets

- A fixed dataset may be randomly split into:
  - Training dataset: enables training different classifiers
  - Validation dataset: enables choosing the best classifier based on its performance
  - Test dataset: enables estimating reliably the performance of the best classifier
- Improving model selection after using the test set defeats its purpose: use the test set as a final step
- In most cases, after the best combination of learning algorithm and hyperparameters is found, all three datasets may be used to train a final classifier

cross-validation

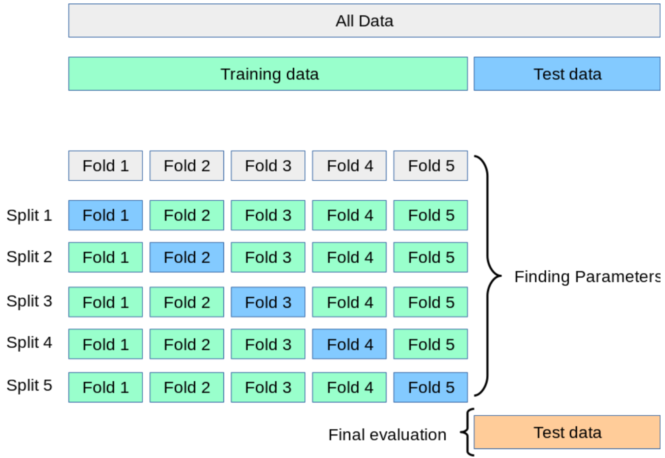




K-fold cross-validation

- A representative validation set is crucial to select the best classifier
- However, there is a trade-off between the size of the validation set and the size of the training set
- In K-fold cross-validation, a fixed dataset is randomly split into a training dataset D and a test set D^0
- The training dataset is further randomly split into K folds (subsets) of equal size
- Each fold is used as a validation set when the remaining folds are used as an effective training set
- The classifier that achieves maximum average performance across folds (validation sets) is selected

5-fold cross-validation



Summary

- Classification
  - Classification task
  - K -nearest neighbours classifier
  - Naïve Bayesian classifier
  - Model evaluation
  - Model selection

Questions?

also please use the forum on QM+