CSE2510 Machine Learning Review

Yuhang Tian

November 1, 2020

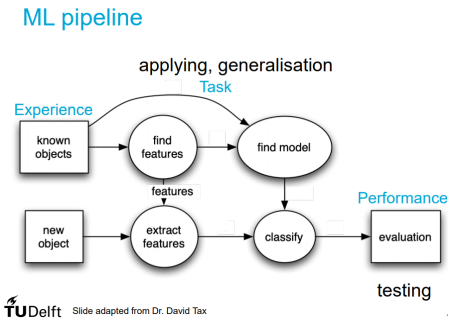
1 Week 1

1.1 Lecture 1

(1) What is machine learning?
learning (training on data) and generalisation (learn some data and say something about a new situation). Machine learning is about probabilities.
The learning of patterns or regularities in data by computer algorithms in order for these computer algorithms to carry out a specific task without using explicit instructions, but instead relying on these patterns and inference

(2) Supervised Learning vs. Unsupervised Learning?
Supervised Learning: The algorithm learns on a labeled dataset Unsupervised Learning: The algorithm learns on a unlabeled dataset

(3) ML pipeline?



(i) Train the ML algorithm using a dataset of examples with features for a specific task

(ii) Test the generalisability of the ML algorithm on an unseen testset

(iii) Quantify performance using an accurate and suitable measurement

(4) Training set vs. Test set?
Traning set

- To train the parameters
- To learn to carry out the task

Test set

- Independent from the training set
- To test the generalisability
- The probability distribution should be the same as the training set

(5) Features and Labels?
Learning and predicting is based on counting the frequency of occurrence of objects.

(6) Learning and Goal of Training?
Learn a function that can predict a label y for a new x with as little error as possible.
Training = Learning the optimal model parameters.

(7) No free lunch theorem?
Averaged over all possible data-generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points
In some sense, no machine learning algorithm is universally any better than any other

(8) Goal of ML?
NOT to seek a universal learning algorithm or the absolute best learning algorithm
BUT what kinds of machine learning algorithms perform well on the data drawn from the kinds of data-generating distributions we care about
→ Design ML algorithm for a specific task

1.2 Lecture 2

(1) Train/test Performance?
Train on traning data → training error
Evaluate the model → test error

(2) Joint probability?

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N} \tag{1}$$

(3) Marginal probability?

$$p(X = x_i) = \sum_{j=1}^{n} p(X = x_i, Y = y_j) \tag{2}$$

(4) Conditional probability?

$$p(Y = y_j|X = x_i) = \frac{n_{ij}}{c_i} \tag{3}$$

(5) Bayes' Rule?

$$p(Y|X) = \frac{p(X|Y) \times p(Y)}{p(X)} \tag{4}$$

(6) Classification - Trainning and Testing?
Training: Estimate the probability distribution over a set of classes
Testing: Estimate the probability that sample belongs to a class

(7) How to calculate the posterior probabilities?

(i) Estimate the class probabilities $p(x|C_1),\ p(x|C_2)$

(ii) Muliply with the class priors $p(x|C_1)p(C_1),\ p(x|C_2)p(C_2)$

(iii) Compute the class posterior probabilities (divided by p(x)) $p(C_1|x),\ p(C_2|x)$

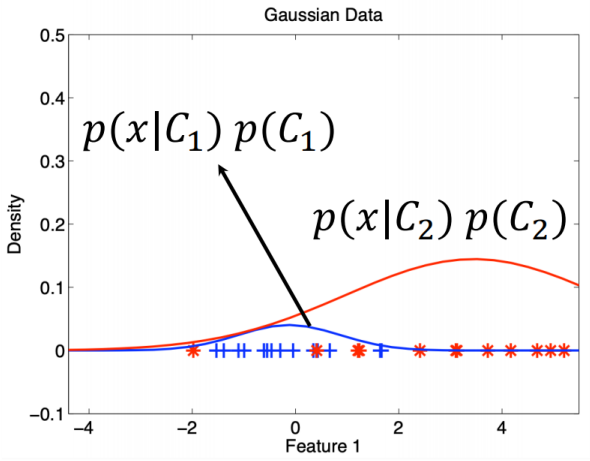
(iv) Assign objects to the class with the highest posterior probability

(8) Decision Theory?

There are several ways to describe a classifier:

- if $p(C_1|\mathrm{x}) > p\ (C_2|\mathrm{x})$ then assign to C_1 otherwise C_2
- if $p(C_1|\mathrm{x}) - p(C_2|\mathrm{x}) > 0$ then assign to C_1
- or $\frac{p(C_1|\mathrm{x})}{p(C_2|\mathrm{x})} > 1$ then assign to C_1
- or ...

(9) Missing decision boundary?



A class can be too small (class prior is low) or too dispersed, that no objects are assigned to that class

(10) Bayes Error?

Even if we know the true distribution, errors predicting y from x will occur because the posteriors $p(y|x)$ are often not exactly 0 or 1

Bayes Error is the lowest possible prediction error, which is irreducible.

The Bayes' error does not depend on the classification rule that you apply, but on the distribution of the data

In general you cannot compute the Bayes' error:

- You don't know the true class conditional probabilities
- The (high) dimensional integrals are very complicated

(11) Misclassification Costs?

We want to make as few errors as possible with the assignment of x to class C

Sometimes: misclassification of class A to class B is much more dangerous than misclassification of class B to class A

(12) Generative vs. Discriminative models

Generative models:

- Assume some functional form for $P(Y)$, $P(X|Y)$
- Estimate parameters of $P(X|Y)$, $P(Y)$ directly from training data
- Use Bayes rule to calculate $P(Y|X)$

Examples:

- Naïve Bayes
- Bayesian networks

- Markov random fields
- Hidden Markov Models (HMM)

Discriminative models:

- Assume some functional form for $P(Y|X)$
- Estimate parameters of $P(Y|X)$ directly from training data

Examples:

- Logistic regression
- Scalar Vector Machine
- Traditional neural networks
- Nearest neighbour
- Conditional Random Fields (CRF)s

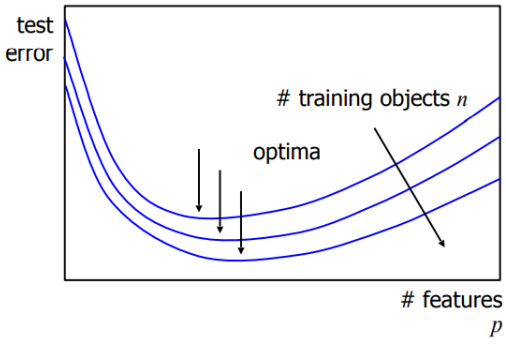
2 Week 2

2.1 Lecture 3

(1) Histogram-based Density Estimation?
Relatively simple approach : approximate density by histogram

$$\hat{p}(x) = \frac{dP(x)}{dx} = \frac{fractionofobjects}{volume} \tag{5}$$

(2) Curse of Dimensionality?
Intuitively, using more features [e.g. width, height, color etc.] should give us more information about the outcome to predict
But never know densities, so have to estimate
Number of parameters [e.g. histogram bins] to estimate increases with the number of features
To estimate these well, you need more objects
Consequence : there is an optimal number of features to use



(3) Parametric vs. Non-parametric?
Parametric: Assumptions can greatly simplify the learning process, but can also limit what can be learned. Algorithms that simplify the function to a known form are called parametric machine learning algorithms.
Examples:

- Logistic Regression
- Linear Discriminant Analysis
- Perceptron
- Naive Bayes
- Simple Neural Networks

Advantages:

- Simpler: These methods are easier to understand and interpret results.
- Speed: Parametric models are very fast to learn from data.
- Less Data: They do not require as much training data and can work well even if the fit to the data is not perfect.

Disadvantages:

- Constrained: By choosing a functional form these methods are highly
- constrained to the specified form.
- Limited Complexity: The methods are more suited to simpler problems.
- Poor Fit: In practice the methods are unlikely to match the underlying mapping function.

Non-parametric: Algorithms that do not make strong assumptions about the form of the mapping function are called nonparametric machine learning algorithms. By not making assumptions, they are free to learn any functional form from the training data.
Examples:

- k-Nearest Neighbors
- Decision Trees like CART and C4.5
- Support Vector Machines

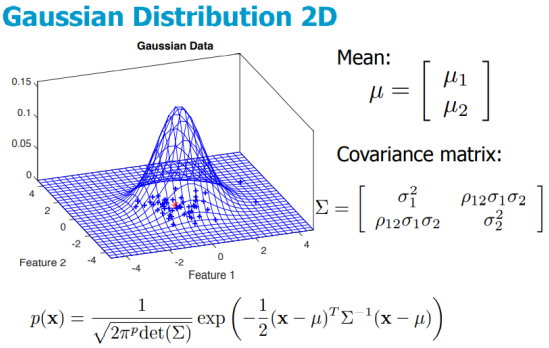
Advantages:

- Flexibility: Capable of fitting a large number of functional forms.
- Power: No assumptions (or weak assumptions) about the underlying function.
- Performance: Can result in higher performance models for prediction.

Disadvantages:

- More data: Require a lot more training data to estimate the mapping function.
- Slower: A lot slower to train as they often have far more parameters to train.
- Overfitting: More of a risk to overfit the training data and it is harder to explain why specific predictions are made.

(4) 2D Gaussian Distribution?



(5) Maximum Likelihood Estimates?

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{6}$$

$$\hat{\sigma} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})(x_i - \hat{\mu})^T \tag{7}$$

2.2 Lecture 4

(1) Inverse of the covariance matrix?
If there is no inverse of the covariance matrix - number of objects is insufficient to find the inverse, it is not possible to obtain the distribution. Therefore, it should degenerate Gaussian distribution.

(2) Mixture Models?
Model: mixture of 2 Gaussians

(3) Quadratic vs. LDA vs. Nearest-Mean?

- Separate mean and covariance matrix per class gives the quadratic classifier
- Separate mean, equal covariance matrix per class gives the linear classifier
- Separate mean, identity covariance matrix per class gives the nearest mean classifier

3 Week 3

3.1 Lecture 5

(1) Non-parametric?
no knowledge on the distribution
manage the smoothness of the distribution

e.g. Parzen, k-Nearest Neighbors, Naïve Bayes

(2) Histogram Method?

$$\hat{p}(x) = \frac{1}{h}\frac{k_N}{N} \tag{8}$$

h: the width of the subregions
k_N: the number of objects in each region
N: total number

(3) Parzen Density?

(i) fix cell shape (kernel/window function)

(ii) fix size of the kernel function

(iii) apply the kernel function on every datapoint

(iv) use the estimator:

$$\hat{p}(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - x_i}{h}) \tag{9}$$

(v) example:

Given a set of five data points:
$x_1 = 2, x_2 = 2.5, x_3 = 3, x_4 = 1, x_5 = 6,$
find Parzen probability density function (pdf) estimates at $x = 3$,
using the Gaussian function with $\sigma^2 = 1$ as kernel function.

- $\hat{p}(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x-x_i}{h})$
- $K(x) = K(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(\frac{(x-\mu)^2}{2\sigma^2})}$
- $\hat{p}(x) = \frac{1}{5}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}} e^{-(\frac{(x-x_i)^2}{2})} = \frac{1}{5}\left(\frac{1}{\sqrt{2\pi}} e^{-(\frac{(3-2)^2}{2})} + \frac{1}{\sqrt{2\pi}} e^{-(\frac{(3-2.5)^2}{2})} + \frac{1}{\sqrt{2\pi}} e^{-(\frac{(3-3)^2}{2})} + \frac{1}{\sqrt{2\pi}} e^{-(\frac{(3-1)^2}{2})} +$

(4) Parzen width parameter: optimization?

- Optimize the likelihood, eg. Maximize log-likelihood

$$LL(X) = \log\left(\prod_i \hat{p}(x_i)\right) = \sum_i log(\hat{p}(x_i))$$

- Use the average 10-nearest neighbor distance
- Use a heuristic (out of scope)

$$h = \sigma\left(\frac{4}{p+2}^{\frac{1}{p+4}}\right) n^{\frac{-1}{p+4}}$$

$$\sigma^2 = \frac{1}{p}\sum_{i=1}^{p} s_{ii}$$

(5) K-Nearest Neighbours?

(i) locate the cell on the new point x

(ii) do not fix the volume of the cell: grow the cell until it covers k objects: find the k-th neighbours

(iii) predict the class y of new point x

(iv) use the estimator:

$$\hat{p}(x|y_i) = \frac{k_i}{n_i V_k} \tag{10}$$

V_k: the volume of the sphere centered at x with radius r
k_i: is the number of neighboursof class i within V_k
$\hat{p}(y_i) = \frac{n_i}{n}$: class priors

(v) use Bayes:
$\hat{p}(x|y_i)\hat{p}(y_i) > \hat{p}(x|y_j)\hat{p}(y_j) \rightarrow k_i > k_j$

(6) The influence of neighbours k (KNN)?

- Large value → everything classified as the most probable class
- Small value → highly variable, unstable decision boundaries

(7) Equal Voting (KNN)?

- use odd k
- random: flip the coin to decide positive/negative
- prior: pick class with greater prior
- nearest: use 1-nn classifier to decide

(8) Distance measures (KNN)?

- The key component of the kNN algorithm:
- defines which examples are similar and which aren't
- can have strong effect on performance

- Examples: Euclidean, Manhattan distance, Hamming (categorical features), Kullback-Leibler (KL) divergence (for histograms), Custom distance measures (BM25 for text)

(9) Naive Bayes Classifier?

(i) We make a strong assumption: all features are independent

(ii) We assume conditional independence given y

(iii) We just estimate $p(x_i|y)$ per feature and multiply them.

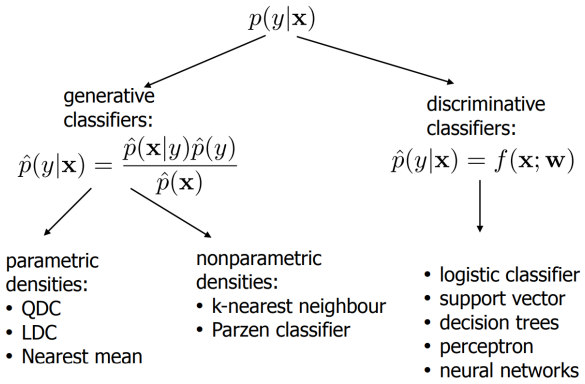
$$p(x|y) = \prod_{i=1}^{d} p(x_i|y) \tag{11}$$

(iv) No curse of dimensionality

(10) Zero frequency problem (NB)?
Solution: never allow zero probabilities

3.2 Lecture 6

(1) Recap for generative vs. discriminative and parametric vs. non-parametric?



(2) General Idea of Error Estimation?

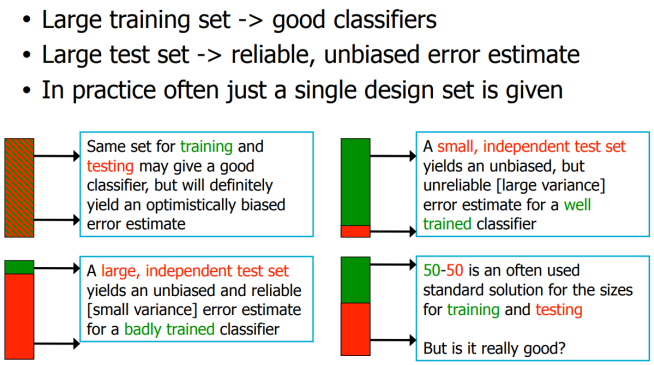
$$\hat{\epsilon} = \frac{1}{N} \sum_{i=1}^{N} Z_i \tag{12}$$

Z_i: 0 if x_i is correctly classified or 1 if x_i is incorrectly classified

$$\sigma_{\hat{\epsilon}}^2 = \frac{\epsilon(1-\epsilon)}{N} \tag{13}$$

N: the size of test set

(3) Training Set Size vs. Test Set Size?

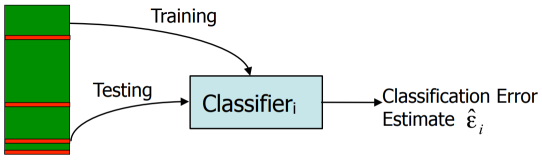


(4) Bootstrapping?

(i) Randomly draw n objects from N objects (with replacement)

(ii) Left-over objects are used for testing

(iii) Repeat m times



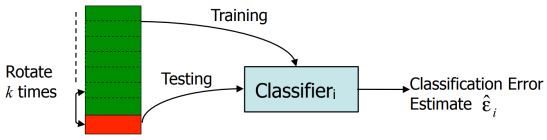
$$\hat{\varepsilon} = \frac{1}{m}\sum_i \hat{\varepsilon}_i$$

(5) K-fold Cross Validation?

(i) Divide the set into k folds

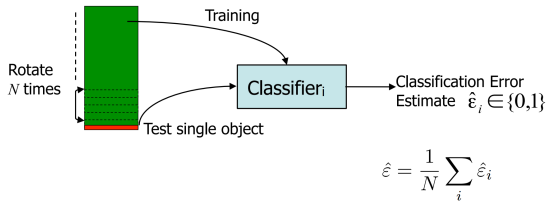
(ii) Pick k-1 folds for training, and the rest 1 fold for validating each round

(iii) Repeat k times

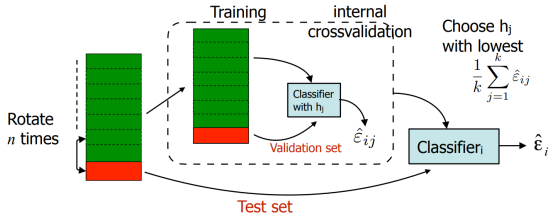


$$\hat{\varepsilon} = \frac{1}{k}\sum_i \hat{\varepsilon}_i$$

(6) Leave-one-out Procedure?

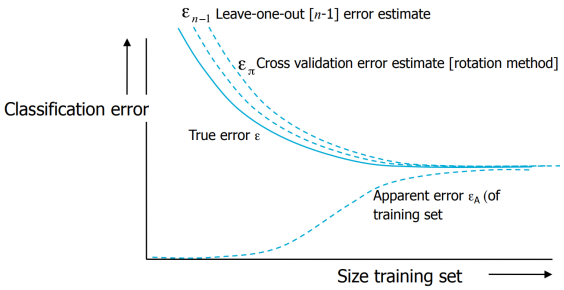


(7) Double cross-validation?
To optimise hyperparameters, do cross-validation inside another cross-validation:

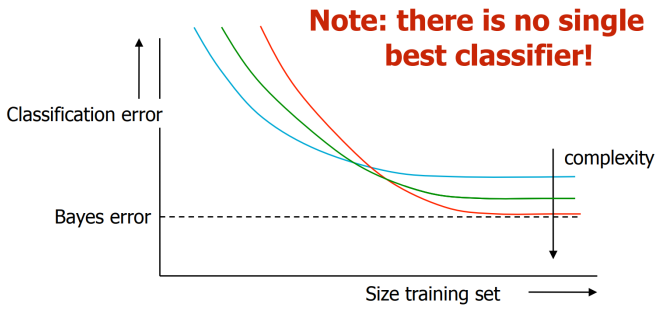


(8) Learning Curve?
Curves that plot [estimated] classification errors against the number of samples in training set
Give insight in:

- Amount of overtraining
- Usefulness of additional data
- How different classifiers compare
- Stability of training



(9) Different Classifier Complexity?



(10) Conclusion for Training and Testing?

- Larger training sets yield better classifiers
- Independent test sets needed for unbiased error estimates
- Larger test sets yield more accurate error estimates
- LOO cross validation "optimal", but might be infeasible
- 10-fold crossvalidation is often used
- More complex classifiers need larger training sets
- Same holds for larger feature set sizes
- Small training sets need simpler classifiers or smaller feature sets

(11) Bias-Variance Dilemma?

$$MSE = E_D[(g(x;D) - E_D[g(x;D)])^2] + E_D[(E_D[g(x;D)] - E[y|x])^2] \tag{14}$$

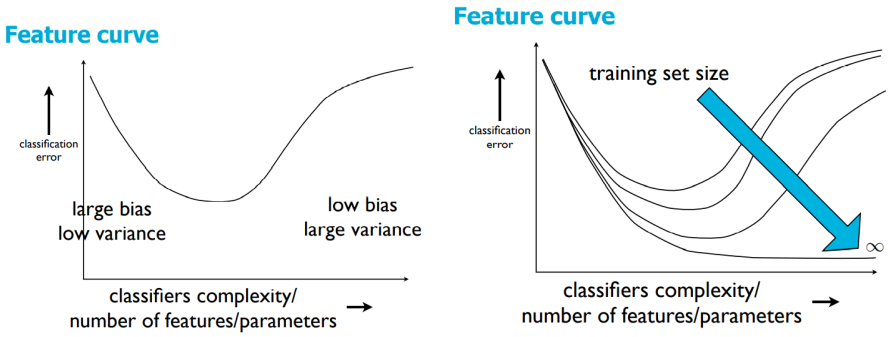
$$\rightarrow MSE = Variance + bias^2 \tag{15}$$

Variance: how much does classifier g vary over different training sets
The variance is an error from sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

Bias: how much does the average classifier g differ from the true output
The bias error is an error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).

(12) Bias-Variance Tradeoff?
More simple classifier is more stable (and need less data to train)
More complex classifier only works when you have sufficient number of training data

(13) Feature Curve?



(14) F1-score?

$$F_1 = 2\frac{precision \times recall}{precision + recall} \tag{16}$$

F_1 is useful applied in the unbalanced dataset.

(15) Error/Performance Measures?

- Error: probability of erroneous classifications
- Performance / accuracy: 1 – error
- Sensitivity of a target class [e.g. diseased patients]: performance for objects from that target class
- Specificity: performance for all objects outside target class
- Precision of a target class: fraction of correct objects among all objects assigned to that class.
- Recall: fraction of correctly classified objects; identical to sensitivity when related to particular class
- True positive rate: identical to sensitivity
- False positive rate: error for all objects outside target

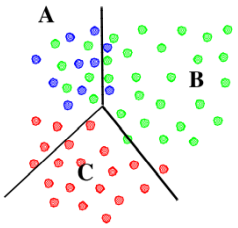
(16) Confusion Matrices?

- Provides counts of class-dependent errors : How many object have been classified as A that should have been classified as B?
- Give a more detailed view than overall error rate
- Can be used to estimate overall cost for particular classifier
- Example:

$N_A = 10,\ N_B = 30,\ N_C = 20$

$$E = \frac{c_{12} + c_{13} + c_{21} + c_{23} + c_{31} + c_{32}}{N_A + N_B + N_C}$$

$E = 14/60 = 0.2333$



$C = \text{confmat}(\Lambda, L)$

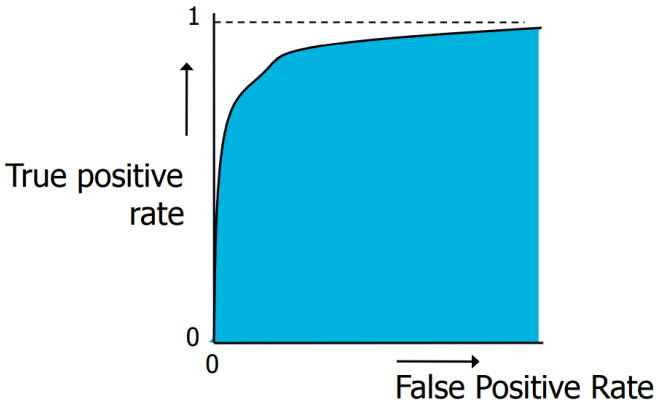
Λ real labels

L obtained labels

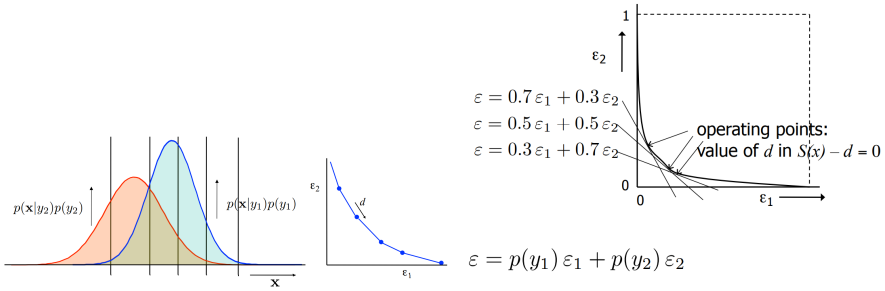
objects from	classified to A	B	C	
class A	8	2	0	0.20 error in class A
class B	6	23	1	0.23 error in class B
class C	4	1	15	0.25 error in class C
				0.228 averaged error

(17) ROC Analysis?

ROC: Receiver-Operator Characteristic (from communication theory)



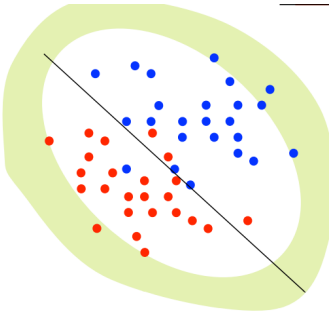
- Integrate the area: perfect classifier gives AUC=1.0
- Random classifier gives AUC=0.5
- Performance measure that is insensitive to class priors



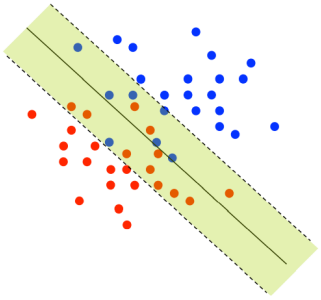
$$\varepsilon = p(y_1)\,\varepsilon_1 + p(y_2)\,\varepsilon_2$$

(18) Rejection?

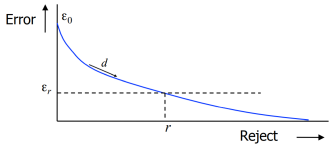
- Outlier Reject



- Ambiguity Reject



- Reject Curve



- Reject for solving ambiguity: reject objects close to the decision boundary → lower costs
- Reject option for coping with outliers
- ROC analysis in case of unknown or varying priors
- ROC analysis for comparing [and combining] classifiers

4 Week 4

4.1 Lecture 7

(1) Minimizing the "Risk"?

$$\hat{Y} = L(h(X), Y) \tag{17}$$

- X,Y: random variables corresponding to the input features and outcome of interest

- h: hypothesis function that maps input to a decision value
- L: loss/cost function that measures how well the predicted outcome matches the real outcome

Goal $\rightarrow min_{h \in H} E_{X,Y}[L(h(X), Y)]$

(2) Classifier Construction using Empirical Risk Minimisation?

$$min_{h \in H} \frac{1}{N} \sum_{i=1}^{N} L(h(x_i), y_i) \tag{18}$$

(i) Define the class of possible functions

(ii) Define a cost (loss) function to measure the "quality" of each hypothesis

(iii) Measure the average cost on the training data

(iv) Find the function that minimises this cost

(3) Hypotheses?

- When the outcome is continuous (regression), h(x) can directly correspond to the function we want to find
- In classification h(x) is the discriminant function which gives a real-valued output

(4) Why linear Hypotheses?

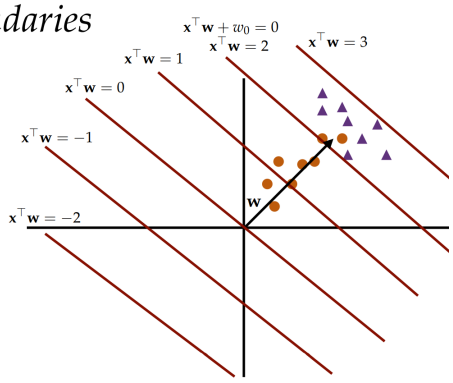
- "Simple"
- Optimization is often possible and fast
- Interpretable
- Often/sometimes a reasonable approximation, locally

(5) Linear Decision Boundaries?

Linear Decision Boundaries

Decision boundary is where $\mathbf{x}^\top \mathbf{w} + w_0 = 0$. This means $\mathbf{w}$ is orthogonal to every vector "within the decision boundary". For a 2D problem, this means it has to be a hyperplane of dimension 1 (a line).

What does the linear decision boundary look like for a 3D, 1D or KD problem?



(6) Classifier Construction using Empirical Risk Minimisation (Linear)?

$$min_{w,w_0 \in R^{D+1}} \frac{1}{N} \sum_{i=1}^{N} L(x_i^T w + w_0, y_i) \tag{19}$$

(i) Define the class of possible functions

(ii) Define a cost (loss) function to measure the "quality" of each hypothesis

(7) Regression solution?
Goal:

$$min_{w,w_0 \in \Re^{D+1}} \frac{1}{N} \sum_{i=1}^{N} L(x_i^T w + w_0, y_i) = min_{w,w_0} J(w, w_0) \tag{20}$$

Gradient Decent:

$$w_j := w_j - \alpha \times \frac{\partial J(w, w_0)}{\partial w_j} \tag{21}$$

∂: Learning rate $\sim$ Step size
Derive the gradient:

$$\frac{\partial J(w, w_0)}{\partial w_j} = \frac{1}{N} \sum_{i=1}^{N} 2(x_i^T w + w_0 - y_i) x_i^{(j)} \tag{22}$$

$$\frac{\partial J(w, w_0)}{\partial w_0} = \frac{1}{N} \sum_{i=1}^{N} 2(x_i^T w + w_0 - y_i) \tag{23}$$

(8) Stochastic Gradient Descent?

(i) To calculate the full gradient, we need to sum over all objects, before we take a step

(ii) Instead we could estimate the gradient using one, or a few objects, and take a step using this estimate of the gradient

(iii) The step is less "precise", but we can take many more steps in the same amount of time

(iv) Epoch: visiting all the data once

(v) So in stochastic gradient descent, we do updates within an epoch, while regular gradient descent does only one update per epoch.

(9) Flexible Step Size?
Fixing the step size seems naïve: smarter choices

- Second order methods: take the curvature of the function into account
- Line search
- Momentum

- Many other descent procedures, for instance, conjugate gradient
- Note: using this gradient descent iterative procedure was not necessary here! There is a closed form/analytic solution! For our next model, this will not be the case.

(10) GD leat to the best solution?

- Depends on the cost function and function class: What does the objective function look like?
- For convex functions, with the right settings, and the right amount of patience, we can reach the global minimum.
- The global minimum is the classifier with the minimal value for the cost function, which may not be the best solution for the problem! Recall: we want the solution to work well for the expected loss

4.2 Lecture 8

(1) Why Logistic Regression?

(i) There are other objectives in classification: different weight for different mistakes, accuracy of probabilities, correct "ordering" of the objects, etc.

(ii) Find a good solution for accuracy, by using some other "surrogate" loss

(2) Logistic Regression?

(i) Assume Y is either 0 or 1

(ii) Model the class posterior probability $p(Y = 1|X)$, using $\sigma(h(X))$

(iii) Measure of quality of the model: Given a choice of h(X), what is the

(iv) probability we would have observed the labels we observed in the data.

(v) N Bernoulli trials. The estimated probability of observing label y_i for object i:

$$\begin{cases} \sigma(h(x_i)) \ if \ y_i = 1 \\ 1 - \sigma(h(x_i)) \ if \ y_i = 0 \end{cases} \tag{24}$$

(3) Combining this, the likelihood is

$$L(h) = \prod_{i=1}^{N} \sigma(h(x_i))^{y_i} (1 - \sigma(h(x_i)))^{1-y_i} \tag{25}$$

$$log(L(h)) = -\sum_{i=1}^{N} y_i log[\sigma(h(x_i))] + (1 - y_i) log[1 - \sigma(h(x_i))] \tag{26}$$

(4) Logistic Regression Gradient?

$$J(\mathbf{w}) = -\sum_{i=1}^{N} y_i \log[\sigma(\mathbf{x}_i^\top \mathbf{w})] + (1 - y_i) \log[1 - \sigma(\mathbf{x}_i^\top \mathbf{w})]$$

Find the gradient, by taking the derivative and using $\frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z))$ *Homework: prove this!*

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = -\sum_{i=1}^{N} y_i \frac{1}{\sigma(\mathbf{x}_i^\top \mathbf{w})} \sigma(\mathbf{x}_i^\top \mathbf{w})(1 - \sigma(\mathbf{x}_i^\top \mathbf{w}))\mathbf{x}_i + (1 - y_i) \frac{1}{1 - \sigma(\mathbf{x}_i^\top \mathbf{w})} \cdot -\sigma(\mathbf{x}_i^\top \mathbf{w})(1 - \sigma(\mathbf{x}_i^\top \mathbf{w}))\mathbf{x}_i$$

Some terms cancel:

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = -\sum_{i=1}^{N} y_i (1 - \sigma(\mathbf{x}_i^\top \mathbf{w}))\mathbf{x}_i + (1 - y_i) \cdot -\sigma(\mathbf{x}_i^\top \mathbf{w})\mathbf{x}_i$$

Note that we can write this as:

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = -\sum_{i=1}^{N} (y_i - \sigma(\mathbf{x}_i^\top \mathbf{w}))\mathbf{x}_i$$

(5) Support Vector Machine?

- Intuitively, as long as all objects are correctly classified, maximize the size of the "margin": the distance of the closest points to the decision boundary
- The objects on the margin are the support vectors: they completely define the decision boundary. The other objects can be moved (outside the margin) without changing the classifier.
- Goal: intuition and principles behind SVMs. There are lots of extensions and theory that will have to wait.
- At first, assume the classes are linearly separable (we shortly cover more general setting, but the details will be in later courses)
- Intuition: stable solution (slight changes in the data would not have led to different decisions)
- Formally: you can prove nice generalization behaviour

We want to closest objects to be"away" from the decision boundary.
Problem: we always have the freedom to scale w to reach M, without changing the decision boundary. Let's choose the scale such that the closest object has decision value -1 or 1.
What is the distance of the decision boundary to the closest object?
The answer is **margin**
For the points on the margin, we have thus know that their distance to the decision boundary is:

$$\frac{|x_i^T w + w_0|}{||w||} = \frac{1}{||w||} \tag{27}$$

so the margin is: $\frac{2}{||w||}$
We wanted to maximise this margin: $max_{w,w_0} \frac{2}{||w||}$

Which gives the same solution as minimising: $min_{w,w_0} \frac{1}{2}||w||^2$
subject to:
$x_i^T w + w_0 \geq 1 \; if \; y_i = +1$
$x_i^T w + w_0 \leq -1 \; if \; y_i = -1$

(6) What if the data is not linearly separable (Soft-Margin SVM)?

Soft-Margin SVM

$$\min_{\mathbf{w},w_0} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}[1 - y_i(\mathbf{x}_i^\top \mathbf{w} + w_0)]_+$$

Where $[.]_+$ indicates the value of the function if the value is positive, and 0 otherwise.

(7) SVM as Empirical Risk Minimization?

$$\min_{\mathbf{w},w_0} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}[1 - y_i(\mathbf{x}_i^\top \mathbf{w} + w_0)]_+$$

Regularization Term

(8) Multiclass Classification?
Algorithm specific adaptations: directly construct a multi-class classifier
General techniques: combining multiple 2-class classifiers

(9) Solving the Ambiguity?
Possible solution to the ambiguity: use the decisionvalues rather than predicted labels of the classifiers
Ideally, we train these classifiers jointly, to optimize performance on the multiclass problem
Often, we can adapt the original model from 2 class to K-class, as in the logistic regression example

5 Week 5

5.1 Lecture 9

(1) Implicit bias
A preference or inclination for or against something
Based on learned coincidences, which unknowingly affect everyday perceptions, judgement, memory, and behaviour

Subconscious thought
We all have it
Might result in prejudice and discrimination

(2) Multiple sources of bias in ML
Bias in the training data - Human/cultural biases
Lack of diversity in ML developers
Algorithms
Evil programmers

5.2 Lecture 10

(1) Debiasing ML to Fairness in ML
Fairness is a multi-faceted concept
To improve fairness:

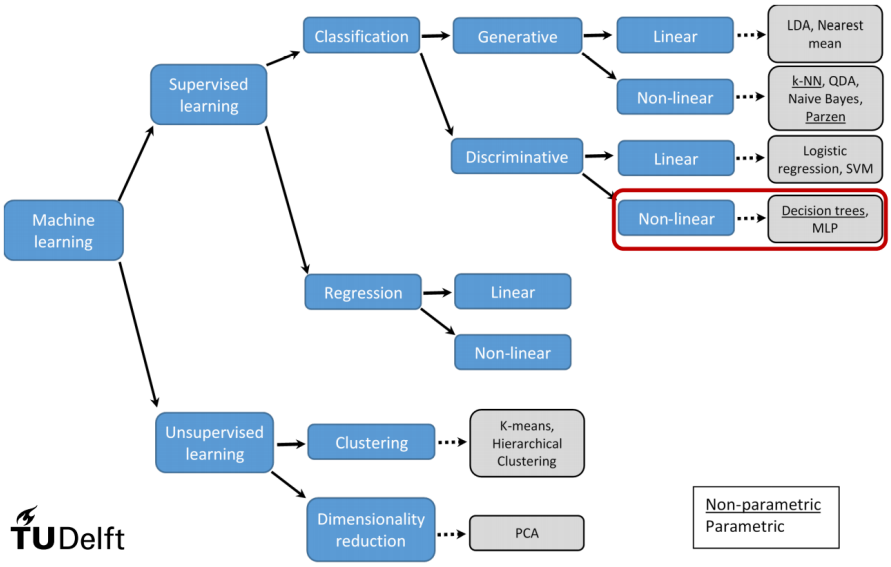
- Debias training data
- Use unbiased features
- Build smart algorithms

(2) Main points
Bias can occur at any point in a system/organization
Building fairness and non-discriminatory behaviour into AI and ML models is not only a matter of technological advantage but of social responsibility

6 Week 6

6.1 Lecture 11

(1) Overview of ML models?

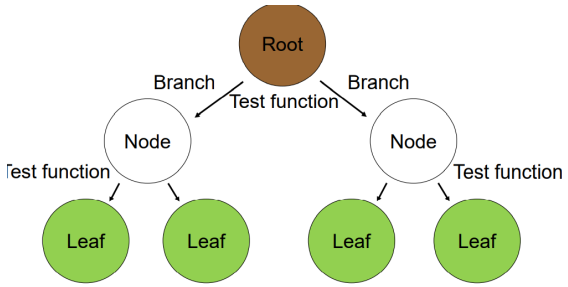


(2) Linear vs. Non-linear Classifier?

- Linear classifier:
  - 2D: a line
  - Higher dimensions: a hyperplane
- Nonlinear classifier
  - Locally, it can be linear
  - In general, has a complex shape

(3) Decision Trees?

(a) Some definitions:



(b) Walk through an example on how to classify an object given a tree

(i) Split the training data into unique regions sequentially

(ii) Structure of the tree is not predefined

(iii) Structure grows depending on the complexity and structure of the training data

(iv) At every node, a decision is made which splits the training data in smaller subsets

(c) What attribute/feature to use for splitting (note: several criteria exist)

(i) Splitting criterion: Entropy

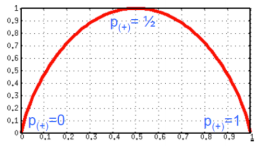
Splitting criterion: Entropy

- Entropy: $H(S) = - p_{(+)} \log_2 p_{(+)} - p_{(-)} \log_2 p_{(-)}$ bits
  - S ... subset of training examples
  - $p_{(+)}$ / $p_{(-)}$ ... % of positive / negative examples in S
- Interpretation: assume item X belongs to S
  - how many bits need to tell if X positive or negative
- impure (3 yes / 3 no):

$$H(S) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1 \text{ bits}$$

- pure set (4 yes / 0 no):

$$H(S) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0 \text{ bits}$$



(ii) Information gain

Information gain

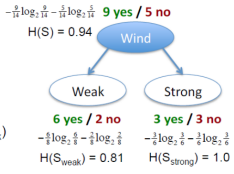
- Want many items in pure sets
- Expected drop in entropy after split:

$$Gain(S,A) = H(S) - \sum_{V \in Values(A)} \frac{|S_V|}{|S|} H(S_V)$$

V ... possible values of A
S ... set of examples {X}
S_v ... subset where $X_A = V$

- Mutual Information
  - between attribute A and class labels of S

Gain (S, Wind)
= H(S) – $^8/_{14}$ H(S_{weak}) – $^6/_{14}$ H(S_{weak})
= 0.94 – $^8/_{14}$ * 0.81 – $^6/_{14}$ * 1.0
= 0.049



(iii) Gain ratio:

$$SplitEntropy(S,A) = -\sum_{V \in Values(A)} \frac{|S_V|}{|S|} \log \frac{|S_V|}{|S|}$$

A ... candidate attribute
V ... possible values of A
S ... set of examples {X}
S_v ... subset where $X_A = V$

$$GainRatio(S,A) = \frac{Gain(S,A)}{SplitEntropy(S,A)}$$

penalizes attributes with many values

(d) How to avoid overfitting: pruning (note: several criteria exist)

(i) Stop splitting when not statiftically significant

(ii) Grow, then post-prune based on validation set

(iii) Sub-tree replacement pruning

- for each node: pretend remove node + all children from the tree and measrue performance on validation set
- remove node that results in greatest improvement
- repeat until further pruning is harmful

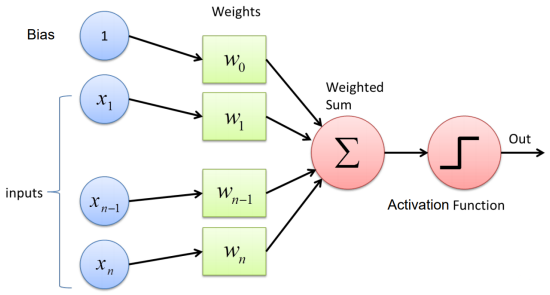
(e) Continuous features
Determine a binary threshold

(f) Multi-class decision trees

- Predict most frequent class in the subset
- Entropy: $H(s) = -\sum_c p_{(c)} log_2 p_{(c)}$
- $p_{(c)}$: % of examples of class c in S

6.2 Lecture 12

(1) A simple perceptron?
$x_j = input\ unit \in \Re,\ j = 1, .., d$
$x_0 = bias(always\ 1)$
$w_j = connection\ weight$
$y = output\ unit$
$Activation\ function = step\ function$



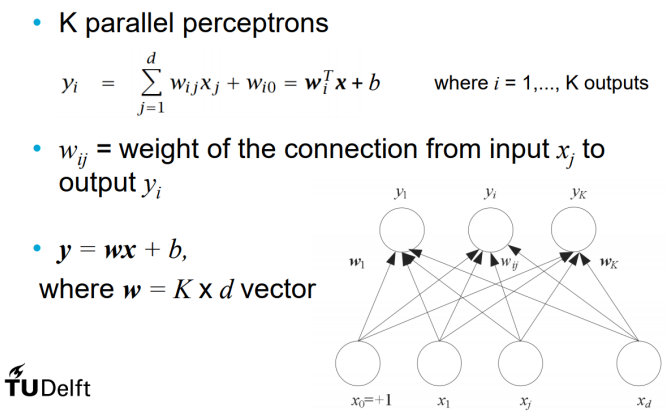
(2) How a single perceptron work?

(i) Multiply all input x with their weights w
(ii) Sum over all weighted inputs (= weighted sum)
(iii) Add the bias
(iv) Apply activation function to the weighted sum, e.g., unit step function
(v) A perceptron outputs the label of the winning class

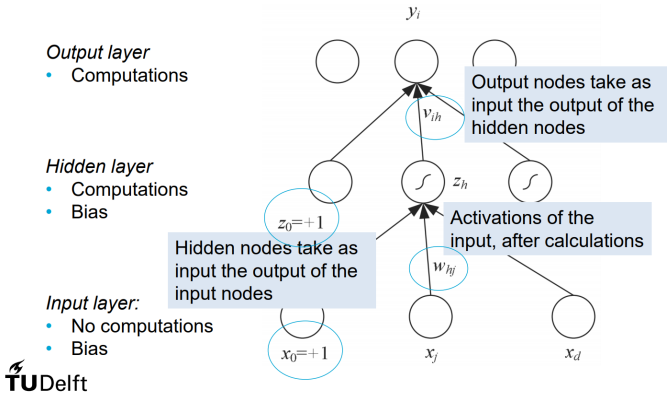
(3) Training Perceptron?
Training Perceptron → train the weights:

(i) Push a training example through the perceptron
(ii) Compare the produced output to the expected output (is label in the training data)
(iii) Feed that information back into the network so that the weights gradually converge on values that are appropriate for the task
→ Apply the Perceptron learning rule to each training example:
$w_{new} = w + (\alpha \times (output_{expected} - output_{calculated}) \times input)$
α: learning rate
Stop when $output_{expected} \simeq output_{calculated}$

(4) Multi-class classifier?



(5) Multi-layer perceptron?



(6) Forward and Backward Propagation?

- Feed-forward pass
  - Initialise weights with random value
  - Push input through the MLP row by row
  - Calculate and push forward the activations
  - Produce output value
- Backpropagation pass
  - Output value is compared to the label/target
  - Calculate error, using a loss function
  - Calculate gradient
  - Error gradient is propagated back through the network, layer by layer
  - Update weights depending on how much they contributed to the error
    → Find the weights that minimise the loss function

(7) Gradient vs. stochastic gradient descent?
Gradient descent (GD): forward pass using all training data before the backpropagation pass starts
Stochastic gradient descent (SGD): forward pass on a subset of the training set. It converges faster than GD, while minimisation of the error function might be less good than for GD

(8) Epoch, Batch training, and Online training?

- Epoch: All data used once for updating weights
- Batch training: Update weights after all training data are processed
- Online training: Update weights after each training sample

(9) Different classifiers can be combined to build better classifiers

7 Week 7

7.1 Lecture 13

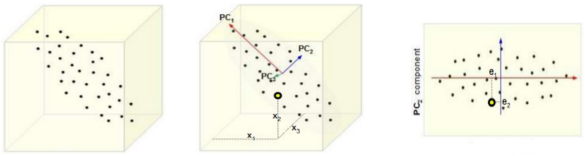
(1) Unsupervised learning?
- Dimensionality reduction: does not use information about the labels
- Clustering: Discover structures in unlabelled data

(2) Dimensionality reduction?

- Why do we want to reduce data dimensionality?
  . Make storage or processing of data easier
  . (Visual) discovery of hidden structure in the data
  . Remove redundant and noisy features
  . Intrinsic dimensionality might be smaller
- What does it mean to reduce dimensionality?
  . Transform high-dimensional data to data of lower dimensionality, whilst preserving the structure in the original data as good as possible
  . Feature selection Or Feature extraction
  Pick a subset of the original dimensions
  Use domain knowledge
  Use statistics-based selection methods
  Or
  Construct a new set of dimensions
  (Linear) Combinations of original

(3) How Principal Component Analysis reduces dimensionality?
Principal Components Analysi (PCA): maps the data onto a linear subspace, such that the variance of the projected data is maximized.

- Defines a set of principal components
  - 1st: direction of the greatest variability in the data
  - 2nd: perpendicular to 1st, greatest variability of what's left
  - ... and so on until d (original dimensionality)
- First *m* components become *m* new dimensions
  - change coordinates of every data point to these dimensions



(4) Eigenvalues & eigenvectors (eigenpairs)?
If data is zero-mean, the covariance matrix is simply: $M = \frac{1}{n} X X^T$
Principal components are given by the eigenvectors of the covariance matrix

$$Me = \lambda e \tag{28}$$

M:square matrix
λ: constant, eigenvalue of M
e: a non-zero column vector, eigenvector of M

(5) Pivotal condensation to find eigenpairs?

- Set *M* to $\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$
- Then $M - \lambda I$ is $\begin{bmatrix} 3-\lambda & 2 \\ 2 & 6-\lambda \end{bmatrix}$
- Determinant is $(3-\lambda)(6-\lambda) - 4$
- Setting to zero, solving equation $\lambda^2 - 9\lambda + 14 = 0$
- Gives solutions $\lambda = 7$ and $\lambda = 2$ being principal eigenvalues
- Let **e** be vector of unknowns $\begin{bmatrix} x \\ y \end{bmatrix}$
- Solve $\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 7 \begin{bmatrix} x \\ y \end{bmatrix}$

- Two equations: $\begin{bmatrix} 3x+2y & = & 7x \\ 2x+6y & = & 7y \end{bmatrix}$
- Both saying the same thing $y = 2x$
- Possible eigenvector: $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$
- Make unit vector (divide by length): $\begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$

- Second eigenvalue: repeat with $\lambda = 2$
- Equation becomes: $x = -2y$
- Second eigenvector: $\begin{bmatrix} 2/\sqrt{5} \\ -1/\sqrt{5} \end{bmatrix}$

(6) Power iteration to find eigenpairs?

- Let $M = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$
- Start with $\mathbf{x}_0$ being vector with 1s
- Multiply $M\,\mathbf{x}_0$: $\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$
- Frobenius norm equals $\sqrt{5^2+8^2} = \sqrt{89} = 9.434$
- Obtain $\mathbf{x}_1$: $\mathbf{x}_1 = \begin{bmatrix} 0.530 \\ 0.848 \end{bmatrix}$
  - Next iteration: $\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 0.530 \\ 0.848 \end{bmatrix} = \begin{bmatrix} 3.286 \\ 6.148 \end{bmatrix}$
  - Frobenius norm equals 6.971 so x_2 becomes

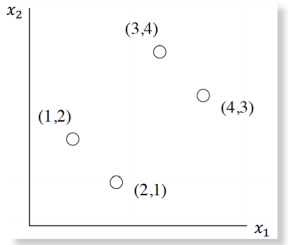
$$\mathbf{x}_2 = \begin{bmatrix} 0.471 \\ 0.882 \end{bmatrix}$$

  - Repeat, converges to $\boldsymbol{x} = \begin{bmatrix} \mathbf{0.447} \\ \mathbf{0.894} \end{bmatrix}$
  - Principal eigenvalue

$$\lambda = \mathbf{x}^T M \mathbf{x} = \begin{bmatrix} 0.447 & 0.894 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix} = 6.993$$

(7) PCA in a nutshell?

- X matrix whose rows represent (zero-mean) points in Euclidean space
- Compute covariance XX^T and its eigenpairs
- E matrix whose columns are the eigenvectors, ordered as largest eigenvalues first
- X^TE : points of X transformed into new coordinate space
  - First axis (largest eigenvalue) most significant
  - Second axis (second eigenpair), next most siginificant
- Let E_k be first k columns of E
- Then X^TE_k is *k-dimensional* representation of X



$$X^T = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{bmatrix}$$

Not zero-mean

$$M = X^TX = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix}$$

not a covariance matrix because X is not zero-mean; M should be $\frac{1}{n}X^TX$, **where X is zero-mean**

- Assuming M is a covariance matrix $M = \begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix}$ (when X is not zero-mean)
- Find eigenvalues det $(M - \lambda I)$ = 0 $(30-\lambda)(30-\lambda) - 28 \times 28 = 0$
- Solution $\lambda = 58$ and $\lambda = 2$
- Solve $\begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 58 \begin{bmatrix} x \\ y \end{bmatrix}$
- Two equations telling same thing $\begin{bmatrix} 30x + 28y &= 58x \\ 28x + 30y &= 58y \end{bmatrix}$ $x = y$
- Unit eigenvector corresponding to eigenvalue 58: $\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$
- Similarly for eigenvalue 2

$$\begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 2 \begin{bmatrix} x \\ y \end{bmatrix} \quad \begin{bmatrix} 30x + 28y &= 2x \\ 28x + 30y &= 2y \end{bmatrix} \quad x = -y \quad \begin{bmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

- Given $X^T = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{bmatrix}$
- Mean of each dimension:

$$\mu_1 = \frac{1}{n}\sum_{i=1}^{n} x_{1i} = \frac{1}{4}(1+2+3+4) = 2.5$$
$$\mu_2 = \frac{1}{n}\sum_{i=1}^{n} x_{2i} = \frac{1}{4}(2+1+4+3) = 2.5$$

- Zero-mean data:

$$X^T = \begin{bmatrix} x_{11}-\mu_1 & x_{12}-\mu_1 & x_{13}-\mu_1 & x_{14}-\mu_1 \\ x_{21}-\mu_2 & x_{22}-\mu_2 & x_{23}-\mu_2 & x_{24}-\mu_2 \end{bmatrix} = \begin{bmatrix} -1.5 & -0.5 & 0.5 & 1.5 \\ -0.5 & -1.5 & 1.5 & 0.5 \end{bmatrix}$$

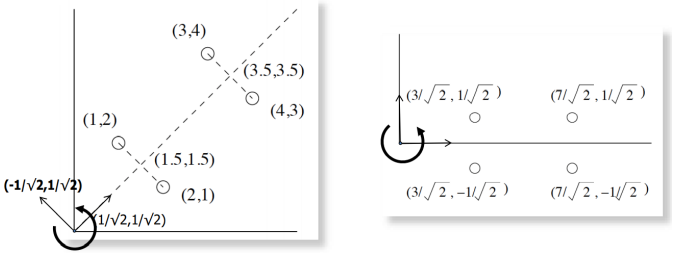
- Covariance matrix: $M = \frac{1}{n}X^TX = \frac{1}{4}\begin{bmatrix} -1.5 & -0.5 & 0.5 & 1.5 \\ -0.5 & -1.5 & 1.5 & 0.5 \end{bmatrix} \begin{bmatrix} -1.5 & -0.5 \\ -0.5 & -1.5 \\ 0.5 & 1.5 \\ 1.5 & 0.5 \end{bmatrix}$

- Matrix of eigenvectors for XX^T becomes

$$E = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

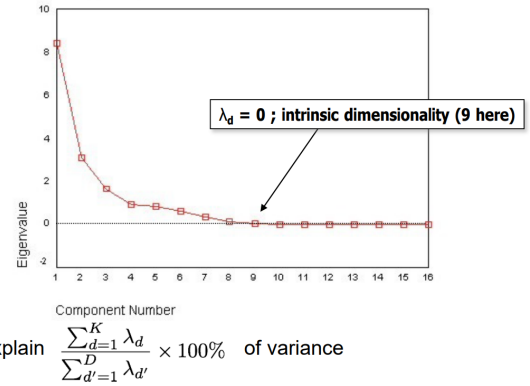
$$X^T E = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 3/\sqrt{2} & 1/\sqrt{2} \\ 3/\sqrt{2} & -1/\sqrt{2} \\ 7/\sqrt{2} & 1/\sqrt{2} \\ 7/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

- Any matrix of orthonormal vectors represents a rotation of the axes in a Euclidean space. Matrix *E* can be viewed as a rotation (in this case 45 degrees counterclockwise)



(8) PCA scree plot?

- *Scree plot* of eigenvalues shows amount of variance retained by the eigenvectors (*principal components, PCs*):



- First *K* PCs explain $\frac{\sum_{d=1}^{K} \lambda_d}{\sum_{d'=1}^{D} \lambda_{d'}} \times 100\%$ of variance

(9) PCA pre-process?

Covariance extremely sensitive to large values
- Multiply some dimensions by 1000
- Normalize each dimension to zero mean and unit variance

PCA assumes underlying subspace is linear

(10) PCA and classification?

PCA is unsupervised:

- maximizes overall variance of the data along a small set of directions
- does not know anything about class labels

- can pick direction that makes it hard to separate classes

Discriminative approach:

- look for a dimension that makes it easy to separate classes

(11) PCA Pros and Cons?
Pros

- reflects our intuitions about the data
- dramatic reduction in size of data
  faster processing (as long as reduction is fast), smaller storage

Cons

- too expensive for many applications (Twitter, web)
- understand assumptions behind the methods (linearity etc.)

7.2 Lecture 14

(1) Intuitively understand of Clustering?
Finding natural groups in data where
– Items within the group are close together
– Items between groups are far apart

(2) Distance Measure?

$$Euclidean: \ d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{29}$$

$$Manhattan: \ d(x,y) = \sum_{i=1}^{n}|x_i - y_i| \tag{30}$$

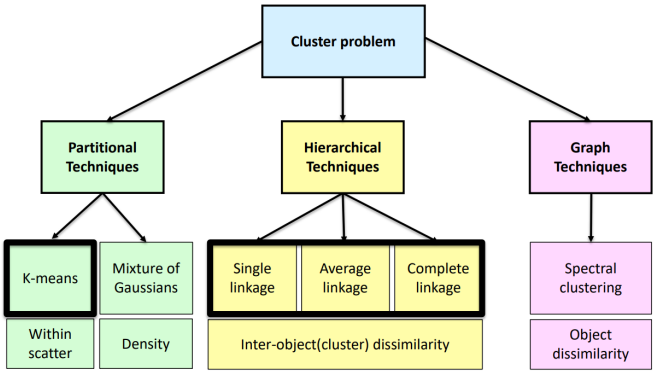
Cosine similarity, Pearson's Correlation Coefficient ...

(3) Clustering evaluation is a hard problem?

- Intra-cluster cohesion (compactness):
  – Cohesion measures how near the data points in a cluster are to the cluster's mean.
  – Sum of squared errors (SSE) is a commonly used measure.
- Inter-cluster separation (isolation):
  – Separation means that different cluster means should be far away from one another.

- In most applications, expert judgments are still the key

(4) Clustering techniques?



(5) Hard vs. Soft assignment?
Hard: each point assigned to 1 cluster
Soft: each point assigned cluster membership

(6) K-means?

(i) Choose k (random) data points (seeds) to be the initial centroids, cluster centers

(ii) Assign each data point to the closest centroid

(iii) Re-compute the centroids using the current cluster memberships

(iv) If a convergence criterion is not met, repeat steps 2 and 3

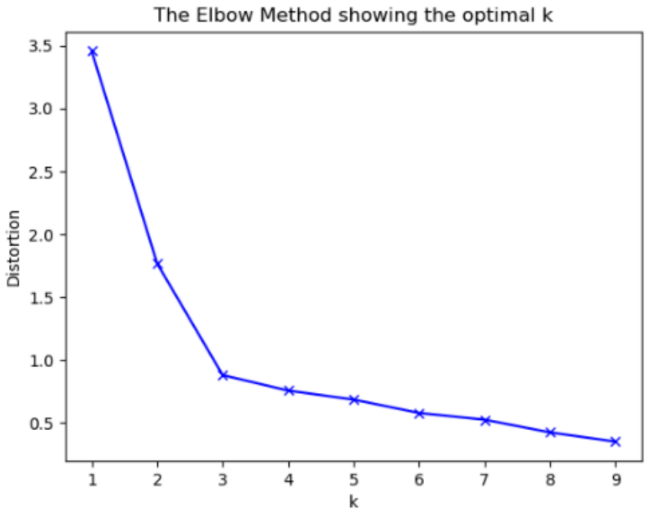
(7) K-means convergence (stopping) criterion?

- no (or minimum) re-assignments of data points to different clusters
- no (or minimum) change of centroids
- minimum decrease in the sum of squared errors (SSE):

$$J(c, \mu) = \frac{1}{n} \sum_{i=1}^{m_{C_i}} ||x_i - \mu_{C_i}||^2 \tag{31}$$

μ_{C_i} is cluster center to which x_i is assigned

(8) Elbow Method?



(9) Pros and Cons (KNN)?

Disadvantages:

– Finds only convex clusters ("round shapes")

– Sensitive to initialization

– Can get stuck in local minima

Advantages:

– Very simple

– Fast

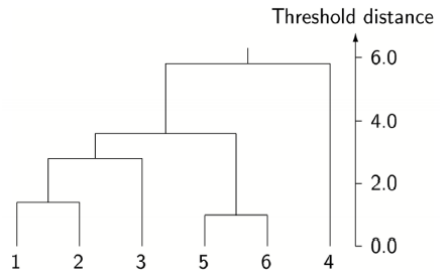
(10) Hierarchical clustering?

Agglomerative (bottom-up):

– each point starts as cluster

– group two closest clusters

– stop at some point

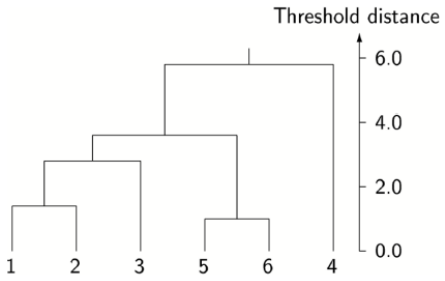


Divisive (top-down):

– all points start in one cluster

– split cluster in some sensible way

– stop at some point



(11) Agglomerative clustering?

(a) Determine distances between all clusters

(b) Merge clusters that are closest

(c) IF #clusters¿1 THEN GOTO 1

when to stop:

Different merging rules

- **Single linkage:** two nearest objects in the clusters :
$$g(R,S) = min_{ij}\{d(x_i, x_j): x_i \in R, x_j \in S\}$$
- **Complete linkage**: two most remote objects in the clusters :
$$g(R,S) = max_{ij}\{d(x_i, x_j): x_i \in R, x_j \in S\}$$
- **Average linkage**: cluster centres :
$$g(R,S) = \frac{1}{|R||S|}\sum_{ij}\{d(x_i, x_j): x_i \in R, x_j \in S\}$$

example is on slide 1039

(12) Pros and Cons (Hierarchical clustering)?

Pros:

– Dendrogram gives overview of all possible clusterings

– Linkage type allows to find clusters of varying shapes

– Different dissimilarity measures can be used

Cons:

– Computationally intensive

– Clustering limited to “hierarchical nestings”