Week 1　Data cleaning preprocessing Similarity measures

Data explosion society produces and store huge amounts data
Data is collection of examples (also called instances, records, observations, objects) Examples are described with attributes (features, variables)
nominal (categorical) - their values belong to a pre-specified, finite set of possibilities ←先指定的数据，数字数据↓
numeric (continuous) - their values are numbers
DM process: Business understanding→Data understanding→Data preparation→Modelling(there are many iterations in 3&4)→Evaluation→Deployment
Noise(3 type) due to- 1 distortion of values / 2 addition of spurious examples / 3 inconsistent and duplicate data(need to be detected and corrected)
Reducing noise **1.**Using signal and image processing and outlier detection techniques before DataMining. **2.**Using ML algorithms that are more robust to noise – give acceptable
results in presence of noise.
Dealing with missing values 1.Ignore all examples with missing values(if small % missing values). 2.Estimate missing values by using the remaining values(**Nominal -** replace the missing values for the most common value. **Numerical** – replace with the average value of the nearest neighbors)
Data aggregation Combining two or more attributes into one.
Data aggregation purpose **Data reduction**: less memory and computation time. **Change of scale**: provides a high-level view. **More stable data**: aggregated data is less variable than non-aggregated. **Disadvantage of Data aggregation**: potential loss of interesting details.
Feature extraction is the creation of features from raw data.
Feature subset selection process of removing irrelevant and redundant features and selecting a small set of features that are necessary and sufficient for good classification.
Feature subset selection methods 1.Brute force 2.Embedded(decision trees or other ML algorithms) 3.Filter(select features before the ML algorithm is run) 4. Wrapper(select the best subset for a given ML algorithm)
Feature weighting Can be used instead of feature reduction or in conjunction with it. **2type**: manually, automatically(use classification algorithms)
Converting attributes numeric to nominal(discretization) 2.numeric and nominal to binary(binarization). Needed as some ML algorithms work only with limited attributes.
Binarization(no best method) numeric→categorical→integer→binary.
Discretization 离散化 Converting numeric into nominal. **2 types:** unsupervised and supervised. **2 question:** How many categories? Where should the splits be?
Unsupervised discretization The user specifies intervals. Use 3 methods to splits data: 1.equal width. 2.equal frequency. 3.clustering(e.g. k-means)
Supervised discretization(entropy-based) Splits are placed so that they maximizes the purity of the intervals. Entropy(S) is a measure of the purity of a dataset (区间). The higher the entropy, the lower the purity of the dataset(means bad).

$$entropy(S) = -\sum_i P_i \log_2 P_i \qquad totalEntropy = \sum_i^n w_i\, entropy(S_i)$$

Entropy-based Discretization Algorithm: evaluate all possible splits and choose the best one (the lowest total entropy); repeat recursively until stopping criterion satisfied.
Normalization and standardization Attribute transformation to a new range, e.g. [0,1]. **Used to** avoid the dominance of attributes with large values over attributes with small values. **Required for distance-based ML algorithms**.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad x' = \frac{x - \mu(x)}{\sigma(x)}$$

Left is Normalization Right is Standardization.(x-original value)(x'-new value)(μ(x)-mean value) (σ(x)-standard deviation of the attribute)

$$\sigma(x) = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \mu(x))^2}$$

Similarity measures 相似性度量 Two main types of measures 1.Distance 2.Correlation.
Euclidean distance(L2 norm)-most frequently used

$$D(A,B) = \sqrt{(a_1-b_1)^2 + (a_2-b_2)^2 + \ldots + (a_n-b_n)^2}$$

Manhattan distance (L1 norm)

$$D(A,B) = |a_1-b_1| + |a_2-b_2| + \ldots + |a_n-b_n|$$

Simple Matching Coefficient (SMC)-SMC is not suitable for sparse 稀疏数据 data. SMC = (f11+f00)/(f01+f10+f11+f00)
Jaccard coefficient(替代 SMC)- J=f11/(f01+f10+f11)
Cosine similarity Useful for sparse data (e.g text documents, suit for both binary and non-binary)

$$\cos(d_1, d_2) = \frac{d_1 . d_2}{\|d_1\| \|d_2\|}$$

d1 • d2 两个点对点相乘 ||d1||自己乘自己开根号(L2norm)
Correlation Measures linear relationship between numeric attributes

$$corr(\mathbf{x},\mathbf{y}) = \frac{covar(\mathbf{x},\mathbf{y})}{std(\mathbf{x})std(\mathbf{y})}$$

$$covar(\mathbf{x},\mathbf{y}) = \frac{1}{n-1}\sum_{k=1}^{n}(x_k - mean(\mathbf{x}))(y_k - mean(\mathbf{y}))$$

Corr(x,y) also called Pearson correlation coefficient.
Distance measures for nominal attributes Various options depending on the task and type of data;对字段进行距离比较，比如字段一致就是 1 不一致就是 0.

Week 2　KNN/1R and PRISM

K-Nearest Neighbour algorithm
Two types of attributes - categorical(nominal): values belong to a pre-specific, finite set of possibilities. / numeric(continuous): values are numbers.
Normalization - attribute transformation to a new range, e.g. [0, 1]. Used to avoid the dominance of attributes with large values over attributes with small values when calculating distance.
Min-max scaling - method for normalization.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Distance measure - Euclidean distance(L2 norm): p = 2, most frequently used / Manhattan distance(L1 norm): p = 1, 注意绝对值.
Distance for nominal - 0 if values are the same, 1 if not.
K-nearest neighbour - very sensitive to the value of k. More k increases the robustness to noisy examples. Can also be used for regression, prediction be the avg() of neighbours.
Weighted nearest neighbour
Closer neighbours count more than distant neighbours. Weight can be w = 1/d^2.
KNN discussion - often very accurate / slow for big datasets, requires efficient data structure KD-trees. / distance based: need normalization / produces arbitrarily shaped decision boundary defined by a subset of Voronoi edges. / slow for high-dimensional data. / sensitive to k.
Rule based algorithm - 1R
generate 1 rule that tests the value of a single attribute, like a single layer decision tree.
Determine label - 对每个叶子节点而言，选择 label 中更多的一项作为规定结果。
Determine best rule - (best attribute) one with smallest error rate(highest accuracy) on training data.
Algo - generate a rule for eacg attribute. / evaluate each rule on the training data and calculate the errors. / choose one with smallest number of errors.
1R discussion - simple and computationally efficient algo / rules easy to understand./ numerical datasets require discretization.
Rule based algo - PRISM
rule based covering algo. construct a set of if-then rules that cover all examples from this class and no examples from other classes. / accuracy on training data always 100%.
algo - for each class: start with an empty rule and add conditions / each condition tests the value of 1 attribute / by adding new test, the rule coverage is reduced, become more specific./ FINISH when p / t = 1, the rule only covers examples from 1 class.
p / t - t: total number of examples covered. / p: numbers from the class that is considered.
PRISM discussion - order-independent. / some test examples may not be covered by the PRISM rules and will not receive classification: assign them to the class with most training examples(default rule) / discretization is needed when dealing with numeric attributes.

W3　Linear regression and logistic regression

1 Simple Linear regression
Ordinary least squares(OLS) - 最小二乘法。将 y 的公式带入残差，对 x 求导，最终化简得到 slope = cov(x, y) / var(x) = r * sd(y) / sd(x)。
r = corr(x, y) = covar(x, y) / sd(x)sd(y)
R^2 - 所有 y 中，有多少比例的 y 被解释了。/ **R^2 = 1 - SSE / SST = SSR / SST**。/ 越接近 1 则模型越 fit，但不一定越 precious。
SSE - sum of squared errors
SSR - sum of squared regression - 预测值与因变量均值的差距
SST - sum of squared total - 因变量自己与均值的差距
SST = SSE + SSR
MAE – mean absolute error
MSE – mean squared error
RMSE – root mean squared error
2 Multiple linear regression

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}(h_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{d}\theta_j^2$$

Model fit to data　regularization

3 Gradient Descent
How to learn theta - choose an initial value / iteratively choose a new value to reduce J(theta) / until reach minimum.
Gradient decent - 使用 cost function 计算不同 theta 下的损失，当损失最小时结束。如何选择不同的 theta 就需要 gradient decent。使损失函数对目标 theta 求偏导，化简得到：

$$\theta_j \leftarrow \theta_j - \alpha * \frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right) x_j^{(i)}$$

Batch gradient descent - scan through all training set and take one step.slow.
Stochastic - choose one in training set and take one step. may not converge to minimum but fast.
mini -batch - trade-off choice.
4 Logistic regression
使用 logistic 也称为 sigmoid function。

$$h_\theta(\mathbf{x}) = g(\theta^T \boldsymbol{x}) = \frac{1}{1+e^{-\theta^T x}}$$

当得出大于等于 0.5 的值，则预测值为 1。
Cross-Entropy - 对于对数回归，使用 SSE 作为损失函数会导致函数非凸，有多个最小值而无法通过梯度下降获取全局最优。这里引入 cross-entropy，也成为 log loss。

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n} cost(h_\theta(\mathbf{x}^{(i)}), y^{(i)})$$

$$cost(h_\theta(\mathbf{x}), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

可被压缩为一个式子：

$$J(\theta) = -\frac{1}{2n}\sum_{i=1}^{n}[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))]$$

根据不同预测值选择最小化或最大化该值。
经过化简后，得到：

$$\frac{\partial}{\partial\theta_j} J(\theta) = (g(z) - y)x_j$$

Multi-class classification - one vs rest: train one logistic regression classification for each class i to predict. For each x, pich the class having highest value of probability. / use softmax function.
Regularization - l1(lasso): estimates sparse coefficients; equivalent to feature selection(能将某些 theta 压到 0，等同于选择特征) / l2(ridge): minimizes coefficients; pull coefficients towards 0;(倾向于不使用 0 而压到最小值)

W4　Probability-based: Naive Bayes / evaluating methods

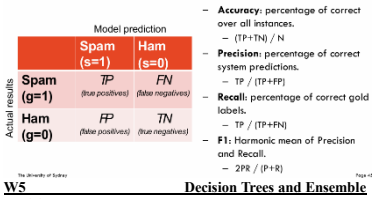
Naive Bayes
Probabilistic classification - compute the class membership probability.

$$P(H \mid E) = \frac{P(E \mid H)\,P(H)}{P(E)}$$

P(H|E) - posteriori probability: probability of an event after seeing the evidence. / also called conditional probability: probability of H given E.
P(H) - prior probability of H: probability of an event before seeing evidence. independent of E.
Naive Bayes algo - assumptions: 1 independence: attributes are conditionally independent of each other. / 2 Equally important: all attributes are equally important.
Those assumptions are almost never correct, that is why so called NAIVE.
Laplace correction / smoothing - when one attribute is 0, add 1 in case of 0 probability.
Missing value - ignore it. No need to calculate it. Keep processing the rest of the attributes.
Bayes for numeric attributes - assume that the numeric attributes follows a normal distribution(Gaussian distribution) and use *probability density function.*

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

无偏 n-1 标准差。
Naive bayes - discussion - probabilities are calculated easily due to independent assumption. / fast - requires on scan of the training data to calculate all statistics for both nominal and continuous attributes. / robust to isolated noise points - such points only negligible impact on conditional probabilities. / afraid of correlational attributes, they will reduce the power of bayes. Should apply feature selection ahead to identify and discard correlated attributes. / For numeric attribute, assume that they are normally distributed. If not: 1 discretize the data(numerical -> nominal attributes) 2 use other probability density function(poisson or binomial or gamma)
Evaluating machine learning algorithm
Evaluation procedures
Holdout method - split data randomly into **training set** and **test set**. build the model with training set, evaluate the model on test set, calculate the accuracy or other performance measures.
validation set - split into training, validation and test set. training set - Build the model / validation set - tune hyperparameters / test set - evaluate accuracy.
Stratification - some classes might be missing from the training set or test set, or be under-represented. Solution: stratification.
Repeated holdout method - repeating random split several times and calculating the average accuracy. (realize stratification)
Cross-validation - 10-fold: split the dataset into 10 sets, in each run, choose one of them as the test set, and training on the rest of the sets. / even better: repeat stratified 10-fold cross-validation.
For parameter tuning - grid search with cross-validation for parameter tuning
create the parameter grid / split the data into training set and test set. / For each paramater combination(grid): {traing a kNN classification on training data using 10 fold cross-validation / compute the cross - validation accuracy **cv_acc** / if cv_acc > best_cv_acc: best_cv_acc = cv_acc, best_para = current para./ rebuild the kNN model using the whole training data and best parameter, evaluate on test set.}
Leave-one-out cross-validation - special form of n-fold cross-validation: set the number of folds to the number of training examples. To say, build the model n times where n is the number of all examples.Advantages: make best use of data. / deterministic procedure: no random sampling, the same result will be obtained every time. Disadvantage: high computational cost, especially for large datasets.
More performance measures



W5　Decision Trees and Ensemble

Decision tree
contain nodes and branches. Each non-leaf node corresponds to a test for the values of an attribute. / each branch corresponds to an attribute value. / each leaf-code assigns to a class.
Constructing decision trees - strategy: top-down using recursive divide-and-conquer process.
First: select best attribute for root node and create branch for each possible attribute value. / Then: split examples in to subset, one of each branch extending from the node. / Finally: repeat recursively for each branch, using only the example that reach the branch.
Stop when all examples have the same class.
Entropy - 我们希望纯净的结果越早出现越好，于是需要一个衡量 purity 的measurement, information gain. It is based on another measure called entropy. / Entropy measures the homogeneity(purity) of this set with respect to the class. The smaller the entropy, the purity./

$$H(S) = I(S) = -\sum_i P_i . \log_2 P_i$$

(Pi: the proportion of examples that belong to class i) (measured in bits, and assume log2(0) = 0)
Split on *shape*:
$H(S_{circle})$=I(3/3, 0/3) = -3/3 log(3/3)-0/3 log(0/3) = 0 + 0 = 0 bits
$H(S_{square})$=I(2/4, 2/4) = -2/4 log(2/4)-2/4 log(2/4) = 0.5 + 0.5 = 1 bit
$H(S_{triangle})$=I(1/1, 0/1) = -1/1 log(1/1)-0/1 log(0/1) = 0 + 0 = 0 bits

H(S|shape)=3/8*0 + 4/8*1 + 1/8*0 = 0.5 bits
gain(shape)=0.95 - 0.5 = 0.45 bits

Information Gain - measure the reduction in entropy caused by using an attribute to partition the set. So the best attribute to use is the one with highest information gain./ Gain = T1 - T2 (T1: before split, T2: after split)
Pruning decision tree - if grow tree too perfect, the tree may be too specific and overfit(high accuracy on training set and low on test set). / When occurs?: training data is too small. / tree learnt noise.
Strategy for pruning - Pre-pruning: stop growing the tree earlier, before it's too specific. / Post-pruning: fully grow the tree and prune it.(Preferred). Post method: e.g. Sub-tree replacement / sub-tree raising / converting the tree to rules and prune.
Using validation set to decide how much to prune.
Sub-tree replacement - bottom up. 对于非叶子节点进行处理：移除其所有子节点，使用其子节点 example 中占大多数的 class label 作为新的叶子节点而替换他 / 在 validation set 上比较精确度。如果更高则保留。
Numerical Attributes
Discretizing - numerical attributes need to be discretized.(converted into nominal) / standard method: binary splits.
Procedure: sort the examples according to the values. / whenever the class changes, split it halfway. / evaluate information gain for every possible split, and choose the best. / IG for the best split = IG for the attribute.
Alternatives to information gain - gain ratio
当某个 attribute 有多个值时，他的 IG 会很大：因为更多的分支意味着更小的每个子集，意味着更确定的结果，意味着更低的熵！这将导致过拟合。
Gain ratio - takes into account the number of branches when choosing an attribute and penalized highly-branching attributes.
Desicion tree - summary - using pruning to prevent overfiting. / numeric attributes are converted into nominal(binary split).
Ensemble method
combines the prediction of multiple "base" classifiers. It tend to work better than the base classifiers they combined.
When ensembles better? - base classifiers are good enough(better than random guessing) / they are also independent.
Identical / independent - if identical, all base classifier would make same mistakes and combining will be meaningless / if independent, the error rate will be much lower.
It is not possible to ensure total independence among the base classifiers, slightly correlated is OK.
To achieve independent - generating disagreement:
Manipulating training data - creating multiple training sets from the original training set by **sampling**(Bagging and boosting)
Bagging - so called bootstrap aggregation. Given a dataset D, sample D' from D: randomly choose n examples from D **with replacement.** (some examples from D will appear more than once in D', and some may not appear at all.) / Probability = (1 - 1/n)^n. When n-> infinite, it equals to 63%.
Bagging performs better than single classifier. / effective for unstable classifiers(small change lead to big difference in prediction, e.g. decision tree, MLP). / When applied to regression, prediction will be the avg() instead of majority vote.
Boosting - most widely used ensemble method.
Ada boost - use a weighted training set. the higher the weight, the more difficult the example was to the previous classifier./ examples with higher weight will have higher chance to be selected in the training set of next classifier./combine the classifier using a weighted vote based on how the classifier performed on the training set. 能够将一系列表现平平的弱学习器组合成一个在训练数据上表现出色的强学习器。同时，它也指出了 Boosting 有效的一些关键条件，例如基础分类器不宜过于复杂，以及在迭代过程中弱学习器的性能不能迅速恶化。这些条件共同构成了 Boosting 算法强大且广泛应用的基础。
Gradient boosting - add a new model that minimized the error of previous model.
当前模型的训练集是(x, y')，其中 y'是实际结果减去前一个预测结果，也就是上一个模型的残差（也许是残差的残差的残差...）。最终的模型将每个模型预测结果串行相加即可。
Bagging - Boosting comparison
Similarity - using voting(class) and average(predict) to combine the output of the individual learners./ combine simple classifiers of same type. like decision

tree.
Difference - Base classifier: bagging separately / boosting iteratively. // combination method: bagging equally weighs / boosting(ada) weights based on performance on training data.
Manipulating training attributes
Random forest - 每个分类器只使用特征的一个子集，使用投票决定最终预测结果。同时还使用了 bagging 选择训练集。/ 准确度依赖于每个树都准以及每个树各不相同(not correlated.) / 训练很快因为只用部分 attribute.
Summary - 当单个分类器表现很差时使用 / 有些集成方法组合多种分类器，有些不 / 大多数使用投票 / 经常获得比赛胜利。

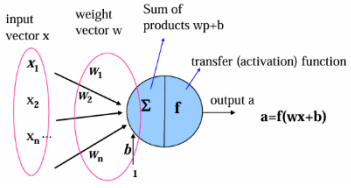
W6 Support vector machines / PCA

Hyperplane
Support vectors - points that lie closet to the decision boundary.
Margin - separation between boundary and closet points. Boundary is on the middle of the margin.一边到另一边的宽度。
maximum margin hyperplane - hyperplane with the biggest margins. it will have the highest possible distance to the training examples. It is better than the rest.
小间隔更容易过拟合，鲁棒差，更大的间隔有更好的泛化能力。
SVM
Process of linear SVM Learning a maximum margin, This could be formulated as a constraint optimization problem 约束优化问题. 这里的约束就是 $y_i(wx_i+b) \geq 1$,意思就是所有标签要被正确分类. This is an optimization problem that can be solved using Quadratic Programming (QP) and the Lagrange multiplier method.
SVM with soft-margin can allow some misclassifications, i.e. by considering the trade-off between the margin width and the number of misclassifications 这个方法允许出现一些错误分类. the modified method will construct linear boundary even if the data is not linearly separable. C is a hyper-parameter that allows this action, Large C means more emphasis on minimizing the training error than maximizing the margin.意思是 C 参数越大就越不允许错误分类.
Non-linear SVM Transform the data from its original feature space to a new space where a linear boundary can be used to separate the data, The learned linear decision boundary in the new feature space is mapped back to the original feature space, resulting in a non-linear decision boundary in the original space.就是把原始数据往高维变换以找到线性边界. This method use **kernel trick** to effectively compute dot products in high dimensional space, The kernel function specifies the relationship between the dot products in the original and transformed space.
Kernel trick means do the calculations in the original, not in the new higher dimensional space.
SVM can be applied to multi-class classification problems, they are transformed into 2-class problems, **and there is an extension for regression tasks.**
Summary - can form arbitrary decision boundaries (both linear and non-linear) / Three key concepts: The **decision boundary** is the **maximum margin hyperplane** – the task is formulated as an optimization problem // **Transform data into a** new (typically **higher dimensional space**) where it is more likely to be **linearly separable** // Kernel trick – do the **calculations in the original**, not in the new higher dimensional space

PCA(also called feature projection method) main idea is to find a new set of dimensions (axes) and project the data into it. (The dimensionality of the new space is smaller than the original space, the new axes capture the essence of the data.新空间维度小于原来的-新轴捕获数据本质.). It is an unsupervised method which dont use class infromation. **PCA can also be used for compression.**
Process of PCA The principal components are vectors that define a new coordinate system, They are ordered based on how much variance they capture. 1.The first axis goes in the direction of the highest variance in the data. 2.The second axis is orthogonal to the first one and goes in the direction of the second highest variance. 3.The third one is orthogonal to both the first and second and goes in the direction of the third highest variance, and so on.就是就是对 m 个维度给出 m 个新的轴，彼此一直正交然后朝大方差方向移动。
Idea - dimensionality **reduced** / **essence(variability) captured**.
Given: N examples with dimensionality m(m features)
Find: m new axes Z1, …, Zm, **orthogonal to each** other and such that Var(Z1) > Var(Z2) …
Z are the **principal components.**
how to reduce dimensionality? Select the k largest principal components and project our data points on them.通过投影.
How to select dimensions? 1.Set min % of variance that should be preserved 选择保留 95%的主成分. 2.Elbow method, Plot the number of dimensions as a function of variance, There is usually an elbow in the curve where the variance stops growing fast 画图然后选停止快速增长的点.
How to find the principal components? Using a standard matrix factorization method, called **Singular Value Decomposition (SVD)**. Any n x m matrix X can be written as X=U * V^T * Λ, 这里是完整的矩阵 U=mn, 但是降维后就只获取前 k 个 components 就变成 U=kn, 相当于把公式里所有 m 换成 n。**PCA 就是用 SVD 方法实现的.**
How to find principal components? - singular value decomposition (SVD)

W7 Deep Learning: Feedforward Neural Networks

Neural networks consist of **neurons** (units, nodes) that connected with each other with directed links where each connection has an associated numerical **weight**
Perceptron - using activation function(step transfer function) / binary output(1 / 0)



Feed Forward **Learning rule** - t – target output, a – actual output; x – input vector
Define error: e = t - a

$$\mathbf{w}^{new} = \mathbf{w}^{old} + e\mathbf{x}^T$$
$$\mathbf{b}^{new} = \mathbf{b}^{old} + e$$

Algo - initialize the weights w and bias b to small random values, epoch = 1./ **For** each training example {x, t}: 1 calculate the output(network activation) 2 compute the output error e = t - a. 3 update weights using **blue**. / check if stop condition is satisfied: all examples are correctly classified or a maximum number of epochs is reached; if yes - stop, otherwise epoch++ and repeat from step 2.
Perceptron capabilities - if examples are linearly separable, it will converge to a solution.
Multi-layer perceptron and backpropagation algorithm
The transfer function needs to be differentiable(可微分) - sigmoid
Gradient descent algo - wont guaranteed to find the global minimum, it converges to the nearest local minimum depending where it starts.
Backpropagation algo -

$$w_{pq}(t+1) = w_{pq}(t) + \Delta w_{pq}$$

Weight update formulas - 权重由此更新。

$\Delta w_{pq} = \eta \cdot \delta_q \cdot o_p$ 其为学习率 * q 的误差（靠近 output 端的神经元） * p 的输出（靠近输入端的神经元）
如果 q 是输出层：$\delta_q = (t_q - o_q) f'(z_q)$
如果 q 是隐藏层：$\delta_q = f'(z_q) \sum_i w_{qi} \delta_i$
如果使用 sigmoid 则
$f'(x) = f(x)(1 - f(x))$ 简化得到：
$\delta_q = (t_q - o_q) o_q (1 - o_q)$
$\delta_q = o_q (1 - o_q) \sum_i w_{qi} \delta_i$
Bias 视作一个输入永远为 1 的 weight（o = 1）使用同样的误差计算。

Stochastic gradient descent(SGD) - update the weights after each example, faster. / standard: update only when all examples are trained. / mini-batch: sum the error of a mini batch of training examples, update the weights.
Architecture
number of neurons in input layer: numerical attributes: 1 neuron per attribute / categorical with k values: k neuron per attribute.
number of hidden layers: much - overfitting / few - unferfitting.
training examples: 每一轮随机打乱 / 呈现更多训练出错的样本 / 不逐一，而是 mini-batch 输入，汇总误差
Learning rate - time-dependent(随时间变化)
Momentum – 动量，优化后：后者为动量优化
$\Delta w_{pq}(t) = \eta \delta_q o_p + \mu(w_{pq}(t) - w_{pq}(t-1))$
Weight initialization – 1 Random: small random values / 2 Xavier initialization: weight sampled from normal distribution, centred at 0.

$$\sigma = \sqrt{\frac{2}{N_{in}+N_{out}}}$$

Summary:
Perceptron form linear decision boundaries / multi-layers perceptron trained with backpropagation algorithm can form **arbitrary** decision boundaries.
Both need careful tuning, as may get stuck in bad local minimum.
Performance sensitive to: starting condition(weight initialization) / neurons much or few / learning rate / momentum / epochs.
Modern neural networks use techniques such as ReLu activation functions to reduce the vanishing gradient problem, dropout to reduce overfitting, better initialization of weights and more sophisticated optimisers (variations of the backpropagation algorithm).

applying ex.2: p_2= [0 1 1], t_2=0
a=step([0 0 0][0 1 1]+0)=step(0)=1
e=t_2-a=0-1=-1
w^{new}=[0 0 0]+(-1)[0 1 1] =[0 -1 -1]
b^{new}= 0+(-1)=-1
i.e. w an b have been updated

W7 CNN / RNN

Convolutional neural networks
Traditional NNs do not consider **the spatial structure of the image.**
CNN do this by **convolutional** and **pooling**
INPUT – （CONV – RELU – POOL） – FC
Filters are not pre-designed but learnt during training with backpropagation algorithm.
Padding allows for better coverage of the image around the image and feature extraction.用 0000 作为边框，左右都要 padding。
ReLU – *y = max(0, x)*
Pooling – max pooling: take the maximum value of the region / average polling: take the average value of the region.
MP selects the max value (brightest pixel) -> **useful for images with dark background, when we are interested in the lighter pixels**, e.g. MNIST dataset – white digits on a dark background
AP smooths the image.
Flatten – do this before the fully connected layers.
CNN summary:
are designed for image data but can be applied to other data types / The **convolutional layers extract features** / The **pooling layers reduce** the number of **parameters** to **prevent overfitting** and **improve the robustness** to shifts in the data / **fully connected layer** combines the extracted features and outputs the **probabilities for each class**
Recurrent Neural Networks (RNN) – for sequential data.
每次取出一个 element（词）进行处理, called **recurrent** because it contains feedback connections.(e.g. from hidden layer to the input layer，转回去了)
Recurrent – feedforward NN is a directed **acyclic** graph, RNN is directed **cyclic** graph.
每次这个神经元的输出会在下一个时间点传给自己继续处理。Hidden layer 每一刻接受 input 输入和上一个时间点的自己给自己输入（在一层 hidden layer 的情况下）。
Memory – RNN has a memory over past activations, can capture long-distance dependences.
New state / Old state / Input vector

$$h_t = f_W(h_{t-1}, x_t)$$ x-h, h-h, h-y
$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$
$$y_t = W_{hy} h_t + b_y$$

Backpropagation through time – propagated through the previous state of the network(into last time state).全局更新 Wxh, Whh, Why 三个变量。
RNN shortcoming – vanishing gradient problem 梯度消失问题。经过多个时间点的 Whh 处理，早期的输入可能逐渐趋于 0 而对遥远的的新的输入缺乏影响力。/
Long-term dependencies

LSTM networks
Long Short Term Memory networks
Forget gate layer – decide what information should be forget or keep. 0 – forget / 1 – keep.
Input gate layer – 判断今天的输入和上次的状态输入哪些内容值得长期保留，这个是输入门的工作。还有一个 tanh 函数将今天和昨天内容整合形成草稿，与前一个门同时作用形成待录入的知识。
Output gate layer – 决定向外输出哪些总结性的信息。
Summary – RNN can be applied for various type of tasks: 1 -many, many – many, many – 1.
LSTM are specifically designed to model dependencies between features that are **long distances apart** in the input sequence / They have a complex internal structure which includes 4 fully connected layers and 3 gates which control the information flow: forget, input and output / An LSTM cell can recognize an important input (via the input gate), store it in the long-term state and keep it as long as needed (via the forget gate) and extract it when needed (vis the output gate) / As RNN, LSTMs are trained with the "**backpropagation through time**" algorithm / However, there is no repeated multiplication by Whh as we backpropagate the error – no vanishing gradient problem / LSTMs have a complex architecture but have produced state-of-the-art results for many tasks, e.g. machine translation, speech recognition, handwriting recognition, speech synthesis and video analysis.

W9/10 Clustering

Partitioning - K-Means clustering algorithm
Selecting good initial centroids
Method 1: Run K-means several times with different randomly selected initial centroids; evaluate each clustering using SSE, select the clustering with the smallest Sum of Squared Error (SSE)多次选，每次算 SSE。
Method 2: K-means++: each point has a probability to be selected as a centroid that is proportional to the square of its distance to its closest centroid/ 在选取 centroid 的时候，每个点被选中的概率与他距离最近 centroid 的距离成反比。
Outliers – remove them
Shortcoming - Doesn't work well for natural clusters with **complex** (non-spherical) / clusters with vastly **different size** / clusters with **different density.**
K Means – Summary - Sensitive to **centroid initialization** / **Not sensitive** to the **order** in which the input examples are applied
Model-based – GMM

Hierarchical
Agglomerative (bottom-up) – merges clusters iteratively
Divisive (top-down) – splits a cluster iteratively
每次选择最近的合并（可以同时多个等值最近合并），直到合并为一体。
Distance – Single link(min) / complete link(max) / average link.
Hierarchical – Summary - Does not require the number of clusters to be specified in advance / Computationally expensive which limits its applicability to high dimensional data(space O(n2), time O(n3)) / Sensitive to noise and outliers
Density-based clustering: DBSCAN
DBSCAN - Clusters are regions of **High density**, separated from one another by regions with **Low density**.
In contrast to K-means which find circular clusters, DBSCAN can find clusters with **arbitrary** and **complex shapes**.使用密度区分 cluster 区域与其他区域。能发现随机和复杂的图形。
Neighbourhood of A: the area within a radius Eps 半径内则成为邻域。
Density of A: the number of points in the Eps including A itself.邻域内的点的数量。
Density threshold *MinPts:* the minimum number of points in the Eps.使成为 cluster 的最少密度。
Points – Core: 密度大于 threshold 的点 / Border：并非 core，但位于 core 的邻域内。/ Noise： 不属于上面二者的点。
Algorithm – 区分 core，border 与 noise / 将互相在领域内的 core 分为一个 cluster / 将 core 领域内的 border 分配给 core 的 cluster。如果出现 tie 则先到先得。
Method for selecting *Eps* and *Minpts* - 使用 k-th nearest neighbouthood 计算。
对于 core 与 border 点，在 k 小于 cluster 本身大小的情况下，能得出较小的 k-th nearest 值。
对于 noise 点，总能得出较大的值。
对所有点求取该值后，可以作图观察该值分布。
将 Eps 与 MinPts 设置为恰好稍大于图中拐点的值以获取最佳 cluster 效果。
clusters of varying density – 当一个区域的 noise 的 density 与另一个区域的 core density 一致时，DBSCAN 将无法有效地将所有 cluster 分辨。
Summary – Time：n^2(nlogn for using KD trees) / space n / Strengths: can find clusters that k-means cannot: form clusters of arbitrary shapes and different sizes. / No need for setting the number of clusters (but need Eps and MinPts) / Resistant to noise as it uses a density based defination.
Weakness: clusters with widely varying density (for k-means it's different size) / high dimensional data – more difficult to define density / sensitive input parameters Eps and MinPts / Eps and MinPts may be difficult to determine.
Grid-based clustering
break the data into grid cells and then form clusters from the cells that are dense enough. 比起 DBSCAN，Eps 定义的球形区域变成了矩形空间，Minpts 保留不变，density 的计算更加简单直观.

1. Define a set of grid cells.
2. Assign objects to the appropriate cells and compute the density of each cell.
3. Eliminate cells having a density below a threshold (MinPts).
4. Form clusters from contiguous groups of dense cells.

Cell define - split the value into **equal** width intervals / Break the values into intervals so that each interval contains an equal number of points. (**equal frequency intervals**).
Density - Denstiy of a grid cell = number of points / volume of a cell.
Summary – time nlogn, if search tree. / normally n.
Weak to grid partitioning / dependent on the choice of density threshold. / not good with different densities and noise. / cells at the boundary may be lost.
CLIQUE - CLustering In QUEst - dimension-growth, subspace grid-based clustering
It finds the subspace of the *highest* dimensionality where high-density clusters exist（找到有高浓度聚类存在的最高维度空间）
Using **Apriori pinciple:** only considers the dense cells from the previous lower dimensional subspace. No need to care about caring for non-dense cells at last-dim. (Help reduce the search space) 若一组点组成了一个基于密度的聚类与 k-dim 空间中，则同一组点在在所有可能的子集中都构成基于密度的聚类。
Summary - Won't work well with widely differing densities, since the threshold is fixed

Evaluation
Internal - Evaluate using *Unsupervised* measures / High similarity within a cluster, low similarity between clusters
Cohesion – ci 是聚类中心，x 是聚类中所有个体。将所有个体距离中心的距离加起来，就得到了 Cohesion

$$cohesion(K_i) = \sum_{x \in K_i} dist(x, c_i)$$

Separation - 将该聚类中心当作新的个体，总体样本中心当作新的中心，计算得到的 cohesion 即为 separation。

$$separation(K_i) = dist(c_i, c)$$

当处理全部 cluster 时，使用每个聚类大小作为 weight

$$separation = \sum_{i=1}^{k} |K_i| dist(c_i, c)$$

以上也会用平方处理
SSE = distance within a cluster $SSE = \sum_{i=1}^{k} \sum_{x \in K_i} (c_i, x)^2$
BSE = distance between clusters $BSE = \sum_{i=1}^{k} |K_i| (c_i, c)^2$

Silhouette coefficient – ai 是 cohesion，bi 是 i 到另一个 cluster 全部点的平均距离（遍历所有 cluster 选择最小的平均距离）

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

对每个点都求，最终取平均值。
越高越好。

Similarity matrices – 相似矩阵，如果值为 1 则在同一个 cluster，0 则不在。
先计算出 distance matrix，然后
sim = 1-(d – d_min) / (d_max – d_min)
在给出 matrix2，如果在一个 cluster 则值为 1，否则 0.
对这两个矩阵计算相关性，corr(x, y)

$$\mathrm{corr}(\mathbf{x},\mathbf{y}) = \frac{\mathrm{covar}(\mathbf{x},\mathbf{y})}{\mathrm{std}(\mathbf{x})\,\mathrm{std}(\mathbf{y})}$$

$$\mathrm{covar}(\mathbf{x},\mathbf{y}) = \frac{1}{n-1}\sum_{k=1}^{n}(x_k - mean(x))(y_k - mean(y))$$

External – using supervised measures / compare result with ground truth.
Entropy

$$entropy(K) = -\sum_i P_{i\cdot}\log_2 P_i$$

对每个簇分别计算，总的 entropy 要加权求和。越小越好
Purity

$$purity(K) = \max_i Pi$$

同样加权求和，越高越好

HMM
1 evaluation 已知观测，求发生概率

$$f_k(1) = A_0(k)\cdot e_k(x_1)$$

$$f_k(i) = e_k(x_i)\sum_j f_j(i-1)\,a_{jk}$$

$$a_{jk}=\mathrm{P}(\pi_i = k|\pi_{i-1} = j)$$

$$e_k(x_i) = P(x_i|\pi_i = k)$$

$$P(x) = \sum_k f_k(m)$$

2 decoding，使用 **viterbi algorithm**

$$V_k(i) = e_k(x_i)\max_j V_j(i-1)\;a_{jk}$$

$$a_{jk} = \mathrm{P}(\pi_i = k|\pi_{i-1} = j)$$

$$e_k(x_i) = P(x_i|\pi_i = k)$$

同时每一步还要

$$Ptr_k(i) = \mathrm{argmax}_j V_j(i-1)a_{jk}$$

储存来时的路