1 Key ingredients

Machine learning is the design and analysis of algorithms that improve their performance at some task with experience. The learning may take place offline(face detection), online(adaptive interface) or both(speech recognition). When a machine improves its performance at a given task over time, without reprogramming, it can be said to have learned something.

big picture We wish to develop methods and tools for building learning machines that can solve problems in combination with available data sets of training examples.

ml problem Learning involves improving performance at some task T, with experience E, evaluated in terms of performance measure P.

learning problem Components Task: the behavior or task that's being improved. Data: the experiences that are being used to improve performance in the task. Measure of performance: Provide more accurate solutions, cover a wider range of problems, Obtain answers more economically, Simplify codified knowledge, New skills that were not presented initially.

Machine learning ingredients Prior assumptions, Data, Representation, Model / hypothesis space. Feedback / learning signal, Learning algorithm, Evaluation.

Key types Supervised, Semi-supervised, Unsupervised(find structure in the data), Reinforcement(occasional, delayed information)

	Supervised learning	Unsupervised learning
Discrete	Classification / labeling Binary / multi-class	Clustering
Continuous	Regression Real-valued prediction	Dimensionality reduction

dataset •Training data is used for learning the parameters of the models, •Validation data is used to decide which model to employ, Sometimes reduced to training and testing a single mode (no validation step) •Test data is used to get a final, unbiased estimate of how well the model works.

AI vs ML Deductive reasoning vs Inductive reasoning. •Learning from a set of known facts and rules to produce additional rules or conclusions that are guaranteed to be true. •Learning from a set of examples to produce a general rules. The rules should be applicable to new examples, but there is no guarantee that the result will be correct.

Homogeneous coordinates embeds an N-dimensional representation in an N+1-dimensional space Advantage: The decision boundary passes through the origin of the extended coordinate system

Overfitting Learning that results in good performance on the training data but poor performance on the real task

Generalization generalize to the range of inputs/data that will be seen - not just solutions that work well on the training data. model the true regularities in the data and to ignore the noise in the data

Reducing model complexity Ockham's Razor, Prefer the simplest hypothesis that is consistent with the data

dimentionality Distance measures lose their usefulness in high dimensionality. intrinsic dimensionality is lower and the problem is feasible if the relevant dimensions can be identified.

1.1 tasks

A task requires an appropriate mapping – a model – from data described by features to outputs. Obtaining such a model from training data is what constitutes a learning problem. Tasks are addressed by models. Learning problems are solved by learning algorithms that produce models.

tasks, an abstract representation of the problem we want to solve.•Classification – assign the target variable to one of N states. •Regression – assign the target variable to a real-valued (scalar or vector) function of the input. •Clustering – grouping data without prior information. •Association rule learning •Subgroup discovery •Anomaly detection

predictive and descriptive •predict/estimate a target variable from features •exploiting underlying structure in the data, finding patterns.

1.2 models

a mapping from data points to outputs.

Geometric models use intuitions from geometry such as separating (hyper)planes, linear transformations and distance metrics. **Probabilistic models** view learning as a process of reducing uncertainty, modelled by means of probability distributions **Logical models** are defined in terms of easily interpretable logical expressions

Grouping models divide the instance space into segments; in each segment a very simple (e.g., constant) model is learned(Don't distinguish between individual instances within each segment, a finite (possibly coarse) resolution of the instance space)

Grading models learning a single, global model over the instance space.(Infinite resolution (in theory) possible, can distinguish between arbitrary instances)

probabilistic models aim to model the relationship between the feature values X and the target variables Y using probability distributions. Predict Y based on X and the posterior distribution P(Y — X) using Bayes'Rule.

MAP estimation or maximum a posteriori (MAP) rule, Choose Y that maximizes the value of P(Y — X).

maximum likelihood estimation or maximum likelihood (ML) rule, Choose Y that maximizes the value of P(X — Y).

generative model a probabilistic model from which we can sample values of all the data variables(Alternative: discriminative models)

Logical models focus at the level of human reasoning. often provide explanations for their results. often organized in tree structures: feature trees – that iteratively partition the space of all possible inputs (the instance space). Feature trees whose leaves are labelled with classes are called **decision trees**

1.3 features

Features – how we describe our data objects. Features are measurements performed on instances.

feature construction transform the features into a new feature space, to make the measurements in the appropriate coordinate system. Encapsulate the key similarities and differences in the data, Are robust to irrelevant parameters and transformations, Have a high signal-to-noise ratio.

kernel trick is a way of mapping features into another (often higher dimensional) space to make the data linearly separable, without having to compute the mapping explicitly.

dimensionality reduction There may be redundancies – correlations among features, Some of these features may be useless.

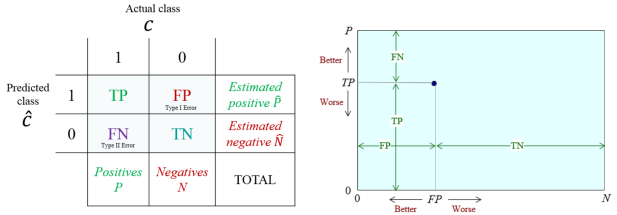
intrinsic dimensionality of (N-dimensional) data describes the real structure of the data embedded in N-space.

2 Classification

A classifier is a mapping that maps instances to classes. $\hat{c}(x)$ is an estimate of the (presumably) true but unknown function c(x) (aka the oracle). The training data comprises labeled instances.

2.1 Binary Classification

Two-class classification (k = 2, binary classification) is known as concept learning.



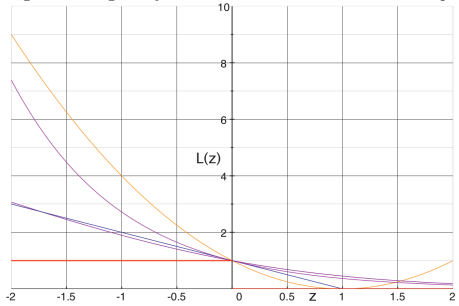
In a coverage plot, classifiers with the same accuracy are connected by line segments with slope 1. In an ROC plot, line segments with slope 1 connect classifiers with the same average recall.

2.2 Scoring & Ranking

A scoring classifier is a mapping, along with a class decision based on the scores (typically highest score). We can turn the feature tree into a scoring tree by computing a score for each leaf: **Score:** $\hat{s}(x) = log_2 \frac{spam}{ham}$

Margin: $z(x) = c(x)\hat{s}(x)$, •Positive if the estimate $\hat{s}(x)$ is correct, Negative if $\hat{s}(x)$ is incorrect. •Large positive margins mean "strongly correct", Large negative margins mean "the classifier screwed up".

In learning a classifier, we'd like to reward large positive margins and penalize large negative margins by the use of a loss function that maps the margin to an associated loss.



Loss functions: from bottom-left (i) 0-1 loss $L_{01}(z) = 1$ if $z \leq 0$, and $L_{01}(z) = 0$ if $z > 0$; (ii) hinge loss $L_h(z) = (1 - z)$ if $z \leq 1$, and $L_h(z) = 0$ if $z > 1$; (iii) logistic loss $L_{log}(z) = log_2(1 + exp(-z))$; (iv) exponential loss $L_{exp}(z) = exp(-z)$; (v) squared loss $L_{sq}(z) = (1 - z)^2$ (this can be set to 0 for $z > 1$, just like hinge loss).

the loss function is often chosen to be **convex**, since optimizing a convex function is computationally more tractable

Ranking: •Ranking error rate: $rank - err = \frac{err}{P+N}$, Ranking accuracy: $rank - acc = 1 - rank - err$

2.3 Probability Estimation

A class probability estimator is a scoring classifier that outputs probabilities over the k classes. A key issue here is that we generally do not have access to the true probabilities for training data. **Empirical Probabilities:** •relative frequencies

•$m - estimate = \frac{N_i + m\pi_i}{|S|+m}$, $\sum_i \pi_i$ •$Laplace\_correction = \frac{N_i+1}{|S|+k}$, k is the number of classes

3 beyond binary classification

multi-class contingency table, confusion matrix. •One-versus-rest scheme, learn k-1 models, apply in sequence •One-versus-rest scheme, learn a one-class model for each class •One-versus-one scheme, learn a model for each pair of classes •One-versus-one scheme with a decision tree

3.1 Multi-Classification

K-class classifiers: Combine several binary classifiers to build it:•learn k-1 models, apply in sequence •learn a one-class model for each class •learn a model for each pair of classes, Train k (k-1)/2 binary classifiers, apply them all to **x** and vote •with a decision tree

	Predicted			
	15	**2**	**3**	20
Actual	**7**	**15**	**8**	30
	2	**3**	**45**	50
	24	20	56	100

Accuracy = (15+15+45)/100 = 0.75

Class 1 precision = 15/24 = 0.63

Class 1 recall = 15/20 = 0.75

Etc.

3.2 Regression

Regression is another predictive ML task, learns a function: $\hat{f} : x \to R$ from examples - $f(x_i)$ •If fitting N-degree polynomial, choose one as low degree as possible to prevent overfitting. •The number of data points should be much greater than the number of parameters to be estimated. •**Residual:** $r(x) = f(x) - \hat{f}(x)$ •**loss function L** $L(x) = r^2(x) = (f(x) - \hat{f}(x))^2$

3.3 Unsupervised & Descriptive learning

skipped

4 Concept learning

means learning (typically binary) concepts from examples. The learned concept is the positive class. Everything else is the negative class. In concept learning, we want to learn a Boolean function over a set of attributes+values. The target concept c is the true concept. The **hypothesis** is a Boolean function over the features.

challenge decide which hypothesis is best, given the training data, we want to learn a concept that will generalize well to new, unseen instances.

4.1 hypothesis space

The space of all possible concepts is called the hypothesis space. how many possible instances are there for a given set of features: $F_1 \times F_2 \times F_N$. All combinations of feature values. The hypothesis space is the number of binary functions on these instances $_2F_1 \times F_2 \times F_N$.

conjunctive hypothesis space hypotheses that can be expressed as a conjunction of literals. add not care to each feature. (F1+1)x(F2+1)x(FN+1). In this conjunctive hypothesis space, we can't represent concepts like "all courses in AI or Graphics".

CHS learning a specific-to-general approach in coming up with a hypothesis.

4.2 pathes through hypothesis space

Every node connects upward to every more general hypothesis that includes it. Number of Donot know increases by 1 each level. Note that **True** can be appended to any proposition, so this is what's left when all the literals are removed.

Reducing the hypothesis space rule out all the hypotheses (concepts, nodes) that don't include at least one of the instances in our example. choose the least general.

Least general generalization (LGG) We want to generalize beyond our specific training data, but not too much – the most general hypothesis is to accept everything. the more general our hypothesis, the lower our False negative rate. less general our hypothesis, the lower our False positive rate.

Algorithm LGG-Set(D) – find least general generalisation of a set of ir

Input : data D.
Output : logical expression H.
$x \leftarrow$first instance from D;
$H \leftarrow x$;
while instances left **do**
  $x \leftarrow$next instance from D;
  $H \leftarrow$LGG(H, x) ; // e.g., LGG-Conj (Alg. 4.2) or LGG-Conj-ID
end
return H

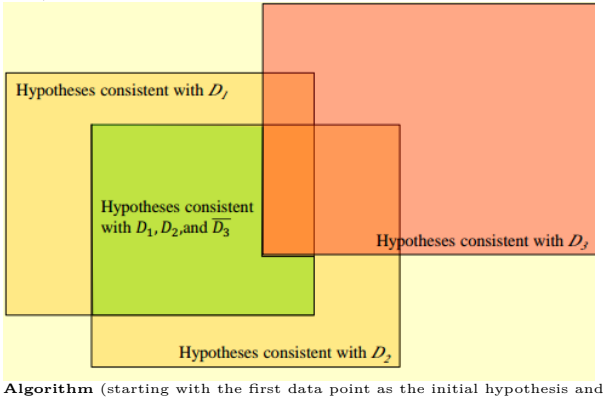
Algorithm LGG-Conj(x, y) – find least general conjunctive generalisation of two conjunctions.

Input : conjunctions x, y.
Output : conjunction z.
$z \leftarrow$conjunction of all literals common to x and y;
return z

Negative examples help to prune the hypothesis space. Rule out hypotheses that include negative examples.

internal disjunction $(2^{F1} - 1)(2^{F2} - 1)(2^{Fn} - 1)$ When all values of a feature are included, the feature becomes donot care.

Additional note •The number of hypotheses consistent with the data: For a given training data example, which is a leaf node at the lowest level in the graph, all connected nodes above it (including the data point itself) are hypotheses consistent with that data point. Adding a new, different data point prunes the nodes that represent hypotheses consistent with both data points. •The generality of various hypotheses: Higher/lower nodes represent more/less general hypotheses.



Algorithm (starting with the first data point as the initial hypothesis and generalizing with more data points) prunes hypotheses at each step, choosing the least general consistent hypothesis as the current hypothesis. **Adding a positive/negative training example** prunes the space of consistent hypotheses by eliminating hypothesis that **do not/do** have a link to the new example. A negative example will always prune the top node (True)

4.3 beyond conjunctive concepts

skip

4.4 learnability

Hypothesis languages can be extended to more realistic data by modifying the "all or nothing" nature of positive/negative training data. We can make richer hypothesis representation languages: CHS, CHS with internal disjunctions, Conjunctions of Horn clauses, Clauses in first-order logic. The richer the representation and thus the more expressive the hypothesis language, the more difficult the learning problem.

Complete and consistent A concept is complete if it covers all positive examples. A concept is consistent if it covers none of the negative examples. The version space is the set of all concepts that are both complete and consistent.

PAC learning Probably Approximately Correct. PAC outputs, with probability at least

$1-\sigma$ (most of the time), a hypothesis h such that $err_D < \epsilon$ (mostly right). we can guarantee this by choosing a large enough training set $|D| \geq \frac{1}{\epsilon}(ln|H| + ln\frac{1}{\sigma})$ This leads to the concept of **VC dimension**.

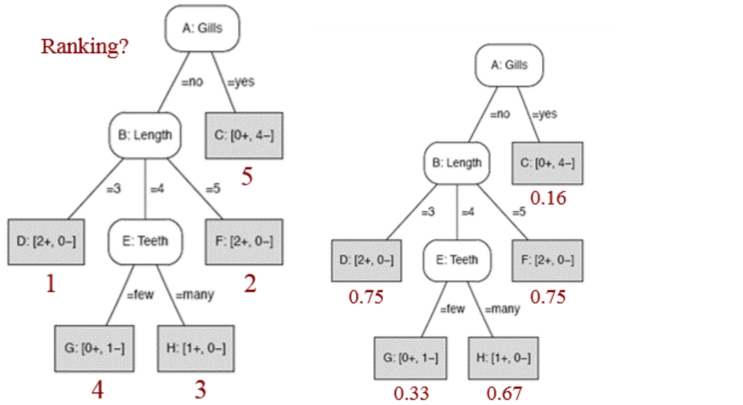
5 Decision trees

5.1 Decision Tree

A **decision tree** partitions the instance space by branching on feature values (literals), with leaves representing the learned concept. •Each leaf represents a conjunction of literals on its path •The learned concept is the disjunction of the positive leaves: $L_1 \vee L_2 \vee L_3$. •maximally expressive. they can separate any consistently labeled data – Thus more powerful than the conjunctive hypothesis space, overfitting. •**Simplify decision tree**: Merge feature labels and test the difference, Enforce a depth limit, Enforce a purity threshold, Enforce a purity increment threshold, Build the complete tree and then iteratively merge leaves based on lowest purity decrease up to a number of leaves N or a purity δ, Combinations of depth and purity measures.

5.2 Ranking & Probabilities

•Use empirical probabilities(Using Laplace correction) $\dot{p}$ - for segments i and j, order $i > j$ if $\dot{p}_i > \dot{p}_j$ •Ranking is with respect to a particular class (e.g., dolphin) •A ranking is on a set of m instances $X = \{x_1, ..., x_m\}$ •A decision tree with N leaves will have N different ranks •We can use those empirical probabilities (for each class) calculated for every leaf to create a probability estimation tree(Using Laplace correction)



5.3 Tree learning as Variance reduction

skipped

6 Linear models

6.1 least-square

are geometric models for which the regression functions or decision boundaries are linear. (Lines, planes, hyperplanes,N-dimensional planes). An affine function is a linear function plus a constant. **Pros**, Many functions can be reasonably approximated as linear, or at least as piecewise linear. They're simple, and thus easy to train. The math is tractable. They avoid over-fitting, they generalize well when the data is noisy. **Cons** prone to under-fitting, over-simplifying complicated function.
low variance stability, robustness, Performance on, different testing sets, should be similar.
High bias limited accuracy, underfitting, systematic (but consistent) errors.
parametric models we just need to learn the model parameters. kNN is non-parametric model. **Feature correlation** if the features in a multivariate regression problem with d input features are uncorrelated, then the problem reduces to d univariate problems.
Regularization avoid overfitting, if we think the training data may not be representative, or we have external knowledge about the problem beyond the data. **Classification** encoding the two classes as real numbers and thresholding the output function
Outliers An outlier is a measurement/observation that is distant from other observations. Could be due to measurement error or "heavy-tailed distribution" events. In other words, experimental anomalies. linear regression is sensitive to outliers(RANSAC).

6.2 perceptron

A linear classifier that will achieve perfect separation on linearly separable data. iterates over the training data, updating w every time it encounters an incorrectly classified example, until all examples are correctly classified. Guaranteed to converge if the training data is linearly separable - but won't converge otherwise.
Perceptron duality the weight vector is a linear combination of the training instances. we can view perceptron learning as learning coefficients of misclassified times.
Margin (z) of a sample is its distance from the classification boundary •Positive if it's correctly classified •Negative if it's incorrectly classified. $m = y(w^T x - t)$ $z = m/\|w\|$ The margin of a classifier on a training set is the minimum margin of the data points for that classifier. The **version space** of a linear classifier applied to linearly separable data is infinite.

6.3 svm

A support vector machine is a linear classifier whose decision boundary is a linear combination of the support vectors. we find classifier parameters (w,t) to maximize the margin. we can instead fix m and minimize w, Provided that none of the training points fall inside the margin. This leads to a constrained optimization problem. solve via a quadratic optimization solver!

$$\mathbf{w}^*, t^* = \arg\min_{\mathbf{w},t} \frac{1}{2}||\mathbf{w}||^2 \qquad \text{subject to } y_i(\mathbf{w}\cdot\mathbf{x}_i - t) \geq 1, 1 \leq i \leq n$$

$$\alpha_1^*, \ldots, \alpha_n^* = \arg\max_{\alpha_1,\ldots,\alpha_n} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i\cdot\mathbf{x}_j + \sum_{i=1}^{n}\alpha_i$$

$$\text{subject to } \alpha_i \geq 0, 1 \leq i \leq n \text{ and } \sum_{i=1}^{n}\alpha_i y_i = 0$$

$w = \sum_{i=1}^{n} \alpha_i y_i x_i$
decision boundary is defined only by the (typically few) support vectors from the training set - those that are nearest to the decision boundary (at the margin) Far examples for which $\alpha_i = 0$ can be removed from the training set without affecting the learned decision boundary. And the weight vector w is merely a linear combination of the (typically few) support vectors. The threshold can be found by solving m=1=wx-t for any support vector
Soft Margin SVM We introduce a slack variable for each training example to account for margin errors. The **complexity parameter** C is a user-defined parameter that allows for a tradeoff between maximizing the margin (lower C) and minimizing the margin errors (higher C). Note that when C = 0, this gives no penalty to outliers – which makes it equivalent to our basic linear classifier! minimal-complexity (low C) soft margin classifier summarizes the classes by their class means in a way very similar to the basic linear classifier. In effect, these slack variables implement what was called **hinge loss**
training examples divided into three cases:
$\alpha_i = 0$ these are outside or on the margin, $C > \alpha_i > 0$ these are the support vectors on the margin, $\alpha_i = C$ these are on or inside the margin.
Perceptron vs SVM In the perceptron model, we iteratively learn the linear discriminant w, which is a linear combination of the misclassified input vectors. In SVM learning, we solve a constrained optimization problem. In both, the linear decision boundary is a linear combination of the training data points. In the perceptron, just the ones that get misclassified in the iterative training. In the SVM, just the (few) support vectors. Both have a dual form in which the dot product of training data points is part of the main computation. Perceptron and (basic) SVM learning only converge to a solution if the training data is linearly separable. If the data is not linearly separable, we can employ a soft margin SVM, where we introduce a slack variable.

6.4 non-linear & probabilities

We can adapt our linear methods to learn (some) nonlinear decision boundaries by transforming the data nonlinearly to a feature space in which is suited for linear classification.
kernel trick is a way of mapping features into another (often higher dimensional) space to make the data linearly separable, without having to compute the mapping explicitly. The dot product operation in a linear classifier x1x2 is replaced by a kernel function K(x1, x2) that computes the dot product of the values (x1, x2) in the new (linearly separable) space.
Assumption achieving linear separation is worth the effort, the dot product is the key computation.
kernel svm We can then replace the Gram matrix G with the kernel matrix K, and use entries of K in the learning computation. After learning the α_i parameters, we can then classify a new instance x using $\sum_{i=1}^{K} \alpha_i y_i K(x, x_i) > t$ This sum is only over the support vectors, so it's an efficient computation
kernel functions linear, polynomial $(x_1^T x_2 + c)^d$, Gaussian(RBF, $exp(-\|x_1 - x_2\|^2/2\sigma^2)$, a measure of similarity between x1 and x2, scaled by σ, The larger σ is, the more effect a distant point x will have)
choose a kernel function 1. Selecting a kernel function entails: •Choosing the function family (polynomial, RBF)•Determining the parameters of the function. **2.** using cross-validation, **applying machine learning to machine learning 3.** Knowledge of the problem space can be helpful, Collected wisdom, In cases like this, try that kernel function.
probability

$$P(\oplus|d(\mathbf{x})) = \frac{\exp(d(\mathbf{x}))}{\exp(d(\mathbf{x}))+1} = \frac{\exp(\mathbf{w}\cdot\mathbf{x}-t)}{\exp(\mathbf{w}\cdot\mathbf{x}-t)+1}$$

$$LR = \frac{P(d(\mathbf{x})|\oplus)}{P(d(\mathbf{x})|\ominus)} = \exp\left(\gamma(d(\mathbf{x}) - d_0)\right)$$

$$\gamma = \frac{\bar{d}^{\oplus} - \bar{d}^{\ominus}}{\sigma^2} = \frac{\mathbf{w}\cdot(\mu^{\oplus} - \mu^{\ominus})}{\sigma^2}, \quad d_0 = \frac{\bar{d}^{\oplus} + \bar{d}^{\ominus}}{2} = \frac{\mathbf{w}\cdot(\mu^{\oplus} + \mu^{\ominus})}{2} - t$$

7 Distance-based models

Similarity is some function of distance. Clustering is grouping data without prior information(unlabeled data). **Why cluster?** •make apparent the natural groupings/structure in the data (perhaps for further processing) •To discover previously unknown relationships •To provide generic labels for the data
the basic linear classifier can be interpreted from a distance-based perspective as constructing exemplars that minimise squared Euclidean distance within each class, and then applying a nearest-exemplar decision rule
Clustering we organize data into classes such that: •The within-class (intra-class) similarity is high(Lower intra-class variance) •The between-class (inter-class) similarity is low(Higher inter-class variance) •Objects in the same group (a cluster) are more similar to one another than to objects in other groups (clusters)
Distance measures D(x1,x2) is a function D: $X \times X \to R$ such that for any xyz in X: 1. D(x,x) = 0 2. If x != y then $D(x, y) > 0$ 3. D(x,y) = D(y,x) 4. $D(x, z) \leq D(x, y) + D(y, z)$.
Common Distance measures Hamming(count differences), Manhattan, Euclidian, Minkowski(L_p), Chebyshev, Mahalanobis
Distance-based methods Methods for classification and clustering based on distances to exemplars or neighbors. Exemplar - a prototypical instance. Neighbor – a nearby instance or exemplar.
1NN Assign the new instance x to the nearest labeled training point (or exemplar). •Training = memorizing the training data •Each point is an exemplar, or exemplars are computed from the data •But it generalizes, unlike the lookup table approach •The implicit decision boundaries of a 1NN classifier comprise a **Voronoi tesselation**, Leads to piecewise linear decision boundaries.

8 Probabilistic models

skipped

9 Features

features (aka attributes) A mapping from the instance space X to the feature domain F. model can be thought of as just a new feature, particular combination of the input features, constructed to solve the task at hand. **Types** Quantitative(numeric scale, age), Ordinal(ordered set, ranks), Categorical(unordered set, color). **Statistics** Mode, Median, mean, range, std, Skewness, Kurtosis, Percentiles, Deciles, Quartiles, Interquartile range
Plots histogram, Percentile plot(CDF, Shows cumulative fraction of data - what % fall below a given value)
tree models ignore the scale of quantitative features, treating them as ordinal.
Feature construction and selection constructed new features from the given feature set, then select a suitable subset prior to learning(equal-frequency/width Discretisation, divisive/agglomerative discretisation, Thresholding, kernel methods, N-grams, Cartesian products of categorical features.)
kNN •Classify a new instance by taking a vote of the ≥ 1 nearest exemplars. •vote among all neighbors within a fixed radius r •combine the two, stopping when $count > k$ or $dist. > r$ •distance weighting, the closer an exemplar is to the instance, the more its vote counts. **ties** Preference to the 1NN, or Random choice. **Pros** train $O(n)$, simple, intuitive **Cons** test $O(n)$ require a good deal of storage, and can't easily represent a specific boundary geometry, rely on a useful distance metric.

Clustering vs. classification In a classifier, possible class labels are provided. In a clustering problem, possible labels are the cluster labels learned from the training set.
Assigning of labels or clusters to data points is classification.
Clustering find clusters (groupings) that are compact with respect to the distance metric.
Cons disregards the shape of the clusters
Scatter Matrix The scatter of X is defined as the trace of the scatter matrix.(The trace is the sum of the diagonal elements of a square matrix).
Kmeans is to find a partition that minimises the total within-cluster scatter. NP-complete, no efficient solution to find the optimal clustering (data partition). **K-means algorithm** heuristic algorithm, not optimal, converge to local optimal, works quite well in most cases. run several times (with a random starting point) and then the best solution is selected(the solution with the smallest within-cluster scatter).
Kmedoids medoid of a set of points is the point with the minimal average dissimilarity (distance) to all other points in the set. **partitioning around medoids (PAM)** that tries to improve a clustering locally by swapping medoids with other data points

$Q \leftarrow \sum_j \sum_{\mathbf{x}\in D_j} \text{Dis}(\mathbf{x}, \boldsymbol{\mu}_j)$;
for each medoid **m** and each non-medoid **o** **do**
  calculate the improvement in Q resulting from swapping **m** with **o**;
end
select the pair with maximum improvement and swap;

PCA dimensionality reduction, finding the intrinsic linear structure in the data. eigenvalues can give clues to the inherent dimensionality of the data.

10 Model ensembles

are "meta" methods. Construct different models from adapted versions of the training data, Combine the predictions by averaging voting, weighted voting. a strong classifier from a set of weak classifiers. **Pros** improve performance, avoid over-fitting.
Bagging bootstrap aggregation. T models on different samples. Samples are training data with replacement choosen with uniform probability, called bootstrap sample. Decision boundary is *piecewise linear*.(useful for tree model). **Subspace sampling** Build each decision tree from a different random subset of the features.
$Bagging + subspacesampling = randomforestsmethod.$
Boosting create diverse training sets, add classifiers that do better on the misclassifications from earlier classifiers, by giving them a higher weight(less susceptible to overfitting).Can be extended to multi-class classification. **Adaboost**(adaptive boosting), As long as the performance of each classifier is better than chance$\epsilon < 0.5$, it is guaranteed to converge to a better classifier.**Procedure**1. Train a classifier assign it a confidence factor based on the error rate 2. Give misclassified instances a higher weight. 3. Repeat for T classifiers 4. The ensemble predictor is a weighted average of the models (rather than majority vote) 5. Threshold for binary output.
Boosting1 vs bagging2 •1 Can achieve zero training error by focusing on the misclassifications. A bias reduction technique (increase accuracy) •2 With relatively large bootstrap sample sets, there tends to be little diversity in the learned models. A variance-reduction technique (increase consistency).

11 ML experiments

machine learning experiments pose questions about models that we try to answer by means of measurements on data
ML experiments confirm the various assumptions in a ML problem, establish realistic expectations for performance. **Questions** specific model perform on data from D? a set of models has the best performance on D? models from learning algorithm A perform on D? learning model A produces the best models for domain D?
Performance measure accuracy assume the same class distribution of read and test. average recall assume real problem is uniform class distribution.
the combination of precision and recall, and therefore the **F-measure**, is insensitive to the number of true negatives
predicted positive rate=pos x tpr + (1âĹŠpos) x fpr
Performance estimation(testing), Holdout Method(leave $\sim$ 30%), cross-validation, leave-one-out.
Cross-validation reduce the sample variance by $1/\sqrt{k}$. large data sets, fewer folds. For sparse, leave-one-out best. If $var > acceptable\ var$, we need more data. If statisfied, run over the entire data set produce the final model.
stratified cross-validation If we expect the learning algorithm to be sensitive to the class distribution.
Testing Applied after the model is finalized.

12 Neural network

When to use input high dim. target function unknown. long training time. human readability. good for complex pattern recognition.
RNN Directed cycles exist.
GD search *hypothesis space* to find w for minimum error(slow, local minimum). *Variation* incremental GD, SGD.
BP minimize the squared error, iteratively propagate errors backwards from output units.$\Delta w_{ji} = \eta\delta_j z_{ji}$. **inductive bias** smooth interpolation between data points.
termination condition number of iterations, predetermined error threshold. **Overfitting** every iteration measure error on the validation set(cross-validation approach).
RL(Reinforce learning) agent learns behavior through trial-and-error interactions with a dynamic environment based on a reward signal(Game playing). **Rewards** from sequences of actions, can be separated temporally. **Q-learning** finds optimal action-selection policy for Markov Decision Process, learns an action-value function that gives the expected utility(convergence slow).
Multi-label classification class labels are not mutually exclusive.feature-to-class mapping, dependence between classes are learned.(blog post tags)
Transfer learning(inductive transfer) apply knowledge to a different but related problem(walk to run)
Online (incremental) learning data available sequentially over time. updates the model each time a new data point arrives(visual tracking)
Active learning semi-supervised learning, the algorithm queries information source(the user) to obtain the desired model(robotics, where to point the camera or sensors to gain useful information)
DL learn many-layered NN, more abstract features. learn good representations of the data through unsupervised learning. **Cons** complex, slow, hard to explain.