Machine Learning

Unit 1

* Well Posed problems

→ machine learning is a multidisciplinary field that draws on results from artificial intelligence, probability and statistics, computational complexity theory, control theory, information theory, philosophy, psychology and other fields.

→ Definition of learning - A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

→ In general, to have a well-defined learning problem, we must identify these three features: the class of tasks, the measure of performance to be improved, and source of experience.
→ Eg checkers problem, robot driving and handwriting recognition

* The machine learning process

1. Data collection and preparation - ★
2. Feature selection - As well as identifications of features that are useful for the learner, it is also necessary that the features can be collected without significant expense or time, and that they are robust to noise and other corruption of data that may arise in the collection process.
3. Algorithm choice
4. Parameter and model selection - for many of the algorithms there are parameters that have to be set manually, or that require experimentation to identify appropriate values.
5. Training - Given the dataset, algorithm and parameters, training should simply the use of computational resources to build model of the data in order to predict outputs on new data
6. Evaluation - Before a system can be deployed it needs to be tested and evaluated for accuracy on data that it was not trained on.

* machine learning algorithms need significant amounts of data, preferably without too much noise, but with increased dataset size comes increase computational costs

* Types of machine learning system

Supervised learning

A training set of examples with correct responses (targets) is provided, and based on this training set, the algorithm generalises to respond correctly to all possible inputs. This is also called learning from exemplars

Unsupervised learning

The algorithm tries to identify similarities between the inputs that have something in common are categorised together. The statistical approach to unsupervised learning is known as density estimation.

Reinforcement learning

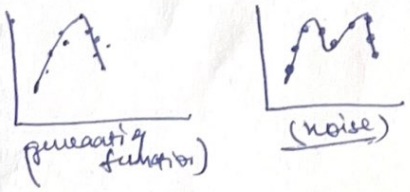
The algorithm gets told when the answer is wrong, but not told how to correct it. It has to explore and try out different possibilities until it works out how to get the answer right. It is sometimes called learning with a critic because of this monitor that scores the answer, but does not suggest improvements

Evolutionary learning

Biological evolution can be seen as a learning process; biological organisms adapt to improve their survival rates and chance of having offspring in their environment we can model this in a computer, using an idea of fitness, which corresponds to a score for how good the current sol. is.

* Testing ML Algorithms

* overfitting - If we are training for too long, then we will overfit the data, which means that we have learnt about the noise and inaccuracies in the data as well as the actual function. Therefore, the model that we learn will be too much complicated, and won't be able to generalise.



We want to stop the learning process before the algorithm overfits, which means that we need to know how well it is generalising at each timestamp. We can't use the training data for this because we will not be able to detect overfitting, but we can't use the testing data either because we are saving it for the final tests. So we need a third set of data to use for this purpose, which is called the validation set because we are using it to validate the learning so far. This is also known as cross-validation in statistics. It is a part of model selection.

* Training, Testing and validation sets

3 sets of data:
① training set to actually train the algorithm
② validation set to keep track of how well it is doing as it learns
③ test set to produce final results.

The semi-supervised learning attempts to deal with this need for large amounts of labelled data.
Each algorithm is going to need some reasonable amount of data to learn from (precise needs vary, the more the data the algorithm sees, the more likely it is to have seen examples of each possible type of input, although it also increases the computational time). However the same argument can be used to argue that validation and test sets should also be reasonably large.
Generally, the exact portion of training : testing : validation is upto you, but it is typical to use 50:25:25 when sufficient data is available, and 60:20:20 otherwise.

In order to split the data, it is advised that randomly reordering the data is done first, or by assign each datapoint randomly to one of the sets.

If you are short of training data, so that if you have a seperate validation set there is a worry that algo will not be sufficiently trained; then it is possible to perform leave-some-out, multi-fold cross-validation. dataset is randomly partitioned into k subsets and one subset is used as validation while algo trains on the others all the others. A different is then left out and a new model is trained on that subset, repeating the process on all subsets. Finally the model with lowest validation error is tested and used. We've traded off data for computation time. In the most extreme case of this we is leave-one-out cross-validation, where the algo is validated on just one piece of data, training on all the rest.

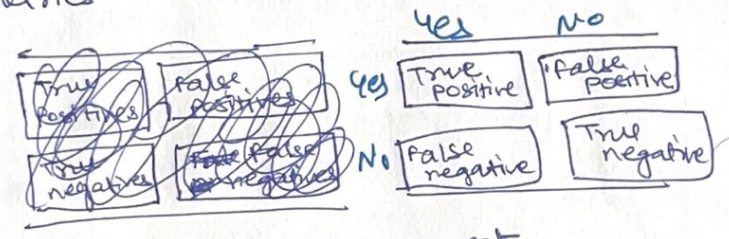
* confusion matrix - It is a method, suitable for classification problems, to work out whether or not the result is good make a square matrix that contains all possible classes in both horizontal and vertical directions and list the classes along the top of the table as predicted outputs and then down to the left-hand side as the targets

The element (i, j) tells how many input parameters were put into class i in targets but class j in algorithm. Anything on the leading diagonal is the correct answer

outputs	C_1	C_2	C_3
C_1	5	1	4
C_2	0	4	2
C_3	0	0	4

Here accuracy is the sum of elements on the leading diagnol divided by the sum of all elements in the matrix.

* Accuracy metrics



The entries on the leading all correct
Accuracy is sum of no. of TP and TN divided by sum of no. of examples total
(Based on binary classifications)

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

$$\text{sensitivity} = \frac{\#TP}{\#TP + \#FN}$$

$$\text{Specificity} = \frac{\#TN}{\#TN + \#FP}$$

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

$$\text{Recall} = \frac{\#TP}{\#TP + \#FN}$$

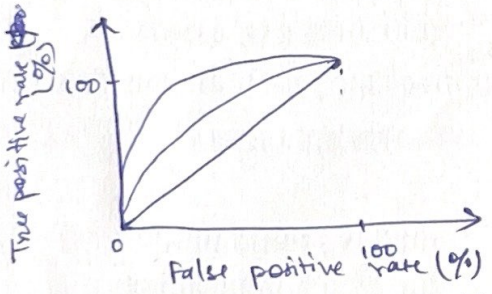
Two complement pairs to interpret the performance of a classifier, namely sensitivity and specificity, and precision and recall.

$$f_1 = 2 \frac{\text{Precision} \times \text{recall}}{\text{Precision} + \text{Recall}}$$

$$\text{or} \quad f_1 = \frac{\#TP}{\#TP + (\#FN + \#FP)/2}$$

* Receiver operator characteristic (ROC) curve

We can compare classifiers either with the same classifier with different learning parameters, or completely different classifiers. In this case, ROC is useful.
This is a plot of the percentage of true positives on y axis against false positives on the x axis.
A perfect classifier would be at point (0,1), while the anti-classifier that got everything wrong would be at (1,0).
Any classifier that sits on the diagnol line from (0,0) to (1,1) behaves exactly at the chance (assuming that the positive and negative classes are equally common)



Anything above the line is better than chance, and the further from the line the better.

In order to compare classifiers, or choices of parameters settings for the same classifier, you could just compare the point that is furthest from the chance line on the diagnol. However it is normal to compute the Area under curve (AUC) instead.
The key to getting a curve rather than a point on ROC curve is to use cross-validation. If you use 10-fold cross-validation, then you have 10 diff classes, with 10 diff test sets and you also have the ground truth tables.
The truth tables can be used to produce a ranked list of diff cross-validation-trained results which can be used to specify a curve through the 10 points on ROC that correspond to results of this classifier.

By producing Roc for each classifier it is possible to comp the results

* Unbalanced datasets

Balanced dataset → when there are same no. of positive & negative examples in dataset, otherwise unbalanced dataset

$$\text{Balanced accuracy} = \frac{\text{sensitivity} + \text{specificity}}{2}$$

Matthew's correlation coefficient, MCC

$$= \frac{\#TP \times \#TN - \#FP \times \#FN}{\sqrt{(\#TP + \#FP)(\#TP + \#FN)(\#TN + \#FP)(\#TN + \#FN)}}$$

If any of the brackets in denominator are 0, then the whole denominator is set to 1. This provides balanced accuracy computation.

In case of more than 2 classes, the calculations become too complex, however, it is possible that you use one class as the positives and everything else as negatives, and repeat this for each of different classes.

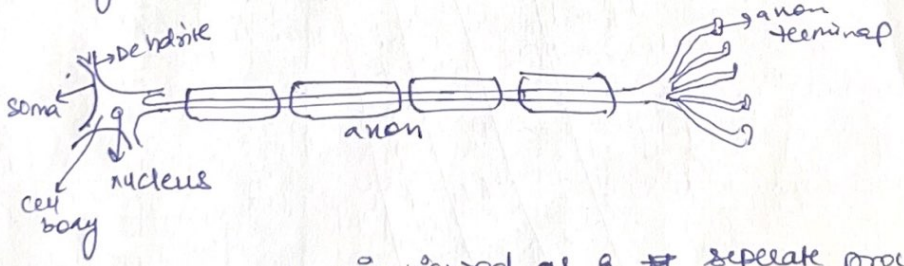
* Measurement precision

If we feed in a set of similar outputs in algorithm, then we should expect similar outputs. This measure of variability of the algorithm is also known as precision and it tells how repeatable the predictions that algo. makes are. Just because an algo is precise, does not mean it is accurate; it may be precisely wrong. One measure of how well the algo's predictions match reality is known as trueness, and it can be defined as the avg distance b/w correct output and the prediction.

Unit 2

* The Brain and the neuron

Neurons are the processing units of the brain. There are 100 billion (10^{11}) neurons and are of different types depending upon their task. However, their general operation is similar in all cases: transmitter chemicals within the fluid of the brain raise or lower the electrical potential inside the body of the neuron. If this membrane potential reaches a certain threshold, the neuron spikes or fires and a pulse of fixed strength and duration is sent down the axon. The axons divide (arborise) into connections to many other neurons, connecting to each of these neurons in a synapse. Each neuron is typically connected to thousands of other neurons, estimated 100 trillion = 10^{14} synapses in the brain. After firing, the neuron must wait for some time to recover its energy (the refractory period) before it can fire again.



Each neuron can be ~~the~~ viewed as a ~~the~~ seperate processor performing a simple ~~computed~~ computation: deciding whether or not to fire.

How does learning occur? The principal concept is plasticity: modifying the strength of synaptic connections between neuron and creating new connections

* Neural networks ommarde in notes.

* PCA steps:
  1. Normalize data
  2. Covariance matrix $cov(x,y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N-1} = \frac{\sum x_i y_i}{N-1}$
  3. Calculate eigen values and eigen vectors
  4. Order the eigen values from largest to ~~smallest~~
  5. Form feature vector
  6. Form principle components

Unit 3

* Decision tree:

Supervised learning algo used to build classification and regression models

A decision tree is a tree where each -

① Node - a feature (attribute)
② Branch - a decision (rule)
③ leaf - an outcome (categorical or continuous)

* ID3 (Iterative Dichotomiser 3):

Entropy, $H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$

Information gain, $IG(A,S) = H(S) - \sum_{t \in T} p(t) H(t)$

Steps: ① Calculate entropy for dataset
② for each attribute/feature
- calculate entropy for all its categorical values
- calculate IG for ~~each~~ feature
③ Find feature with maximum information
④ Repeat until we get the desired tree

for continuous attributes, we need to create a new boolean value that is true when humidity $\leq c$ and false otherwise.
Calculate best threshold of c: ① sort examples
② value of c that maximises IG must lie at changing boundaries
③ Find IG for all such values
④ Select best attribute.

$H(\text{value}) = ?$
$H(\text{feature}) = \sum p(\text{value}) H(\text{value})$ for all values of feature
$IG = H(S) - H(\text{feature})$

Characteristics of IG:
① uses greedy approach; can get stuck in local optima
② can overfit to training data
③ not always smallest tree
④ harder to use on continuous data.

* CART (Classification & Regression trees):
- is a decision tree methodology
- umbrella term for following types of DT: ① classification trees ② Regression trees

In classification:
- Gini index is a metric for classification tasks in CART. It stores sum of squared probabilities of each class
$Gini = 1 - \sum (P_i)^2$ for $i = 1$ to no. of classes

steps: ① calculate gini index for each feature
② select feature with lowest gini index
$G(\text{feature}) = \sum p(\text{value}) H(\text{value})$

Unit 4
Clustering

* measuring sim. & dissimilar.

① Distances:

→ Manhattan distance,
$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots$

→ Euclidean:
$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + \dots}$

② Binary vectors:

m_{01} = the number of attributes with p 0 and q was 1

m_{11} =

m_{10} =

m_{00} =

Simple matching coefficient

$$SMC = \frac{m_{11} + m_{00}}{m_{11} + m_{00} + m_{10} + m_{01}}$$

~~Jacquard~~
Jaccard coefficient,

$$J = \frac{m_{11}}{m_{01} + m_{10} + m_{11}}$$

③ Cosine similarity:
If d_1 and d_2 are two document vectors, then

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

$$\|d_1\| = \sqrt{\sum x_{i,d_1}}$$

* Types of clustering

- Partitioning approach
  1. K means
  2. K-medoids
  3. CLARANS
- Hierarchical approach
  1. Diana (Divisive)
  2. ~~Agnes~~ Agnes (Agglomerative)
  3. BIRCH
  4. ROCK
  5. CAMELEON
- Density-based approach
  1. DBSACN
  2. OPTICS
  3. Denclue

* Centroid, Radius & Diameter of cluster

Centroid (middle of circle), $Cm = \frac{\sum_{i=1}^{N} (t_{ip})}{N}$

Radius (Sq. root of avg dist from any point to centroid), $Rm = \sqrt{\frac{\sum_{i=1}^{N} (t_{ip} - C_m)}{N}}$

Diameter, $Dm = \sqrt{\frac{\sum_{i=1}^{N} \sum_{i=1}^{N} (t_{ip} - t_{iq})^2}{N(N-1)}}$

* k-means clustering: Partition clustering approach

Pseudo code:

1. select k points as initial centroids
2. repeat
3. form k clusters by assigning all data points to the closest centroid
4. Recompute the centroid of each cluster
5. until centroids dont change

⇒ Initial centroids chosen at random
⇒ closeness is measured by Euclidean dist, cosine similarity, correlation, etc.
⇒ often stopping condition changed to "until relatively few points change clusters"
⇒ complexity is $O(n * k * I * d)$
(n: num of points, k: no. of clusters, I: no. of iterations, d: no. of attributes)
⇒ Relatively ~~less~~ strength $O(I k n)$ than PAM and CLARA
⇒ often terminates at local optima. The global optima may be found using deterministic annealing and genetic algorithms

Disadv
① Need to specify k in advance
② unable to handle noisy data and outliers
③ not suitable for clusters with non-convex shapes
④ has problems when clusters are of differing sizes, densities and non-globular shapes

Evaluating k-means:

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} dist^2(m_i, x)$$

sum of squared errors

one easy way to reduce SSE is to increase k

* Hierarchical:
→ uses distance matrix as clustering criteria
→ method does not need no. of clusters as input, but needs a termination condition
→ represented in tree like structure, called dendogram

Types → Agglomerative - known as bottom up approach
consider it as bringing things together
↳ Decisive - (top down approach) we take a large cluster and divide it into two, three, four or more clusters

~~Algo~~

⇒ Agglomerative

Stoping criteria:

① Pick several clusters(k) upfront
② Stop when the next merge will create a cluster with low cohesion.
③ a. Diameter of cluster
b. Radius
...

⇒ Decisive:

* more efficient than agglomerative clustering especially when all the way down to individual leaves. there is no need to generate a complete hierarchy
* can be considered as global approach, since it contains the complete info. before splitting data.

Issues in Decisive:

① Splitting criteria - Ward's kmeans square error is used well (SSE)

The greater reduction obtained in difference in SSE should reflect goodness of split.

Gini Index ~~food~~ for handling nominal data

② Splitting method - bisecting k-means approach

③ Choosing cluster to split - by checking the square errors of clusters and splitting the one with largest value.

* K-means vs Heirarchical

① Heirarchical cannot handle big data, k-means can (because order of kmeans is linear $O(n)$, while that of hierarchical is $O(n^2)$)

② In k-means, since we start with random choice of clusters, the results produced might differ. ~~result~~ while results in hierarchical are reproducible

③ k-means works well with hyper spherical shaped clusters

④ kmeans require prior knowledge of k. but not in hierarchical

* Partitioning method

This clustering method classifies the information into multiple groups based on characteristics and similarity. In partitioning methods, when database(D) that contains multiple (N) objects then the partitioning method constructs user-specified (k) partitions of data in which each partition represents a cluster and a particular region

Eg: ① kmeans
② PAM (k-medoids)
③ CLARA (Clustering large Applications)

PAM: ↦ find representative objects, called medoids, in clusters
built in 1990s by Kaufman and Rousseeuw

Pseudo code:
① select k representative objects arbitrarily
② For each pair of non-selected object h and selected object i, calculate the swapping cost TCih
③ For each pair of i and h,
- If TCih < 0, i is replaced by h
- then assign each non-selected object to the most similar representative object
④ Repeat step ~~2~~ 2-3 until there is no change

* Total swapping cost, $TC_{ih} = \sum_j C_{jih}$

$$C_{jih} = d(j,h) - d(j,i)$$

* PAM is more robust than k-means in the presence of noise and outliers because medoids are less influenced by outliers
* works efficiently for small datasets but does not scale well for large datasets

$$O(k(n-k)^2)$$

CLARA: It draws multiple samples of data, applies PAM on each sample and gives best clustering as output.
strength: deals with large datasets
weakness: efficiency depends on sample size
A good clustering on samples does not ensure good clustering of the whole dataset if the sample is biased.

* Distribution-based methods: It is a clustering model in which we will fit the data on the probability that how it may belong to the same distribution - the grouping of data may be normal or gaussian. Gaussian is more prominent where we have a fixed no. of distributions and all the incoming data is fitted into it such that the distribution of data may get maximised.
Eg: expectation-maximization uses gaussian distribution, multivariate normal distribution
This model works well on synthetic data and diversely sized clusters.
Problems: ① if the constraints are not used to limit the model's complexity.
② produces clusters that assume concisely defined mathematical models underlying the data, rather strong assumption for data distribution

* Density based
  * based on density (local cluster criteria), such as density connected points.

* major features
  ① Discover clusters of arbitrary shape
  ② Handle noise
  ③ One scan
  ④ Need density parameters as termination condition

* Two parameters:
  ① Eps : max. radius of neighbourhood
  ② MinPts : min. no. of points in an Eps neighborhood of that point

* $N_{Eps}(p) : \{ q \text{ belongs to } D \mid dist(p,q) \leq Eps \}$

* Directly density-reachable: A pt is directly density reachable from point q wrt Eps, MinPts if
  ① p belongs to $N_{Eps}(q)$
  ② core point condition:
  $|N_{Eps}(q)| \geq MinPts$

* Density reachable : A point p is density reachable from a point q wrt Eps, MinPts if there is a chain of points $p_1, \ldots p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density reachable from p_i

* Density connected : A point p is density-connected to a point q wrt Eps, MinPts if there is a point o such that p and q are density-reachable from o wrt Eps, MinPts.

* DBSCAN (Density based Spatial clustering of Applications of Noise):
  → Relies on a density based notion of cluster: A cluster is defined as a maximal set of density-connected points
  → Discovers clusters of arbitrary shape in spatial databases with noise.

Algorithm: ① Arbitrary select a point p
② Retrieve all points density-reachable from p wrt Eps and MinPts
③ If p is a core point, a cluster is formed
④ If p is a border pt., no points are density reachable from p, and DBSCAN visits next point of database.
⑤ Continue the process until all of the points have been processed

* Fuzzy clustering

It is a type of clustering algorithm in machine learning that allows a data point to belong to more than one cluster with different degrees of membership. (between 0 and 1)

Applications: ① Img segmentation, pattern recognition, risk assessment, medical diagnosis.

Advantages ① Flexibility
② Robustness
③ Interpretability

Disadv: ① complexity
② model selection

Algorithm: ① Initialize the data points into the desired no. of clusters randomly
② Find out the centroid
③ Find distance of each pt from centroid
④ Update membership values
⑤ Repeat steps 2-4 until constant values are obtained for membership values or the difference is less than a tolerance value
⑥ Defuzzify the obtained membership values.

centroid formula, $V_{ij} = \left(\sum_1^n \gamma_{ik}^m x_k\right) / \sum_1^n \gamma_{ik}^m$

m is fuzziness parameter

updating membership values,

$$\gamma = \left[\sum_1^n \left(d_{ki}^2 / d_{kj}^2\right)^{\frac{1}{m-1}}\right]^{-1}$$

* Evaluation of unsupervised learning models

2 classes of techniques to validate results → Internal validation
→ External validation

* Internal validation revolves around two metrics:

① cohesion within each cluster

$$\text{cohesion}(C_k) = \sum_{x \in C_k, y \in C_k} \text{similarity}(x, y)$$

② Seperation b/w different clusters

$$\text{seperation}(C_j, C_k) = \sum_{x \in C_j,\ y \in C_k} \text{distance}(x, y)$$

A set of clusters with high cohesion & high seperation are considered good.

Some measures that combine both of the above into a single measure:

① Silhouette coefficient
② Calinski - Harabasz coeff.
③ Dunn index
④ Xie - Beni score
⑤ Hartigan index.

* External validation - This can be carried out if true cluster labels are available. The idea is to measure the statistical similarity to the true cluster set.

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1 factor} = \frac{\text{Precision} \times \text{Recall} \times 2}{\text{Precision} + \text{Recall}}$$

TP = No. of pairs of records which are in same clusters in both P and S

FP = No. of pairs which are in same cluster in S but not in P

* Twin sample validation: How to validate the results of our unsup. learn. model in absence of true cluster labels. This step takes it as a given that we have already performed clustering on our training data and now we want to validate the results.

Steps: ① create twin sample of training data
② Performing unsup. learn. on twin sample
③ Importing results for twin sample from training set
④ Calculating similarity b/w two sets of results