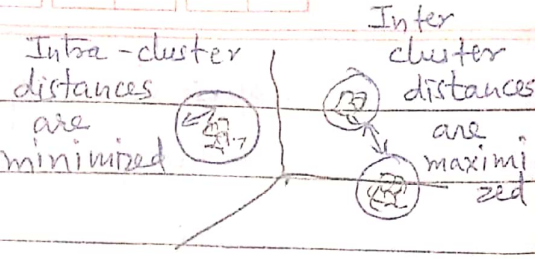
Page Date

# Clustering




A grouping of data objects, such that the objects within a group are similar (or related) to one another and different from (or unrelated) to the objects in other groups

→ It is a classic case of unsupervised learning → as there is no right answer, no $y = f(x)$ situation.

If I have 1 million records and say 100 columns, it becomes very difficult to describe the data or in other words, getting coherent pieces / homogeneous groups. So purpose of clustering is to find homogeneous groups which are close to each other and can help us in describing the data.

→ It is an important step for preprocessing of data before running models

→ Also helps in identifying outliers

**Outliers** — objects that do not belong to any cluster or forms clusters of very small cardinality

Outlier analysis — when we are interested in discovering outliers (interesting cases) for some study, not clusters

Clustering results are used:

- As a standalone tool to get insight into data distribution
  - visualization of clusters may unveil important information
- As a preprocessing step for other algorithms
  - Efficient indexing or comparision often relies on clustering

Page Date

## Applications of clustering

→ Image processing - cluster images based on their visual content

→ document clustering - where documents are clustered based on various words, length of docs, themes, topics, to b create better user experiences, if this person reads this book is considered to read another book also

→ Ex, segmenting for efficient marketing of cars based on a survey, on a scale of 1 to 10, we analyze the attitude of driver towards driving and come up with 3 grops to target
- speed oriented
- Comfort oriented
- Safety oriented

Supervised (there is right answer) → classification - you begin with known classes
eg. will you enroll in this program or not?

unsupervised (there is no right answer) → clustering - you do not know any classes beforehand
eg. what can be the target programs based on needs
classification is kind of individuals problem, clustering is more of group problem

### Attributes (dimensions)

→ Real value attributes / variables / Number where we can do all sort of mathematical functions, eg. Salary, height

→ Binary attribute - Yes / No, Male / Female, True / false

→ Nominal attribute - Categorical - extension of binary attribute (Default / Not default)
eg. religion (christian, Muslim, Buddist, Hindu etc)

→ Ordinal / Ranked attributes, eg. Military rank (Soldier, sergeant, lieutenant, Captain etc)

Page Date

Clustering algorithm works best when all variables are real valued.

→ To know whether 2 rows / people / object are close to each other or not, we need to measure difference b/w them. 2 important conditions are

* distance b/w 2 rows cant be -ve
* distance b/w 2 identical rows is 0 (say Row1 and Row1)

Row 1 - - - -
Row 2 - - - -

The distance $d(x,y)$ between two objects $x$ and $y$ is a metric if

- $d(i,j) \geq 0$ (non-negativity)
- $d(i,i) = 0$ (isolation)
- $d(i,j) = d(j,i)$ (symmetry)
- $d(i,j) \leq d(i,h) + d(h,j)$ (triangular inequality)

distance is measured differently for real, boolean, categorical and ordinal - for ex, distance for Yes/No/categorical variable can not be measured like real and it becomes more like a compromise

Sometimes different weights may be associated with different variables based on applications and data semantic for ex. if age is more significant than income, we can give it weightage

fundamental matrix for clustering

Data Matrix

attributes / dimensions

| tuples / objects | | | | |
|---|---|---|---|---|
| | $x_{11}$ | $x_{12}$ | - - - | $x_{1d}$ |
| | : | | | |
| | : | | | |
| | $x_{n1}$ | $x_{n2}$ | | $x_{nd}$ |

Distance Matrix

objects

| objects | | | | |
|---|---|---|---|---|
| | 0 | | | |
| | $d(2,1)$ | 0 | | |
| | $d(3,1)$ | $d(3,2)$ | 0 | |
| | | | | 0 |
| | $d(n,1)$ | $d(n,2)$ | | |

for FA/PCA, fundamental input is correlation matrix

Page | Date

Dice coefficient is also used

## Distance functions for Binary vectors

eg. to find groups (clusters) of matching pairs, series of binary questions are asked (response in form of 0 or 1)

Jackard Similarity b/w binary vectors X and Y

$$JSim(X, Y) = \frac{X \cap Y}{X \cup Y}$$

many a times 1 and 0 does not give equal ~~also~~ amount of information like do you like football — 1 means clear info but 0 does not give clear info. If not football then what. However, for male/female 1 and 0 gives equal information.

→ Hence 0 is not considered in Jaccard calculation

Jaccard distance b/w binary vectors X and Y

$$Jdist(X, Y) = 1 - JSim(X, Y)$$

| | $Q_1$ | $Q_2$ | $Q_3$ | $Q_4$ | $Q_5$ | $Q_6$ |
|---|---|---|---|---|---|---|
| X | 1 | 0 | 0 | 1 | 1 | 1 |
| Y | 0 | 1 | 1 | 0 | 1 | 0 |

JSim = 1/6 (Both 1)

Jdist = 5/6 (At Least 1)

→ Do not consider double 0's

## Distance function for real valued vector

Minkowski distance — Lp norms — generic.

$$L_p(x, y) = \left[|x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_d - y_d|^p\right]^{1/p} = \left[\sum_{i=1}^{d} |x_i - y_i|^p\right]^{1/p}$$

p = +ve integer

d = no of columns/variables

$x_1, y_1$ & $x_2, y_2$ are corresponding values of variables in row x and y.

| (Root) | column1 | column2 | ... | columnd |
|---|---|---|---|---|
| x | $x_1$ | $x_2$ | ... | $x_d$ |
| y | $y_1$ | | | |

* Binary variables are not always symmetric i.e 1 & 0 might not give always similar information.

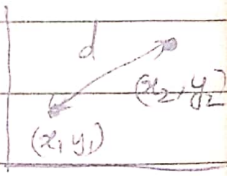
Page . . Date

Manhattan (or city block) distance — if $p=1$, $L_1$

$$L_1(x,y) = |x_1-y_1| + |x_2-y_2| + \cdots + |x_d-y_d|$$

Euclidean distance — if $p=2$, $L_2$

$$L_2(x,y) = \sqrt{|x_1-y_1|^2 + |x_2-y_2|^2 + \cdots + |x_d-y_d|^2}$$




distance b/w 2 points = $d = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$

Typically in clustering we use euclidean distance. Sometimes we use manhattan distance.

* For different versions of models (to have maximum efficiency) we use different values of $p$ and see which one works best

Weightage distance —

$$d(x,y) = \sqrt{(w_1|x_1-y_1|^2 + w_2|x_2-y_2|^2 + \cdots + w_d|x_d-y_d|^2)}$$

* In general we will have mixed variables (binary as well as real). 2 ways of clustering
  - Hierarical clustering
    - Single linkage
    - Complete linkage — ward's procedure
    - Advanced linkage
  - K-means clustering (partitioning clustering)

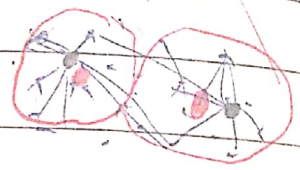
R package "cluster"

Page Date

## K-Means

K stands for how many clusters so if K=3, 3 means clustering, Given a set of X of n points in a d-dimensional space and an integer K.

Generally 2 to 7 is a healthy number of clusters, you start with an arbitrary number and eventually discover how many clusters are good enough so that overall distance is minimized.



→ say for all these points we decide 2 clusters
→ Random centers are picked and distance of each point is calculated from that center (euclidean or jacard)
→ Based on minimum distances, 2 clusters are formed.
→ New center for each cluster is chosen based on average of all points in that cluster. (Centroid)
→ Repeat the above process till we don't see any improvement in variation reduction i.e. Convergence (no further reduction in distance)

| | Age | Income |
|---|---|---|
| | 45 | 60 |
| | 35 | 50 |
| average | 40 | 55 → Centroid |

→ we may get clusters of different densities as well as different sizes.
→ Any outliers may force the centroid to change and hop in hop-out of outliers can happen.

for mixed data type we use Gowar distance instead of euclidean, otherwise by default it is Euclidean.

→ Inter cluster > Intra cluster

→ between sum of square should be greater than Within sum of squares - to make groups different

$$\text{Total SS} = \sum_{i=1}^{1000} (age_i - \overline{age})^2 + (inc_i - \overline{inc})^2 + (wt_i - \overline{wt})^2$$ for entire sample

Within SS = Same formula as above, however for specific clusters, So if 3 clusters, 3 within-SS (of 3 samples)

Between SS (for 3 clusters of 2 variables) $= n_1(\overline{age}_1 - \overline{age})^2 + n_1(\overline{wt}_1 - \overline{wt})^2 + n_2(\overline{age}_2 - \overline{age})^2 + n_2(\overline{wt}_2 - \overline{wt})^2 + n_3(\overline{age}_3 - \overline{age})^2 + n_3(\overline{wt}_3 - \overline{wt})^2$

Generally $\frac{\text{B/w SS}}{\text{Total SS}} > 70\%$ is a good number

Out of total variability of 100%, I am able to explain/attribute by 57% to 3 groups. So I am able to reduce the variability by 43%

Scaling of variables — In case one variable is significantly big that it would overshadow other variables, you need to scale, normalize, standardize the data

Height, wt, Income example

Scaled average is on −3 to +3 scale

→ Other normalization technique — Max-Min normalization - also called Range norm.

→ Log transformation (though it is complicated)

Anova has to be done on raw/unscaled dataset and not on changed/scaled/weighted dataset.

Page Date

→ We can give weight to a particular variable. Like d1 $wt ← 2 * d1 $wt, say you are promoting a weight loss drink and you have to emphasize on weight by giving specific multiplier/weight. It is kind of opposite of scaling.

→ Outliers on individual variables can be ignored.
→ However, if entire cluster is outlier (eg a cluter of 5 elements very different from other clusters), then we need to study it.
→ Outlier analysis is not univariate typically, has to be multivariate so that whole row becomes outlier

→ In K means, it is possible to give your own center point/centroid instead of allowing K-means algorithm to have random start (random center). However, it is a tedious task.
If 3 groups with 20 variables, we will have to key in centroid in the dataset by 60 times (20×3).

# Hierarical Clustering

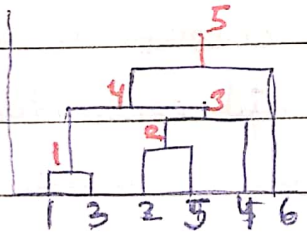
K-means is used when we want to cluster quickly, we are not bothered about small shift in random center, also we have a huge data set to work with

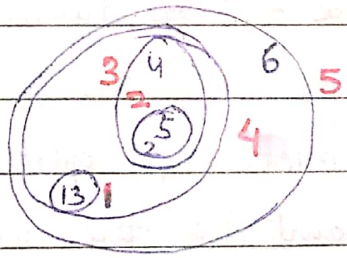
But many a times you want to know how clusters are forming, at what point they are merging, how story is unfolding, so that your clarity about clustering groups is better, you do hierarical clustering.

→ Produces a set of nested clusters organized as a hirerical tree

→ Can be visualized as a dendogram (a tree like diagram that records the sequence of merges or splits)

→ Each row is treated as 1 cluster, then based on distances b/w each row, rows are merged, and at the top becoming one cluster




Unlike K means, we know here that 1,3 joined, 2,5 joined etc etc and finally reached to one cluster, So we have whole story unfolded in front of us. We can cut the tree based on logical visualization and come up with a quantitative number of clusters

The higher the join, more distance rows have b/w them -

Page Date

## Strengths of hierarical clustering

→ No assumptions on the number of clusters. Any desired number of clusters can be obtained by cutting the dendogram at proper level (we can actually see that due to strong commonality, which news are joining)

→ Hierarical clustering may correspond to meaningful taxonomies In K-means, we start with random # of clusters with no clarity what exactly is going in it.

## Two main types of hierarical clustering

- **Agglomerative** - Bottom up - Mostly used/Popular
  - Starts with the points as individual clusters
  - At each step, merge the closest pair of clusters until only one cluster (or K cluster left)

- **Divisive** - Top down
  - starts with one, all inclusive cluster
  - At each step, split a cluster until each cluster contains a point (or there are K clusters)

Traditional hierarical algorithms use a similarity or distance matrix

- Merge or split one cluster at a time based on distance.

Page Date

## Agglomerative clustering algorithm

Suppose there are 10 rows, we can find $^{10}C_2$ combination of distance b/w rows. We first join 2 rows which are minimum distance and form a temp cluster. Then we either join 2 more rows with min distance or join another row with temp cluster based on distance

→ Compute distance matrix b/w the input data inputs
→ Let each data point be a cluster
→ Repeat
    Merge two closest cluster
    Update the distance matrix
Until only a single cluster remains

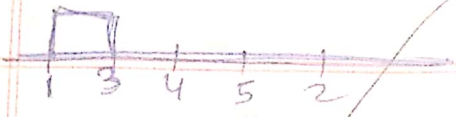
Imp → Key operation is the computation of the distance b/w two clusters. Eg. how to find distance b/w 1, 2 or 1,2 or 2, 4 (refer last page)

→ We have 3 ways to compute distances and work in above situation which leads to different. As distance matrix initially consists of distances b/w actual rows, we are not sure that after joining 1,3, we should join 1,2 first or 1,2 or 2,4 etc. (we dont have distances b/w groups/temp clusters). So 3 linkage method.

1) Complete linkage
2) ~~Simple~~ Single linkage
3) Average linkage

Based on the type of linkage, shape of dendogram changes.

example




All possible combinations

| Points taken | Possible distances (PD) | max of PD Complete | min of PD Single | average of PD Average |
|---|---|---|---|---|
| (1,3), 2 | (7) (1,2) (3,2) (8) | 8 | 7 | 7.5 |
| (1,3), 4 | (1,4) (3,4) | | | |
| 2, 5 | 2,5 | | | |
| 4, 5 | 4,5 | | | |
| 2, 4 | 2,4 | | | |

You will have to use all 3 linkages and see which one is working best for our clustering problem.

→ for dynamic data (new data is coming), we need to recluster after a certain frequency (daily for big data or say monthly) so that centroid changes based on new data. For less amount of new data, we may try to fit it in already existing clusters. For large data, we may need to recluster.

Generally for segmenting, grouping, labelling, clustering data is static (say marketing data which wont change overnight), for dynamic data clustering may not be a very good option (daily changing info)

dist(d2) would calculate distances (euclidean) b/w all rows. Creates distance matrix dt.

hclust (dt, method = "complete") → hcomplete

cutree (hcomplete, K = 4) → c1

we can also plot average of each variable (histogram) cluster wise to analyze our dataset

clustering is done on "Measures" (Numerical Data). Not good for categorical data (Dimensions). For dimensions, "Association Rules" are applied as part of Unsupervised learning.

Page Date

of PD After you cut the tree, you get the cluster # each row belongs to

R-steps

say if d1 is large dataset, we create d2 with 100 rows with age, wt, income variables

→ d2 ← d1[1:100,] , check structure str(d2)

In hierarchical clustering, we created distance matrix

→ dist(d2) → dt - It with create pairwise euclidean matrix and store in dt (fundamental matrix)

→ hclust(dt, method = "complete") → hcom or single/average, It creates a cluster/whole dendogram using selected method

→ plot(hcom) , plot(hcom, hang = -1)

~~Visually~~ Visually we can see that there are 4 groups. 1,2 and 3,4 join at a very later stage so clearly they are seperate groups.

→ For ~~single~~ average hclust(dt, method = "average") → hang, plot(hang)

→ for single hclust(dt, method = "single") → hsin, plot(hsin)

After analyzing above 3 plots, we can see that complete is giving us 4 clear groups, so we would cut the tree with 4 groups.

→ cutree(hcom, K=4) → cl , cl

cl would tells us which row is in which group. So cluster is reduced to a vector with label names.

→ d2$cluster ← cl , here you are adding 'cl' vector to d2 as column cluster, view(cl)

we can now do ANOVA or summary or boxplot etc to analyze various population parameters "cluster-wise"

As in dendogram, you can see which rows joined first so their distance is minimum & they have strong commonality. So here you are able to create complete taxonomy in mind. In K-means you do not know about whole story but just the groups as an output. For the rows which are joining at large distance, you need to study individual variables to determine due to which variable distance is large.

Silo coefficient - Once you are done with group membership and you want to verify whether the membership is correct or if there could be some improvement, you calculate silo coefficient

→ suppose we consider row i.

→ First we find the average distance of row(i) from all members in its group (say row(i) belongs to group 2, find distance of row(i) from all members of group 2, say it is "$a_i$"

→ Then find the average distance of row (i) from other groups/clusters. Say 3 more clusters of $b_i = b_1 + b_2 + b_3 / 3$

→ Now to conclude that row(i) is in correct group

$$a_i < b_i$$

distance within own grp should be less than avg distance of other groups.

→ $$\text{Silo coefficient} = \frac{b_i - a_i}{\max(b_i, a_i)}$$

max distance of $b_i$ and $a_i$

Page Date

Silo coefficient varies b/w -1 to 1

→ It is calculated for each row. So, for rows where silo is +ve, it is good, as for these rows $a_i < b_i$.

→ However, if silo is -ve, it indicates that $a_i > b_i$ so that row is closer to some other group and its better to recluster.

→ In all clustering problems, there would be some -ve silos. Which is OK. However, if more than we need to revisit clusters or need to increase / decrease number of clusters.

In R

→ silhouette (cl, dt) → sl (give input cluster vector and distance matrix)

→ plot (sl)

→ sl — this table shows row's current cluster, its neighbours and silo coff

Ideally if more than 10% rows have -ve silo, we need to revisit. Lot of subjectivity in clustering. Depends on interpretation

→ If there are clusters of very small size, that means those rows are outliers and very different from others.

→ So far we have worked only on 'Real-valued' variables. Now we work on mixed data (Multiple data types)

Page Date

# Mixed data distance computation - Gower distance

→ As we work on categorical, it is a compromise, say distance b/w red and blue car

→ Idea is to use distance measure between 0 and 1 for each variable, i.e. standardize it

→ Aggregate = average distance over variables

→ Binary, nominal - Calculate Jaccard distance

→ Interval-scaled - $\frac{x_{if} - \min(x_{if})}{Rf}$ - Called as Range normalization

$x_{if}$ = value for object i in variable f

$Rf$ = Range of variable f for all objects

say, age = 45 , ~~min range among~~

min age among all ages = 35

max age among all ages = 75

After normalization $age_i = \frac{45-35}{75-35} = 0.25$

→ Ordinal - Use normalized ranks, then like interval-scaled based on range, normalized rank

$$= z_{if} = \frac{r_{if} - 1}{Mf - 1}$$

Say for a survey, allowed ranks are 1, 2, 3, 4, 5

However no one gave 5, but there are some 4. So 5 is not present in your data set. Still we would consider 5 as max rank.

So maximum posible rank is 5 and not 4.
If rank given by a person is 3, then normalized rank = $\frac{3-1}{5-1} = 0.5$

→ Once every variable is brought to 0, 1 scale, compute Euclidean distance.

## In R - Gower Matrix

→ Remove cl column from original d2
d2 ← d2[, -4]

→ adding a column satisfaction score on a scale of 1 to 5
d2 $ ss ← sample(1:5, 100, repl = T), str(d2)

→ Specify right data type for ss as currently showing integer
d2 $ ss ← as.ordered(d2 $ ss)

→ Use this data to calculate Gower matrix.
library(cluster)
daisy(d2, metric = "gower") → dt1
It would compute 100 x 100 matrix, normalize all variables and then calculate euclidean distance.

→ Imp - If using gower, you need not scale any particular variable, in case one variable is significantly higher, As normalization is inbuilt process in Gower, by normalizing everything to 0-1.

→ hclust(dt1, method = "complete") → hcom

→ Rest all is similar as previous.

→ Fuzzy is a technique of classification where membership is not strict but everything is a probability

Page Date

Custer Sampling – a method/kind of clustering where you are creating a cluster which has all characteristics of population.

Page Date

# Decision Tree

Two Kind of problems, one where we already know classifications (y is categorical) and we try to classify various records into (Classification tree)

Another is when y is not categorical, it is called decision tree → Regression Tree

Very popular technique (No. 1)

→ if we are asked to guess a number b/w 1 to 100,

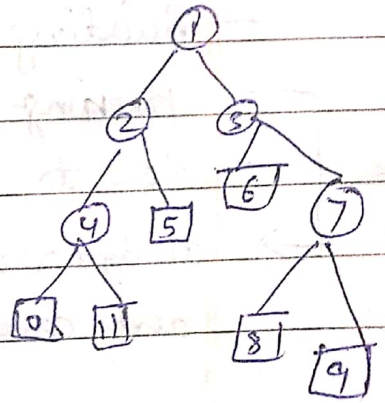
* one way is to call out each number 1 by 1 and reach to the result.
* another is to ask, if more than 50 (No), if more than 25 (Yes), more than 37 — so we should reach the number in 7 hits : DT works on same principle

- DT is an inductive learning task — it works on multiple iterations and learns various rules on its own, which it uses to make more generalized conclusions.
- A predictive modelling based on a series of Boolean test for branching, for ex number > 50 (node only), so every time left/right

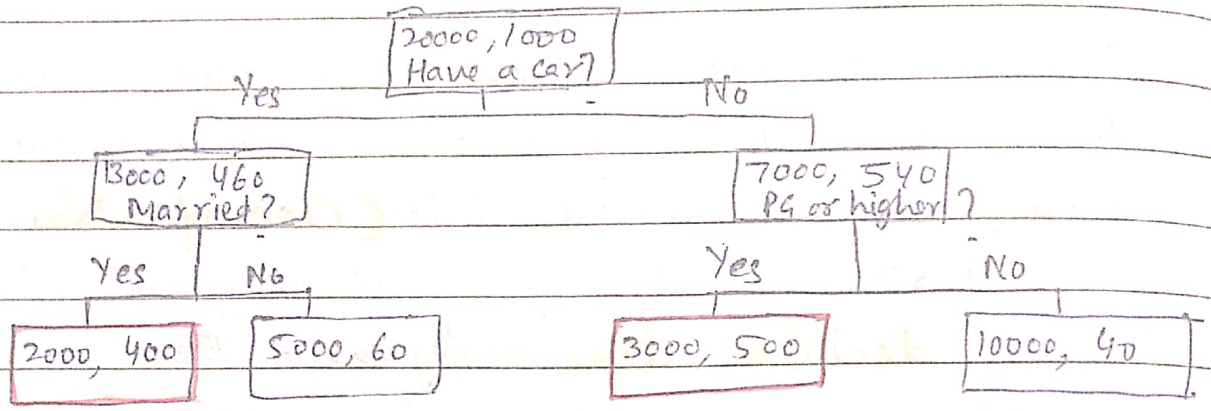
① — root node

□ — terminal nodes which do not have further branching (Leaf node)

○ — Non terminal nodes, have further branches,



2 primary objectives of any model - explanation, Prediction




If we are having an exhibition of cars and we invite 20,000 people and only 1000 people turn up for it. So response rate is only 5%

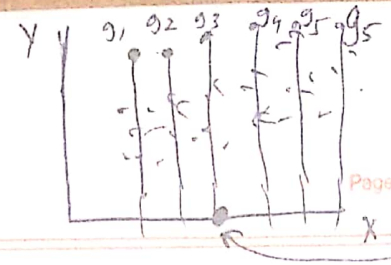
However ① if we create a decision tree, we found out that out of 2000 people who are having a car and are married, 400 were interested in exhibition.

② Also out of 3000, who do not have a car but have PG/higher degree. ~~too~~ 500 turned up for exhibition. So we can focus on these 2 grps for higher response rate instead of complete population.

3 steps of tree construction

- Building the tree
- Pruning - cross validation
- How to do it in R

→ analyse the tree, accuracy, error, rebuilt, again revalidate and come up with final 'optimal' version.




rpart partitions the data (say x b/w 0 to 100) into multiple partitions (say 10) and then calculate Gini Index g1, g2 -- g10 whichever partition gives minimum 'Gini', that x value is picked in tree

## Building the tree -

- build algorithm to choose the best split at every level
- To know what is best, best split is something which reduces the impurity level of the node the most
- Methods used to determine impurity
  - Gini
  - Entropy
  - Classification

## To find out impurity

Case is to find which colour ball would come if I pick from different bags.

bag 1 | 5R5G | - high impurity - unambiguous - not sure which ball would come if I pick one

bag 2 | 8R2G | - low impurity - high chances that red would be the outcome

bag 3 (Pure node) | 10R0G | - 0 impurity - 100% chance of red outcome

For classification - idea is to reduce this impurity at each node

For classification tree, ultimate goal is to find out those boolean rules (Married, Yes/No) which reduces the impurity of each node.

So these are different measures of impurity which ultimately gives you same answer, as they all are based on probability.

Page Date

# Gini Measure

It is the probability of randomly selected elements belonging to the same class

Sum ( p(i) ^ 2) for all class i.

Higher the gini score, its better.

2 Assumption

- Sample size is large

eg- What is the probability of getting 2 balls of same colour from bag 1

$$RR + GG = \frac{5}{10} \times \frac{5}{10} + \frac{5}{10} \times \frac{5}{10} = \frac{1}{2}$$

(without replacement, probability of 2nd ball would have been 4/9

but because assumption is that the sample size is large, we will consider $\frac{5}{10}$ instead of $\frac{4}{9}$

So impurity = 1 − Purity = $1 - \frac{1}{2} = \frac{1}{2} = .5$

For Bag 2 = $1 - \left(\frac{8 \times 8}{10 \times 10} + \frac{2}{10} \times \frac{2}{10}\right) = \frac{32}{100} = .32$

So impurity for bag 1 is higher than Bag 2

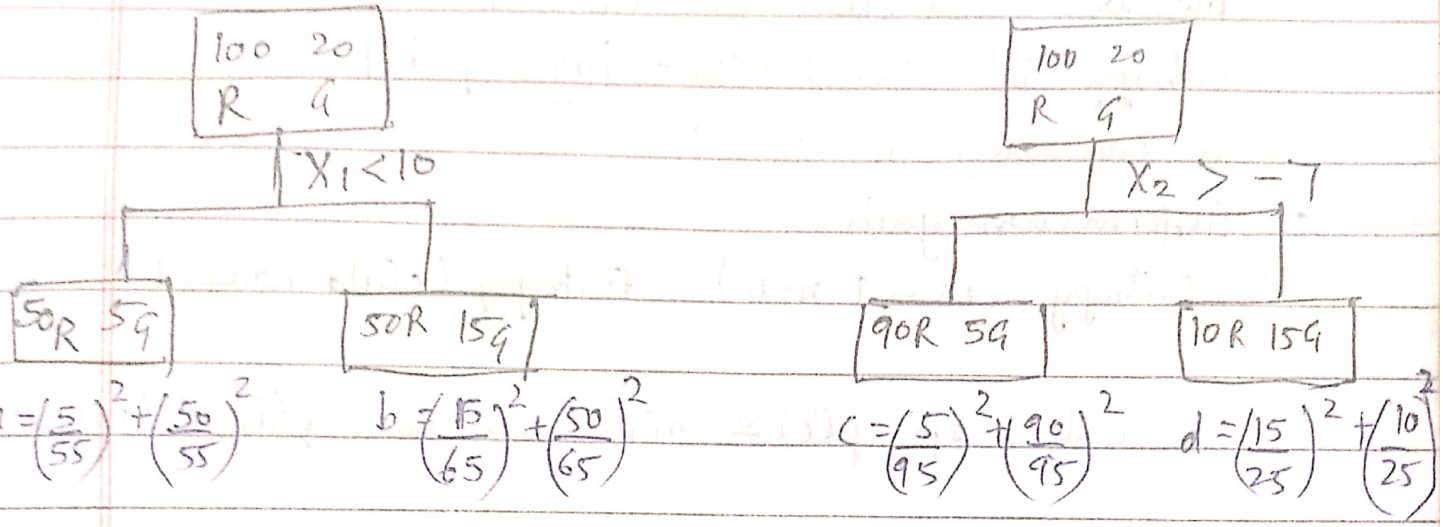
$p1 = P(G)$

$p2 = P(R)$

$P(i) \wedge 2 = P(G)P(R) + P(G)P(R)$

Gini is the measure of purity, so higher is better
1-Gini is impurity so lesser is better.

To consider which variable to pick for splitting




a is level of purity
To compare 2 different options, we need to find out the weighted average at each node.

pure1 - $a \cdot \frac{55}{120} + b \cdot \frac{65}{120}$ - weighed measure of purity for this split. (no. of observations that land in the node gives weight)

pure2 - $c \cdot \frac{95}{120} + d \cdot \frac{25}{120}$

purity should be higher or 1 - purity should be lower. So split 2 is better
For further splits, these a, b, c, d childs would become parent and again we would compute purity (im)

## Entropy

Another measures, to find out best split. All gives same results as all are based on probability.

Gini

- measures reduction in entropy
- Entropy is $sum[p(i) * \log_2 p(i)]$ where $p(i)$ is probability in class i.
- Information gain

  Entropy (Parent node) − Entropy (child nodes)

So instead of $p(i) * p(i)$ we use $p(i) * \log(i)$

## Misclassification

error rate Kind of

Proportion wrongly classified $= \frac{5}{11}$

$C_1 = 5$

$C_2 = 6$

Pay for box 1 **5 OR 5** because of the majority, we are classifying this box as red.

However error we are making is $\frac{5}{55}$ as 5 balls are green

So for a, b, c, d nodes calculate misclassification. Take weighted average

Wherever misclassification is less, choose that split.

Use package "rpart" (Recursive partitioning)
rpart.plot to visualize the tree

Page Date

If too many outliers in the dataset, go for decision tree instead of Regression as it asks only % < certain number, does not cut outliers.

If too many variables, use decision tree first to identify important variables and then go for logistic regression.

In DT, you directly talk about variables interaction in Logit you do not directly talk about interaction if you have a car and married

as this is a classification tree →

rpart(y ~ age + gender, data = d, method = "class") → dt

rpart.plot(dt)

x norm(1000, 45, 4) = age

It creates 1000 records with mean 45 and stv = 4

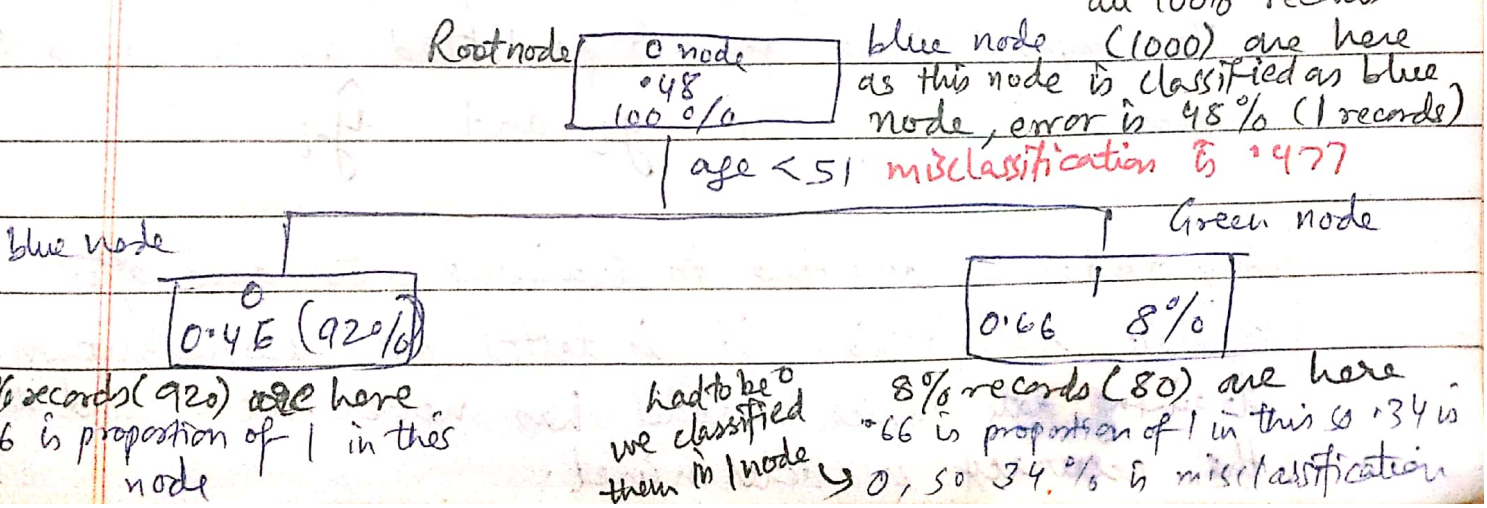
table(d$y) y is 0 and 1

0 1

523 477

so 0 (blue node) = 523

1 (Green node) = 477 = 477/1000 = 47.7%



So age < 51 is one variable which is helping in the split by reducing the impurity to the maximum.

dt would give you detailed picture of the tree
* marked are terminal node

rpart.plot(dt, type = 1, extra = 2) instead of % provides absolute numbers

By default it is doing Gini measure.
For others you need to specify.

So I can now pass a new record into this tree to do the classification prediction (which node that record would fall into)

predict(dt, d) → pred
View(pred)
will give 0 and 1 probability for each record

predict(dt, d, type = "class") → pred
View(pred)
Here it would directly give the classification each record would fall into.

you can use test data here with dt = final dt

Confusion Matrix

table(actual = d$y, predicted = pred)

To compare how good I predicted as compared to actuals. Comparing of $y_i$ and $\hat{y}_i$

Any model is developed in training data and after complete satisfaction, it is tested on new/unknown training data which model has never seen and then the accuracy is determined.

Precision – $\frac{TP}{TP+FP}$ – What proportion of the identifications was actually correct

Recall – $\frac{TP}{TP+FN}$ – What proportion of actual +ves was predicted identified correctly

Page Date

| Actual \ Predicted | 0 | 1 |
|---|---|---|
| 0 | 408 | 115 ← False positive (was 0, you predicted 1) |
| 1 | 287 | 190 |

↑ False -ve (was 1, you predicted 0)

(False Positive Rate – FPR) = 115/(408+115)

= 190/(287+190) TPR (True +ve rate)

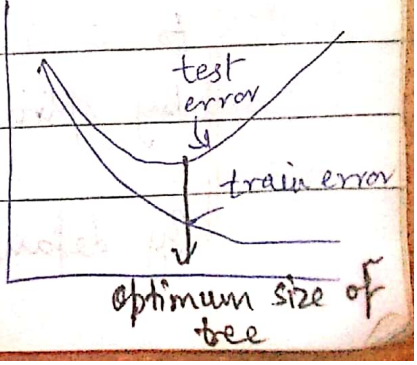
misclassification Rate = 115+287 = 402 = 40.2%

We started with 47.7% ~~misclas~~ misclassification from 523 initial 0's we are now classifying 408 only as 0

Another algorithm to use for split is chi-sq test using expected and observed frequency for gender, you calculate chi-sq (Gender), same for other variables Whichever is lower, that variable you choose. However, these days Gini and Entropy only are used.

If for any regression problem, we add 1 more variable to already used 5 variables, $R^2$ would increase (1 more variable would add to explanation) but adjusted $R^2$ may decrease.

Similarly for DT, if we keep on splitting, accuracy on training data would increase. But accuracy may decrease for an unforeseen dataset of test. This is called problem of overfitting So we need to find out optimum size of tree and stop there.



Page Date

By do

10 fold cross valid

Bigger trees means more complex tree so complexity parameter (like adjusted $R^2$) is an important parameter for Pruning.

with every increase in node size, what is the reduction in misclassification error. - complexity parameter is an interpretation of that - CP value

Say 10 nodes - error = '09

11 nodes - error = '085

So reduction in error = '005

Every split leads to a CP. so every leaf will have a CP value.

printcp(C.dt)

| CP | nsplit (no of splits) | rel error (misclassification error) | xerror | xstd |
|---|---|---|---|---|
| 0·0545 | 0 | 1·00000 | 1·000 | 0·0331 |
| 0·0167 | 1 | - 0·9454 | 1·014 | 0·0331 |
| 0·0157 | 4 | 0·8951 | 1·050 | 0·0331 |
| 0·0104 | 6 | 0·8637 | 0·9937 | 0·0331 |
| 0·0100 | 8 | 0·8427 | 0·9874 | 0·0330 |

rel_error = root node error ('477) is considered 1 (base line)

0·9454 says misclassification error reduced from 1 to 0·9454 i.e from 477 to 477 x ·9454 = 451 by splitting the tree once.

By default, at rel error = 0·01, tree would stop growing.

By default it is done. Can do 5 for CV also. However 10 fold is most accepted

10 fold cross validation → out of 1000, if tree is performing on 900 records only (randomly) leaving 100 records each time (which inbuilt it is considering as test data set), and to cross validate the tree performance, it calculates multiple misclassification errors, mean of these errors is xerror and std dev is xstd.

It is kind of inbuilt process of software to fine tune the tree.

So this error would reduce to a certain point and then again start increasing. So we need to stop building the tree here.

→ To ensure we build big tree

rpart(y ~ age + gender, data = d, method = "class", control = rpart.control(cp = .001)) → dt

it would create much bigger tree as cp = .001 so we are allowing even small improvements. By default cp value is 0.01.

(train error)

Here we see that rel-error continuously decreasing till 69 splits, but xerror (test error) started increasing after decrease at 1st split only,

(tree)

So we should stop splitting the tree after 1st level.

std dev rule - Pick that CP value where diff in xerror from higher split to lower split is within 1 std dev i.e. xerror + stddev contains 1 level above xerror. So pick 1 level above cp instead of the row where CP is lowest

plotcp(dt)

Now you know what level you need to cut the tree based on CP value, cut/prune the tree.

prune(dt, cp = 0.015) → finaldt

rpart.plot(finaldt)

→ Some strategis to make tree more understandable

* In this example we have used 2 variables, age (continuous) and gender (categorical with only 2 values) We can have categorical variables with more levels or we can even convert continuous in categorical (say 6→ ranges of age) for better interpretation
* DT does not bother about outliers
* for ex, we added occupation with 4 levels 'academician' 'lawyer', 'doctor', 'entrepreneur' in the existing example.

rpart(y ~ ., data = d, method = "class") → dt

rpart.plot(dt)

In this decision tree, only age and gender based rules are showing up. Not the occupations. So DT is telling you which variable is important or not. Now in subsequent logistic Regression you can leave occupation and only important variable age and gender are considered.

Imp →

Here for node 7, it uses condition occupation = "l', 'd', 'e', other condition is occupation = 'a'

So it has again taken the rules as boolean only

y ~ . - age = all except age
y ~ . = all variables

Page Date

Conditions in which we use DT -

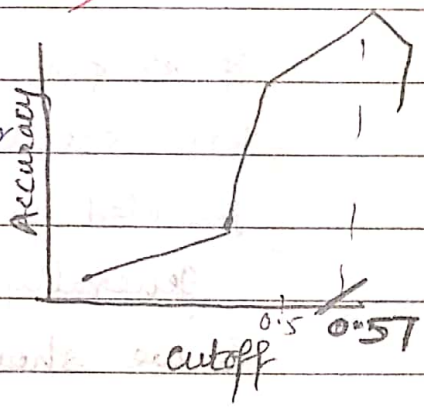
1) Have very large rows and columns
2) Have very large columns (count), you can use DT to find important ones to be used in subsequent models
3) Outliers in the data
4) Tells about variable interactions easily

3 performance measures

Higher variation in a variable is good for explanation. Lesser or no variation does not give much information.

① ROCR Curve - To determine good cutoff probability (instead of 0.5)

```
library(ROCR)
predobj <- prediction(pred[,2], d$y)        # takes prob 1 for each record; instead of d, use test data for fine tuning
perf <- performance(predobj, "acc")         # plots probability vs actual 0/1
plot(perf)
```




0.5 may not necessarily be a right number. So need to find right cutoff to label something as 0 or 1

| Actual | Predicted 0 | Predicted 1 |
|---|---|---|
| 0 | 100 | 200 |
| 1 | 200 | 500 |

It shows, max accuracy is reached at cutoff = 0.57

→ If I need to give loan, I can not take probability = 0.5 everytime for taking the decision.

→ Instead I will double check to find a healthy cutoff. (Next level decision)

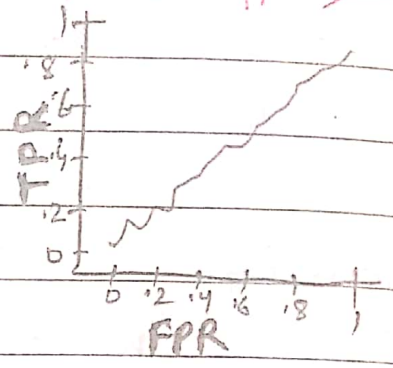
→ predobj has probabilities of all 1000 rows with prediction = 1

pred[,2] → we are considering prediction = 1 only, and d$y which has actual 0,1 values for y

Page Date

## ② Make a Trade off b/w FPR and TPR

```
perf1 ← performance (predobj, "tpr", "fpr")
plot (perf1)
```




| | 0 | 1 | |
|---|---|---|---|
| 0 | a | b | → false +ve |
| 1 | c | d | |

False -ve

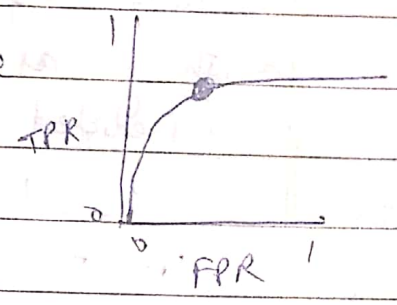
$$\text{accuracy} = (a+d)/N$$

$$TPR = \frac{d}{c+d}$$

$$FPR = \frac{b}{a+b}$$

High TPR is good as we are predicting 1 as 1
FPR should be less as something which was 0, we predicted as 1

Generally, the curve would be like
So we should choose a point whose TPR is maximum and FPR is minimum.




Eg - In case of sending mail for a new book, False +ve is not bad, because for those ppl (actual 0) who did not buy the book, we are predicting that they will buy the book. So we would send few extra mails.
- But False -ve is not very good. In above example, who actually bought the book (actual 1), we predicted

that they will not buy, So we would miss/loose those customer.

TPR and FPR changes (confusion matrix) when you change cut off, instead of '5, you use '1 or '01.

```
plot(perf1, print.cutoffs.at = seq(0, 1, by = '01))
plot(perf1, print.cutoffs.at = seq(0, 1, by = -1)
```

(3) Lift of the model - how much I am gaining from the model. Is the model giving me ROI

Say you have ~~pos~~ response of 1000 customers buying a product (0/1)
Prior probability is out of 100, 20 would be buy, so P(1) = '2

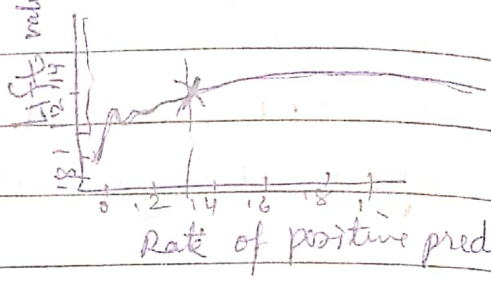
| y (1000) | | y (1000) | |
|---|---|---|---|
| 0 ] 20 | divide into | '87 ] 35 | Here you determine probability and sort them in descending. Now 1st decile has say 35 |
| 0 ] 20 | 10:10% | '86 | |
| 0 ] 20 | deciles | '81 ] 29 | |
| 0 ] 20 | for each | '74 ] 31 | |
| 1 | decile P(1) = 20 | | |
| 1 ] 20 | customers buying | ] | |
| 1 | | | |
| | | '02 | |
| | | '01 ] | |

Total 200 customers

Without using Model (WUM) — After using Model (AUM)

Now, WUM, if I was spending x amount on to get 20 cust here AUM, I am getting 35 cust by spending same x amount or you can say WUM, I got 200 cust spending y amount. However AUM, I got those 200 cust way earlier by spending less amount. So we have achieved the lift.

perf2 ← performance (predobj, "lift", "ppp")
plot (perf2)




It says beyond certain point (30% ppp) we are not gaining in lift value

Working on real data set G.csv

3 important questions to answer.

1) What are the important variables which need to be considered
2) What would be the healthy cutoff to decide who is 0 or 1
3) What is model's lift. How model is really performing on test dataset (ROI?)

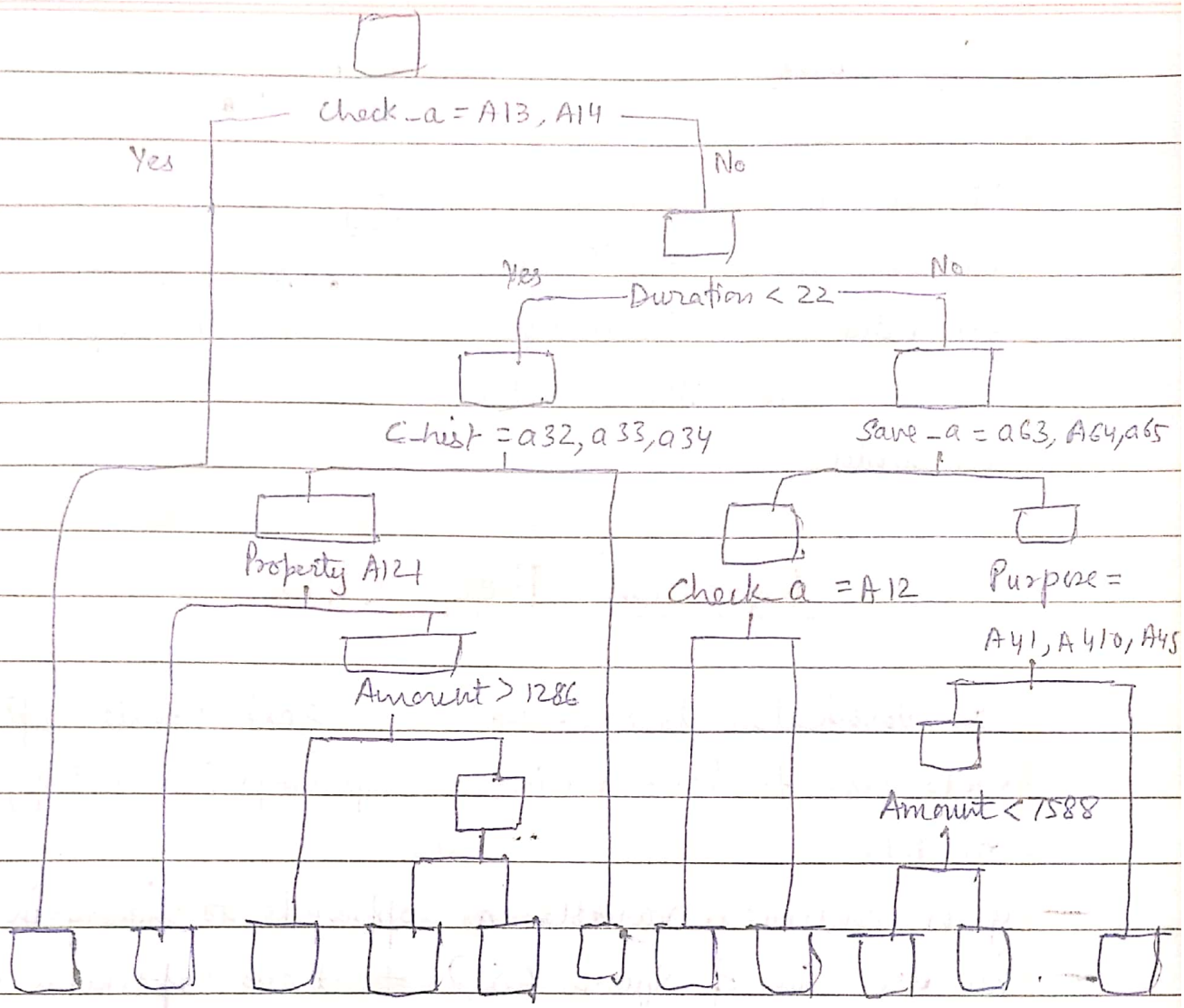
Intermediate question →

What should be DT size? Do you need to prune it? At what level you need to prune?

| CP | nsplit | rel - error | xerror | xstd |
|---|---|---|---|---|
| 0.0822 | 0 | 1.000 | 1.000 | 0.05577 |
| 0.0400 | 2 | 0.8355 | 0.8355 | 0.0527 |
| 0.0355 | 3 | 0.7955 | 0.8355 | 0.0527 |
| ~~0.0222~~ 0.0222 | 4 | 0.7600 | 0.8355 | 0.0527 |
| ~~0.0133~~ 0.0207 | 7 | 0.6933 | 0.8222 | 0.0524 |
| 0.0088 | 10 | 0.6311 | 0.7866 | 0.0516 |
| ~~0.0077~~ 0.0067 | 12 | 0.6044 | 0.8267 | 0.0525 |
| 0.0059 | 15 | 0.5777 | 0.8577 | 0.0532 |

Choosen CP →

* Decision Tree serves as a important, quick variable selection tool.
* tell lot about interaction



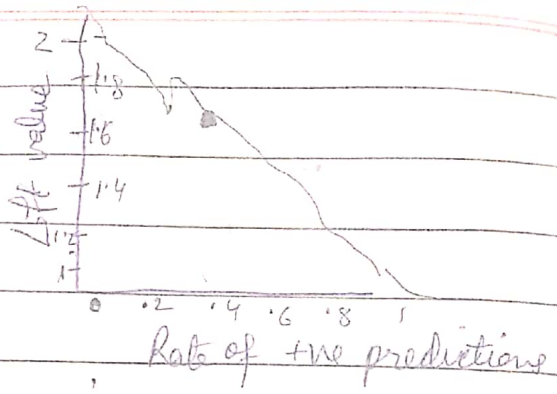

(1) This tree shows that, out of 20 variables in the file, check_a, Duration, c_hist, Save_a, property, amount ~~dtest~~ and purpose are important variables. If we have to perform linear Reg, we can perform on these variables only, Also we can come up with fewer dimensions.

(2) This tree also shows that there is interaction b/w duration and c_hist or c_hist and property. There is no interaction b/w duration and ~~amount~~ c_hist.

(3) If duration and c_hist is coming many times in the tree, we can say that they are strongly interacting

Page Date

## Lift curve




It shows with .35 TRR, we are achieving 1.6 of lift value i.e. 1.6 times more value than without using the model.

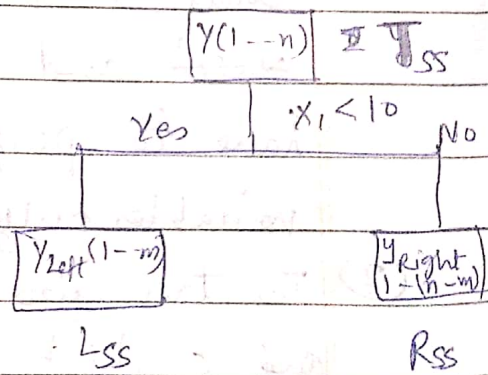
## Regression Tree (RT)

In classification tree, ~~where~~ our goal while splitting each node was to have minimum impurity (Gini, Entropy - etc) In RT

- y is continuous variable as opposed to binary in CT.
- We use Sum of Squares (SS) to have optimum split.
- $L_{SS} + R_{SS} < T_{SS}$, those variables are used at ~~at~~ splitting, which makes this equation true (Reduces SS) where $T_{SS} = \sum(y_i - \bar{y})^2$

$$L_{SS} = \sum (y_{L_i} - \bar{y}_L)^2$$

$$R_{SS} = \sum (y_{R_i} - \bar{y}_R)^2$$




- Note doing any estimation here, we are just calculating ss only.

Most of data mining techniques do not have any assumptions.

Page Date

- Linear Reg vs Regression tree — how a linear reg situation can be outsmarted by Regression tree (ex)

- R Code

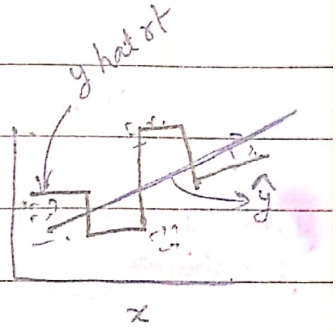
Picked up y variable having 10 values each in ranges 15-25, 0-5, 30-35 and 15-17

x is variable from 1 to 40

fit lm on it, lm(y ~ x) → ft

predict (ft) → yhat

lines (x, yhat, col=1)




Not getting good output from Lm.

rpart(y ~ x, method = "anova") → rt

predict (rt) → yhatrt

lines (x, yhatrt, col=2)

Correctly predicting y, so it says y nx is not a linear relationship.

Imp → In regression, you are getting very poor $r^2$, it tells that there is no strong linear relationship, So by RT there

To check the extent of agreement (accuracy) in both cases.

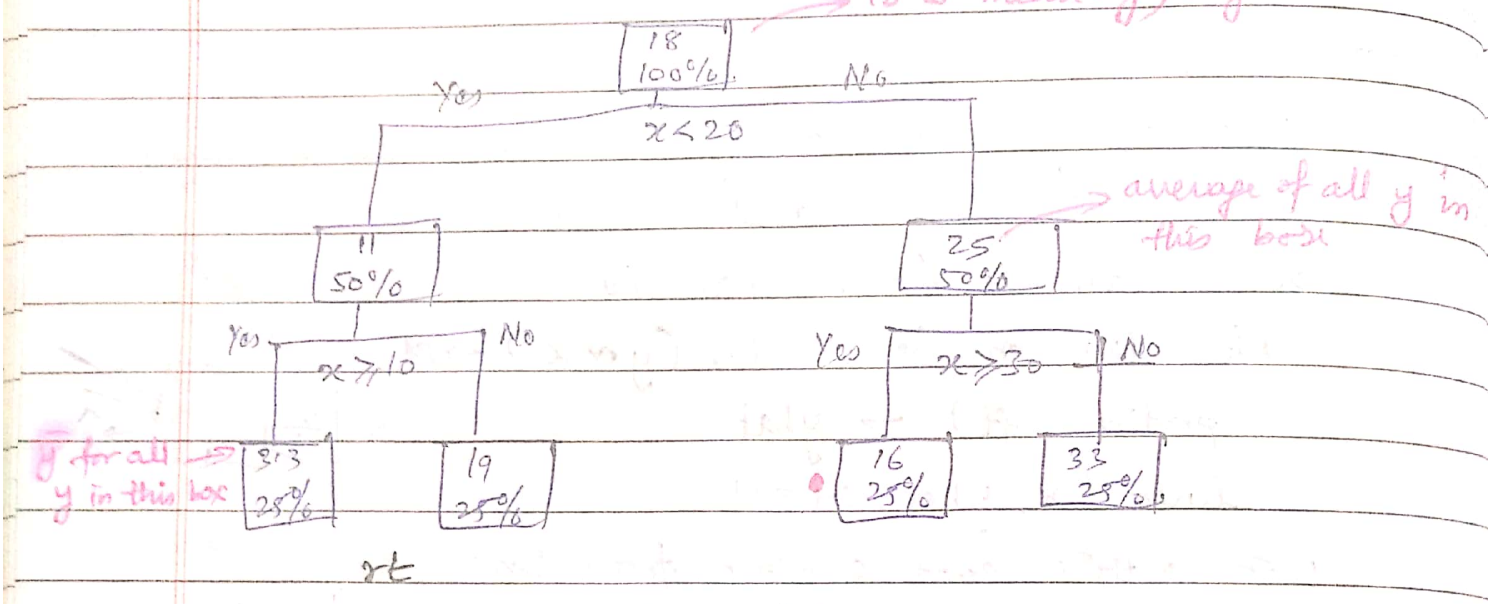
- LM $\Sigma(y - yhat)^2 = 4401.167$
- RT $\Sigma(y - yhatrt)^2 = 92.62$

LM shows lot of disagreement of yhat values from y as compared to yhatrt/y.

Page Date

Summary (ft) Shows $r^2 = 0.0424$ (very poor)

rpart.plot (rt)




which (x < 20.5) → id

mean (y[id]) → gives 11

printcp (rt) prints complexity parameter (CP) x error and stderror.

rel_error of next row = rel error of previous row − CP

Root node error = $\frac{4596.4}{40}$ = 114.91

= sum ((y − mean(y))^2)

= 4596.4

So objective is to reduce this SS at each node so that node error is minimum.

predict (rt)

gives that average y mentioned in each box for corresponding y. So predicted y for x = 37 is 16 as x is < 20 and x > 30,

Page Date

Imp - RT is very helpful in finding missing values If there are many missing values in data, its good to have some estimation (mean y in that box) for that value to go further.

— RT does not worry about Outliers. If there is an outlier in y variable, it may change the structure of RT. However outlier in x does not impact much.

Imp — Influence point - prof added outliers in y,x values. Also added z and gender. Then ran lm, which gives high (.59) $x^2$. It is not that the relationship has become linear, but this one observation is pulling the line from non-linear to linear. So need to be careful about such situation.

Imp — Cooks. distance (ft)

Threshold value of Cooks distance = $\frac{1}{\text{No of observations} - \text{No of variables}}$

$= \frac{1}{39}$ in this case $(40-1)$ without adding extra var

For 41st row, cooks distance = 12.3

For other rows, $< \frac{1}{39}$

So we can say 41st row is an influencer. Cooks distance is the measure of how each value is influencing the linear lines. CD is

Imp fundamental is finding outliers in lm

Page Date

# Random Forest (RF)

Eg – To find out who will win KA election would not settle down on input of 1 person, ~~I~~ ~~used~~ instead would ask many, and based on their answers, would find out real answer. Not dependent on 1 predictor, instead dependend on multiple predictors (also called democracy) for an estimate

close of techniques called

Ensemble (collection) Techniques

mainly 2

Extremely popular because of accuracy, not because of explanatory power.

(1) Bagging

| Random Forest |
|---|
| |

(2) Boosting

| Gradient boosting. |
|---|
| |

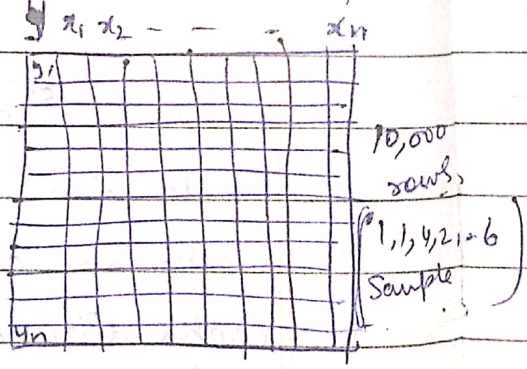
Base of both bagging and boosting is trees (DT)

– Ex. dataset with $x_1$ – $x_n$ var and y var, we create RF of 200 trees

Random Forest (RF) randomizes on following 2 things.

– Random Sample (Sample with replacement) ↳ Bootstraping.

– Random Columns.




So if I am creating RF of 200 trees, in each

Page Date

Sample Some rows would be duplicated and some would be left out (as samples are created with replacement)

These which are left out are called Out of bag (OOB) samples.

Second is we do not use all columns in each sample. In KA ex. input from a family of 4 ppl. would be same (with same mindset). so no point in taking response of complete family.

If we create identical trees with all columns, it would be like same trees. So objective is to create different trees with different no of columns (thumbrule is $\frac{1}{3}$ or sqrt(no. of columns)) to be considered in a particular tree for splits

→ Creating Random Samples (10,000 records from the population serves as sample again and again with replacement) is called Bootstraping/Bootstrap sample.

→ Here we enforce diversity.

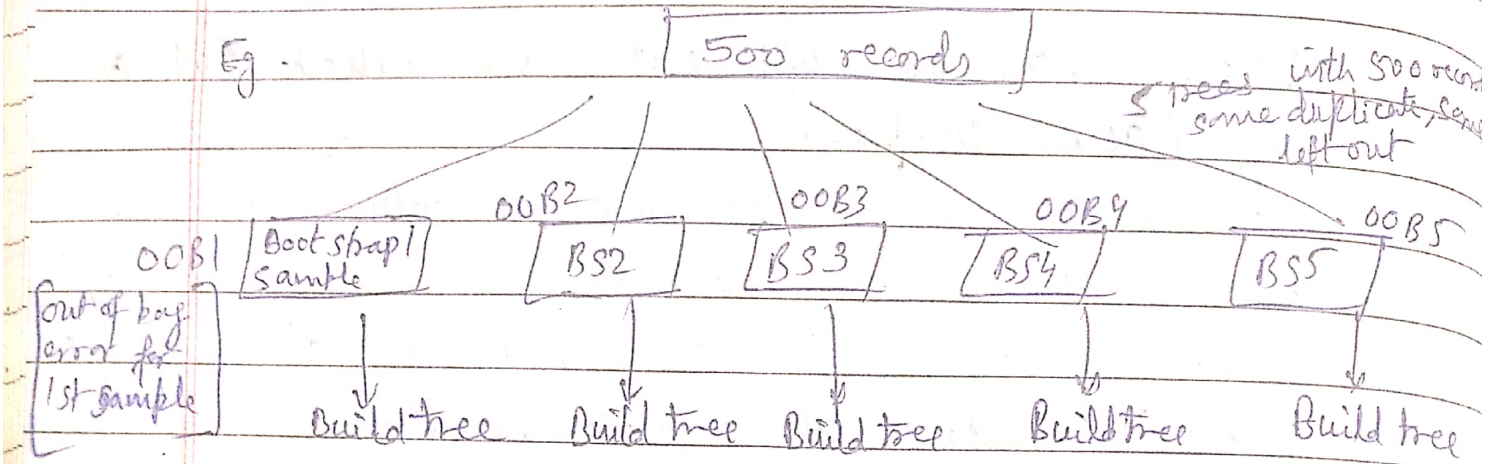
→ If I create samples without replacement all 1000 would come each time so all trees would be identical

6) → In DT we use all vars of the dataset for splitting and use that var finally for split which reduces the impurity the most. In RF, we use subset of columns each time to create a fully grown tree. So at each node split, $\sqrt{\text{vars}}$ are used to determine impurity. For each node set of $\sqrt{\text{vars}}$ can be different.

Page Date

→ In RF, we create fully grown trees i.e. no CP value to be used, as it would have cancelling effect.

Eg.




Consider only a fraction of variables at each split) → random var for each split

best var out of √var is chosen for the split.

↓

full grown tree

Now for prediction, classification, we pass a new record to each tree and then take the majority decision. For some ex, if 3 trees predict 1 and 2 trees predict 0, the the outcome should be 1

for prediction, Regression, again we pass the new record to each tree and then take the average / aggregated decision. for ex, we get 5 different numbers for $\hat{y}_{new}$ using 5 trees, now we take the average of all to get final estimate of $\hat{y}_{new}$

Result is the ensemble (collection) of outputs in RF. RF can be classification or Regression type based on y variable.

9mk

RF makes sense when you have large number of rows and columns. For less rows/columns, you will not get enough variation to create a forest.

eg. of classification forest using credit-data

```
library (randomForest)
x randomforest (CREDIT ~ ., data = d, ntree = 200,
                mtry = 5) → rf
```

mtry = 5 means 5 variables to be used.

```
randomforest (as.factor(CREDIT) ~ ., data = train, ntree
              = 200, mtry = 5) → rf
```

rf

Output interpretation

OOB estimate of error = 23.2%

Confusion matrix

| | 0 | 1 | class.error |
|---|---|---|---|
| 0 | 634 | 66 | 0.0942 |
| 1 | 166 | 134 | 0.5533 |

OOB estimate means those Out of Bag records which were not the part of the sample are passed through their respective trees, there would be some misclassification. So the count of this misclassification is generated as % error by the output.

Hence OOB error is basically an estimate of how your RF model will perform with an unknown data set.

In RF, you dont worry about collinearity. So you are able to include anything & everything.

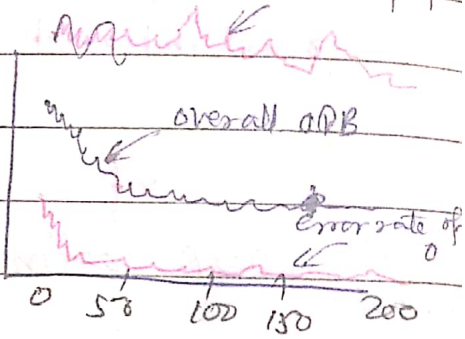
Page Date

rf $err

would give output for all trees

| OOB error rate with 1 tree → | OOB | 0 | 1 |
|---|---|---|---|
| [1,] | 0.3649 | 0.2682 | 0.5752 |
| [2,] | 0.3299 | 0.2355 | 0.5505 |
| [3,] | 0.3337 | 0.2390 | 0.5502 |
| [4,] | 0.3452 | 0.2418 | 0.5826 |

OOB error with 4 trees




plot (rf)

It tells that after 150 trees we are not getting and significant improvement in overall error rate.

So no need to build a forest of 200 trees.

To predict, predict (rf, test) → pred

table (test $ CREDIT, pred)

| | pred 0 | pred 1 |
|---|---|---|
| 0 | 164 | 11 |
| 1 | 48 | 27 |

error = 59/250 = .236

Imp

To chk which variables are important

Sort (importance (rf)), varImpPlot (rf)

It prints a table of Gini's measure which tells for all possible splits, how much is the contribution of each variable in reducing the impurity.

Page Date

getTree(rf, 15) gives complete information about 15th tree. However we cannot visualize RF.

Importance table

| Variables | Mean Decrease Gini | |
|---|---|---|
| check_A | 37.33 | It take across the RF, for all possible splits, contribution of a particular variable. Hence it gives you some sort of explanatory power |
| Duration | 32.71 | |
| c_hist | 19.08 | |
| Purpose | 25.13 | |

you can use RF misclassification as a benchmark. (best black box model's accuracy) and use these important variables with further Regression models.

Suppose RF gives error 23% ~~but~~

DT or any Regression model gives 22%

So you can decide whether, 1% gain in accuracy by RF is worth the explanation loss which we are not getting because of using RF

However, if you are not clear about imp variables, have large rows and columns, so you can go for DT/RF and use that information to use in further models.

RF gives best case scenario and imp variables.

Big Problem - Boosting algorithm can not work with Categorical variables. All variables should be numeric

Page Date

## (2) Boosting ensemble techniques — Gradient boosting

Here we build small trees (not fully grown) sequentially. First, we build 1 small tree and calculate the errors, Errors/Prediction of this tree is used to build another small tree and so on. This process reduces the gap of errors with each step. You keep on doing it until you dont get further improvement.

Tree 1 (small tree)

↓

Errors → Tree 2

↓

errors → Tree 3

prediction (t) = prediction (1) + prediction (2) — — —

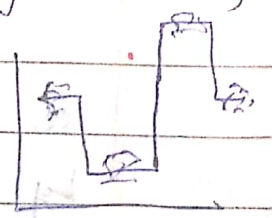
+ prediction (t-1)

Ex —

y = 40 values with 10 values each in range 15-25, 0-5, 30-35 and 15-17.

x = 1 : 40

plot (x, y)




create 1st tree — rpart (y ~ x, method = "anova") → st1

plot predicted value of ŷ with y — predict (st1) → p1

lines (x, p1, col=1)

Now to boost the model, we need to check the errors it is making.

e1 = y - p1 — error = actual - predicted

2nd tree: rpart(e1 ~ x, method = "anova") → rt2

p2 = p1 + predict(rt2)

lines(x, p2, col = 2)

sum((y - p1)^2) = 92.62

sum((y - p2)^2) = 81.63

Here sum of squares error reduced from tree 1 to tree 2.

e2 = y - p2

rpart(e2 ~ x, method = "anova") → rt3

p3 = p2 + predict(rt3)

sum((y - p2)^2) = 73.419

Guideline rule is Error should be related to x

Hetroscadascity. So it gives strong results

Absence of hetroscadascity = Homoscadasity

= No relation b/w ~~err~~ residuals/errors with x values.

## Using R Package xgboost

Imp for boosting all x should be numeric, as we use derivatives here

$\frac{\delta}{\delta b} \Sigma (y - a - bx)^2$ which needs all x to be numeric.

If you are creating CRedit scoring model, you cant rely on RF as accuracy is not the only intention, You need the explanation there.

Page Date

Imp

Popular method to code factor variables to Numeric is One-hot coding — Dummy variables with all levels —

| Factor Occupation | One-hot coding |
|---|---|
| | D E L |
| Doctor (D) | 1 0 0 |
| Engineer (E) | 0 1 0 |
| Lawyer (L) | 0 0 1 |

Imp Short cut to do one hot coding for all variables

Say we our glm and fetch model matrix from it which we can directly use for one hot coding,

```
glm(CREDIT ~ ., data = train) → lt
model.matrix(lt) = m
View(m)
```

has all variables with one hot coding

Use this now for gradient boosting.

```
xgboost(data = m, label = train$CREDIT,
nrounds = 3, max_depth = 2, objective = "binary:logistic")
→ xg
```

3 trees sequentially

not fully grown but only till 2 splits

Output — trainerror : 0.264000
train-error : 0.264000
train-error : 0.274667

} misclassification measure

if you increase trees to 20 and depth to 4, error reduces a lot, So play with these parameters

gwp

Gradient Boosting works better than RF. On kaggle these days, boosting is the winner for most competitions

Gives very good accuracy.

As test dataset is not in 0,1 column structure which is required for xg model (in case of factors), so need to convert test dataset to model matrix.

glm(CREDIT ~ ., data = test) → lt

model.matrix(lt) → mtest

predict(xg, mtest) → pred

View(pred) ← predicts probability of 1

ifelse(pred >.5, 1, 0) → pclass To chk classification

table(test$CREDIT, pclass)

| | pclass | |
|---|---|---|
| | 0 | 1 |
| 0 | 152 | 33 |
| 1 | 43 | 32 |

error = 66/250

= .264

To improve performance of gradient boosting, we use parameter "eta", it tells how much weightage to give on subsequent level trees generally a value b/w 0 and 1 as low weightage should be given

Thumbrule for eta = .1

As we earlier added previous level p to next level p directly ($p_2 = p_1 + pred$), using eta we are adding the weighted p to next one.

xgboost(data = m, label = train$CREDIT, nrounds 20, max_depth = 2, objective = "binary:logistic", eta = 0.05) → xg

Using eta = .1 in next step performed better

Overfitting

Using high eta or max depth, you can get less train error but test error would increase, so be cautious here.

age - 35 ≥ 0 buy  age - 35 < 0 not buy  age ≥ 35

age - x ≥ 0  ← To keep threshold constant we add that bias

↑ Can change anytime

Page  Date

# ANN - Artificial Neural Network

1) Massive connectivity and parallel processing

Draws parallels with brain neurons. The way brain neurons learn on its known through learning, experience same way ANN learns on its known. 100 billion neuron in brain, each one connected to 5000 other neurons.

eg identifying a, a, a, a

how brain knows that above all comes in 'a' category.

not just by knowledge, brain tries to map slant, shape etc to the learning of a and classify anything in this category as a. So brain parallely processes many!

ANN works on same principles as brain

2) Robust and Fault Tolerant, Capability to adapt to surroundings, if today you are in one city, tmrw in another city, brain adapts to surroundings

3) Ability to learn & generalises from known examples - 3 pics of cat, you can identify 4th also as cat

4) Collective behaviours is more imp than individual behaviour.

Artificial Neural N/w aim is to mimic Biological Neural N/W

Highly complex, non linear, and parallel processing system - Brain

Eg. No linear answer, why we like a person and why we do not like.

linear.output = T when we do not want to apply any activation function on final output. We are good with $y_k$ as linear.

## In R

```
library(neuralnet)
data.frame(y = runif(100, 40, 50)) → d
d$x <- sqrt(d$y)
```

x, y have sqrt relationship

Building a nn, to make computer understand this relation.

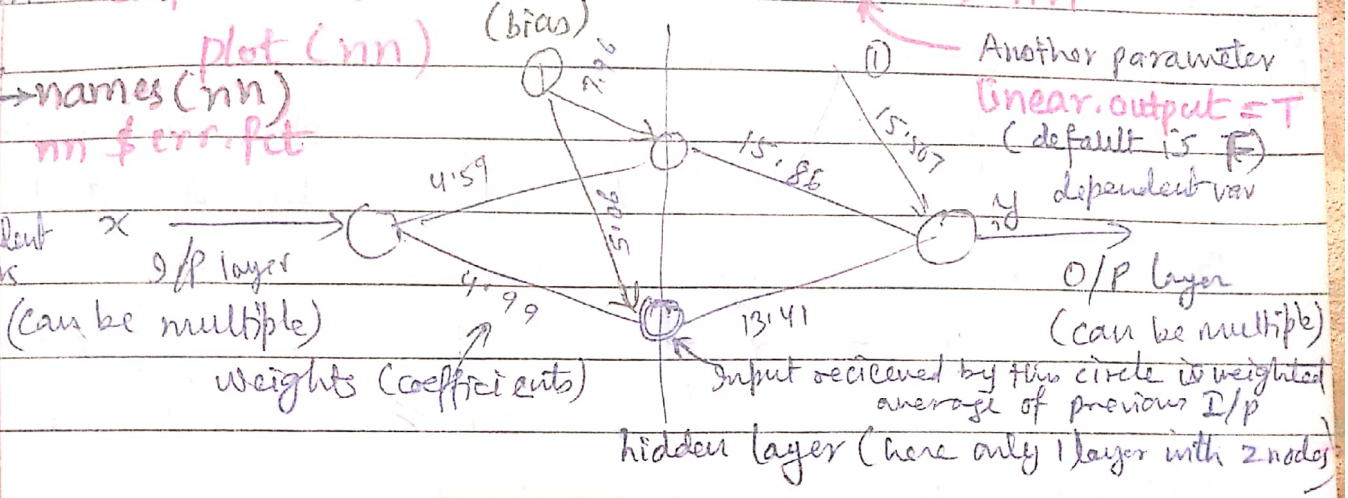
```
neuralnet(formula = y ~ x, data = d, hidden = 2,
err.fct = 'sse', threshold = 0.01) → nn
```

Another parameter
linear.output = T
(default is F)

```
plot(nn)
```

will give info for all fns in nn → `names(nn)`

`nn$err.fct`




typically we start with more nodes and then converge

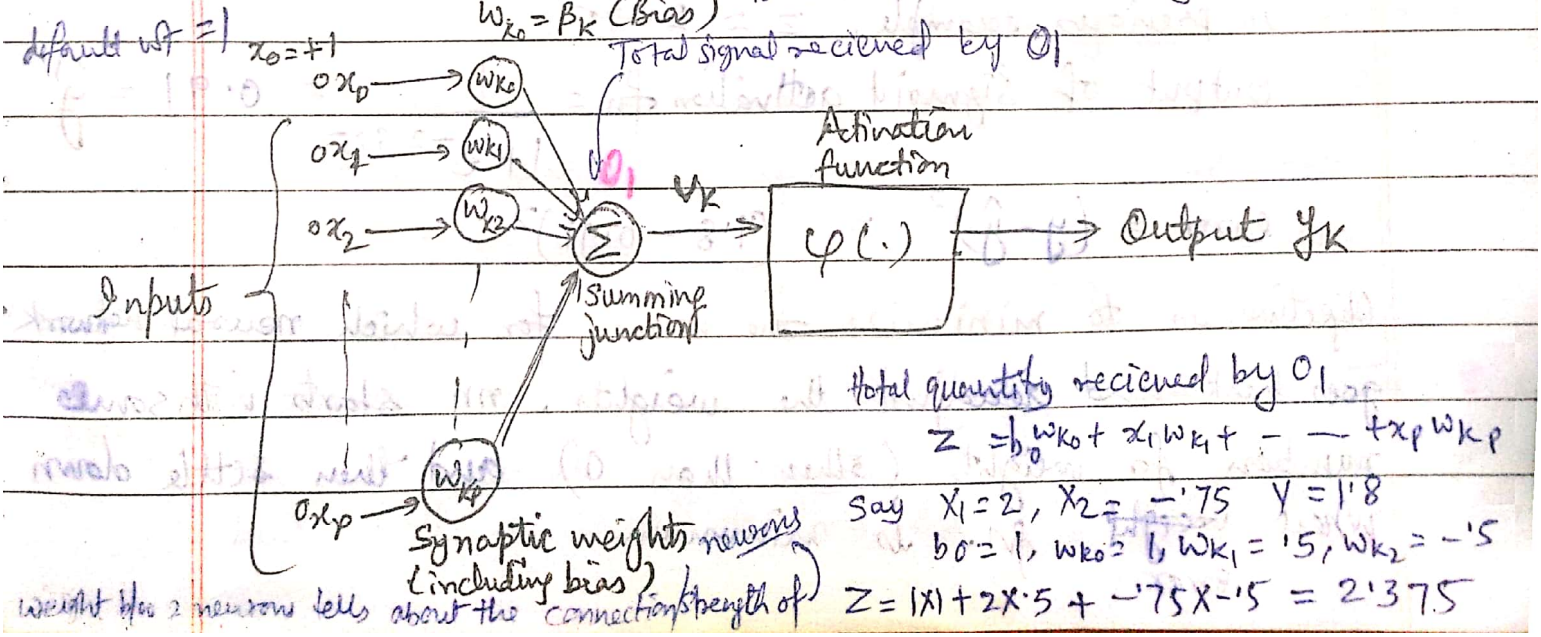
hidden = c(3,2) — would give 2 hidden layers with 3 and 2 nodes.

for ex (10, 5, 2)

```
compute(nn, d$x) → p ; view(p)
p$net.result
```

here it is predicting $\hat{y}$ for all values $y = 44.78$.

No variation so bad nn.




Total quantity recieved by $O_1$

$z = b_0 w_{k0} + x_1 w_{k1} + \dots + x_p w_{kp}$

say $X_1 = 2$, $X_2 = -.75$ $Y = 1.8$

$b_0 = 1$, $w_{k0} = 1$, $w_{k_1} = .5$, $w_{k_2} = -.5$

$Z = 1(1) + 2(.5) + -.75(-.5) = 2.375$

learning rate and structure (no. of hidden layers) bring more variable in prediction. Err.fct and Act.fct do not bring much diff.

Page | Date

## Activation functions –

1) Softmax (Generalized logistic function)

$$y_j = \frac{\exp(z_j)}{\sum_{i=1}^{n} \exp(z_i)}$$

→ output is b/w 0 and 1

→ Sum of all output values is 1

default fn.

2) Sigmoid function (most popular) (Logistic function)

$$y_j = \frac{1}{1+\exp(-z_j)}$$

The output is continuous b/w 0 and 1

3) Heaviside function (also known as Perceptron)

$$y_j = \begin{cases} 0, & z_j < 0 \\ 1, & z_j \geq 0 \end{cases}$$

Output is 0 or 1 (Binary)

4) Hyperbolic (Tanh) function

$$y_j = \frac{\exp(z_j) - \exp(-z_j)}{\exp(z_j) + \exp(-z_j)}$$

Output is continuous b/w −1 to 1

5) Rectified Linear Unit (ReLU) – $\max(0, z)$ if z is −ve, 0; if z is +ve, z

You have to try with various activation functions, however whichever gives best results. However Sigmoid and Tanh are used mostly.

In previous example, $z = 2.375$

Output of Sigmoid activation fn $= \frac{1}{1+e^{-2.375}} = 0.91 = \hat{y}$

Error $= (y - \hat{y})^2 = (1.8 - 0.91)^2$

Objective is to minimize the error, for which neural network goes back and readjust the weights. nn starts with some numbers for weights (other than 0) and then settle down where ~~weight~~ error fn. is minimum

Imp — You can't use Categorical variables (Factors) for NN, as we want be able to calculate Gradient descent. So factor variables need to be converted to 0/1 using One-hot coding

$i = 1$ to $n$ (for all rows)

→ Sum of square error fn (quadratic) $= \frac{1}{2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

In 0/1 variables both err. func can be tried
or In continuous variables, use SSE

→ Cross entropy error fn $= \sum (y_i \log p_i + (1 - y_i) \log (1 - p_i))$

For classification, better to use Cross Entropy through SSE can also be used.

As neural network looks at error, access whether large/small, go back to re-calaitate weights, this process is called through Gradient descent.

If more than one neuron, say 2 neuron, $z_1$ and $z_2$ would be calculated. For $z_2$, weights would be different. If more layers of neurons, $z_1$ and $z_2$ becomes $x_1$ and $x_2$ for them. However, weights again would be different. Bias for these layer/neurons would also be different.

Gradient descent - fundamental method of estimation

Say $y \sim x$ $b$ is coefficient of $x$

$\frac{1}{2}(y - bx)^2 = SSE$

Take derivative of SSE w.r.t b (which talks about slope)

$\frac{\delta SSE}{\delta b} = (y - bx)(-x) =$ slope = gradient

It tells with 1 unit change in b, how much SSE changes. Derivative is either an upward or downward slope. To reduce SSE we will have to descend downwards.

Adaptive learning rate

$b_{new} = b - (y - bx)(-x) \times \eta$ — eta (learning parameter)

However if we descend this fast, it can be a problem. So we want to descend slowly not by complete slope but with a small fraction eta $= \eta$ (typically 0.1)

# Matt Mazur step by step back propogation example

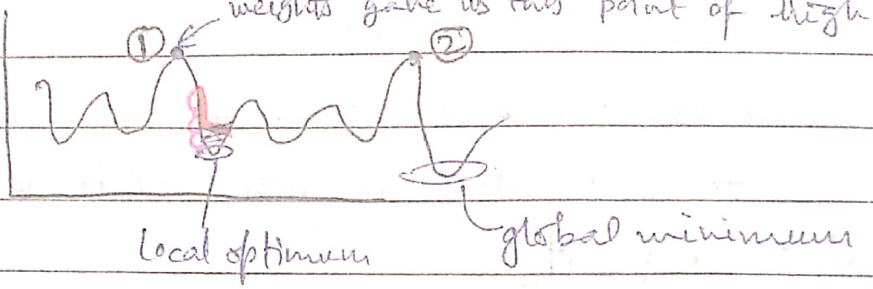
Page Date

$$SSE_{new} = \frac{1}{2}(y - b_{new}x)^2$$

If two outputs $SSE_{new} = SSE_{o_1} + SSE_{o_2}$

Repeat this process again n again for revising weights until there is no more improvement possible

plot of SSE for multiple x vals




If we take huge jump in eta for descent, we may miss the bottom point (local optimum here) and keep on oscillating as every time we have to go down the slope by eta

So better to descend slowly using small eta so that oscillating problem is not there and we reach "local optimum"

Secondly, we may be satisfied with local optimum only without looking at "Global minimum" which was further

To deal with run 'nn' multiple times, as starting at point 1, you may start with point 2, and then you reach minimum SSE (global minimum) so that search area is maximized. Plotting SSS with various values of b

Forward Pass — Computing $\hat{y}$ and various wts (1st time)

Back Propogation — As we calculate SSE of last step ($\hat{y}$) and then propogate back to adjust weights in order to reduce SSE. All weights are recalculated (not just 1 or 2 of previous layer)

No of hidden layers — cant tell beforehand, have to try multiple times.

Uses chain rule of derivatives

V Imp. Tip

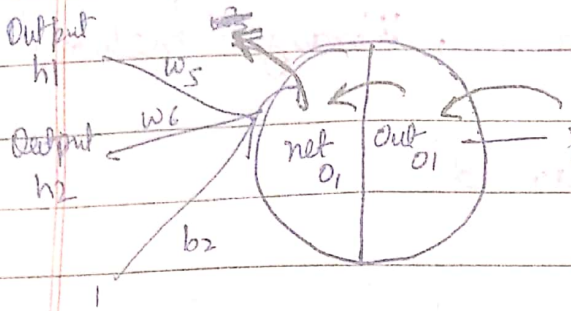
Range depends on the problem

Imp tip for NN – Scale down all the variables on a scale of 0 to 1 to have better predictability (or -1 to 1)

Range normalization $= \frac{age_i - min(age)}{max(age) - min(age)} = age_{norm}$ (1)

or z score normalization

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{Total}}{\partial Out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$




$$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2$$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial\left(\frac{1}{1+e^{-x}}\right)}{\partial(x)} = \frac{\partial\left(\frac{1}{1+e^{-x}}\right)}{\partial(1+e^{-x})} * \frac{\partial(1+e^{-x})}{\partial(x)}$$

Example

For some variables which we want to give more importance, can be normalized from -3 to +3 (or an expanded range)

Rest others 0 to 1 or -1 to +1. After NN, these variables can be converted back to original

## Example – Car Raw

Classification exercise for prediction

Dataset is about probability of purchase of 2nd hand cars

$V_1$ – vhigh, high, medium, low – Price

$V_2$ – vhigh, high, medium, low – maintenance cost

$V_3$ – 2, 3, 4, 5 and more – no of passengers it can hold

$V_4$ – 2, 3, 4, more – no of doors in the car

$V_5$ – small, medium, big – bootspace

$V_6$ – low, medium, high – safety features

$V_7$ – unacceptable, acceptable – Dependent variable

This data can't be fit into ANN so need to do One-hot coding

Imp Prog

```
colnames(d) <- c('price','mc','seats','doors','bootspace','safety','buy')
install.packages("onehot") ; library(onehot)
codddd <- onehot(d)
dnew <- predict(codddd, d)
view(dnew)
```

All factor variables have now been converted into one-hot variables. Convert dnew into dataframe

```
as.data.frame(dnew) -> dnew
```

(1728 observations, 25 variables)

In case of neural network we can't directly use ~~y~~ y ~. (. is not acceptable and it is very cumbersome to type all 25 variables)

```
str_replace_all(colnames(dnew), "=", "_") -> colnames(dnew)
paste(colnames(dnew)[1:21], collapse = "+") -> st
paste("buy_acc~", st) -> st1
as.formula(st1) -> f
```

to convert string into formula.

# Time Series Forecasting

Page Date

## Time series Analytical techniques

- Moving Average methods
- Stationarity, ARIMA

### Objectives

- Understand the importance of forecasting and its impact on the effectiveness of the supply chain and overall performance of the organization
- Learn various components of time series data such as trend, seasonality, cyclical components and random components.
- Learn different techniques such as moving average, exponential smoothing methods.
- Learn practical challeges associated with forecasting model

We all want to Know/predit about the future. However if forecast is inaccurate, it can impact top line & bottom line of the organization. So it becames necessary to forecast the demand and services for a product as accurately as possible.

eg:- Non availability of products / too much inventory
- forcasting bench strength based on anticipated projects in Infosys.

## Forecasting Methodologies

→ There are many different TS techniques
→ It is usually impossible to know which technique will be best for a particular dataset - Reason is data would be completely different for different techniques.
→ It is customary to try out several techniques and select the one that seems to work best
→ To be an effective TS modeller, keep several TS techniques in tool box of yours like Simple moving average, Exponential moving average, Holt winter, ARIMA, Auto Regression methods
→ Auto Regression is subset of linear regression.
→ Eg - In linear, $X_t$ = ad expenditure, $Y_t$ = Sales in a particular month

$$Y_t = \beta_0 + \beta_1 x_t$$

Now if we change $y_{t-1}$ as previous month sales and try to predict this month sales, it becomes auto regression problem (extension of simple linear regression)

$$y_t = a + b\, y_{t-1}$$

here instead of using another variable X to predict Y, we use historical data of Y to predict current/future Y
Above is AR model with Lag1. If $y_t \sim y_{t-2}$, it is AR model with Lag2

## What is times Series data

Time Series data is a sequence of observations collected from a process with equally spaced time periods
Examples — Daily closing stock prices

Libraries used in R – Forecast, date, Table, TTR

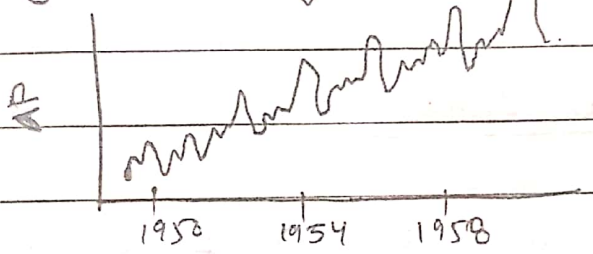
Ts data → Col 1 (Date/Y/M/any timeframe (equidistant), Col 2 (some measure)

Page Date

- Daily data on sales
- Monthly inventory
- Daily customers
- Monthly unemployment rates
- GDP

Time series should be equally spaced. Not like you have stock price data of weekly and daily mixed.

Monthly airpassengers data travelled

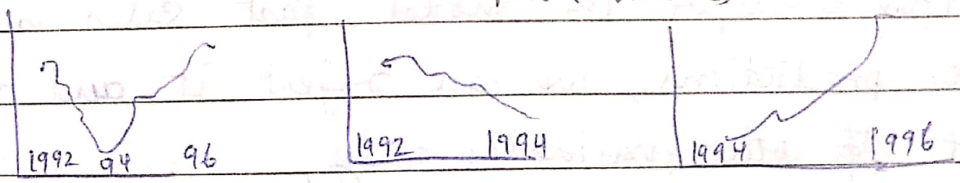



class(AP) = ts
↑ time series

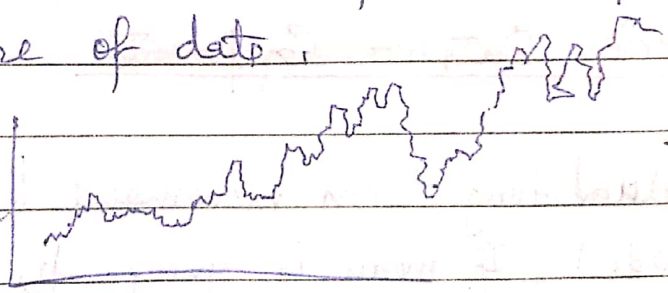
date is converted into ts

Same pattern (upward/downward) slopes is repeating every year. Also standard deviation and mean is increasing each year. xrate.ts = ts(xrate, start = 1991, frequency = 4)

plot(xrate.ts)




v pattern = breaking into 2 time periods to get clear picture of data.



Microsoft data with upwards trend but lot of noise (lot of jumping)

Page Date

## Why do we study TS analysis

- To understand past and to predict future, in order to take informed decisions.
- TS analysis quantifies the main feature in data and the random variation (Random variations are not always bad)
- Widely applicable in govt, industry and commerce.
- Deals with huge datasets.

## Typical Time Series

$$\hat{y}_{t+1} = f(y_t, y_{t-1}, y_{t-2} \ldots) + f(x_1, x_2, x_3 \ldots)$$

historical y — other variables

* f can be linear or nonlinear

If after developing the model, past sales is not giving better predictions, we can reject it and consider effect of other variables only.

## Time Series Data - Components

Trend - Gradual long term movement (upwards or downwards). If mean is going up, it is upwards. Trend is easiest to detect in the data.

eg. population growth in India

**Cyclical patterns** — Results from events which are recurrent but not periodic in nature (no fixed pattern but it repeats). An up/down repetative movement in demand repeats itself over a long period of time. Hard to detect. Repeating in infrequent interval

Eg. Recession in US economy.

**Seasonality** — Results from events that are periodic and recurrent in nature. An up/down repetitive movement with a trend occurring periodically Repeating with frequent interval. Seasonality is generally within year.

Eg. Sales in festive season.

**Random Components /Irregular components** —

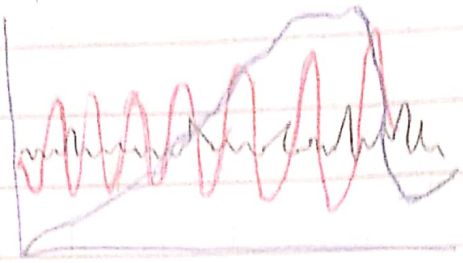
Disturbances or ~~residue~~ residual that remain after all above behaviours have been accounted for. Erratic movements that are non predictable because they do not follow a pattern. Completely Noisy.

Eg. Earthquake.

For a given TS data, we can seperates out Trend, Seasonality and random components by using R function "Decomposition"

Sales ← read.csv("Sales_RegressionTime.csv")
Salestimeseries ← ts(sales$Sales_1000, frequency = 4)
Sales ← data.frame(Sales) ; attach(sales)
Sales$time ← seq(1:20) (adding 1 to 20 values as time)
Salestm1 ← lm(Sales$Sales_1000 ~ Sales$time)
Salestm2 ← lm(sales ~ poly(time, 2), raw = True) poly means $x + x^2$

for microsoft data, after decomposing, we can get

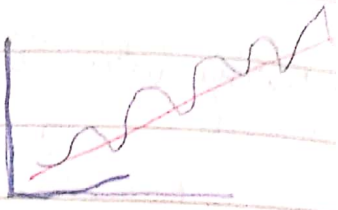


- Seasonal
- Irregular
- Trend

after decomposition, data starts showing patterns and information

## Regression against time

predicting sales based on time



**Linear Regression** →

Sales = $\alpha + \beta$ time $(y_t = 375 + 92.5 X_{1t})$

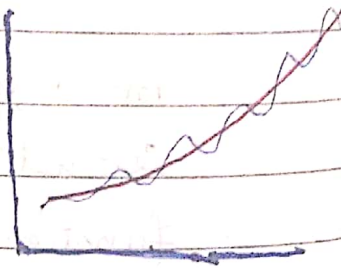
purpose of the model is to minimise RMSE (root mean square error) i.e error b/w predicted sales and actual sales. Also we need to fit linear line. To improve prediction, we are adding another coefficient

**Quadratic equation**

Sales = $\alpha + \beta$ time $+ \gamma$ time2

polyfm in R

$\gamma$ = gamma



so instead of straight line, we have started getting curvilinear line.

$y_t = 683.09 + 8.47 X_{1t} + 4.001 X_{2t}$

**Seasonal Regression Model**

Looking at data pattern, we can say that it goes up and coming down every quarter

Bring Seasonality - Sales $ Seasonal ← as.factor(rep(c(1:4), 5)) adds column with repeated 1:4

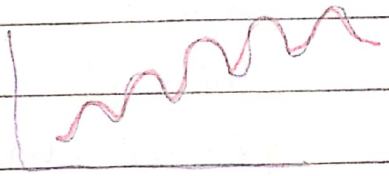
Sales lm2s ← lm(Sales ~ poly(time, 2) + seasonal, data = Sales)

points(time, predict(Sales lm2s), type="l", col="red", lwd=2)

Poly(time, 2) = time + time$^2$ = time + I(time^2) | Poly(time, 1) = time

In general we dont use > time$^3$

So we can try to bring that quaterly seasonal pattern in our model, by declaring some dummy variables



takes care of any contribution by quaterly pattern

| Quarter | value of $x_{3t}$ | $x_{4t}$ | $x_{5t}$ |
|---|---|---|---|
| 1 | 1 | 0 | 0 |
| 2 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 |

← both 0 for quarter 4

$$y_t = \beta_0 + \beta_1 x_{1t} + \beta_2 x_{2t} + \beta_3 x_{3t} + \beta_4 x_{4t} + \beta_5 x_{5t} + \varepsilon_t$$

Above graph shows that forecasted values matched real values very closely by bringing seasonality into the model (was explained & proved by the professor using R coding)

## Forecasting Techniques

- Different forecasting techniques are developed based on different logics.
- Simple techniques like moving average and exponential smoothing predict the future value of a time-series data as a function of past observations. (or weighted mov
- Regression based models such as auto-regressive (AR), moving average (MA), auto regressive

Page Date

## Moving Average method

and moving average (ARMA), auto regressive Integrated moving average (ARIMA), and ARIMA with X (ARMAX) use more sophisticated regression models to forecast the future values of a time series data.

- Important tip is that using complex mathematical models does not guarantee more accurate forecast.
- Simple moving average technique may outperform complex ARIMA models in few cases

## Forecasting accuracy

Some metrices are required to compare and confirm the best model for a particular dataset

- Frequently used forecasting accuracy measures are?

* Mean absolute error
* Mean absolute percentage error
* Mean squared error
* Root mean squared error

Let $Y_t$ is the actual value of Y at time t and $F_t$ is the corresponding forecasted value. Using these two values, we can calculate difference i.e. error

Assume there are $n$ observations in the dataset

eg $n = 100$.

Page Date

## Mean absolute error (MAE) -

$$MAE = \frac{1}{n} \sum_{t=1}^{n} |Y_t - F_t|$$

MAE is the average "absolute" error and should be calculated on the test data set (average of all the errors)

## Mean absolute Percentage error (MAPE)

$$MAPE = \frac{1}{n} \sum_{t=1}^{n} \frac{|Y_t - F_t|}{Y_t} \times 100\%$$

- It is the average of all the errors in terms of percentage so it is easy to use and interpret.
- Widely used, since it is dimentionless, it can be used for comparing different models with varying scales
- Suppose my model has MAPE 5% and Siddharth model has MAPE 6%, we can say, irrespective of different data sets, my model has better accuracy.

## Mean Squared error

$$MSE = \frac{1}{n} \sum_{t=1}^{n} (Y_t - F_t)^2$$

- Lower MSE implies better prediction
- However, it depends on range of TS data.

Page Date

## Root mean squared error (RMSE)

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(Y_t - F_t)^2}$$

$$= \sqrt{MSE}$$

- RMSE and MAPE are 2 most popular accuracy measures of forecasting.
- RMSE is the standard deviation of errors or residuals
- For survey or any challenge ~~any~~ kind of data, one can set the target for a specific value of RMSE.
- MAPE 5% gives good information. However RMSE 5 units is just a ~~not~~ number for error info.

## Moving Average and weighted average method

- MA is one of the simplest forecasting techniques which forecasts the future value of a TS data using average (or weighted average) of past N observations
- Mathematically, a simple moving average is calculated using the formula

$$F_{t+1} = \frac{1}{N}\sum_{t+1-N}^{t} Y_k$$

- It is SMA, as N past observations are given equal

weights, i.e. $\frac{1}{N}$

- In weight moving average, past observations are given differential weights (usually latest data gets more weight and oldest data gets least weight)
- Weighted moving average is given by

$$F_{t+1} = \sum_{K=t+1-N}^{t} W_K * Y_K$$

where $W_K$ is the weight given to value of Y at time K ($Y_K$) and $\sum_{K=t+1-N}^{t} W_K = 1$

In other words, total weights should be equal to 1.

- Eg, we want to predict sales of 13th month based on 12 month data.

SMA function from TTR pkg

for SMA - $F_{12+1} = F_{13} = \frac{1}{12}(S_1 + S_2 + \dots + S_{12})$

$F_{14} = \frac{1}{12}(S_2 + S_3 + S_4 + \dots + S_{13})$

for WMA - $F_{13} = 0.5 S_{12} + 0.3 S_{11} + 0.1 S_{10} + \dots$

where all weights sum = 1

max importance is given to 12th month sales

for SMA - $F_{14} = \frac{1}{12}(S_2 + S_3 + \dots + S_{13})$

$F_{15} = \frac{1}{12}(S_3 + S_4 + \dots + S_{14})$

If we need to calculate based on last 12 months data, our average would move accordingly. Thats why it is called Moving Average method

Check wsb Sales Prediction.R

If data is fairly static, MA or WMA gives good forecast (as no random variations in data so averaging works)

Page Date

## Single Exponential smoothing (ES) (Winters, 196

- One of the drawbacks of SMA is that it gives equal weight to all previous observations to forecast future value. Also how to conclude that 12 or any other number of
- Drawback of weighted moving average is, you need to assign different weights to past observations
- To overcome these problems, we use single Exponential Smoothing (SES) technique
- Like MA, SES assumes fairly steady TS data with no specific trend, seasonal or cyclic components

Imp- In SES, weights assigned to past data decline exponentially with the most recent observations assigned higher weights than old observations

- If our data does not have any trend, cyclicity or seasonality and is random in nature, it is best to use MA, WMA or SES.

$$F_{t+1} = \alpha Y_t + (1-\alpha) F_t$$

- The parameter $\alpha$ in the above equation is called smoothing constant and its value lies between 0 and 1.
- Since model uses only one smoothing constant, it is called single exponential smoothing (SES)

- Substituting for $F_t$ recursively in the above equation we get

$$F_{t+1} = \alpha Y_t + \alpha(1-\alpha) Y_{t-1} + \alpha(1-\alpha)^2 Y_{t-2} + \cdots + \alpha(1-\alpha)^{t-1} Y_1 + (1-\alpha)^t F_t$$

- Here we can see that the weights assigned to older observations decreases exponentially i.e from $\alpha$ to $\alpha(1-\alpha)$ to $\alpha(1-\alpha)^2$ -- to $(1-\alpha)^t$

(of observation should be taken for average)

Eg:

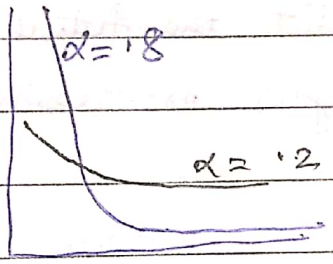
12 month sales ↑ — 12th month forecast sales. ↑

$$F_{13} = \alpha S_{12} + (1-\alpha) F_{12}$$

$$F_{12} = \alpha(S_{11}) + (1-\alpha) F_{11} \qquad \text{Say } \alpha = .90$$

90% of 11th month sales — 10% of forecasted value of 11th month

- So this model considers all past observations, though weights may be different.
- If $\alpha = .1$, it would decrease exponentially very slowly
- for $\alpha = .2$, we are giving 1Q of importance to history. For $\alpha = .8$, we are giving more importance to latest data.



Page Date

# SMA disadvantages and SES advantages

## SMA disadvantages –

- Increasing n makes forecast less sensitive to changes in data
- It always lags behind trend as it is based on past observations, the longer the time period n, the greater the lag as it is slow to recognize the shifts in the level of data points.

Imp
- Not suitable for trend and seasonal pattern data, as it starts exhibiting forecast bias and systematic errors

## SES Advantages –

- Uses all historic data, unlike the moving average where only the past few observations are considered to predict the future value (eg. 12 months MA)
- It assigns progressively decreasing weights to older data

## Optimal Smoothing constant in a SES ($\alpha$)

- Whenever data is smooth (without much fluctuations), we may choose higher value of $\alpha$.
- Whenever data is highly fluctuating, choose lower value of $\alpha$.
- Find optimal value of $\alpha$ by solving non-linear

No need to do optimization problem. There is a package in R which can directly be used. Also Holt winter can also be used for getting $\alpha$

→ optimization problem (subject to constraint $0 < \alpha < 1$)

$$\text{Min}_{\alpha} \left[ \sqrt{\frac{1}{n} \sum_{t} (Y_t - F_t)^2} \right]$$

For WSB Sales data, optimal value of $\alpha$ that minimizes RMSE is 0.1574 and corresponding RMSE is 739399.76

## Double exponential Smoothing (Holt's method)

→ SES drawback - it does not do well in the presence of ~~noise~~ trend

→ So we introduce one more equation for capturing trend in the time-series data

→ Double exponential smoothing uses 2 equations to forecast future values of TS, One for forecasting the level (short term average value) and another for capturing trend.

Equation 1: Level (or intercept) or Average [Level is nothing but average (SMA)]

$$L_t = \alpha Y_t + (1-\alpha) F_t$$

Equation 2: Trend

$$T_t = \beta (L_t - L_{t-1}) + (1-\beta) T_{t-1}$$

$\alpha, \beta$ - smoothing constants $0 < \alpha, \beta < 1$

Forecast at time $t+1$ is given by

$$F_{t+1} = L_t + T_t$$

$$F_{t+n} = L_t + n T_t$$

To initialize,

$L_1 = Y_1$

$T_1 = Y_1$

where $L_t$ = level which represents the smoothed value upto and including the last date,

$T_t$ = slope of the line (Rate of increase/decrease at period t)

$n$ = number of periods into the future

Ex

Initial values : $L_t = Y_t$

$$T_t = (Y_t - Y_{t-1}) \quad \text{or} \quad \frac{(Y_t - Y_1)}{t-1}$$

For WSB Sales data

$$L_1 = Y_1 = 3002666$$

$$T_1 = \frac{Y_{36} - Y_1}{35} = \frac{4732677 - 3002666}{35}$$

$$= 49428.88$$

$$F_2 = L_1 + T_1 = 3002666 + 49428.88$$

$$= 3052095$$

Calculating forecasting values from 37 to 48 months by using $\alpha = 0.0328$, $\beta = 0.9486$

2 ways of initializing $T_1$

$T_1 = Y_1$

$T_1 = \frac{Y_{36} - Y_1}{35}$

Initialization is very imp for the model to perform good.

→ SMA, WMA and SES uses past data to average out. So trend is lost. Only average (or level) was captured

→ But in DES (Holt), trend is captured along with levels

## Triple Exponential Smoothing (TES) Holt Winter

MA and SES / DES handles averaging and Trend but they do not handle seasonality in data.

Above methods indicate systematic error patterns (mainly two errors with — no errors at fixed intervals, (actual − forecast)). This kind of error pattern implies Presence of seasonality

So TES is used

**BASS model** - if a new product is being launched what would be the forecasted sales based on various attributes. BASS diffusion model (by Frank Bass)

Eq.1 Level (or intercept) equation

$$L_t = \alpha \frac{Y_t}{S_{t-c}} + (1-\alpha)\left[L_{t-1} + T_{t-1}\right]$$

Eq.2 Trend Equation

$$T_t = \beta (L_t - L_{t-1}) + (1-\beta) T_{t-1}$$

↑ Trend index

Seasonal equation

Eq.3 $$S_t = \gamma \frac{Y_t}{L_t} + (1-\gamma) S_{t-c}$$

↑ Seasonal index

For TES, directly use Holt Winter fn. in R
It would directly give optimum value of $\alpha, \beta, \gamma$ and various coefficients as well as fitted values, Holt Winter forecast gives Confidence Interval also.

# These kind of models are mainly suitable for static data. for high frequency data (where data is coming every 1 hr, different kind of models are used)

Multiplicative model ↓

$$f_{t+1} = [L_t + T_t] \times S_{t+1-c}$$

where c = number of seasons
if monthly seasonality, c = 12
if quarterly seasonality, c = 4
if daily data, c = 7

To predict seasonality index using method of averages
Step 1 - Calculate average of value of Y for each season
$\bar{Y}_1, \bar{Y}_2, --$ for each season
Step 2 - Calculate average of season's average, $\bar{\bar{Y}}$
Step 3 - Seasonality index for season $k = \frac{\bar{Y}_k}{\bar{\bar{Y}}}$

Interpretation of Seasonality index - can be understood as % change from trend line
eg - Seasonality index for Jan = 1.088 = 108.8% → In Jan, demand will be approximately 8.8% more from trend line.
Seasonality index for March = 0.8885 = 88.85% → means, demand in March would by 11.15% less than the trend line.

# Exponential smoothing techniques are very sensitive to initial values of level, trend & seasonality inde

In this case, best model proved out to be Linear Regression model where Sales was regressed against Promotion Expenses and Competitor Promotion.

LR gave better accuracy than MA, SES, DES, Holt Winter etc.

Use Regression on time when trend is the most pronounced.

Another way of incorporating seasonality is to take the trend prediction and actual value.

Depending on additive or multiplicative model, compute the deviation and map it as seasonality effect for each prediction.

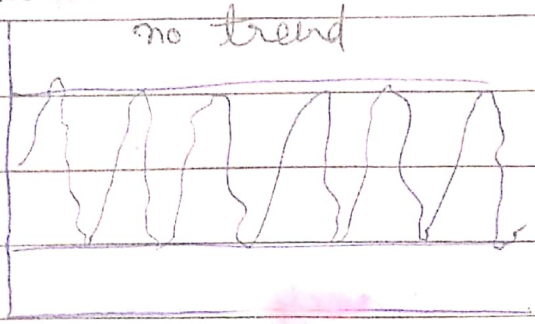
Multiplicative seasonality means going wider and wider with time (expanding)

Holt winter method allows to choose seasonality as either additive or multiplicative, Take, whichever gives best MAPE

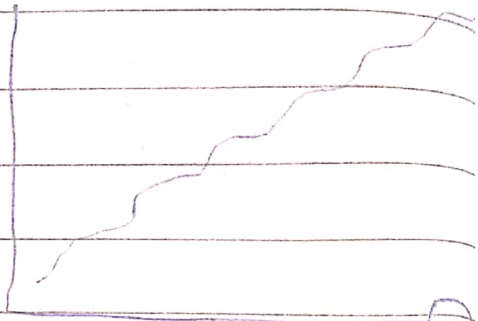
Additive seasonality means ~~being~~ being constant with time

Page Date

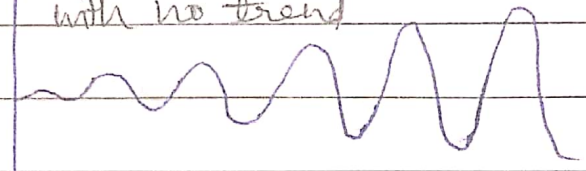
Additive seasonality with no trend



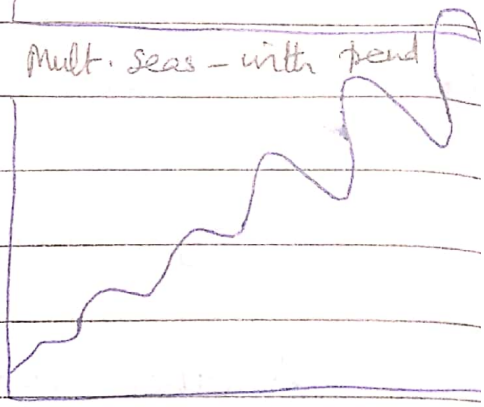
Add. seasonality with tre



Multiplicative seasonality with no trend



Mult. seas – with trend



If there is no trend or if seasonality and fluctuations are more important than trend, then Regression on time's coefficients behave weirdly

# ARIMA

Page Date

## Box Jenkins Methodology

- Individual forecasting methods AR and MA are combined to create ARMA and ARIMA models
- ARMA models are regression models — AR means a variable is regressed on itself measured at different time periods
- Fundamental assumption of AR — Time series is stationary process. (Constant mean/std over pd of time)
  - If TS data is not stationary, we have to convert it to stationary time series data before applying AR.
  - Constant mean/std means if 50 records are picked randomly, multiple times, they should have similar mean/stddev

## Limitations of Holt Winter

- It did not worry much about the correlation b/w current month sales (y var) and previous month sales.
- Relationship was studied algebraically and not from data relationship (correlation)
- What is the guarantee that my current month sales would be based on previous month sales.

Frequency - 252 for stock data (only wkdays, not wkends)

Page Date

| | Sales |
|---|---|
| 1 | 10 |
| 2 | 15 |
| 3 | 20 |
| 4 | 18 |
| 5 | 15 |

Considering $x$ = Sales [1:4]

$y$ = Sales [2:5]

trying to regress current sales on previous sales

Now if Cor $(x, y)$ is high say $> 0.7$, we can say current sales is strongly dependent on previous month sale (Auto correlation)

**Auto correlation** - Correlation with itself with different lag.

In above case it is autocorrelation with lag 1.

$\text{Sales}_{\text{Jan 2018}} = \alpha + \beta \text{ Sales}_{\text{Jan 2017}}$ here lag is 12 months

**Stationary data** - has constant statistical properties
- mean, variance, autocorrelation etc, over time
- for stationary data, forecasting is easy

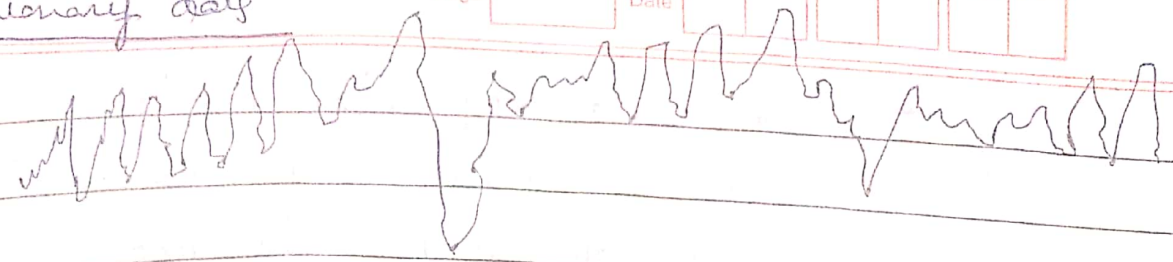
**Differencing technique** - to convert non stationary to stationary
- Generally 1 or 2 differencing are enough to convert a series to stationary

$\text{Sales}_{t,d} = \text{Sales}_t - \text{Sales}_{t-1}$ - first difference

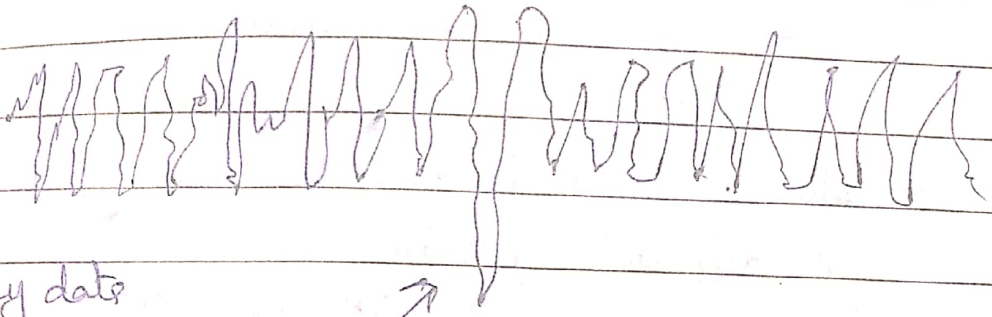
$\text{Sales}_{t,2d} = \text{Sales}_{t,d} - \text{Sales}_{t-1,d}$ - second difference

99% cases, it would convert to stationary

non stationary data




miles.d = diff(miles)




miles.d2 = diff(miles.d) ← second diff.

If many outliers in first diff, go to 2nd diff.

## AR model

$$\hat{y}_t = f(y_{t-1}, y_{t-2}, y_{t-3}, \dots, \epsilon_t)$$

AR(p) – when AR model depends on p past values, where lagged values having significant relationship with most recent value,

$$\hat{y}_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_p y_{t-p} + e_t$$

* Imp factor – how to decide value of p (how many past values to consider). Can't regress against all values in data.

→ for this example, p = 12 and $\beta_{1,1} - \beta_{11} = 0$

Page Date

## Imp task in auto Regression model

- Identify the values of p (number of lags)
- p only determines right model ($y = f(x)$ equation) to be used for forecasting
- Standard approach used for model identification
  - ACF (Auto correlation function)
  - PACF (Partial auto correlation function)
- Auto correlation is correlation b/w $y_t$ measured at diff time periods
- Say auto correlation with lag 1 ($\gamma_1$) = Cor($y_t, y_{t-1}$) = 0.9
- $\gamma_2 = cor(y_t, y_{t-2}) = 0.8$
- $\gamma_{12} = cor(y_t, y_{t-12}) = 0.99$
- Here present sales is strongly correlated with 12 months old sales, so we would build the model accordingly
- $\gamma_3 - \gamma_{11} = 0$

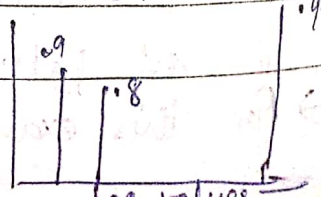
Now diff models like, model 1 = $y_t = \alpha + \beta_1 y_{t-1}$

model 2 = $y_t = \alpha + \beta_1 y_{t-1} + \beta_2 y_{t-2}$

model 3 = model 2 + $\beta_{12} y_{t-12} = y_t$

model 4 = $y_t = \alpha + \beta_{12} y_{t-12}$

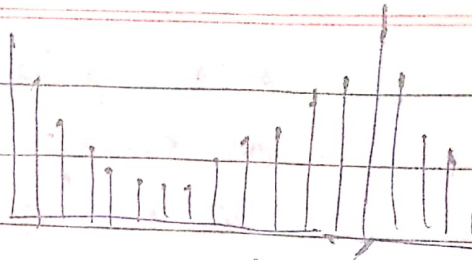
**ACF / Correlogram** — Plot of autocorrelation for different values of K is called ACF or correlogram



Page Date

Function in R –

acf( miles time series)




Here 12th lag shows significal correlation

**PACF** – PACF of lag K, $\rho_{pk}$ is the correlation b/w $Y_t$ and $Y_{t-k}$ when the influence of all intermediate values ($Y_{t-1}, Y_{t-2}, \dots Y_{t-k+1}$) is removed from both $Y_t$ and $Y_{t-k}$

Removing influence by other $y_t$s.

AR(1) $y_t = \beta_0 + \beta_1 y_{t-1}$ PACF(lag 1) = $\beta_1$

AR(2) $y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2}$ PACF(lag 2) = $\beta_2$

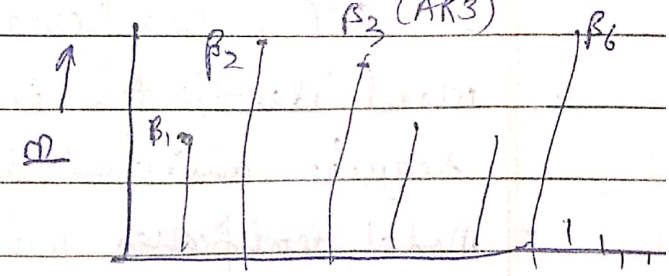
AR(p) $\beta_p$

if $|\beta_p| <$ small for all $p > 12$

that means all data points $y_{t-13}$ – – are insignificant. So need to take only from p = 1 to 12

So PACF is an iterative process for model selection

PACF graph




here after $\beta_6$, $\beta$s are not significant so consider only significant ones in your equation

Page Date

ACF gives the confidence to select based on strong correlation
PACF gives the order of selection

- Plot of PACF values for diff k is called PACF graph
- Hypothesis tests can be carried out to check whether the auto correlation and partial auto correlation values are different from zero
- Null and alternate hypothesis
  $H_0: \rho_k = 0$
  $H_a: \rho_k \neq 0$
- Null hypothesis is rejected when $|\rho_k| > \frac{1.96}{\sqrt{n}}$ and $|\rho_{pk}| > \frac{1.96}{\sqrt{n}}$ (95% quantile)
- To select appropriate p in AR model, following thumb rule may be used
- Number of lags is p when
  - the partial autocorrelation, $|\rho_{pk}| > \frac{1.96}{\sqrt{n}}$ for first p values (first p lags) and cuts off to 0
  - ACF, $\rho_k$ decreases exponentially
- Model identification is an iterative process and may require additional inputs
- Model identification using ACF and PACF can not be taken as conclusive evidence for number of lags in AR process
- Domain knowledge plays a key role.