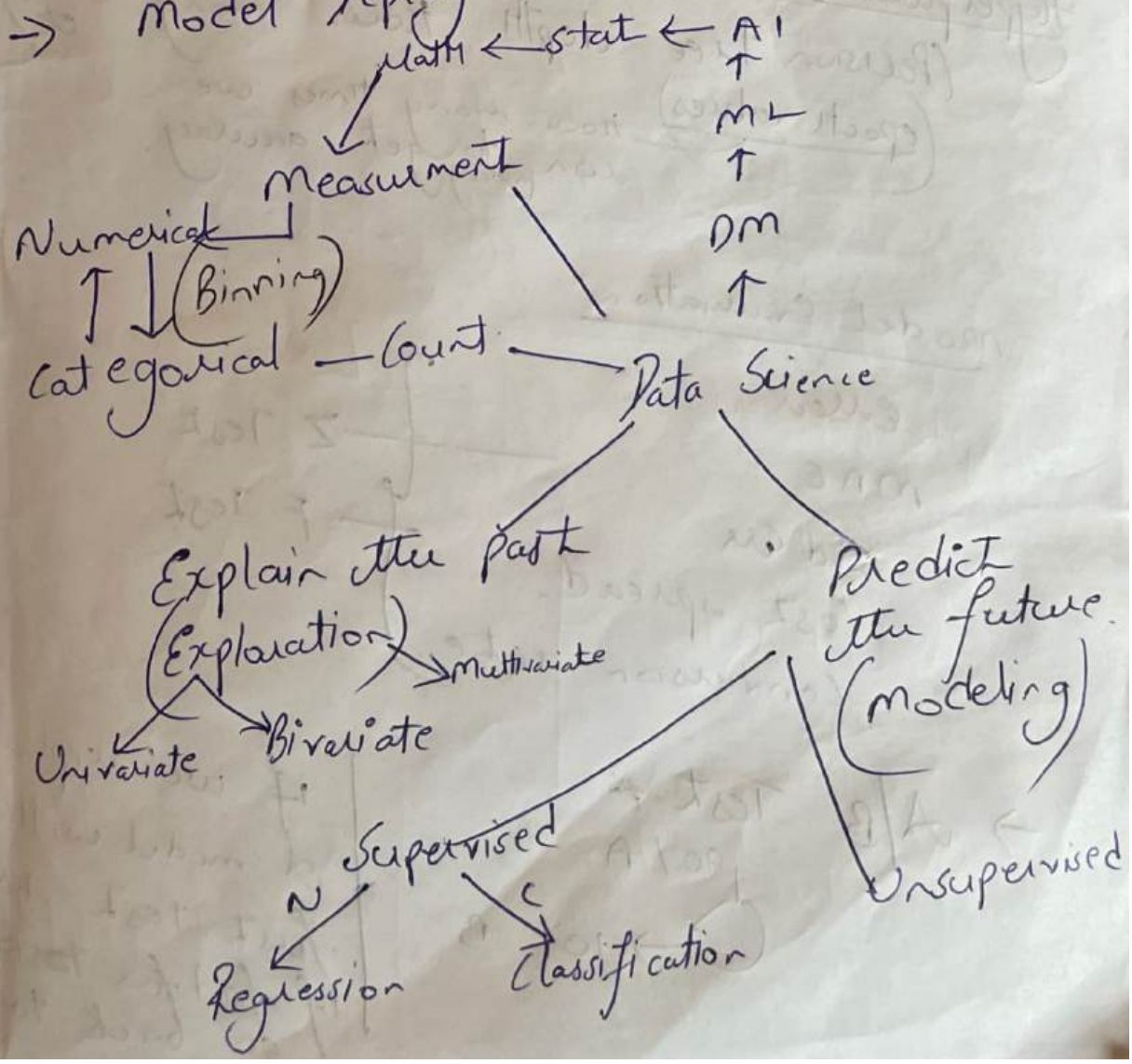
Day-1 [11L116]

Tell story in right way with the data

6 steps ★★★

→ Problem defination + Criteria of Success
→ ~~Criteria~~ How to get data (Data preparation)
→ Data Exploration (with statistics visualization)
→ Modeling (Regression / Classification)
→ Model Evaluation
→ Model Deployment



Discovery → (junction details was completely empty)

Data present in 3 files & it had missing values in it

merge - 3 datasets to 1

(2005-2007) (2009-11) (2012-14)

Structuring

Removed the null values

The column names had special characters which were replaced

Cleaning

Checked all the ~~columns~~ datatypes, Only date ~~,~~ was a character column, changed it to date

Enriching

Answering 3rd question: (Risk factor on road)

we needed columns like

No of fatilities, accident severiety, frequency of fatilities

followed Repeated Discovery, structuring & Cleaning for this data set

& added to our data set

Validation

We then validate our data &

Publish it in

(↓

Data Normalization (Structuring / Transforming)

Reduces data redundancy & improves data Integrity → [Data stored in logical sense]

Anomaly
- Insertion
- Updation
- Deletion

} Example [Employee & department tables]

1 NF

→ Single valued - each cell
→ Entries of column are of same type
→ Row is uniquely identified.

Remove repeating groups - finding entities
- Create seperate table for each entity
- Identify primary key for each table

Emp Id	Name	Phone	Salary
		a1087, a1126	30,000 HOK

2 value

→ Atomicity → each column should have single value

2 NF

→ In 1NF
→ [All attributes depend on key]

should not contain partial dependency

Employee id	Dept id	office location

Dept id → Prime attributes
office location → Non prime Attribute

office location only depends on dept id

→ Split table into 2

Emp id	Dept id

Dept id	location

3NF → In 2NF

There should be no transitive dependency for non-prime attributes

All non prime attributes should only depend on prime attributes

$A \rightarrow C \rightarrow D$ then $A \rightarrow D$

Eg

Student id	Student Name	Subject id	Subject	Address

Student id → subject id → subject

∴ student id → subject

Transform to

Stud id	Subject id	Stud. Name	Address

Subject id	Subject

BCNF (or) 3.5 NF [Boyce codd normal form]

→ In 3NF

Every functional dependency $A \rightarrow B$ then A has to be super key for that table

Eg

student id	subject	Professor

A proffesor can teach multiple subjects

Student id	Professor id

Prof id	subject	Professor

(Super key)

Remove non prime attributes functional dependency

BCNF

→ No multivalued dependency.

Cust id	Cust Name	Address	Newsletter
20	Alan	10 Paul Rob	NYC times
20	Alan	10 Paul Rob	TOI
21	Smith	HIC Phleps Av.	TOI

~~if i am b~~

One can subscribe to multiple newsletters & one can have multiple shipping address

So we have multiple rows of cust name & cust id data.

~~many → 1~~

1 → 1

1 → many

many → many

Problem defination

Understand domain perspective & convert this to data science problem

Example of ~~customer~~ hiring a server for my restaurant

Data ⇒ Bill amount, tip amount

Linear regression

X → Bill amount

Y → Predict tip amount

on avg per month 200 tables

→ total bill (X) ⇒ use model to predict & lure them in job offa

Data preperation

Data wrangling / Data munging → (weight)

Data
- Numerical
  - Ratio → (weight)
  - Interval → Temperature
- categorical
  - Ordinal → (low, med, high)
  - Nominal → yes/no, gender

Database

DBMS

↳ ODBC → Open database connectivity

JDBC → Java data base connectivity

SQL (Data defination language)

DDL

→ create, alter, drop table

create index, drop index

DML → (Data manipulation language)

select

insert into

update

delete

6 steps of Data preperation

Discovery → Structuring → Cleaning

Enriching ← validating ← Publishing

(fact table)

Data Exploration

Explaining the past

→ Univariate

→ Bivariate

Categorical (binning)

Numerical (encoding)

binning

1Φ–30 }
31–50 }

Encoding			
up	1	0	0
up	1	0	0
down	0	1	0
flat	0	0	1

Categorical

→ frequency counts, bar & pie charts

Numerical

count, mode, variance, sd, skewness, → Histogram

min, max, range, mean, median, quantile → Box plot

Bivariate analysis

→ Numerical – Numerical

Scatter plot

Linear correlation

$$r = \frac{\text{Covariance}(x, y)}{\sqrt{\text{var}(x), \text{var}(y)}}$$

If r is –1 negative correlation

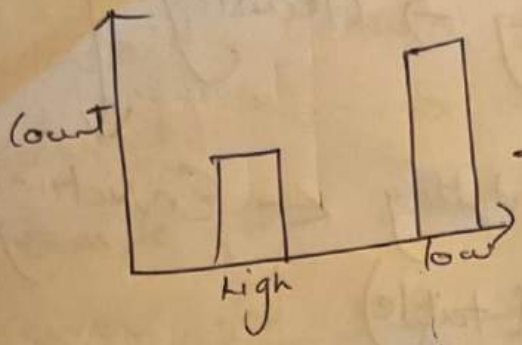
& if it is +1 positive correlation

$$\text{Cov}(x,y) = \frac{\Sigma(x-\bar{x})(y-\bar{y})}{n}$$

$$\text{var}(x) = \frac{\Sigma(x-\bar{x})^2}{n}$$

$$\text{var}(y) = \frac{\Sigma(y-\bar{y})^2}{n}$$

Categorical – Categorical → Stacked colum
→ Combination ch
→ Chi square test



→ 2 columns are

failures	chances
High	Yes
low	No.

★ Chi square test ★

Difference b/w expected frequency (e) & observed frequency (n)

→ Returns a probability for computed chi square & degree of freedom

(Total values − 1)

p = 1 Two categorical variables are completely independent

p = 0 completly dependent

$$\chi^2 = \sum_{i=1}^{r} \sum_{1}^{c} \frac{(n_{ij} - e_{ij})^2}{e_{ij}}$$

$$p = \sqrt{\frac{\chi^2}{n\sqrt{(c-1)(r-1)}}}$$

$$df = (r-1)(c-1)$$

$$e_{ij} = \frac{n_i \, n_j}{n}$$

r, c ⇒ rows, columns

Categorical – Numerical

Visuals → line charts
→ Combination charts

→ Z test, t test, ANOVA.

Z test & T test

Access if averages of 2 groups are statistically different from each other.

$$Z = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}}}$$

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{S_2\left(\frac{1}{N_1} - \frac{1}{N_2}\right)}}$$

$\bar{X}_1$, $\Rightarrow$ averages

$S \Rightarrow$ variance.

$N \Rightarrow$ counts

$$S^2 = \frac{(N_1-1)S_1^2 + (N_2-1)S_2^2}{N_1 + N_2 - 2}$$

(we use t test if N_1 (or) N_2 are less than 30)

If probability of Z/t is small, difference b/w 2 averages is more significant.

ANOVA - (Analysis of variance)

Averages of more than 2 groups are different statistically from each other

	df	SS	MS	F	P value
Regression	k	SSR	MSR = SSR/df	MSR/MSE	P(F)
Residual	n-k-1	SSE	MSE = SSE/df		
Total		SST			

$$SSR = \Sigma(y' - \bar{y})^2 \quad SSE = \Sigma(y - y')^2 \quad SST = \Sigma(y - \bar{y})^2$$

Coefficient of determination $R^2 = SSR/SST$

Missing values - Rectification

Numerical	Categorical
1) Average	Missing value as a seperate category
2) Mean	Majority value.
3) Median	

Numerical Binning - (Bins of equal width)

Temp	Temp-bin	Play golf
85	80-90	N
80	80-90	N
70	70-80	Y
68	60-70	Y

Numerical variable

Assumption that it has a gaussian probability distribution (pdf) - mean & sd.

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

Simple / multiple Regression

SSE - Sum of Squared Errors.

Goal is to create a linear model that minimizes the sum of squares of residuals (SSE)

Reduce SSE

(Comparision with only dependent variable)

Linear Regression Assumptions

→ Linear Relationship.
→ No Multicollinearity
→ Multivariate normality - (Residuals are normally distributed)
→ No autocorrelation
→ Homoscendasticity - (Same variance)

X_1	X_2	X_3	Y
Number of vehicles	number of casualities	Casuality Severity	Accident Severity

if we dont have any data of only accident severity then we predict the next accident severity by taking a mean of accident severity

SSE - Sum of square of residuals

⇒ Any best fit line would reduce this SSE

Step 1 - Scatter plot

b/w Accident Severity & Casuality severity

Step-2

find correlation coefficient

(It shows if there is any strong +ve correlation b/w 2 columns)

Centroid (mean(x), mean(y))

Best fit will pass through centroid

$\hat{y} = b_0 + b_1 x_i$ → slope.

$SSR = \boxed{SST}$ (with mean) $- \boxed{SSE}$ (with best fit line)

$r^2 = \frac{SSR}{SST}$ — **Coefficient of Determination**

Multiple Regression

Adding more IV to multiple regression does not mean the regression will get better. It can make things more worse. (Overfitting)

Multicollinearity (Variance inflation factor values)

More IV creates more relationships b/w them. If IV's are related to each other then

Ideal → IV to be correlated to dependent variable but not with each other

This leads to wider confidence intervals & they lower the statistical significance of regression coefficients

Do correlation, scatter plot & simple regression for each IV & to Avoid overfitting & multicollinearity.

Multiple Regression Model $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_p x_p + \varepsilon$ (error)

Multiple Regression Equation $E(y) = \beta_0 + \beta_1 x_1 + \cdots \beta_p x_p$

Estimated Multiple Regression Eqn $\hat{y} = b_0 + b_1 x_1 + \cdots b_p x_p$

$b_0, b_1, b_2 \ldots$ estimates of $\beta_0, \beta_1, \ldots \beta_p$

$\hat{y}$ = predicted value of dependent variable

variance inflation factor (VIF)

$$C_{ii} = \frac{1}{1 - R_i^2}$$

> 5 (Multicollinearity is a problem)

Example

93% of the total sum of squares can be explained by using the estimated regression equation.

(1) SSR ⇒ Variance explained by the predictors

SST ⇒ Variance that has to be explained by predictors (Independent variables)

SSE - Variance not explained by the predictors.

Radjusted, C is column vector (element-wise product)

n - number of items in the set

k - means no of cell means or groups (No of columns)

$F = \frac{\text{explained variance}}{\text{Unexplained variance}}$ (or) $\frac{\text{b/w group variability}}{\text{within group variability}}$ variance

T-test

To check if 2 sets of data are significatily different from one another. A null hypothesis is used to test for significant difference.

Confidence interval for regression coefficients is

An interval that contains β_i (true) in 95% of all samples

VIF estimates how much the variance of regression coefficient is inflated due to multicollinearity in the model.

T test

$$T \text{ value} = \frac{\text{mean}(1) - \text{mean}(2)}{\sqrt{se^2 \, C_{ii}}}$$

if t is greater than 1, there is more signal ie there is more noise

Null Hypothesis - No statistical difference between the samples

T critical value

if $t <$ critical dont reject

$t >$ reject

if β_1 in CI 95%.

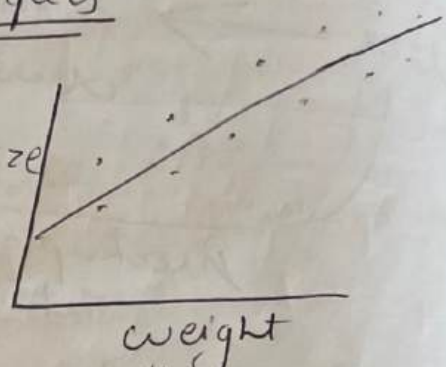
we calculate the P value.

Eg this would say if we have to reject the null hypothesis or

Ridge & Lasso Regression

Regularization Techniques

Least squares line



(what if we only have 2 points)

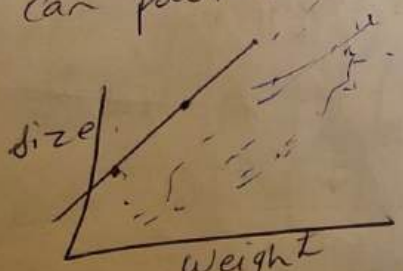
Training data - least square will be 0

But testing data has high error (overfitting)

So we introduce a bias to the line

How new line fits the data

By slightly worst fit, ridge regression can provide better long term prediction



Least squares method minimizes sum of squared residuals

L2 norm

Ridge regression minimizes sum of squared residuals + $\lambda * \text{the slope}^2$

penalty term

Lamda determines how severe the penalty is

it has a small amount of Bias

but the variance is low

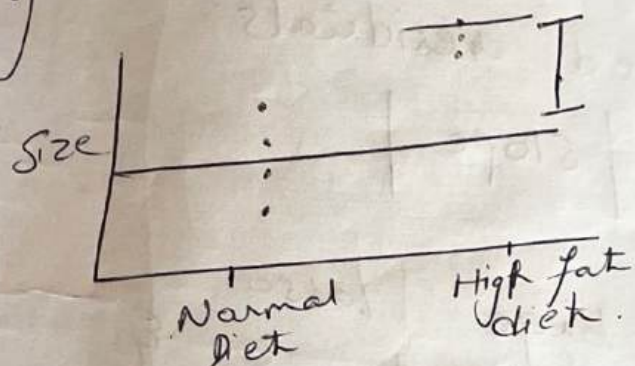
The slope of ridge regression is small, Hence less sensitive to weight than least squares line

$\lambda \Rightarrow 0$ to ∞

↓ sum of squares (least squares)

If λ is large model becomes less sensitive to weight

10 fold cross validation — a High value of λ would give lowest variance.



Size = 1.5 + 0.7 × High fat

When applied to logistic regression ridge regression optimizes the sum of the likely-hoods instead of squared residuals

Sum of likelyHood + $\lambda * (\text{slope})^2$

⇒ Ridge regression — Every parameter except for the y intercept is scaled by the measurements

$\lambda (\text{Slope}^2 + \text{diet difference}^2 + \cdots)$

$x = y + \text{slope} \times \text{weight}$ → minimum 2 data point

if we have 100 parameters, we need 100 data points atleast to predict the parameters

Least squares can't find an optimal solution when we have less data points than parameters (gene data example)

But Ridge Regression can find a solution with cross validation & the regression penalty that favours smaller parameter values.

Lasso Regression

Sum of squared residuals + $\lambda \times |\text{slope}|$

Differences

when λ is large lasso can shrink the slope to 0 & Ridge can only shrink asymtotically close to 0.

$\lambda(|\text{slope}| + |\text{diet diff}| + |\text{heart rate}| + |\text{location}|)$

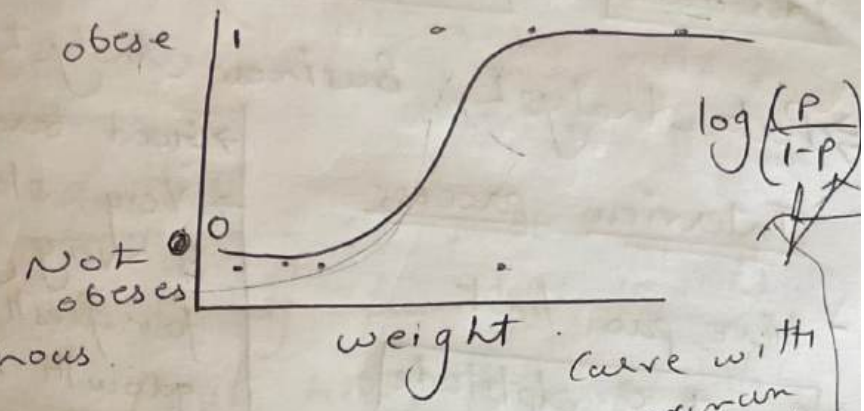
shrinks a little

becomes 0 removes the silly stuff

Lasso can exclude useless variables from equations, so little better than Ridge regression.

Ridge is better when all variables are useful.

Logistic Regression



[Predicts True | false instead of predicting something continous like size]

Used for classification

if > 0.5 obese.
< 0.5 not obese

Curve with maximum likelyhood is selected

Scale of coefficient is log(odds)

Probability.

Complicated models have more x values.

we check if the variables test to see if they are significantly different from 0.

(Walds test)

* gives probabilities & classifyies new samples using continuous & discrete measurements.

Big difference b/w linear & logistic

How line is fits to the data

Maximum likelyhood — Pick a probability, scaled by weight of observing an obese mouse - just like the curve above & use that to calculate likelyhood of all ~~not~~ obese mouse

Logistic Regression & Perception Algo

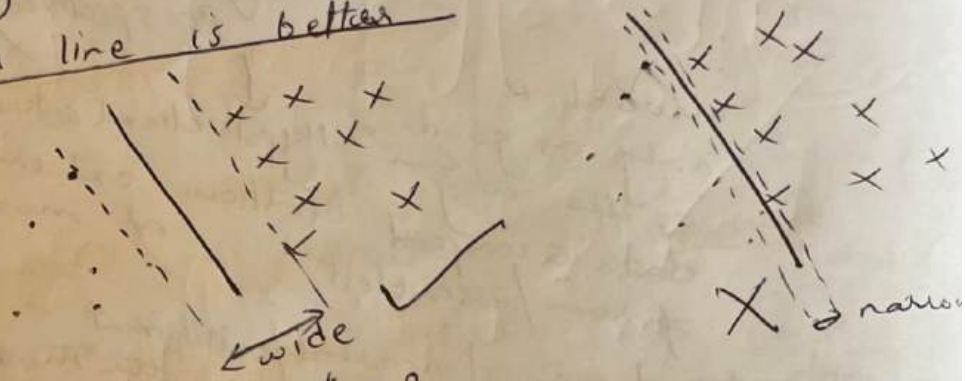
- Start with random line, with blue & red sides
- Pick a large no - 1000 (No of repetitions / epoch)
- Repeat 1000 time
  - Pick a random point.
  - if classified correctly
    continue.
    else.
    move towards the point.

Result - we get a line that seperates the data

Support Vector Machine (SVM)

It tries to seperate points of 2 classes using a line (Best line, farthest from pts)

Which line is better

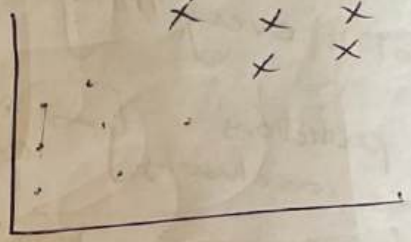


How to seperate line?

Eqn ⇒ $2x+3y-6=0$

$2x+3y-6=1$

$2x+3y-6=-1$



Expanding rate (0.99)

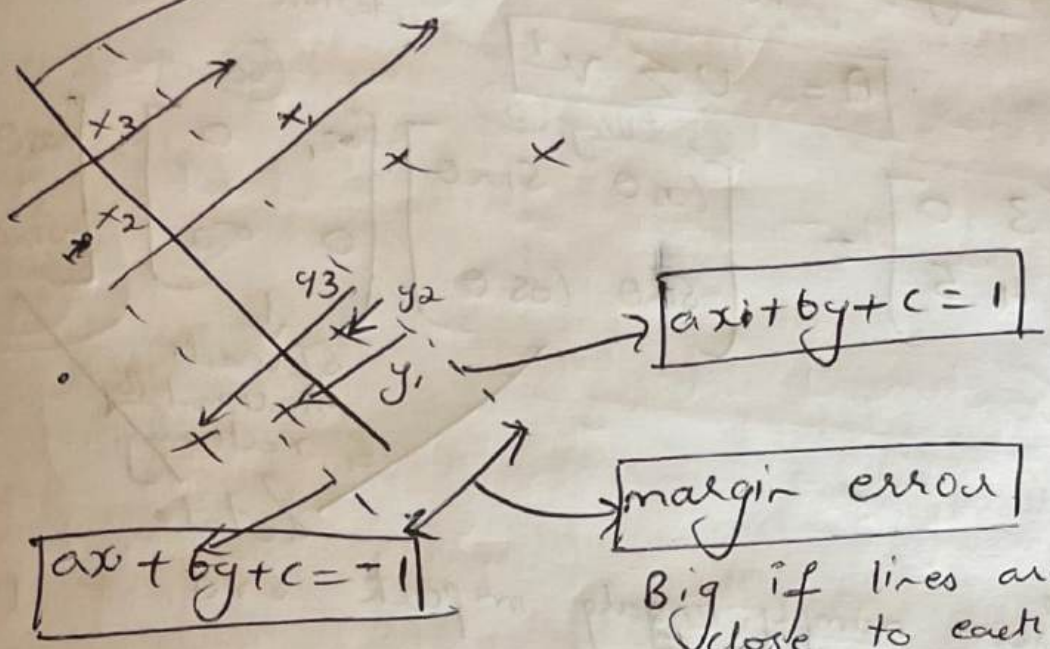
Algorithm

1) Start with a line Eq
   2 parallel lines to it
2) Pick a large no (No of repetitions / epochs)
3) Expanding factor - Pick close to 1
4) Repeat 1000 times:
   - Pick random point
   - If point is classified correctly
     continue

     else
     - Move line towards point.
     - Seperate lines using expanding factor

$$0.99(2x+3y-6)=1$$
$$0.99(2x+3y-6)=-1$$

Error calculation → $ax+by+c=0$



Big if lines are close to each other & vice-versa.

$$\text{Margin} = \frac{2}{\sqrt{a^2+b^2}}$$

$$\text{margin error} = a^2+b^2$$

$$\text{Error} = C[\text{classification error}] + \text{margin error}$$

$$\frac{d(\text{error})}{da} = 2a \qquad \frac{d(\text{margin error})}{db} = 2b$$

$\eta \Rightarrow$ learning rate

$$a \to a - \eta 2a = a(1-2\eta)$$

$$b \to b - \eta 2b = b(1-2\eta)$$

→ Expanding factor.

C parameter ⇒ we use it to choose if classification error is important or margin error is

Eg Large C ⇒ importance to classification error

Hyperparameters ⇒ C, expanding factor, learning rate

Singular Value Decomposition (SVD)

$$A = U \Sigma V^T$$

rotation → V^T (1), scalings → Σ (2), rotation → U (3)

$$\begin{bmatrix} 3 & 0 \\ 4 & 5 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

(3) Rotate & (2) Stretch horizontally & vertically (1) Rotate

```
from numpy.linalg import svd
```

Rank-1 matrix

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ -1 & -2 & -3 & -4 \\ 2 & 4 & 6 & 8 \\ 10 & 20 & 30 & 40 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \\ 10 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 4 \end{bmatrix}$$

16 numbers → 8 numbers

$$\begin{bmatrix} x & x \\ x & x \\ x & x \\ x & x \end{bmatrix} \begin{bmatrix} x & x & x & x \\ x & x & x & x \end{bmatrix}$$

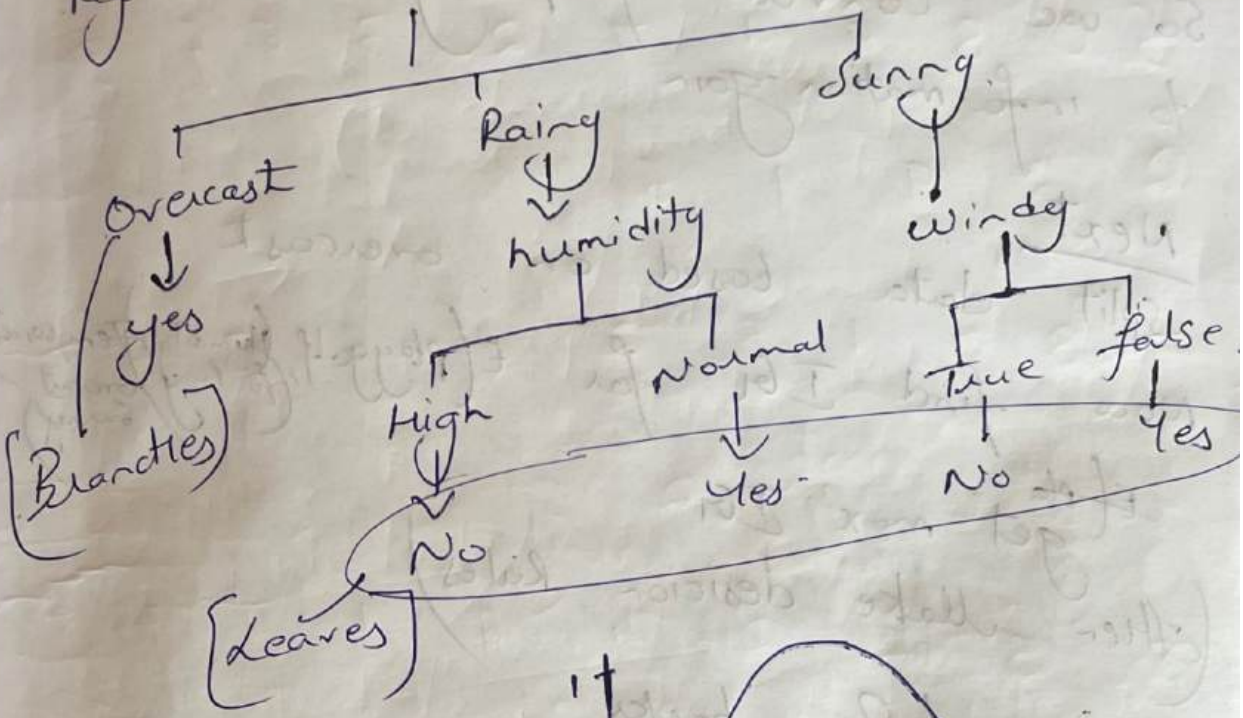
Rank 2

(ignore lower σ values → approximate original matrix)

Svd gives 3 outputs ⇒ U, Σ & V^T

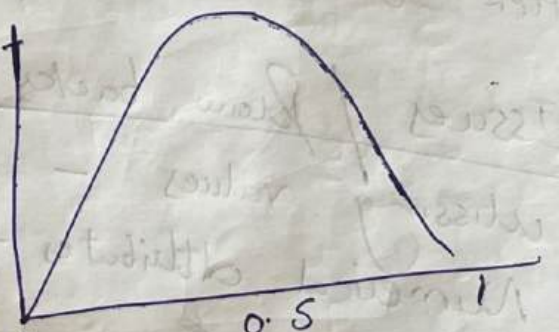
Decision Trees

→ Can build both classification & Regression models.



Entropy

$E = -p\log_2 p - q\log_2 q$



$= -0.5\log_2 0.5 - 0.5\log_2 0.5$

$\Rightarrow$

$\text{Entropy}(3,2) = \text{Entropy}(3/5, 2/5)$

$= -(0.6 \times \log_2 0.6) - (0.4 \times \log_2 0.4) = 0.971$

Frequency Table

Outlook	Play golf: Yes	Play golf: No
Overcast	4	0
Sunny	3	2
Rainy	2	3

$E(\text{Play golf}, \text{outlook}) = 0.693$

Information gain ⇒

$Gain(T, X) = Entropy(T) - Entropy(T, X)$

So we convert frequency table to information gain

Next:
Split data based on overcast!

Now find IG for E(playgolf, (Humidity, Temp, for Rainy / overcast Sunny)

get max IG

(Then Make desicion Rules)

Issues / Draw backs

- Missing values – 2 sol^n
- Numerical Attributes → Binning
- Overfitting & pruning
- Attributes with many values (Super Attributes)

gain(T, X) is higher for super Attributes & super attributes will easily be selected as root. To avoid this we calculate gain ratio

$Split(T, X) = -\sum P(x) \log_2 P(x)$

Split on least gini value

gain / Split

$1 - \sum (P_i)^2 = Gini$

Gini index – demonstrates degree of inequality in distribution

0 – only 1 class | 1 – Random dist | 0.5 – Equal dist

Descion Trees – Regression

~~SDR~~ =

Stop if coefficient of variation is $\leq 10\%$.

Average is assigned to the leaves

Split is done by calculating

~~SDR~~ Average $= \bar{x}$

S.D $\Rightarrow$ $s = \sqrt{\frac{\sum (x - \bar{x})^2}{n}}$

Coefficient of variation $(CV) = \frac{s}{\bar{x}}$

SD b/w 2 attributes

$$S(T, X) = \sum P(c)\, S(c)$$

Standard deviation Reduction

$$SDR(T, X) = S(T) - S(T, X)$$

Split on max SDR

Random forest

Binary classification problem (Eg)

Decision trees are highly sensitive to training data & thus could result in high variance

- Hence random forest - which is much less sensitive to training data
- We Bootstrap datasets → making multiple datasets. at random from 1 dataset
- Train decision tree on each of the bootstrap dataset independently (with feature subsets (random)) ← this reduces correlation b/w trees
- Parse new data through each tree & note down predictions
- We combine all the predictions & take majority predicted value as the answer

Bootstrapping + Aggregation (Bagging)

ideal size of feature size = log/sqrt of (features)

for regression problem - while combining the predictions, just take the avg

Bagging

DT1 DT2 DT3 DT4 DT5

Bootstrap + Aggregation

Split data - N Trees

majority of prediction are taken

Random forest

feature bagging - Randomly Select subset of p features dim at each split. This is to deliberately avoid features that lead to similar splits in DT. Therefore it reduces correlation b/w each DT. E ↑ predictive accuracy or average.

Use $\sqrt{p}$ features (Rule of thumb) E.g. consider at each split

The proportion of out of bag samples that were incorrectly classified is the out of bag error

Proximity Matrix

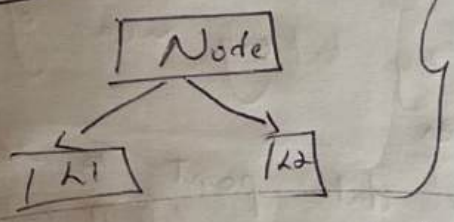
Test for a bunch of different setting to get correct p value for

Boosting

Idea is to iteratively learn weak machine learning models or a continual updated training data set & add them to produce final result.

Differs from bagging (simply take avg of entire models)

Ada Boost



Stump is a tree with one node & 2 leaves

Stumps are weak learners

In forest of stumps (Ada boost), some stumps have higher say than others

→ Order is important.

Calculate gini index.

Total error = $\frac{\text{Error}}{N}$

Amount of say for each stump = $\frac{1}{2}\log\left(\frac{1 - \text{Total error}}{\text{Total error}}\right)$

If stump does a good job AOS is large +ve value

New sample weight (To increase) = sample weight $\times e^{\text{amount of say}}$

(To decrease) = s.w $\times e^{-\text{amount of say}}$

→ Normalize it → random number (take subset) → New data

Batch Normalization

Attention

GB performs better on unbalanced data (dist...
Random forest builds ... → multi class ... detection

Gradient Boosting → Regression

- Boosting ⇒ converting weak learners to strong learners

⇒ Easy Explanation is Ada boost

3 - 32 Leaves (Trees are larger)

Step-1 - Calculate the average of target label (680)

Step-2 Calculate residuals for each sample

Residual = Actual - predicted value.

Step-3 ⇒ Construct D.T

Decision trees predict target variables, in GB it tries to predicts the residuals

If some residuals end up in same leaf we take the average.

Step-4

Predict target labels using all the trees within ensemble.

Predict value = Avg price + Learning rate × Residual predicted by DT

650

(Tunable bias but better variance) → small step in right direction

Step 5

Compute new residual

Actual value - Predicted value =)

Data 650

Step 6

Repeat 3 – 5 ~~until~~ for a given hyperparameter – No. of iterations (ie no of estimators)

(Until the additional tree does not significantly reduce the size of residuals)

Step 7

Once trained, we use all the trees in ensemble to make a prediction as to the value of target variable

Avg + 0.1 Residual of T1 + 0.1 × Residual of T2 + ...

Classification

Prediction is $\frac{\log(\text{odds})}{\text{Predicted}}$

Residual = Actual – Predicted

P(Y) = 1

P(N) = 0

Residual ⇒ (Yes)

1 – 0.7 ⇒ 0.3

&

0 – 0.7 ⇒ – 0.7

Residuals

Eg

Y – Yes

N – No.

$\log\left(\frac{4}{2}\right) = 0.7$ (initial prediction)

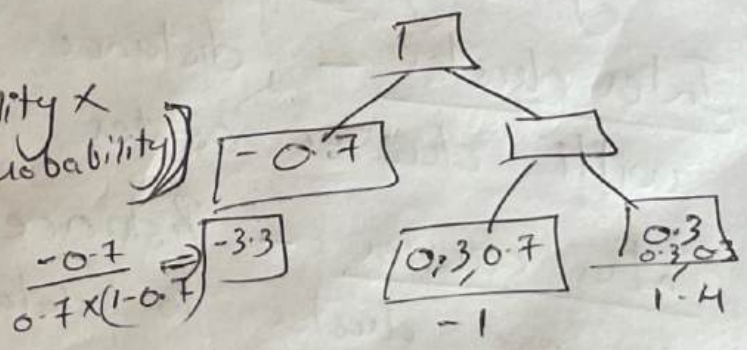
$P' = \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}$

P = 0.7

Tree

Σ Residual

$\Sigma \left[\frac{\text{Previous Probability} \times (1 - \text{Previous Probability})}{} \right]$



$\frac{-0.7}{0.7 \times (1 - 0.7)} \Rightarrow -3.3$

⇒ 0.7 + 0.8 × (Tree o/p) ⇒ 0.7 + 0.8 × 1.4 = 1.8

Calculate residual

Probability ⇒ $\frac{e^{1.8}}{1 + e^{1.8}} = 0.9 \rightarrow$ Yes

XGBoost

Extreme Gradient boost

Speed of processing, performance

Advanced - gradient boost

- parallelization
  - CPUs
- cache optimization
- Out of memory computation

- Each tree boosts attributes that lead to misclassifications of previous trees

- (Regularized boosting to avoid overfitting)

Hyperparameters - booster (gbtree gblinear, dart)

Verbosity - 0, 1, 2, 3 (quality of using more words than needed)

nthread - max number of threads to run XGBoost.

Time Series

(Daily / monthly / Quarterly / yearly)

Trend — linear / Non linear

Cyclical variation

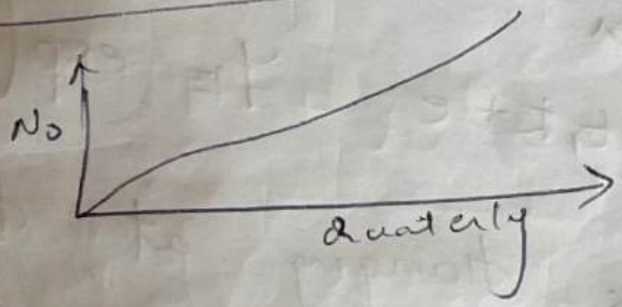
Seasonal variation

Residual variation.

$$y_t = T_t + C_t + S_t + R_t$$

Linear Trend

$$Y = b_0 + b_1 t + e$$



Increase by a constant amount at each successive time period - Linear trend

Increase by increasing amount - curvilinear Trend

Increase by equal percentages - can be made Linear by applying Logarithmic transformation.

$$\log(Y) = b_0 + b_1 t + e$$

Time Series model Building

⇒ What is the trend that we see?
- Linear - logarithmic - Polynomial

→ How can we smooth the data?

→ Should we model distinct seasonal patter

Non linear Trend

- Time series changing at increasing rate over time, then logarithmic model in Y works better

$$\log(Y) = b_0 + b_1 t + e \qquad Y = \exp(b_0 + b_1 t + e)$$

- Time series changing at decreasing rate over time, then logarithmic model in t works better

$$Y = b_0 + b_1 \log(t) + e$$

Curvlinear trends

Model with polynomial

Linear (first order) $\Rightarrow Y = b_0 + b_1 t + e$

Quadratic (second order) $\Rightarrow Y = a_0 + b_1 t + b_2 t^2 + e$

Cubic (third order) $\Rightarrow Y = a + b_1 t + b_2 t^2 + b_3 t^3 + e$

* Moving Averages *

Compute average of last m no consecutive observations.

They merely smooth the fluctuations in the data

Works well when data has
- fairly linear trend
- definite rythmic pattern of fluctuations

$$y_{ma(4)} = \frac{y_t + y_{t-1} + y_{t-2} + y_{t-3}}{4}$$

Exponential smoothing

It is a weighted average that assigns positive weights to the current values & to past values of the time series.

- Greater weights to recent values & weights decrease exponentially as users go farther back in time.

$$S_1 = y_1$$

$$S_t = \alpha y_t + (1-\alpha) S_{t-1}$$

* Smaller the value of α smoother the plot.

Eg $\alpha = 0.5$

$S_1 = y_1$

$S_2 = 0.5 y_2 + 0.5 y_1$

$S_3 = 0.5 y_3 + 0.25 y_2 + 0.25 y_1$

$S_4 = 0.5 y_4 + 0.25 y_3 + 0.125 y_2 + 0.125 y_1$

Two parameters Exponential smoothing

(α, β) - (Holts method)

One Parameter (α)

forcasted values beyond the data, into the future remain the same.

Two parameters

Adds a parameter (β) that accounts for trend in the data.

$$S_t = \alpha y_t + (1-\alpha)(S_{t-1} + T_{t-1})$$

$$T_t = \beta(S_t - S_{t-1}) + (1-\beta)T_{t-1}$$

forcasted value

$$y_{t+1} = S_t + T_T$$

S_t = weighted average of current observation & previous forecast value

T_t = weighted average of change in S_t & previous estimate of the trend parameter.

Holt-winters Seasonality

(α, β, γ)

$$I_p = \gamma \frac{y_t}{S_t} + (1-\gamma) I_{t-p}$$

The value I_t represents seasonal index at a point p in the season

Accuracy of forecast

$$MSE = \frac{\sum_{t=1}^{n} (y_t - \hat{y}_t)^2}{n}$$

mean absolute Deviation

$$MAD = \frac{\sum_{t=1}^{n} |y_t - \hat{y}_t|}{n}$$

We choose weight w to minimize MSE or MAD

Akaike information criteria (Estimator of prediction error)

$$AIC = 2k - 2\ln(\hat{L})$$

$\hat{L}$ - Maximum likelyHood for the model

k - No of estimated parameters in the model.

→ It tells us about the relative quality of statistical models for given data set. (Estimates the information lost by models) representing the

* **Bayesian Information criteria** / (Schwarz IC)

$$BIC = k\log(n) - 2\log(\hat{L})$$

n - no of data points in observed data

Models with low AIC & BIC are selected.

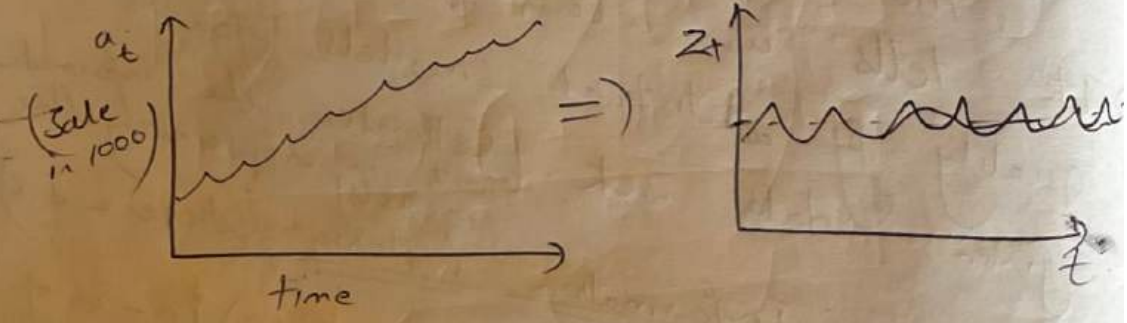
x - observed data $\hat{L} = P(x/\hat{\theta}, m)$

$\hat{\theta}$ is parameter values that maximize likelyHood

while fitting models, it is possible to increase the likelyHood by adding parameters, d but by doing so may result in over fitting. Both BIC & AIC attempt to resolve this problem by ~~adding~~ introducing a penalty term (No of parameters) in the model Penalty term is large in BIC than AIC.

ARIMA model

Auto Regressive Integrate Moving Average



d=1 $\boxed{Z_t = a_{t+1} - a_t}$ (Differences from one time stamp to its previous)

ARIMA – 3 parameters (p, d, q)

$d=2$ $Z_t = Z_{t+1} - Z_t$

Simplest form $(1,1,1)$ p,d,q.

$Z_t = \phi_1 Z_{t-1} + \theta \varepsilon_{t-1} + \varepsilon_t$

get back at

$a_k \Rightarrow Z_{k-1} + a_{k-1} \Rightarrow Z_{k-1} + Z_{k-2} + a_{k-2} \ldots$

$$a_k = \boxed{\sum_{i=1}^{k-d} Z_{k-i} + a_d}$$ → last a value we know

ARMA (1,1) = AR + MA

$$\hat{L}_t = \beta_0 + \beta_1 L_{t+p-1} + \phi_1 \varepsilon_{t-1} + \varepsilon_t$$

β_0 ↓ constant

L_{t+p-1} ↓ last month

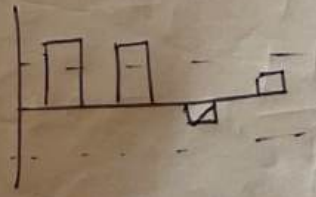
$\phi_1 \varepsilon_{t-1}$ (error from last month prediction)

ε_t ↓ error from this month

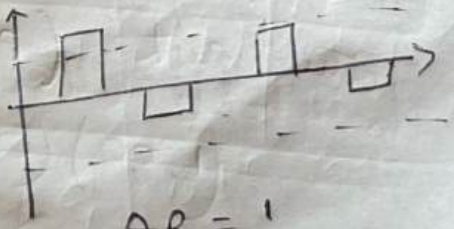
Prediction

$$\boxed{\hat{L}_t = \beta_0 + \beta_1 L_{t-1} + \phi \varepsilon_{t-1}}$$

MA ⇒ ACF , AR ⇒ PACF



MA ⇒ 2



AR = 1

SARIMA model

ARIMA (p, d, q)

Seasonality - repeating pattern over the year that happens over & over aga

7 parameters - $(p, d, q)\,(P, D, Q)_m$

$m = 4$ AR

$$(1-\phi_1 B)(1-\Phi_1 B^4)(1-B)^d(1-B^4)^D y_t = (1+\theta_1 B)(1+\Theta_1 B^4)\varepsilon_t$$

(p, P) (d, D)

q MA Q

Back shift time series by 4 series in the past. (P Q D)

$(1,90)(0,1,1)_4$

$z_t = y_t - y_{t-4}$ =

$$z_t = \phi_1 z_{t-1} + \Theta_1 \varepsilon_{t-4} + \varepsilon_t$$

Association rules

$X \Rightarrow Y$

- Support: $\frac{freq(X,Y)}{N}$
- Confidence: $\frac{freq(X,Y)}{freq(X)}$
- Lift: $\frac{Support}{Supp(X) \times Supp(Y)}$

First pass - Support of each individual items
Second pass
Third pass

Apriori Algorithm
- Candidate item set is generated using only largest itemset of previous pass. without considering items in previous pass
- Large itemset of previous pass is joined with itself to generate itemsets whose size is higher than 1
- Each item set that has subset with support 1 is deleted & remaining are taken.

Example

1 2	1 2	1 2	2
2 3 4	2 4	1 3	1 3
2 3 5	3 4	1 5	1 5
1 2 3 5	× 4 1	2 3	3
	5 2	2 5	2
		3 5	2

$\Rightarrow$ $\Rightarrow$ $\Rightarrow$

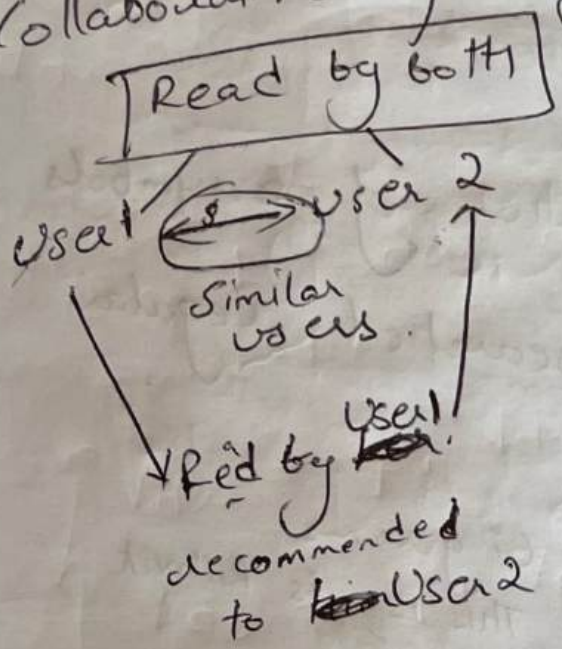
123 - 1
125 - 1
235 - 1

$\Rightarrow$

23 ⇒ 5

Recommendation Systems

→ Collaborative filtering



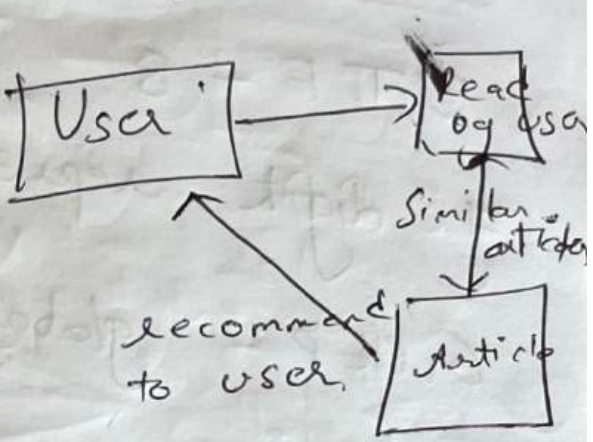
Similarly users will like similar things

Amazon recommendation

→ People who bought ~~this~~ X also bought ~~this~~ Y

→ Cosine similarity b/w users.

→ Content based filtering



Similar content would be liked by user

Youtube recommendation

$$v(x,i) = \cos(x,i) = \frac{x_i}{|x|\,|i|}$$

large number → small angle → large similarity.

Similarity functions

Cosine, euclidian, manhatten, persons

Feature extraction is key

1. 3x
2.

general equation of Perceptron

$$w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b \geq 0$$

{ 1 else 0

→ correct answer – don't change weights

→ wrong answer – adjust weights till it classifies it correctly.

error = t − y

$$\varepsilon = 1 \quad \begin{cases} y = w_1 x_1 + w_2 x_2 + \dots + x_n w_n < \theta \\ t = 1 \end{cases}$$

$$\varepsilon = 1 \quad y = w_1 x_1 + w_2 x_2 + \dots + w_n x_n \geq 0$$

$$y = 1$$

Learning rate (η)

Rate at which training rule converges to correct answer

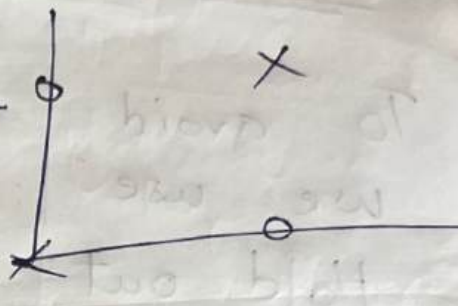
$\eta < = 1$ (Typically)

Too small – slow convergence

Too large – η can cause oscillations in the process.

$$\Delta w = \eta \times \varepsilon \times x \quad (\varepsilon = t - y)$$

XOR will fail with linear model (perceptron)



Multi layer perceptron

output layer

Hidden layer.

Input layer

(Transfer function — Activation function)

same.

output layer - $\varepsilon_o = y \times (1-y)$

Hidden layer - $\varepsilon_i = y_i \times (1-y_i) \times (w_i \times \varepsilon_o)$

weight adjustment - $\Delta w = \eta \times \varepsilon \times x$

Steps

Summation

$s = \sum w \cdot x.$

To solve local minima they use relu.

Transfer/Activation

$f(s) = \frac{1}{1+e^{-s}}$

Sigmoid function

Returns (0-1) → Predicts probability of o/p

(Softmax)

Back propogation

Propogates the error backward by apportioning them to each unit according to the amount of this error (the unit is responsible for).

3 Neural Networks (3 Blue 1 Brown Youtube)

image - 3

Recognize handwritten digits

Plain vanilla concept ⇒ Multilayer perception

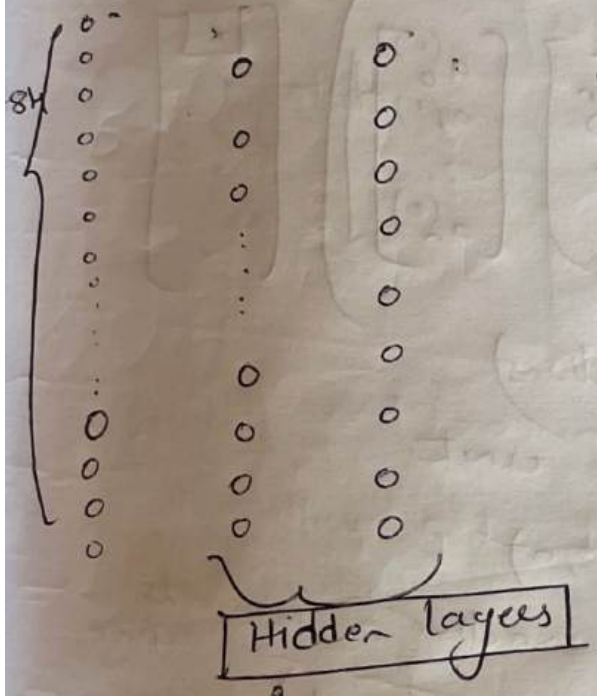
(02) → neuron

28 * 28 pixels = 784

each pixel holds grey scale value
0 - Black
1 - white.

Activation

784



Activations in 1 layer determines activations in another layer.

$\sigma(w_1 a_1 + w_2 a_2 \ldots + w_n a_n)$

Compute weighted sum & to make it a value between 0-1 we use

Sigmoid $\sigma(x) \Rightarrow \frac{1}{1+e^{-x}}$

other activation functions are relu, tan

Lets say we need weighted sum to be active only if weighted sum > 1

bias to be inactive

$\sigma(w_1 a_1 + w_2 a_2 \ldots + w_n a_n - 10)$

Layer 1 → layer 2 784×16 weights, 16 biases

Learning ⇒ finding right weights & biases to all the ~~weight~~ neurons

Math

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} + \begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & & & \\ w_{n,0} & w_{n,1} & \cdots & w_{n,n} \end{bmatrix} \begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} = \begin{bmatrix} \times \\ \times \\ \vdots \\ \times \end{bmatrix}$$

(matrix multiplications)

ReLu - Rectifyied linear unit

$\text{ReLu}(a) = \max(0, a)$

if a is < 0 then inactive or else its active

Learning Part (Gradient Decent)

Train - Test

initiallize w, b randomly

Cost function

Add $(x_0 - y_0)^2 + (x_1 - y_1)^2 \ldots (x_n - y_n)^2$

$x_0 \Rightarrow$ last layer neuron activation value

$y_0 \Rightarrow$ Expected value

Example $(0.43 - 0)^2 + (0.03 - 1)^2 + (0.5 - 0)$

→ Compute average cost over many examples

goal is to find min of cost f(x).

$C(\vec{w})$ = (Ball rolling down the Hill)

gradient → the direction of steepest increase $\nabla C(x, y)$

⇒ So we take "−" of gradient gives direction to step.

$$-\nabla C(\vec{w}) = \begin{bmatrix} 0.18 \\ 0.45 \\ -0.51 \\ \vdots \\ 0.44 \\ 0.82 \end{bmatrix}$$

$w_0(\uparrow)$

$w_2(\downarrow)$

How to nudge w & bias.

↑ Gradient decent

Properly labelled data will have faster gradient descent, & they all will have many local minima roughly of equal quality.

Since we are using structured data (image + labelled) → this is by Human.

→ Supervised learning.

Back propogation — Concept

$$-\nabla C(\dots) = \begin{bmatrix} \vdots \\ 3.2 \\ \vdots \\ 0.1 \\ \vdots \end{bmatrix}$$

(... : weights)

So weights in propotion to ai will

~~have a~~ Hebbian theory

[Neurons that fire together wire together]

Average over all training data & change them

$w_0, w_1 \dots w_{13,001}$

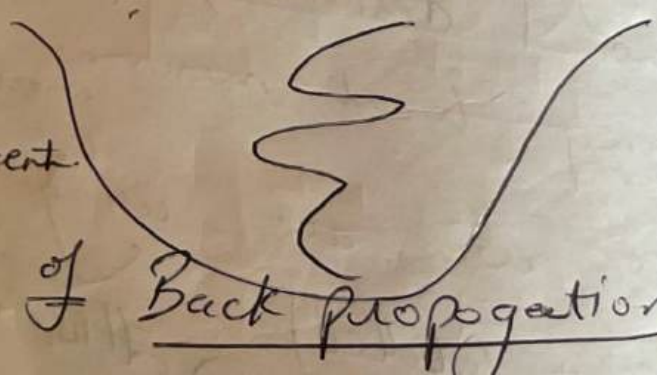
10,000 images

→ 100 × 100 — 100 mini batches of 100 images each

→ Compute gradient decent step with each mini batch.

Stocastic gradient decent

Calculation / Calculus of Back propogation



CNN Images → matrix of pixel values

Convolutional Neural Network

Covid +ve

Precision does ↑ include TN

[Image analysis]

Hidden layers called convolutional layers

Detect patterns in images

Hand written images

filter – small matrix with n x m rows (n, m we initialize) values random

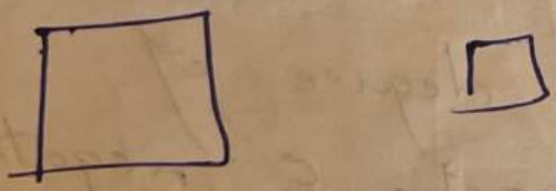


image patch * filter

add pixels

Sum of pixels in filter → move

max pooling, avg pooling

↓

Better as it discards the noisy activations altogether

Convolutional layer + Pooling

↓ th layer in CNN

flatten o/p given to regular NN for

Pooling layer → Extract dominant features
=> (Stack of images become smaller)

- Pick window size
- Pick stride
- walk window across filtered images
- from each window pick the max value (To decrease the computational power required to process data)

* Relu → No negative values

Layers can be repeated several times
(Convolution Relu Convolution Relu pooling
Convolution Relu)

Back propogation

Error = (right answer) - (wrong Answer)

Minimize error

Adjust weights up & down & see how error changes

Padding → Amount of pixels added to an image when it is being processed by kernel of CNN

Extends the area of which CNN processes an image.

Kernel is a NN filter which moves across the image, scanning each pixel & converting data into a smaller/larger format

In order to assist the kernel to process the image padding is done & it leads to more accurate analysis of images.

Batch Normalization

General technique to normalize i/ps to the layer before or after activation function

(Makes NN more stable during training)

faster training.

log odds (Easier for calculations for new data)

$$\log\left(P(A)/P(\neg A)\right)$$

$$P = \frac{e^{\log(odds)}}{1+e^{\log(odds)}}$$

Brief

$\log(5/100 / 95/100) \Rightarrow -2.944$

Breaking

$\log('1/2 / 1/4) \Rightarrow 0.69$

-2.25

$P \Rightarrow e^{-2.25} \Rightarrow 0.1053$

$\Rightarrow \frac{0.1053}{1+0.1053}$

$\Rightarrow 0.095$

9.5%

CNN

→ feed forward NN using filters & pooling

→ Size of i/p & o/p is fixed

→ Spatial data (images)

Image recognition / classification, medical analysis etc

RNN

→ recurring network that feeds the result back into the network

→ Size of input & output may vary

→ Good for temporal data / sequential

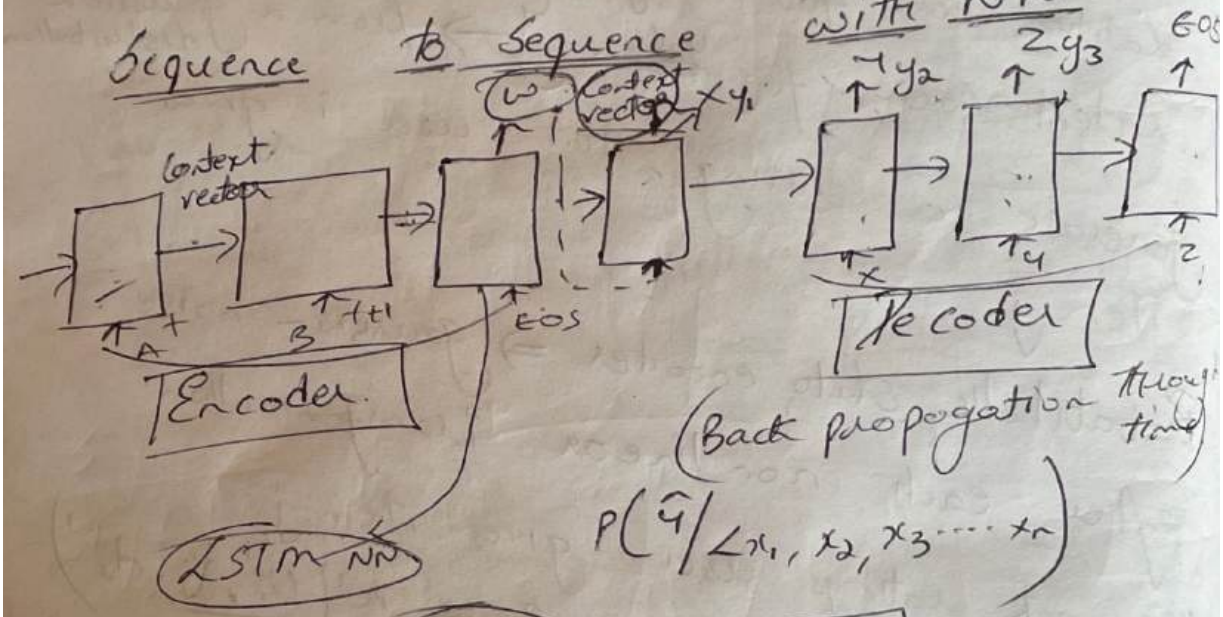
Text translation, sentiment analysis (NLP tasks) speech analysis

Max pooling is a pooling operation that calculates the max value of a patch in feature map.

Usually used after ~~CNN~~ Convolutional layer

[Translating the image by small amount does not ~~change~~ affect the values of most pooled o/p]

Sequence to Sequence with NN



(Back propagation through time)

$P(\hat{y} / \langle x_1, x_2, x_3 \cdots x_n)$

~~Text~~

$L(y, \hat{y})$ Loss function

image → Advanced CNN

we get image vector - flattend layer → Decoder

Disadvantages

Long sentence, the accuracy decreases

⇓

Self attention (Bi directional LSTM) architecture.

LSTM & GRU

long short term memory / Gated Recurrent units

Vanishing gradient problem (Recurrent NN)

gradient value becomes small with time & it doesn't contribute much to learning.

internal gates that regulate flow of info

(which data to keep / throw away)

Tanh ⇒ -1 to 1

gates - forget gate, i/p gate, o/p gate

Sigmoid activation (0 - 1)

0 → forget

1 → keep

GRU

Reset gate / update gate

(forget gate)

(How much past info to forget)

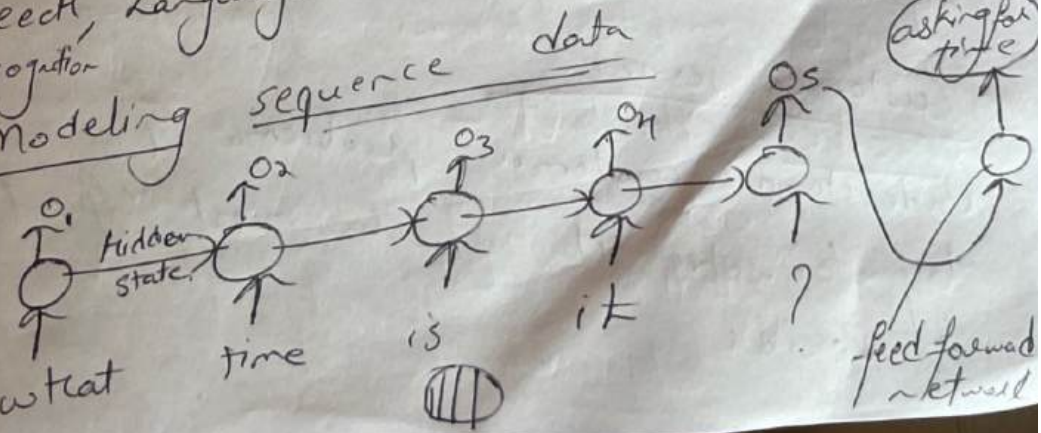
Less operations so little faster to train than LSTMs

Recurrent N.N

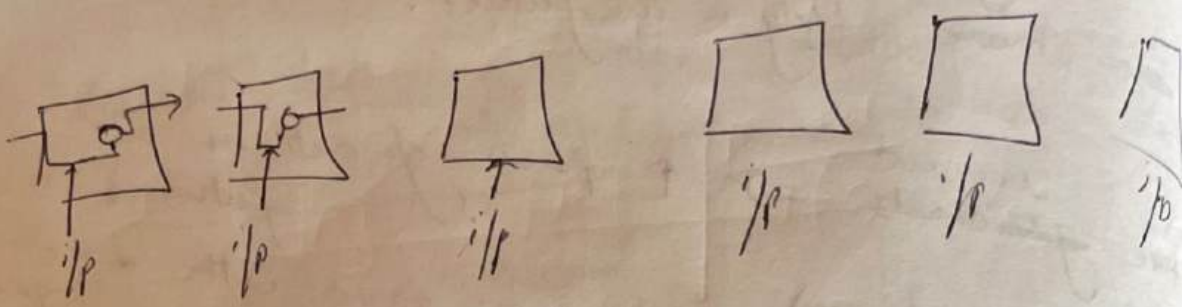
Problem

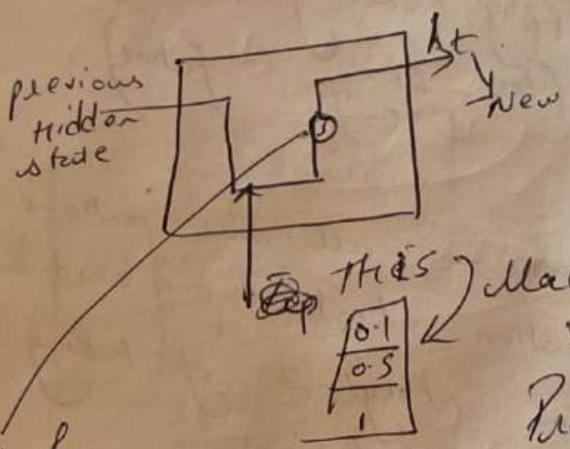
Speech recognition, Language translation, stock prediction

Modeling sequence data



Recurrent Neural Network

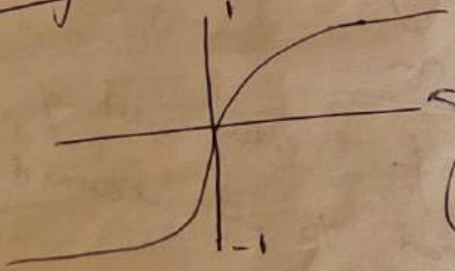




Machine readable vectors

Process sequence of vectors 1 by 1

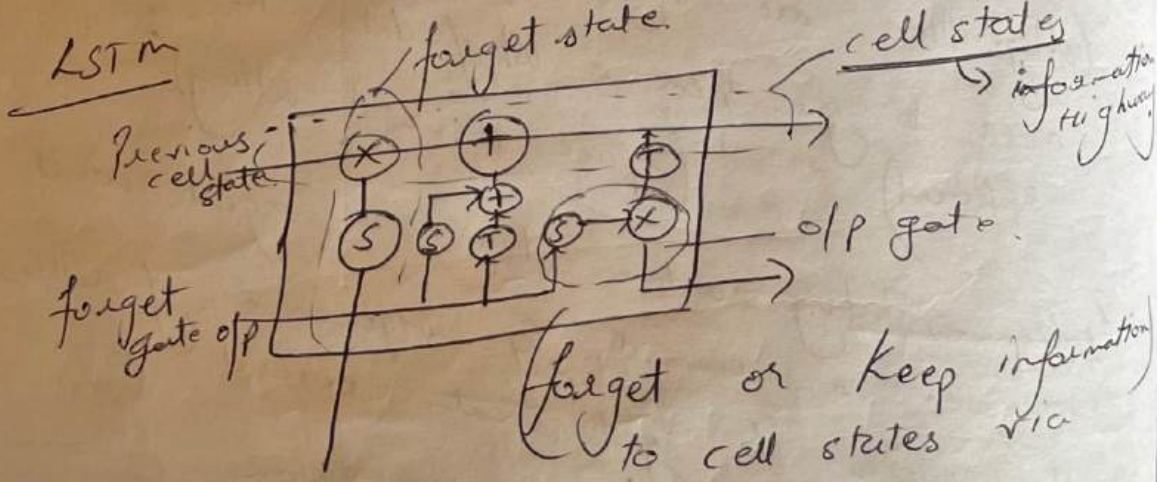
Tanh fn



(Some values exploded to astronomical)

RNN uses lot less computational resources

LSTM



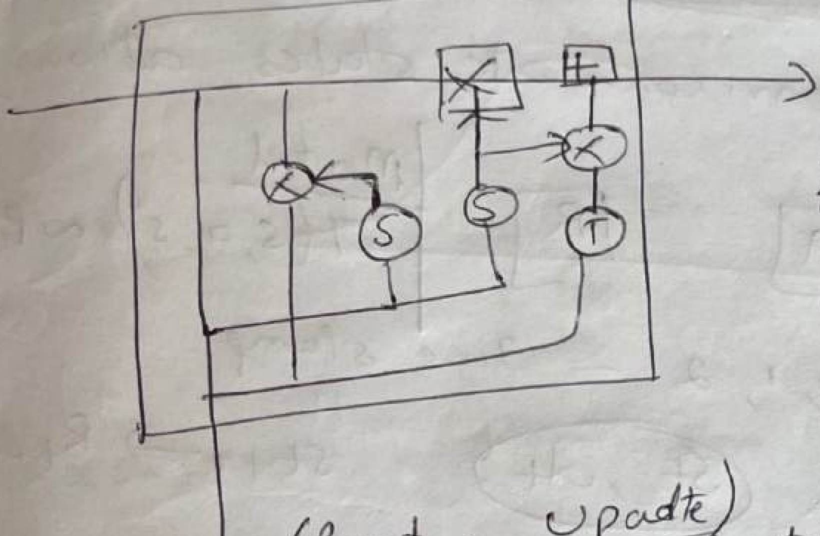
(forget or keep information to cell states via gates

(gates contain sigmoid activation)

(0-1)

(forget) (keep)

GRU



Removed cell state. & used hidden state to Transfer info instead.

How much (Reset
~~1/4 to forget gate~~
past i/p to ~~i/p gate~~

Update)
forget + i/p gate

Mitigate short term memory using gates
→ Gates are NN that regulate flow of info to be passed from 1 step to next.

Quartile forecasting

(Driver) → (Store) → (Customer Delivery)
↓
(ETA 15 - 25 minutes)

Quartile Regression

we want to penalize loss if (τ) percentile is ↓ & predict ↑ percentile is ↑, predict is ↓

10% — 90% chance
(25 mins — 35 mins)

(Package lightGBM)

LGBMRegressor(objective = 'quantile' alpha = tau)

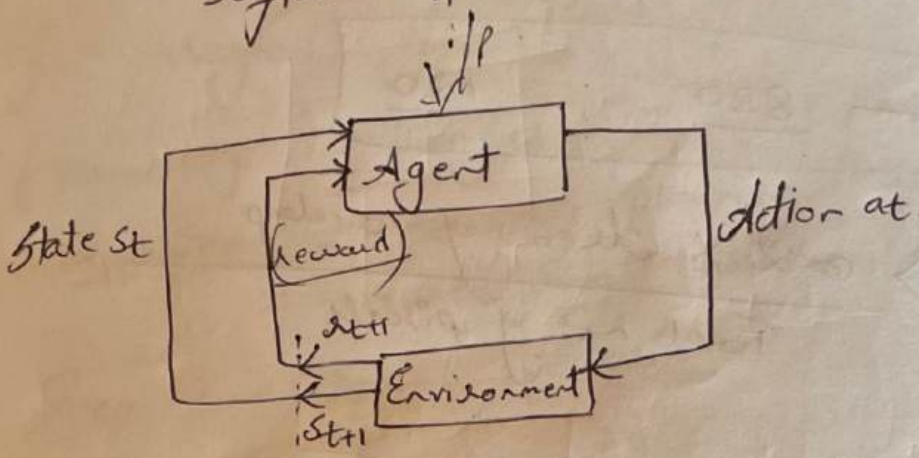
Deep Reinforcement Learning

(Robots)

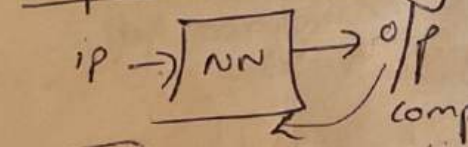
Current technology has good hardware but it lacks the Intelligence to do a task

↓

Software Problem



Supervised Learning



compute gradients using back prop } To train

game

Dataset ← Difficult to create

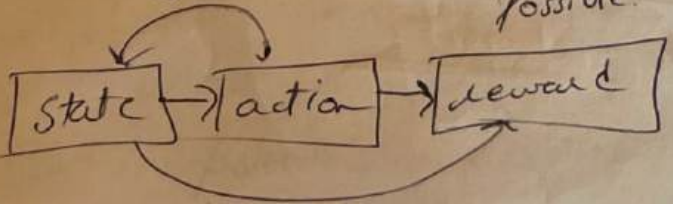
Agent can be better than human to play that game

In Reinforcement learning → we don't know the target label. (Go up/down)

(Policy Network)

NN o/p ⇒ Up 0.7 / Down 0.3

Reward → +1
Penalty → -1
} optimize its policy to recieve as much reward as possible.



that would make
compute gradients for the actions that our agent choose
~~are~~ more likely in the future.

+ve reward ⇒ ↑ prob of actions in future.

-ve reward ⇒ Apply same grad with - ve sign
(less likely in future)

Downsides

Policy gradients assume all the actions that we took when lost are bad actions
& it reduces the likelyHood of taking those actions in the future.

Credit assignment problem

Instead of given reward for entire set of actions, which actions lead to the reward.

(Algorithms are very sample inefficient & we need to give them a lot of data to be effiuent) (Sparse reward setting)?

- Problem → Taking random steps, the agent wont ever see a reward & sequence of actions it requires to get that reward +3
too complicated.
(Robot stacking)

Reward shapping

Manually design reward fn that needs to guide agent for desired behaviour

Computer vision

we have target label for every frame & this lets us to calculate gradient descent using backprop

Markov's Decision Process

Agent, environments, states, actions, re

S | A | R | Model $T(s, a, s') \sim P(s'|s$

$t = 0, 1, 2$ — Time stamp.

$(S_t, A_t) \rightarrow S_{t+1} \rightarrow R_{t+1}$

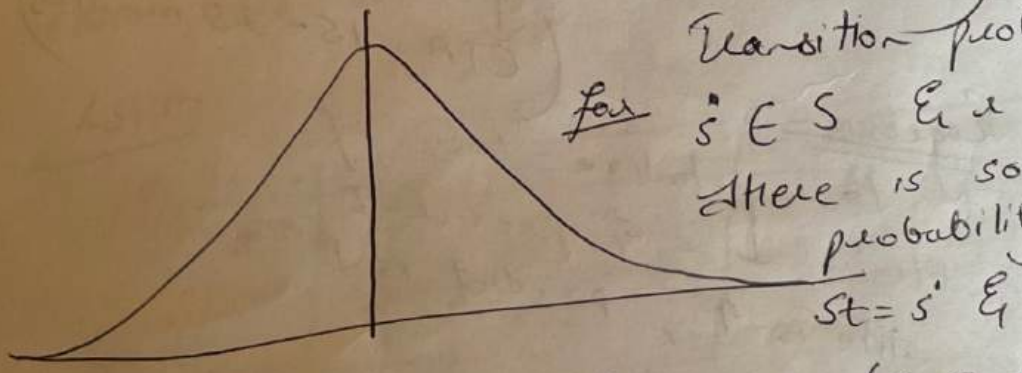
$f(S_t, A_t) = R_{t+1}$

At time t

- Env. is in state S_t.
- Agent selects action A_t.

At time t+1

- R_{t+1}



Transition probabilities

For $s' \in S$ & $r \in R$

there is some probability that

$S_t = s'$ & $R_t = r$

This probability is determined by a particular values of preceding states $s \in S$ & ~~action~~

$a \in A(s)$ → set of actions that can be made from state s

Only present matter

Policy $\pi(s) \rightarrow a$ (action) | what action to take at any state

(state)

$\pi^* \rightarrow$ Maximizes the reward over a long time.

$\langle sa \rangle \langle sa \rangle \rightarrow$ learn for policy

$\langle s\,a\,s\,r \rangle \langle s, a, r \rangle$

If policy is optimal, it does tell us what to do in what ever situation we are in.

$$\pi^* = \text{argmax}\, E\left[\sum \gamma^t R(s_t) \mid \pi\right]$$

Utility of a policy $U^{\pi}(s) = E\left[\sum \gamma^t R(s_t) \mid \pi,\ s_0 = s\right]$

Reward (Utility for the state is going to be long term reward.) gives immediate feedback

Delayed rewards

optimal policies

$$\pi^*(s) = \text{argmax} \left\{ \sum T(s,a,s')\, U(s') \right\}$$

$U(s) = U^{\pi^*}(s)$

$$V(s) = R(s) + \gamma \max \sum T(s,a,s') V(s')$$

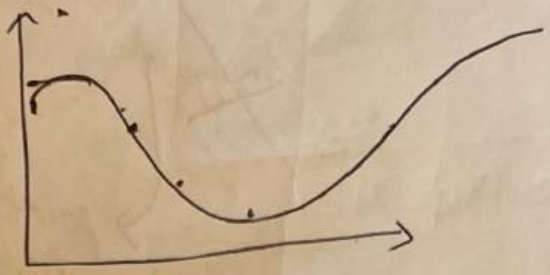
Utility (Reward) (Discount of all rewards that i get at that point)

Bellmans Equation

Optimization

→ Minimize error / loss / cost function by Adjusting hyperparameters

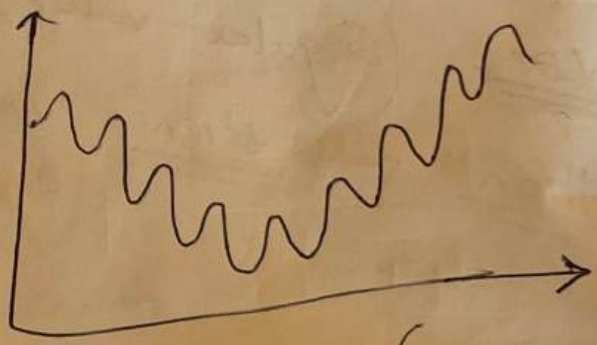
Gradient Decent



Extaustive search
gradient decent
Genetic algorithms

if slope is steeper take a big step
& if the slope is shallower take small steps

Local minimas, we are interested in global minima



(Race track)

(walking) ← Genetic Algorithms, simulated annealing etc → Gradient decent

(truck)

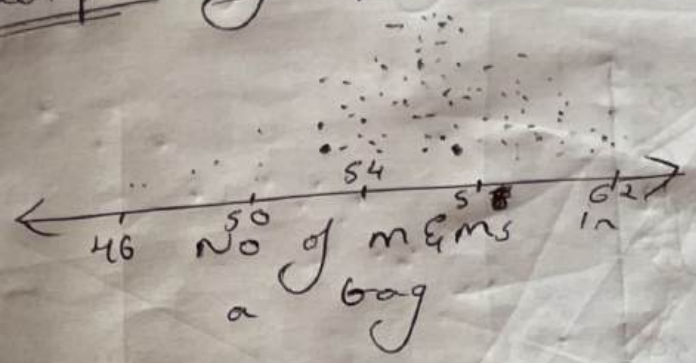
Extaustive search
- Few assumptions
- Robust
- Expensive to compute

- More assumptions
- Sensitive
- Efficient to compute

(Randomness to the steps & jumps)

Examples of optimizations – Bag of mEms

(Single parameter)



d = actual − guess

Cost ⇒ d^2	1	4	16	64
$\lvert d \rvert$	1	2	4	8

Loss function

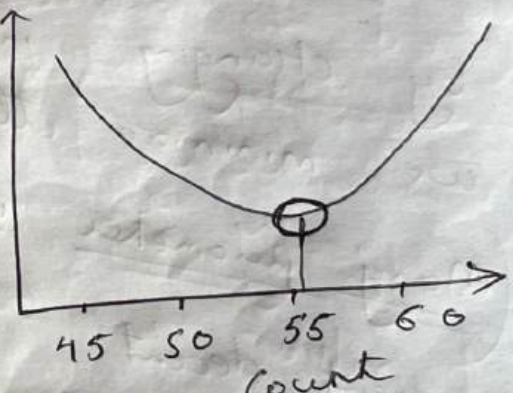
$$\text{loss} = d_1^2 + d_2^2 + \dots\dots d_n^2$$

(if we take |d| then best ans would be median)

$$d = \Sigma_i (n_i - n_{est})^2$$

$$\frac{d}{dn}\left(\Sigma_i (n_i - n_{est})^2\right) = 0$$

loss



Slope = 0

$$\Rightarrow \Sigma_i \frac{d}{dn}\left[(n_i - n_{est})^2\right]$$

$$0 = \Sigma_i \, 2(n_i - n_{est})$$

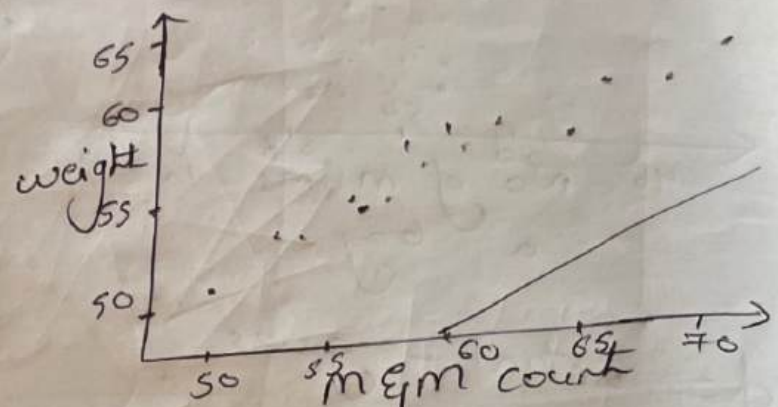
$$\Sigma_i n_i - \Sigma_i n_{est} = 0$$

$$n_{est} = \frac{\Sigma_i n_i}{m} = \bar{n}$$

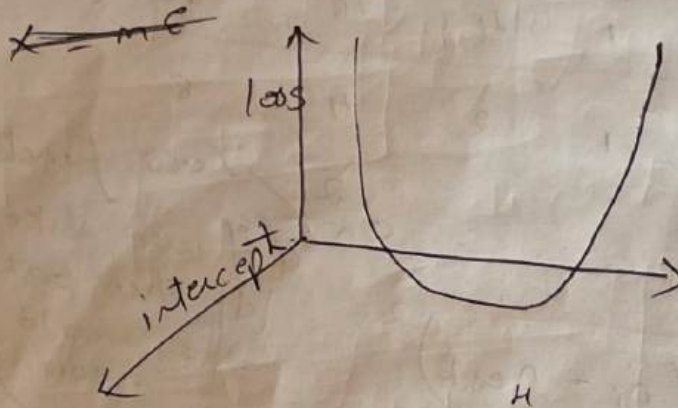
(Average of count)

mEm count (vs) bag weight

Cost function is square deviation (d^2)



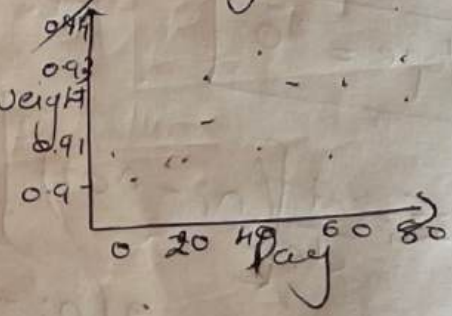
~~MSE~~ (Linear model loss)



by changing intercept & slope we minimize the loss function.

(Many parameter model)

Gradient decent gives different answers or where we start. Answer is sensitive to initial guess



• (Non-Convex loss function)

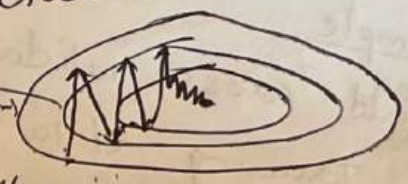
Scatter start approach

→ ~~Several~~ Several random starting points & running gradient decent on each of them (Better chance of global optima)

Stochastic Gradient Descent with Momentum

It accelerates gradient vectors in right directions, thus leading to fast convergence.

We define a momentum, which is moving averages of our gradients we then use it to update the weights of the network



These up & down ~~GD~~ delays GD (α) overshoots the minima.

↕ slow learning rate

⟺ faster learning rate.

On iteration t

⇒ Compute dw & db to current mini batch.

(Ball rolling down analogy)

friction

$$v_{dw} = \beta v_{dw} + (1-\beta)\, dw$$

$$v_{db} = \beta v_{db} + (1-\beta)\, db$$

velocity

acceleration

Change weights & Bias to ~~instead of~~

$$w = w - \alpha v_{dw}, \quad b = b - \alpha v_{db}$$

↓ Smooths out the ~~steps~~ steps of gradient decent

α = Learning rate.

$\beta = 0.9$

Most common value

Initially $v_{dw} = 0$, $v_{db} = 0$

Adam Optimization

$V_{dw} = 0$, $S_{dw} = 0$, $V_{db} = 0$, $S_{db} = 0$

On iteration t

Compute dw, db using current mini batch.

momentum β_1:
$$V_{dw} = \beta_1 V_{dw} + (1-\beta_1) dw$$
$$V_{db} = \beta_1 V_{db} + (1-\beta_1) db$$

RMSProp β_2:
$$S_{dw} = \beta_2 S_{dw} + (1-\beta_2) dw^2$$
$$S_{db} = \beta_2 S_{db} + (1-\beta_2) db^2$$

$$V_{dw}^{corrected} = V_{dw} / (1-\beta_1^t) \qquad V_{db}^{corrected} = \frac{V_{db}}{(1-\beta_1^t)}$$

$$S_{dw}^{corrected} = S_{dw} / (1-\beta_2^t) \qquad S_{db}^{corrected} = \frac{S_{db}}{(1-\beta_2^t)}$$

$$w = w - \alpha \frac{V_{dw}^{corrected}}{\sqrt{S_{dw}^{corrected}} + \varepsilon} \qquad b = b - \alpha \frac{V_{db}^{corrected}}{\sqrt{S_{db}^{corrected}} + \varepsilon}$$

α = Learning rate (to be tuned)

$\beta_1 = 0.9$ (dw) | $\beta_2 = 0.999$ (dw^2)

$\varepsilon = 10^{-8}$

2) Naive Bayes

likelyHood ↑ , Prior probability →

$$P(B/A) = \frac{P(A/B) \cdot P(B)}{P(A)}$$

Posterior probability ← Predictor

Decision Trees, Random forest & Neural Networks

Only difference here in classification is that the output is discrete rather than continous.

Unsupervised learning

Patterns of input data without reference to labeled outcomes.

→ Clustering → Dimensionality Reduction.

Clustering ~~KNN~~ (k means)

grouping of data points

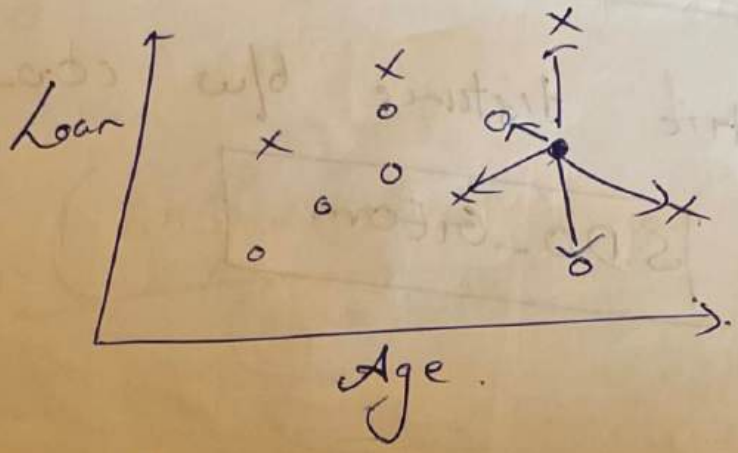
frequently us in customers segmentation, fraud detection, document classification etc

(k means, Hierarchial, mean shift, Density based)

K nearest neighbours (KNN - Classification)

Stores all available cases & classifies new cases based on a similarity measure.

Eg



Similarity functions

Manhattan, $\sum (x_i - y_i)$

Euclidian, $\sqrt{\sum (x_i - y_i)^2}$

Cosine,

Minkowski $\left(\sum_{i=1}^{k} (|x_i - y_i|)^q\right)^{1/q}$

$q = 1$ (manhattan)

$q = 2$ Euclidian

Should k be always odd?

25	40,000	New point
35	60,000	43 30,000
52	100,000	

(it is wrong as more weightage is given to loan & age is almost discarded)

So

Standardized Euclidian

$$x = \frac{x - Min}{max - Min}$$

Number of neighbours

Select most freq neighbours
take the ratio & say it

(Eg 3 - Red 2 - blue
60% it is red, 40% it is blue)

Categorical Attributes

Male Male 0
Male Female 1

closer to 0 they are similar
& away from zero they are dispair

→ As number of dimensions increase distances become more closer

Kmeans Clustering

Unsupervised learning

Goal is to find groups in data with number of groups represented by k

Algorithm

- Randomly assign k centroids
- (each data point is assigned to its nearest cluster)
- Update centroids by taking mean of data pts in the cluster
- Stop when no data point is reassigned to its nearest cluster (will converge)

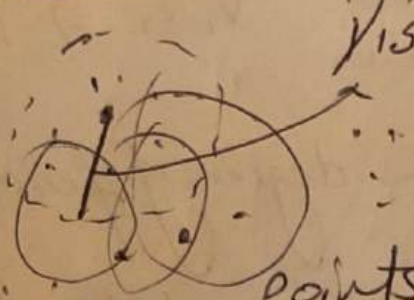
Output - k centroids for k clusters.
it can be used to label new data
Labels for training data.

Example grouping inventory by sales

Choosing K - for range of k plot mean distance to the centroid & pick the elbow point where rate of decrease sharply shifts.

DBSCAN

Epsilon — Distance that we set



Minimum points — Minimum number of points that must exist for a cluster to be formed. (including the start/core)

Each member of cluster will broadcast their own perimeter.

- They look to find new data points to join the cluster

→ New points again search for new ones after no more data points could be found, the cluster is finalized.

⇒ We start again by randomly selecting new starting points; ~~again~~

Linear discriminant Analysis

Searches linear combinations of variables that best seperates 2 classes.

$$Z = B_1 x_1 + B_2 x_2 + \cdots B_d x_d$$

$$\beta = C^{-1}(\mu_1 - \mu_2)$$

Mean vectors

Pooled covariance matrix

$$C = \frac{1}{n_1+n_2}(n_1 C_1 + n_2 C_2)$$

Assement

$$\Delta^2 = B^T(\mu_1 - \mu_2)$$ (Mahalanobis distance)

$\Delta > 3$ means 2 groups ~~are~~ averages differ by 3 standard deviations & overlap b/w 2 groups is small.

Model Evaluation for classification model

To avoid overfitted & overoptimistic model we use

Hold out & cross validation

↓

To evaluate the model.

Hold out

Training set
validation set
Test set

Speed, simplicity, flexibility

Disadvantages

↓

Variability in test & train data set

Cross validation

Split data into training set

Eg k fold

k=10 (Data split into 10 subsets)

trained on 9

& tested on 1

Hyperparameters – Depth,

Confusion matrix

Model \ Actual	Yes	No
Yes	TP	FP
No	FN	TN

TP – True positive
FP – False positive
FN – false negative
TN – True negative

Sensitivity = TP/(TP+FN)

Specificity = TN/(TN+FP)

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

(precision) Positive predicted value - TP/TP+FP

Negative predictive value - TN/TN+FN

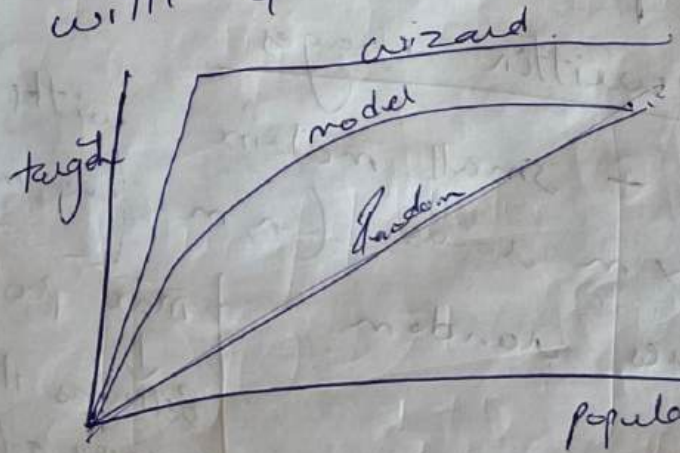
Sensitivity - Accuracy of model for actual positive cases.

Specificity - Accuracy of model for actual negative cases.

39961

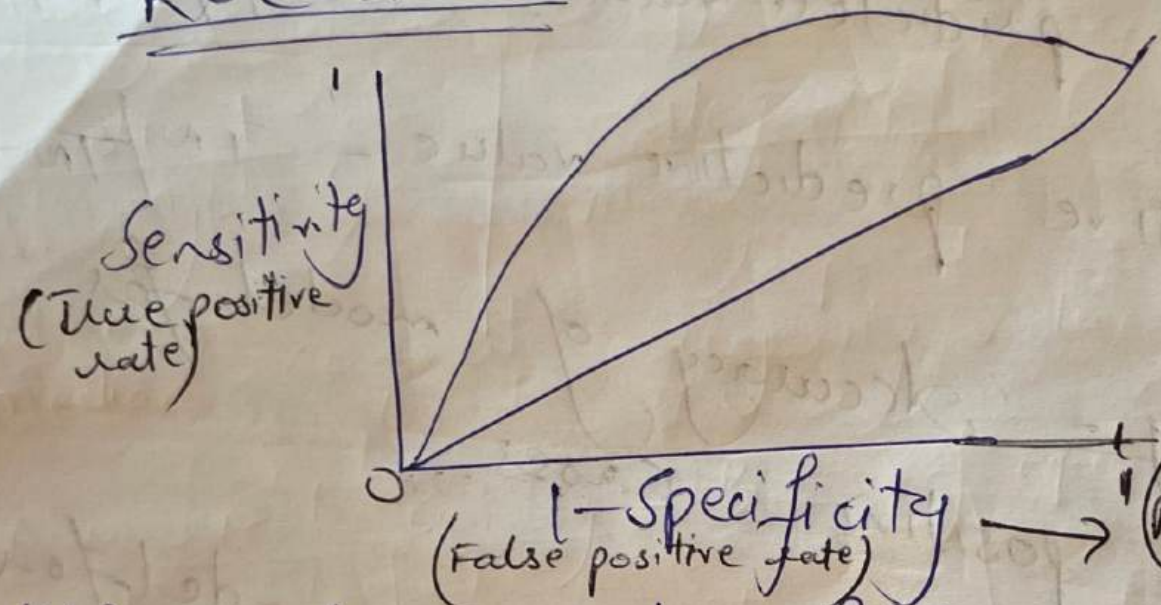
Gain chart

Measure of effectiveness of classification model calculated as the ratio b/w the results obtained with & without the model.



K-S is a measure of degree of seperation between positive & negative distributions Best - 100

ROC chart — Reciver operation characte



Summarizes confusion matrix that each threshold produces.

Replace Precision

AUC - Area under ROC curve

~~Provides~~ o/w [0.5 - 1] for ch models

Easy to compare 2 ROC's for model evaluation

CNN — Convolutional Neural Network

Images → matrix of pixel values

Covid -ve are ↑ Precision doesn't include TN

Basics of Text Analysis

Character encoding

UTF-8

7 digit representation of 6 symbols

GREP - globally searched regular expressions

N grams - Unigrams, 6i grams
This, is, Rithvik
This is, is Rithvik

Creating Text database

Corpus - stores actual data

Tidy text, Tm - (packages)

map reduce Algorithm

↓ Mapping text to lookup table which has words

Reduce it by aggregation

Preprocessing → Remove stop words

Eg to, the, and, of etc

They dont add value

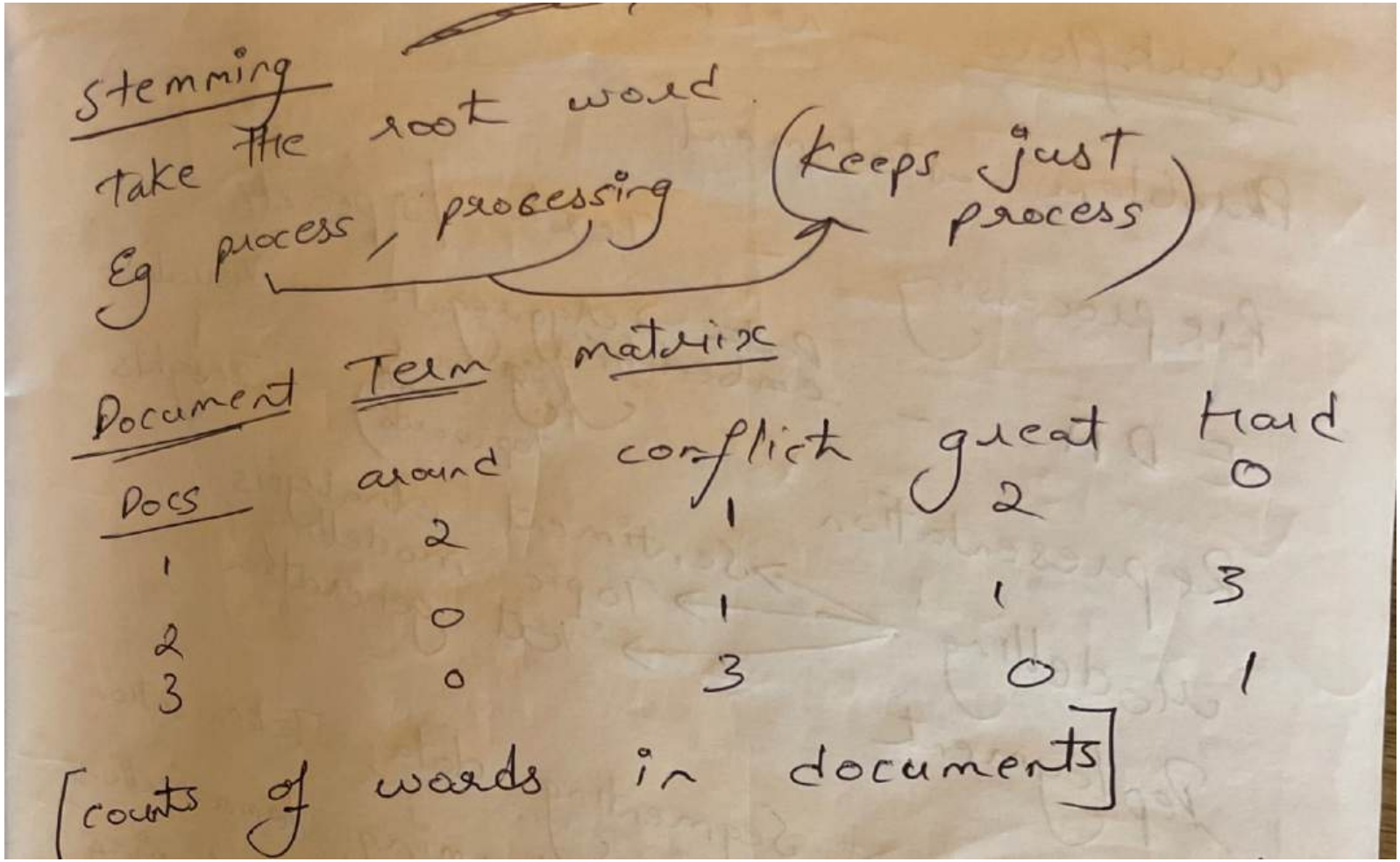
→ Remove punctuations

→ Based on usecase, remove numbers

→ Word case - lower or upper case.

→ Remove whitespace

} Tidytext



Natural Language Processing

How computers carryout instructions

How to deal with text data

Sentiment Analysis

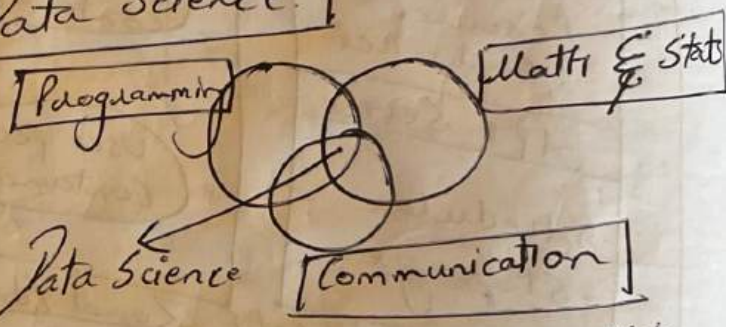
[Text is positive or negative]

Topic Modelling - Label the document based on text

Text generation - Replies to messages

Analytics → Data Science

[Using data to make desicision]



Programming - pandas, sklearn, de, nltk, Textblob, gensim

Math & Stats - corpus, document term matrix, sets, sentiment, topic modelling, text generation

Communication → Scope, visualize, extract insights, expertise

workflow

Start with question → Get & Clean Data → EDA → Apply Techn → Share insights

Data gathering

[How, How much, which data, time range]

Data cleaning

- Corpus

DTM

Pandas data frame.

corpus

Comedian	Transcript

Clean

→ Remove punctuations

→ lower case

→ remove numbers

re → python

Tokenize

Unigrams, bigrams etc

Bag of words

Document term matrix

Comedian	right	wish	me	back	...
Ali wong	1	0	1	2	
dave chappelle.	0	1	3	0	

sk learn

Count vectorizer (To create DTM)

Question

How is Ali wong different?

Shift + tab → function details

Exploratary data analysis

→ Top words

→ Vocabulary

→ Amount of Profanity

Visualization

word cloud

Comedian	unique words	total words	run times	words/minute

Sentiment Analysis

⇒ Input ⇒ Corpus ⇒ [order matters]

→ Textblob - Python library - averages polarity & subjectivity of

→ Output = Sentiment score
~~Objectivity~~ Subjectivity Score [How opinionated they are]

Topic Modelling

Disney movies ⇒ 

→ Input - DTM & no of topics, no of iterations

→ Library - gensim
(Latent Dirichlet Allocation)
LDA

⇒ Output - find themes across the comedy routine

LDA - Learn top mix in each document, add word mix in each topic

Steps → Choose no of topics in corpus
→ Randomly assign each word in doc. to a topic
→ Go through every word & its topic assignment in each document
Look how often it occurs in topic overall. Based on this assignment info assign word to new topic

⇒ if topics dont make sense, use nouns only
[Linguist manually labelled them]

Analytical Modelling

After analysing patterns → Build a model to predict x.
(To get multiple perspectives)

↓

Hypothetical situations
(How extreme data can model)

Model is only as good as the way data is composed / assembled.
(Avoid missing / incorrect / irrelevant data as it skews the model)

(Include everything that is necessary & not everything that is possible)

make predictions & draw conclusions → others can draw their own conclusions

→ No good in knowing it ⇒ one has to use information to take decisions

Dimentionality Reduction

The process of reducing dimensions of feature set

→ feature elimination

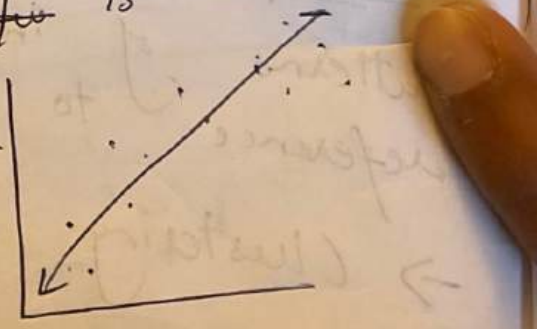
→ feature extraction

Popular method — Principal component Analysis (PCA) → We get principle components & the variance explained by PC.

Weight & bias (Behaviour when you train)

Bias - Inability of model to capture the true relationship ~~of~~ is called bias

Eg (Linear regression) Straight line can't be curved like true relationship



(Behaviour when you Test)

Variance - The difference in fits b/w data sets is variance

Eg - Model fits perfectly with training set (overfitting) but has high sum of squares in test set

A good ML model should have low bias (Model true relationship accurately) & variance (Consistent prediction across different datasets)

Variance bias trade off

We want to ~~fit~~ train the model well but we dont want to overfit the data ~~is a way that~~ other how do we optimize our hyperparameters ~~we~~ if we dont touch the test set?

K fold cross validation (very usefull for hyperparameter optimization)

Example

→ 10 fold cross validation

Split training data into 10 equal sections we train the algorithm / model on 9/10ths of the training set & evaluate its performance on remaining 1/10th of train set

Then we train on different 9/10 of train set & test on 1/10th of train set

Repeat this process 10 times & we get 10 different measures of model performance

After each iteration, we compare the output results, access the accuracy & adjust the hyperparameters.

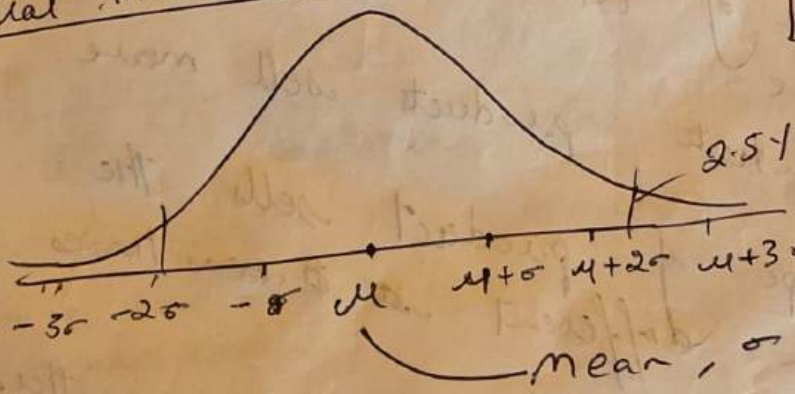
P-value

Probability of obtaining result at least as extreme as the observed result of a statistical hypothesis test, assuming that null hypothesis is correct.

⇒ Probability is coming from distribution (Area under the distribution)

$P < 0.05$ [we reject null Hypothesis]

Central limit theorm



P value - 0.05
142 - 169

151 - 156
P value - 1
Prob - 0.04

P value is the probability that random chance generated the data or something else that is equal or rarer

prob of getting HH - 0.25

P value prob of getting HH - 0.5

(Since its not < 0.005, its not unusual to get 5 heads in a row)

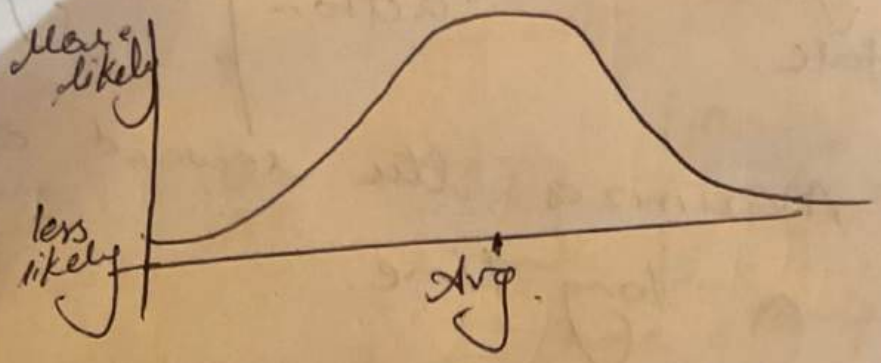
HHHHH

$1/32 = 0.031$

P(5 heads) P(5 Tails)

$\frac{1}{32} + \frac{1}{32} = 0.0625$

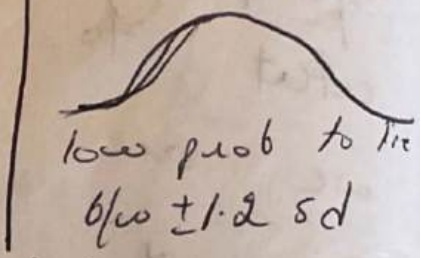
Distributions



95% of measurement falls in +/- 2 standard deviations of around mean

Width tells us the probability in which measurements like 6/w sds

→ High prob to be like 6/w ±1.2 sd



Central Limit Theorem

Means are normally distributed.

of samples

(we don't know what distribution our data comes from. ⇒ Sample means are normally distributed)

(Sample size must be atleast 30)

↓ Rule of thumb

Cauchy dist → exception

↓ & make confidence intervals ✓ t tests

ANOVA

Difference b/w means of samples

Normalization & Standardization for feature scaling

25 years → magnitude
weight = 70 kgs
units

(sklearn min max scalar)

⇒ Normalization helps to scale down features between 0 - 1.

$$\frac{x - x_{min}}{x_{max} - x_{min}}$$

Standardization - based on standard normal dist

(Standard scalar) $\mu = 0$ & $SD = 1$

$$Z = \frac{x - \mu}{\sigma}$$

Euclidian distance / Gradient Descent

⇒ Standardization performed better

images - Normalizations

CNN / ANN — we scale down between 0-1

↓ Helps to learn weights quickly

R^2 Coefficient of determination

$R^2 \uparrow$ Amount of variation in the data are explained by variables

0 - 1 good

But some fields of study have great amounts of unexplainable variance (Human behaviour) $R^2 < 50\%$.

* If you have Low R^2 but independent variables are statistically significant; we can still draw important conclusions.

Elastic net (Combination of ridge & lasso regression)

$$\frac{1}{2n} \sum (y - xw)^2 + \alpha(\text{Ratio}) \sum_{j=1}^{p} |w_j| + \alpha(1 - \text{Ratio}) \sum_{j=1}^{p} w_j^2$$

If no of predictors (p) > no of observations (n) lasso will at most pick n predictors as non zero.

If there are 2 or more highly collinear variables lasso randomly picks one of them which is not good for interpretation

Inter cluster - distance b/w data points with cluster center

Intra cluster - Distance b/w data point in one cluster to data point in another cluster.

F distribution

$$F = \frac{X_1/\nu_1}{X_2/\nu_2}$$

$X_1, X_2 \Rightarrow$ Random chi-square variable

$\nu_1, \nu_2 \Rightarrow$ Degree of freedom

$0 - \infty$, Value depends on degree of freedom

(Assumes – Populations are normally distributed, Populations are independent of each other)

F ratio: $\frac{S_1^2}{S_2^2} = F$ (To find variance b/w 2 samples)

$$S^2 = \frac{\sum (x - \bar{x})^2}{n-1}$$

F statistic / F value

To identify means b/w two populations are significant or not.

Ratio of 2 variances

Used in ANOVA

t-test single variable diff or not

F group of variable stat significant or not

To compare more that 1 level of independent variable with multiple groups we use f distribution.

Rolling a dice & expected payoff

Roll -1

Average $\Rightarrow$ $\boxed{3.5}$ expected payoff.

2 Rolls

I wont roll if I get a 4, 5, 6 as my expected payoff is 3.5

However if i roll 1, 2, 3 ; I roll again

$$\frac{3}{6}[3.5] + \frac{3}{6}[5] \Rightarrow \boxed{4.25}$$

3 Rolls

I wont roll if I get 5 (or) 6 as with 2 rolls my expected pay off is 4.25

$$\frac{4}{6}(4.25) + \frac{2}{6}(5.5) = 4.\overline{66}$$

[Theory of optimal stopping for markov chain]
Assigns a payoff for each state of markov chain
This amount you'd collect if you stop playing with the chain in the state.