WEEK 2

Scale	Description	Examples
Nominal (Bar Chart)	Data is categorized without order or ranking. Categories are just names or labels.	Gender (M/F), Blood Type (A, B, AB)
Ordinal (Histogram)	Categories are ordered/ranked, but differences between ranks are not consistent or meaningful.	Movie Rating (Poor, Fair, Good), Education (HS, MS, B.Tech, M.Tech)
Interval (Histogram, Line Chart)	Ordered categories with equal intervals, but no true zero.	Temperature in °C/°F, IQ scores, Calendar Years
Ratio (Scatter Plot, Histogram)	Ordered data with equal intervals and a true zero point. Zero means no quantity, and values can't be negative.	Weight, Height, Age, Income, Marks, Temp in Kelvin

Feature	Nominal	Ordinal	Interval	Ratio
Countable	✓	✓	✓	✓
Order Defined		✓	✓	✓
Difference Defined (+ -)			✓	✓
Zero Defined (×, ÷)				✓
Mode (Measures of Central Tendency)	✓	✓	✓	✓
Median		✓	✓	✓
Mean			✓	✓
Count / Distribution	✓	✓	✓	✓
Minimum, Maximum		✓	✓	✓
Range		✓	✓	✓
Percentiles		✓	✓	✓
Standard Deviation / Variance			✓	✓

Median = Middle value | Avg of 2 middle values | **Range**= Max-Min | **IQR**=Q3-Q1(75th-25th) | **Mean**= Avg of all vals | **Variance**(σ^2)= $\frac{\sum(x-\mu)^2}{N-1}$ | **SD**(σ)=$\sqrt{variance}$ | **10th percentile**= Item at Index 0.1*N | **90th percentile**=Item at index 0.9*N | Q1-25th, Q2-50th/Median, Q3-75th

Typical Cleaning steps: Type & Name conversion→Filter missing/ Inconsistent data→Unify→Match entry→Rescaling&Dimension Reduction

Discrete-Only values (no decimals) Ex: Jersey numbers

WEEK 3

Bar Chart	Nominal, Ordinal	Compare values across categories (**frequencies**)	Gender distribution, product category counts
Histogram	Numeric (Interval, Ratio)	Show **distribution** of a **single continuous** variable	Detect skewness, modality, normality
Box Plot	Interval, Ratio	Compare **spread**, detect **outliers** using quartiles	Comparing income distributions across groups
Scatter Plot	Interval, Ratio	Show **relationship** between **two continuous** variables	Correlation between age and income, trend lines

Numpy->Multi domain arrays & Math functions
CSV vs Pandas-> Reads everything as strings vs Automatically guesses Datatypes | **scipy**- Computing libraries | **Pandas**-> Series(1D), **DataFrame(2D)**- For row access loc- label base, iloc- integer position

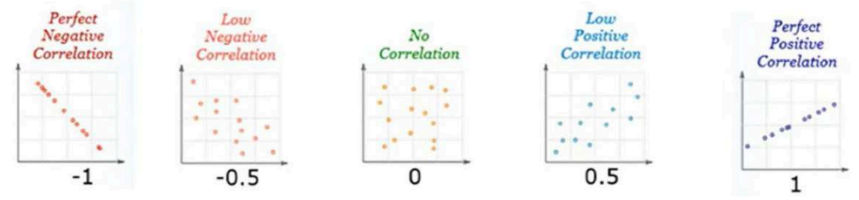
BOXPLOT: Lower Inner fence=Q1-1.5×IQR | Upper Inner fence=Q3+1.5×IQR | IQR = Q3 - Q1 | Outliers-> Values outside Inner fence
Boxplot Outliers - Low: < Q1-1.5·IQR | High: > Q3+1.5·IQR | Box Plot -> Mean/ Std dev for skewed data | Outer fence= Q1-3*IQR & Q3+3*IQR

Text Process: Convert lowercase->Tokenisation->Remove stop words->Lemetisation/Stemming-> Numeric Filtering

PEARSON CORRELATION: Measures the strength of a linear relationship between two continuous variables. Since the relationship here is non-linear (income rises, then flattens), Pearson will underestimate the strength of the relationship. You might get a lower correlation coefficient (r), even though there is a strong monotonic trend overall. It produces a value between –1 and 1, where 1 indicates a perfect positive linear relationship, –1 indicates a perfect negative linear relationship, and 0 indicates no linear relationship. This means that when one variable increase or decreases, the Pearson correlation shows whether the other variable tends to increase or decrease in the same or opposite direction.

SPEARMAN CORRELATION: Looks at monotonic relationships, considers rank order rather than actual values. It's more suitable here because: Income still generally increases as age increases (monotonic). Spearman captures this non-linear but consistent direction better. Spearman's correlation measures the strength and direction of the relationship between two variables when they are monotonically related. This means that the relationship is consistent in direction (either always increasing or always decreasing), regardless of whether it is linear or nonlinear. The data can be ranked but the distances between ranks may not be equal. Like Pearson's correlation, Spearman's correlation produces a value between – 1 and 1, capturing both linear and nonlinear monotonic relationships.

Pearson Correlation - Best for: Interval and Ratio variables | **Spearman's Rank Correlation** - Best for: Ordinal, Interval, and Ratio | **Kendall's Tau** - Best for: Ordinal, Interval and Ratio



WEEK 4 & 5

Relation - Named, 2D table with rows and columns | **Schema** - Describes structure- column names & data types | **Instance** - Actual data in table

Data Definition Language (DDL)	CREATE, DROP, ALTER	CREATE TABLE Students (id INT PRIMARY KEY, name VARCHAR(50), age INT CHECK (age > 0)); (Defines structure of the table)
Data Manipulation Language (DML)	INSERT, DELETE, UPDATE, SELECT	INSERT INTO Students VALUES (1, 'Alice', 21); UPDATE Students SET age = 22 WHERE id = 1; DELETE FROM Students WHERE id = 1;

Queries - SELECT * FROM Students; SELECT AVG(age) FROM Students; SELECT age, COUNT(*) FROM Students GROUP BY age;
Joins - SELECT s.name, o.order_id FROM Students s JOIN Orders o ON s.id = o.student_id;
Candidate Key - Unique, Minimal Identifier |
Primary Key - Chosen Candidate Key (1 per table) |
Foreign Key - Reference candidate key in another relation (logical pointer)
Composite Key - Multiple attributes forming a key

Data Loading: Connect PostGre->Load CSV(pandas)->Create tables->Insert data->Handle errors
Normalisation: 1NF->2NF->3NF: few restriction | BCNF->4NF->5NF: less redundancy | Used to eliminate redundant data and prevalent anomalies
OLTP (Online Transactional Processing): Focussed on normalisation and Efficient for updates and Flexible for dynamic relationships
OLAP (Online Analytical Processing): Data warehousing approach and Optimised for Queries and Handles historical data
Data Warehouse: Organised by subject not application | Multiple heterogeneous data sources | Large historical data with time attributes | Infrequent updates, often append only.

Aspect	Star Schema	Snowflake Schema
Best for	Simple, high-performance OLAP systems	Complex data relationships with storage optimization
Use in	Business Intelligence tools (e.g., Power BI, Tableau)	Data warehouses with strict normalization and integrity requirements
Example	Retail company analyzing daily sales across stores, products, and time	Bank analyzing transactions with detailed customer and account hierarchies
Pros	- Easy to understand and query - Fast performance (fewer joins)	- Reduces redundancy - Better data integrity
Cons	- Redundant data (due to denormalized dimension tables)	- Slower queries (more joins) - More complex to understand

A table is a relation only if every value in every attribute is atomic which means only 1 value in each row and column. To fix this create another relation or table that maps to it.

Ex: CREATE Table Sales_Order(ORDER_ID CHAR(4) PRIMARY KEY, Date_Of_Purchase DATE, Store_ID VARCHAR(10), MOVIE_ID INT, FOREIGN KEY(MOVIE_ID) **REFERENCES** Movie_Details)Movie_ID));

Ex InnerJoin: SELECT employees.first_name, departments.dept_name FROM employees **INNER JOIN** departments ON employees.dept_id = departments.dept_id;

Ex: SELECT [DISTINCT] column_list FROM table_list WHERE conditions **GROUP BY** grouping_attributes **HAVING** group_conditions **ORDER BY** (sorting_crieteria)[Asc/Desc] **LIMIT** n

Group By - How you group rows: Group rows that have same values in specified columns into summary rows. Often used with aggregate functions like SUM(), AVG(), COUNT() and so on
HAVING - Like WHERE but works only after GROUPING. Filters groups and not individual rows. Used to filter result of GROUP BY based on aggregate conditions.

Ex: SELECT dept, COUNT(*) FROM employees GROUP BY dept;

Use CHAR for fixed length, VARCHAR for variable length
Date/ Time: DATE, TIME, TIMESTAMP, INTERVAL
LIKE: LIKE '%h'- values starting with H | LIKE '%n'- values ending with 'n' | LIKE '%rr%'- values containing 'rr' anywhere | LIKE '_a%'- Values with second letter is 'a'
JOIN - Cross Product.

Ex: SELECT columns FROM Table1 JOIN Table2 ON Table1.key=Table2.key; OR **SELECT** Table1 **JOIN** Table2 USING(common_column)

Aggregate functions: COUNT(*) - count all rows, **COUNT(attr)** - Counts non null values, **AVG(attr)**- AM, **SUM(attr)**- Sum of values
Inner Join explicitly joins table based on 1 attribute so only rows with matching attributes will be joined but with **Natural joins** all attributes with same name is considered. SO you end up missing few rows.

FOR date questions: SELECT f_name, l_name, CURRENT_DATE-enrollment_date AS days_enrolled FROM Students;
SELECT f_name, h_date FROM employees **WHERE EXTRACT**(YEAR FROM h_date)=2023;

SELECT e_id EXTRACT (YEAR FROM e_date) AS event_year, EXTRACT(MONTH FROM e_date) AS event _month FROM EVENTSchedule

Add 1 hour and 30 minutes to current time - SELECT NOW() + INTERVAL '1 hour 30 minutes';

SELECT e.employee_name, d.department_name AS employee_department FROM employees e JOIN departments d ON e.department_ID = d.department_ID JOIN assignments a ON e.employee_ID = a.employee_ID JOIN projects p ON a.project_ID = p.project_ID WHERE e.department_ID <> p.department_ID;

SELECT f.title FROM Film f JOIN Film_Actor fa ON f.film_id = fa.film_id JOIN Actor a ON fa.actor_id = a.actor_id WHERE a.first_name = 'JOHNNY' AND a.last_name = 'CAGE' ORDER BY f.title;

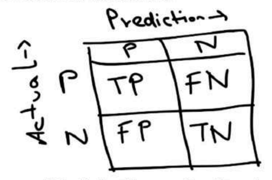
WEEK 6

Observational Studies: Observing with applying treatments | Passive Participation of researcher | Records obs without controlling conditions | Only establishes correlation, not causality | Sample Surveys | Ex: Tanning beds & Skin Cancer Tudy
Experimental Studies: Records info while applying treatments and controlling conditions | Active Participation of Researchers | Establishes causality | Randomised control trials | Strong hypo with controlled data collection | Ex: Screen time exp (Ctrl grp(30mins) vs Treatment Grp(2 hrs))
Dependent - Change that happens coz of independent variable
Independent - One thing you change
Controlled - Things you want constant and unchanging
Research Question(Q)- Ask whether independent variable has effect
Null hypo(H0)-Assumption that there is no effect
Alternate Hypo(H1)- There is effect
P value - Prob of observing data at least as extreme as what was observed assuming H0|**Significance level**(α)- Prob of wrongly rejecting H0 when true

	H0 True	H0 False
Accept H0	✓	Type 2 error
Reject H0	Type 1 error	✓

(<α) → Strong evidence against H0-> **Reject H0** | (>α) → Weak evidence against H0-> **Accept H0** | (=α) → Marginal-> **Nothing to do**

Confusion Matrix:



Accuracy = $\frac{TP+TN}{N}$ | Recall = $\frac{TP}{TP+FN}$ |
Precision = $\frac{TP}{TP+FP}$ | F1 = $\frac{2PR}{P+R}$ (HM of P,R)

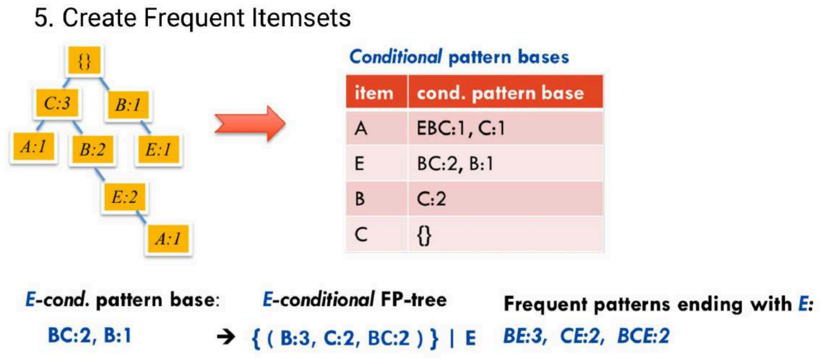
Cross Validation: Split data randomly-> Training(2/3) & Testing (1/3) | Repeat k times and take avg accuracy===**Holdout Method**
Partition data into K mutually exclusive subsets | Use k-1 for training, 1 for testing | Repeat k times| Leave one out(Incase k=no of samples)===**K fold cross validation**

Testing:	Normal Data	Not Normal Data
2 independent groups	Unpaired t- test Eg: Test score GrpA(boys), GrpB(girls)	Mann Whitney U test Eg: Reaction Time: 2 UIs but data skewed
Multiple Independent Groups	ANOVA Eg: Avg marks across 3 schools(A,B,C)	Kruskall Wallis H test Eg: Customer satisfaction scores 4 diff centres with not normal data
Paired/ Related Data	Paired t test Eg: Weight b4 and after 3month program for same people	Wilcoxin signed rank test Eg: Stress level from before and after meditation for survey

WEEK 7

FP TREE

1. Calculate Frequency of each item
2. Check frequency with min_sup
3. Arrange transactions based on frequency
4. Create FP Tree from Root Node based on transactions
5. Create Frequent Itemsets



All Frequent Itemsets: { A:2, E:3, B:3, C:3, BC:2, EC:2, AC:2, EB:3, BCE:2 }

APRIORI ALGORITHM

1. Generate Frequency itemset for each combination Eg: {a}{b}{ab}
2. Check with min_sup
3. Create Frequent Itemset

CONFIDENCE

1. Create possible association rules w.r.t frequency itemset

$$\{C,E \rightarrow B\} \qquad confidence(\{C,E\} \rightarrow \{B\}) = \frac{\{B,C,E\}}{\{C,E\}} = \frac{2}{2} = 1$$

2. Check with min confidence threshold.

WEEK 8

Supervised - Training data has labels (classification/ regression) and Each input comes with the correct output and Maps inputs to outputs | **Unsupervised** - No labels and discover hidden patterns or groups.

	Apriori	FP-Growth
Cand Generation	Yes	No
Database Scane	Multiple	2
Memory Usage	Lower	Higher
Performance	Slower	Faster
Implementation	Simpler	More Complex

Clustering is an unsupervised learning technique used for grouping similar data.

- Group data points so that similar ones are within the same cluster.
- Dissimilar data points should be in different clusters.
- Intra-cluster distances should be minimized.
- Inter-cluster distances should be maximized.

Distance Measurement:
Normally, distances are calculated using the Minkowski Distance:
Minkowski Distance Formula:
$D(p) = (\sum |x_i - y_i|^p)^{1/p}$ where p=parameter defining distance
Manhattan Distance (p = 1):
$D = \sum |x_i - y_i|$
Points (2, 3) and (5, 1)→D = |2 − 5| + |3 − 1| = 3 + 2 = 5
Euclidean Distance (p = 2):
$D = \sqrt{\sum (x_i - y_i)^2}$
Points A(1, 2), B(1, 4)→ D= $\sqrt{(1-1)^2 + (2-4)^2} = \sqrt{0+4} = \sqrt{4} = 2$

1. **Partitional Clustering**- Divides data into non-overlapping subsets | Each data point belongs to exactly one cluster | Ex: K-means
2. **Hierarchical Clustering**- Creates a hierarchy of clusters (tree-like structure) | Two approaches: **Agglomerative (Bottom-Up)**: Starts with each data point as its own cluster and merges them iteratively | **Divisive (Top down)** - Starts with all points in 1 cluster
K-means- Choose the number of clusters, *k* | Initialize *k* centroids randomly | Assign each data point to the nearest centroid | Recalculate centroids as the mean of all points in each cluster | Repeat steps 3 and 4 until centroids no longer change significantly (convergence) |
Time Complexity -O(n*k*i*d)–(points, clusters, iterations, dimensions)
Ex: P1(0,2), P2(2,0), P3(3,1), p4(5,1)- Assume Clusters-P1, P3

Data Points	0	2	3	1	Cluster
P1	0	2	3.16		1
P2	2	0	2.83	1.41	2
P3	3	1	3.16	0	2
P4	5	1	5.10	2	2

Now find mean cluster 1-(1,2), 2-((2+3+5)/2, (0+1+1)/3). Now these become new cluster. Keep doing this till 2 consecutive clusters match
Agglomerative algo: Start with each point as its own cluster | Find closest pair of clusters | Merge them into 1 cluster | Recompute distances with min value coming | Repeat until desired number of clusters (Single Linkage-Min dist btn 2 points, Complete Linkage- Max dist btn 2 points, Avg Linkage- Mean distance btn pairs)
Cluster Evaluation: External Index(Accuracy, Precision, Recall)- Compare with known labels | Internal Index(SSE, Silhouette)- Values w/o external info | Compare different clusterings
External Measures: Homogeneity-Each cluster only contains 1 class(like prec) | Completeness-All members of class in same cluster(like recall) | V-measure-HM of H & C-> like F1-score

Internal Measures: Sum of Squared Errors(SSE)=$\sum_{i=1}^{k}\sum_{x \in C_i} dist^2(x, m_i)$
where m is centroid of cluster C-> Lower the SSE, better the clustering
Ex: 3 clusters->1-[2,4] w centroid 3, 2-[5,6,7] w centroid 6, 3-[8,10,12] w centroid 10. So SE1=(2-3)²+(4-3)²=1+1=2, SE2=(5-6)²+(6-6)²+(7-6)²=1+1=2, SE3=4+4=8. **SSE=SE1+SE2+SE3**= 2+2+8=12

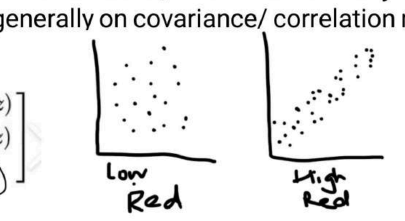
Silhouette Coeff(S)= $\frac{(b-a)}{max(a,b)}$ where a-> avg distance of points to same cluster, b->avg distance to points in nearest other cluster->**Closer to 1 is better and Range[-1,1]**

Ex: 3 clusters-1[(1,0),(1,1)], 2[(1,2),(2,3),(2,2),(1,2)], 3[(3,1),(3,3),(2,1)]→Take a point in C1-(1,0)→a=$\sqrt{(1-1)^2+(0-1)^2}$=1→ Now for b, for same point **calculate avg** to all points in cluster 2 and 3 and take min one. Of these take min distance. So S=$s_1 = \frac{(b-a)}{max(a,b)}$

Find no of clusters:
Elbow Method → Plot SSE vs k and count number of elbows |
Silhouette analysis → Plot avg silhouette coeff vs k, Choose k with highest avg silhouette, Check silhouette plots for uniform cluster quality

High avg silhouette-> Points far from neighbouring clase | **Uniform Silhouette**-> Similar quality clusters | **Varying silhouette**-> Some clusters better than others.
Pre-processing for clustering: Data Cleansing-> Data Transformation-> Data Normalisation -> Dimensionality Reduction
PCA (Principal Component Analysis) - Used for dimensionality reduction (Transform high dimensional data to lower dimensions) -> Visualisation-Better Plotting and insights -> Noise Reduction-Remove redundant Information | When to use? Variables are highly correlated, need to reduce dimensions for computation/ visualisation, want to identify most important features PCA is done generally on covariance/ correlation matrix

$$C = \begin{bmatrix} var(x) & cov(x,y) & cov(x,z) \\ cov(x,y) & var(y) & cov(y,z) \\ cov(x,z) & cov(y,z) & var(z) \end{bmatrix}$$

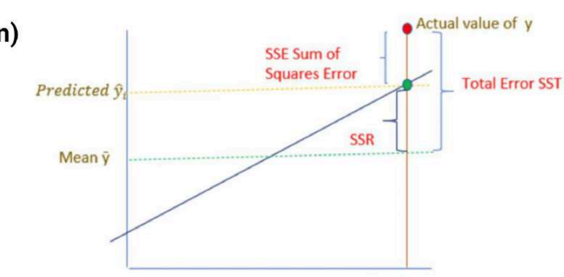


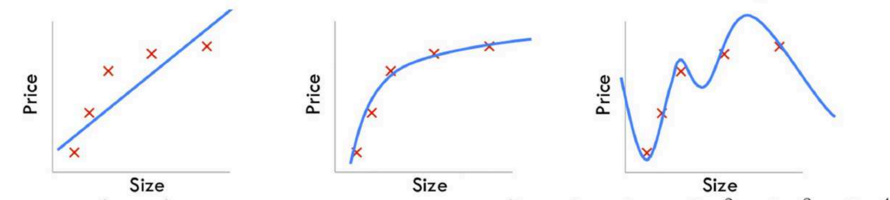
Diagonal is variance of x,y,z
Var(x), var(y), var(z) should be equal otherwise the data is redundant.
1st PC=(var(x)/(var(x)+var(y)+var(z)) also called variability.

WEEK 9

R² Range -> [0,1]
(Coefficient of Determination)

- It is square of correlation coefficient between x&y.
- It conveys goodness of fit but not precision.
- Higher R² -> Better Fit



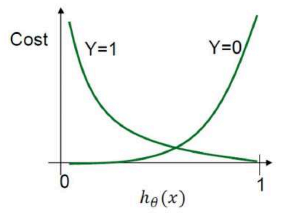


$\theta_0 + \theta_1 x$	$\theta_0 + \theta_1 x + \theta_2 x^2$	$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$
Under-fitting (high bias)	Correct fit	Overfitting (high variance)

Underfit -> High Bias, Low Variance | Overfit -> Low Bias, High Variance

Model	Cost Function Type	Why?
Linear Regression	Mean Squared Error (MSE)	Square of linear function
Logistic Regression	Log Loss / Cross-Entropy	Log of sigmoid is convex

Polynomial transformations (Vector Form) can fit non-linear datasets using linear regression techniques
Gradient Descent: Uses cost functions (how far our predictions are from actual value) **α is learning rate** (too small → low convergence) (too large might overshoot minima) | Minimize cost functions to get better predictions by optimizing parameters. In case of multiple local minim
Basic: New Value = Old Value +/- Stepsize (Stepsize = α * Slope)
To find value of x, to minimize y(x),derivate it every time and equate it to 0. Pick a random no. (x) derivate equation once and put x and equate to 0. If result is +ve next no. picked should be lesser than x and vice versa since, derivative at minimum should be 0. Keep doing this till we get minima.
Batch GD – uses entire dataset per update → accurate but slow
Stochastic GD – uses one example per update → fast but noisy.
Regression: Assigns numerical value | Output is continuous value | Eg: Predict House price. **Classification**: Assigns class to each example | Output is discrete (categorical variable). Eg: Yes/No, Spam/Ham
Logistic Regression: Classification Model | Output b/w 0 & 1.



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

The objective is to minimize cost
- For class 0, it minimizes $h_\theta(x)$
- For class 1, it maximizes $h_\theta(x)$

Regularizations are for solving overfitting by adding penalty term to cost function). It aims to penalize for large values of coefficients. We can also address overfitting by eliminating features. → **L1 (Lasso):** Estimates sparce coef. which is feature selection | Adds penalty proportional to SSE of coef. → **L2 (Ridge):** Minimizes coef. i.e pulls coef. towards 0 | Adds penalty to SSE (absolute).

WEEK 9

Simple Linear Regr. $Y = \alpha + \beta \cdot X + \varepsilon$ | $\hat{\alpha} = \bar{y} - \hat{\beta} \cdot \bar{x}$ | x_i, y_i = actual/observed value of x,y | $\hat{y}_i$ = predicted value | $\bar{x}, \bar{y}$ = arithm. mean of x, y

$$\hat{\beta} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{cov(x,y)}{\sigma^2(x)} = \frac{\rho(x,y) \cdot \sigma(y)}{\sigma(x)} \qquad R^2 = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{SSE}{SST}$$

SSE	Sum of squared errors	$\varepsilon = SSE = \sum (y_i - \hat{y}_i)^2$	→ the smaller, the better the predictive power
SSR	Sum of squared regression	$SSR = \sum (\hat{y}_i - \bar{y})^2$	→ the larger the value, the better the model fits the data
SST	Sum of squared total	$SST = \sum (y_i - \bar{y})^2$	→ Overall dispersion from its mean. The smaller, the closer the values are to the mean

Multiple linear regression

$$Y = h_\theta(x) = \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \dots + \theta_d \cdot x_d = \sum_{j=0}^{d} \theta_j \cdot x_j$$

with polynomial transformation

$$Y = h_\theta(x) = \theta_0 + \theta_1 \cdot x^1 + \theta_2 \cdot x^2 + \dots + \theta_d \cdot x^d = \sum_{j=0}^{d} \theta_j \cdot x^j$$

Cost function $J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda \cdot \sum_{j=1}^{d}\theta_j^2$ ⇒ Fit model by solving $\min_\theta J(\theta)$

Gradient descent: Choose initial value for $\theta_0, \theta_1, \dots, \theta_d$, calculate cost $J(\theta)$ until you reach minimum (until convergence)

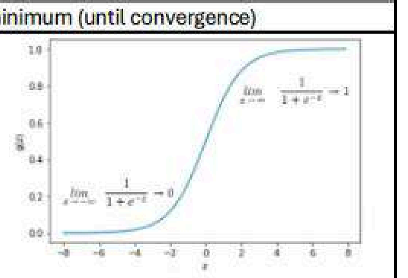
$$\theta_j \leftarrow \theta_j - \alpha \cdot \frac{\partial}{\partial \theta_j} J(\theta)$$

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

Logistic regression

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

If $Z \geq 0$, then $g(z) = Y \geq 0.5$ → predict Y = 1
If $Z < 0$, then $g(z) = Y \leq 0.5$ → predict Y = 0



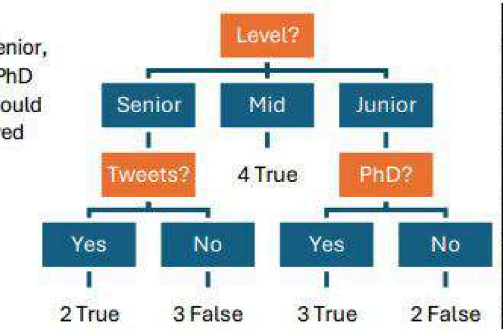
$$J(\theta) = -\frac{1}{2n}\sum_{i=1}^{n}\left[y^{(i)}\log\left(h_\theta(x^{(i)})\right) + (1 - y^{(i)})\log\left(1 - h_\theta(x^{(i)})\right)\right]$$

min $J(\theta)$ minimizes $h_\theta(x)$ for Y = 0 and maximizes $h_\theta(x)$ for Y = 1

WEEK 10

IG = effective change in entropy after making a decision based on the value of an attribute X	$IG(Y\|X) = H(Y) - H(Y\|X)$	Y = class label, X = attribute
Entropy = measures the uncertainty associated with data (higher entropy = higher uncertainty)	$H(Y) = -\sum_{i=1}^{m} p_i \log_2(p_i)$	$p_i = p(Y = y_i)$ = probability of Y being y_i, m = number of classes
Conditional entropy	$H(Y\|X) = \sum_j p(X = v_j) \cdot H(Y\|X = v_j)$	$p(X = v_j)$ = probability of X being v_j, $H(Y\|X = v_j)$ = entropy of Y with $X = v_j$

Example:
Applicant X = (Level = Senior, Lang = R, Tweets = No, PhD = No) → Decision tree would yield "No" for "interviewed well".



Q: Is "Level" a good 1st split of the decision tree?

Applicant	Level	Lang	Tweets	PhD	Interviewed well
A1	Senior	Java	No	No	False
A2	Senior	Java	No	Yes	False
A3	Senior	Python	No	No	False
A4	Senior	R	Yes	No	True
A5	Senior	Python	Yes	Yes	True
A6	Mid	Java	No	No	True
A7	Mid	R	Yes	Yes	True
A8	Mid	Python	No	Yes	True
A9	Mid	Java	Yes	No	True
A10	Junior	Python	No	No	True
A11	Junior	R	Yes	No	True
A12	Junior	Python	Yes	Yes	False
A13	Junior	R	Yes	Yes	False
A14	Junior	Python	No	Yes	False

$$H(Iw) = -\left(\frac{9}{14} \cdot \log_2\left(\frac{9}{14}\right) + \frac{5}{14} \cdot \log_2\left(\frac{5}{14}\right)\right) = 0.94$$

Senior	$p(Level = Senior) = \frac{5}{14} = 0.36$	2 interviewed well, 3 did not	$H(Iw\|Level = Senior) = -\left(\frac{2}{5} \cdot \log_2\left(\frac{2}{5}\right) + \frac{3}{5} \cdot \log_2\left(\frac{3}{5}\right)\right) = 0.97$
Mid	$p(Level = Mid) = \frac{4}{14} = 0.29$	4 interviewed well, 0 did not	$H(Iw\|Level = Mid) = -\left(\frac{4}{4} \cdot \log_2\left(\frac{4}{4}\right) + \frac{0}{4} \cdot \log_2\left(\frac{0}{4}\right)\right) = 0$
Junior	$p(Level = Junior) = \frac{5}{14} = 0.36$	3 interviewed well, 2 did not	$H(Iw\|Level = Junior) = -\left(\frac{3}{5} \cdot \log_2\left(\frac{3}{5}\right) + \frac{2}{5} \cdot \log_2\left(\frac{2}{5}\right)\right) = 0.97$

$(Iw\|Level) = 0.36 \cdot 0.97 + 0.26 \cdot 0 + 0.36 \cdot 0.97 = 0.7$ | $HIG(Iw\|Level) = 0.94 - 0.7 = 0.24$

WEEK 11

Tokenisation	Normalisation	Indicator features	Term frequency weighting
split a string into **tokens**, remove punctuation	Lemmatisation: avoid gram. sparseness, e.g. "was" ⇒ "be". Lower-casing, encoding	binary indicator feature (0 or 1) for each word, ignore frequencies	give more weights to common terms in a doc

Given the attribute vector $X(x_1, x_2, \dots, x_n)$, what is the probability of classifying it as C_i?

$$P(C_i\|X) = P(X\|C_i) \cdot P(C_i) \text{ w/ } P(X\|C_i) = \prod_{k=1}^{n} P(x_k\|C_i) = P(x_1\|C_i) \cdot P(x_2\|C_i) \cdot \ldots \cdot P(x_n\|C_i)$$

Bayes theorem: $P(C_i\|X) = \frac{P(C_i \cap X)}{P(X)} = \frac{P(X\|C_i) \cdot P(C_i)}{P(X)}$

WEEK 11

Example

	Head	Eyes	Size	Ven.
1	Triangle	Elliptical	Small	Yes
2	Round	Round	Small	No
3	Narrow	Elliptical	Small	No
4	Narrow	Round	Large	No
5	Narrow	Elliptical	Large	Yes
6	Triangle	Round	Small	No
7	Narrow	Round	Large	No
8	Round	Elliptical	Small	No
9	Triangle	Elliptical	Small	Yes

$X = (Head = narrow, Eyes = elliptical, Size = large)$

Solution

$P(C_{Yes}) = \frac{4}{9} = 0.444$ | $P(C_{No}) = \frac{5}{9} = 0.556$

Compute $P(X\|C_{Yes})$ for each attribute:
$P(Head = narrow\|C_{Yes}) = \frac{1}{4} = 0.25$
$P(Eyes = elliptical\|C_{Yes}) = \frac{3}{4} = 0.75$
$P(Size = large\|C_{Yes}) = \frac{1}{4} = 0.25$
$P(X\|C_{Yes}) = 0.25 \cdot 0.75 \cdot 0.25 = 0.0469$
$P(C_{Yes}\|X) = P(X\|C_{Yes}) \cdot P(C_{Yes}) = 0.0469 \cdot 0.444 = 0.02083$

Compute $P(X\|C_{No})$ for each attribute:
$P(Head = narrow\|C_{No}) = \frac{3}{5} = 0.6$
$P(Eyes = elliptical\|C_{No}) = \frac{2}{5} = 0.4$
$P(Size = large\|C_{No}) = \frac{3}{5} = 0.6$
$P(X\|C_{No}) = 0.6 \cdot 0.4 \cdot 0.6 = 0.144$
$P(C_{No}\|X) = P(X\|C_{No}) \cdot P(C_{No}) = 0.144 \cdot 0.556 = 0.08$

$P(C_{Yes}\|X) < P(C_{No}\|X)$ – The prediction will be that the example it not venomous.

Example Text Classification

Text	Category
"A great game"	Sports
"the election was over"	Not sports
"Very clean match"	Sports
"A clean but forgettable game"	Sports
"It was a close election"	Not sports

→ Build a classifier that tells whether, e.g., "a very close game" belongs to the category "Sports"

$$p(Sports\|\text{"a very close game"}) = \frac{p(\text{"a very close game"}\|Sports) \cdot p(Sports)}{p(\text{"a very close game"})}$$

p("a very close game"|Sports) = p("a"|Sports) · p("very"|Sports) · p("close"|Sports) · p("game"|Sports)

The problem: p(close|Sports) = 0 because "close" does not appear in the category Sports

Therefore: Laplace smoothing

$$p(w\|c) = \frac{count(w\|c) + 1}{count(word, c) + count(word)}$$

count(w|c) = count of word w in class c; count(word, c) = count of all words in class c; count(word) = count of all distinct words in the dataset

$p(close\|Sports) = \frac{0+1}{11+14}$

w	p(w | Sports)	p(w | Not Sports)
a	$\frac{2+1}{11+14}$	$\frac{2+1}{9+14}$
very	$\frac{1+1}{11+14}$	$\frac{0+1}{9+14}$
close	$\frac{0+1}{11+14}$	$\frac{1+1}{9+14}$
game	$\frac{2+1}{11+14}$	$\frac{0+1}{9+14}$

Example Linear Regression

X	Y
1	1
2	2
3	1.3
4	3.75
5	2.25

$n = 5$
$\bar{x} = \frac{1}{5}(1.0 + 2.0 + 3.0 + 4.0 + 5.0) = 3.0$
$\bar{y} = \frac{1}{5}(1.00 + 2.00 + 1.30 + 3.75 + 2.25) = 2.06$
$Cov(x, y) = \frac{1}{n-1}\sum(x_i - \bar{x})(y_i - \bar{y})$
$Cov(x, y) = \frac{1}{4}[(1.0 - 3.0)(1.00 - 2.06) + \dots + (5.0 - 3.0)(2.25 - 2.06)] = 1.0625$
$Var(x) = \frac{1}{n-1}\sum(x_i - \bar{x})^2$ | $Var(x) = \frac{1}{4}[(1.0 - 3.0)^2 + \dots + (5.0 - 3.0)^2] = 2.5$
$b = \frac{Cov(x, y)}{Var(x)}$ | $b = \frac{1.0625}{2.5} = 0.425$
$a = \bar{y} - b\bar{x}$ | $a = 2.06 - 0.425 \times 3.0 = 0.785$

Therefore, the linear regression model for the data is
$y = a + bx$ $\qquad y = 0.785 + 0.425x$