Applied Machine Learning

Dr. Harikrishnan N B
Computer Science and Information Systems

BITS Pilani
Pilani Campus

innovate achieve lead

SE ZG568 / SS ZG568, Applied Machine Learning

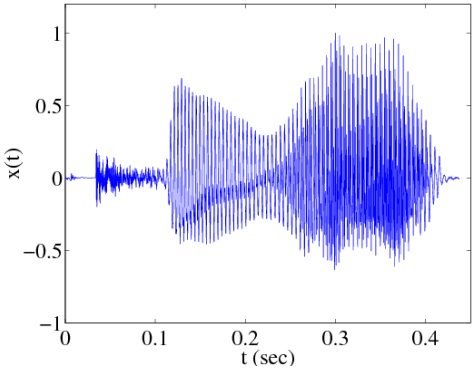
Lecture No. 2 [02- Feb-2025]

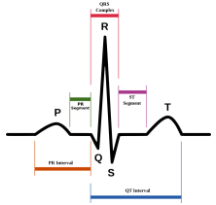
Recap

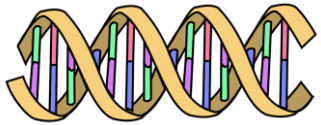
‘Learning’ in Machine Learning
Types of Learning
Supervised Learning Setup
k-NN
Training, Crossvalidation, Testing
Performance Metric

Data is the king - AI Community

What is **Data?**





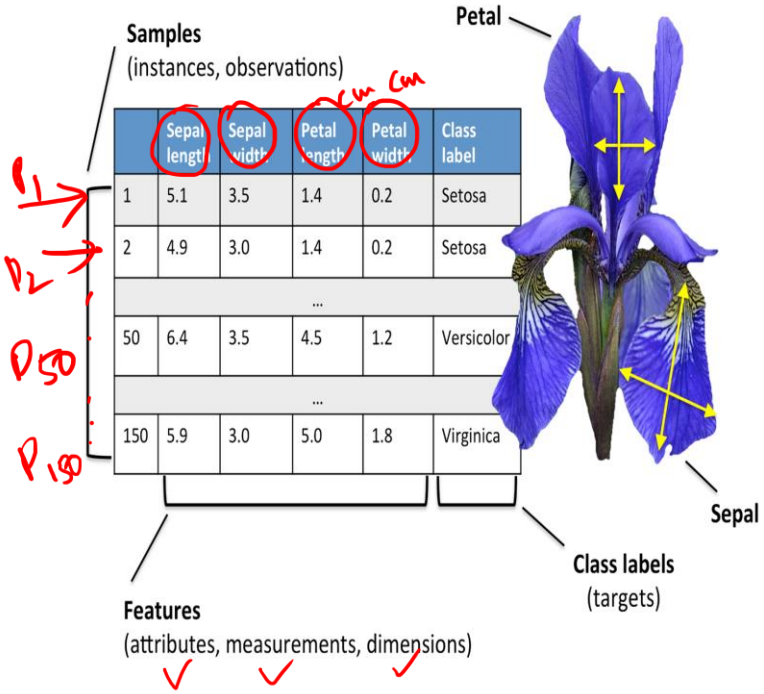


Data is everywhere!

Data Format

f_1	f_2	….	f_n	Output (Y)
a_1	a_2	….	a_n	y_1
b_1	b_2	….	b_n	y_2
.	.	….	.	.
.	.	….	.	.

- f_1, f_2, … f_n are called features
- Each row is a data instance or a feature vector
- Y is the output variable.



Iris Plant
Iris Setosa
Iris Versicolor
Iris Virginica
Three class classification

f_1 f_2 f_3 f_4

150 × 4

1 < 2 < 3 < 4

Blue > Brown

ID	Name	Eye Color	Blood Type	Favorite Sport
1	Alice	Blue 1	O	Football
2	Bob	Brown 2	A	Basketball
3	Charlie	Green 3	B	Cricket
4	David	Hazel 4	AB	Tennis

Nominal Data

What is Nominal Data?

- **Definition:** Nominal data is categorical data that represents names or labels with no inherent order or ranking.
- **Key Characteristics:**
  - Discrete categories
  - No numerical meaning
  - Cannot be logically ordered
  - Used for classification & identification

Examples of Nominal Data

- **Personal Information:** Eye color (Brown, Blue, Green), Blood type (A, B, AB, O)
- **Demographics:** Marital Status (Single, Married, Divorced)
- **Technology:** Web Browsers (Chrome, Firefox, Safari)

Preprocessing Nominal Data

ID	Color	Shape	Category
1	Red	Circle	A
2	Blue	Square	B
3	Green	Triangle	A
4	Red	Square	C
5	Blue	Circle	B

Why label encoding is not preferred for Nominal Data

ID	Color	Shape	Category
1	Red	Circle	A
2	Blue	Square	B
3	Green	Triangle	A
4	Red	Square	C
5	Blue	Circle	B

Here, the `LabelEncoder` assigns:

- `Color`: Red → 2, Blue → 0, Green → 1
- `Shape`: Circle → 0, Square → 1, Triangle → 2
- `Category`: A → 0, B → 1, C → 2

ID	Color	Shape	Category
1	2	0	0
2	0	1	1
3	1	2	0
4	2	1	2
5	0	0	1

Issue with Label Encoding for Nominal Data

Since label encoding assigns numbers, models might mistakenly interpret them as ordered (e.g., thinking **Red > Green > Blue**). But in reality, **nominal data has no inherent order**.

One Hot Encoding

ID	Color	Shape	Category
1	Red	Circle	A
2	Blue	Square	B
3	Green	Triangle	A
4	Red	Square	C
5	Blue	Circle	B

ID	Color_Blue	Color_Green	Color_Red	Shape_Circle	Shape_Square	Shape_Triangle	Category_A	Category_B	Category_C
1	0	0	1	1	0	0	1	0	0
2	1	0	0	0	1	0	0	1	0
3	0	1	0	0	0	1	1	0	0
4	0	0	1	0	1	0	0	0	1
5	1	0	0	1	0	0	0	1	0

Key Takeaways

- **Label Encoding** is **not ideal for nominal data** because it may create a false sense of order.
- **One-Hot Encoding** is preferred since it represents each category as a separate binary column.

Ordinal Data

What is Ordinal Data?

- Ordinal data is categorical data with a meaningful order but no fixed distance between categories.
- It represents **ranked** information.
- Example: **Customer Satisfaction Levels** (Poor, Average, Good, Excellent)

Characteristics of Ordinal Data

- Categories are **ordered** but not necessarily equidistant.
- **Comparison** is possible (e.g., "Good" is better than "Average").
- Arithmetic operations (addition, subtraction) **~~are not valid~~**. Why?
- Used in **surveys, grading systems, and Likert scales**.

Examples of Ordinal Data

- **Educational Level**: High School < Bachelor's < Master's < Ph.D.
- **Pain Severity**: No pain < Mild < Moderate < Severe
- **Hotel Ratings**: 1-star < 2-star < 3-star < 4-star < 5-star
- **Agreement Scale**: Strongly Disagree < Disagree < Neutral < Agree < Strongly Agree

Key Take Away

Tree-based models (e.g., Decision Trees, Random Forests) handle ordinal data and nominal data well without assuming a linear relationship.

Learning - Function Approximation

Problem Setting

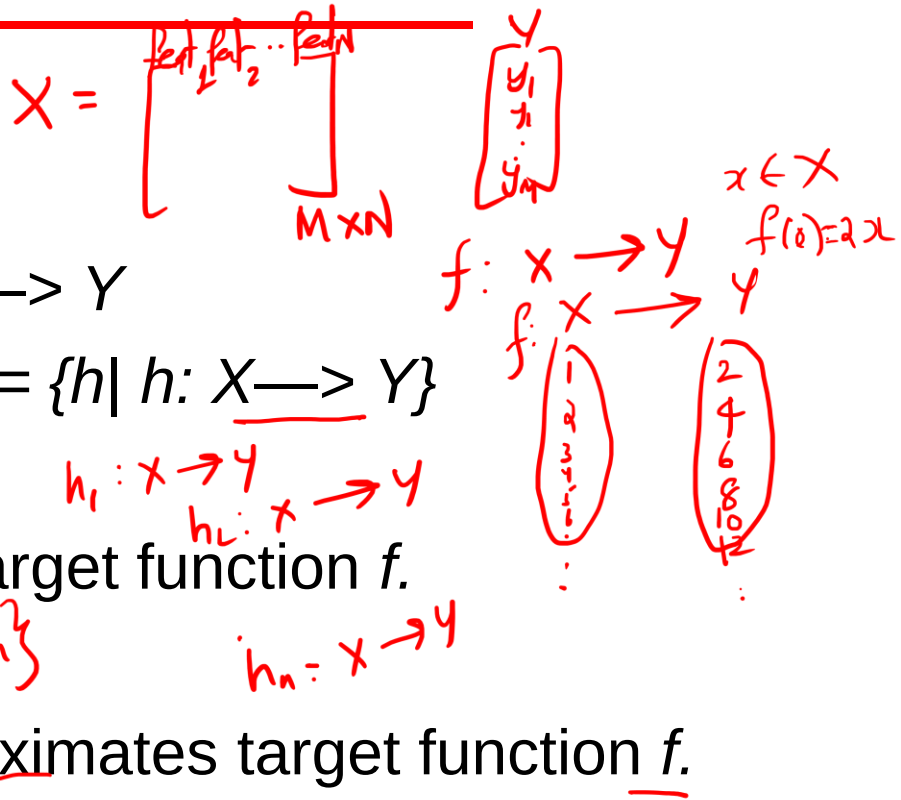
- Set of possible instances *X.*
- Unknown target function *f: X—> Y*
- Set of function hypotheses *H = {h| h: X—> Y}*

Input

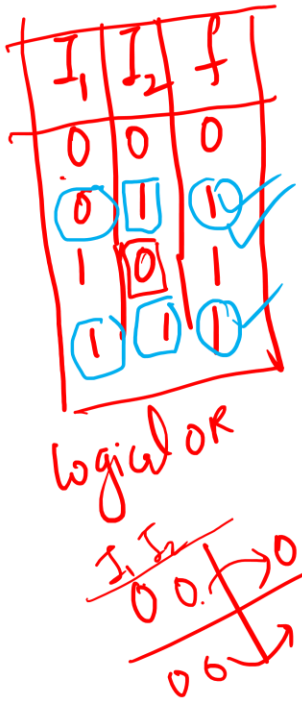
Training Examples of unknown target function *f.*

Output

Hypothesis *h ∈ H* that best approximates target function *f.*



Probabilistic Interpretation



Let $H = \{h \mid h : X \to Y\}$.

Maximum A Posteriori Hypothesis:

$$\arg\max_h P(h \mid D) = \arg\max_h \frac{p(D \mid h)P(h)}{P(D)}$$

Since $P(D)$ is constant with respect to h, we have:

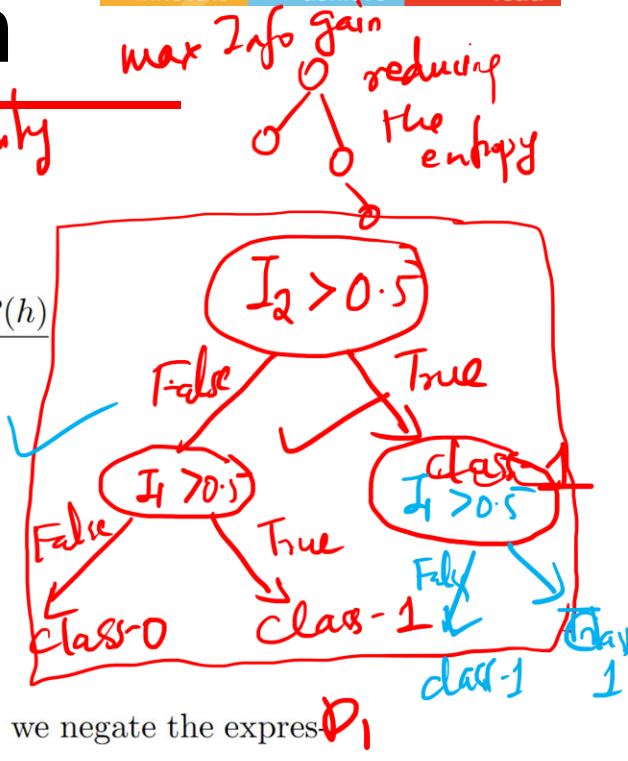
$$\arg\max_h P(D \mid h)P(h)$$

Applying the logarithm to both sides:

$$\arg\max_h \log P(D \mid h) + \log P(h)$$

Finally, to convert the maximization to minimization, we negate the expression:

$$\arg\min_h -\log P(D \mid h) - \log P(h)$$



Data Compression Interpretation

Minimum Description Length (MDL)

The MDL Principle states that the best hypothesis is the one that minimizes the total description length of the data and the model, which leads to a trade-off between fit and complexity.

Thus, the minimization problem:

$$\arg\min_h -\log P(D \mid h) - \log P(h)$$

corresponds to selecting the hypothesis h that compresses the data D efficiently (good data fit) and describes the hypothesis itself in a minimal number of bits (simple model).

Summary: The MDL principle balances two competing factors:

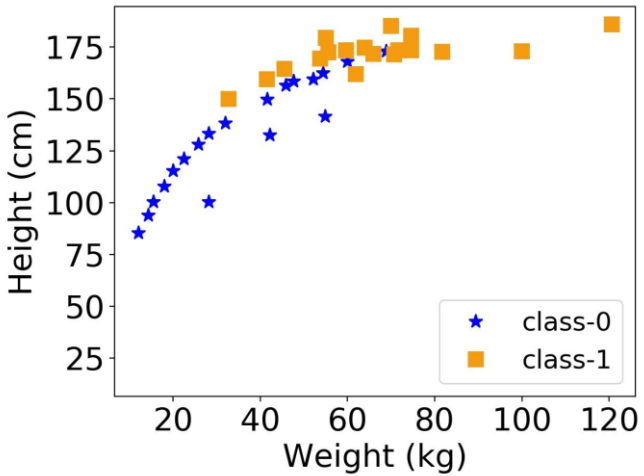
- **Fitting the data:** You want a model that fits the data well, but not too well (which could lead to overfitting).
- **Simplicity of the model:** You want a model that is as simple as possible to avoid overfitting and ensure generalization to unseen data.

Principled Way of Doing Supervised Learning

Understanding Classification Via Example

A	B	C
Weight (kg)	Height (cm)	Class
12	85.5	0
14.2	94	0
15.4	100.3	0
17.9	107.9	0
199	115.5	0
22.4	121.1	0
25.8	128.2	0
28.1	133.3	0
28.1	100.4	0
31.9	138.4	0
74.6	173.6	1
71.6	173.6	1
59.6	173.6	1
55.6	172.6	1
74.6	180.5	1
70.6	171.6	1
53.6	169.6	1
45.5	164.6	1
41.44	159.65	1
32.68	149.99	1
54.93	179.62	1

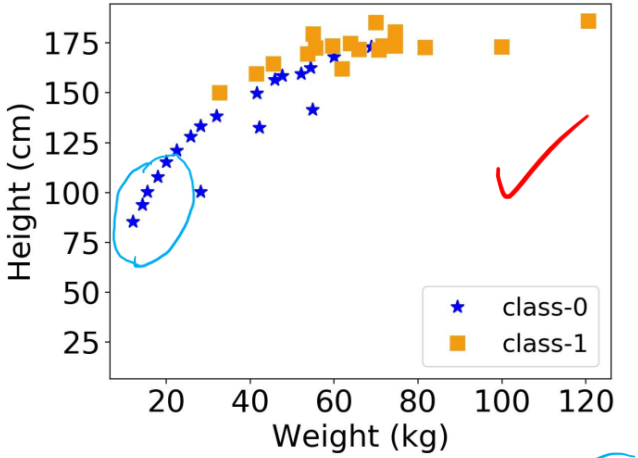
Class-0	below 15 years
Class-1	above 15 years



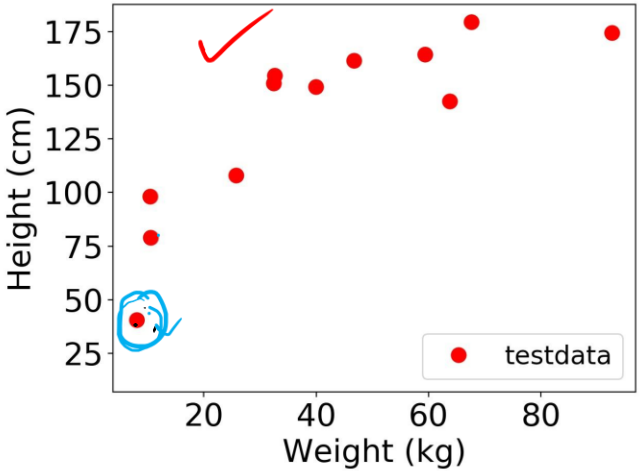
k - Nearest Neighbours

K-Nearest Neighbour Classifier

TRAINDATA



TESTDATA



$$dist(x, z) = \left(\sum_{j=1}^{r} |x_j - z_j|^p \right)^{\frac{1}{p}}$$

Mathematical Description

Let the training data $x_i \in R^2$ and its corresponding labels $y_i \in \{0, 1\}$ where $i \in \{0, 1, \ldots, N\}$

$$D = \{(x_0, y_0), (x_1, y_1), \ldots, (x_N, y_N)\}$$

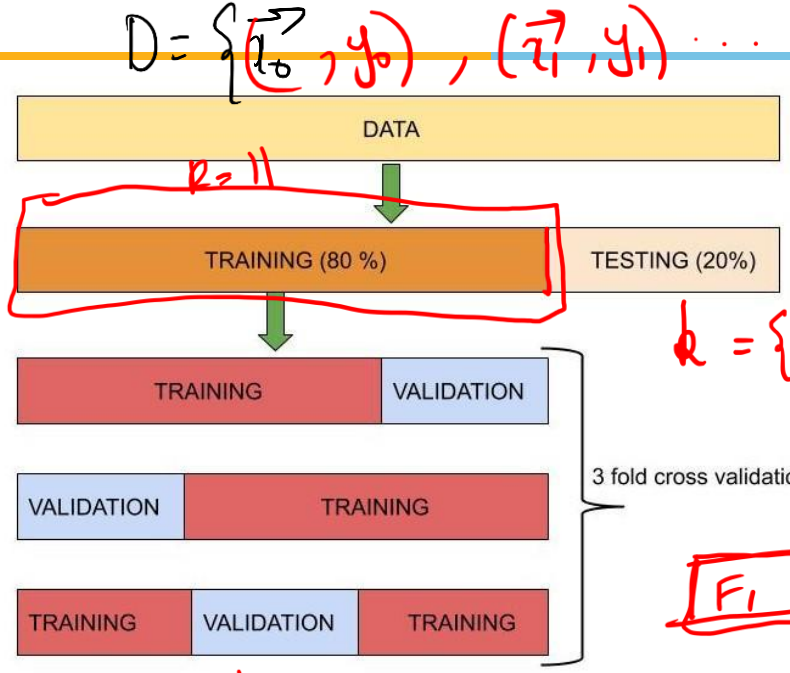
Let the set of k nearest neighbours of z be S_x where $S_x \subseteq D$ and $|S_x| = k$ (no. of elements in S_x is k).

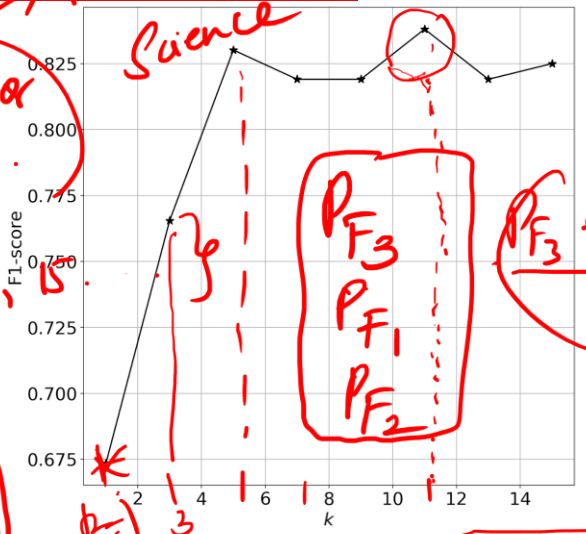
Now $\forall (x', y') \in D \backslash S_x$
(for all data instances in D but not in S_x) the following is true.

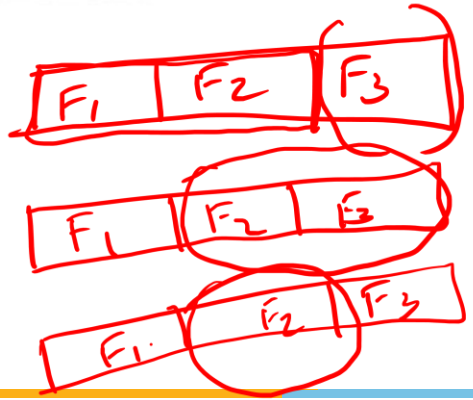
$$dist(z, x') \geq \max_{x'', y'' \in S_x} dist(z, x'')$$

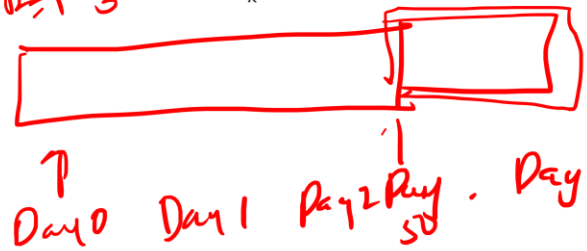
$$dist(x, z) = \left(\sum_{j=1}^{r} |x_j - z_j|^p \right)^{\frac{1}{p}}$$

How to choose K? Principled way of doing ML

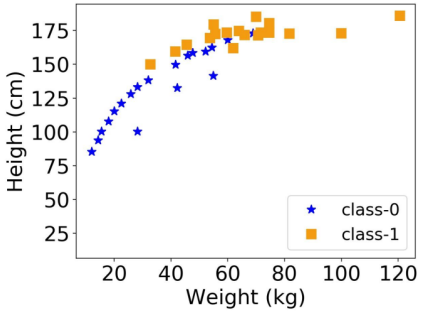


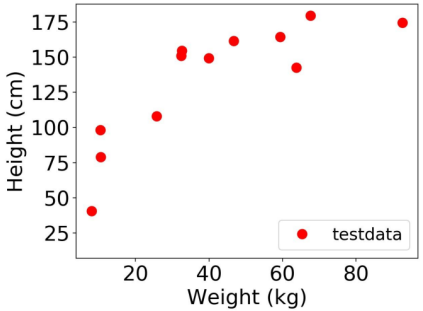


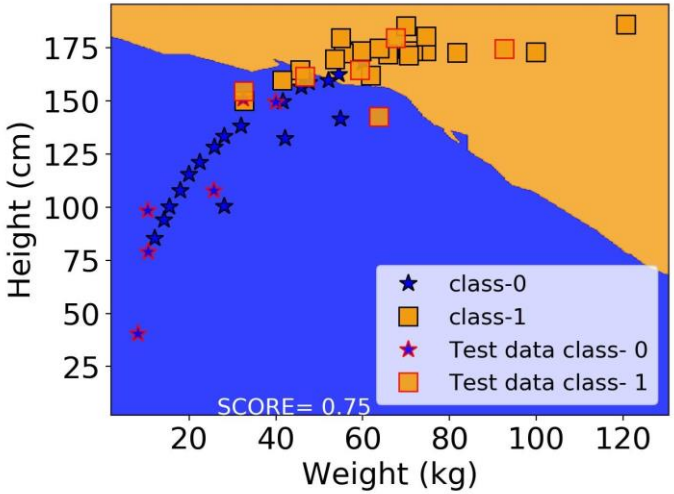




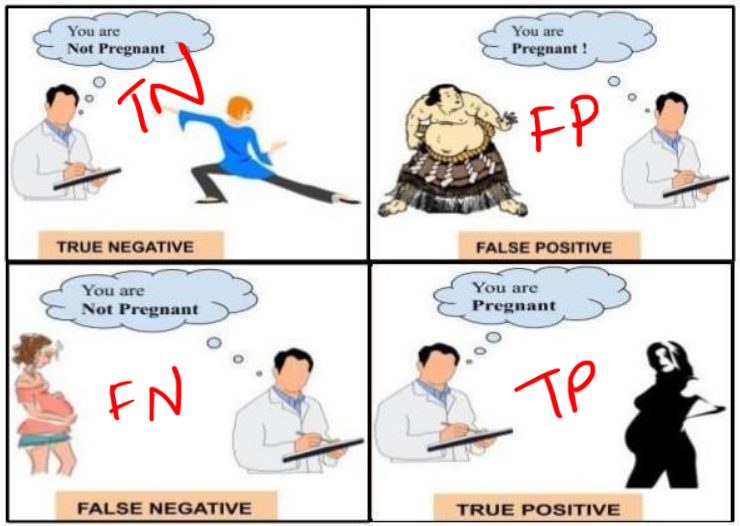
Let us see how the model performs for Test data

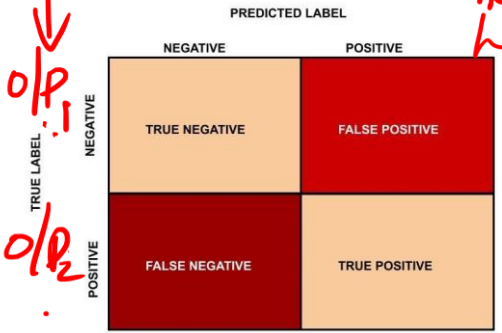




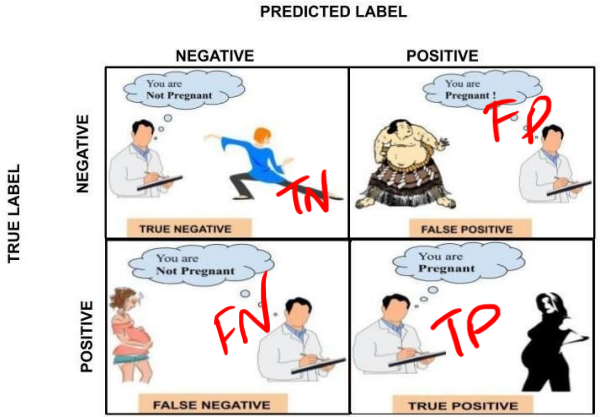


Performance Measures





Performance Measures



If we consider a binary classification problem,
C_{00} represents the count of true negative
C_{01} represents the count of false positive
C_{10} represents the count of false negative and
C_{11} represents the count of true positive.

$$\frac{2}{\frac{1}{P}+\frac{1}{R}} = \frac{2}{\frac{P+R}{PR}} \Rightarrow \frac{2xPxR}{P+R}$$

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

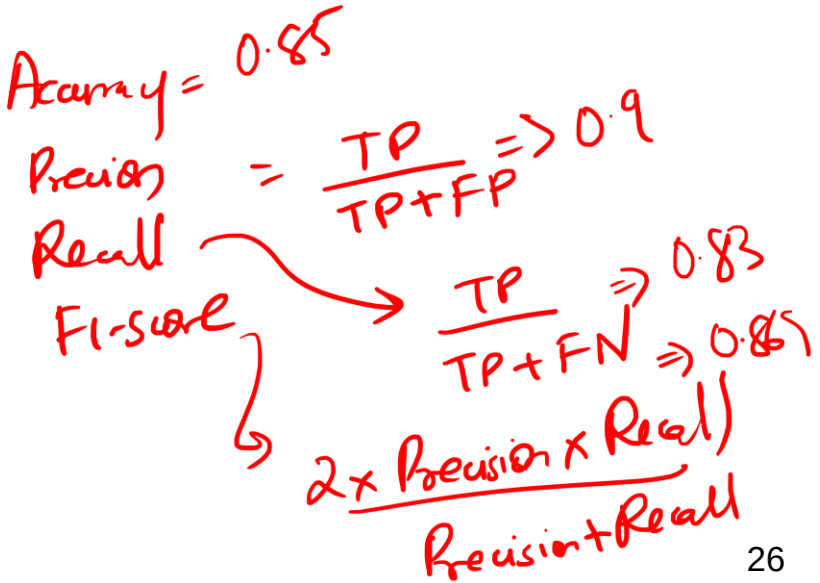
$$Precision = \frac{TP}{TP+FP} \rightarrow FP=0$$

$$Recall = \frac{TP}{TP+FN} \Rightarrow \frac{1}{FN=0}$$

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \Rightarrow \text{max}$$

Example

	Predicted Positive	Predicted Negative
Actual Positive	50	10
Actual Negative	5	35



Example

	Predicted Positive	Predicted Negative
Actual Positive	50	10
Actual Negative	5	35

Model

- Precision: $\frac{TP}{TP+FP} = \frac{50}{50+5} = 0.909$
- Recall: $\frac{TP}{TP+FN} = \frac{50}{50+10} = 0.833$
- F1-Score: $\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times 0.909 \times 0.833}{0.909 + 0.833} = 0.87$

Class Imbalance

Suppose we have 1000 samples, and the confusion matrix (with respect to the original labels) is given by:

	Predicted Positive	Predicted Negative
Actual Positive	TP = 940	FN = 10
Actual Negative	FP = 30	TN = 20

Accuracy = 0.96

Precision = 0.969

Recall = 0.989

F1 score = 0.978

Class Imbalance - Misinterpretation of Results

Suppose we have 1000 samples, and the confusion matrix (with respect to the original labels) is given by:

	Predicted Positive	Predicted Negative
Actual Positive	TP = 940	FN = 10
Actual Negative	FP = 30	TN = 20

Accuracy measures the overall proportion of correctly classified instances:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total}} = \frac{940 + 20}{1000} = \frac{960}{1000} = 0.96.$$

- **Precision$_+$**:
  The fraction of predicted positives that are truly positive.

$$\text{Precision}_+ = \frac{TP}{TP + FP} = \frac{940}{940 + 30} = \frac{940}{970} \approx 0.969.$$

- **Recall$_+$**:
  The fraction of actual positives that are correctly predicted.

$$\text{Recall}_+ = \frac{TP}{TP + FN} = \frac{940}{940 + 10} = \frac{940}{950} \approx 0.989.$$

- **F1-score$_+$**:
  The harmonic mean of precision and recall.

$$\text{F1}_+ = \frac{2 \times \text{Precision}_+ \times \text{Recall}_+}{\text{Precision}_+ + \text{Recall}_+} \approx \frac{2 \times 0.969 \times 0.989}{0.969 + 0.989} \approx \frac{1.916}{1.958} \approx 0.979.$$

For class imbalance data, accuracy leads to misleading interpretation, precision, recall, f1-score computed using positive class alone is not enough to interpret the results.

Macro Precision, Recall F1-Score

	Predicted Positive	Predicted Negative
Actual Positive	50	10
Actual Negative	5	35

	+	−
Actual +	940 TP	FN 10
Actual −	30 FP	TN 20

0.96

Class-wise Metrics:

1. **Positive Class (Class 1):**
   - Precision: $\frac{TP}{TP+FP} = \frac{50}{50+5} = 0.909$
   - Recall: $\frac{TP}{TP+FN} = \frac{50}{50+10} = 0.833$
   - F1-Score: $\frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} = \frac{2\times0.909\times0.833}{0.909+0.833} = 0.87$
2. **Negative Class (Class 0):**
   - Precision: $\frac{TN}{TN+FN} = \frac{35}{35+5} = 0.875$
   - Recall: $\frac{TN}{TN+FP} = \frac{35}{35+10} = 0.778$
   - F1-Score: $\frac{2\times0.875\times0.778}{0.875+0.778} = 0.824$

940 / 9

35 / 35+5

35 / 35+10

Macro-Averaged Metrics:

Macro-averaging treats each class equally by computing the average of the metrics across all classes.

- **Macro Precision:** $\frac{0.909+0.875}{2} = 0.892$
- **Macro Recall:** $\frac{0.833+0.778}{2} = 0.806$
- **Macro F1-Score:** $\frac{0.87+0.824}{2} = 0.847$

= 0.78
= 0.694
0.738

Compute Macro Precision, Recall, F1-score

Suppose we have 1000 samples, and the confusion matrix (with respect to the original labels) is given by:

	Predicted Positive	Predicted Negative
Actual Positive	TP = 940	FN = 10
Actual Negative	FP = 30	TN = 20

Macro Precision

Macro F1-score

Macro Recall

✓

Macro Precision = 0.83

Macro Recall

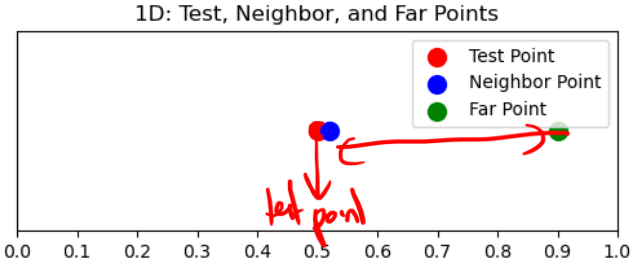
Macro F1-score

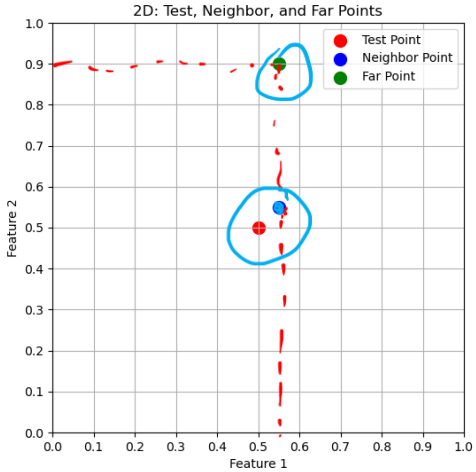
So far…

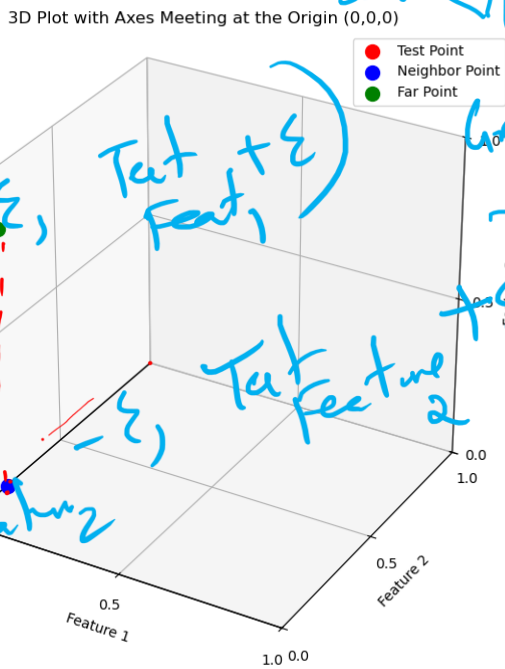
- Knowledge of the problem that you are trying to solve →
- Load the data correctly
- Analyze the data
  - How many classes?
  - Number of data instances per class
  - Train-test split → RandomSplit, TimeSplit
- Algorithm used for solving the problem
  - What are the hyperparameters of the algorithm?
  - Crossvalidation to find the best hyperparameters.
- Retrain the training data with the tuned hyperparameters.
- Used the trained model (with best hyperparameters obtained after crossvalidation) on test data to evaluate the performance of unseen test data. This should be done only once.

DT = {
KNN = {
SVM = {
NB = {
LR = {

Notion of closeness of points







In N dimensions

What is the notion of closeness in high dimension?

$X\_test = [f_1^{test}, f_2^{test}, f_3^{test}, f_4^{test}, \ldots, f_N^{test}]$

A point in X_train is close to X_test when

for all i from 1 to N,

$f_i^{train} \in [f_i^{test} - \epsilon, f_i^{test} + \epsilon]$

$i = 1$ to N

Assumptions

Setup:

- Let there be N independent features:

$$f_1, f_2, \ldots, f_N,$$

where each feature is uniformly distributed on the interval $[0, 1]$.

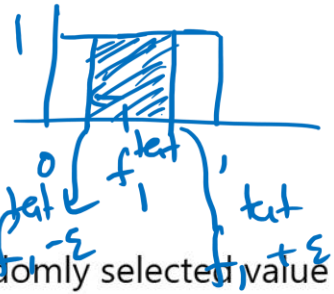
- Consider a fixed test point

$$X_{\text{test}} = (f_1, f_2, \ldots, f_N) \in [0, 1]^N.$$

- For a given small $\epsilon > 0$ (assumed to be small enough so that for every i the interval remains within $[0, 1]$), define for each dimension i the interval:

$$I_i = [f_i - \epsilon, f_i + \epsilon].$$

$$P\left(f_1^{train} \in \left(f_1^{test} - \epsilon, f_1^{test} + \epsilon\right)\right)$$



Probability in One Dimension:

Since each feature is uniformly distributed over $[0, 1]$, the probability that a randomly selected value for feature i falls within I_i is given by the length of the interval:

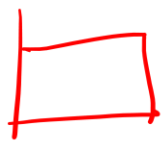
$$P\big(\text{feature } i \in I_i\big) = 2\epsilon.$$

This calculation assumes that $I_i \subset [0, 1]$.

$$\int_{f_1^{test} - \epsilon}^{f_1^{test} + \epsilon} 1 \, dx$$

$$\hookrightarrow \left[x\right]_{f_1^{test} - \epsilon}^{f_1^{test} + \epsilon}$$

$$f_1^{test} + \epsilon - \left(f_1^{test} - \epsilon\right) = 2\epsilon$$



$P(f_1, f_2 \cdots f_n) = P(f_1)P(f_2) \cdots P(f_n)$

$P(f_1^{tst} - \epsilon < f_1^{train}, f_1^{tst} + \epsilon) = 2\epsilon$

$P(f_2^{tst} - \epsilon < f_2^{train} < f_2^{tst} + \epsilon) = 2\epsilon$

$= (2\epsilon)^N$

Joint Probability (All Dimensions):

Assuming the features are independent, the probability that a randomly selected point

$$\mathbf{x} = (x_1, x_2, \ldots, x_N)$$

falls within the "close" region around X_{test} (i.e., $x_i \in I_i$ for every i) is the product of the individual probabilities:

$\epsilon < 0.5$

As $N \to \infty$

$(2\epsilon)^N = 0$

$$P\big(\mathbf{x} \text{ is close to } X_{\text{test}}\big) = \prod_{i=1}^{N} P\big(x_i \in I_i\big) = (2\epsilon)^N.$$

Curse of Dimensionality

The Curse of Dimensionality:

As the number of dimensions N increases, the probability

$$P(\text{close}) = (2\epsilon)^N$$

decreases exponentially. This exponential decay means that—even if each individual feature has a relatively high chance (i.e., 2ϵ) of being within ϵ of the corresponding value in X_{test}—the joint probability that *all* features are simultaneously close is extremely small.

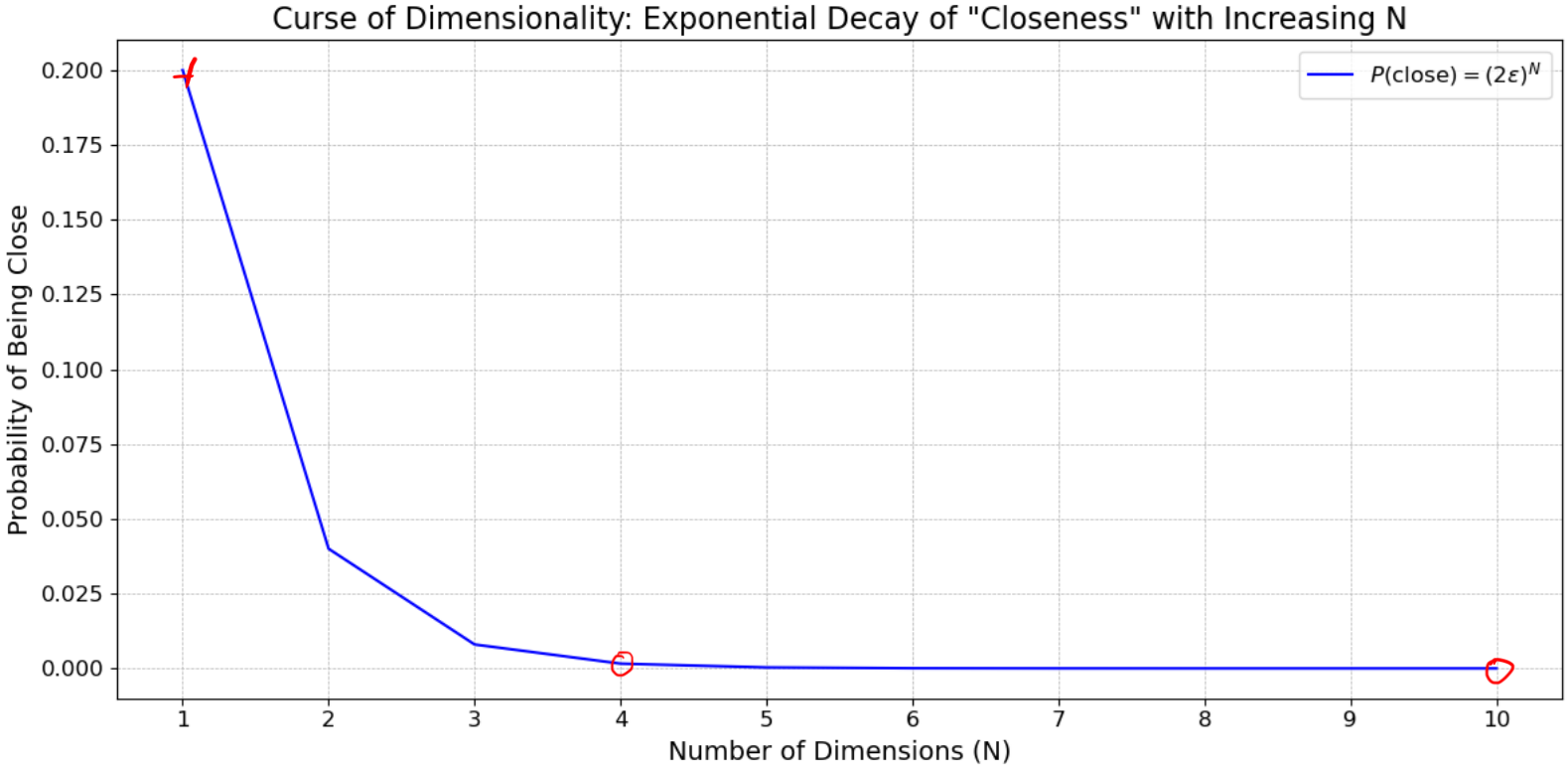
Example:

If $N = 10$ and $\epsilon = 0.1$, then:

$$P(\text{close}) = (2 \times 0.1)^{10} = (0.2)^{10} \approx 1.024 \times 10^{-7}.$$

This extremely small probability illustrates that in a 10-dimensional space, it is highly unlikely for a randomly chosen point to be close to X_{test} in every feature, which is a clear demonstration of the **curse of dimensionality**.

Curse of Dimensionality



Dimensionality Reduction Techniques

Principal Component Analysis

Singular Value Decomposition

t-Distributed Stochastic Neighbor Embedding (t-SNE)