# Introduction to Machine Learning

Praanesh Balakrishnan Nair

March 10, 2025

## Contents

# 1 Introduction

Quote by Herbert Alexander Simon:

> Learning is the process by which any system improves its performance from experience

# 2 Well-posed Learning Problem by Tom Mitchell (1998)

A computer program

- Performs Task T
- Has some Performance P
- Learns from Experience E

| Sl. No | Task | Performance | Experience |
|---|---|---|---|
| 1. | Classifying Emails as Spam/Not Spam | Number of emails correctly classified | Watching you Label Ema |
| 2. | Playing Chess | Percent of games won | Watch enemy play |
| 3. | Handwriting Recognition | Percent of correct recognitions | Sample images |

# 3 Levels/Scales of Measurement

This tells you how precise your data is

| | | **Nominal** | **Ordinal** | **Interval** | **Ratio** |
|---|---|---|---|---|---|
| 1 | Can be Categorized | ✓ | ✓ | ✓ | ✓ |
| 2 | Can be Ranked (Categories have order) | | ✓ | ✓ | ✓ |
| 3 | Categories are evenly seperated | | | ✓ | ✓ |
| 4 | Has Natural Zero (0 doesn't mean the absence of a variable) | | | | ✓ |
| | Example | Gender, Ethnicity | Level: Beginner, Intermediate, etc | Temperature: 0F, 1F, 2F, . | Height, Age, Temperature |

# 4 Components of a dataset

## 4.1 Features

- Individual measurable properties, which are going to be used as input to the machine learning model. Eg. Age of people, Dimensions of a house, etc

## 4.2 Data Points

- Multiple samples of features.
- Eg:

| Sl. No | Age | Height | Weight | BP |
|---|---|---|---|---|
| 1 | 19 | 175 | 68 | 999 |
| 2 | 25 | 169 | 69 | 0 |
| ... | ... | ... | ... | ... |

Each of these rows are data points. In each data point, you have different samples of the same features

### 4.3 Feature Vector

- Features in one data-point is often mathematically represented as a Vector
- Eg:

| Sl. No | Age | Height | Weight | BP | Feature Vector |
|---|---|---|---|---|---|
| 1 | 19 | 175 | 68 | 999 | [19, 175, 68] |
| 2 | 25 | 169 | 69 | 0 | [25, 169, 0] |
| ... | ... | ... | ... | ... | ... |

### 4.4 Central Tendancies

1. Mean is the average of the data and is given by $\sum_{i=1}^{n} \frac{x_i}{n}$
2. Median is the middle-most element and is used when the data is too spread apart
3. Mode is the most frequent element occuring. This is the only thing you can use for nominal and ordinal data.

### 4.5 Distance Matrix

- Basically Adjacency Matrix
- $d(i, j)$ = distance between $i^{th}$ data point and $j^{th}$ data point.
- It's Symmetric
- Diagonal Elements are 0
- The actual distance between $x^{th}$ data point and $y^{th}$ data point can be measured in many ways:
  1. Euclidean Distance = $\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$
     where $n$ is the number of dimensions
     - Eg. For 3D, it would be $\sqrt{(x_{\hat{x}} - y_{\hat{x}})^2 + (x_{\hat{y}} - y_{\hat{y}})^2 + (x_{\hat{z}} - y_{\hat{z}})^2}$
  2. Manhattan distance = $\sum_{i=1}^{n} |x_i - y_i|$
     - It's less sensitive to outliers (extreme data)
     - The right distance to use in grids where you can only move horizontally and vertically (manhattan distance sums up the horizontal and vertical distance, telling you the actual distance you'd cover while moving in a grid)
  3. Minowski Distance = $(\sum_{i=1}^{n} |x_i - y_i|^p)^{\frac{1}{p}}$
     - It's a generalization of manhattan and euclidian distance. You have to choose the right value of $k$ for it
     - It's basically $L_p$ norm
  4. Cosine Distance = $1 - \frac{x.y}{|x||y|}$
     - Prequisite Knowledge:

$$x.y = |x||y| \cos(\theta)$$

$$\cos(\theta) = \frac{x.y}{|x||y|} = \text{Cosine Similarity}$$

     - So the cosine of the angle between two vectors is called cosine similarity.

- $\frac{x.y}{|x||y|}$ is the cosine similarity between two vectors, and hence $1 - \frac{x.y}{|x||y|}$ is called the cosine distance between them.
- If cosine similarity is 0, it means they are independent of each other ($\cos(\theta) = 0 \Rightarrow \theta = 90$)
- If $\theta = 0 \Rightarrow \cos(\theta) = 1$ and this means that the two vectors are dependent. They point in the same direction.
- Example: Find Cosine Similarity for the vectors $[12, 5, 9]$ and $[3, 11, 7]$. Find if the two vectors are independent of each other.
  - $\cos(\theta) = \frac{(12*3)+(5*11)+(9*7)}{\sqrt{12^2+5^2+9^2}\sqrt{3^2+11^2+7^2}}$
  - $\cos(\theta) \approx 0.8$
- Used for high dimensional data

5. Chebyshev Distance = $max_{i=1}^{n}|x_{\hat{i}} - y_{\hat{i}}| = L_\infty$Norm
   - Useful only when the maximum difference between coordinates matter
6. Hamming Distance = $\sum_{i=1}^{n} 1(x_i \neq y_i)$
   - Number of positions where 2 binary coordinates are not the same
7. Mahalanobis Distance = $\sqrt{(x-y)^T S^{-1}(x-y)}$ where $S$ is the covariance matrix

### 4.6 One-Hot Encoding

- Say we have 5 classes: A, B, C, D, E.
- We encode classes as: 10000, 01000, 00100, 00010, 00001

## 5 Types of Learning

### 5.1 Supervised Learning

- You're given with features $X_i$ the label $Y_i$ associated with each of them.

#### 5.1.1 Classification

- You need to classify features $X_i$ into classes/labels $Y_i$
- In other words, the output is a **qualitative/ categorical response**.
- You find the pattern of data that is associated with one label, and use that pattern to classify.

#### 5.1.2 Regression

- Here, you input some data and you get a quantitative response.

### 5.2 Unsupervised Learning

- You have only features $X_i$ and no labels
- You find patterns, so that similar patterns form one label, and anything different will be given another label.

### 5.2.1 Clustering

## 5.3 Semi-supervised Learning

The entire dataset consists of labelled and unlabelled data

1. Perform supervised learning on the labelled data
2. Now you use this to predict the labels of the unlabelled data. The predicted labels are called **psuedo-labels**.
3. Now do supervised learning on the combined data

# 6 Classification

## 6.1 Model Accuracy

### 6.1.1 Error Rate

- In classification, the model accuracy is quantified by the **error rate**.
- **Error Rate** = $\frac{\textbf{number of misclassifications}}{\textbf{total number of data points}}$
- **Error Rate** = $\frac{\sum_{i=1}^{n} I(y_i \neq \hat{y_i})}{n}$, where $I(y_i \neq \hat{y_i})$ is 1 if it's a mismatch, and 0 if it's a match

### 6.1.2 Confusion Matrix

- Matrix where:
  - rows signify Ground truth (Row1: +, Row2: -)
  - columns signify predicted output (Column1: +, Column2: -)
- If row and column have same sign, it means the model has predicted correctly (it's a **true output**).
- If row and column have opposite signs, it means the model has predicted incorrectly (it's a **false output**).

| | Predicted + | Predicted - |
|---|---|---|
| Actual + | **True Positive** | **False Negative** |
| Actual - | **False Positive** | **True Negative** |

- Here are things you can derive from the confusion matrix:

| | Predicted + | Predicted - | |
|---|---|---|---|
| Actual + | **True Positive** | **False Negative** | Sensitivity/Recall |
| Actual - | **False Positive** | **True Negative** | Specificity |
| | Precision | Negative Predictive Value | Accuracy |

  - **Sensitivity / Recall** = $\frac{\textbf{Diag. Element of Row 0}}{\textbf{Row 0}} = \frac{TP}{TP+FN}$
  - **Specificity** = $\frac{\textbf{Diag. Element of Row 1}}{\textbf{Row 1}} = \frac{TN}{FP+TN}$
  - **Precision** = $\frac{\textbf{Diag. Element of Column 0}}{\textbf{Column 0}} = \frac{TP}{TP+FP}$
  - **Negative Predictive Value** = $\frac{\textbf{Diag. Element of Column 1}}{\textbf{Column 1}} = \frac{TN}{FN+TN}$

– **Accuracy** $= \frac{\textbf{Diagonal Elements}}{\textbf{All Elements}} = \frac{TP+TN}{TP+FN+FP+TN}$
\ Here are some other formulae that can be derived:

$$\mathrm{F}_\beta = \frac{(1+\beta^2) \times \text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}}$$

- The more the value of $\beta$, the more emphasis recall gets (hence, more true positives captured).

$$R^2 = \frac{MSE}{variance}$$

#### 6.1.3 Receiver Operating Characteristic Curve (ROC) and Area Under the Curve (AUC)

- ROC is the plot between True Positive and False Positive
- AUC is the area under ROC
- $0 \leq AUC \leq 1$
- $AUC = \int_0^1 (\textbf{ROC Curve})$

### 6.2 Model Validation Techniques

#### 6.2.1 Internal Validation

- Seperation betwen clusters should be high
- Cohesion (distance between points in a cluster) should be low

#### 6.2.2 External Validation

1. Dice Coefficient
   - $D(A,B) = \frac{2|A \cap B|}{|A|+|B|}$
   - If $D(A,B) = 0$, then there's no overlap. Similarly if $D(A,B) = 1$, they are the same set.
   - $A$ could be the data we have and $B$ could be some external data.
   - Here you equally check for the presence and absence of elements
2. Simple Matching Coefficient
   - Say you have two binary numbers
   - Simple Matching Coefficient $= \frac{P_{00}+P_{11}}{P_{00}+P_{01}+P_{10}+P_{11}}$, where:
     - $P_{00}$ is the number of instances where bit $i$ of both the numbers is 0
     - $P_{11}$ is the number of instances where bit $i$ of both the numbers is 1
     - $P_{10}$ is the number of instances where bit $i$ of the first number is 1 and the second number is 0
     - $P_{01}$ is the number of instances where bit $i$ of the first number is 0 and the second number is 1
   - You use this when you're interested in both presence (1) and absence (0) of binary data
3. Jaccard Similarity Index
   - $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$
   - $J(A,B) = \frac{P_{00}}{P_{11}+P_{10}+P_{01}}$
   - Used for clustering or text mining, where the absence of data doesn't matter
   - $|A \cup B|$ doesn't contain $P_{00}$ because it's like null set and does not matter to union

# 7 Regression

## 7.1 Model Accuracy

- In regression, it's called the **quality of fit** and it's the quantification of the degree of closeness of predicted response and the true response
- The most commonly used measure for this, is the **Mean Square Error (MSE)**.
- $MSE = \frac{\sum_{i=1}^{n}(y_i - f(x_i))^2}{n}$, where $y_i$ is the true value, $f(x_i)$ is the predicted value
- Accuracy of the Model $\propto \frac{1}{MSE}$

## 7.2 Types

### 7.2.1 Linear Regression

1. What it is
   - Given input $X_i$, predict a quantitative response $Y_i$.
   - You find the line closest to all of the data points.
   - The line is given as: $h_\theta(x) = \theta_0 + \theta_1(x)$
2. How it works
   - You have a cost function given as $J(\theta_0, \theta_1) =$ **Mean Square Error**
   - Simply minimize the cost function i.e. find values for $\theta_0$ and $\theta_1$ such that $J(\theta_0, \theta_1)$ has the smallest value.
   - To find those values, you either run a **really large loop** and iterate through all the values of $\theta_1$ and $\theta_0$ possible, or you use something called the **gradient descent**.
3. Gradient Descent
   - $\theta_i = \theta_i - \alpha\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_i}$, where $i = 0$ **or** 1, and $\alpha$ is the learning rate (user defined)
   - In every iteration, you update each parameter by subtracting $\alpha\frac{\partial \theta_i}{\partial J(\theta_0,\theta_1)}$
   - $\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_i}$ is the change in $J(\theta_0, \theta_1)$ with respect to $\theta_i$ i.e. how much $J(\theta_0, \theta_1)$ changes for a small change in $\theta_i$.
   - This change tells you the direction you have to go in the graph, to reduce the cost function. The direction is simply a positive or negative value which should be added to each of the parameters.
   - On performing multiple iterations of this method, you finally reach the minimum of this function.
   - When you have two or more parameters like this, you shouldn't directly change $\theta_i$, because . Instead, you do:
     - $temp_0 = \theta_0 - \alpha\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0}$
     - $temp_1 = \theta_1 - \alpha\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1}$
     - $\theta_0 = temp_0$
     - $\theta_1 = temp_1$
4. An extension to this: Multivariate Linear Regression

- So far, we've had one input/variable $x$, and the line was in a 2D plane.
- Now, we have multiple inputs/variables, and hence the line is in a multidimensional space.
- The line is given as:
  $h_0(x) = \theta_0 + \theta 1 x_1 + \theta_2 x_2 + \theta_3 x_3 + ... + \theta_n x_n$

  $h_0 = \begin{bmatrix} \theta_0 & \theta_1 & \theta_2 ... \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x3 \\ ... \end{bmatrix}$

  $h_0 = \theta^T X$

#### 7.2.2 Polynomial Regression

- $h_0(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2$

#### 7.2.3 Logistic Regression

- $0 \leq h_\theta(x) \leq 1$
- $h_\theta = g(\theta^T X)$, where $g(X)$ is called the **sigmoid function** and is given as:

$$g(x) = \frac{1}{1 + e^{-x}}$$

- so $h(\theta) = \frac{1}{1+e^{-\theta^T X}}$

#### 7.2.4 Least Square Method

- Given $h_\theta(x) = \theta_0 + \theta_1 x$

$$\theta_1 = \frac{\sum((x - \bar{x}) * (y - \bar{y}))}{\sum(x - \bar{x})^2}$$

$$\theta_0 = h_\theta(x) - \theta_1 x$$

### 7.3 Performance Measures

#### 7.3.1 Mean Squared Error:

$MSE = \frac{\sum_{i=1}^{n}(y_i - f(x_i))^2}{n}$

#### 7.3.2 Mean Absolute Error:

$MAE = \frac{\sum_{i=1}^{n}|(y_i - f(x_i))|}{n}$

#### 7.3.3 Root Mean Squared Error:

$RMSE = \sqrt{MSE}$

#### 7.3.4 $R^2$ Score

$R^2 = 1 - \frac{SS_{Residuals}}{SS_{Total}}$

# 8 Bias, Variance and Regularization

## 8.1 How to pick the right model

### 8.1.1 By testing different polynomial degrees

- $h_0(x) = \theta_0 + \theta_1 x_1$
- $h_0(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2$
- $h_0(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3$
- $h_0(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3 + \theta_4 x_4^4$
- $h_0(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3 + \theta_4 x_4^4 + \theta_5 x_5^5$
- $h_0(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3 + \theta_4 x_4^4 + \theta_5 x_5^5 + ...\theta_n x_n^n$

### 8.1.2 Validation Dataset

1. Two way split Split the data in the ratio $m : n$. The $m$ part will be used for training and the $n$ part will be used for testing.
2. Three way split Split the data in the ratio $p : q : r$, where $p$ is for **training**, $q$ is for **cross-validation**, and $r$ is for testing.

## 8.2 Regularization

- Update the cost function as:

$$J(\theta) = MSE + \frac{\lambda}{n} \sum_{i=1}^{n} \theta_j^2$$

# 9 Nearest Neighbour

## 9.1 KNN

- If k too small: Overfitting

# 10 Decision Tree

## 10.1 Example Problem

A company wants to predict whether a customer will buy a product or not ("yes" or "no"), based on a feature called "Discount Offered", which has two categories: "High Discount" and "Low Discount". The dataset contains 10 samples with the class distribution, as shown in the table

| Customer | Discount Offered | Buy Product? |
|---|---|---|
| 1 | high | yes |
| 2 | low | no |
| 3 | high | yes |
| 4 | high | yes |
| 5 | low | no |
| 6 | high | yes |
| 7 | low | no |
| 8 | high | yes |
| 9 | high | no |
| 10 | low | no |

1. Compute the entropy of the dataset before any split

$$\text{Entropy} = -\sum_{i=1} p_i log_2(p_i)$$

where $p_1$ = Probability of "yes" and $p_2$ = Probability of "no"

$$\text{Entropy} = -\frac{5}{10} log_2(\frac{5}{10}) - \frac{5}{10} log_2(\frac{5}{10})$$

$$\text{Entropy} = -log_2(\frac{1}{2})$$

$$\text{Entropy} = log_2(2)$$

$$\text{Entropy} = 1$$

2. df

# 11 Normalization

It's where you bring the data into a custom range

## 11.1 Min-max

$$\frac{x_i - x_{min}}{x_{max} - x_{min}} * (y_{max} - y_{min})$$

- Uniform distribution
- Scales data to a fixed range

## 11.2 Z-Score

$$\frac{x_i - \mu}{\sigma}$$

- Used when outliers exist
- Normal distribution
- Z score transforms data to form 0 mean and unit variance