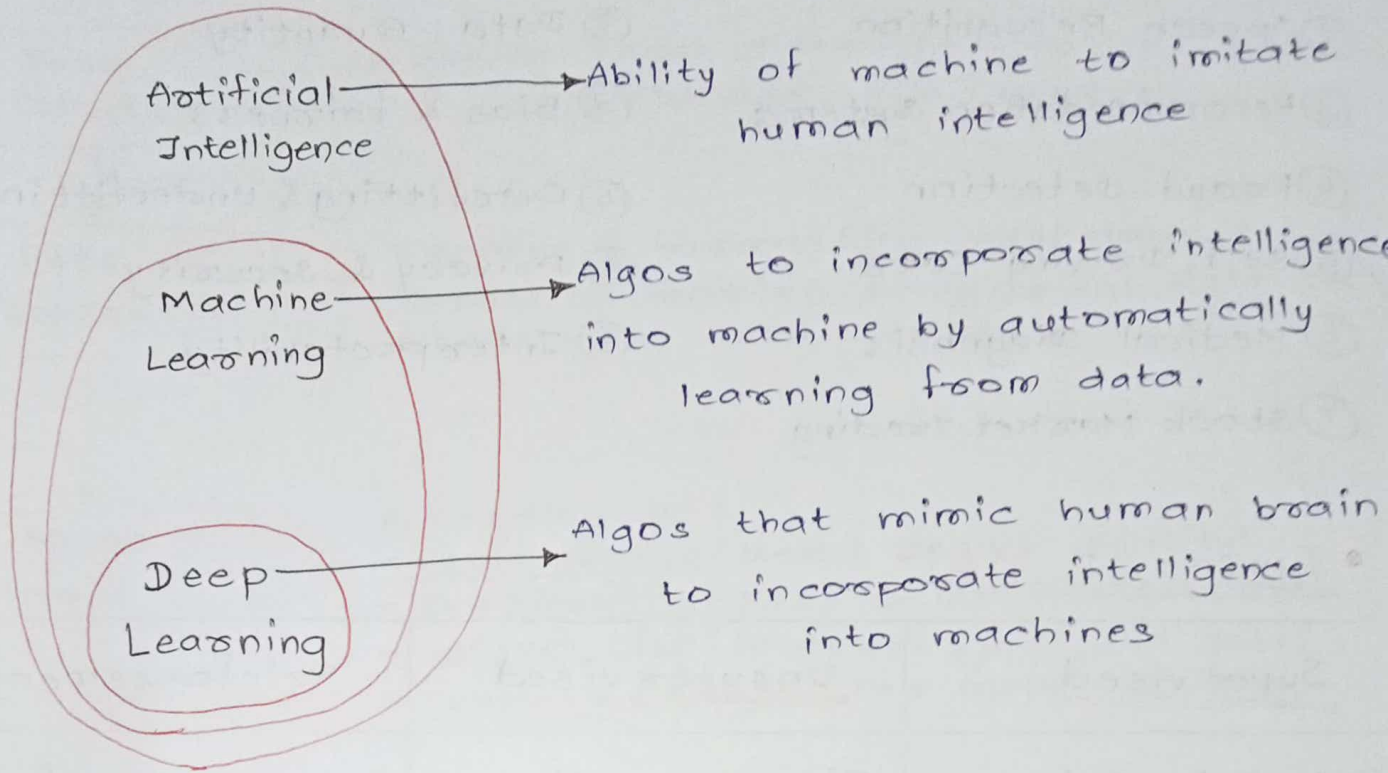


# \* ML Fundamentals \*



● Artificial Intelligence: study of how to train computers, so that computers can do things which humans can do.

Artificial + Intelligence

(Human/non-natural made thing.) (Ability to understand/ think)

● Machine Learning (ML): Enable computers to learn automatically from past data, it can improve its performance by gaining more data.

↳ ML uses algos for building math models & make predictions using historical data/info.

## • Applications of ML:

- ① Image Recognition
- ② Speech Recognition
- ③ Recommendation Systems
- ④ Fraud Detection
- ⑤ Self Driving cars
- ⑥ Medical Diagnosis
- ⑦ Stock Market Trading

## • Limitations of ML:

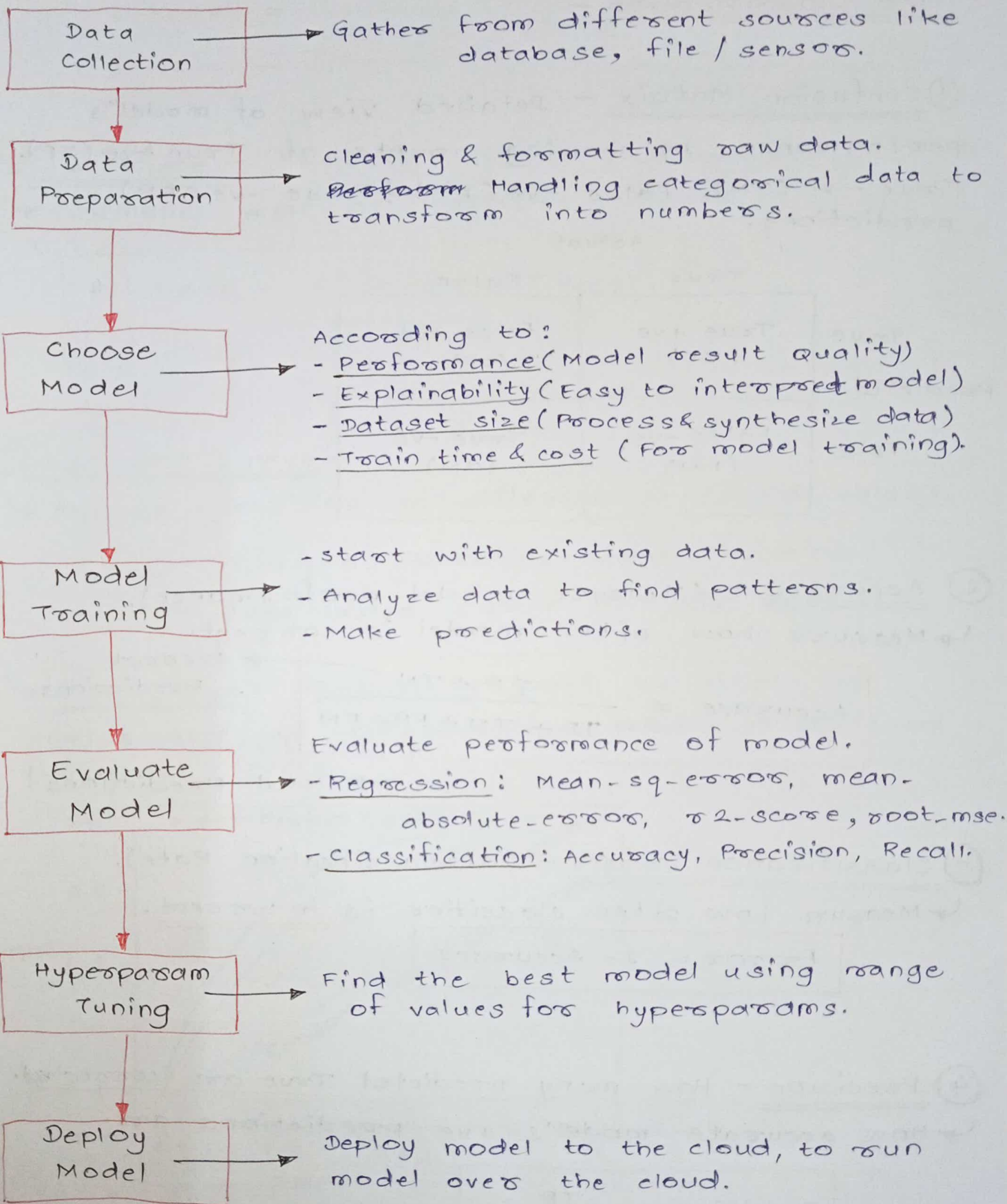
- ① Data Quality
- ② Data Quantity
- ③ Bias & fairness
- ④ Overfitting & Underfitting
- ⑤ Privacy & security
- ⑥ Interpretability

## • Types of ML:-

<u>Supervised</u>	<u>Unsupervised</u>	<u>Reinforcement</u>
Calculate Outcome.	Discovers patterns.	Learn series of action.
Learn pattern bet <sup>n</sup> i/p & their labels.	Find pattern in i/p & divide in classes.	Find best reward for actions.
Build & train, then test model.	Build & train, then test model.	Train & test model simultaneously.
Classification & Regression.	Clustering & Association	Exploration & Exploitation.
• <u>Algos -</u> ① Linear Regress <sup>n</sup> ② Decision Tree ③ KNN ⑤ SVM ④ Random Forest.	• <u>Algos -</u> ① K-mean Clustering ② Agglomerative	• <u>Algos -</u> ① Q-Learning ② SARSA ③ Deep Q network
• <u>Examples -</u> ① Image Detection ② Population Growth Prediction.	• <u>Examples -</u> ① Dust Segmentation. ② Feature Elicitation. ③ Target Market.	• <u>Examples -</u> ① Driverless car. ② Self-navigating cleaners.



## • Machine Learning workflow -



## • Model Evaluation Techniques in ML:

### • For Classification -

① Confusion Matrix - Detailed view of model's performance by showing counts of True +ve (TP), True -ve (TN), False +ve (FP) & False -ve (FN) predictions.

Actual

True

False

True

True +ve  
(TP)

False +ve  
(FP)

Predicted

False

False -ve  
(FN)

True -ve  
(TN)

② Accuracy - (Measure model performance).

↳ Measure how often model is correct.

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

→ correct  
Predictions

→ Total Predictions

③ Classification Error - (Misclassification Rate).

↳ Measure how often classifier is incorrect.

$$\text{Error} = 1 - \text{Accuracy}$$

④ Precision - How many predicted True are correct.

↳ How accurate model's +ve predictions are.

$$\text{Precision} = \frac{TP}{TP + FP}$$



⑤ Recall — How many actual True correctly predicted.

$$\text{Recall} = \frac{TP}{TP + FN}$$

⑥ F1-Score — Evaluate overall performance of model.

↳ Harmonic mean of Precision & Recall.

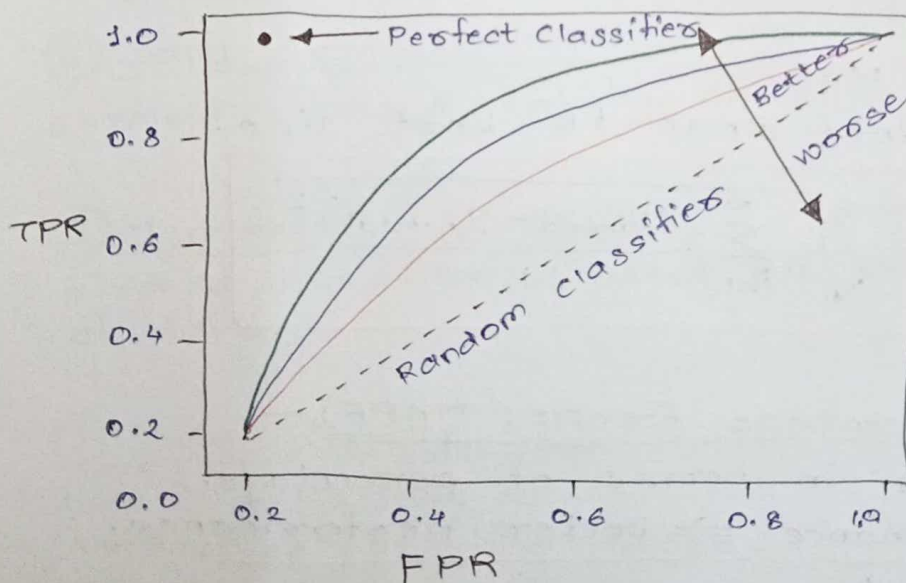
$$\text{F1-Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

⑦ AUC-ROC Curve —

↳ Analyze classifiers at different threshold values.

- ROC — Receiver Operating characteristics
- AUC — Area under curve of ROC Plot.

↳ ROC Curve — Probability graph to show the performance of classification model at different threshold levels.



$$\bullet \text{ FPR (False +ve Rate)} = \frac{FP}{FP + FN}$$

$$\bullet \text{ TPR (True +ve Rate)} = \frac{TP}{TP + TN}$$

## • For Regression -

### ① Mean Absolute Error -

↳ Analyze loss over whole dataset.

↳ Error: Difference bet<sup>n</sup> predicted & actual values.

$$MAE = \sum_{i=1}^N \left| \frac{y_{pred} - y_{actual}}{N} \right|$$

### ② Mean Squared Error - Most commonly used.

↳ Used to calculate loss.

↳ Find error & square it, then find average over the whole dataset.

↳ Always +ve, bcz we square values.

↳ Small MSE, better performance.

$$MSE = \sum_{i=1}^N \frac{(y_{pred} - y_{actual})^2}{N}$$

### ③ Root Mean Squared Error - Evaluate performance.

↳ Indicates how much data points spread around the best line.

↳ Std deviation of MSE.

↳ Lower MSE - Points close to best line.

$$RMSE = \sqrt{MSE} = \sqrt{\sum_{i=1}^N \frac{(y_{pred} - y_{actual})^2}{N}}$$

### ④ Mean Absolute Percentage Error (MAPE) -

Express errors in terms of percentage.

Smaller performance  $\Rightarrow$  better performance.

$$MAPE = \frac{\sum_{i=1}^N \left( \frac{|y_{pred} - y_{actual}|}{y_{actual}} \right)}{N} * 100\%$$



## ① Exploratory Data Analysis (EDA):-

↳ Methods to study & explore record sets to apprehend their predominant traits, discovers patterns, locate outliers, & identify relationships between variables.

### • Goals of EDA -

① Data cleaning: Handle missing values, duplicates, outliers & handle categorical data.

② Data visualization: Visual techniques represent statistics graphically. Histograms, box plots, scatter plots, line plots, heat maps & bar charts to identify styles, trends & relationships within facts.

③ Feature Engineering: contain scaling, normalization, binning, encoding variables.

④ Correlation & Relationships: Allow discover relationships & dependencies bet<sup>n</sup> variables. correlation analysis, scatter-plots & pass-tabulations.

⑤ Data segmentation - Divide info into significant segments based totally on sure standards / traits.

⑥ Hypothesis Generation - Generating hypotheses / studies questions based on preliminary exploration of data.

⑦ Data Quality Assessment - Permits assessing nice & reliability of the info.

↳ Involve checking record integrity, consistency & accuracy to make info suitable for analysis.

## • EDA Implementation using Python:

### • Get Info about Dataset -

```
>>> df.shape
```

O/p → (rows, cols)

```
>>> df.describe()
```

O/p → Give results of basic statistical computations on the numeric columns.

```
>>> df.info()
```

O/p → Information (feature name, no-null counts, dtype) of dataframe.

### • Changing Dtype (Object → Datetime) -

```
>>> df['Date'] = pd.to_datetime(df['Date'])
```

### • Unique elements in dataset -

```
>>> df.nunique()
```

O/p → No. of unique elements in each column.

## • Handling Missing Values -

### • isnull() - Check any missing values in dataset.

```
>>> df.isnull().sum()
```

O/p → Return sum of np.nan(NULL) values in each column.



- fillna() - Fill value at ~~na~~ NULL places.

```
>>> df['column'].fillna(value, inplace=True)
```

- replace() - Used to replace values in dataset.

```
>>> df['col'] = df['col'].replace(np.nan, value)
```

- dropna() - Drop records with Null values.

```
>>> df['col'].dropna(axis=0, how='any')
```

- uplicated() - Check if duplicates present in dataset.

```
>>> df.duplicated()
```

O/p → Return Boolean (False → Original, True → Duplicate) for each row in Dataset.

```
>>> df.duplicated().sum()
```

O/p → Return total number of duplicate rows.

- drop\_duplicates - Drop the duplicate rows.

```
>>> df.drop_duplicates(keep='first', inplace=True)
```

O/p → keep first copy & remove all other duplicates.

## ① Check count of values in each column -

- value\_counts() - Return count of each unique value in the column.

```
>>> df['col'].value_counts()
```

## ① Data Encoding -

↳ Encode categorical data into numerical values.

↳ One-hot Encoding / Label Encoding.

```
>>> from sklearn.preprocessing import LabelEncoder
```

```
>>> encoder = LabelEncoder()
```

```
>>> df['col'] = encoder.fit_transform(df['col'])
```

O/p → Assign num to each category, starting from 0.

## ② Data Visualization :-

↳ Analyze data in the form of graphs/maps, easy understand trends/patterns.

① Histogram - count of numeric values present in regular intervals.

```
>>> sns.histplot(x='col', data=df)
```

② Boxplot -

```
>>> sns.boxplot(x='col1', y='col2', data=df)
```

③ pairplot() - Pairwise distributions in a dataset.

```
>>> sns.pairplot(df, hue='col', height=2)
```



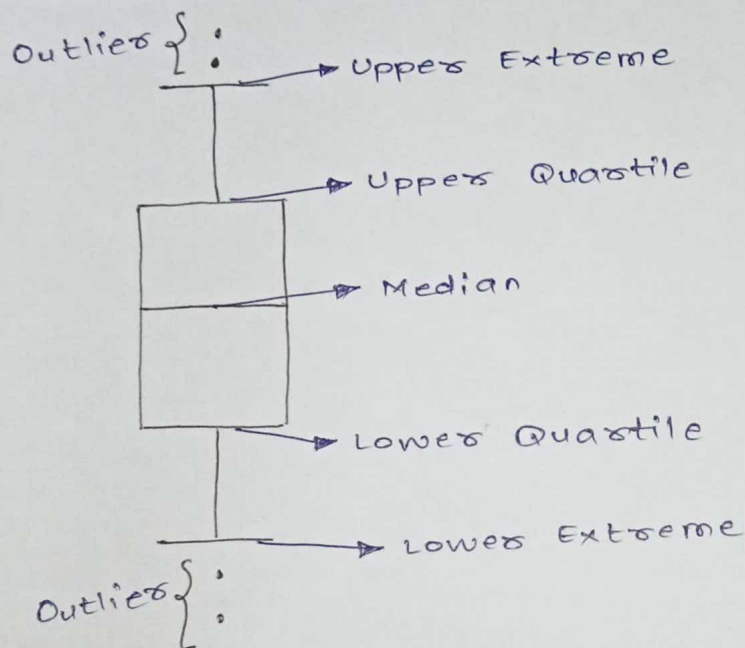
## ① Handling Outliers:-

- ↳ Outlier - Out or range of normal objects.
- ↳ Outlier detection analysis is called outlier mining.
- ↳ Many ways to detect, but same way for removal.

### • Outlier Detection:

```
>>> sns.boxplot(x = 'col 1', data = df)
```

### • Removing Outliers:



### • IQR (Inter Quartile Range):

- ↳ Most commonly used technique.
- This is Outlier base value.

$$\text{IQR} = \text{Quartile 3} - \text{Quartile 1}$$

$$Q1 = \text{np.percentile}(df['col'], 25, \text{method} = 'midpoint')$$

$$Q3 = \text{np.percentile}(df['col'], 75, \text{method} = 'midpoint')$$

$$\text{IQR} = Q3 - Q1$$

### • Upper & Lower bound:

$$\text{upper} = Q3 + 1.5 * \text{IQR}$$

$$\text{lower} = Q1 - 1.5 * \text{IQR}$$

Then, use upper & lower value to remove outliers.

```
upper_val = np.where(df['col'] >= upper)
```

```
lower_val = np.where(df['col'] <= lower)
```

```
# Remove Outliers
```

```
df.drop(upper_val[0], inplace = True)
```

```
df.drop(lower_val[0], inplace = True)
```