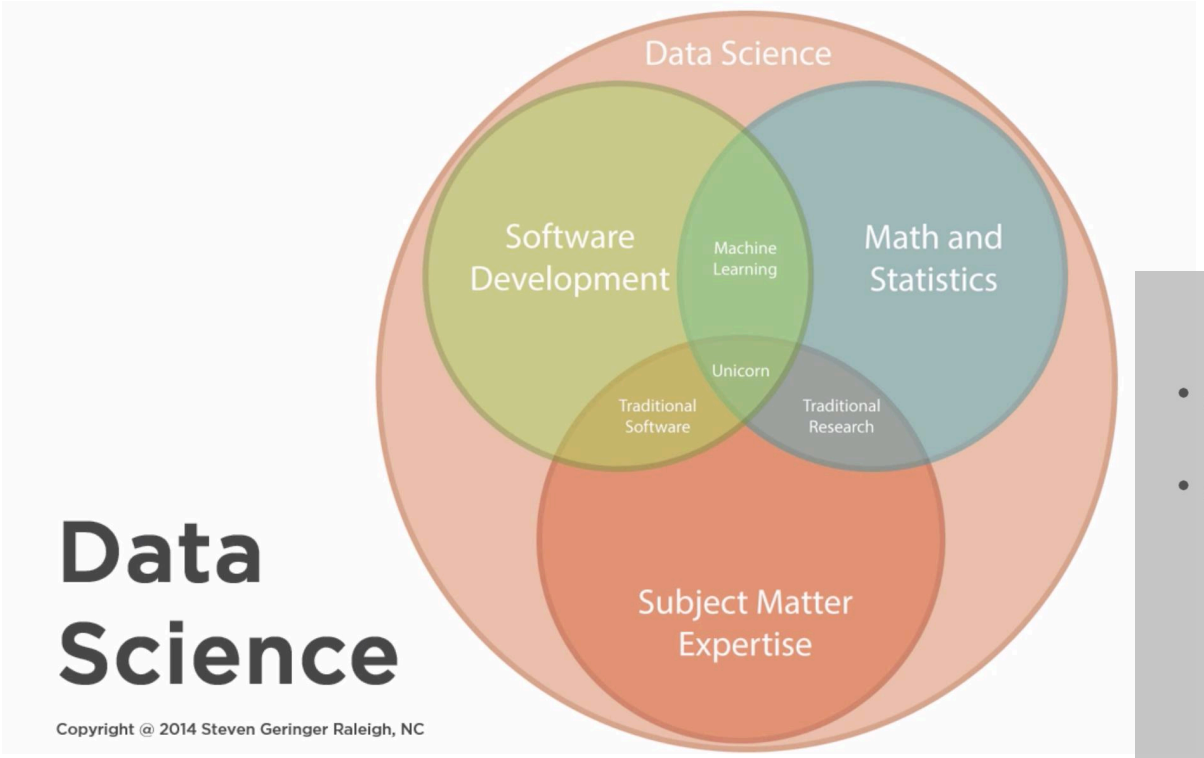
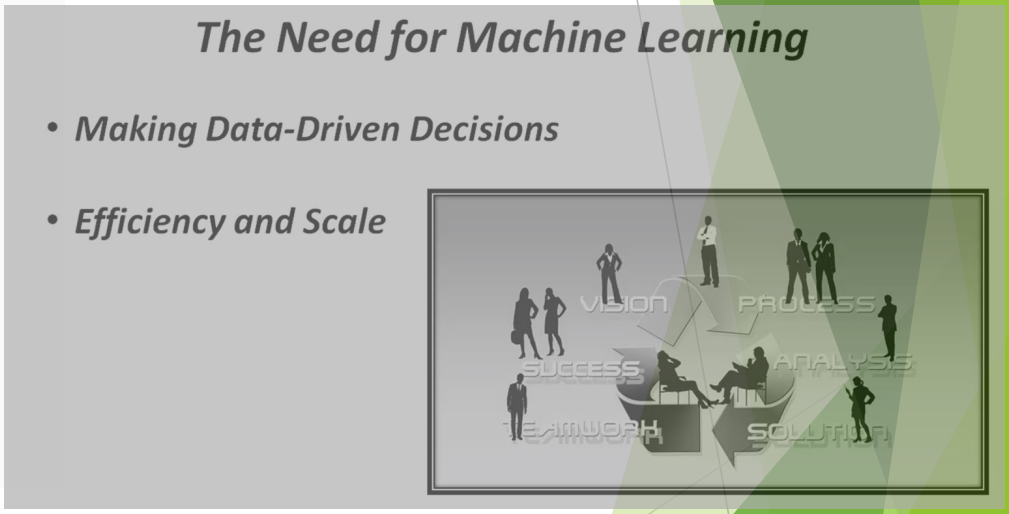
AIML - Data pre-processing and building a prediction model using AIML

Murali Krishna

Basics of Machine Learning



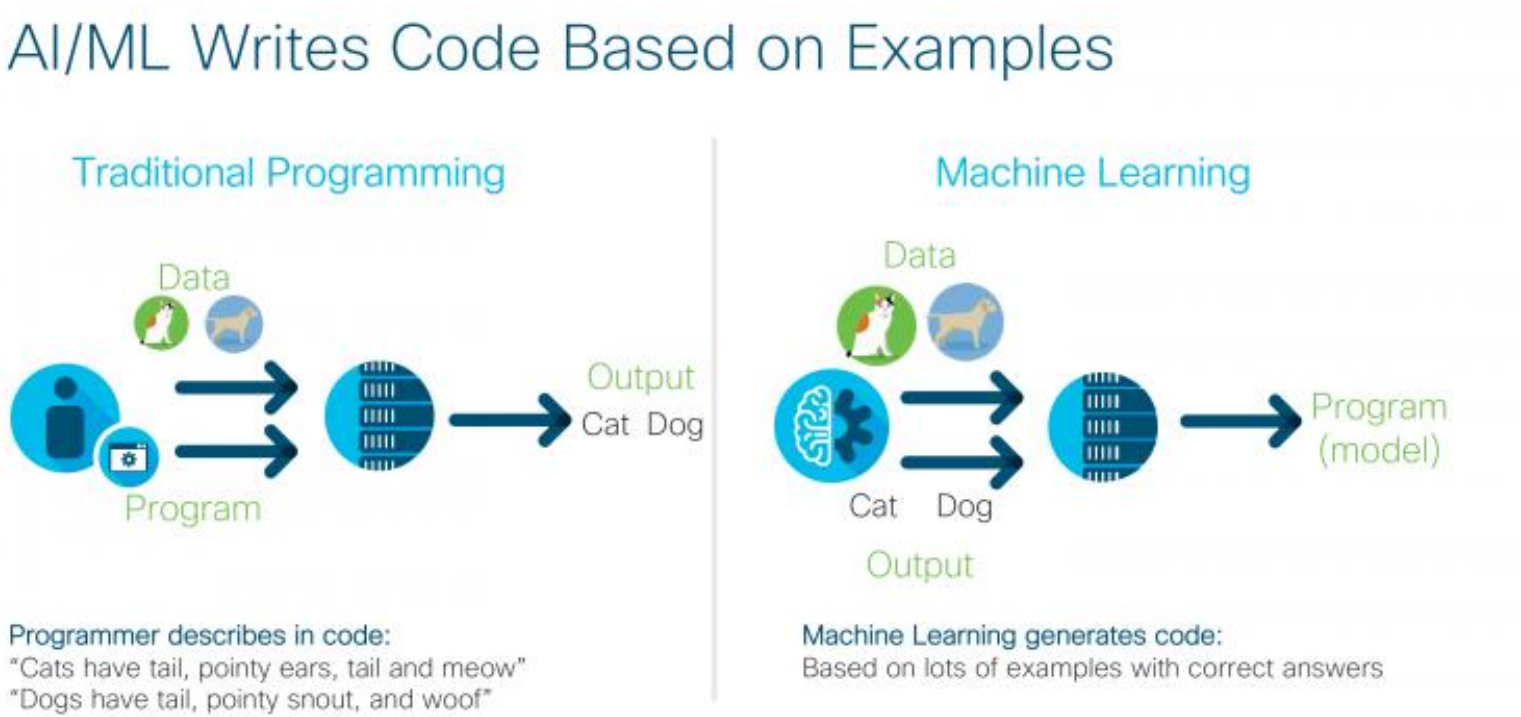


Facts :-

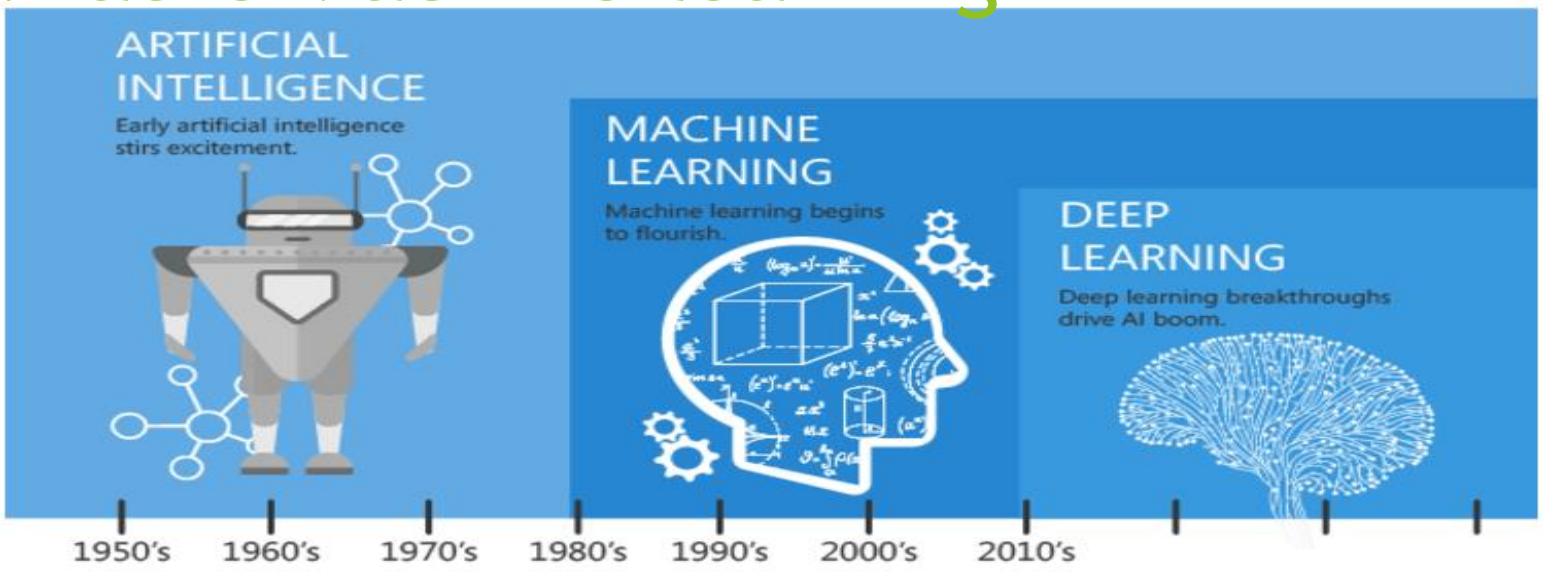
1952: Arthur Samuel, who was the pioneer of machine learning, created a program that helped an IBM computer to play a checkers game. It performed better more it played.

1959: In 1959, the term "Machine Learning" was first coined by **Arthur Samuel**.

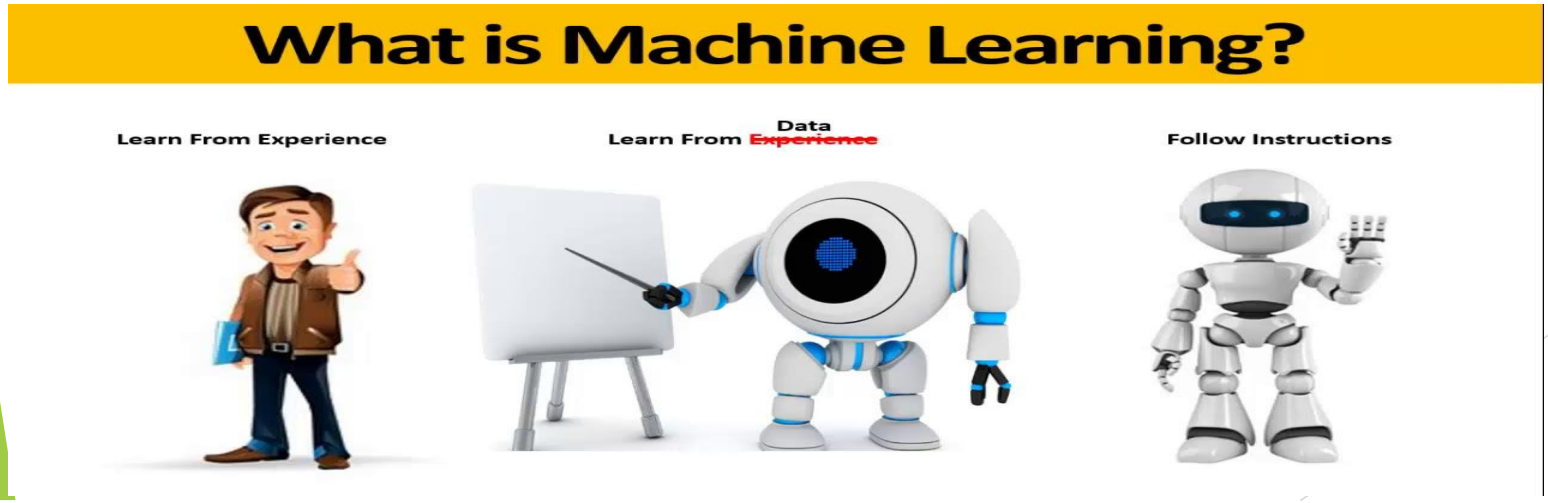
Basics of Machine Learning



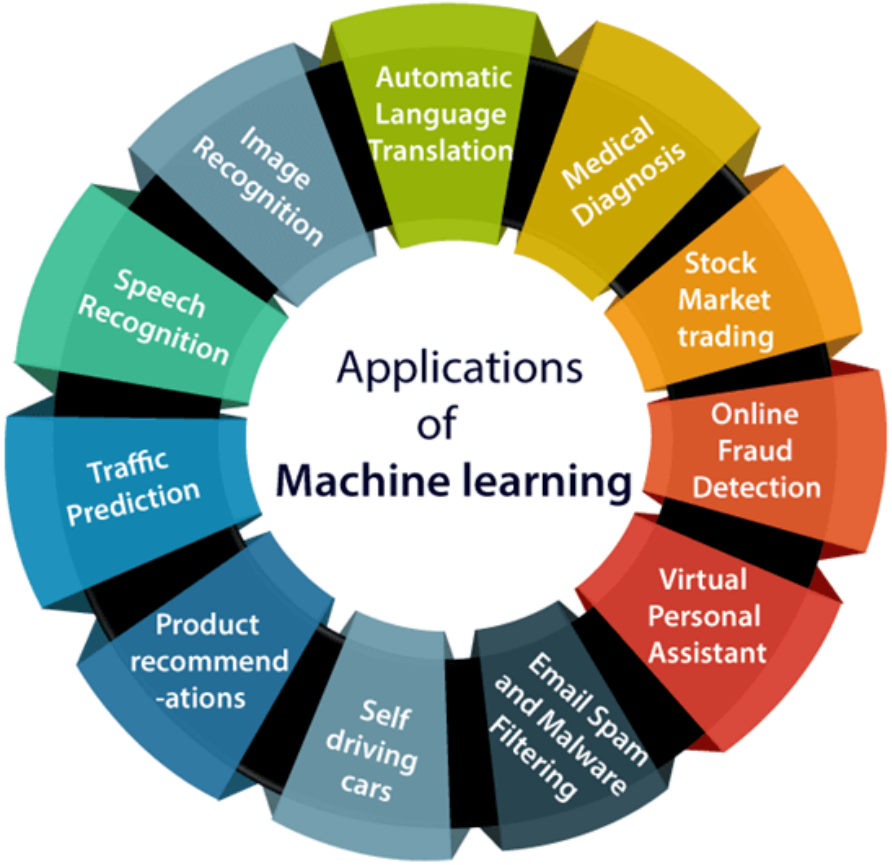
What is Machine learning



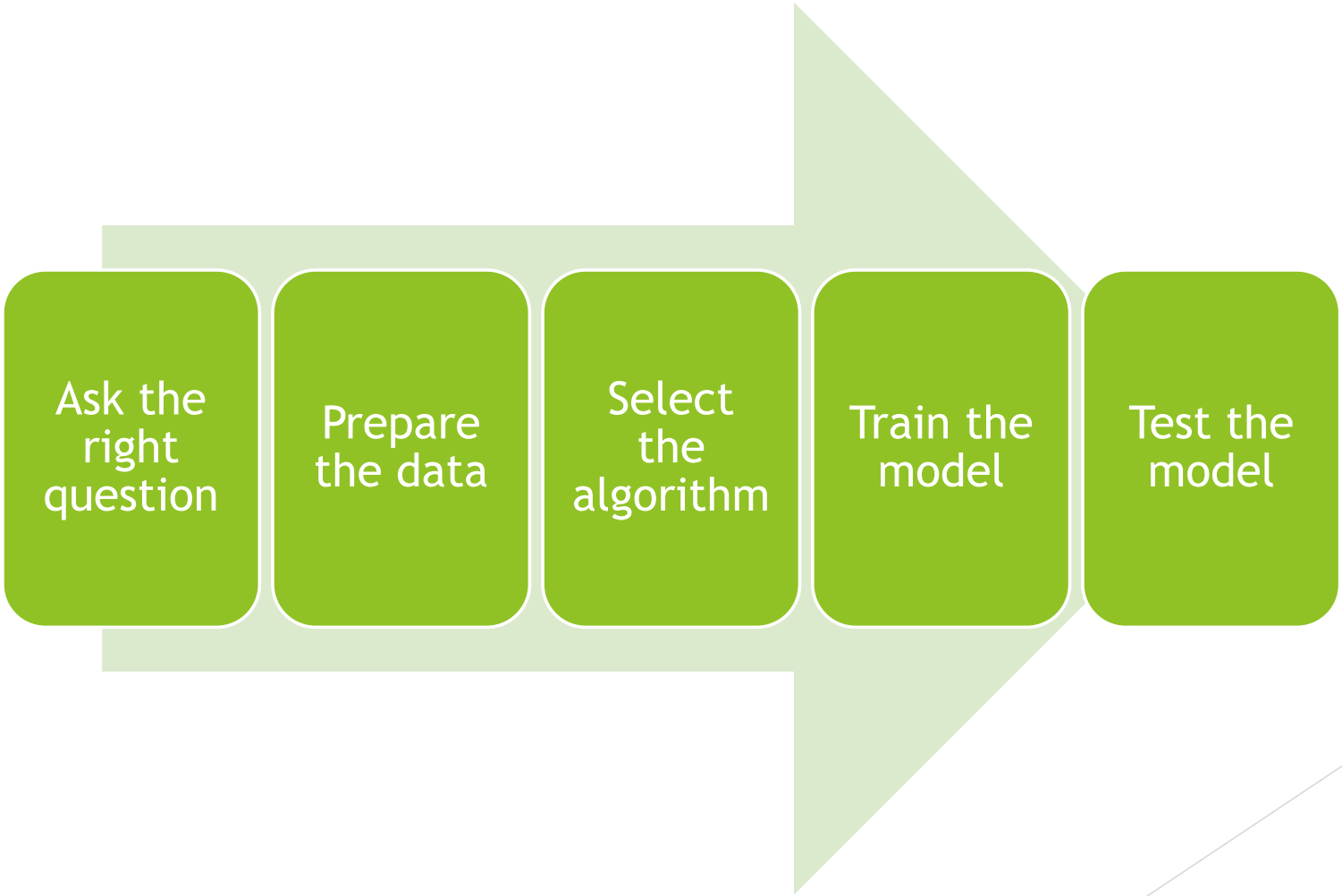
Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.



Machine learning applications

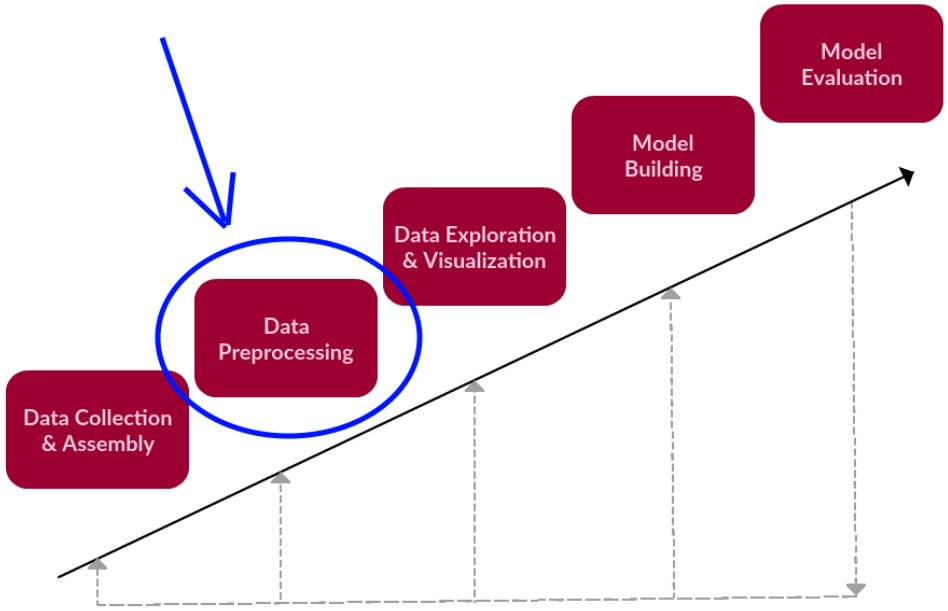


Machine learning workflow



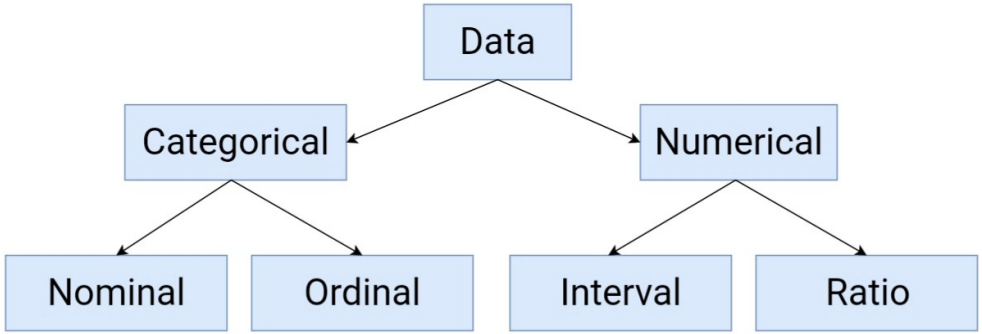
Data Pre-Processing

In any Machine Learning process, Data Pre-processing is that step in which the data gets transformed, or *Encoded*, to bring it to such a state that now the machine can easily parse it. In other words, the *features* of the data can now be easily interpreted by the algorithm.



- Importing the dataset
- Handling missing data
- Handling categorical data
- Splitting the dataset into training and test dataset
- Feature Scaling

Kinds of data



Nominal	Ordinal
Categorical variables without any implied order	Categorical variables with a natural implied order but the scale of difference is not defined
Example : A new car model comes in these colors : Black, Blue, White, Silver	Example : Sizes of clothes has a natural order : Extra Small < Small < Medium < Large < Extra Large - But this does not mean Large - Medium = Medium - Small

Interval	Ratio
Numeric variabes with a defnied unit of measurement, so the differences between values are meaningful	Numeric variables with a defined unit of measurement but both differences and ratios are meaningful
Examples : Calender Dates, Temperature in Celsius or Farhenheit	Examples : Temperature in Kelvin, Monetary quantities, Counts, Age, Mass, Length, Electrical Current

Importing libraries

```
# Importing libraries
# used for handling numbers
import numpy as np

# used for handling the dataset
import pandas as pd

# used for handling missing data
from sklearn.impute import SimpleImputer

# used for encoding categorical data
from sklearn.preprocessing import LabelEncoder, OneHotEncoder

# used for splitting training and testing data
from sklearn.model_selection import train_test_split

# used for feature scaling
from sklearn.preprocessing import StandardScaler
```

Importing data

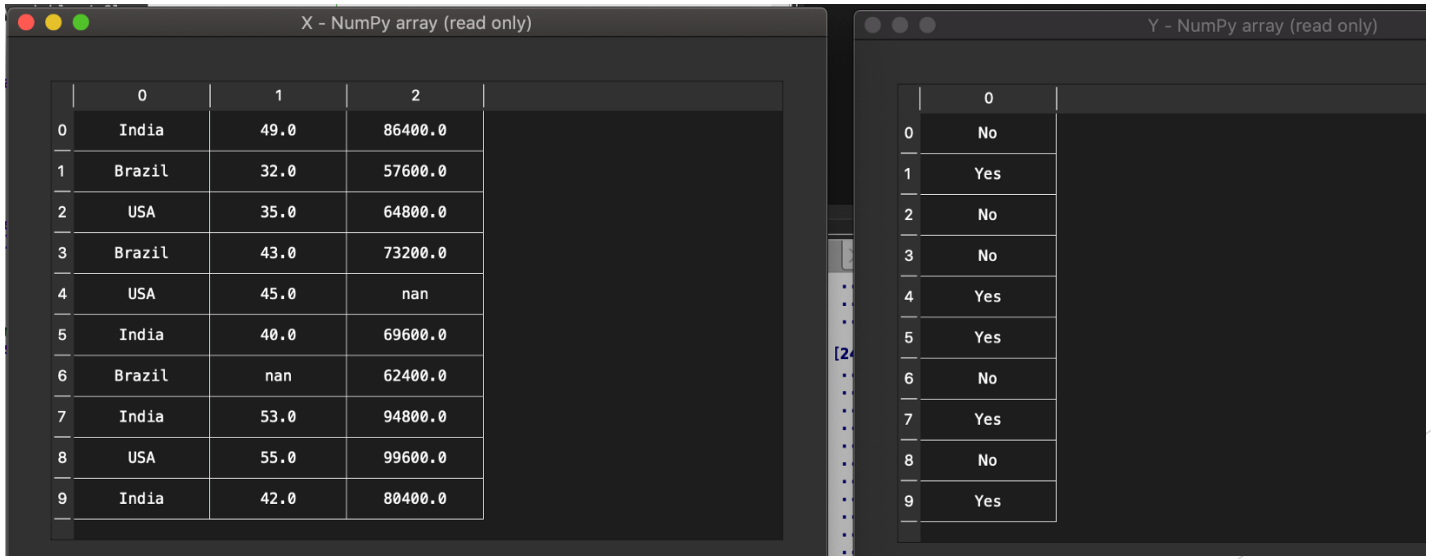
```
# to import the dataset into a variable
dataset = pd.read_csv('Data.csv')

# Splitting the attributes into independent and dependent attributes

# attributes to determine dependent variable / Class
X = dataset.iloc[:, :-1].values

# dependent variable / Class
Y = dataset.iloc[:, -1].values
```

1	Region	Age	Income	Online Shopper
2	India	49	86400	No
3	Brazil	32	57600	Yes
4	USA	35	64800	No
5	Brazil	43	73200	No
6	USA	45		Yes
7	India	40	69600	Yes
8	Brazil		62400	No
9	India	53	94800	Yes
10	USA	55	99600	No
11	India	42	80400	Yes



Handling missing data

```
# handling the missing data and replace missing values with nan from numpy and replace with
# mean of all the other values
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
imputer = imputer.fit(X[:, 1:])
X[:, 1:] = imputer.transform(X[:, 1:])
```

X - NumPy array (read only)

	0	1	2
0	India	49.0	86400.0
1	Brazil	32.0	57600.0
2	USA	35.0	64800.0
3	Brazil	43.0	73200.0
4	USA	45.0	76533.333333…
5	India	40.0	69600.0
6	Brazil	43.777777777…	62400.0
7	India	53.0	94800.0
8	USA	55.0	99600.0
9	India	42.0	80400.0

Note :- Missing Data can be replaced by Statistics – Mean, Median, Mode, IQR

Handling categorical data

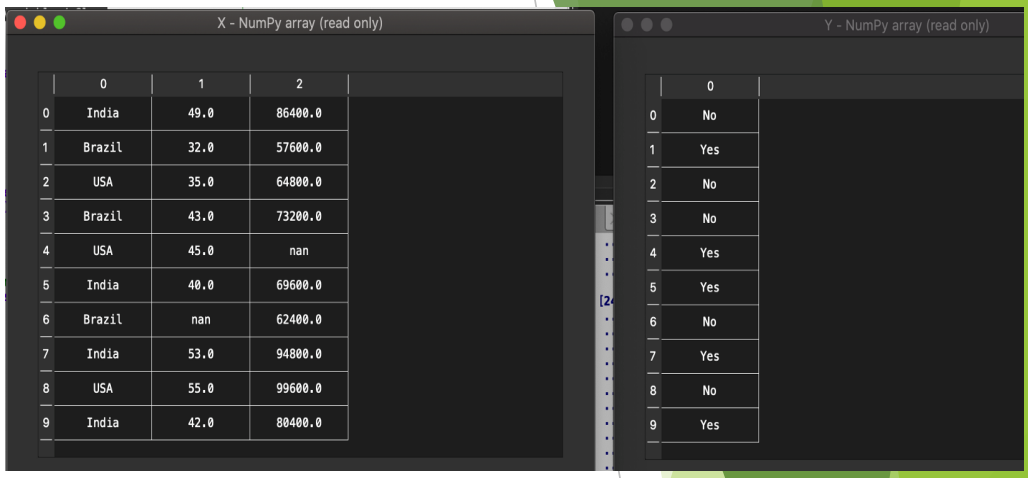
```
# Encode categorical data

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

# Label encoding
labelencoder_X = LabelEncoder()
X[:, 0] = labelencoder_X.fit_transform(X[:, 0])

# One hot encoding
onehotencoder = OneHotEncoder(categorical_features=[0])
X = onehotencoder.fit_transform(X).toarray()labelencoder_Y = LabelEncoder()

Y = labelencoder_Y.fit_transform(Y)
```



	0	1	2
0	India	49.0	86400.0
1	Brazil	32.0	57600.0
2	USA	35.0	64800.0
3	Brazil	43.0	73200.0
4	USA	45.0	nan
5	India	40.0	69600.0
6	Brazil	nan	62400.0
7	India	53.0	94800.0
8	USA	55.0	99600.0
9	India	42.0	80400.0

	0
0	No
1	Yes
2	No
3	No
4	Yes
5	Yes
6	No
7	Yes
8	No
9	Yes

	0	1	2	3	4
0	0	1	0	49	86400
1	1	0	0	32	57600
2	0	0	1	35	64800
3	1	0	0	43	73200
4	0	0	1	45	76533.3
5	0	1	0	40	69600
6	1	0	0	43.7778	62400
7	0	1	0	53	94800
8	0	0	1	55	99600
9	0	1	0	42	80400

	0
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	1
8	0
9	1

Splitting the dataset into training & test

- we use the training set to make the algorithm learn the behaviours present in the data and check the correctness of the algorithm by testing on testing set.

```
# splitting the dataset into training set and test set in the 80:20 ratio
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=0)
```

X_train - NumPy array

	0	1	2	3	4
0	0	0	1	45	76533.3
1	0	1	0	42	80400
2	1	0	0	32	57600
3	1	0	0	43.7778	62400
4	0	1	0	53	94800
5	1	0	0	43	73200
6	0	1	0	49	86400
7	0	1	0	40	69600

Y_train - NumPy array

	0
0	1
1	1
2	1
3	0
4	1
5	0
6	0
7	1

X_test - NumPy array

	0	1	2	3	4
0	0	0	1	35	64800
1	0	0	1	55	99600

Y_test - NumPy array

	0
0	0
1	0

Feature Scaling(Variable transformation)

- **Feature Scaling** is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing.
- Two columns age and income that contains numerical values. We notice that the variables are not on the same scale because the age are going from 32 to 55 and the salaries going from 57.6 K to like 99.6 K.
- So because this age variable in the salary variable don't have the same scale. This will cause some issues in your models. And why is that. It's because lot of machine are based on what is called the Euclidean distance.
- We use feature scaling to convert different scales to a standard scale to make it easier for Machine Learning algorithms.

```
# feature scaling

sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

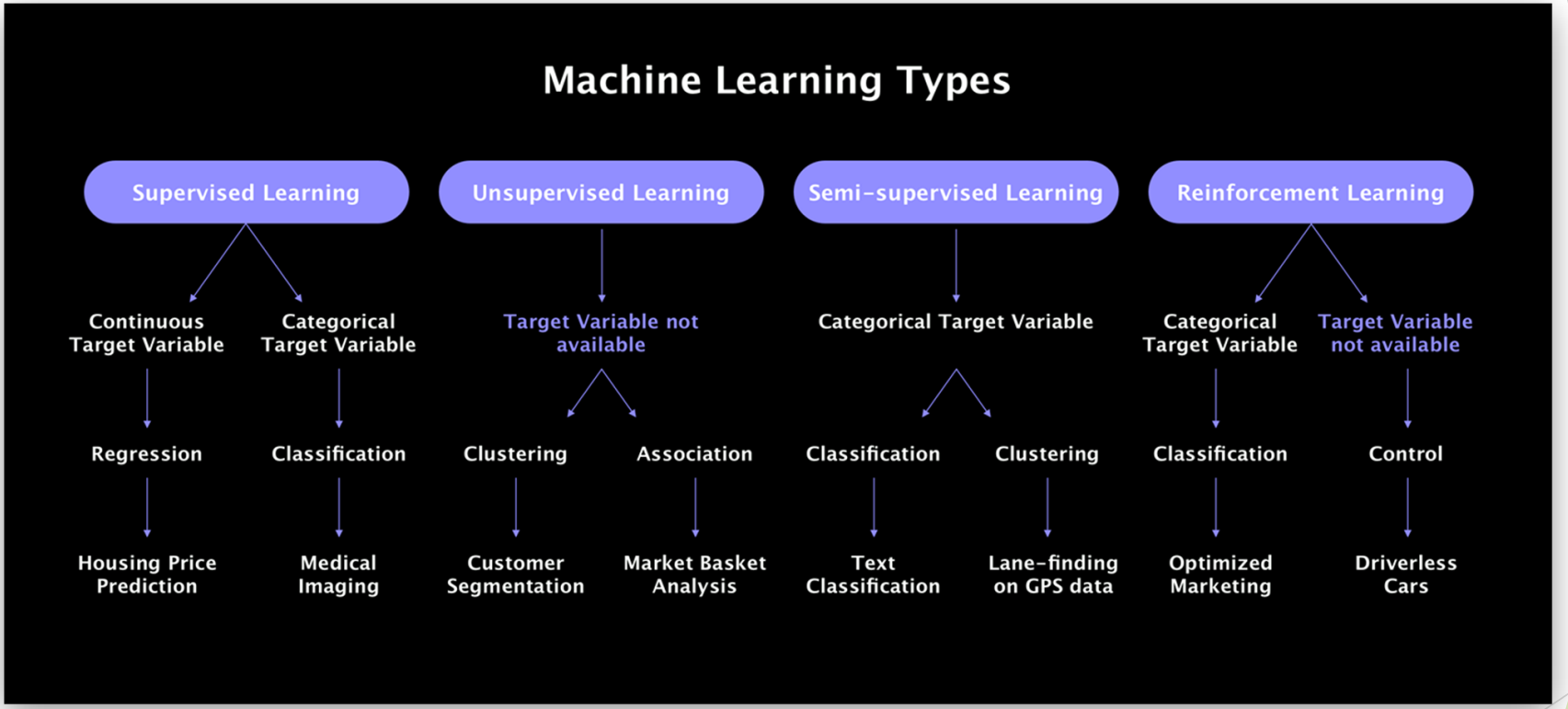
X_train - NumPy array

	0	1	2	3	4
0	-0.774597	-1	2.64575	0.263068	0.123815
1	-0.774597	1	-0.377964	-0.253501	0.461756
2	1.29099	-1	-0.377964	-1.9754	-1.53093
3	1.29099	-1	-0.377964	0.0526135	-1.11142
4	-0.774597	1	-0.377964	1.64059	1.7203
5	1.29099	-1	-0.377964	-0.0813118	-0.167514
6	-0.774597	1	-0.377964	0.951826	0.986148
7	-0.774597	1	-0.377964	-0.597881	-0.482149

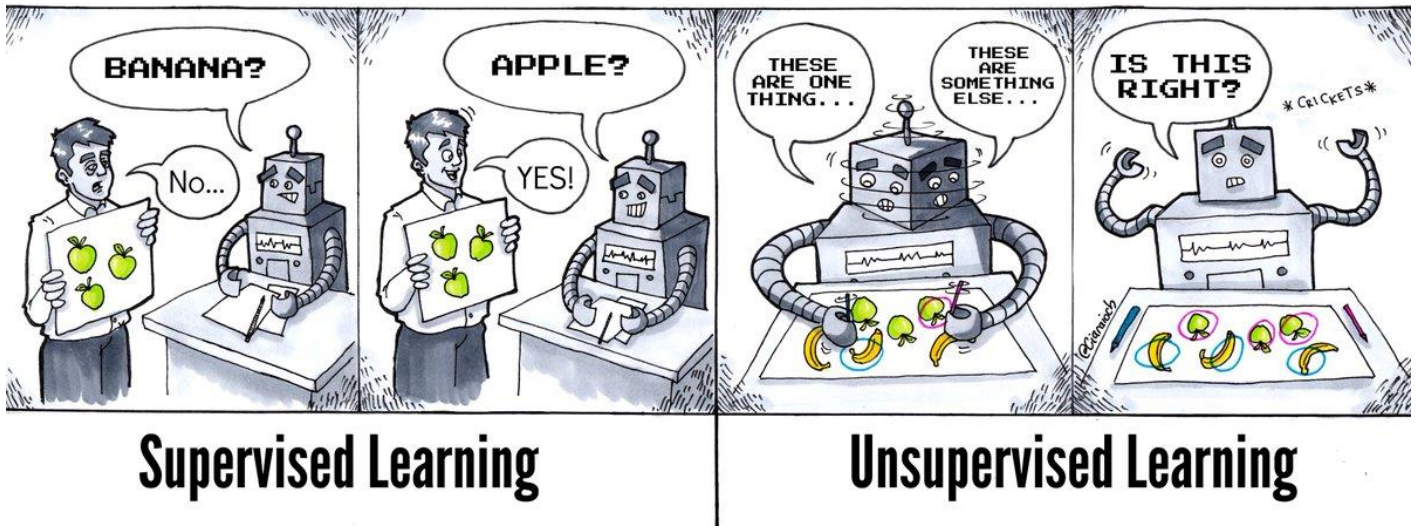
X_test - NumPy array

	0	1	2	3	4
0	-0.774597	-1	2.64575	-1.45883	-0.901663
1	-0.774597	-1	2.64575	1.98496	2.13981

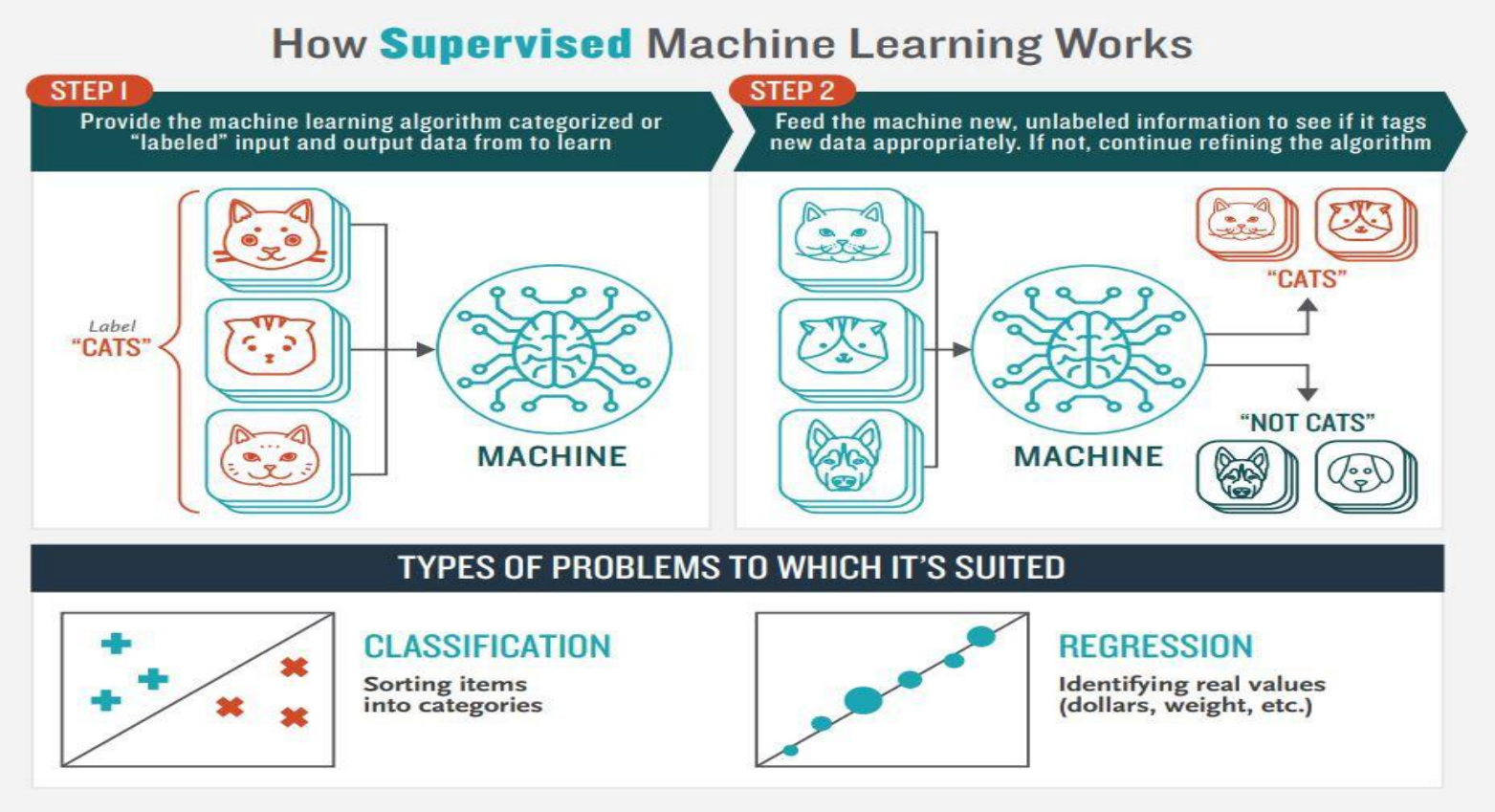
Types of Machine Learning



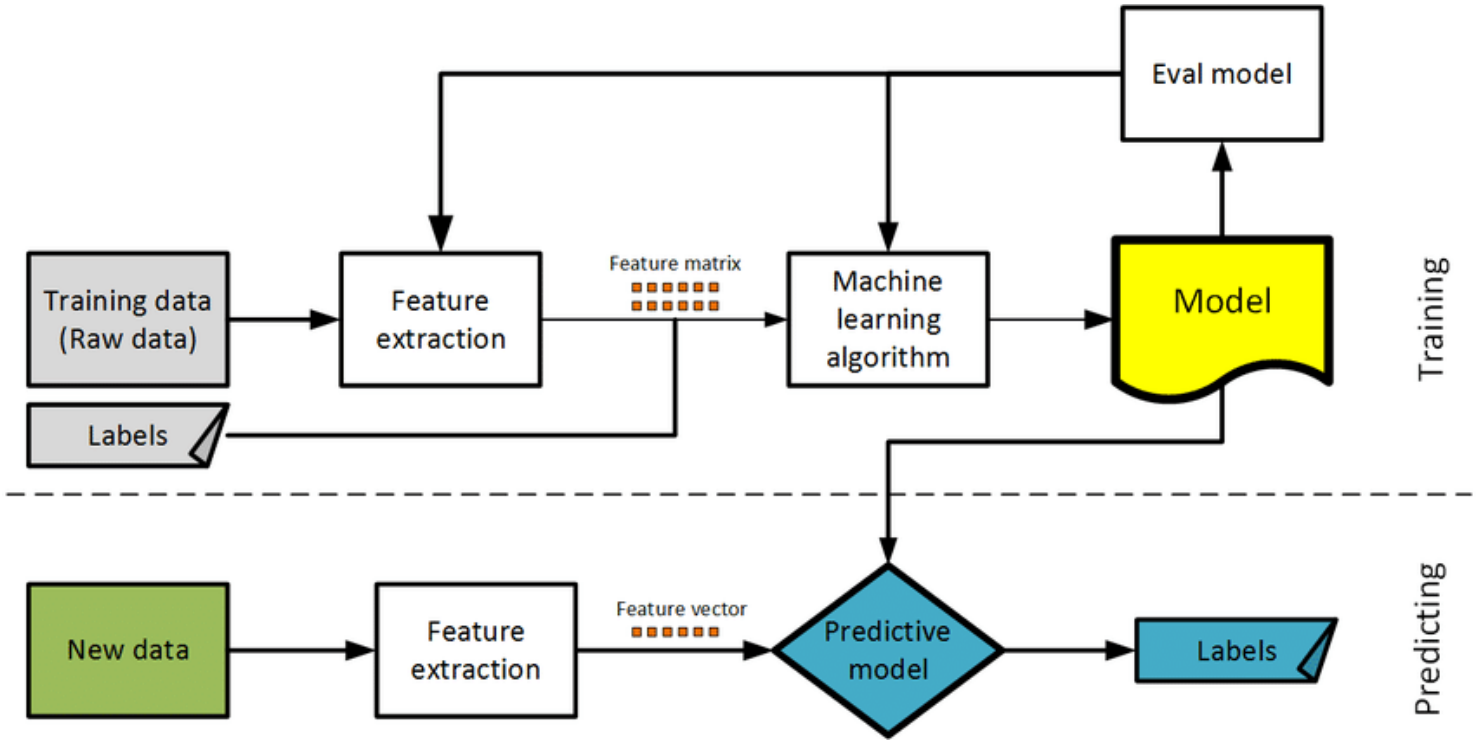
Types of Machine Learning



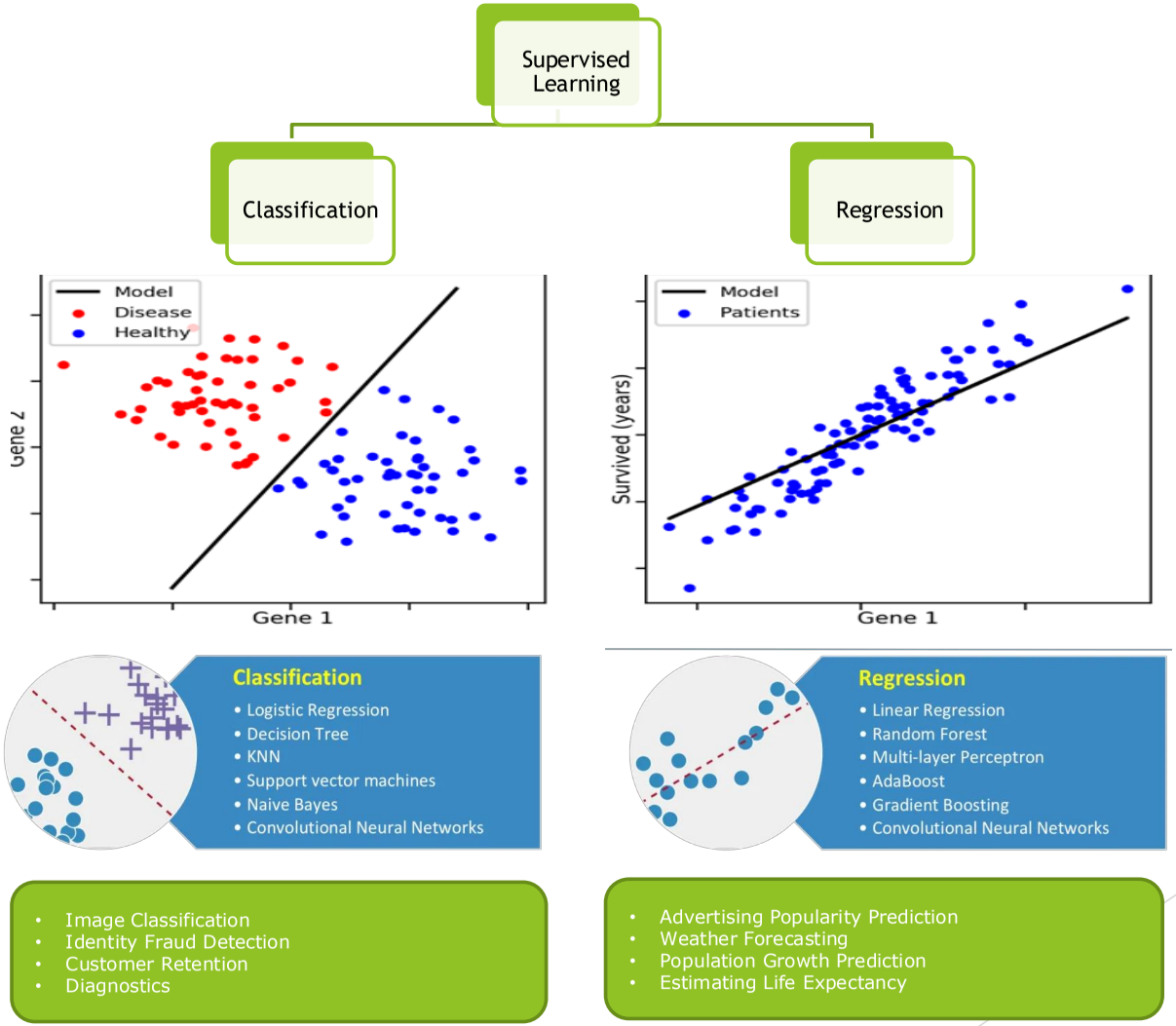
Supervised Learning



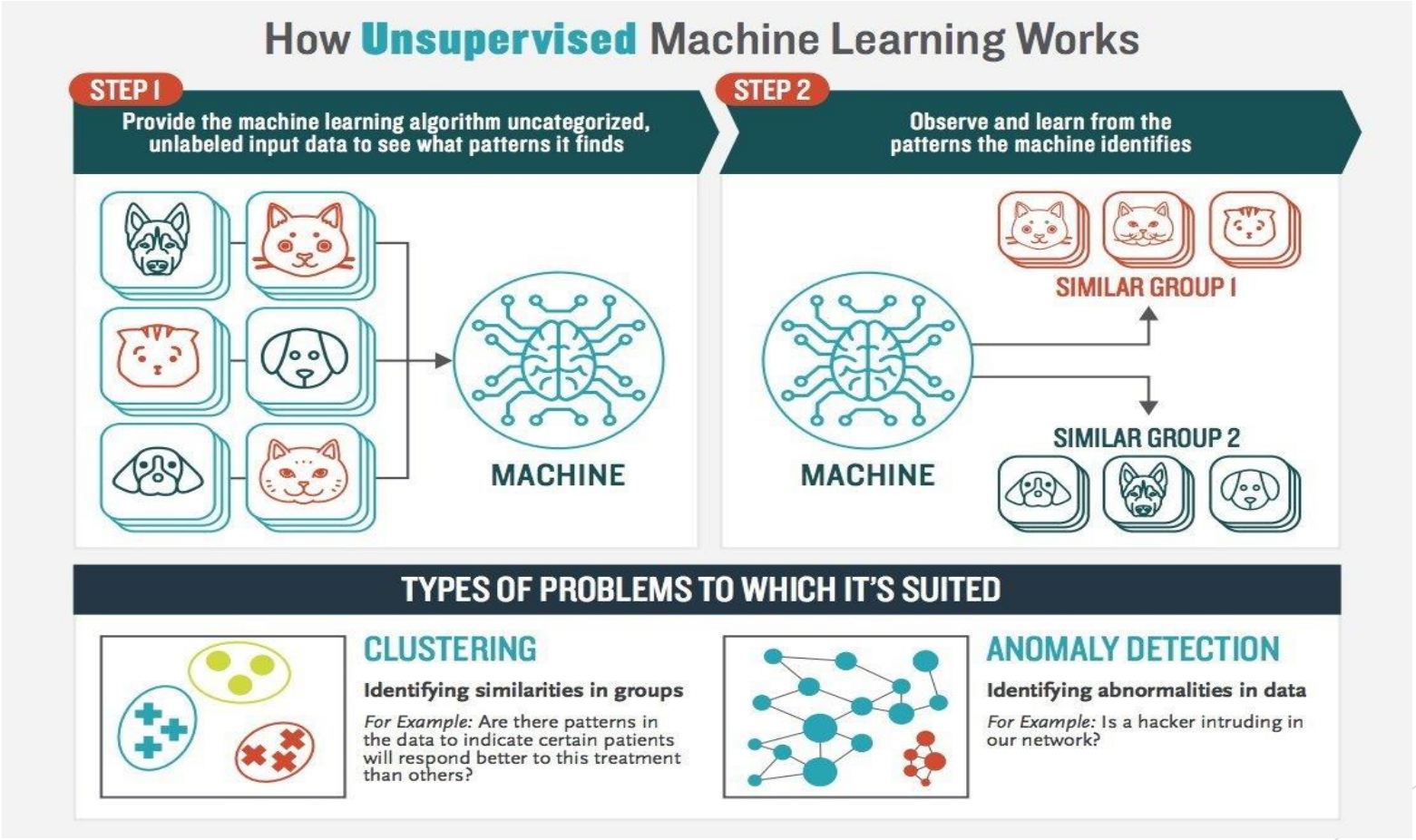
Supervised Learning – Flow chart



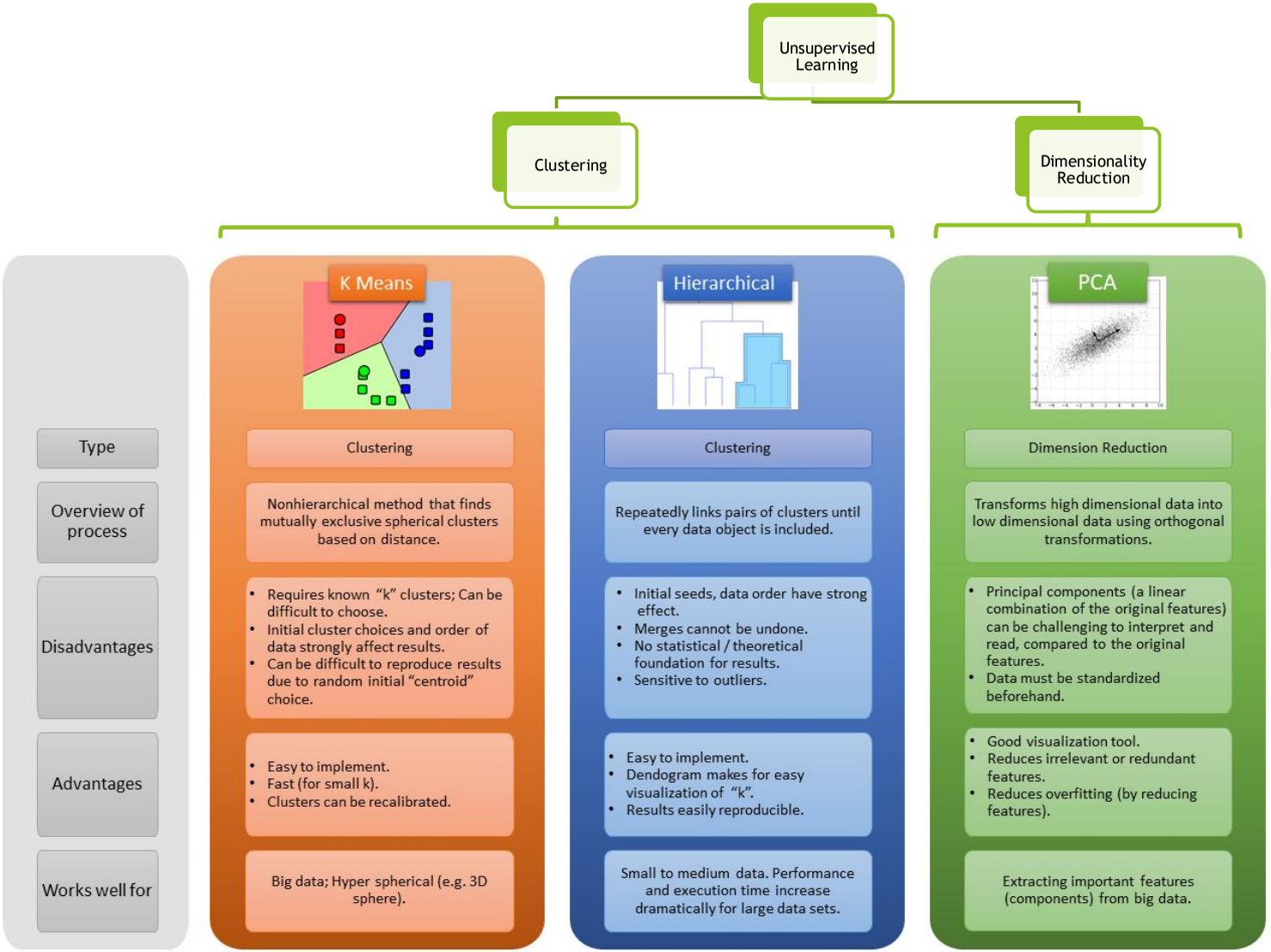
Supervised Learning



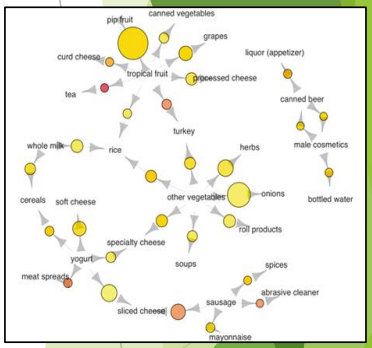
Unsupervised Learning



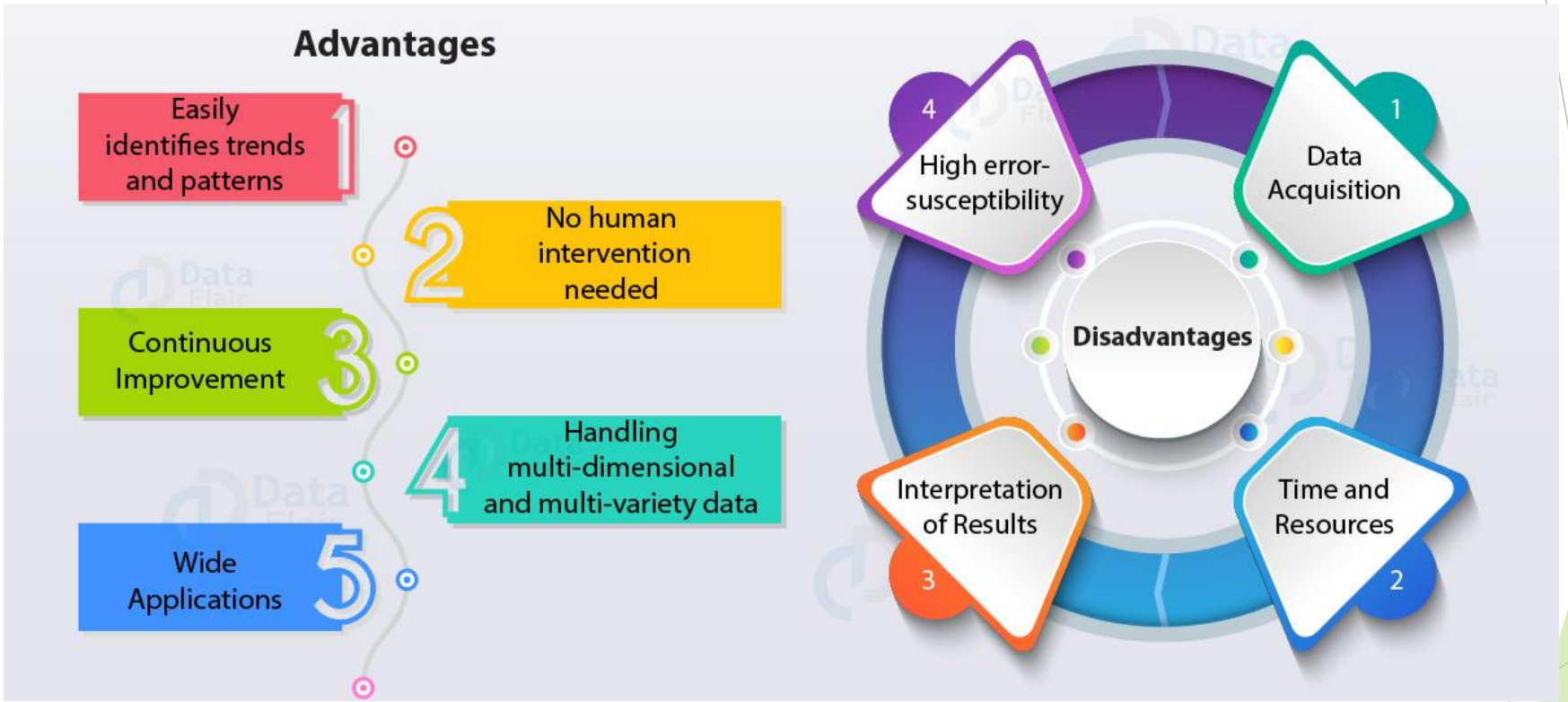
Unsupervised Learning



Association



Machine Learning – Advantages & Disadvantages



Machine Learning – Email spam case study

ABSTRACT
The increasing volume of unsolicited bulk e-mail (also known as spam) has generated a need for reliable anti-spam filters. Machine learning techniques now days used to automatically filter the spam e-mail in a very successful rate. In this paper we review some of the most popular machine learning methods (Bayesian classification, k-NN, ANNs, SVMs, Artificial immune system and Rough sets) and of their applicability to the problem of spam Email classification. The comparison of algorithms performance on the spam corpus is presented.

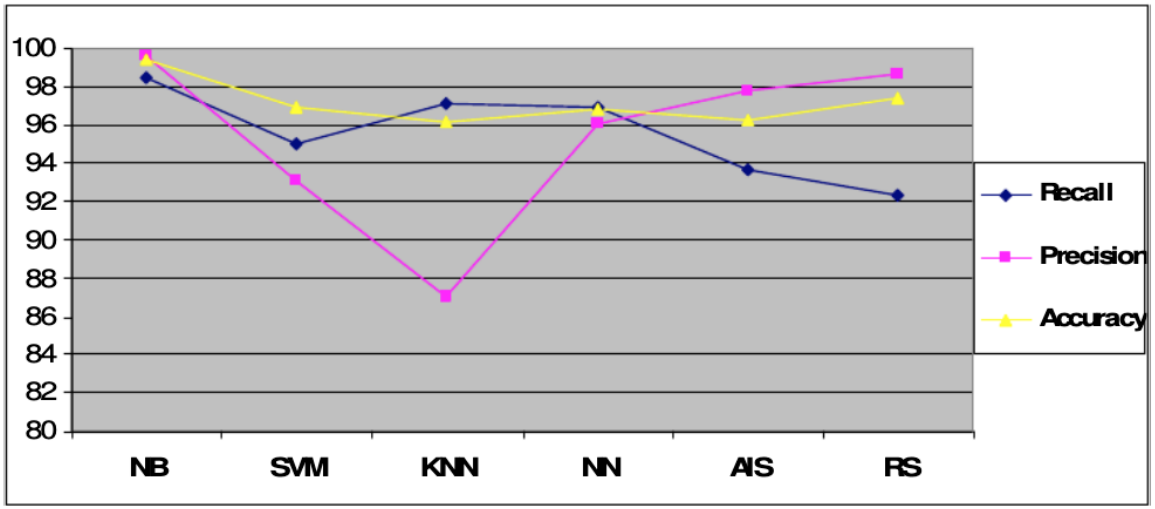


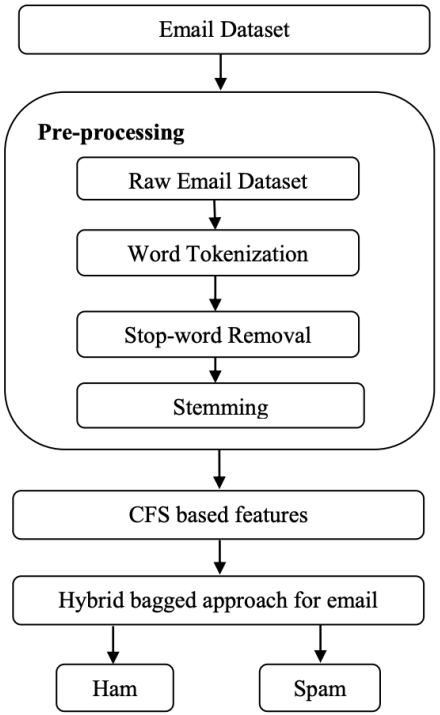
Figure 1. Spam Recall, Spam Precision and Accuracy curves of six classifiers

Note :- Naïve bayes method has a very satisfying performance among the other methods

Ref :- https://www.researchgate.net/publication/50211017_Machine_Learning_Methods_for_Spam_E-Mail_Classification

Machine Learning – Email spam case study

Abstract:
Spam email is one of the biggest issues in the world of internet. Spam emails not only influence the organisations financially but also exasperate the individual email user. This paper aims to propose a machine learning based hybrid bagging approach by implementing the two machine learning algorithms: Naïve Bayes and J48 (decision tree) for the spam email detection. In this process, dataset is divided into different sets and given as input to each algorithm. Total three experiments are performed and the results obtained are compared in terms of precision, recall, accuracy, f-measure, true negative rate, false positive rate and false negative rate. The two experiments are performed using individual Naïve Bayes & J48 algorithms. Third experiment is the proposed SMD system implemented using hybrid bagged approach. The overall accuracy of 87.5% achieved by the hybrid bagged approach based SMD system.



Evaluation Measures	Naïve Bayes	J48	Hybrid Bagged approach
TP	81	89	85
FP	14	6	10
TN	86	94	90
FN	19	11	15
Precision (%)	85.26	93.68	89.47
Recall (%)	81	89	85
Accuracy (%)	83.5	91.5	87.5
F-Measure (%)	89.27	84.8	87.03
TNR (%)	86	94	90
FPR (%)	19	11	15
FNR (%)	14	6	10

Note :- J48 method has a very satisfying performance among the other methods

Ref :- http://oaji.net/pdf.html?n=2017/3603-1524458638.pdf

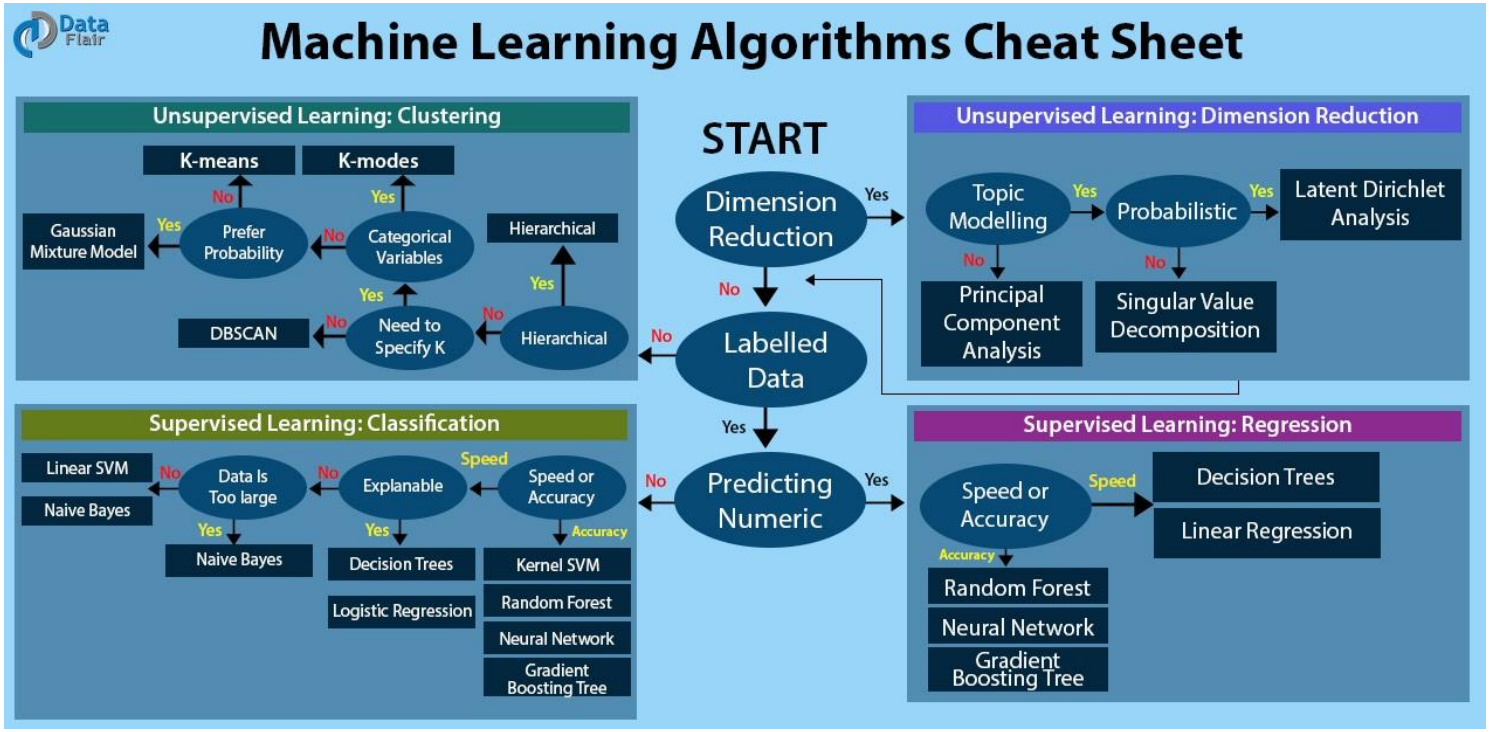
Machine Learning - Summary

	Overview	Process	Subtypes	Examples
Supervised Learning	Majority of algorithms. Machine is trained using **well-labeled data**; inputs and outputs are matched.	Mapping function takes inputs and matches to outputs, creating a target function.	Classification, Regression	Linear regression, Random forest, SVM.
Unsupervised Learning	**Unlabeled data** (inputs only) is analyzed. Learning happens without supervision.	Inputs are used to create a model of the data.	Clustering, Association.	PCA, k-Means, Hierarchical clustering.
Semi supervised	**Some data is labeled**, some not. Goal: better results than labeled data alone. Good for real world data.	Combination of above processes.	All the above.	Self training, Mixture models, Semi-supervised SVM

Thank You

Appendix

Forecasting techniques, Building a Prediction model using AIML



Forecasting techniques, Prediction models, choosing the right algorithm with examples