# Machine Learning & Deep Learning in Python & R

ABHIJITH U
RA1931241010017
BCA A - III Year

# Intro to  Python

Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

## Variables

Variable is a name that is used to refer to memory location. Python variable is also known as an identifier and used to hold value.

## Data Types

Variables can hold values, and every value has a data-type. Python is a dynamically typed language; hence we do not need to define the type of the variable while declaring it.

## Operators

The operator can be defined as a symbol which is responsible for a particular operation between two operands.

## Tuple

Python Tuple is used to store the sequence of immutable Python objects. The tuple is similar to lists since the value of the items stored in the list can be changed, whereas the tuple is immutable, and the value of the items stored in the tuple cannot be changed.

## Sets

A Python set is the collection of the unordered items. Each element in the set must be unique, immutable, and the sets remove the duplicate elements. Sets are mutable which means we can modify it after its creation.

## Conditional Statements

Decision making is the most important aspect of almost all the programming languages. As the name implies, decision making allows us to run a particular block of code for a particular decision. Here, the decisions are made on the validity of the particular conditions. Condition checking is the backbone of decision making.

## Loops

The programming languages provide various types of loops which are capable of repeating some specific code several numbers of times.

## List

A list in Python is used to store the sequence of various types of data. Python lists are mutable type its mean we can modify its element after it created.

## Dictionary

Python Dictionary is used to store the data in a key-value pair format. The dictionary is the data type in Python, which can simulate the real-life data arrangement where some specific value exists for some particular key. It is the mutable data-structure. The dictionary is defined into element Keys and values.

## Jump Statements

Jump statements in python are used to alter the flow of a loop like you want to skip a part of a loop or terminate a loop.

## Exceptional Handling

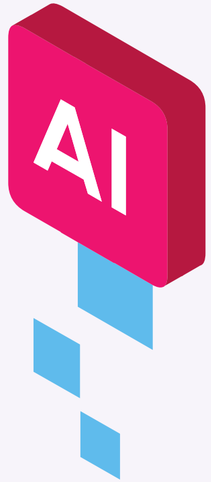
An exception can be defined as an unusual condition in a program resulting in the interruption in the flow of the program. Python provides a way to handle the exception so that the code can be executed without any interruption. If we do not handle the exception, the interpreter doesn't execute all the code that exists after the exception.

## Libraries

There are over 137,000 python libraries present today. Python libraries play a vital role in developing machine learning, data science, data visualization, image and data manipulation applications and more.

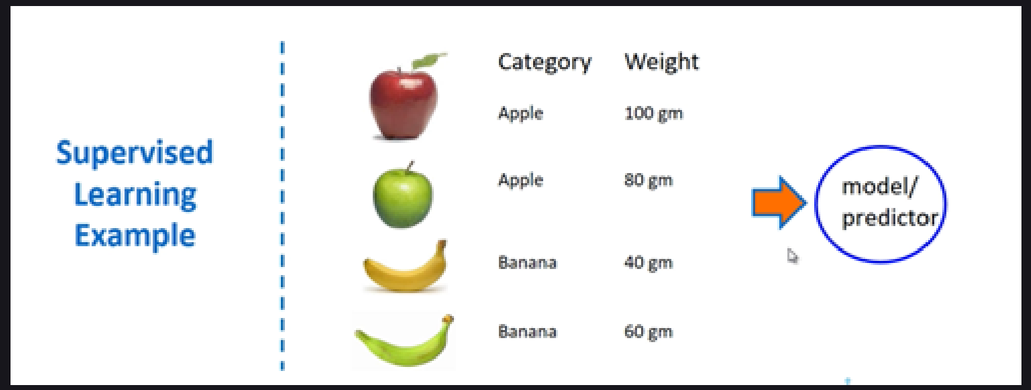
# Intro to Machine Learning

in Python

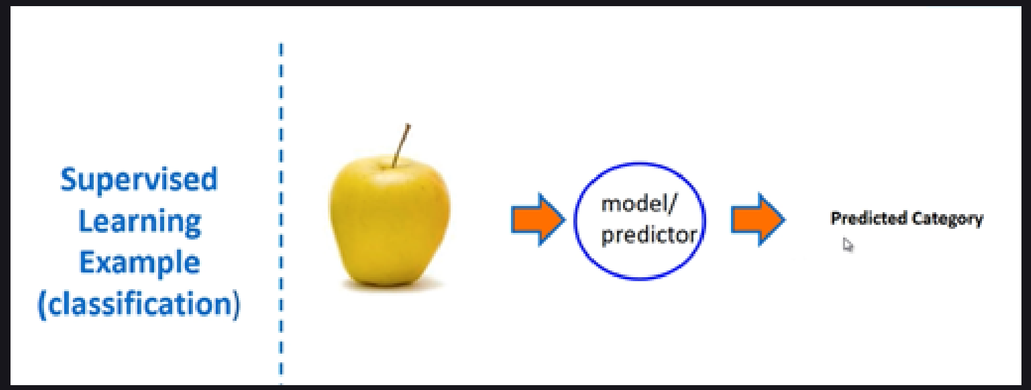



# Supervised & Unsupervised Learning

If you have input variables (X) and an output variable (Y), we use an algorithm to map the relation between X & Y. Such learning is called Supervised Learning.
Example:



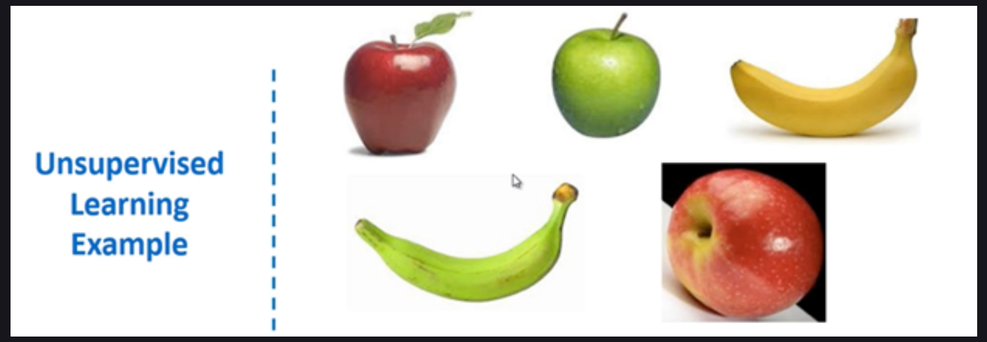

The model learns from this data. Now when you show apple; it understands the object as an apple from the previous data collected and classifies into a predicted category.



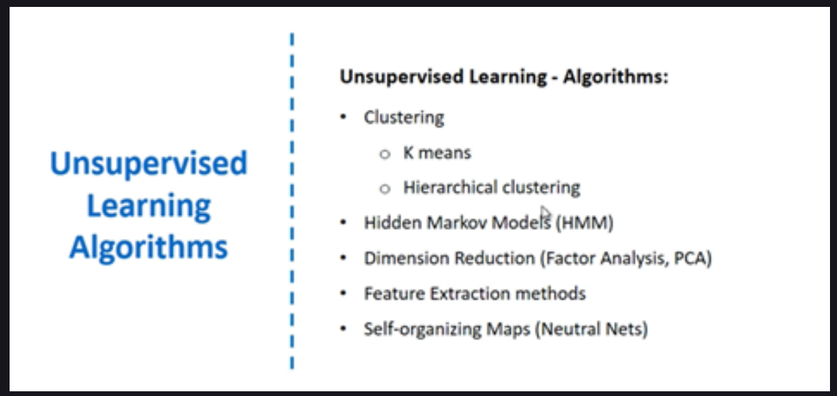
# Applications :

- Pattern Recognition
- Face Recognition
- Character Recognition
- Medical Diagnosis
- Web Advertising

Unlike supervised, here you have only input data(X) and no output data.
We need to learn the relation between these input variables and predict an output variable. Such learning is called Unsupervised Learning.
Example:




With just these 5 images, the model will categorize it into different categories. For eg; maybe based on color or maybe based on shape etc.



# Steps in Building ML Model:

### 1. Problem Formulation

Convert your business problem into Statistical Problem. Clearly define the dependent and independent variable. Identify whether you want to predict or infer.

### 2. Data Tidying

Transform collected data into a useable data table format.

### 3. Pre-Processing

Filter data, Aggregate values, Missing value treatment, Outlier Treatment, Variable transformation, Variable reduction.

### 4. Train-Test Split

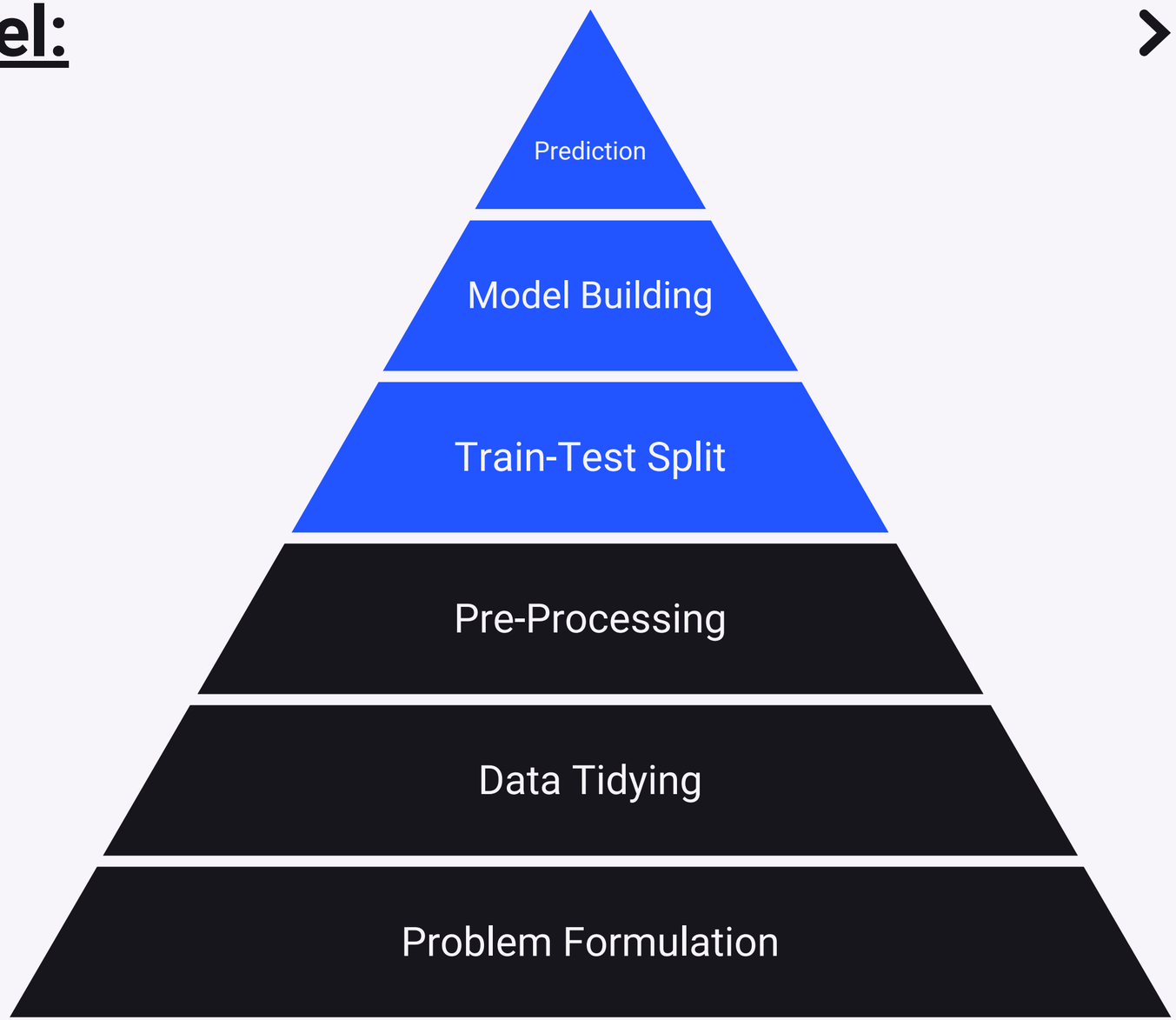
Splitting the data into Training data and Testing data for the model to train from and test its performance respectively.

### 5. Model Building

Build the trained model and test its accuracy. Improve the accuracy using hyperparameter tuning.

### 6. Prediction

Use the model to do predictions for future variables.



# Regression

Simple Linear Regression : A linear approach to modelling the relationship between a dependent variable and one or more independent variables.
Multiple Linear Regression : multiple predictor variables.

In [1]:
```
db = r"D:\Coding_Backup\ML LEARNING\Data Files\Data Files\1. ST Academy
- Crash course and Regression files\House_Price.csv"
```

In [2]:
```
import pandas as pd
```

In [3]:
```
df = pd.read_csv(db)
```

In [4]:
```
df.head()
```

Out[4]:

| | price | crime_rate | resid_area | air_qual | room_num | age | dist1 | dist2 | dist3 | dist4 | teachers | poo |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 24.0 | 0.00632 | 32.31 | 0.538 | 6.575 | 65.2 | 4.35 | 3.81 | 4.18 | 4.01 | 24.7 | |
| **1** | 21.6 | 0.02731 | 37.07 | 0.469 | 6.421 | 78.9 | 4.99 | 4.70 | 5.12 | 5.06 | 22.2 | |
| **2** | 34.7 | 0.02729 | 37.07 | 0.469 | 7.185 | 61.1 | 5.03 | 4.86 | 5.01 | 4.97 | 22.2 | |
| **3** | 33.4 | 0.03237 | 32.18 | 0.458 | 6.998 | 45.8 | 6.21 | 5.93 | 6.16 | 5.96 | 21.3 | |
| **4** | 36.2 | 0.06905 | 32.18 | 0.458 | 7.147 | 54.2 | 6.16 | 5.86 | 6.37 | 5.86 | 21.3 | |

2 Questions from the above dataset :

*Prediction Question :*

How accurately can i predict the price of a house, given the values of all variables.

*Inferential Question :*

How accurately can we estimate the effect of each of this variables on the house price.

# Test-Train Split Techniques

## Validation set approach

1. Randomly divides data into 2 parts (Training Set and Test Set).
2. Usually split is 80:20 (Training : Test)

Limitations:
- Part of the data will not be used for Training.
- Test Error can be highly varying depending on which observations are selected fortraining and testing.

## Leave one out cross validation

Suppose we have 'n' observations. We will use first observation as testing data and remaining 'n-1' observations are used for training. Then we will keep the second observation for testing purposes. and run the model on the remaining. This will continue for 'n' times where everytime we will keep one observation for testing and the remaining will be used for training. Finally we will take the Average of the error on each of these Testing observations.

Limitaitons:
- Since this model is running 'n' times, this method can be expensive.

## K-Fold validation

In this, we will divide the data into k-sets. And then we will train the data on k-1 sets anduse the k-th set for testing. Leave one out cross validation is a special type of k-fold validation. When k = n; then K-fold validation and Leave one out cross validation are same.

# Classification

Problem Statement :
You are a manager in a Real Estate Company. You want to find out the selling potential of a company. You will be provided with the data of past property transactions.You have to predict whether a property will be sold within 3 months or not.

| price | resid_area | air_qual | room_num | age | dist1 | dist2 | dist3 | dist4 | teachers | poor_prop | airport | n_hos_bed | n_hot_roo | waterbody | rainfall | bus_ter | parks | Sold |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 24 | 32.31 | 0.538 | 6.575 | 65.2 | 4.35 | 3.81 | 4.18 | 4.01 | 24.7 | 4.98 | YES | 5.48 | 11.192 | River | 23 | YES | 0.049347 | 0 |
| 21.6 | 37.07 | 0.469 | 6.421 | 78.9 | 4.99 | 4.7 | 5.12 | 5.06 | 22.2 | 9.14 | NO | 7.332 | 12.1728 | Lake | 42 | YES | 0.046146 | 1 |
| 34.7 | 37.07 | 0.469 | 7.185 | 61.1 | 5.03 | 4.86 | 5.01 | 4.97 | 22.2 | 4.03 | NO | 7.394 | 101.12 | None | 38 | YES | 0.045764 | 0 |
| 33.4 | 32.18 | 0.458 | 6.998 | 45.8 | 6.21 | 5.93 | 6.16 | 5.96 | 21.3 | 2.94 | YES | 9.268 | 11.2672 | Lake | 45 | YES | 0.047151 | 0 |
| 36.2 | 32.18 | 0.458 | 7.147 | 54.2 | 6.16 | 5.86 | 6.37 | 5.86 | 21.3 | 5.33 | NO | 8.824 | 11.2896 | Lake | 55 | YES | 0.039474 | 0 |

In our house-price Dataset;

**Data Dictionary**

**House Pricing Dataset**

The data set contains 506 observations of whether the property was sold within three months of getting listed. Corresponding to each house, data of 18 other variables is available on which it is suspended to depend.

| | |
|---|---|
| price | Asking price of the property by the owner |
| resid_area | Proportion of residential area in the town |
| air_qual | Quality of air in that neighborhood |
| room_num | Average number of rooms in houses of that locality |
| age | How old is the house construction in years |
| dist1 | Distance from employment hub 1 |
| dist2 | Distance from employment hub 2 |
| dist3 | Distance from employment hub 3 |
| dist4 | Distance from employment hub 4 |
| teachers | Number of teachers per thousand population in the town |
| poor_prop | Proportion of poor population in the town |
| airport | Is there an airport in the city? (Yes/No) |
| n_hos_beds | Number of hospital beds per 1000 population in the town |
| n_hot_rooms | Number of hotel rooms per 1000 population in the town |
| waterbody | What type of natural fresh water source is there in the city (lake/ river/ both/ none) |
| rainfall | The yearly average rainfall in centimeters |
| bus_ter | Is there a bus terminal in the city? (Yes/No) |
| Parks | Proportion of land assigned as parks and green areas in the town |
| Sold | Whether the property was sold within three months of getting listed |

## Summary

1. Data Collection

2. Data Pre-processing
    *Outlier Treatment*
    Missing Value imputation
    * Variable Transformation

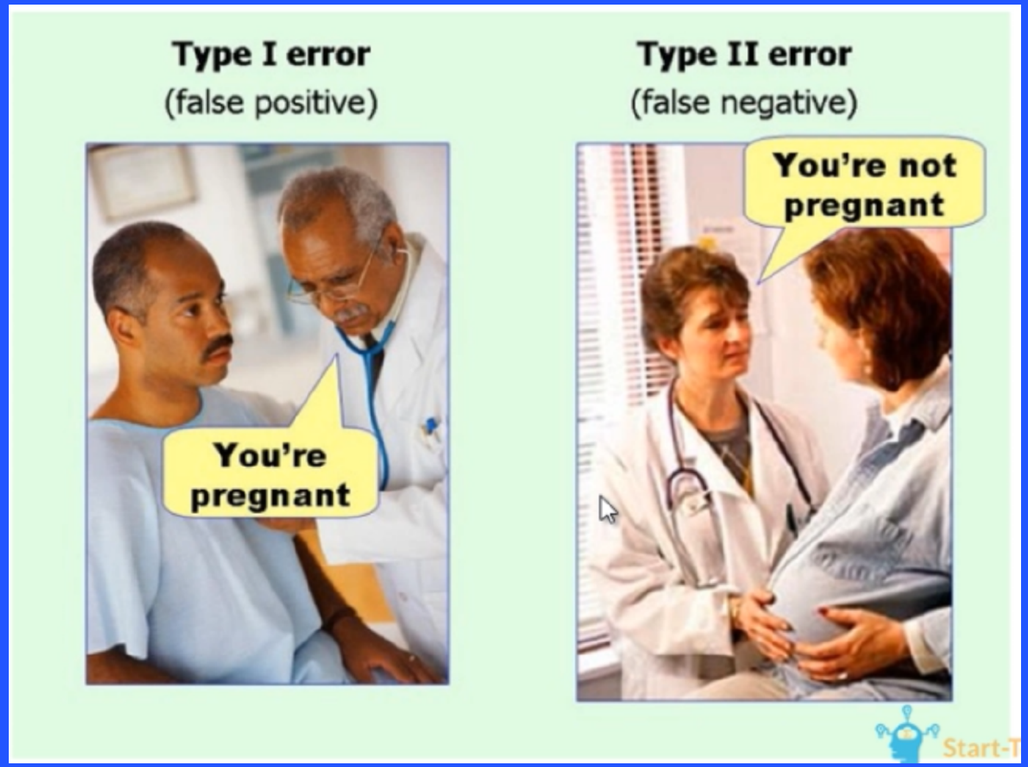
3. Model Training
    *Test-Train Split*
    Use template to train
    *Do iterations*
    Compare performance of different methods using test set

4. Select the best model
    *For prediction purposes use model with best accuracy.*
    For interpretation purposes look at the coefficient values of parametric models.

# Confusion Matrix

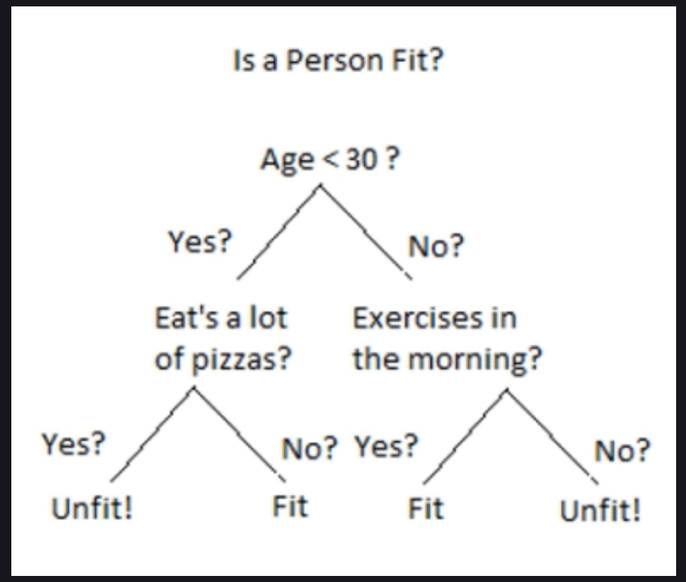
After training the model, we can create a matrix that can measure the accuracy of our model.
This is called as Confusion Matrix .

| | | *True default status* | | |
|---|---|---|---|---|
| | | No | Yes | Total |
| *Predicted default status* | No | 9,432 | 138 | 9,570 |
| | Yes | 235 | 195 | 430 |
| | Total | 9,667 | 333 | 10,000 |



# Decision Trees

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements



```
df = pd.read_csv(r"D:\Coding_Backup\ML LEARNING\Data Files\Data Files\3. ST Academy - Decision Trees resource files\Movie_regression.csv")
```

In [4]:

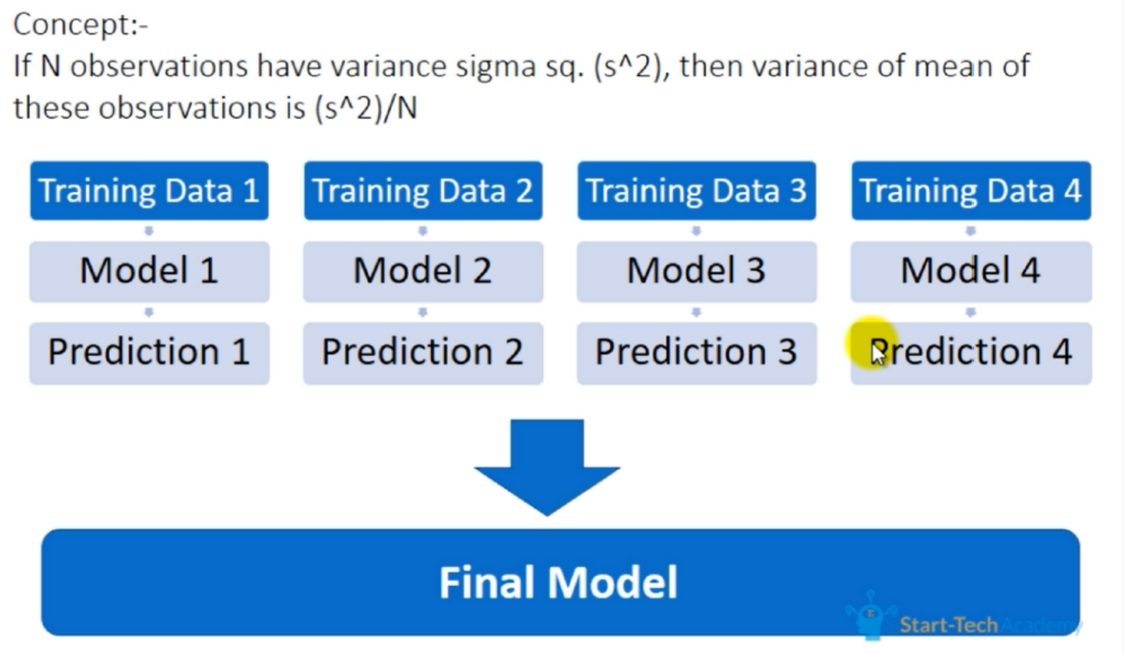
```
df.info()
```

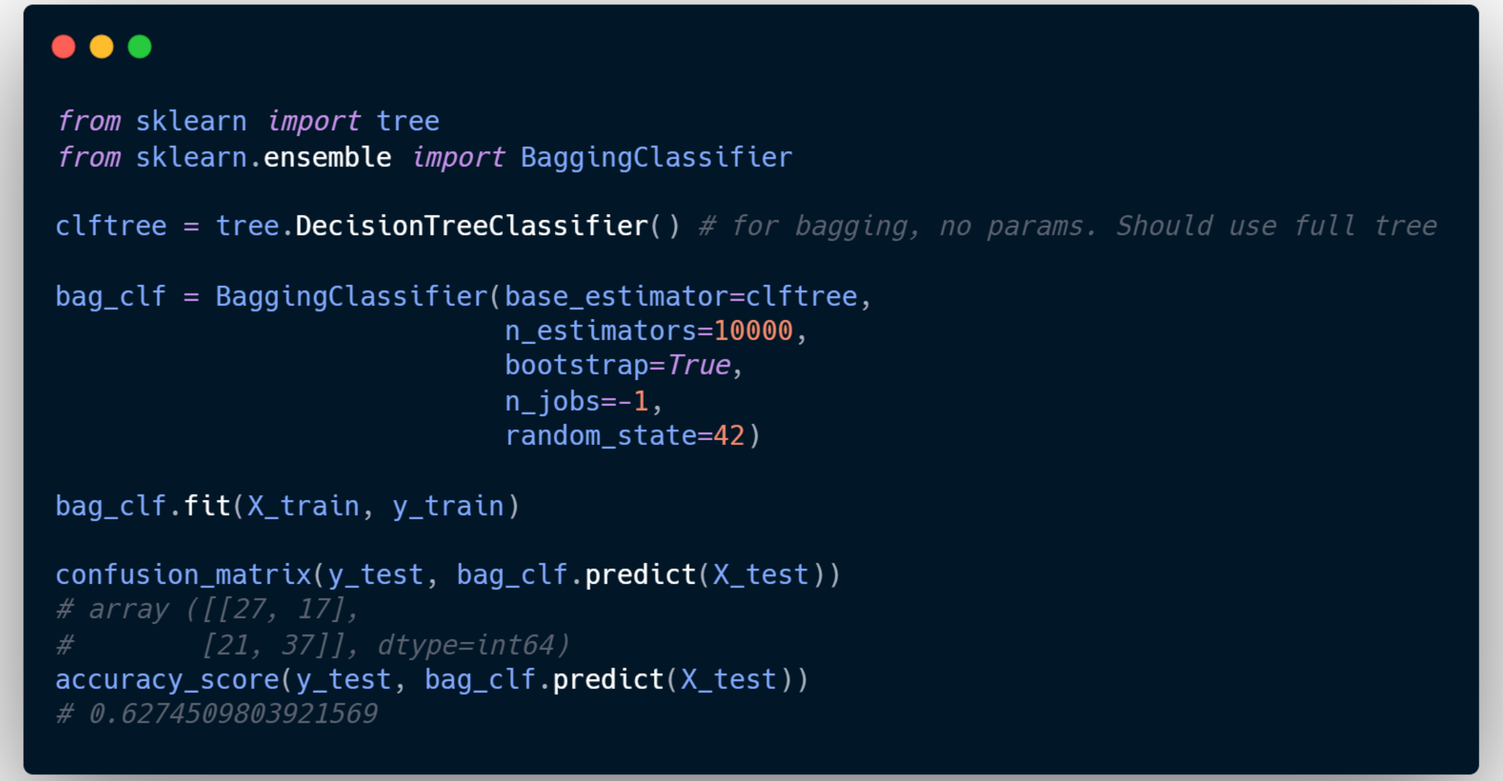
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 18 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Marketing expense    506 non-null    float64
 1   Production expense   506 non-null    float64
 2   Multiplex coverage   506 non-null    float64
 3   Budget               506 non-null    float64
 4   Movie_length         506 non-null    float64
 5   Lead_ Actor_Rating   506 non-null    float64
 6   Lead_Actress_rating  506 non-null    float64
 7   Director_rating      506 non-null    float64
 8   Producer_rating      506 non-null    float64
 9   Critic_rating        506 non-null    float64
 10  Trailer_views        506 non-null    int64  
 11  3D_available         506 non-null    object 
 12  Time_taken           494 non-null    float64
 13  Twitter_hastags      506 non-null    float64
 14  Genre                506 non-null    object 
 15  Avg_age_actors       506 non-null    int64  
 16  Num_multiplex        506 non-null    int64  
 17  Collection           506 non-null    int64  
dtypes: float64(12), int64(4), object(2)
memory usage: 71.3+ KB
```

# Ensemble Techniques

# Bagging

The technique of getting predictions from multiple training data and model is called as Bagging.



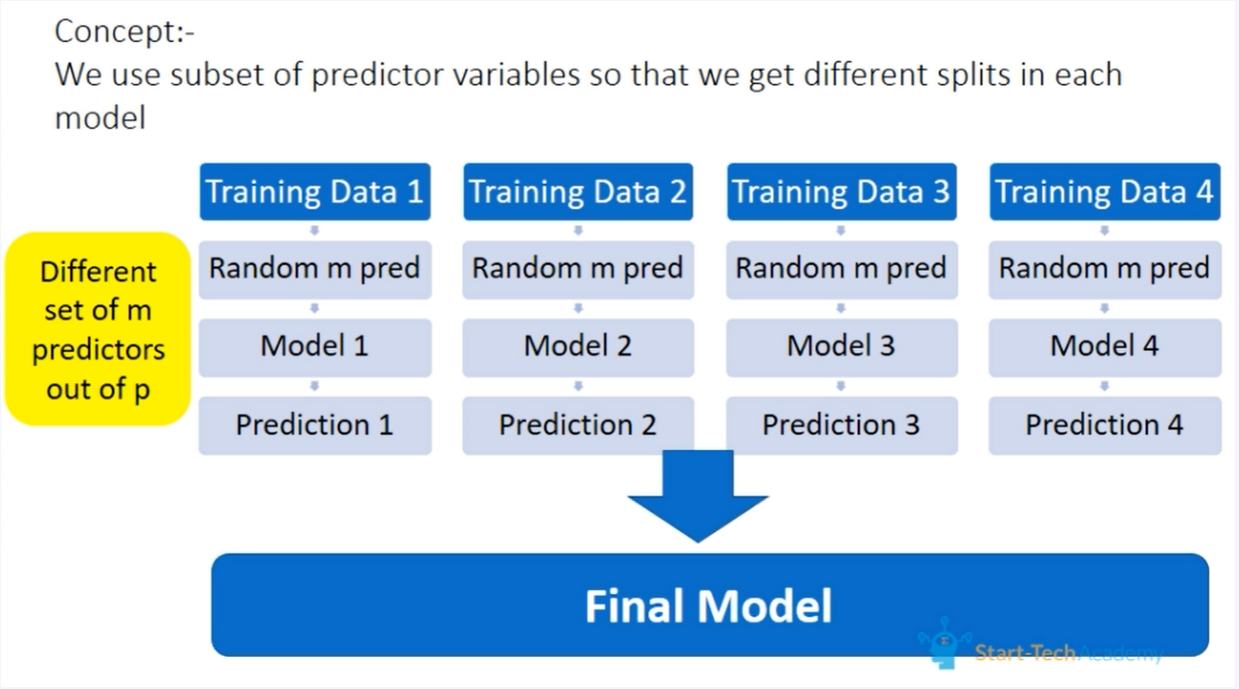


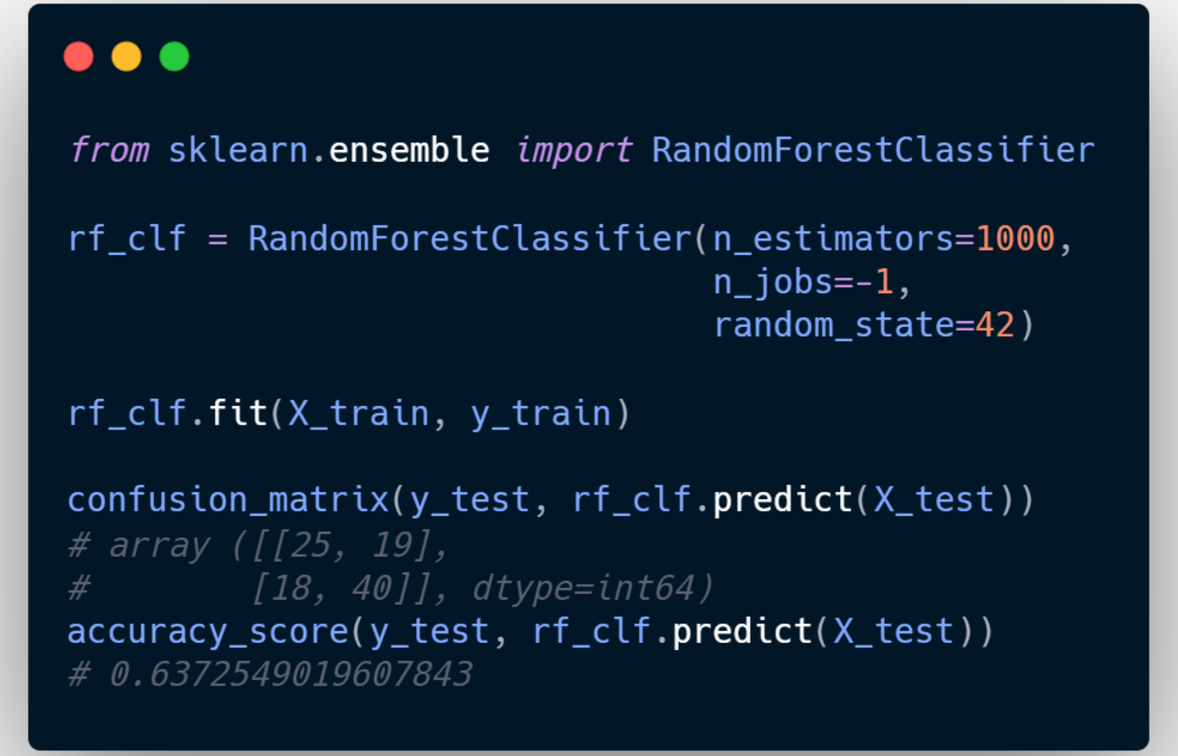
```
from sklearn import tree
from sklearn.ensemble import BaggingClassifier

clftree = tree.DecisionTreeClassifier() # for bagging, no params. Should use full tree

bag_clf = BaggingClassifier(base_estimator=clftree,
                            n_estimators=10000,
                            bootstrap=True,
                            n_jobs=-1,
                            random_state=42)

bag_clf.fit(X_train, y_train)

confusion_matrix(y_test, bag_clf.predict(X_test))
# array ([[27, 17],
#        [21, 37]], dtype=int64)
accuracy_score(y_test, bag_clf.predict(X_test))
# 0.6274509803921569
```

# Ensemble Techniques

# Random Forest

Bagging creates correlated trees. Therefore there isnt much reduction in variance. Solution : building a random forest



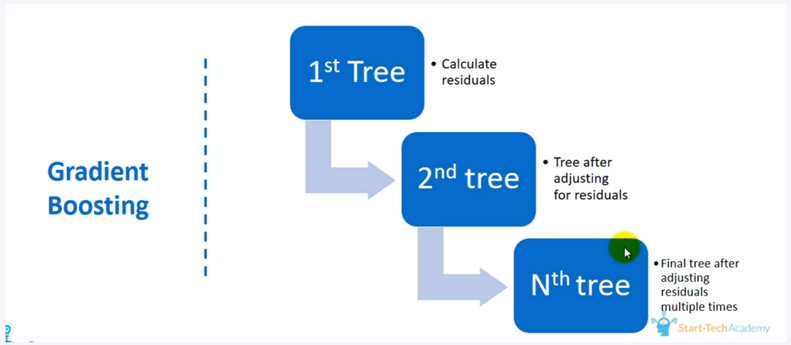


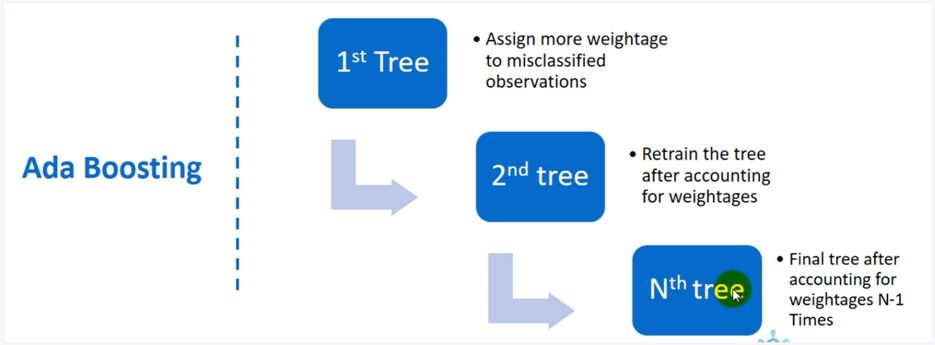
```
from sklearn.ensemble import RandomForestClassifier

rf_clf = RandomForestClassifier(n_estimators=1000,
                                n_jobs=-1,
                                random_state=42)

rf_clf.fit(X_train, y_train)

confusion_matrix(y_test, rf_clf.predict(X_test))
# array ([[25, 19],
#         [18, 40]], dtype=int64)
accuracy_score(y_test, rf_clf.predict(X_test))
# 0.6372549019607843
```

# Ensemble Techniques

# Boosting

In boosting, the trees are grown sequentially. That means each tree grows with the information from its previous tree.




## XG Boost

- Almost similar to Gradient Boost
- XG-boost used a more regularized model formalization to control over-fitting, which gives it better performance.
- For model, it might be more suitable to be called as regularized gradient boosting.

**Regularization**

The cost function we are trying to optimize (MSE in regression etc) also contains a penalty term for number of variables. In a way, we want to minimize the number of variables in final model along with the MSE or accuracy. This helps in avoiding overfitting

XG-Boost contains regularization terms in the cost function.

# Gradient Boosting

```python
from sklearn.ensemble import GradientBoostingClassifier

rgbc_clf = GradientBoostingClassifier()
gbc_clf.fit(X_train, y_train)

accuracy_score(y_test, gbc_clf.predict(X_test))
# 0.5882352941176471

gbc_clf2 = GradientBoostingClassifier(learning_rate=0.02,
                                      n_estimators=1000,
                                      max_depth=1)
gbc_clf2.fit(X_train, y_train)

accuracy_score(y_test, gbc_clf2.predict(X_test))
# 0.6176470588235294
```

# Ada Boosting

```
from sklearn.ensemble import AdaBoostClassifier

ada_clf = AdaBoostClassifier(learning_rate=0.02,
                             n_estimators = 5000)

ada_clf.fit(X_train, y_train)

accuracy_score(y_test, ada_clf.predict(X_test))
# 0.6274509803921569

ada_clf2 = AdaBoostClassifier(rf_clf,
                              learning_rate=0.05,
                              n_estimators=500)

ada_clf2.fit(X_train, y_train)

accuracy_score(y_test, ada_clf2.predict(X_test))
# 0.6176470588235294
```

# XG Boost

```python
import xgboost as xgb

xgb_clf = xgb.XGBClassifier(max_depth = 5,
                            n_estimators=10000,
                            learning_rate=0.3,
                            n_jobs = -1)

xgb_clf.fit(X_train, y_train)

accuracy_score(y_test, xgb_clf.predict(X_test))
# 0.6666666666666666
```

udemy

Certificate no: UC-2e189e21-f164-442c-8f94-0eb4f0cba879
Certificate url: ude.my/UC-2e189e21-f164-442c-8f94-0eb4f0cba879
Reference Number: 0004

CERTIFICATE OF COMPLETION

# Machine Learning & Deep Learning in Python & R

Instructors Start-Tech Academy

## Abhijith Udayakumar

Date Oct. 23, 2021
Length 35 total hours