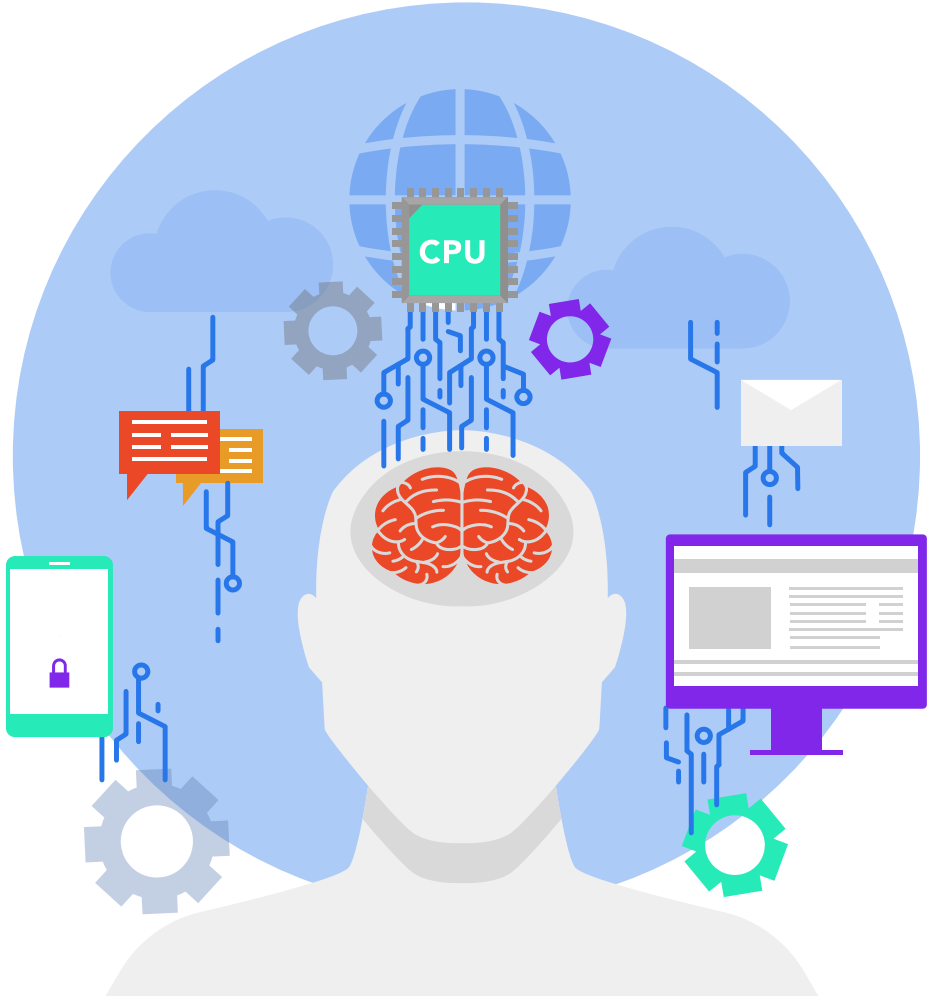
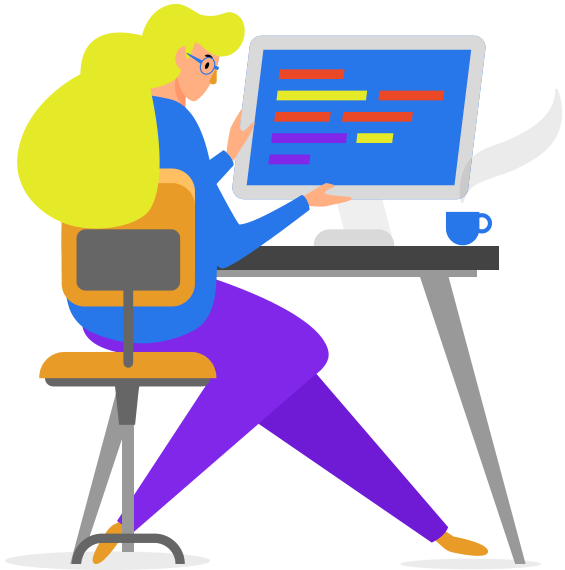


Data Science Machine Learning Python course

Thang Nguyen
Viet Nguyen

http://www.viet-it.com

Nội dung



01 **Machine Learning**
Tổng quan về Machine Learning

02 **Supervised Learning**
Các mô hình cơ bản

03 **Unsupervised Learning**
Các mô hình cơ bản

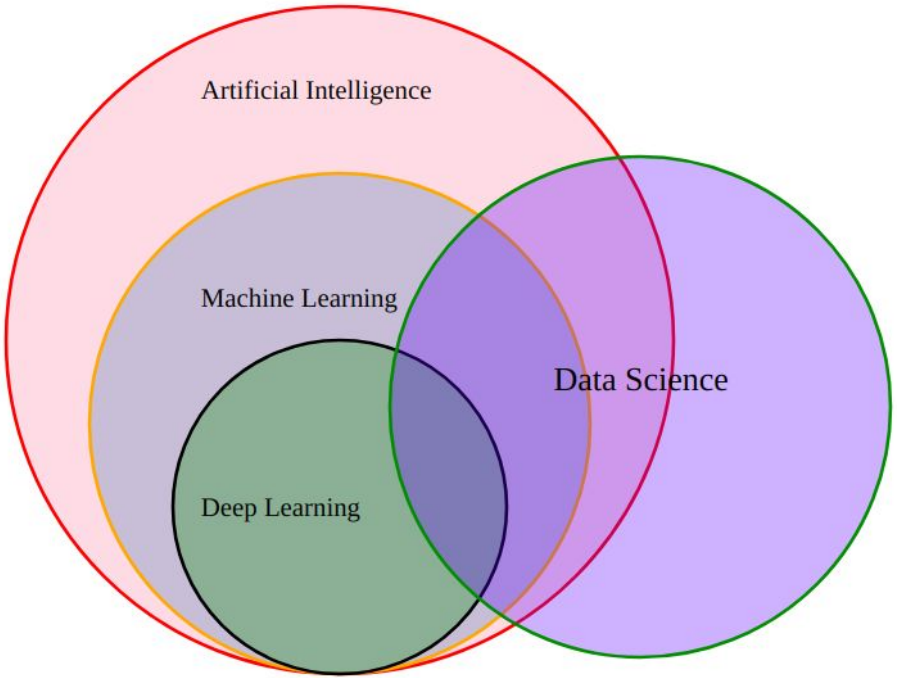
04 **Data Visualization**
Trực quan hóa dữ liệu

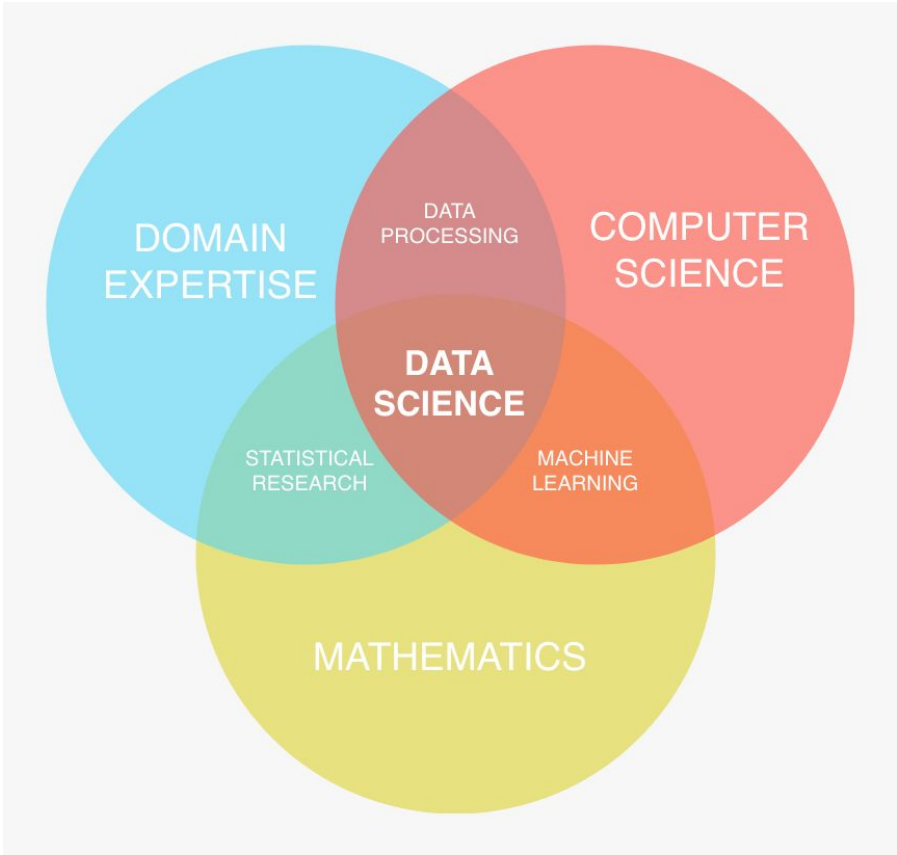
05 **Model evaluation**
Đánh giá mô hình

06 **Projects**
Xây dựng mô hình với các datasets

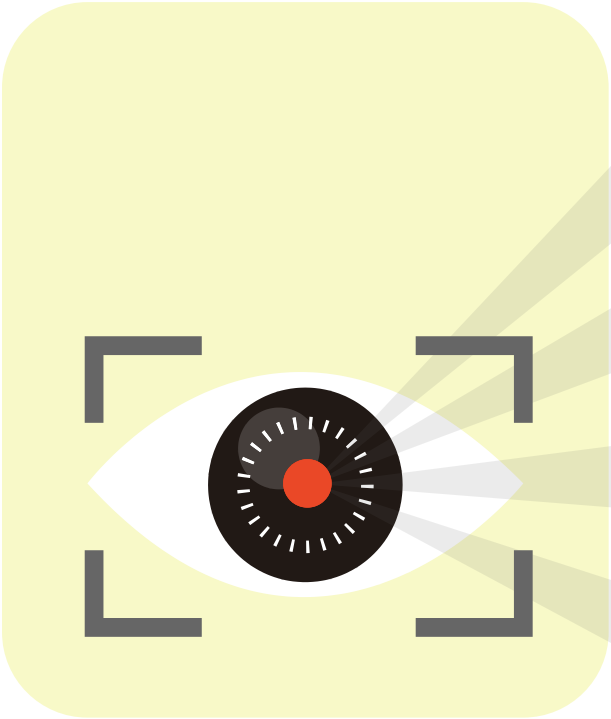
Tổng quan về Machine Learning

Tổng quan về Machine Learning và Data Science





Tổng quan các khái niệm



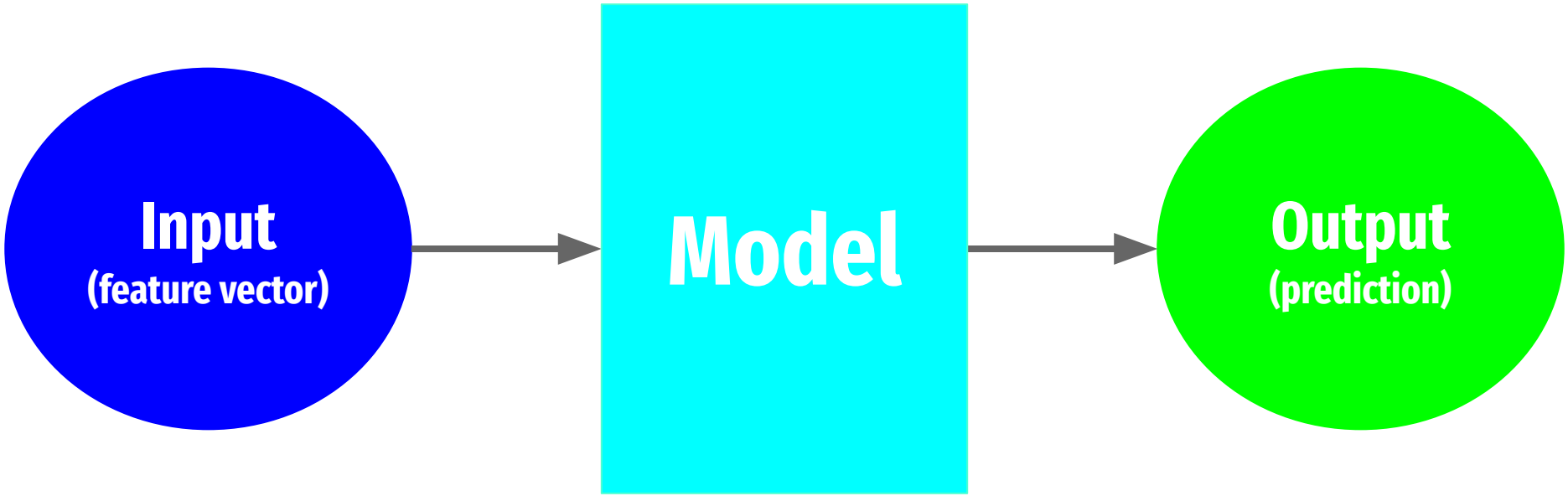
AI — Đưa ra hành động từ dữ liệu

Machine Learning — Học hỏi từ dữ liệu

Deep Learning — Học hỏi từ RẤT NHIỀU dữ liệu

Data Science — Tạo ra giá trị từ dữ liệu

Machine Learning



Dataset (Bộ dữ liệu)

Adult dataset

age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	40	United-States	<=50K
82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	United-States	<=50K
66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	40	United-States	<=50K
54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	United-States	<=50K
41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	United-States	<=50K
34	Private	216864	HS-grad	9	Divorced	Other-service	Unmarried	White	Female	0	3770	45	United-States	<=50K
38	Private	150601	10th	6	Separated	Adm-clerical	Unmarried	White	Male	0	3770	40	United-States	<=50K
74	State-gov	88638	Doctorate	16	Never-married	Prof-specialty	Other-relative	White	Female	0	3683	20	United-States	>50K
68	Federal-gov	422013	HS-grad	9	Divorced	Prof-specialty	Not-in-family	White	Female	0	3683	40	United-States	<=50K
41	Private	70037	Some-college	10	Never-married	Craft-repair	Unmarried	White	Male	0	3004	60	?	>50K
45	Private	172274	Doctorate	16	Divorced	Prof-specialty	Unmarried	Black	Female	0	3004	35	United-States	>50K
38	Self-emp-not-inc	164526	Prof-school	15	Never-married	Prof-specialty	Not-in-family	White	Male	0	2824	45	United-States	>50K
52	Private	129177	Bachelors	13	Widowed	Other-service	Not-in-family	White	Female	0	2824	20	United-States	>50K
32	Private	136204	Masters	14	Separated	Exec-managerial	Not-in-family	White	Male	0	2824	55	United-States	>50K
51	?	172175	Doctorate	16	Never-married	?	Not-in-family	White	Male	0	2824	40	United-States	>50K
46	Private	45363	Prof-school	15	Divorced	Prof-specialty	Not-in-family	White	Male	0	2824	40	United-States	>50K
45	Private	172822	11th	7	Divorced	Transport-moving	Not-in-family	White	Male	0	2824	76	United-States	>50K
57	Private	317847	Masters	14	Divorced	Exec-managerial	Not-in-family	White	Male	0	2824	50	United-States	>50K
22	Private	119592	Assoc-acdm	12	Never-married	Handlers-cleaners	Not-in-family	Black	Male	0	2824	40	?	>50K
34	Private	203034	Bachelors	13	Separated	Sales	Not-in-family	White	Male	0	2824	50	United-States	>50K
37	Private	188774	Bachelors	13	Never-married	Exec-managerial	Not-in-family	White	Male	0	2824	40	United-States	>50K

Feature (thuộc tính)

Numerical (số)

Integer (số nguyên)

- 1, 2, 3
- -2, -1, 0, 1, 2

Float (số thực)

- 0.1, 0.2, 0.3

Categorical (phân loại)

Nominal (định danh)

- Red, green, blue
- US, Vietnam, Thailand

Ordinal (thứ bậc)

- Happy, normal, sad
- XS, S, M, L, XL

Boolean (logic)

- True, False
- Yes, No

Các phương thức học trong Machine Learning

Supervised learning

Classification

- Fraud detection
- Email spam detection
- Diagnostics
- Image classification

Regression

- Risk assessment
- Score prediction

Unsupervised learning

Clustering

- Customer clustering
- Data visualization

Association

- Recommendation System

Reinforcement learning

- Game theory
- Autonomous cars

Supervised learning

Labeled data (dữ liệu có nhãn)

Diabetes dataset

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

Labeled data (dữ liệu có nhãn)

Diabetes dataset

sample

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

Labeled data (dữ liệu có nhãn)

Diabetes dataset

feature

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

Labeled data (dữ liệu có nhãn)

Diabetes dataset

label (target)

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

Labeled data (dữ liệu có nhãn)

Diabetes dataset

feature vector

corresponding label (target)

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

Labeled data (dữ liệu có nhãn)

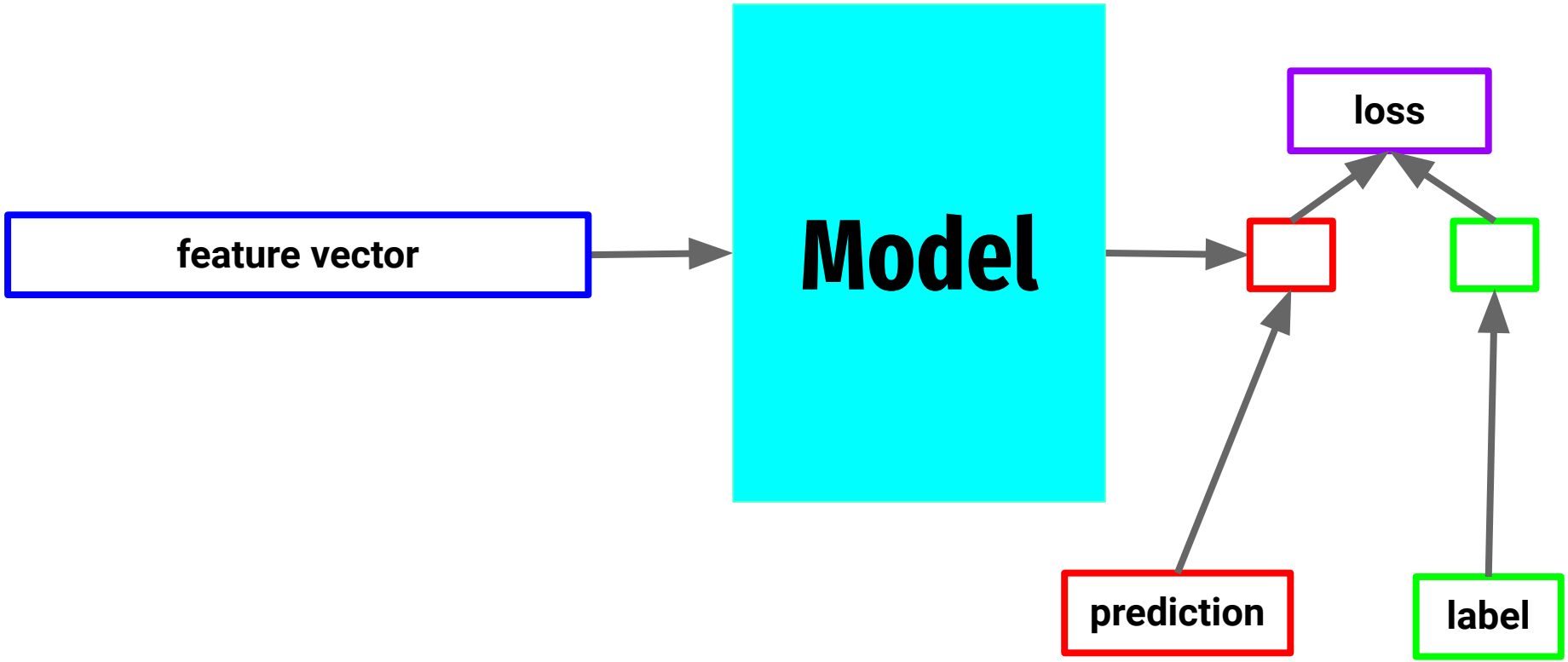
Label/target vector (y)

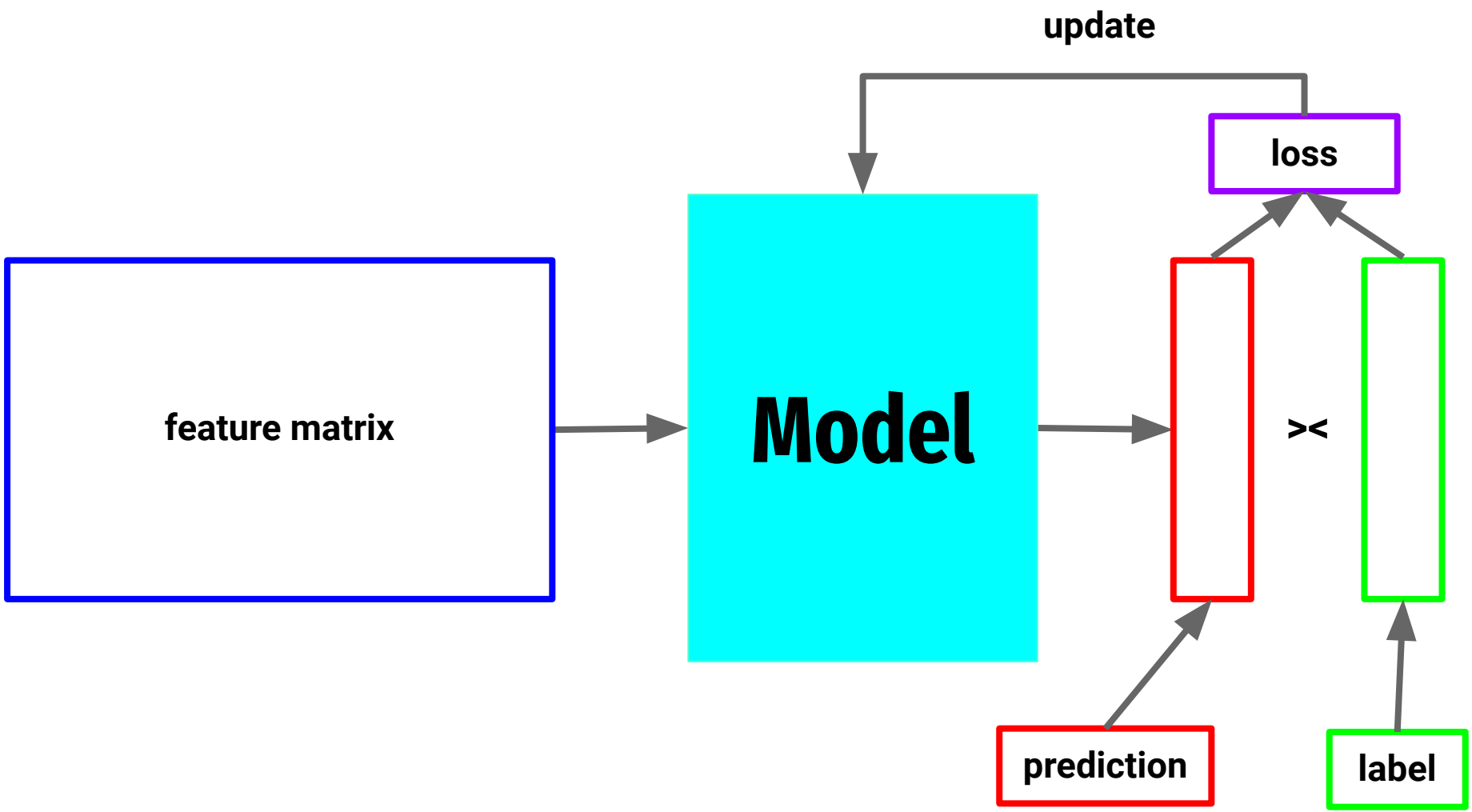
feature matrix (X)

Diabetes dataset

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

Model training (Huấn luyện mô hình)

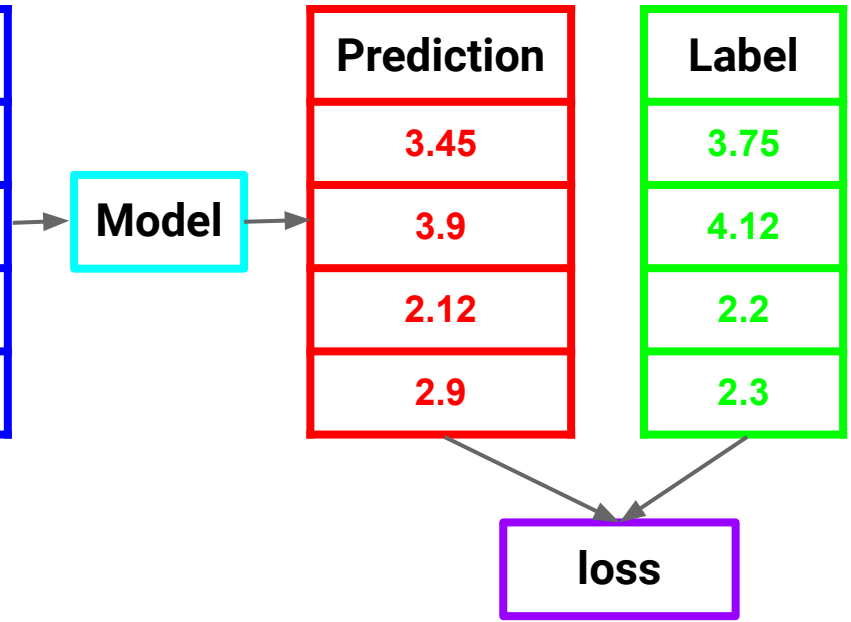



update
loss
feature matrix
Model
><
prediction
label

Loss function (Hàm mất mát)

Ví dụ về mô hình dự đoán giá nhà ở Việt Nam (Đơn vị: Tỉ VND)

Diện tích	Số tầng	Thành phố	Trong ngõ
40	6	HCM	No
60	4	Hà Nội	No
120	3	Hải Dương	Yes
50	4	Huế	No



Loss function (Hàm mất mát)

L1 loss

$$Loss = \Sigma |y_{prediction} - y_{actual}|$$

Các tên gọi:

- Least Absolute Deviations (LAD)
- Absolute Error
- Mean of these Absolute Errors

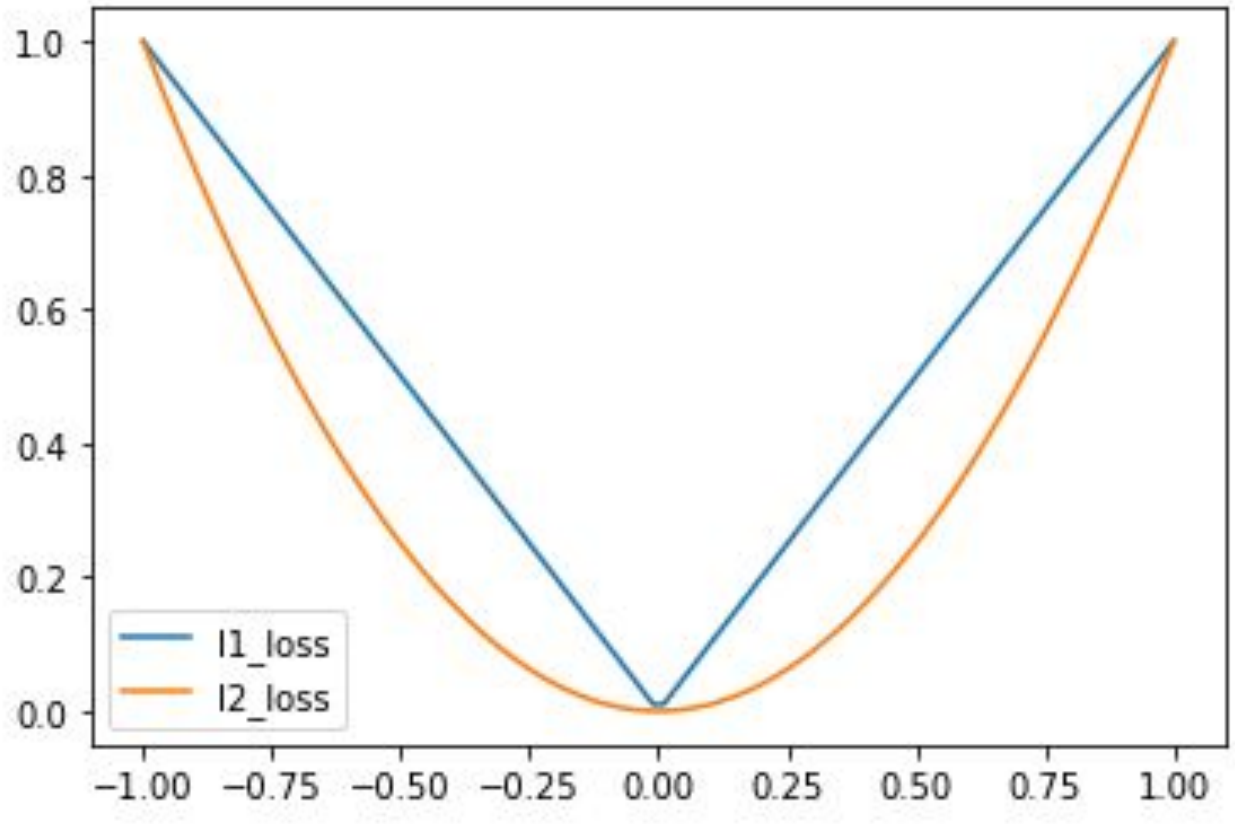
L2 loss

$$Loss = \Sigma (y_{prediction} - y_{actual})^2$$

Các tên gọi:

- Least Square Errors (LS)
- Squared Error
- Mean of these Squared Errors

Loss function (Hàm mất mát)



Phân chia dataset

Diabetes dataset

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

TRAINING SET

VALIDATION SET

TEST SET

Các bài toán trong Supervised Learning

Classification (phân loại)

Binary Classification

- Fraud detection
- Email spam detection
- Diagnostics

Multiclass Classification

- Animals (cat/dog/horse)
- Fruits (apple/orange/lemon)
- Flowers (rose/daisy/sunflower)

Regression(hồi quy)

- House price estimation
- Stock prediction
- Temperature forecast

Classification

"Is this email spam or not spam?"

To: daniel@mrdbourke.com
Hey Daniel,

This deep learning course is incredible!
I can't wait to use what I've learned!

Not spam

To: daniel@mrdbourke.com
Hay daniel...

C0ongratu1ations! U win $1139239230

Spam

Binary classification

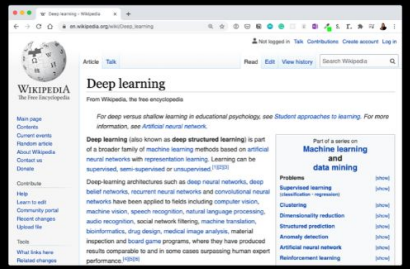
(one thing or another)

"Is this a photo of sushi, steak or pizza?"

Multiclass classification

(more than one thing or another)

"What tags should this article have?"



Machine learning
Representation learning
Artificial intelligence

(multiple label options per sample)

Multilabel classification

Cách đánh giá mô hình trong Supervised Learning

Classification: dog vs cat

model

dog	cat	dog	dog	cat	cat	dog

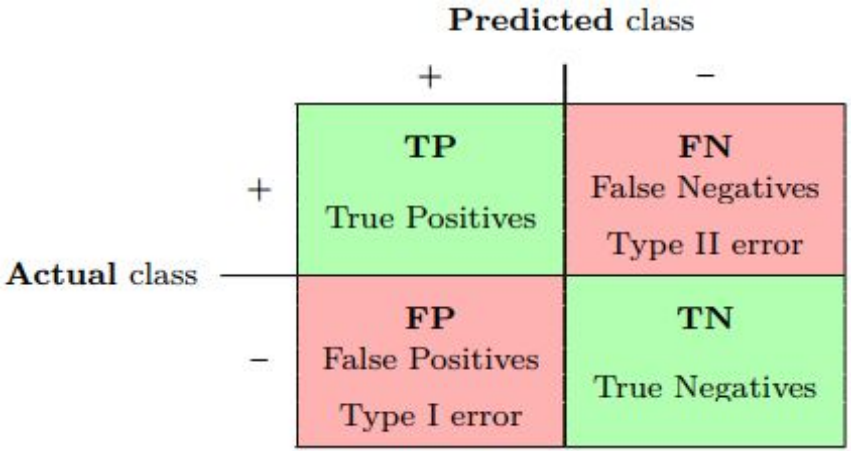
dog	dog	cat	dog	cat	dog	dog

label

Classification metrics

dog	cat	dog	dog	cat	cat	dog

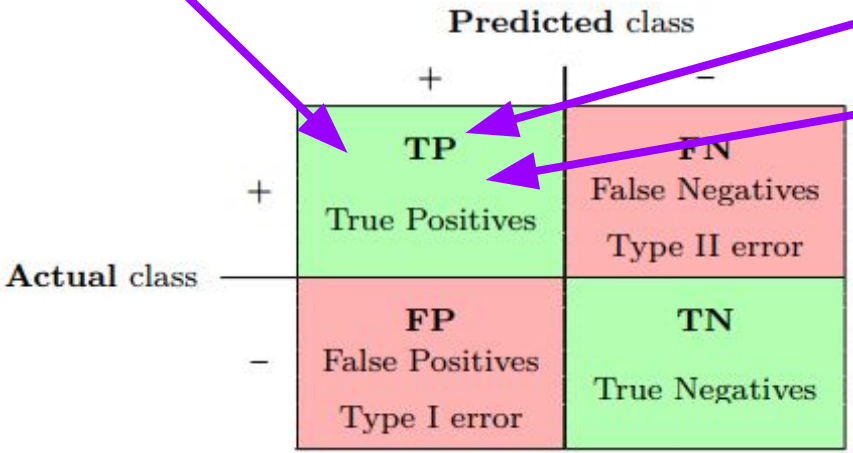
dog	dog	cat	dog	cat	dog	dog



Class +: Class ta quan tâm: e.g. dog
Class -: class(es) còn lại: e.g. cat

Classification metrics

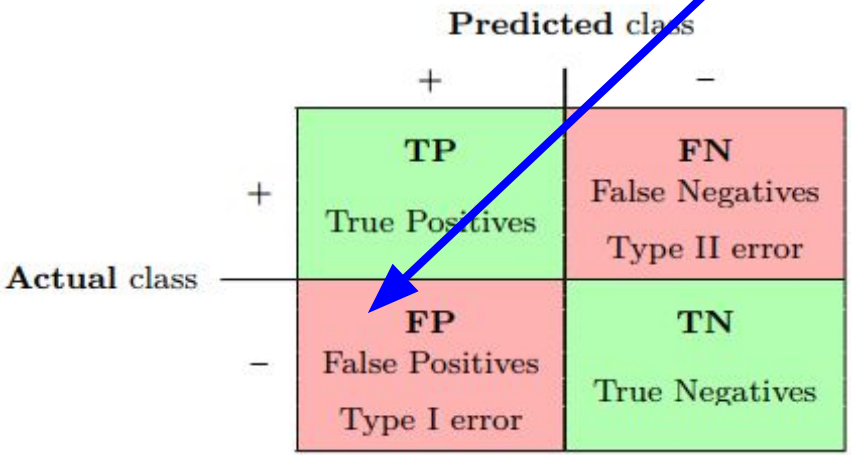
dog	cat	dog	dog	cat	cat	dog
dog	dog	cat	dog	cat	dog	dog



Số lượng TP: 3

Classification metrics

dog	cat	dog	dog	cat	cat	dog
dog	dog	cat	dog	cat	dog	dog

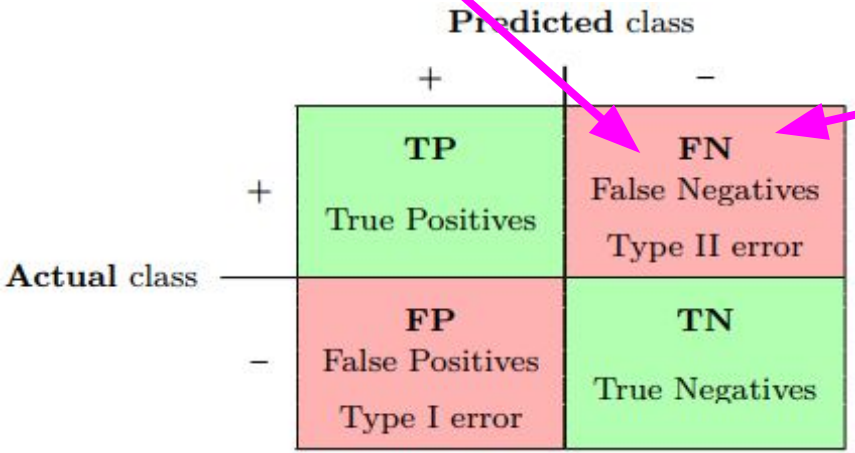


Số lượng TP: 3

Số lượng FP: 1

Classification metrics

dog	cat	dog	dog	cat	cat	dog
dog	dog	cat	dog	cat	dog	dog



Số lượng TP: 3

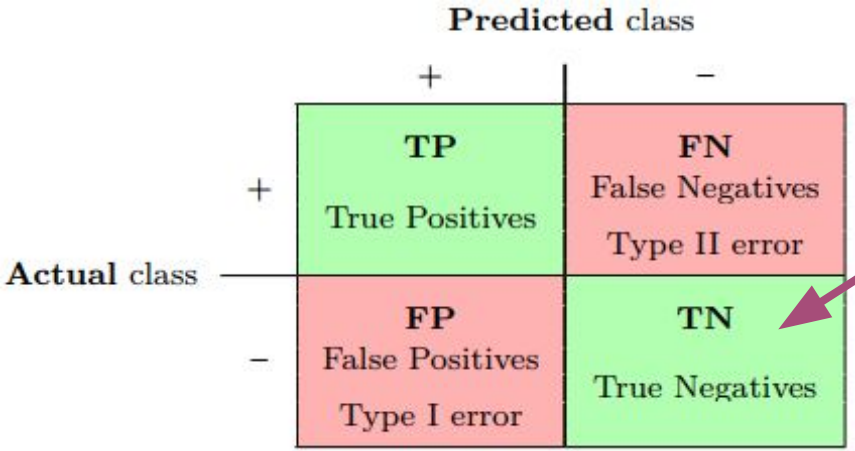
Số lượng FP: 1

Số lượng FN: 2

Classification metrics

dog	cat	dog	dog	cat	cat	dog

dog	dog	cat	dog	cat	dog	dog



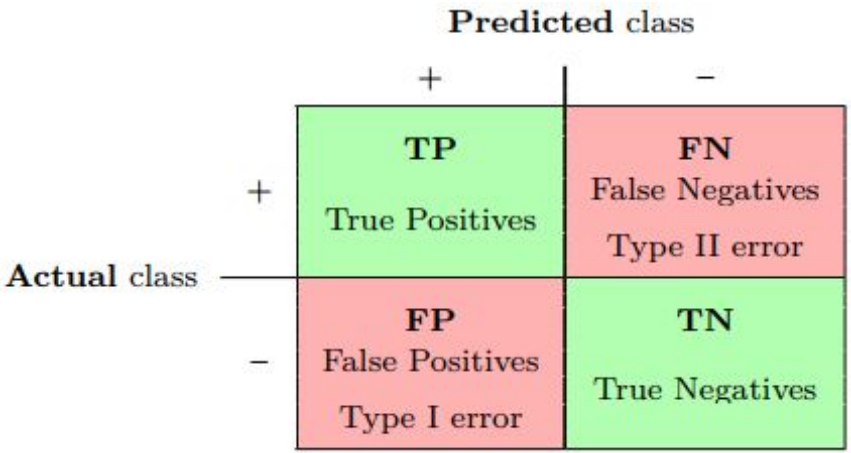
Số lượng TP: 3

Số lượng FP: 1

Số lượng FN: 2

Số lượng TN: 1

Metric	Công thức	Ý nghĩa
Accuracy	(TP+TN)/All	Độ chính xác tổng quát



Số lượng TP: 3

Số lượng FP: 1

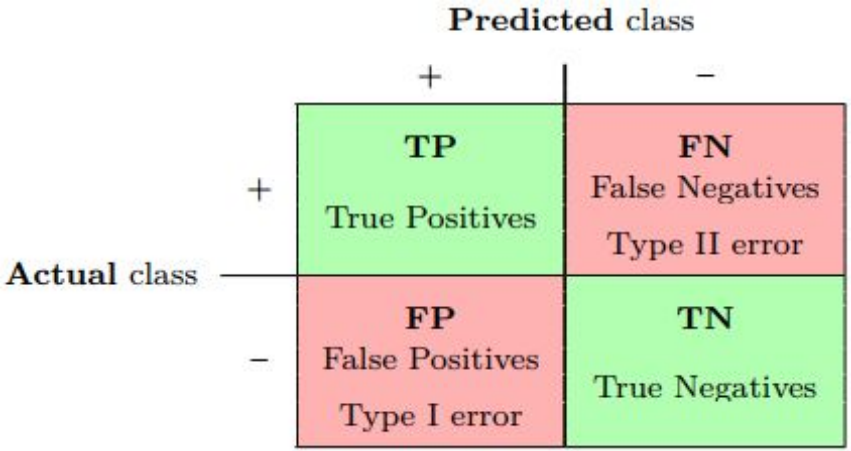
Số lượng FN: 2

Số lượng TN: 1

All: 7

Accuracy = (3+1)/7

Metric	Công thức	Ý nghĩa
Accuracy	(TP+TN)/All	Độ chính xác tổng quát
Precision	TP/(TP+FP)	Độ chính xác đối với class +



Số lượng TP: 3

Số lượng FP: 1

Số lượng FN: 2

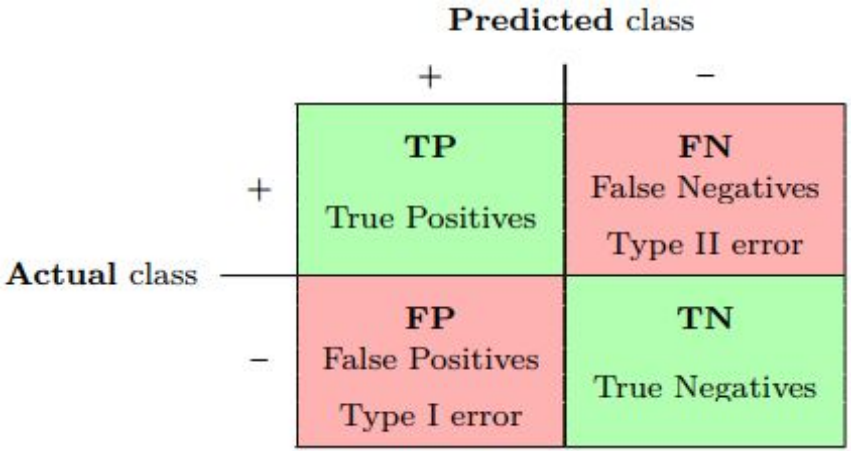
Số lượng TN: 1

All: 7

Accuracy = (3+1)/7

Precision = 3/(3+1)

Metric	Công thức	Ý nghĩa
Accuracy	(TP+TN)/All	Độ chính xác tổng quát
Precision	TP/(TP+FP)	Độ chính xác đối với class +
Recall	TP/(TP+FN)	Độ bao phủ đối với các dự đoán về class +



Số lượng TP: 3

Số lượng FP: 1

Số lượng FN: 2

Số lượng TN: 1

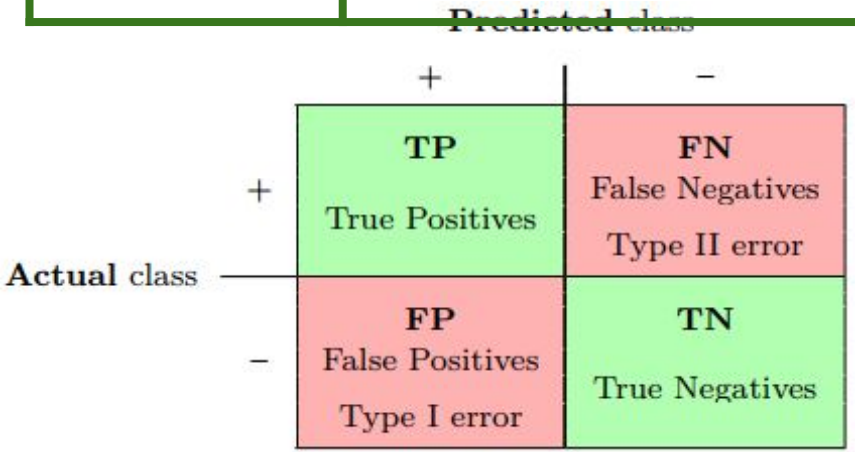
All: 7

Accuracy = (3+1)/7

Precision = 3/(3+1)

Recall = 3/(3+2)

Metric	Công thức	Ý nghĩa
Accuracy	(TP+TN)/All	Độ chính xác tổng quát
Precision	TP/(TP+FP)	Độ chính xác đối với class +
Recall	TP/(TP+FN)	Độ bao phủ đối với các dự đoán về class +
F1 score	(2*Pre*Re)/(Pre+Re): Trung bình điều hòa giữa Precision và Recall	



Số lượng TP: 3

Số lượng FP: 1

Số lượng FN: 2

Số lượng TN: 1

All: 7

Accuracy = (3+1)/7

Precision = 3/(3+1)

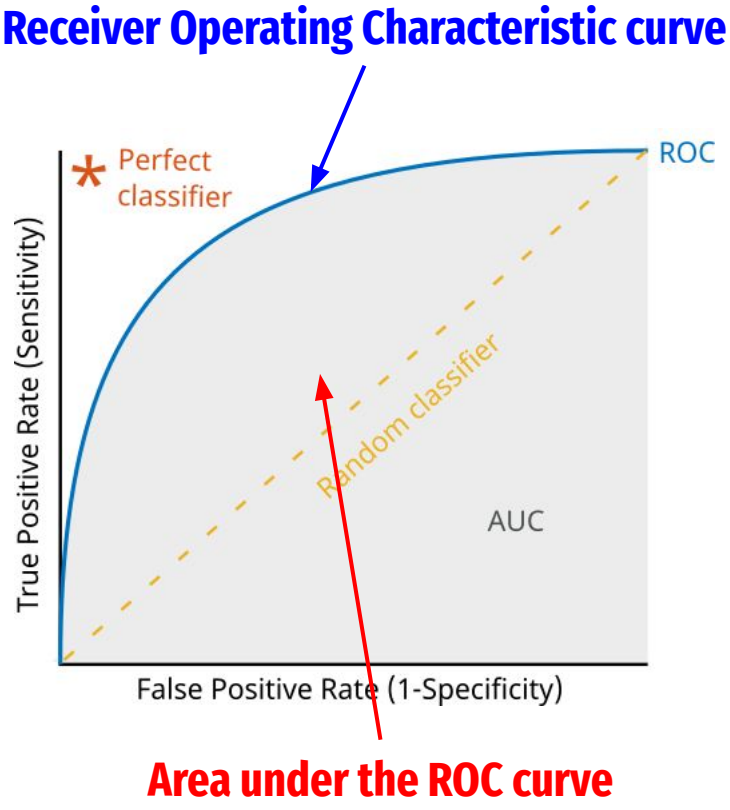
Recall = 3/(3+2)

F1 score = (2*0.75*0.6)/(0.75+0.6)

Classification metrics

		Predicted class: +	Predicted class: −
Actual class	+	**TP** True Positives	**FN** False Negatives Type II error
	−	**FP** False Positives Type I error	**TN** True Negatives

Metric	Công thức
True Positive Rate (= Recall)	TP/(TP+FN)
False Positive Rate	FP/(FP+TN)



Cách đánh giá mô hình trong Supervised Learning

Regression: House price prediction

Diện tích	Số tầng	Thành phố	Trong ngõ
40	6	HCM	No
60	4	Hà Nội	No
120	3	Hải Dương	Yes
50	4	Huế	No

Prediction	Label
3.45	3.75
3.9	4.12
2.12	2.2
2.9	2.3

$$\text{Mean absolute error} = \frac{1}{N}\sum_{1}^{N} |y_{prediction} - y_{actual}|$$

$$\text{Root Mean Squared Error (RMSE)} = \sqrt{\frac{1}{N}\sum_{1}^{N} (y_{prediction} - y_{actual})^2}$$

Machine Learning pipeline

Step 1 01

Data collection

Thu thập dữ liệu

Step 2 02

Statistics

Hiểu dữ liệu thông qua thống kê

Step 3 03

Data preprocessing

Tiền xử lý dữ liệu

Step 4 04

Data visualization

Trực quan hóa dữ liệu

Step 5 05

Model building

Xây dựng và huấn luyện mô hình

Step 6 06

Model evaluation

Đánh giá mô hình

Step 7 07

Model deployment

Triển khai mô hình

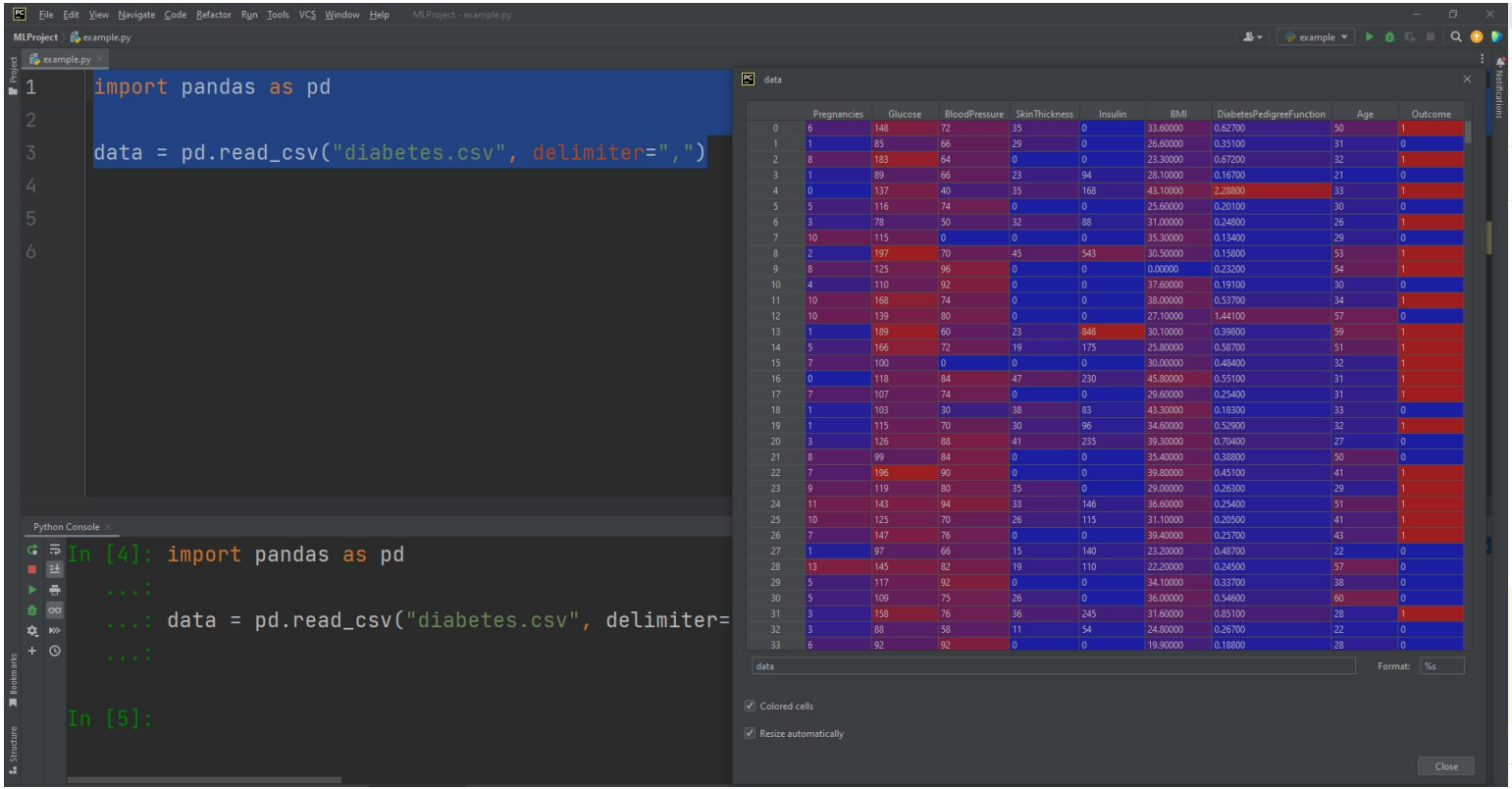
Step 2: Statistics

Diabetes dataset

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1

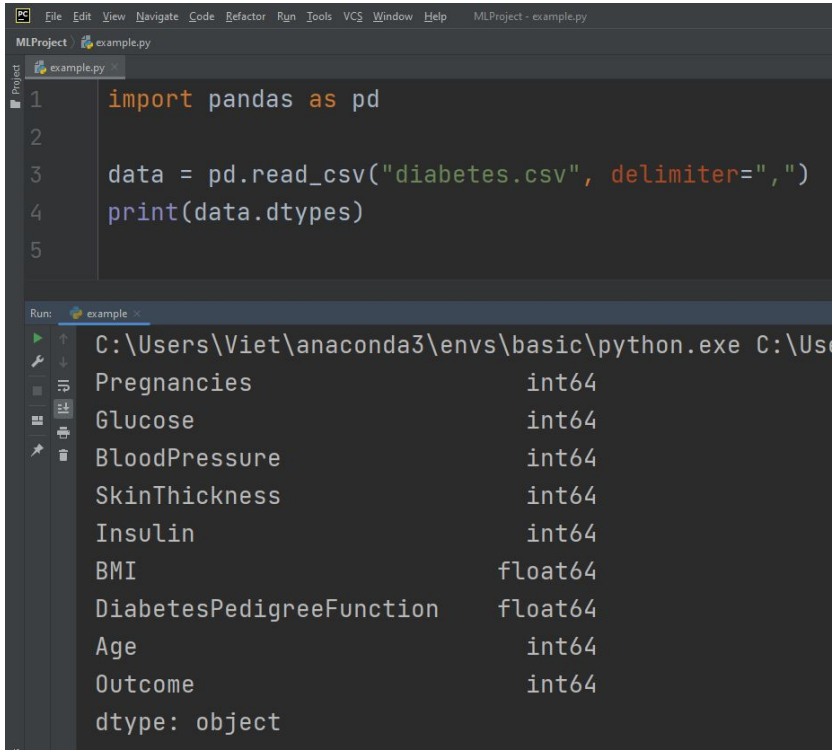
Step 2: Statistics

How data looks like



Step 2: Statistics

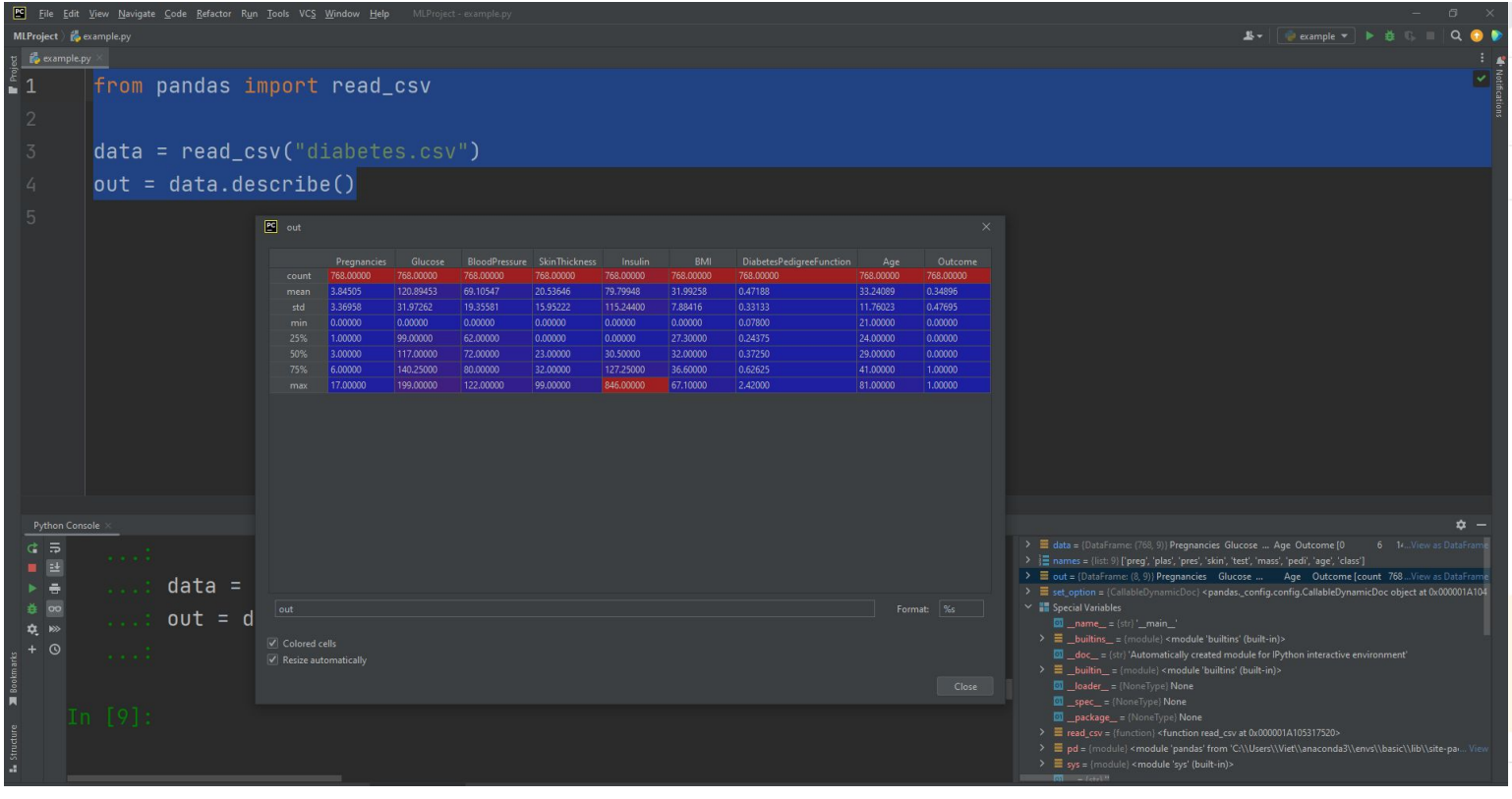
Check each feature’s data type



- Tất cả các features và target đều là numerical data
- Có 2 features là float, còn lại là int

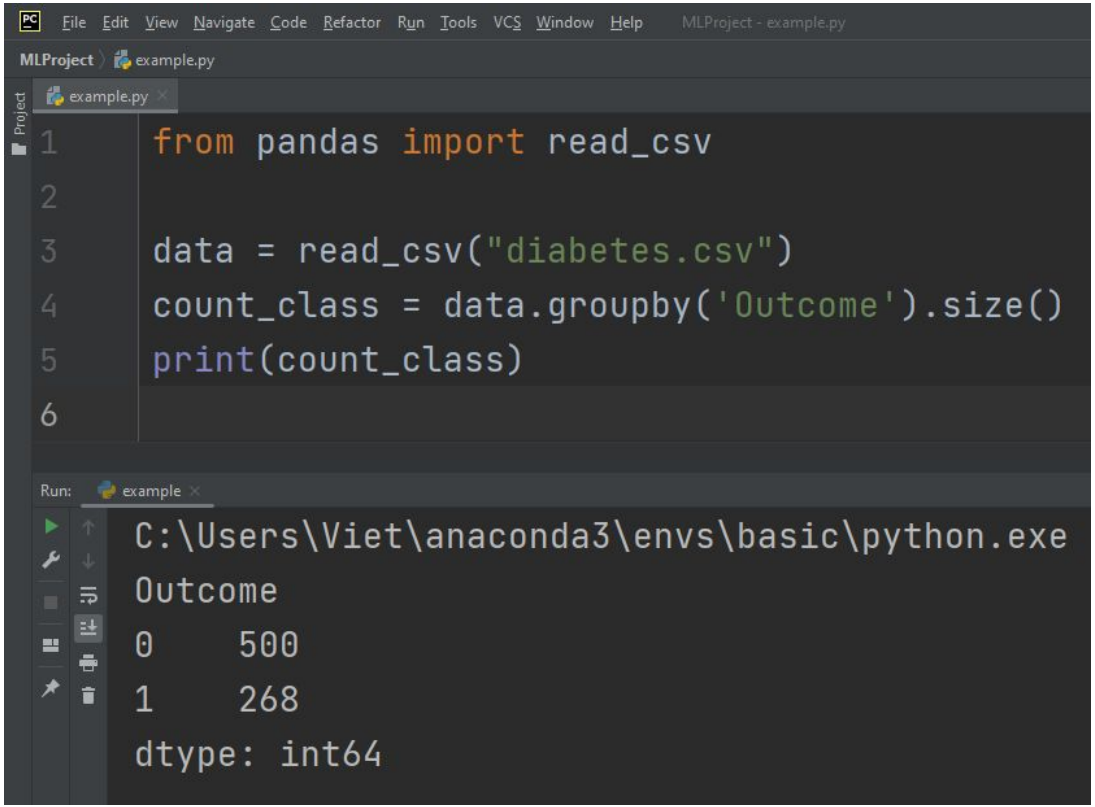
Step 2: Statistics

Statistical Summary of Data



Step 2: Statistics

Category distribution



Step 3.1: Handle missing or invalid values

Step 1 01

Handle missing or invalid value

Xử lý dữ liệu thiếu hoặc sai

Step 2 02

Remove outliers

Loại bỏ outliers

Step 3 03

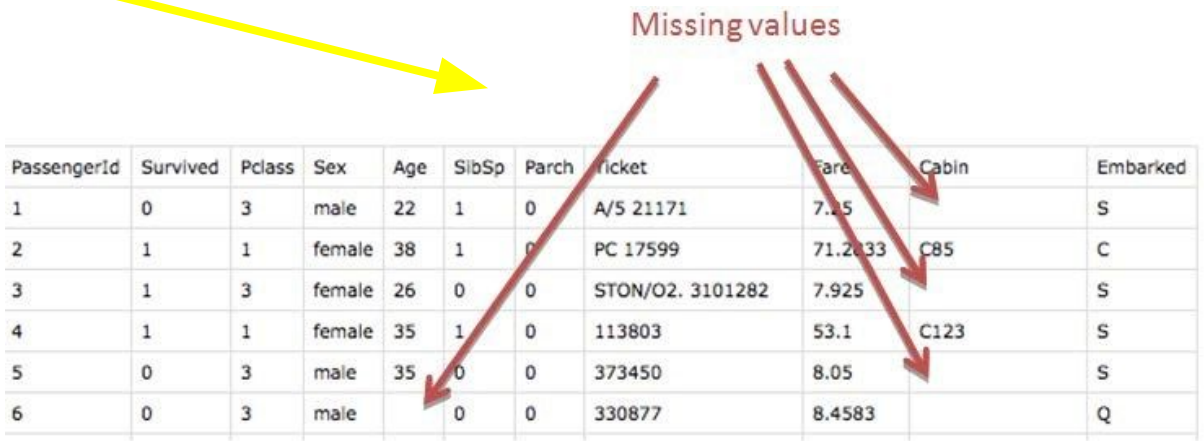
Data Transformation

Biến đổi dữ liệu

Step 4 04

Remove multicollinearity

Loại bỏ đa cộng tuyến



Step 3.2: Remove outliers

Step 1 01

Handle missing or invalid value

Xử lý dữ liệu thiếu hoặc sai

Step 2 02

Remove outliers

Loại bỏ outliers

Step 3 03

Data Transformation

Biến đổi dữ liệu

Step 4 04

Remove multicollinearity

Loại bỏ đa cộng tuyến

Step 3.3: Tiền xử lý numerical features

MinMaxScaler

```python
from pandas import read_csv
from sklearn.preprocessing import MinMaxScaler

data = read_csv("diabetes.csv")
scaler = MinMaxScaler(feature_range=(0, 1))
before = data[["Pregnancies"]].values
after = scaler.fit_transform(data[["Pregnancies"]])
for b, a in zip(before, after):
    print("Before: {}. After: {}".format(b, a))
```

```
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\
Before: [6]. After: [0.35294118]
Before: [1]. After: [0.05882353]
Before: [8]. After: [0.47058824]
Before: [1]. After: [0.05882353]
Before: [0]. After: [0.]
Before: [5]. After: [0.29411765]
Before: [3]. After: [0.17647059]
Before: [10]. After: [0.58823529]
Before: [2]. After: [0.11764706]
Before: [8]. After: [0.47058824]
```

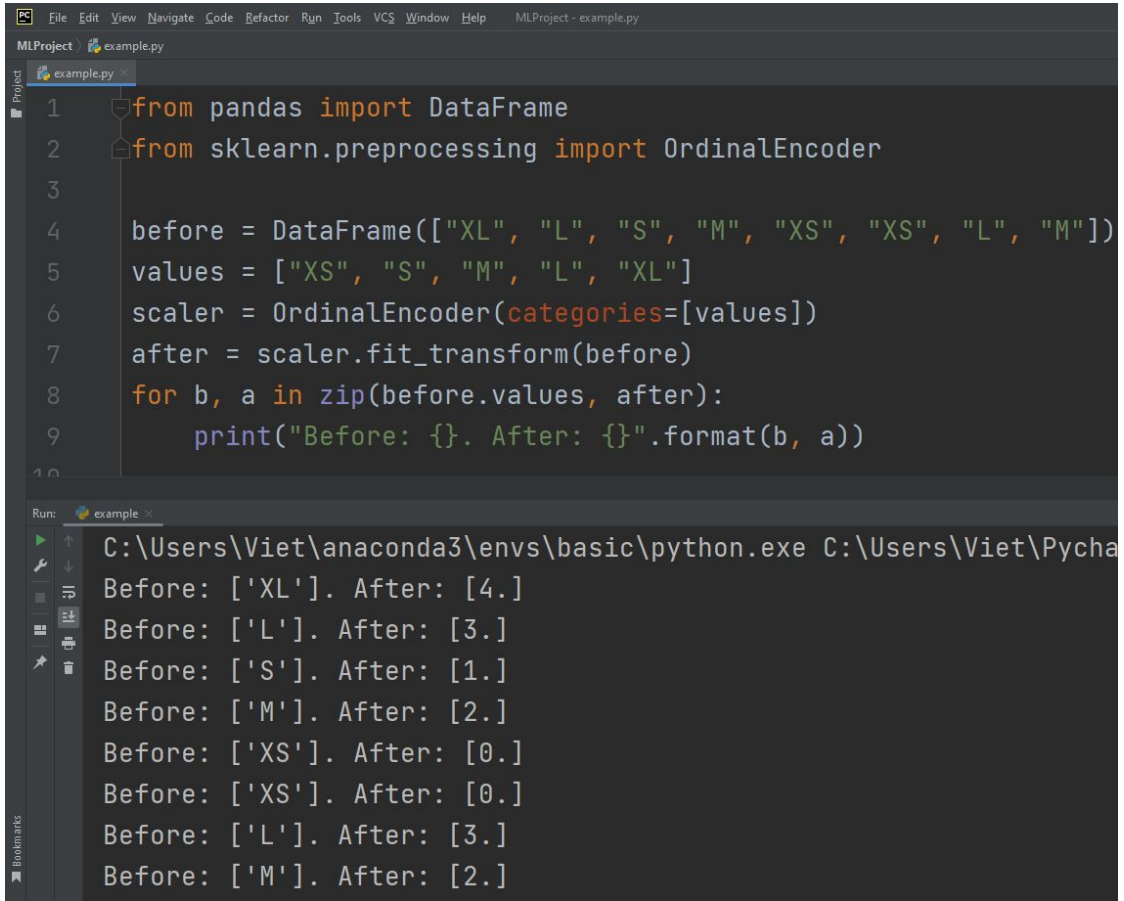
StandardScaler

```python
from pandas import read_csv
from sklearn.preprocessing import StandardScaler

data = read_csv("diabetes.csv")
scaler = StandardScaler()
before = data[["Pregnancies"]].values
after = scaler.fit_transform(data[["Pregnancies"]])
for b, a in zip(before, after):
    print("Before: {}. After: {}".format(b, a))
```

```
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\User
Before: [6]. After: [0.63994726]
Before: [1]. After: [-0.84488505]
Before: [8]. After: [1.23388019]
Before: [1]. After: [-0.84488505]
Before: [0]. After: [-1.14185152]
Before: [5]. After: [0.3429808]
Before: [3]. After: [-0.25095213]
Before: [10]. After: [1.82781311]
Before: [2]. After: [-0.54791859]
Before: [8]. After: [1.23388019]
```

Step 3.3: Tiền xử lý ordinal features

[OrdinalEncoder]



Step 3.3: Các cách mã hóa nominal features

One-hot encoding

Color

- Red -> **1**, 0, 0, 0
- Green - > 0, **1**, 0, 0
- Blue -> 0, 0, **1**, 0
- Black -> 0, 0, 0, **1**

Advantage

- Simple

Disadvantage

- Sparse data

Hash encoding

Country

- USA -> **1, 0, 0**
- UK -> 0, 0, 1
- Korea -> 0, 1, 0
- Russia -> **1, 0, 0**

Advantage

- Simple
- Less sparse

Disadvantage

- Collision
- No inverse-mapping

Word Embedding

Animal

- Dog -> 0.27,-0.31, -0.53
- Lion -> **-0.7, 0.61, 0.42**
- Tiger -> **-0.71, 0.6, 0.38**
- Mouse -> 0.31, -0.34, 0.76

Advantage

- Memory efficient
- Relationship learnt

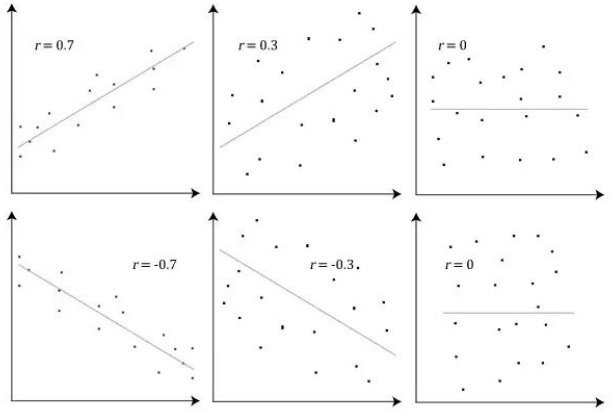
Disadvantage

- Word2vec model needed
- Maybe many dimensions needed

Step 3.4: Mối quan hệ giữa các variables (features, target)

Correlation

- **Mức độ 2 hay nhiều variables quan hệ tuyến tính với nhau**
- **Hệ số tương quan: [-1, 1]**
- **Correlation does not imply causation**



$$r_{xy} = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma(x_i - \bar{x})^2 \Sigma(y_i - \bar{y})^2}}$$

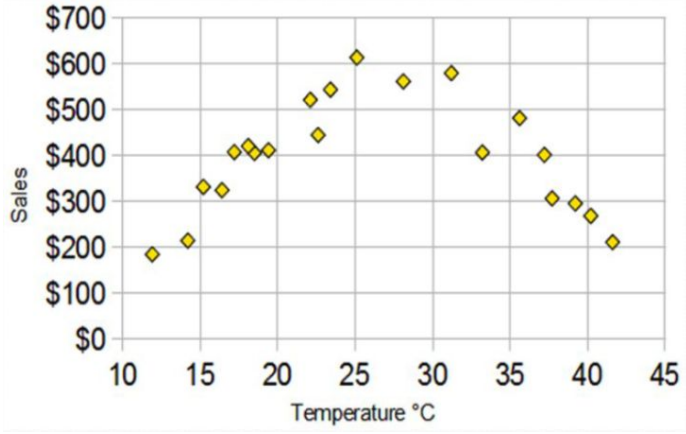
r_{xy} = correlation coefficient between x and y

x_i = the values of x within a sample

y_i = the values of y within a sample

$\bar{x}$ = the average of the values of x within a sample

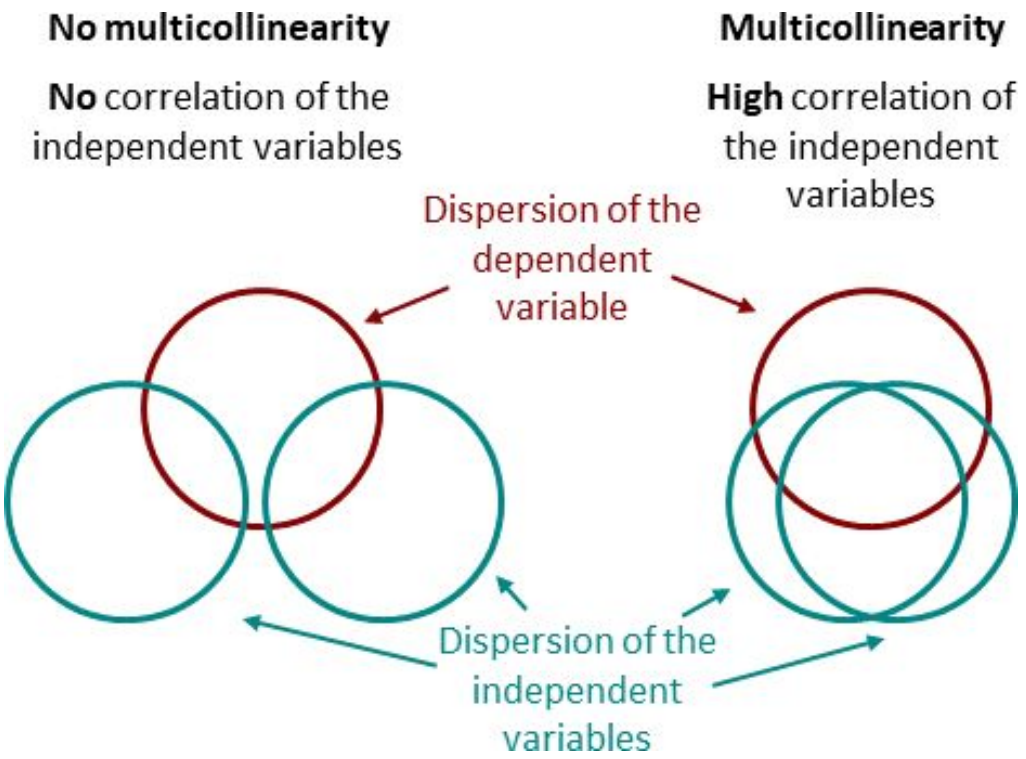
$\bar{y}$ = the average of the values of y within a sample



Step 3.4: Mối quan hệ giữa các features

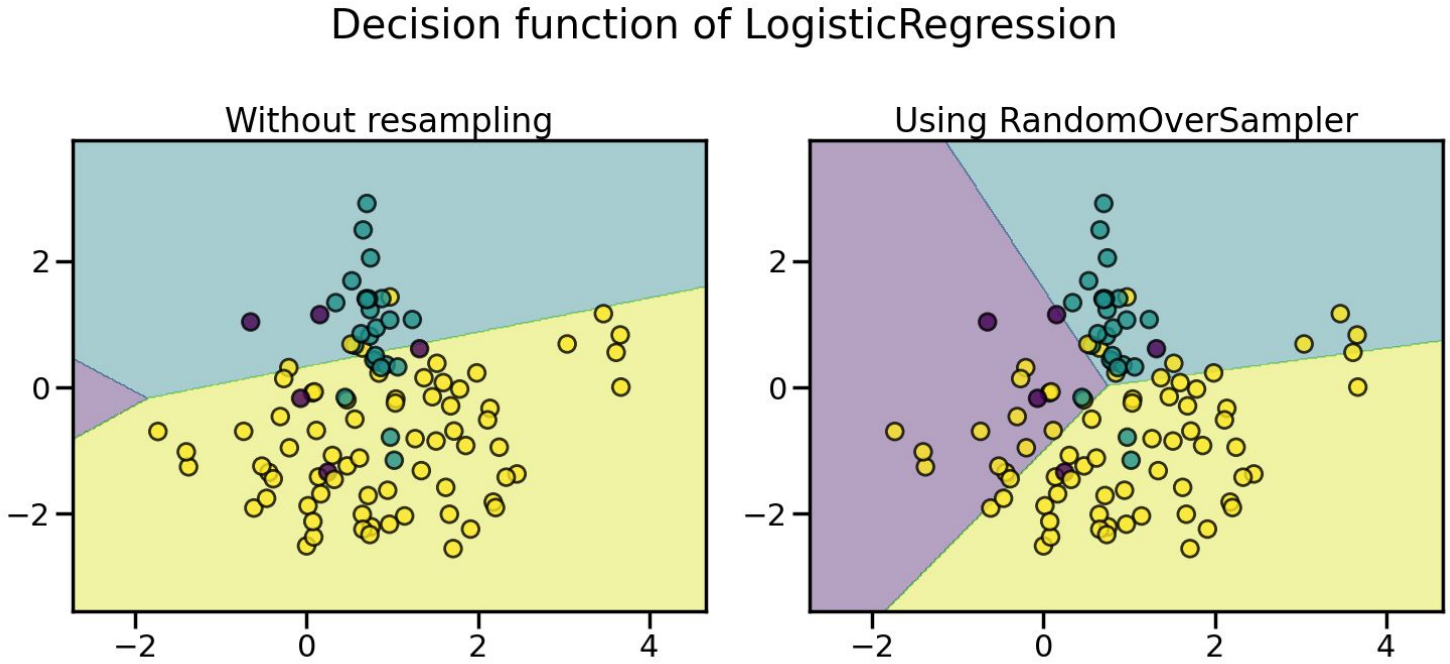
(Multi)collinearity

- Mức độ 2 hay nhiều features quan hệ tuyến tính với nhau
- Thông thường hệ số tương quan > 0.7 hoặc < -0.7 biểu thị rằng 2 hay nhiều features có hiện tượng (đa) cộng tuyến với nhau



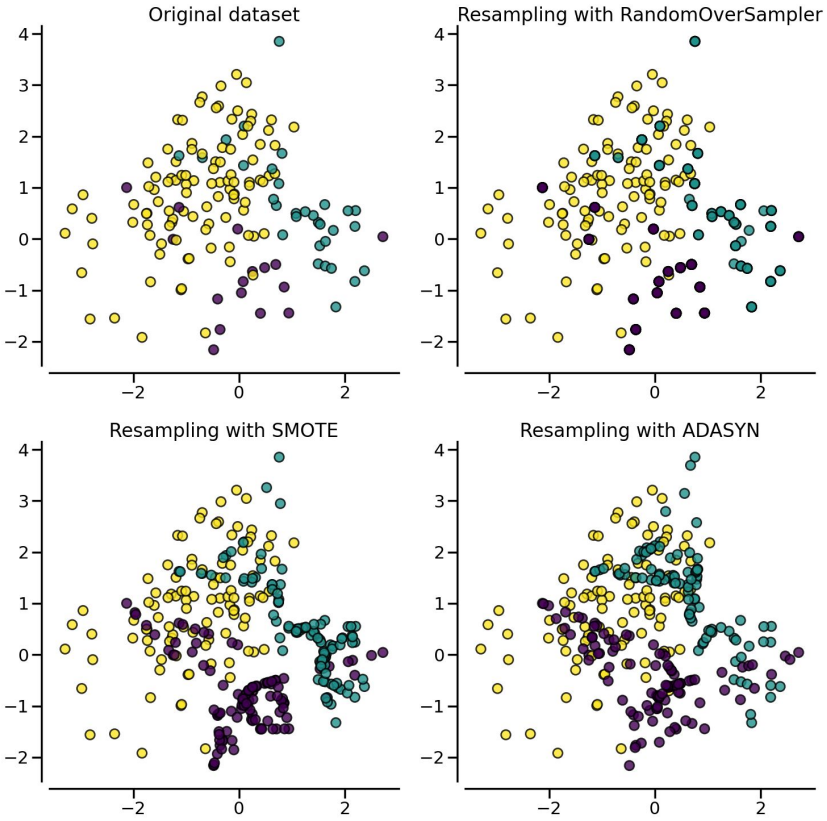
Step 3.5: Balance data

[Over-sampling]



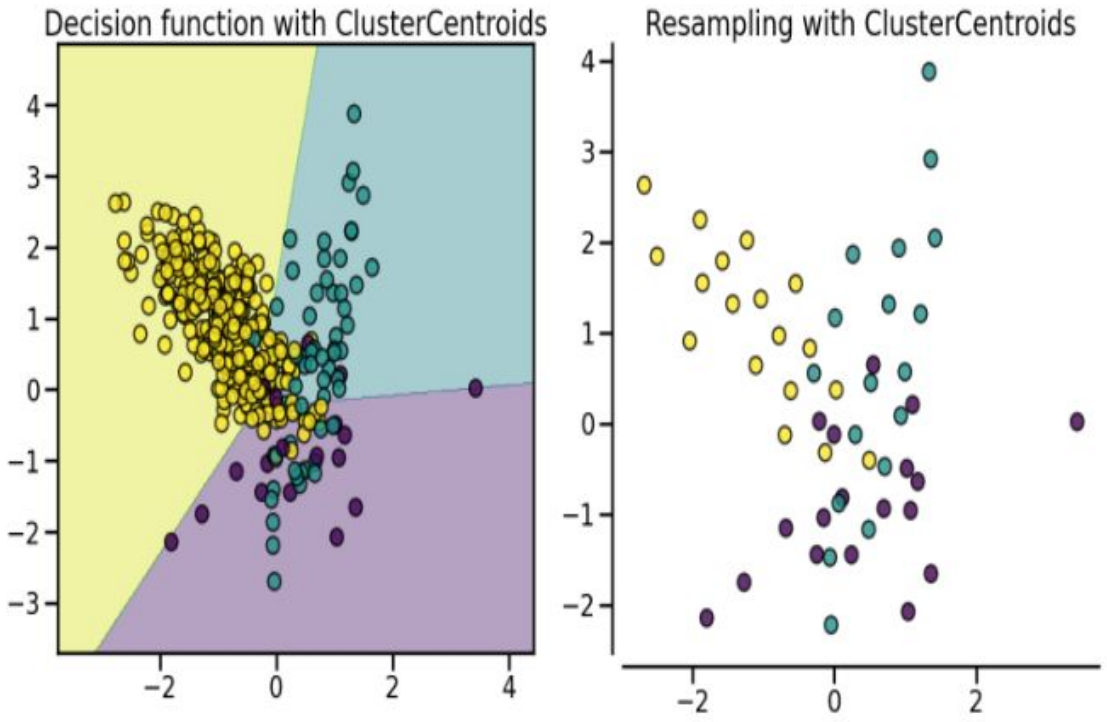
Step 3.5: Balance data

Over-sampling



Step 3.5: Balance data

[Under-sampling]

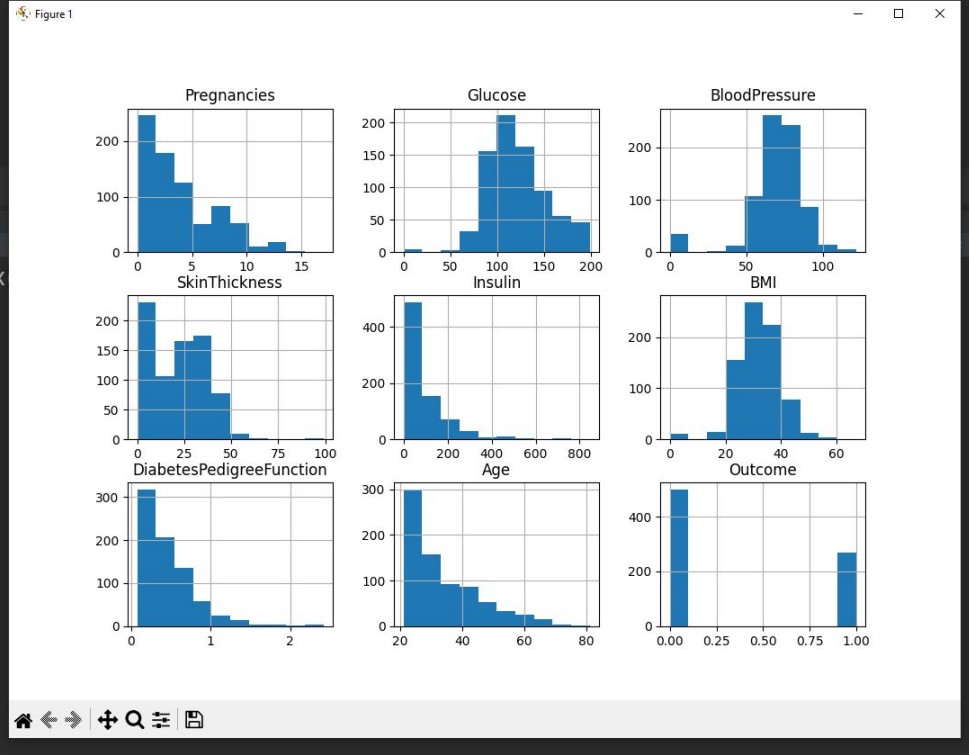


Step 4: Data visualization

Histogram

```
from pandas import read_csv
import matplotlib.pyplot as plt

data = read_csv("diabetes.csv")
data.hist()
plt.show()
```

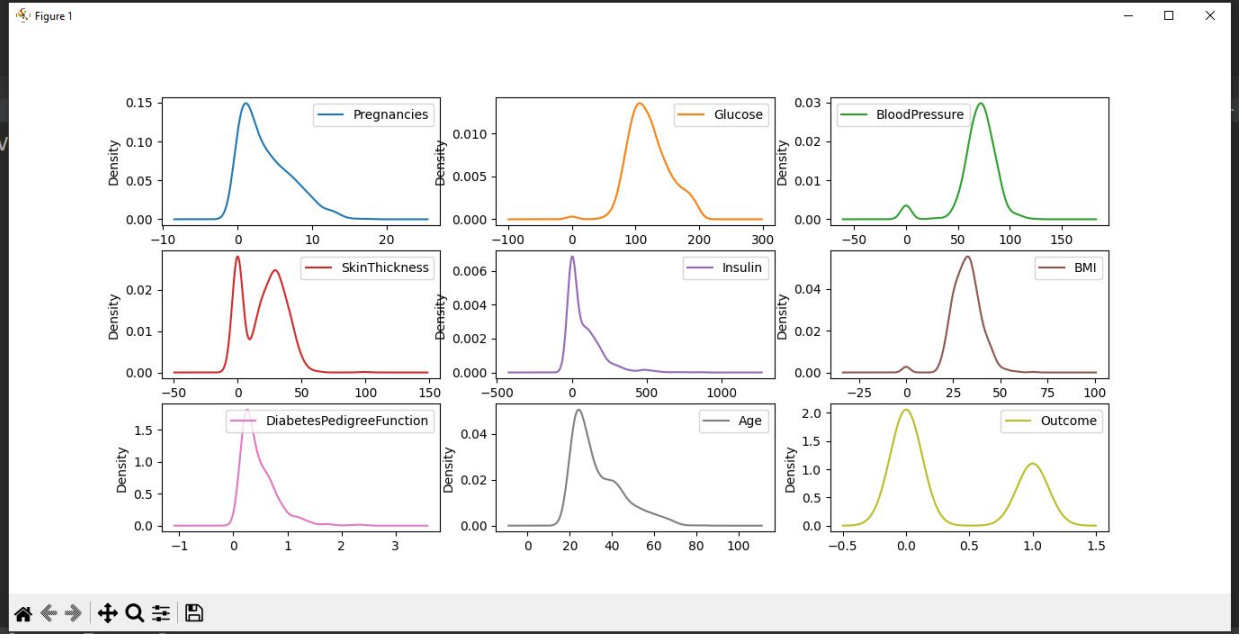
```
C:\Users\Viet\anaconda3\envs\basic\python.ex
```



Density Plot

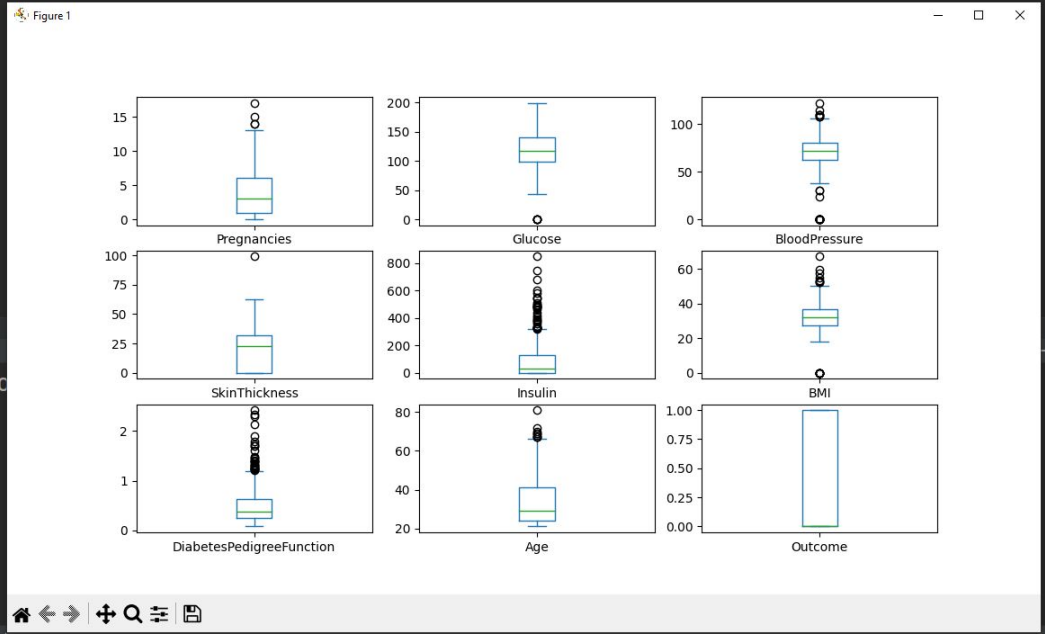
```python
from pandas import read_csv
import matplotlib.pyplot as plt

data = read_csv("diabetes.csv")
data.plot(kind='density', subplots=True, layout=(3, 3), sharex=False)
plt.show()
```

```
C:\Users\Viet\anaconda3\env
```

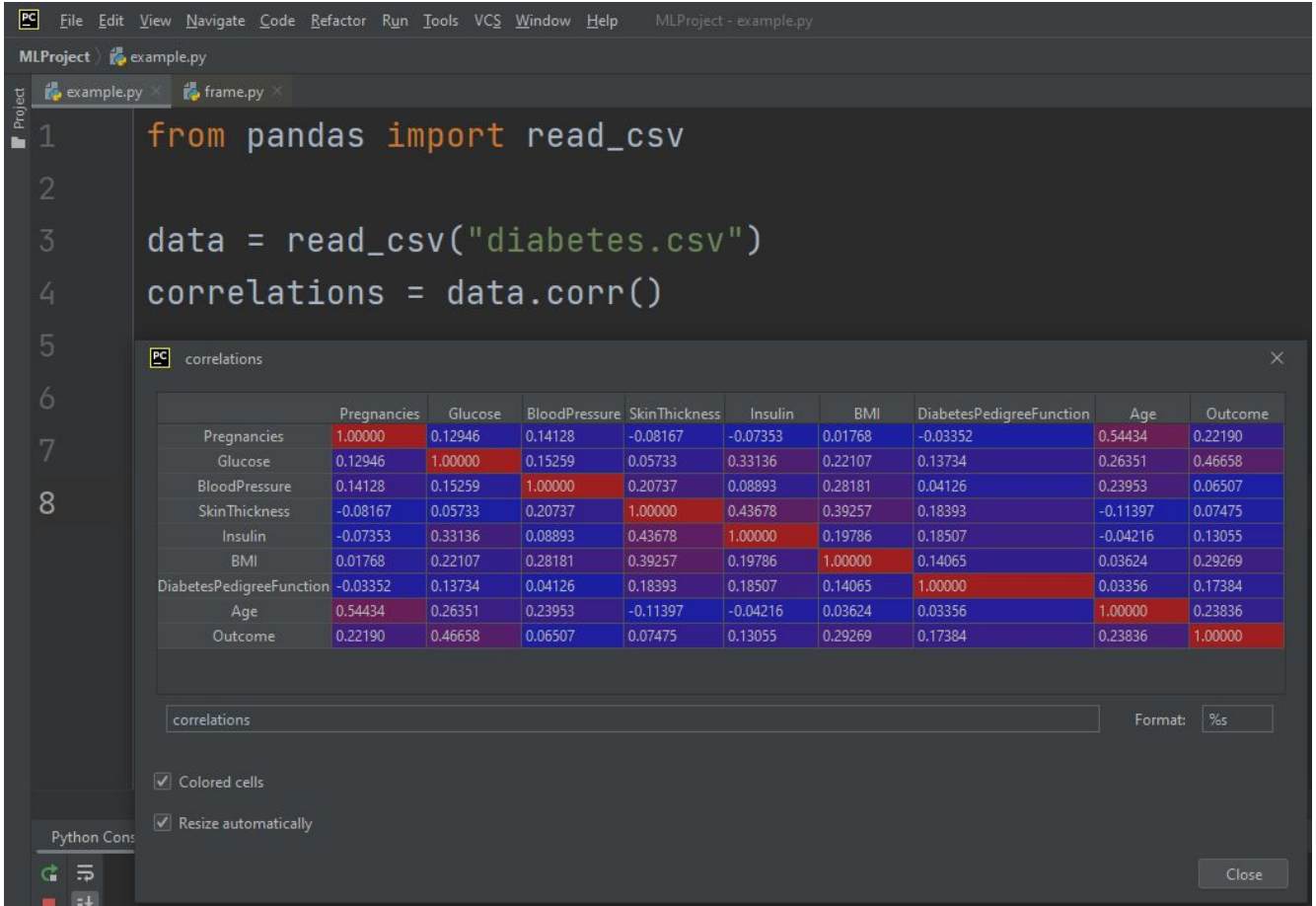


Box Plot

```python
from pandas import read_csv
import matplotlib.pyplot as plt

data = read_csv("diabetes.csv")
data.plot(kind='box', subplots=True, layout=(3, 3), sharex=False)
plt.show()
```



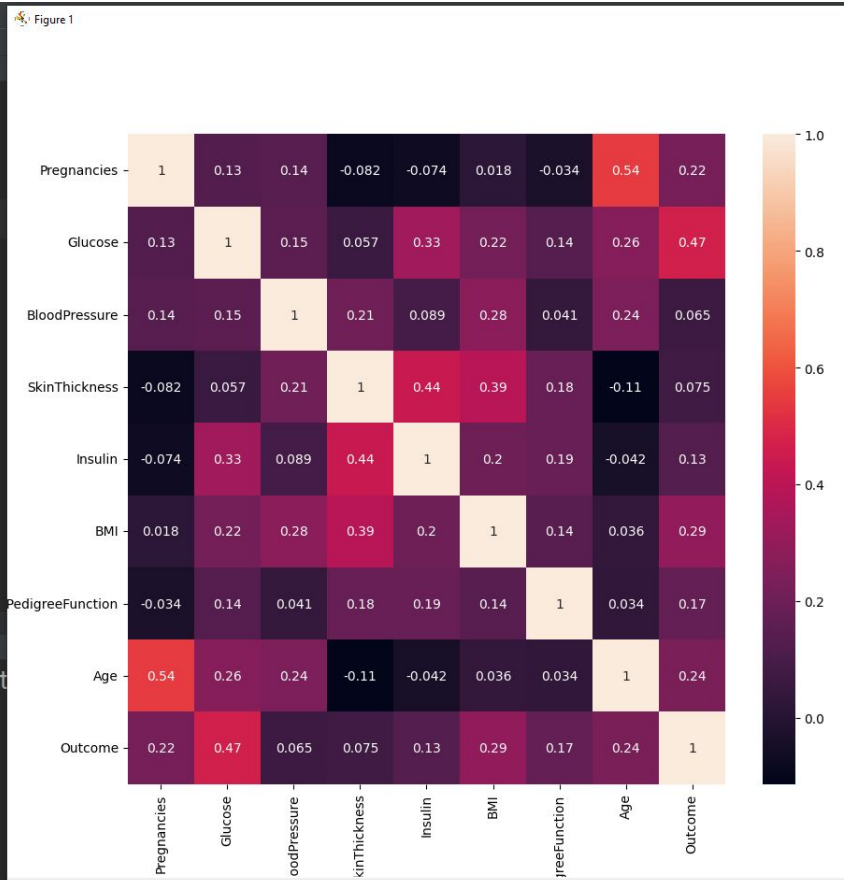
Correlation Matrix Plot



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.00000	0.12946	0.14128	-0.08167	-0.07353	0.01768	-0.03352	0.54434	0.22190
Glucose	0.12946	1.00000	0.15259	0.05733	0.33136	0.22107	0.13734	0.26351	0.46658
BloodPressure	0.14128	0.15259	1.00000	0.20737	0.08893	0.28181	0.04126	0.23953	0.06507
SkinThickness	-0.08167	0.05733	0.20737	1.00000	0.43678	0.39257	0.18393	-0.11397	0.07475
Insulin	-0.07353	0.33136	0.08893	0.43678	1.00000	0.19786	0.18507	-0.04216	0.13055
BMI	0.01768	0.22107	0.28181	0.39257	0.19786	1.00000	0.14065	0.03624	0.29269
DiabetesPedigreeFunction	-0.03352	0.13734	0.04126	0.18393	0.18507	0.14065	1.00000	0.03356	0.17384
Age	0.54434	0.26351	0.23953	-0.11397	-0.04216	0.03624	0.03356	1.00000	0.23836
Outcome	0.22190	0.46658	0.06507	0.07475	0.13055	0.29269	0.17384	0.23836	1.00000

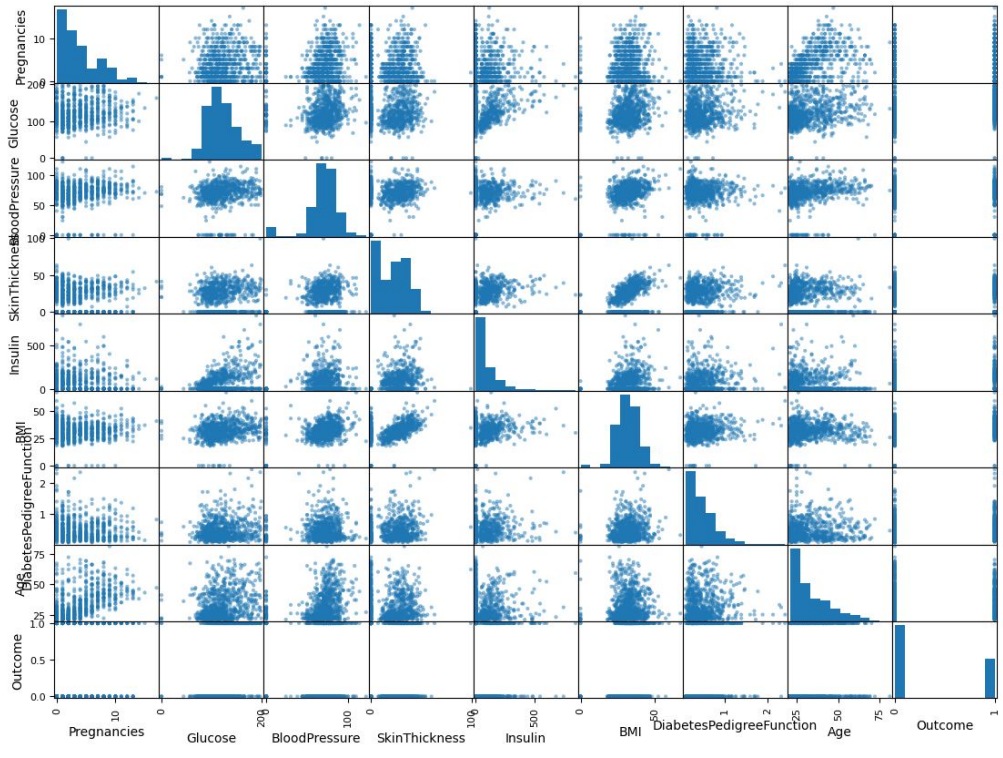
Correlation Matrix Plot

```
from pandas import read_csv
import seaborn as sn
import matplotlib.pyplot as plt

data = read_csv("diabetes.csv")
sn.heatmap(data.corr(), annot=True)
plt.show()
```

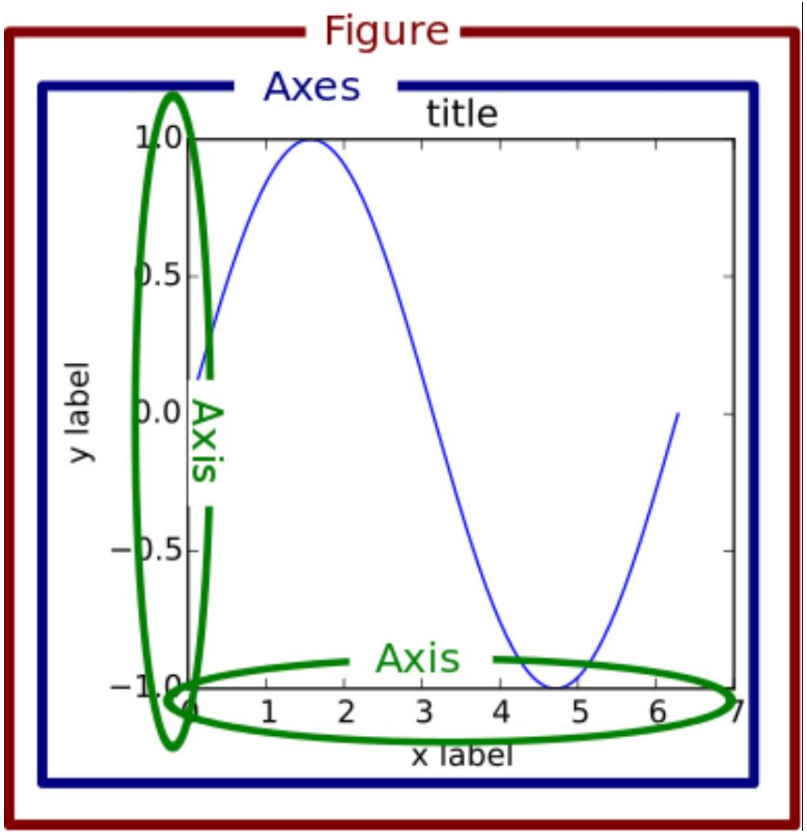


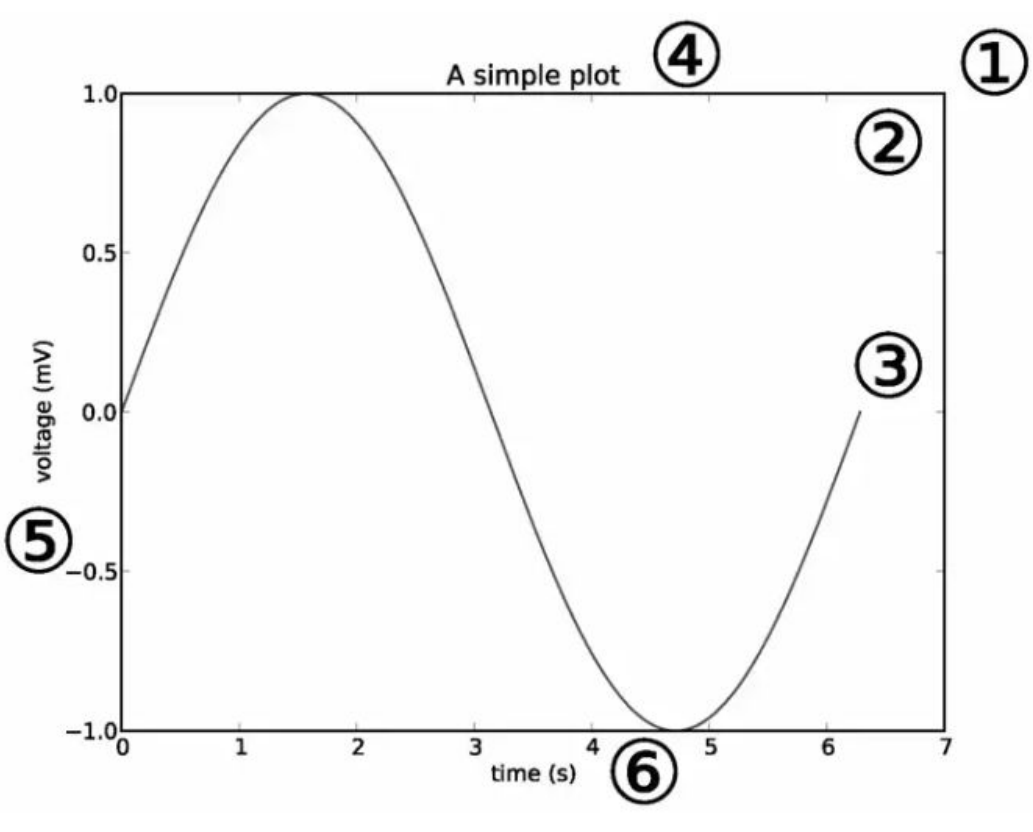
Scatter Matrix Plot

```
from pandas import read_csv
from pandas.plotting import scat
import matplotlib.pyplot as plt

data = read_csv("diabetes.csv")
scatter_matrix(data)
plt.show()
```

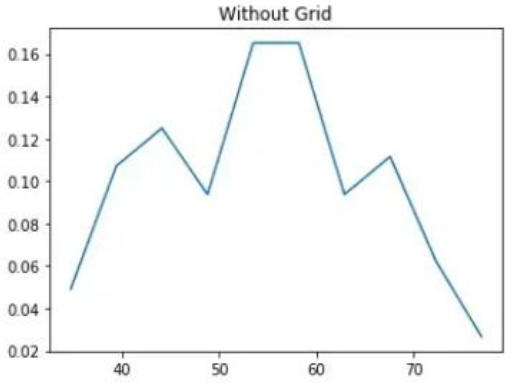


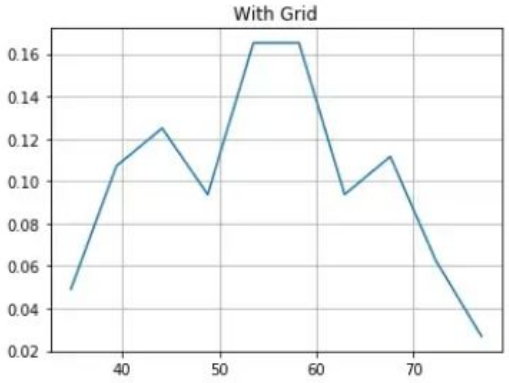
Data visualization with matplotlib

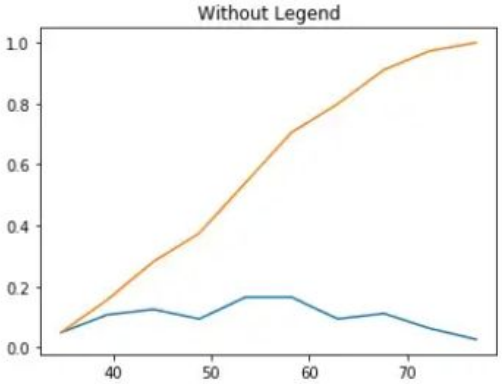


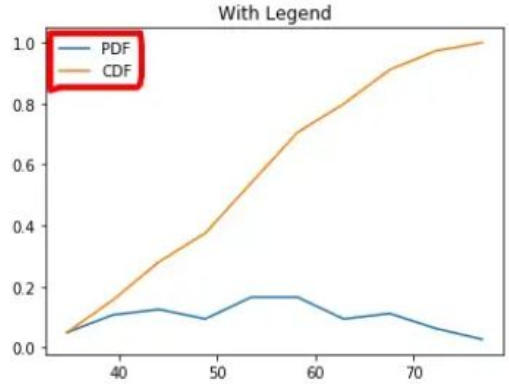


Data visualization with matplotlib









Supervised Learning algorithms

Regression(hồi quy)

Linear Regression
Polynomial Regression

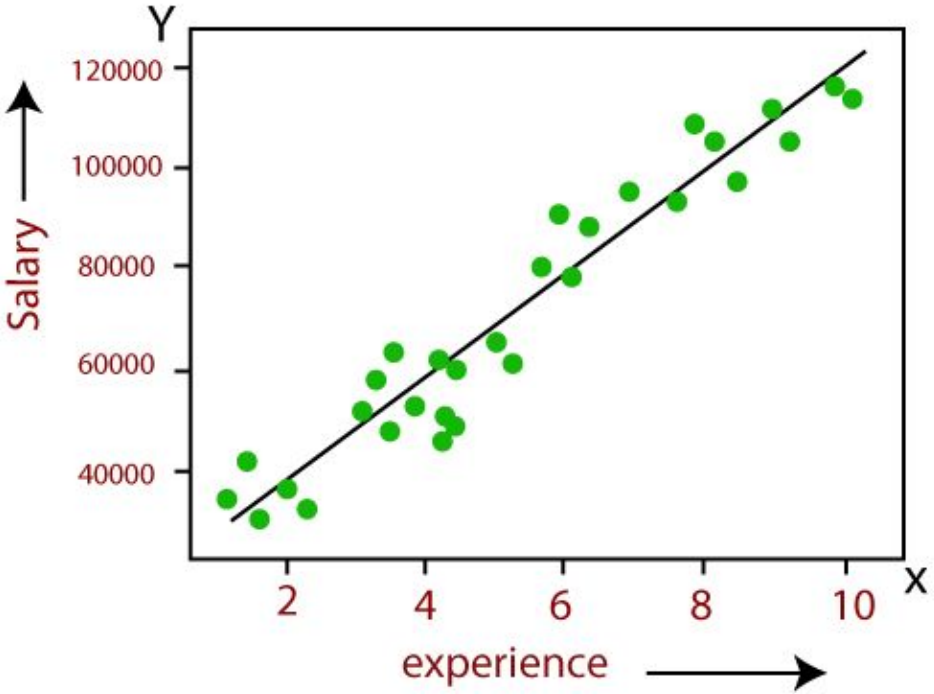
*Could also be used here

Classification(Phân loại)

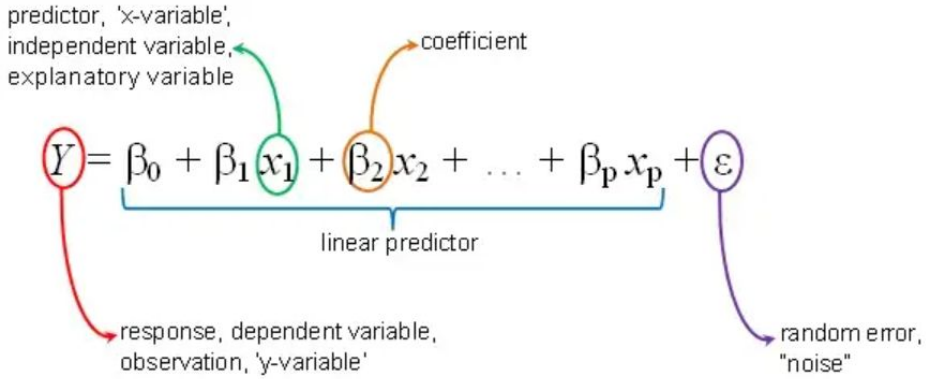
Logistic Regression
Support Vector Machine*

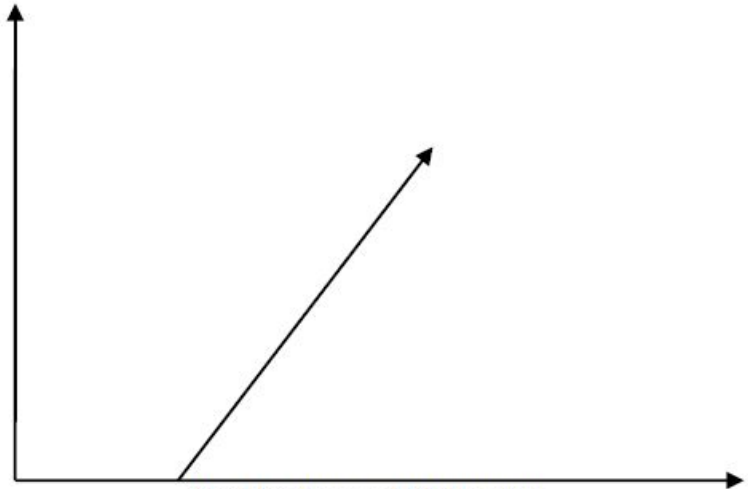
K-nearest neighbours*
Naive Bayes
Decision Tree*
Random Forest*

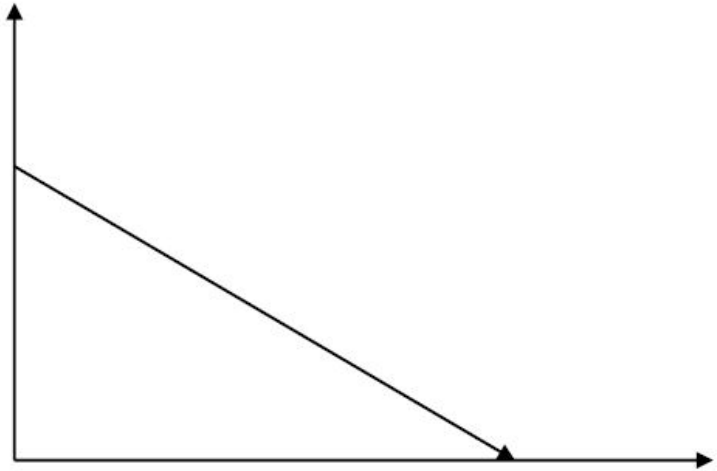
Linear Regression



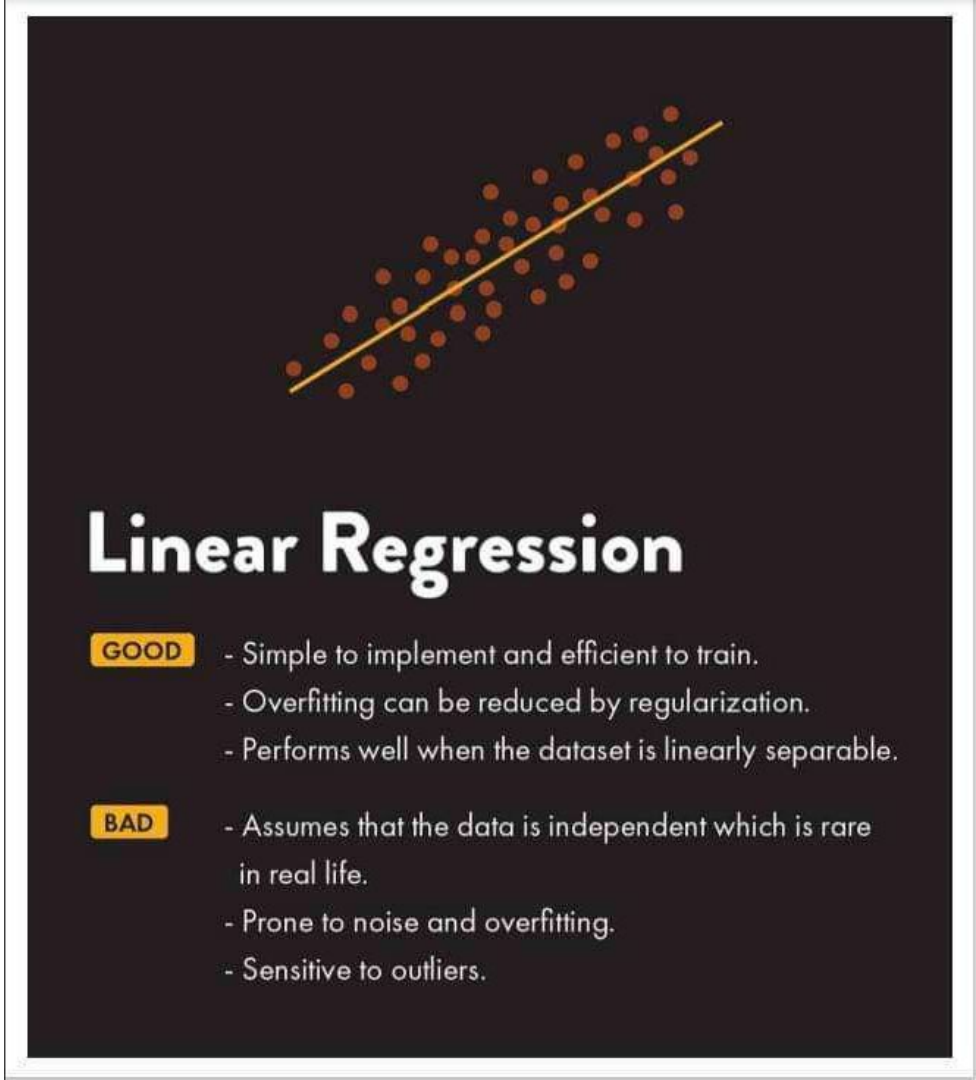
- 1 trong số các thuật toán cơ bản và đơn giản nhất trong Supervised Learning
- Simple linear regression: 1 input feature
- Multiple linear regression: 2 input features trở lên
- Phù hợp với các linear dataset



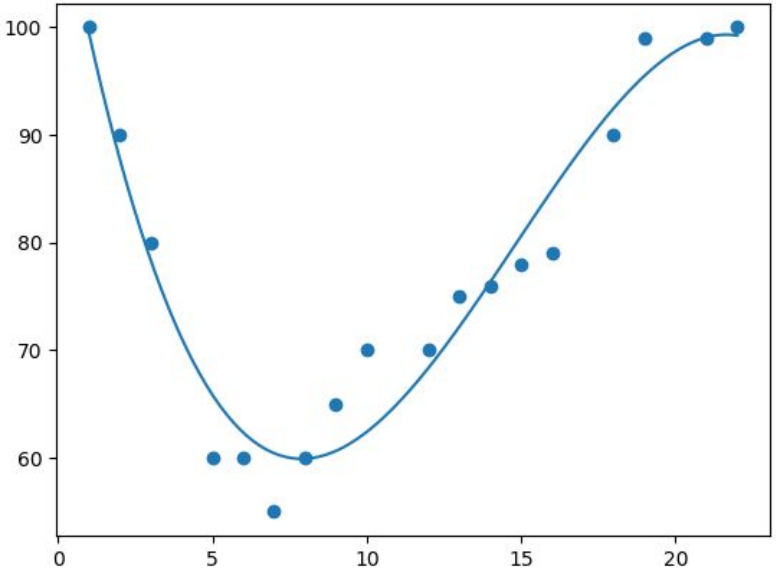


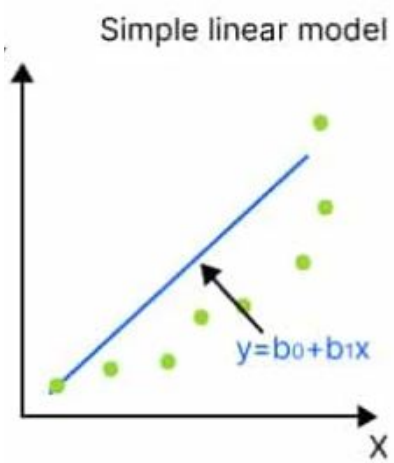


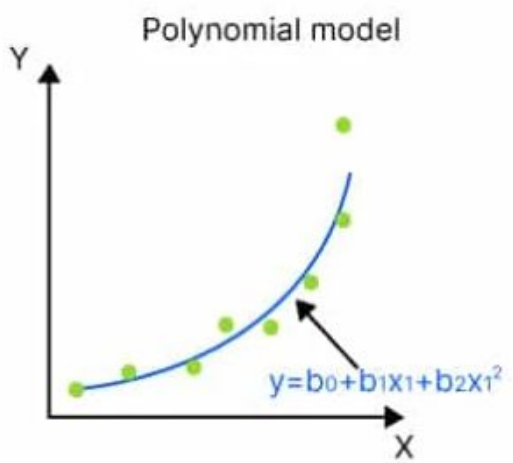
Negative Linear Relationship



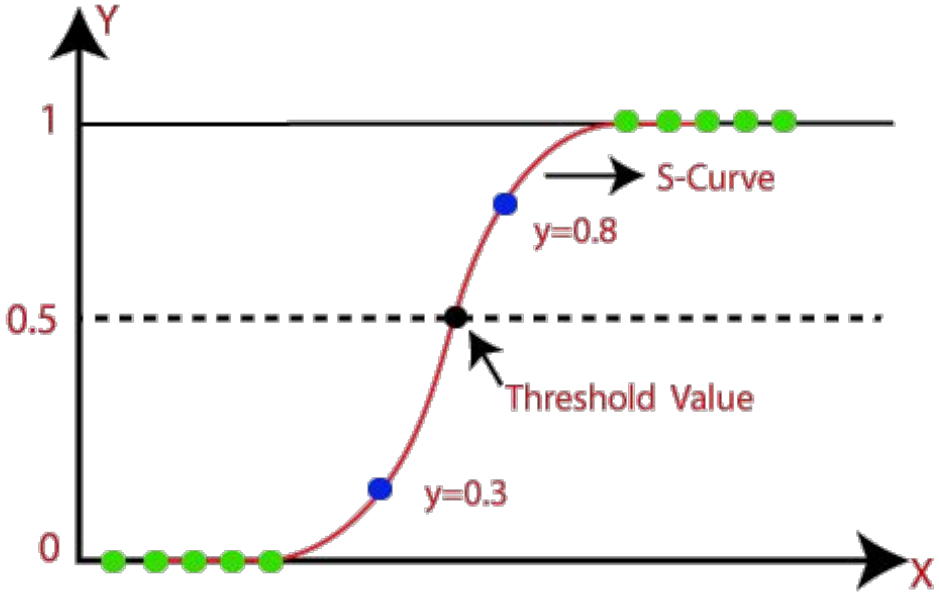
Polynomial Regression

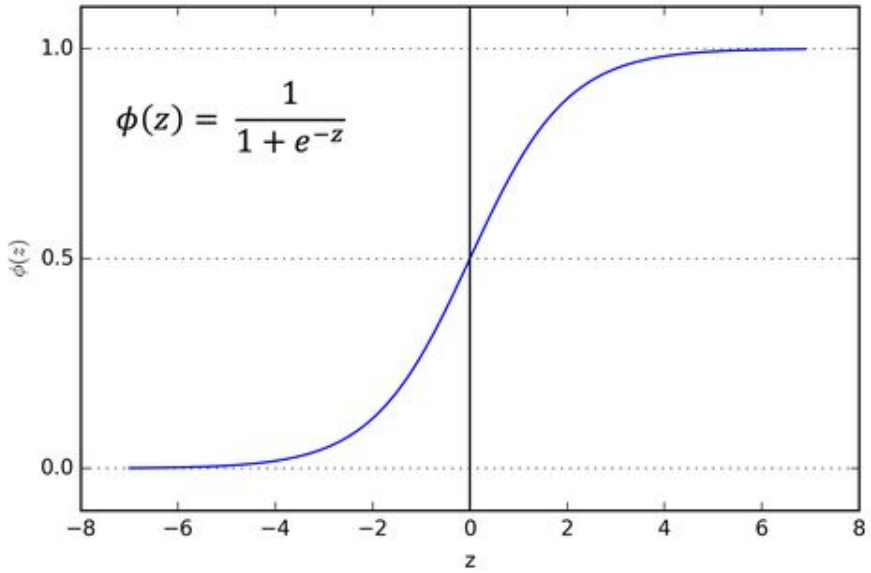


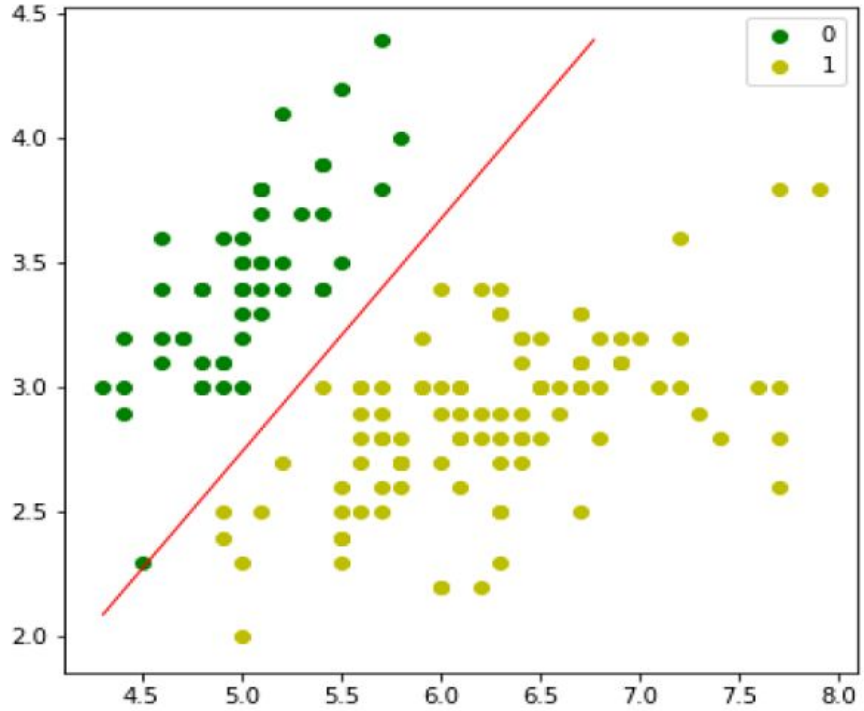




Logistic Regression







Logistic Regression

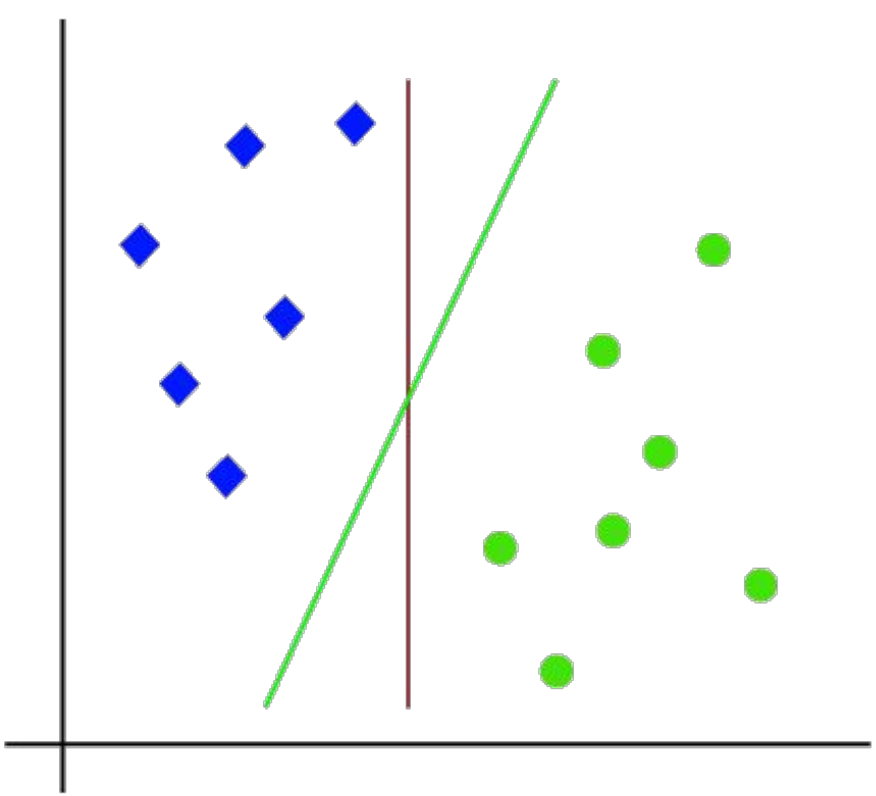
GOOD

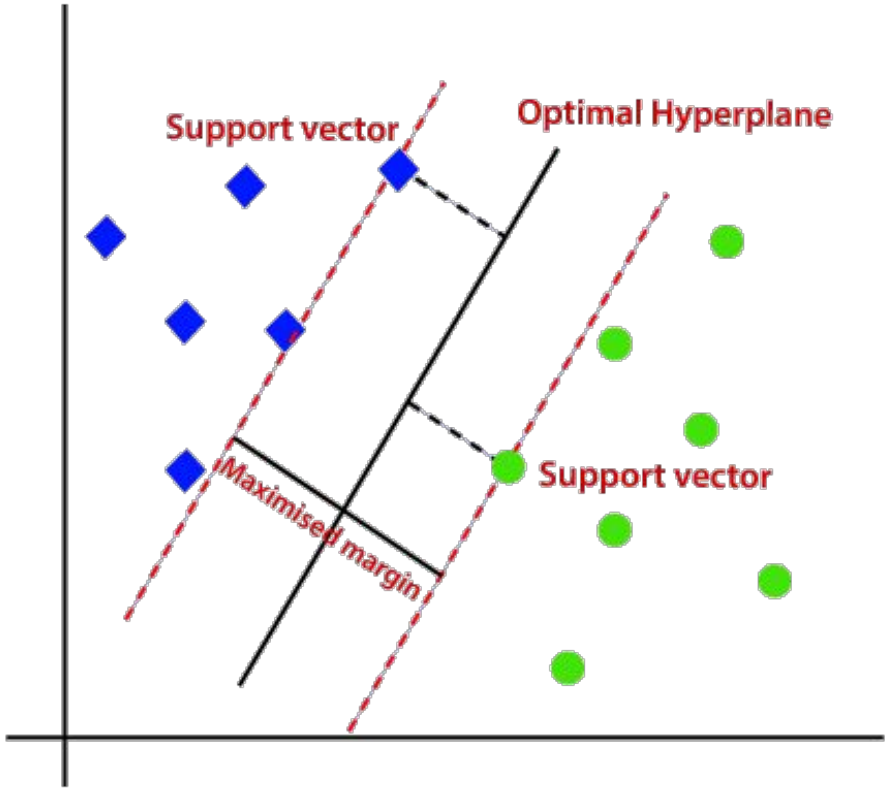
- Less prone to over-fitting but it can overfit in high dimensional datasets.
- Efficient when the dataset has features that are linearly separable.
- Easy to implement and efficient to train.

BAD

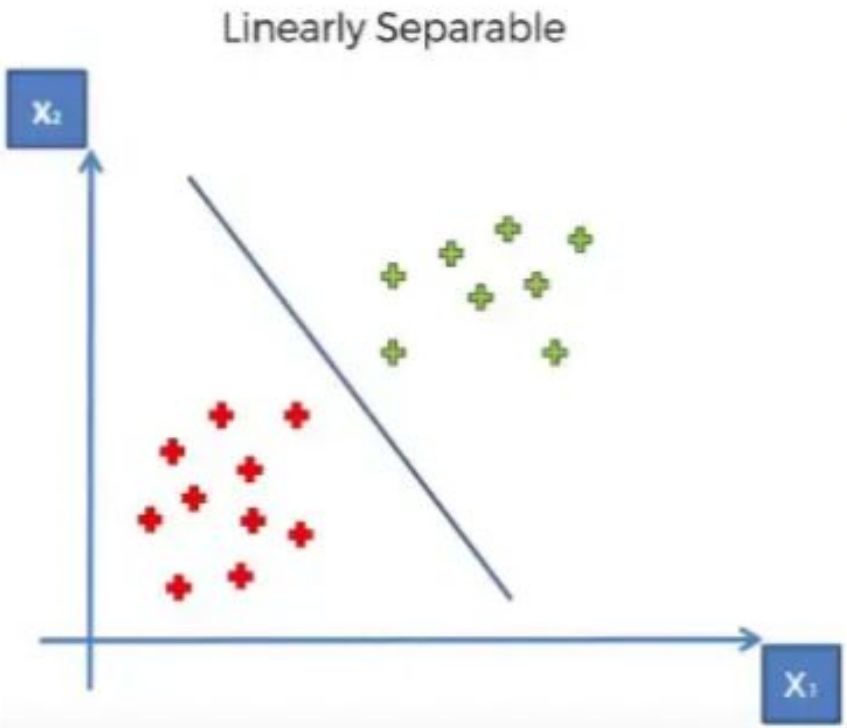
- Should not be used when the number of observations are lesser than the number of features.
- Assumption of linearity which is rare in practise.
- Can only be used to predict discrete functions.

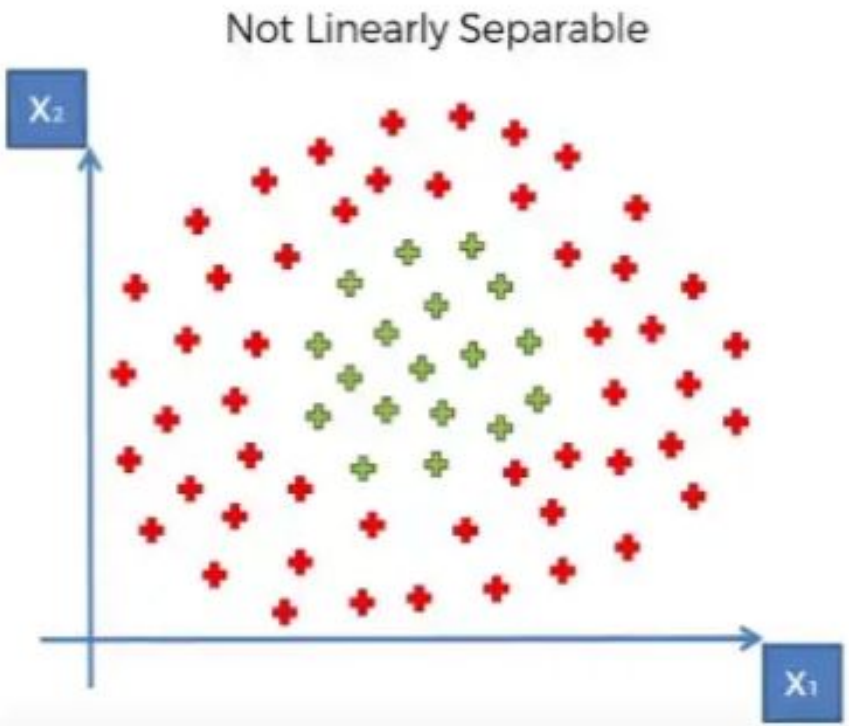
Support Vector Machine



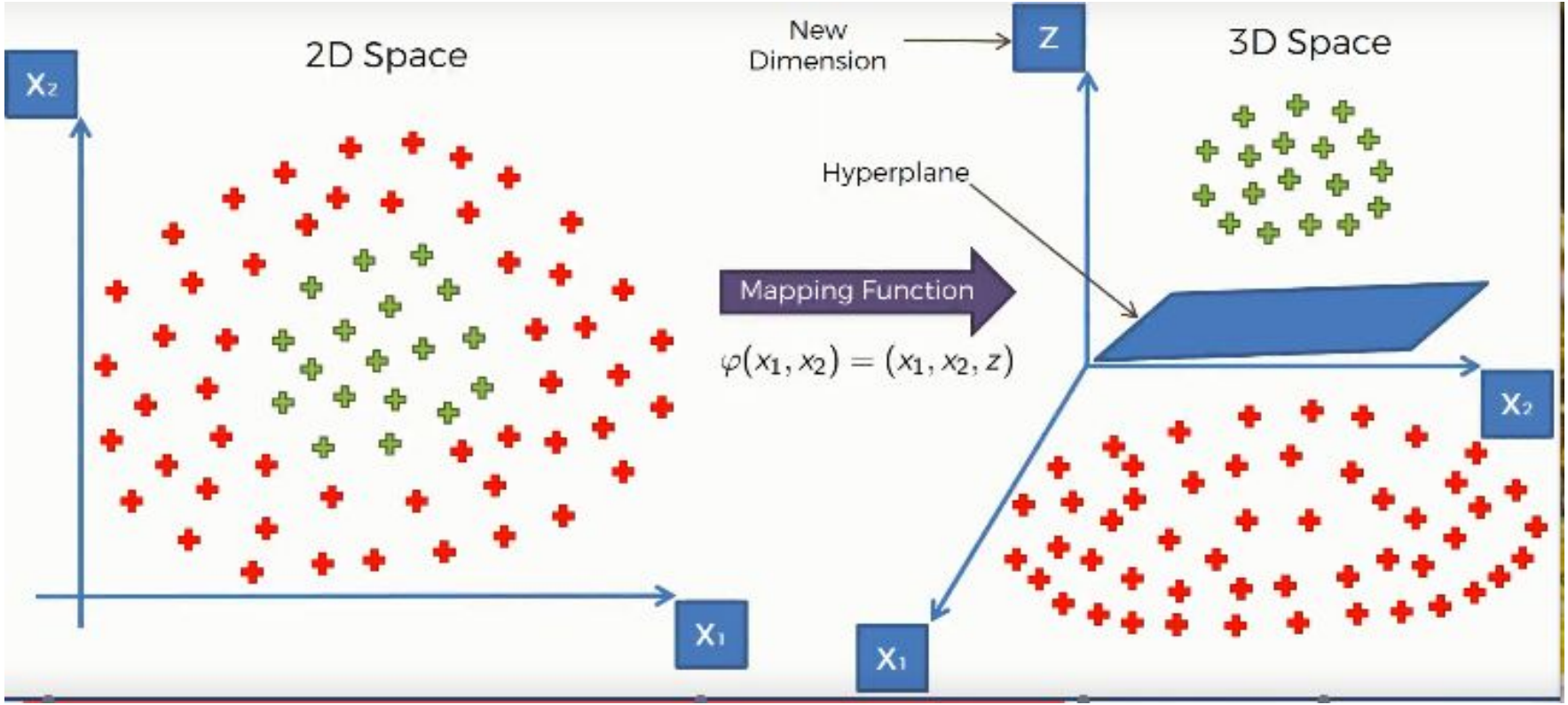


Support Vector Machine

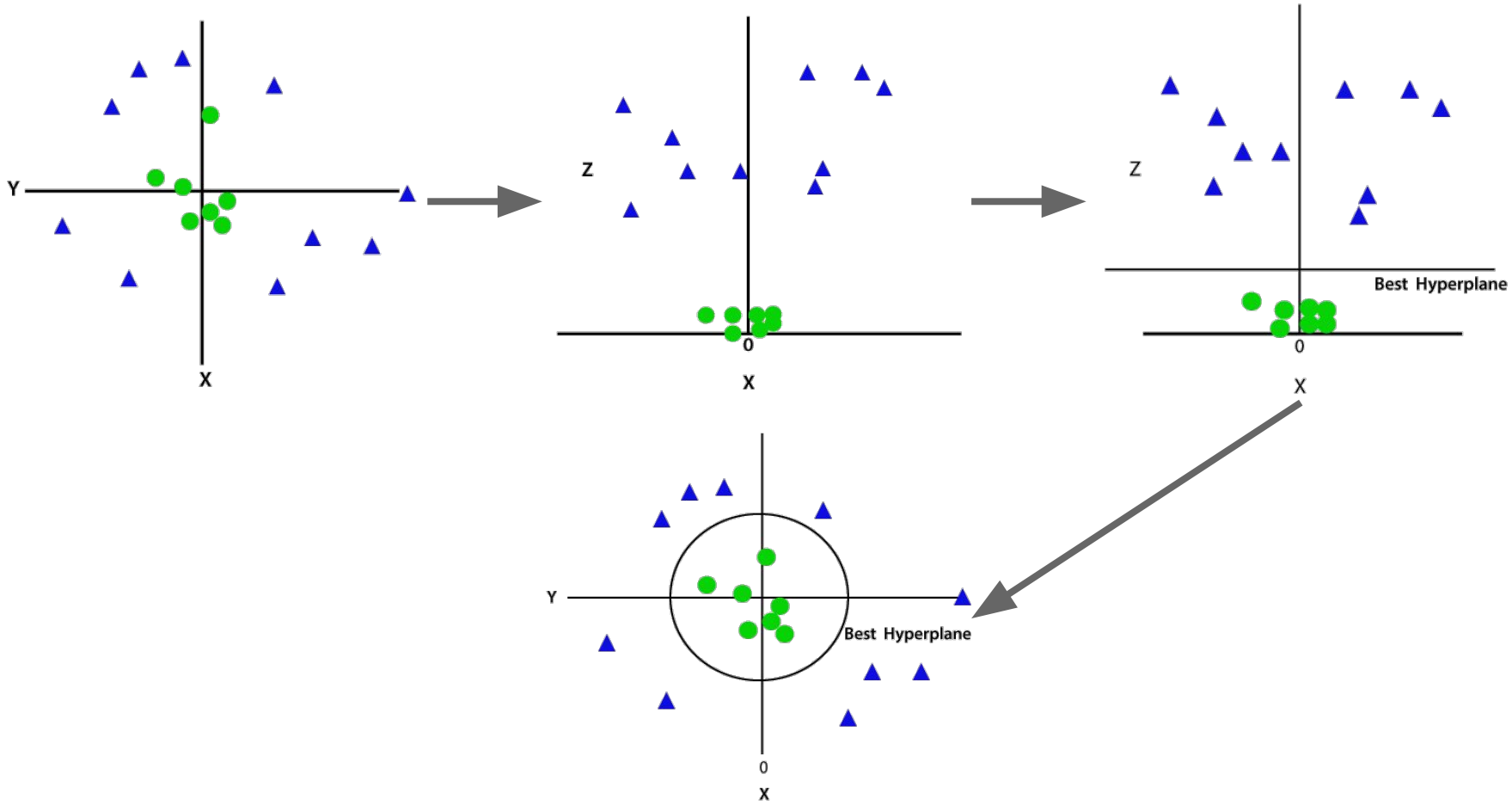


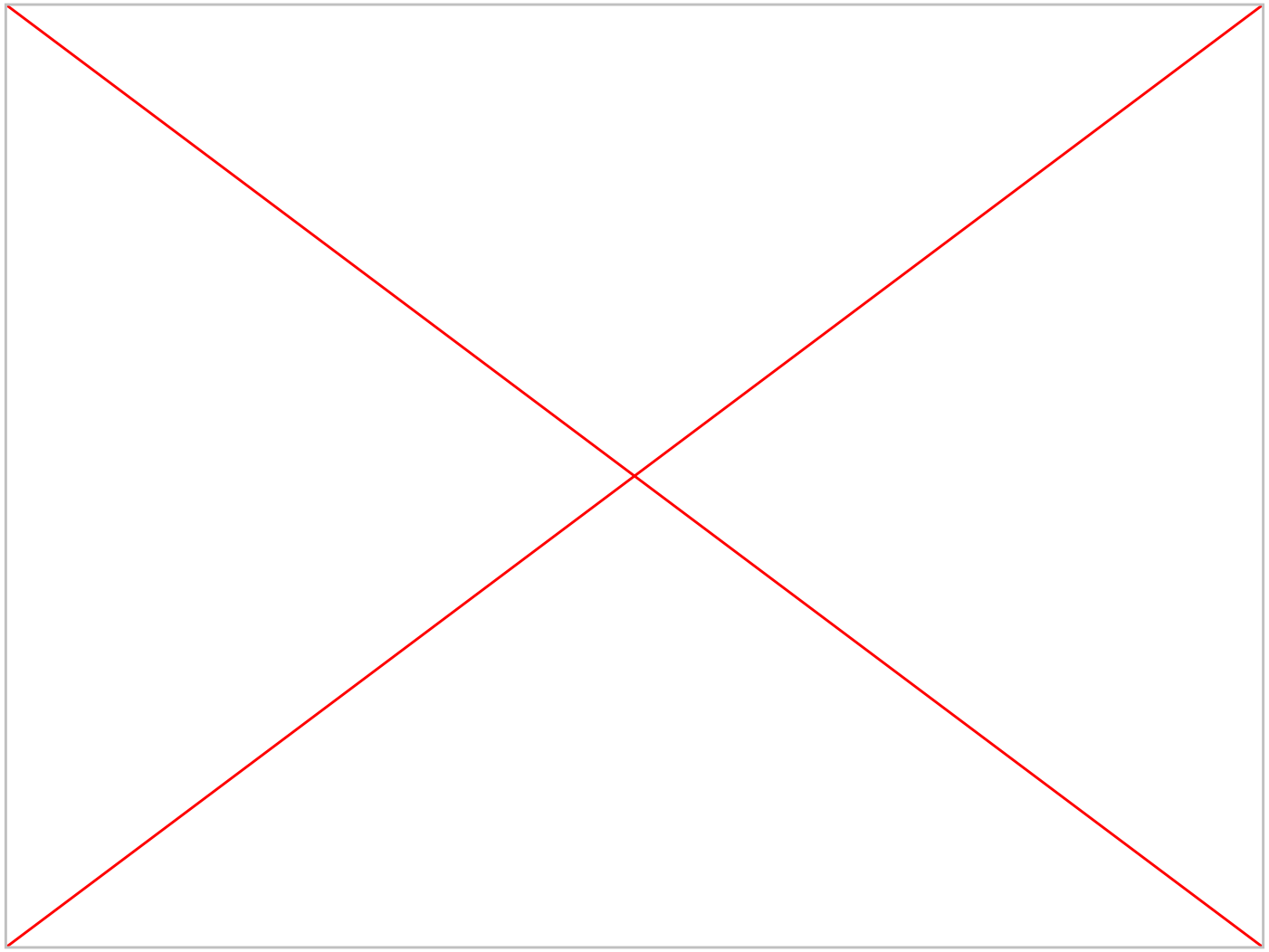


Support Vector Machine

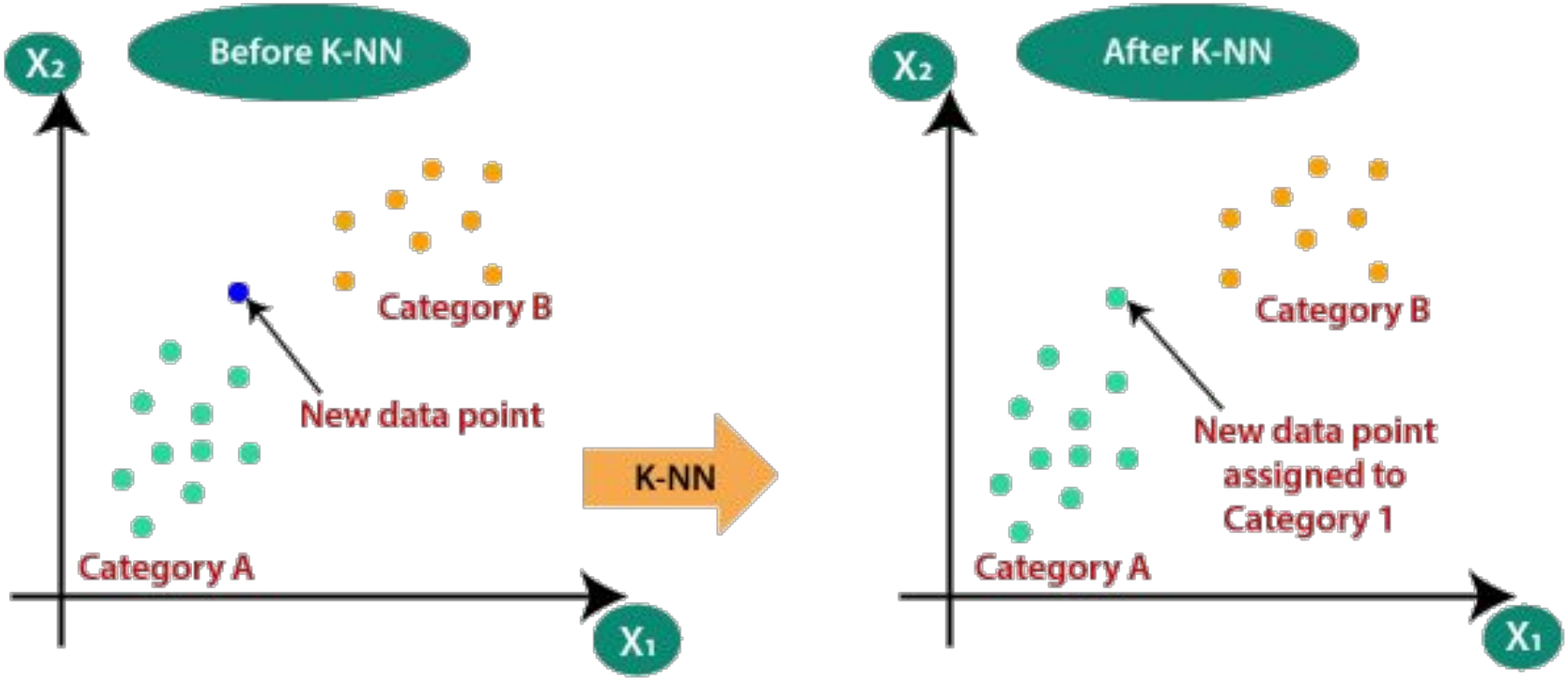


Support Vector Machine

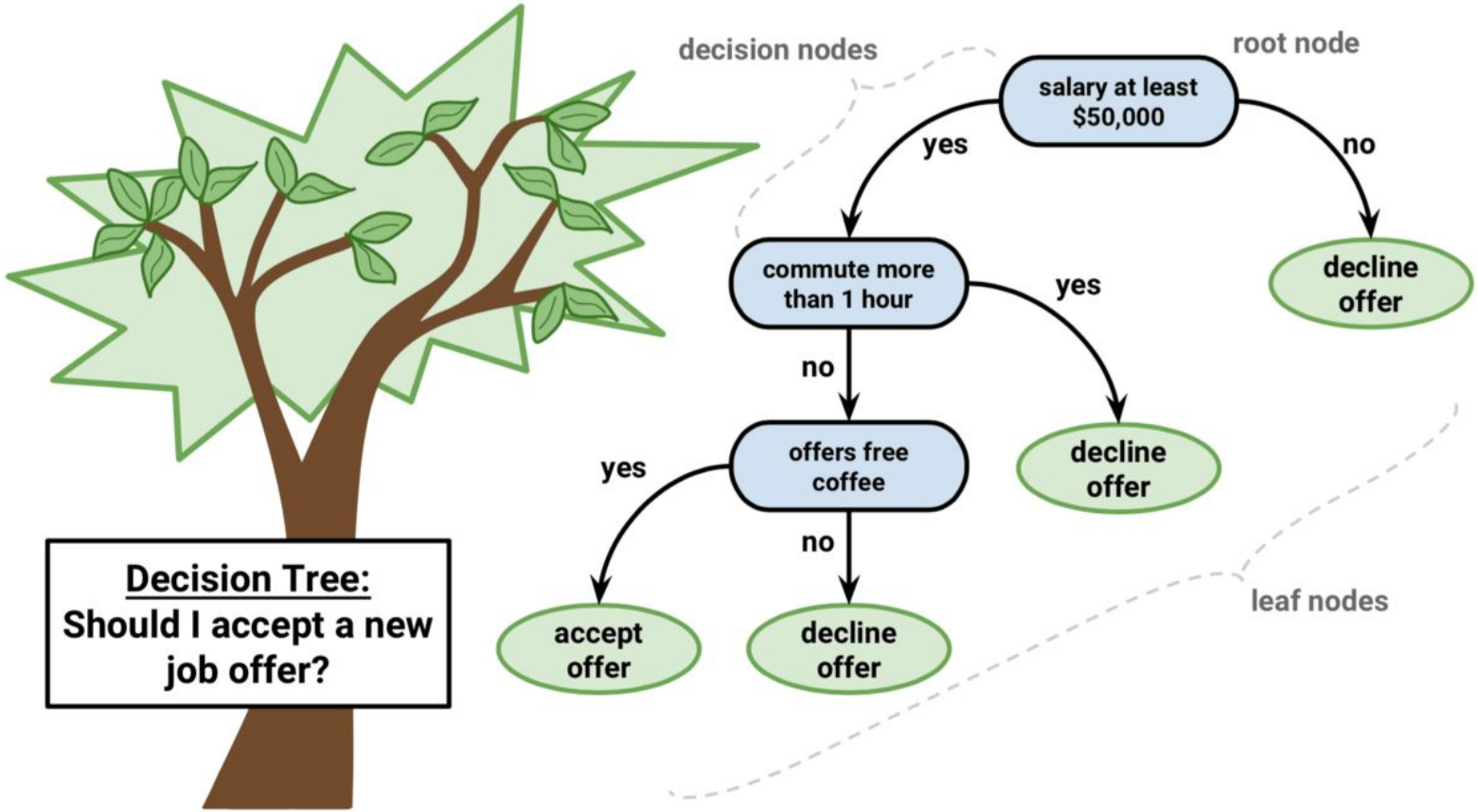




K-Nearest-Neighbour

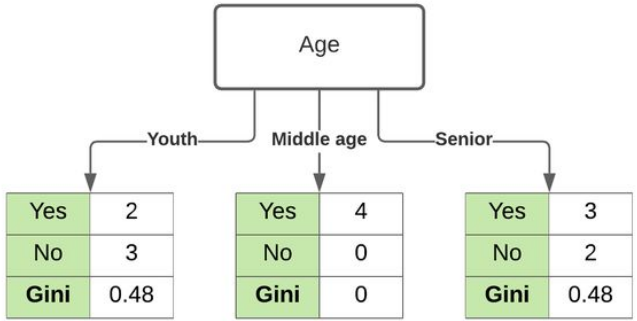


Decision Tree

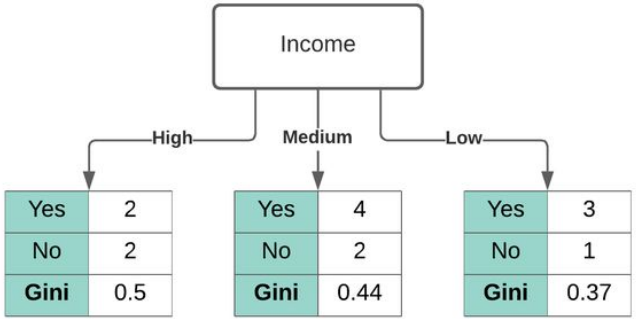


Decision Tree

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

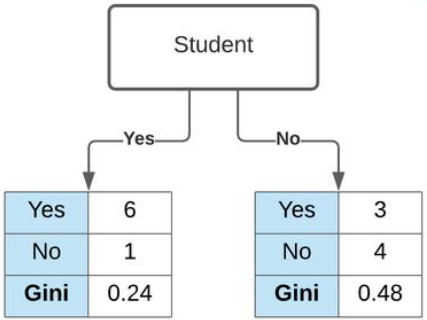


Gini Impurity for Age is 0.343

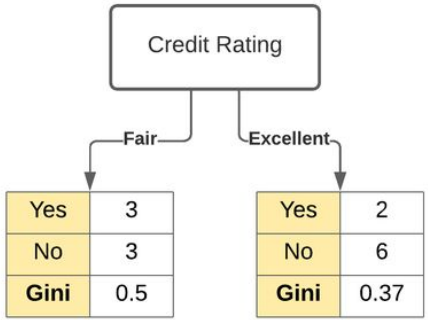


Gini Impurity for Income is 0.440

Best

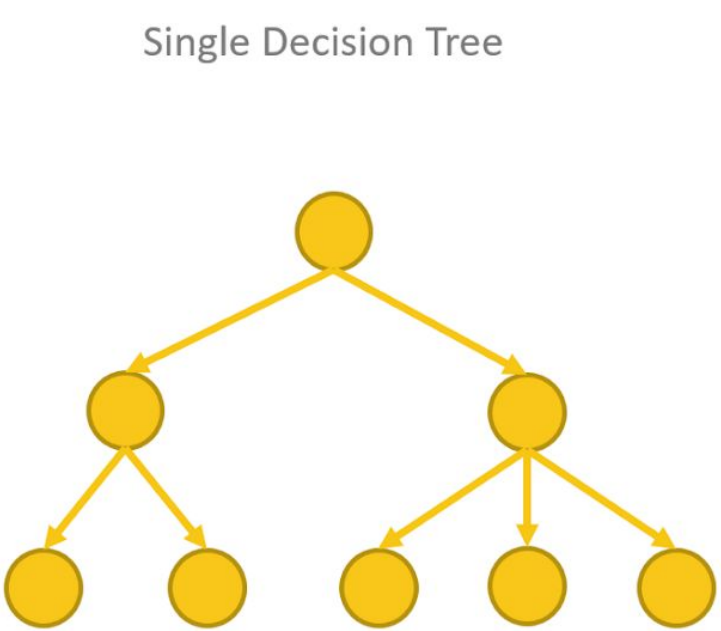


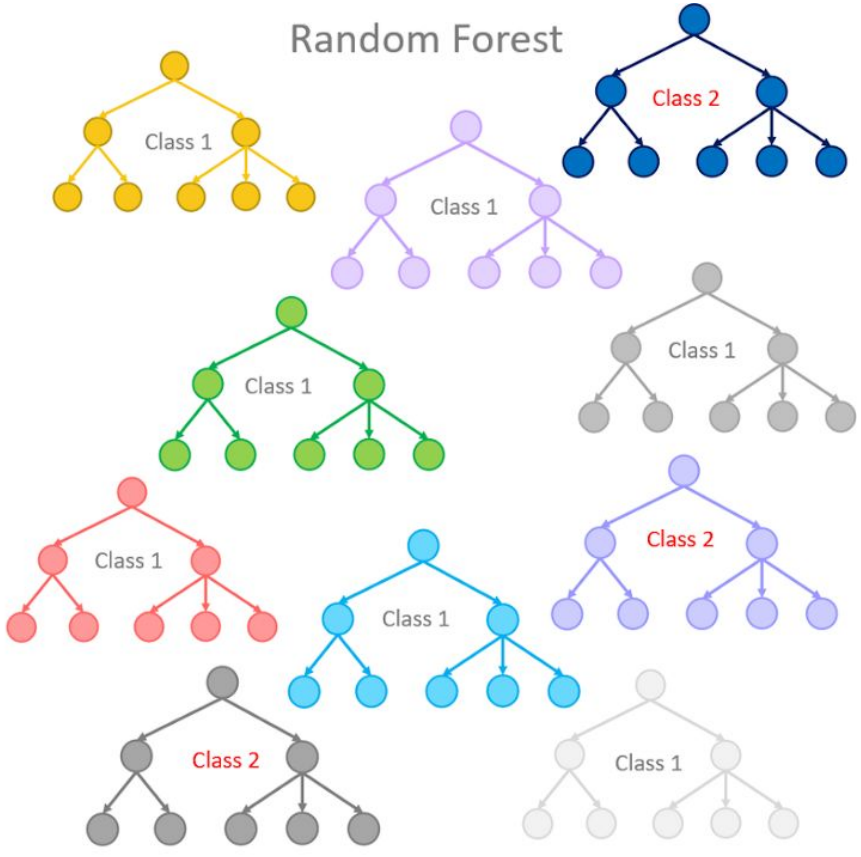
Gini Impurity for Student is 0.367



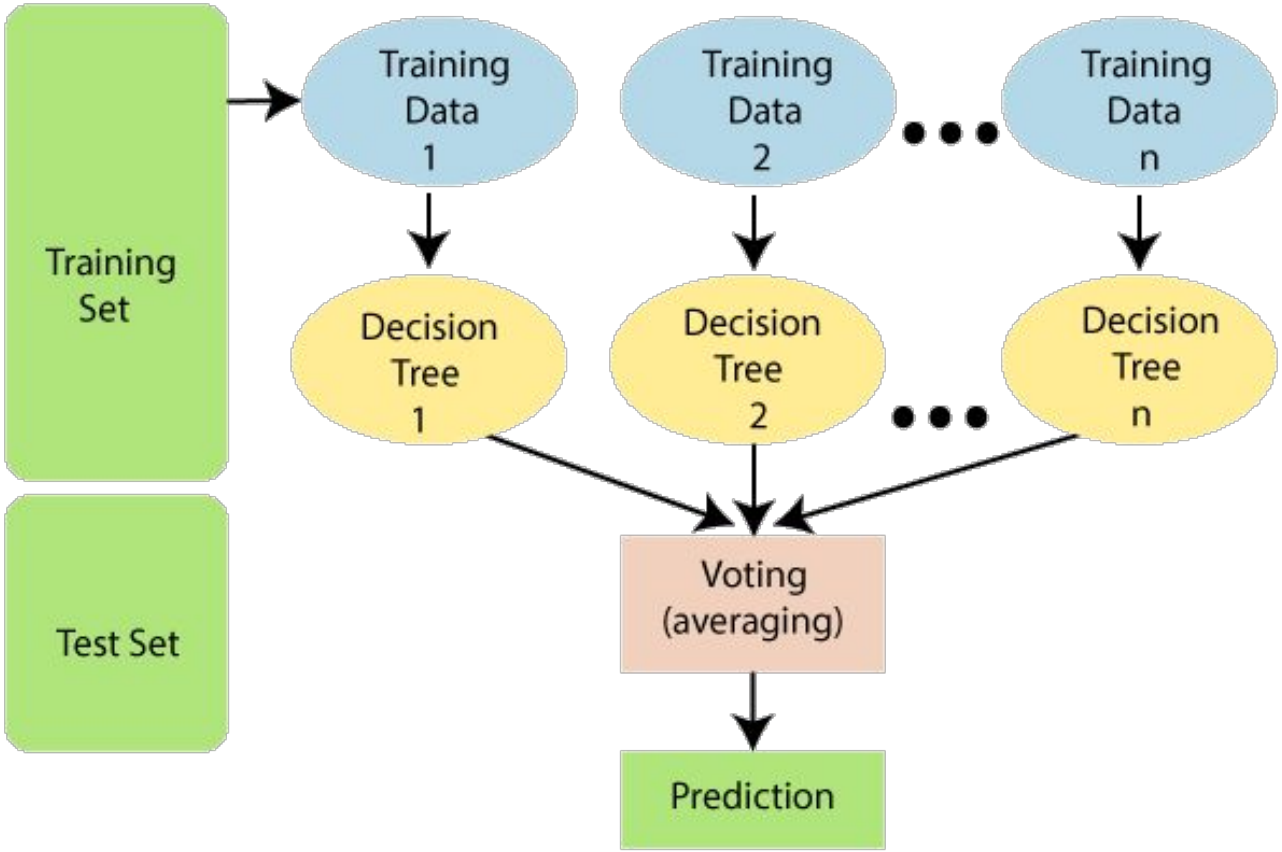
Gini Impurity for Credit Rating is 0.429

Random Forest

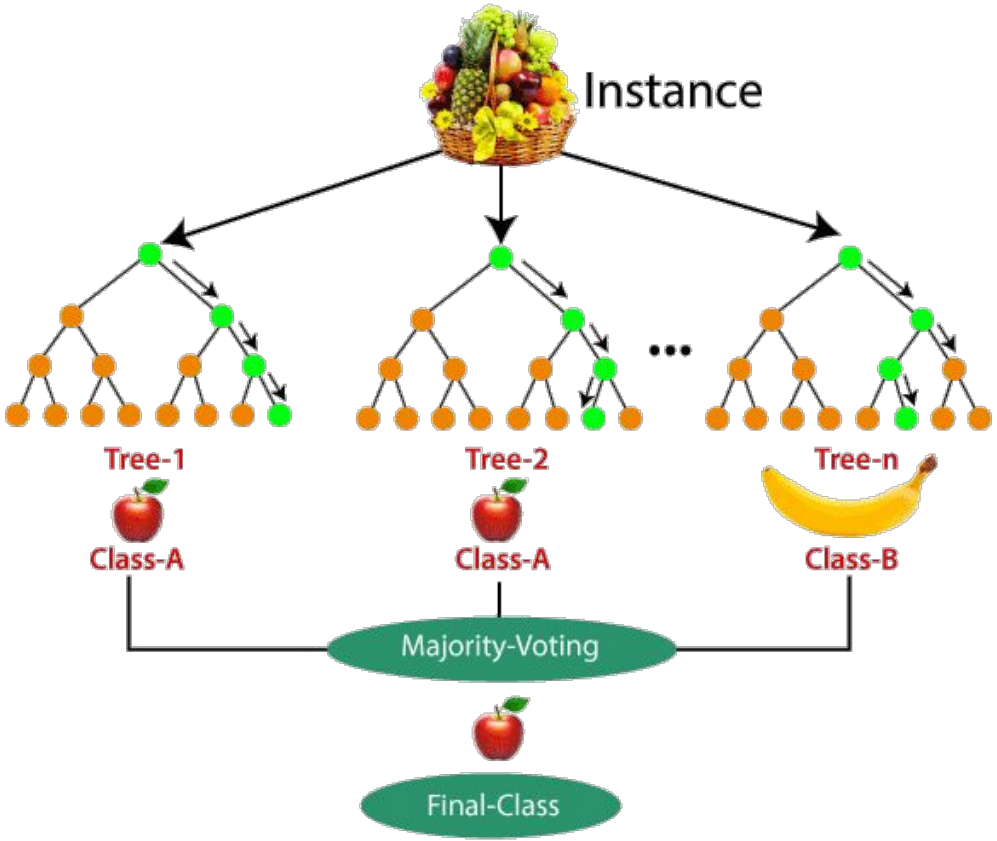




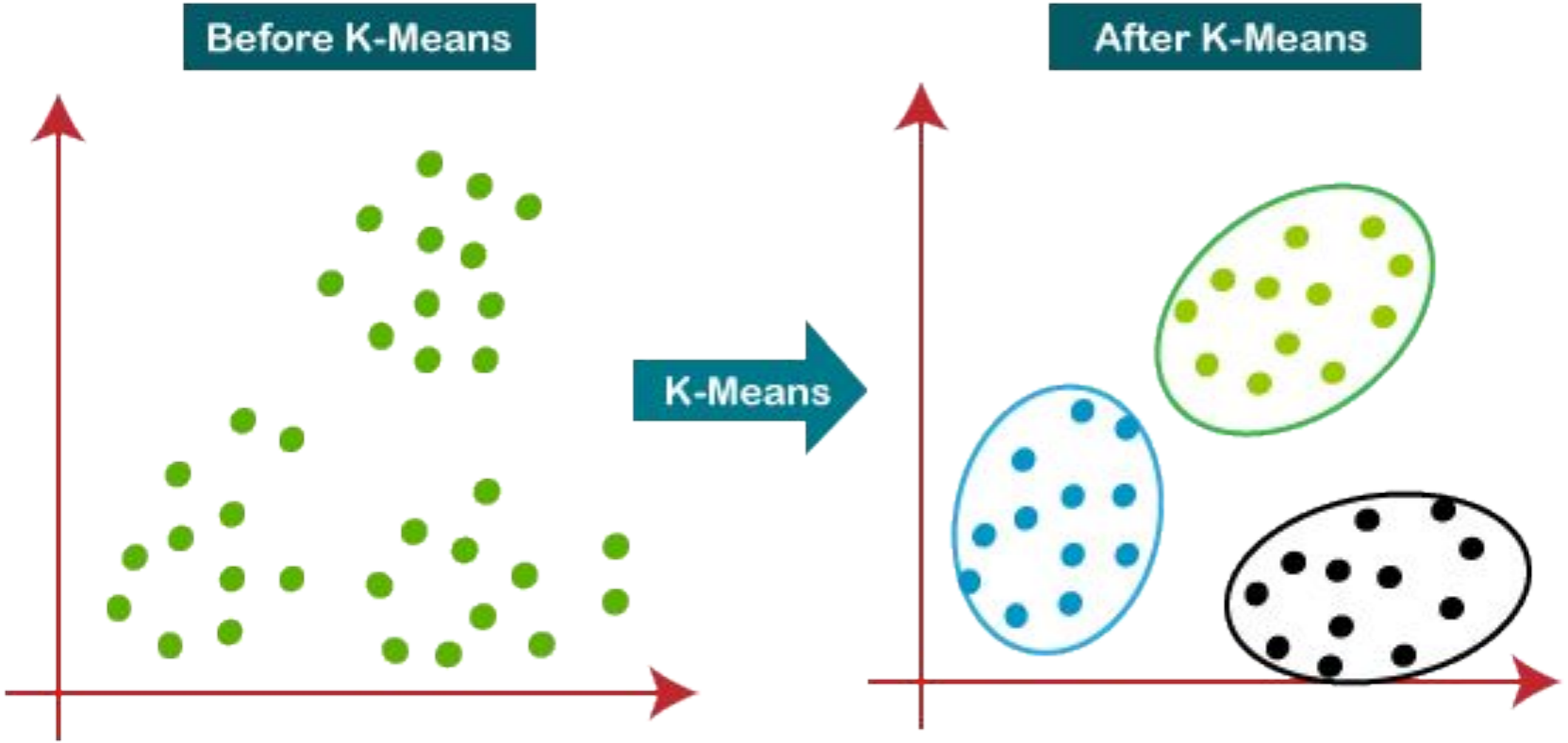
Random Forest

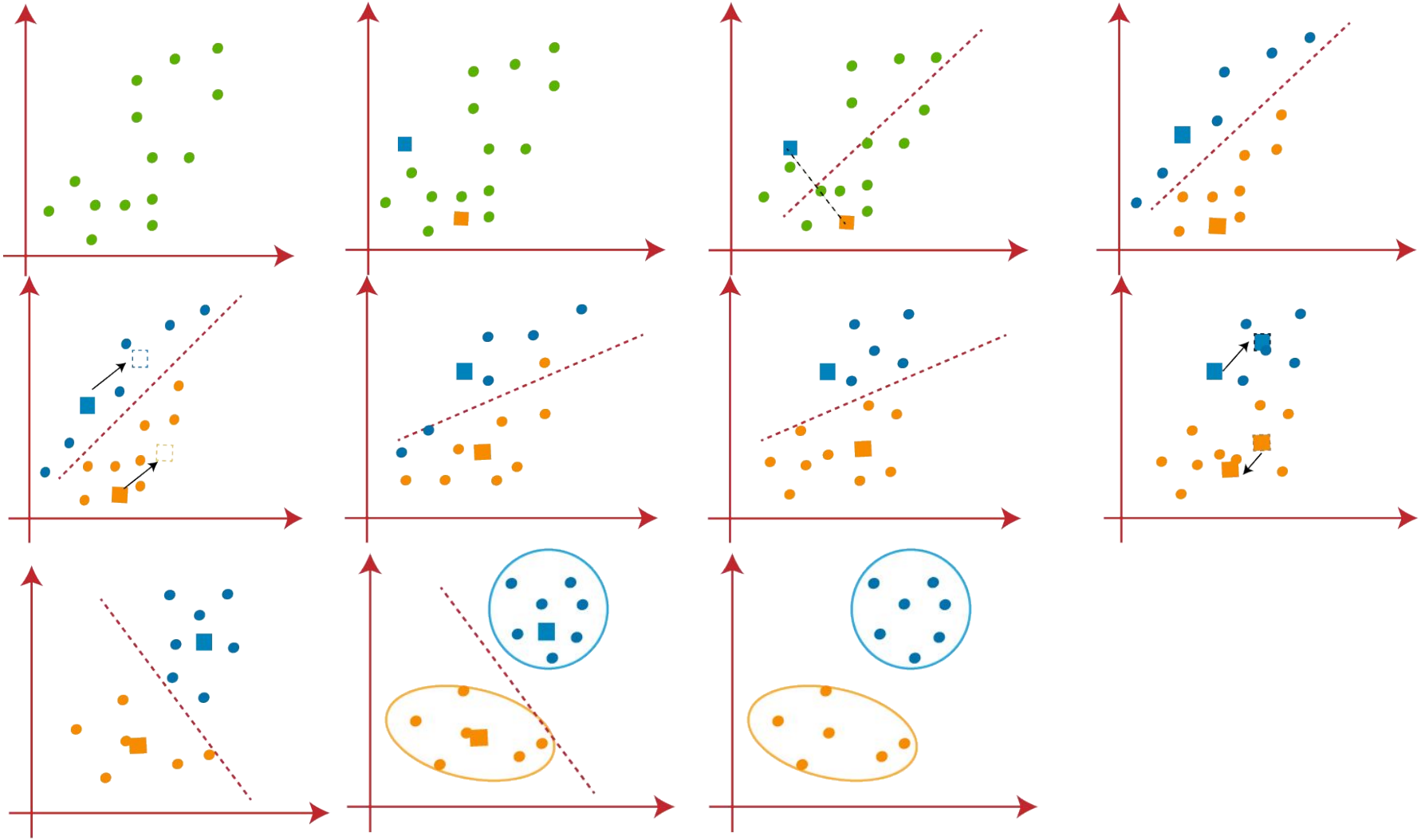


Random Forest

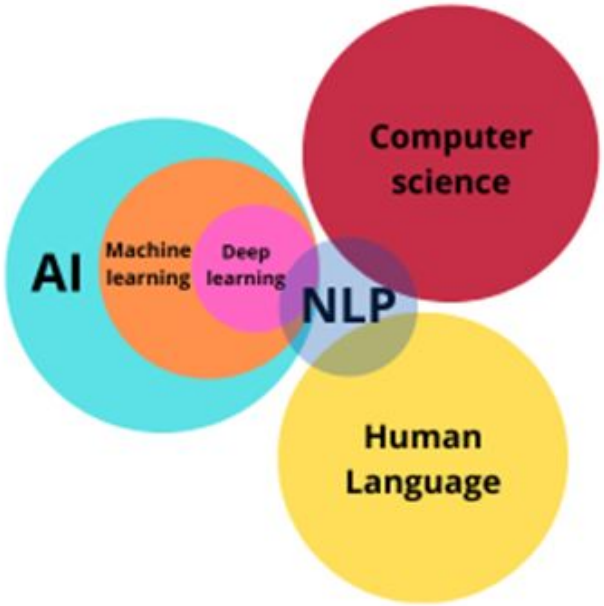


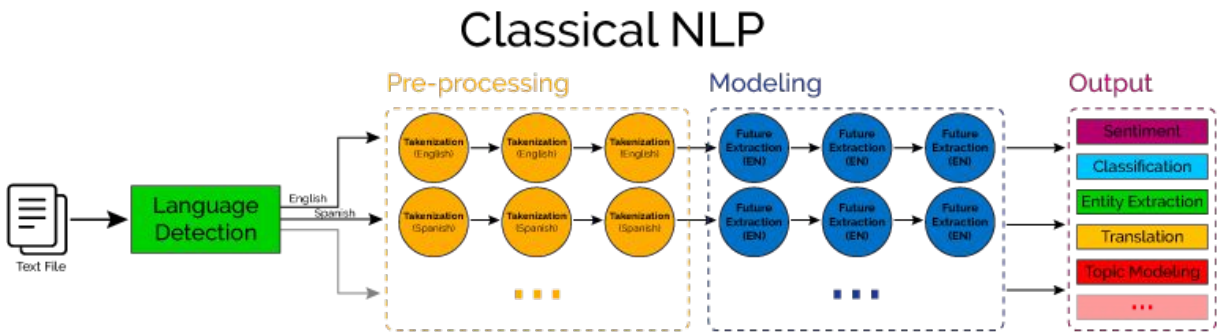
Unsupervised Learning algorithms

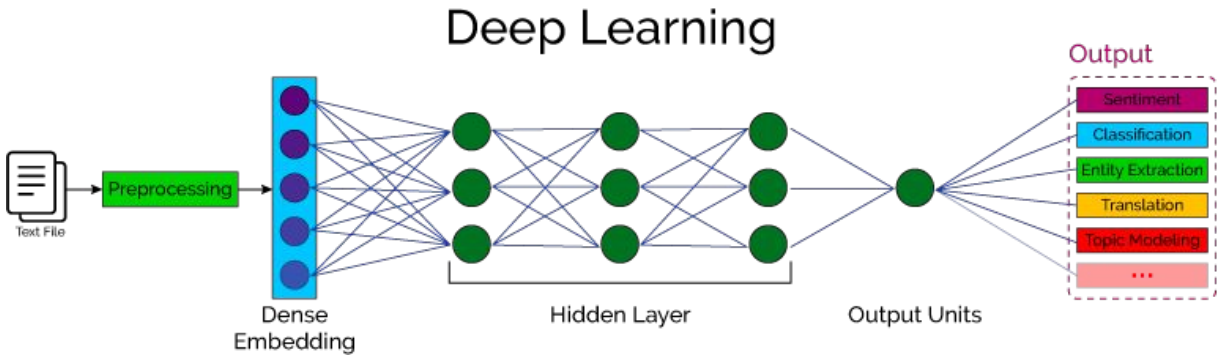




Natural Language Processing







Natural Language Processing

The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

Bag of words

```
corpus = [
    "This is the best book I have ever read",
    "Reading books is the best way to gain knowledge",
    "Which book you prefer. This one or the another one?",
    "Is this the book you love most?"
]
```

Make vocabulary

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
another	best	book	books	ever	gain	have	is	knowledge	love	most	one	or	prefer	read	reading	the	this	to	way	which	you

Score assignment

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	1	1	0	1	0	1	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0
0	1	0	1	0	1	0	1	1	0	0	0	0	0	0	1	1	0	1	1	0	0
1	0	1	0	0	0	0	0	0	0	0	2	1	1	0	0	1	1	0	0	1	1
0	0	1	0	0	0	0	1	0	1	1	0	0	0	0	0	1	1	0	0	0	1

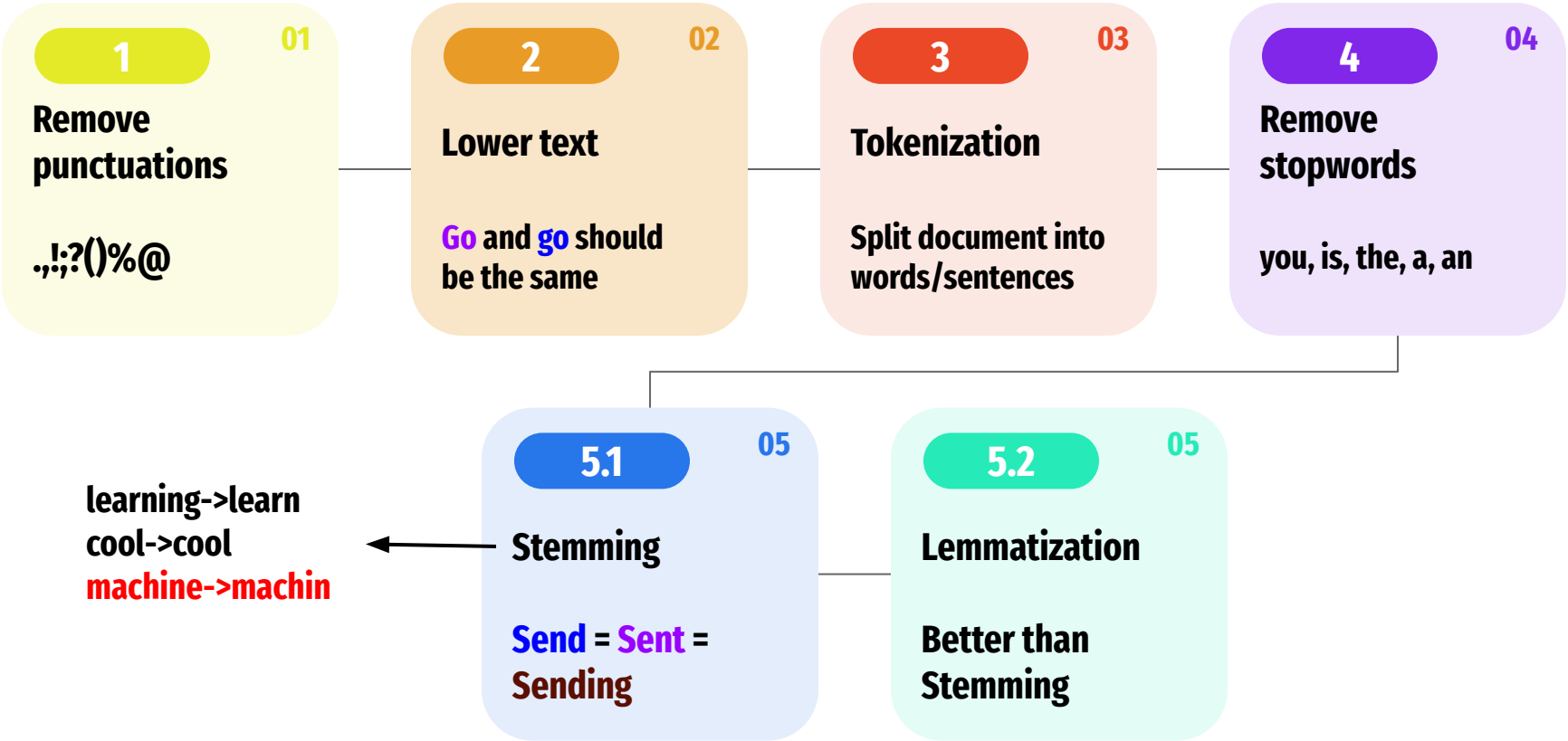
CountVectorizer

```python
from sklearn.feature_extraction.text import CountVectorizer

corpus = [
    "This is the best book I have ever read",
    "Reading books is the best way to gain knowledge",
    "Which book you prefer. This one or the another one?",
    "Is this the book you love most?"
]
vectorizer = CountVectorizer()
data = vectorizer.fit_transform(corpus)
vocabulary = vectorizer.get_feature_names_out()
print("Vocabulary:")
print(vocabulary)
result = data.toarray()
print("Bag of words:")
print(result)
```

example

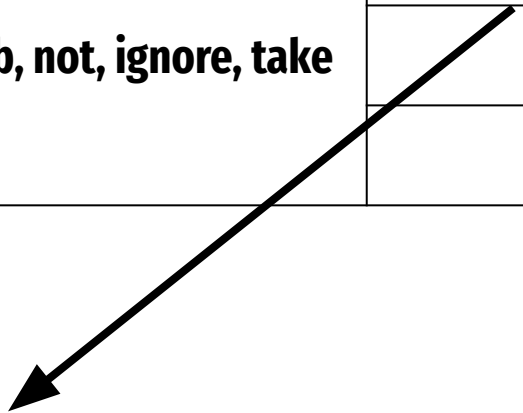
```
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\Viet\PycharmProjects\basic\example.py
Vocabulary:
['another' 'best' 'book' 'books' 'ever' 'gain' 'have' 'is' 'knowledge'
 'love' 'most' 'one' 'or' 'prefer' 'read' 'reading' 'the' 'this' 'to'
 'way' 'which' 'you']
Bag of words:
[[0 1 1 0 1 0 1 1 0 0 0 0 0 0 1 0 1 1 0 0 0 0]
 [0 1 0 1 0 1 0 1 1 0 0 0 0 0 0 1 1 0 1 1 0 0]
 [1 0 1 0 0 0 0 0 0 0 0 2 1 1 0 0 1 1 0 0 1 1]
 [0 0 1 0 0 0 0 1 0 1 1 0 0 0 0 0 1 1 0 0 0 1]]
```

Text preprocessing



Bag of words

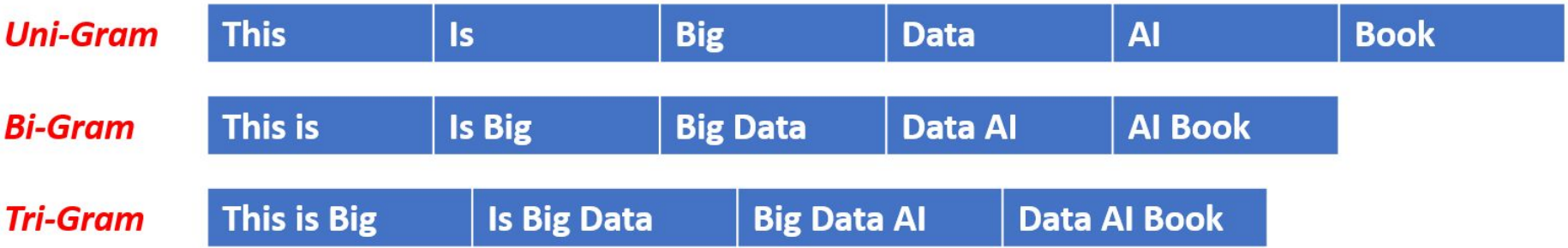
Document	Vocabulary (no stopwords)	Bag of words' vector
This is a good job. I will not ignore it	Good, job, not, ignore, take	1,1,1,1,0
This is not a good job. I will ignore it		1,1,1,1,0
This is a good job. I will take it		1,1,0,0,1



N-grams

N-grams

This is Big Data AI Book



CountVectorizer

```python
from sklearn.feature_extraction.text import CountVectorizer

corpus = [
    "This is the best book I have ever read",
    "Reading books is the best way to gain knowledge",
    "Which book you prefer. This one or the another one?",
    "Is this the book you love most?"
]
vectorizer = CountVectorizer(ngram_range=(1, 2), stop_words="english")
```

BoW

```
Vocabulary:
['another' 'another one' 'best' 'best book' 'best way' 'book' 'book have'
 'book you' 'books' 'books is' 'ever' 'ever read' 'gain' 'gain knowledge'
 'have' 'have ever' 'is' 'is the' 'is this' 'knowledge' 'love' 'love most'
 'most' 'one' 'one or' 'or' 'or the' 'prefer' 'prefer this' 'read'
 'reading' 'reading books' 'the' 'the another' 'the best' 'the book'
 'this' 'this is' 'this one' 'this the' 'to' 'to gain' 'way' 'way to'
 'which' 'which book' 'you' 'you love' 'you prefer']
Bag of words:
[[0 0 1 1 0 1 1 0 0 0 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0
  1 1 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 1 0 1 0 0 0 1 1 0 0 1 1 0 0 1 1 0 1 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0
  0 0 0 0 1 1 1 1 0 0 0 0 0]
 [1 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 1 1 1 1 1 0 0 0 1 1 0 0
  1 0 1 0 0 0 0 0 1 1 1 0 1]
 [0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 1
  1 0 0 1 0 0 0 0 0 0 1 1 0]]
```

Term Frequency-Inverse Document Frequency

$$TF = \frac{Number\ of\ times\ a\ word\ "X"\ appears\ in\ a\ Document}{Number\ of\ words\ present\ in\ a\ Document}$$

$$IDF = log\left(\frac{Number\ of\ Documents\ present\ in\ a\ Corpus}{Number\ of\ Documents\ where\ word\ "X"\ has\ appeared}\right)$$

$$TF\ IDF\ =\ TF * IDF$$

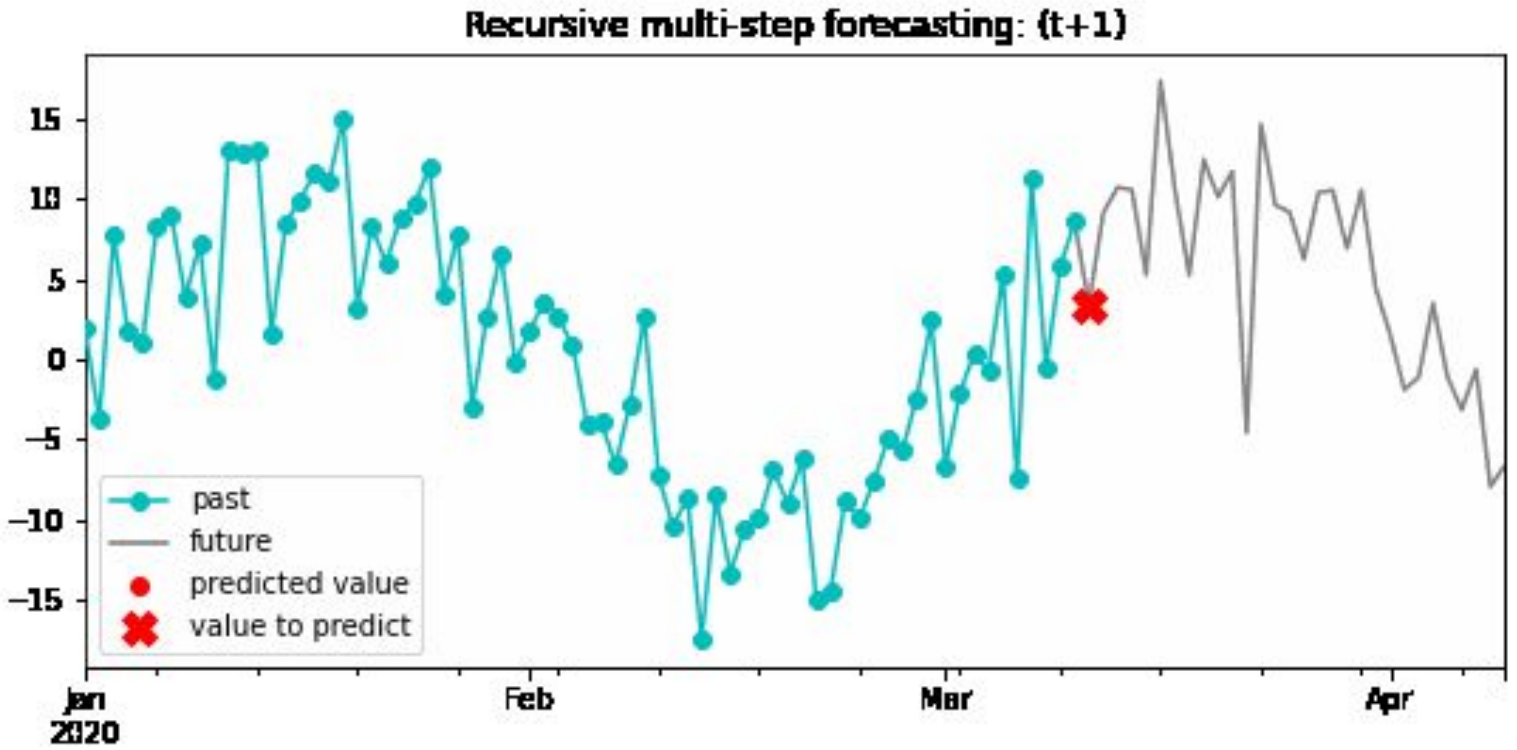
Term Frequency (TF)

Tần số xuất hiện của 1 word trong 1 document

Inverse Document Frequency (IDF)

Mức độ phổ biến/hiếm của 1 word trong toàn bộ các documents

Time series forecasting



Time series data

date	charter_foreign	total_foreign	charter_domestic	total_domestic
date	bigint	bigint	bigint	bigint
Date	Integer	Integer	Integer	Integer
2008-01-01T00:00:00.000Z	57463	5642057	174692	7394181
2008-02-01T00:00:00.000Z	56651	5053435	142931	6837201
2008-03-01T00:00:00.000Z	95878	6071551	176906	8439191
2008-04-01T00:00:00.000Z	100198	5731709	103054	7467710
2008-05-01T00:00:00.000Z	130720	6003335	88631	7840260
2008-06-01T00:00:00.000Z	196656	6467543	99579	8274573
2008-07-01T00:00:00.000Z	195873	6861571	130766	8971734
2008-08-01T00:00:00.000Z	152848	6895232	77253	8717085
2008-09-01T00:00:00.000Z	88267	5539617	40070	6371669
2008-10-01T00:00:00.000Z	76723	5564387	43003	6520788
2008-11-01T00:00:00.000Z	26671	5128350	50322	6223857
2008-12-01T00:00:00.000Z	30963	5624131	54894	6959043
2009-01-01T00:00:00.000Z	35174	5399479	60652	6805171

	Jan-17	Feb-17	Mar-17	Apr-17	May-17	Jun-17	Jul-17	Aug-17	Sep-17	Oct-17	Nov-17	Dec-17
Store 1	20 809	23 832	22 542	18 736	15 347	12 190	11 049	10 913	9 019	9 803	3 172	7 768
Store 2	14 834	14 170	13 109	7 560	6 388	8 124	6 719	3 847	3 187	8 145	10 007	10 036
Store 3	14 453	12 816	14 260	12 954	12 596	11 200	8 374	9 725	10 853	11 609	13 897	19 628
Store 4	8 590	8 822	13 497	13 073	13 278	14 705	13 550	13 962	14 783	11 712	13 234	17 790
Store 5	16 647	12 851	13 433	13 486	13 767	10 482	11 476	13 024	14 459	14 672	12 885	16 681
Store 6	7 323	5 043	4 857	2 584	6 990	4 130	45 110	4 895	5 708	5 106	4 334	10 741
Store 7	8 735	4 645	4 905	4 536	5 385	5 040	6 341	9 403	9 904	9 521	13 473	7 077
Store 8	965	1 254	1 937	1 940	1 756	1 690	1 536	2 502	2 806	4 087	3 704	5 478
Store 9	188	4 375	5 109	5 255	5 667	6 331	9 182	9 867	8 948	10 096	10 860	10 951
Store 10	14 033	30 900	18 864	12 828	10 269	8 464	7 674	4 160	5 497	9 608	16 177	15 798
Store 11	8 185	9 090	12 560	13 682	10 061	8 172	8 045	6 622	7 240	9 923	9 403	10 768
Store 12	911	384	658	949	1 236	1 405	1 366	1 433	1 282	1 515	1 457	2 933
Store 13	2 361	2 830	2 892	2 953	2 929	2 982	6 361	25 969	22 952	27 171	27 199	35 298
Store 14	9 805	8 436	12 878	9 580	6 665	5 069	6 003	6 139	10 987	11 039	11 252	13 844

Code	Name	Population|1990	Population|2000	Population|2010	Population Change|1980-1990	Population Change|1990-2000
AK	Alaska	550043	626932	710231	36.9	14
AL	Alabama	4040587	4447100	4779736	3.8	10.1
AR	Arkansas	2350725	2673400	2915918	2.8	13.7
AZ	Arizona	3665228	5130632	6392017	34.8	40
CA	California	29760021	33871648	37253956	25.7	13.8
CO	Colorado	3294394	4301261	5029196	14	30.6
CT	Connecticut	3287116	3405565	3574097	5.8	3.6
DC	District of Columbia	606900	572059	601723	-4.9	-5.7
DE	Delaware	666168	783600	897934	12.1	17.6
FL	Florida	12937926	15982378	18801310	32.7	23.5
GA	Georgia	6478216	8186453	9687653	18.6	26.4
HI	Hawaii	1108229	1211537	1360301	14.9	9.3
IA	Iowa	2776755	2926324	3046355	-4.7	5.4
ID	Idaho	1006749	1293953	1567582	6.7	28.5
IL	Illinois	11430602	12419293	12830632	0	8.6
IN	Indiana	5544159	6080485	6483802	1	9.7
KS	Kansas	2477574	2688418	2853118	4.8	8.5
KY	Kentucky	3685296	4041769	4339367	0.7	9.7
LA	Louisiana	4219973	4468976	4533372	0.3	5.9

Time series data

1	2	3	4	5	6	7	8	9	10

Time series

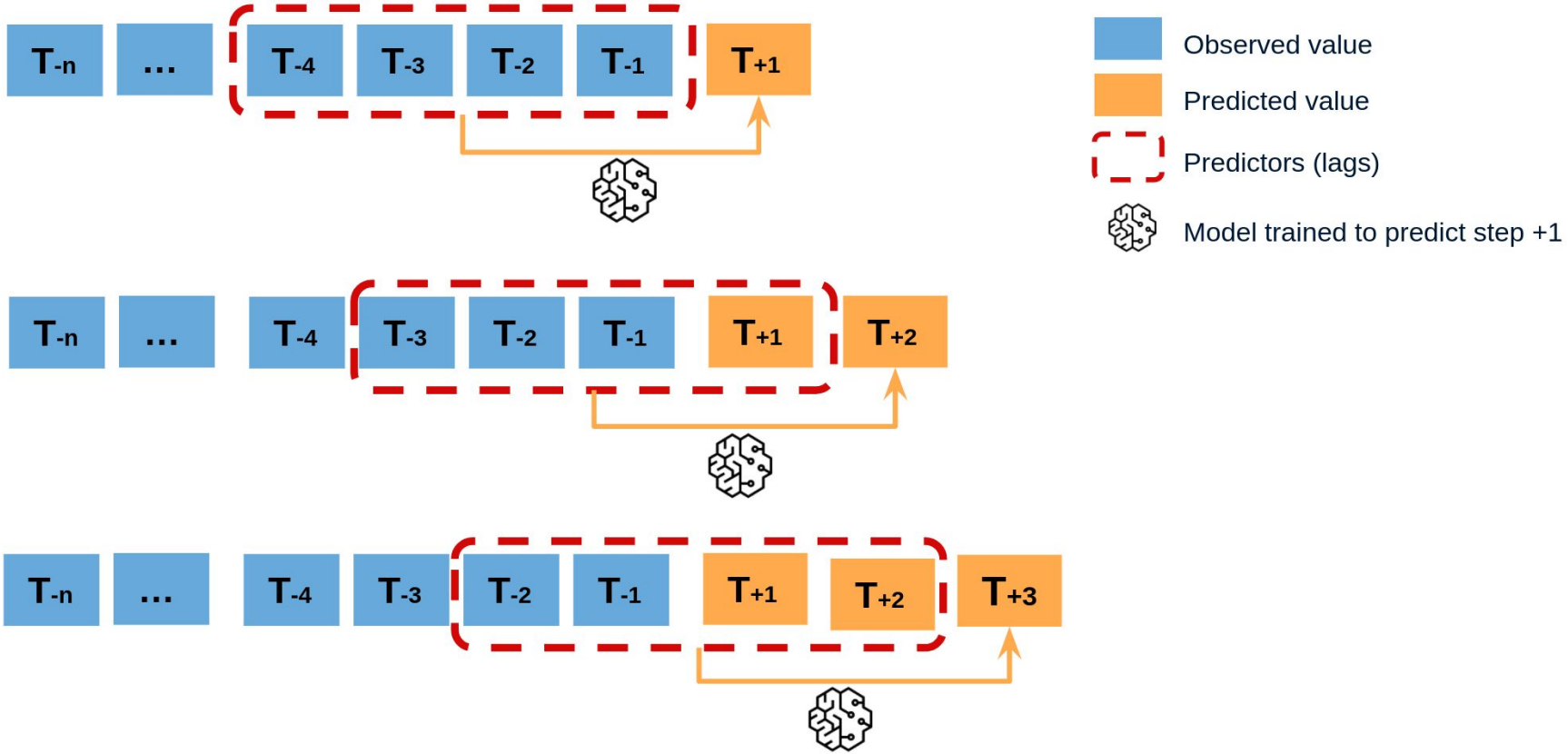
1	2	3	4	5	6	7	8	9	10

Exogenous variable

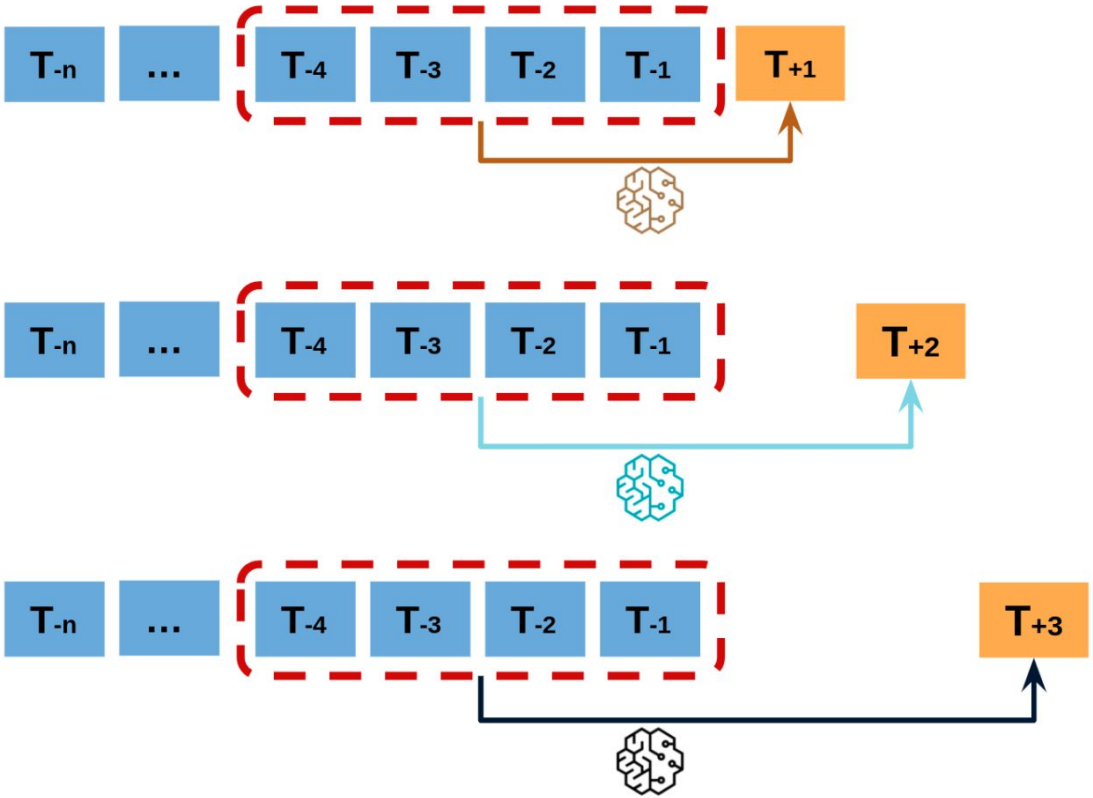
a	b	c	d	e	f	g	h	i	j

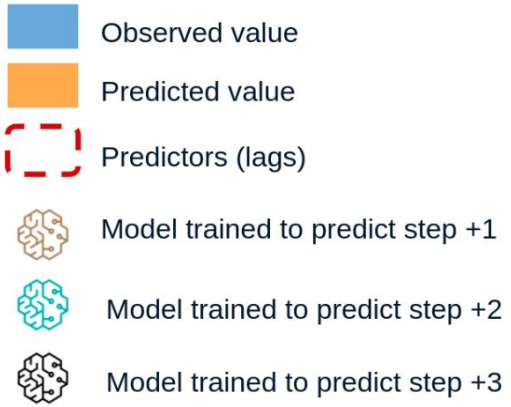
X						y
1	2	3	4	5	f	6
2	3	4	5	6	g	7
3	4	5	6	7	h	8
4	5	6	7	8	i	9
5	6	7	8	9	j	10

Recursive multi-step Time series Forecasting

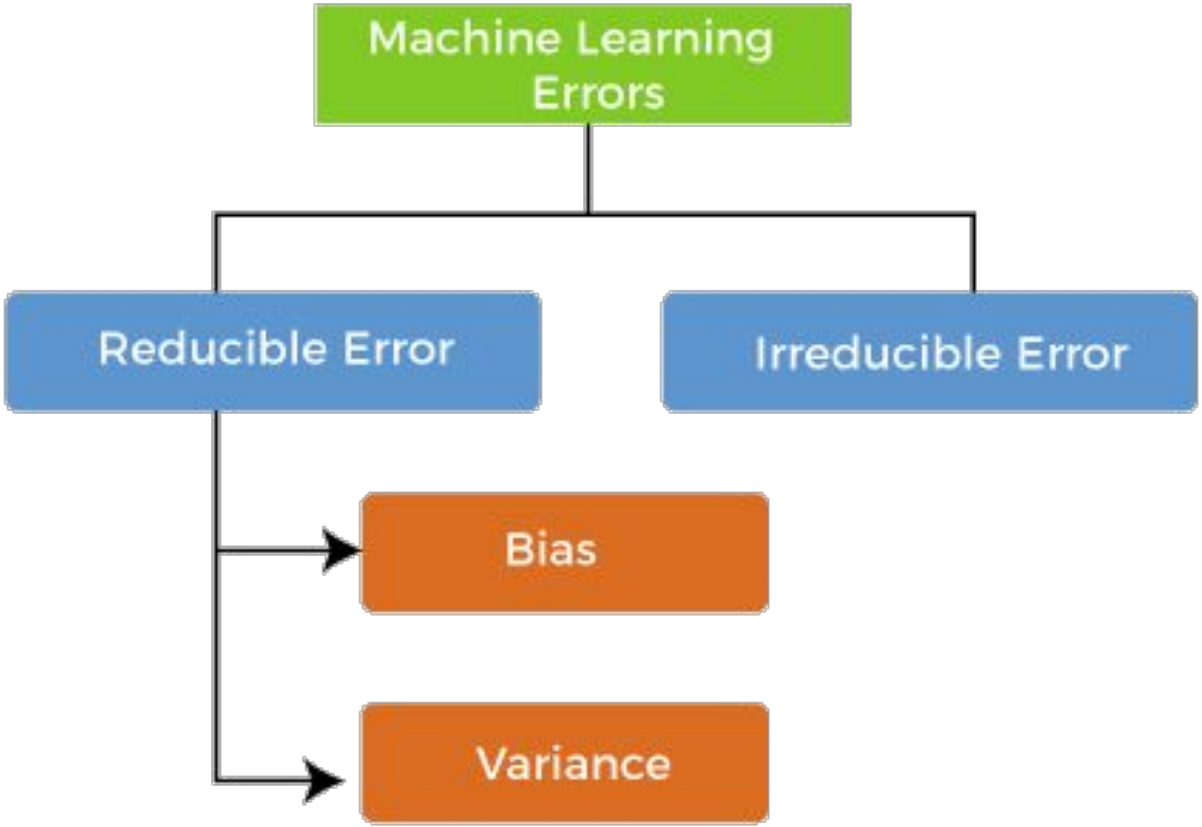


Direct multi-step Time series Forecasting





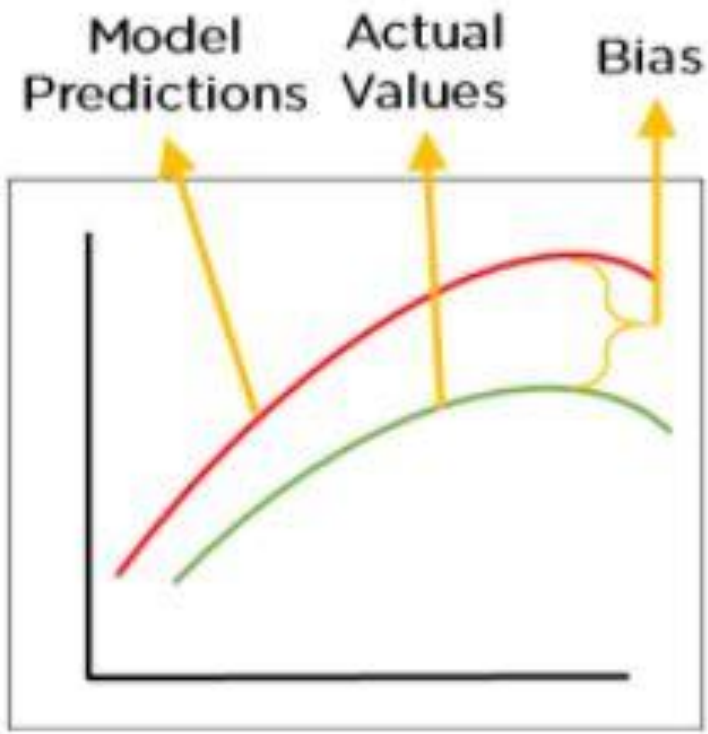
Error in Machine Learning



Error in Machine Learning

Bias

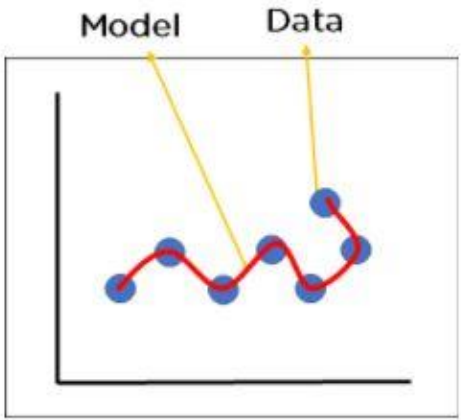
- Sai lệch giữa prediction và actual label
- Cho ta biết khả năng dự đoán chính xác của mô hình
- Bias càng bé càng tốt
- High bias means:
  - Overly-simplified model
  - Under-fitting
  - High error on both training and test sets

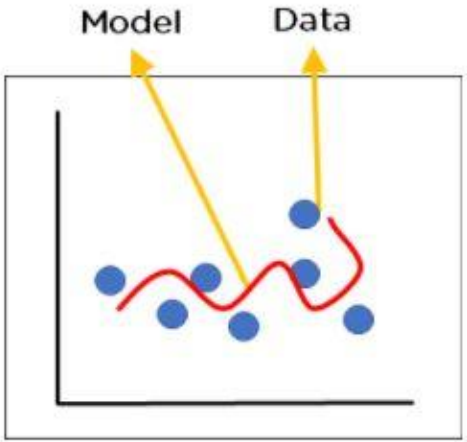


Error in Machine Learning

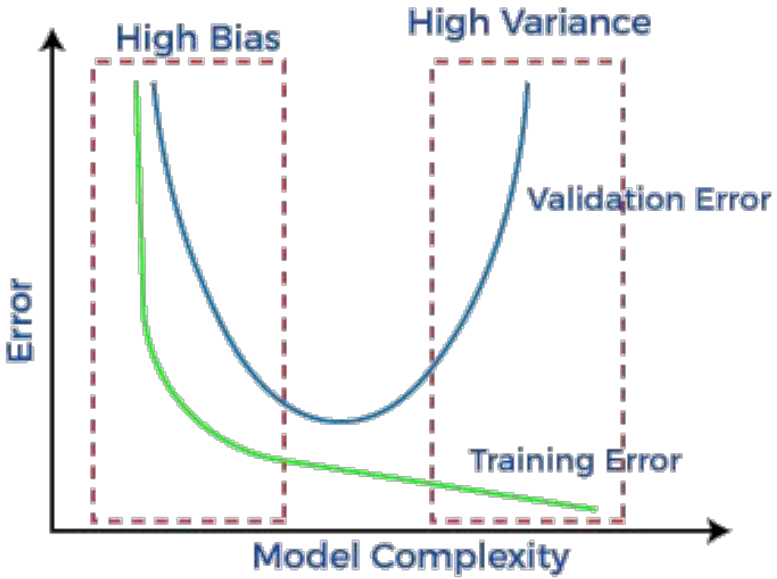
Variance

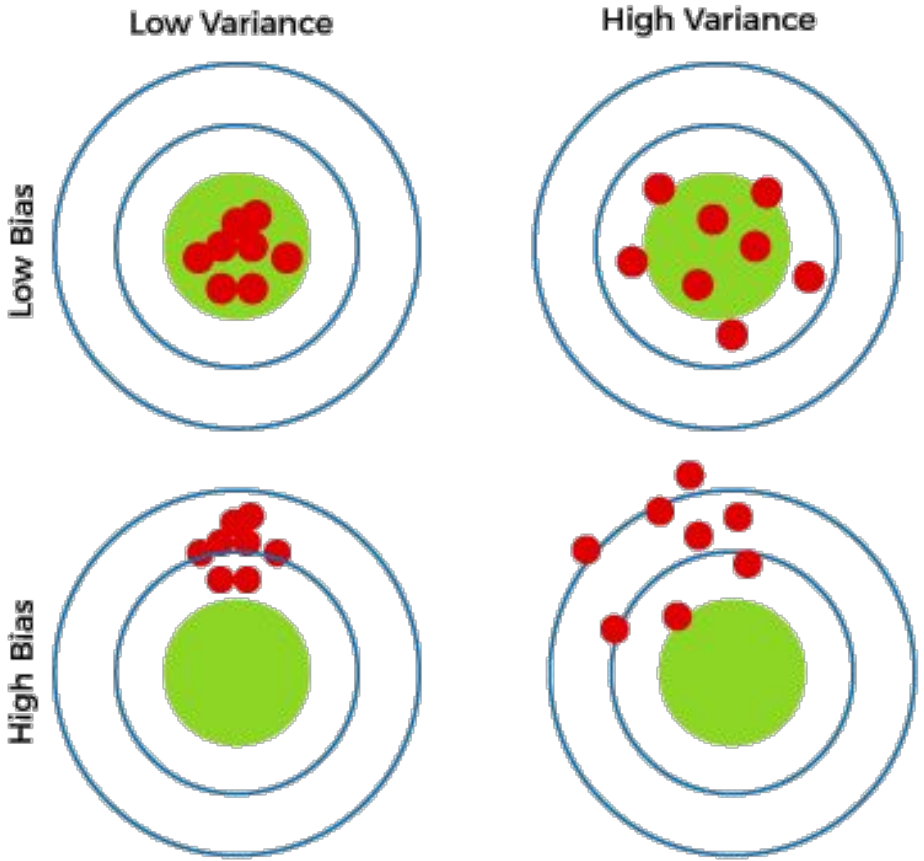
- Mức độ thay đổi của độ chính xác của prediction nếu input thay đổi
- Cho ta biết khả năng tổng quát hóa của mô hình
- Variance càng bé càng tốt
- High variance means:
  - Overly-complex model
  - Over-fitting
  - Low error on training set but high error on test set

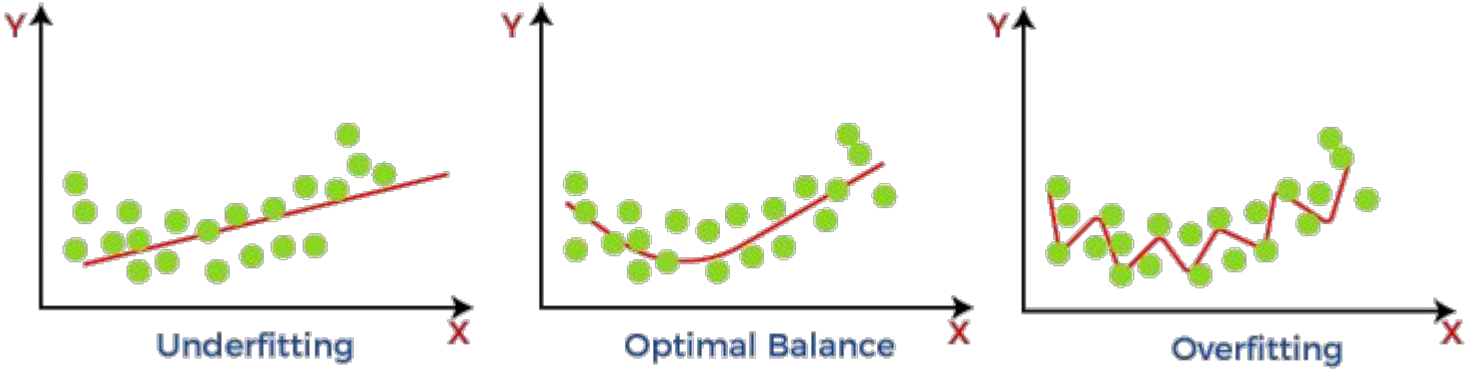


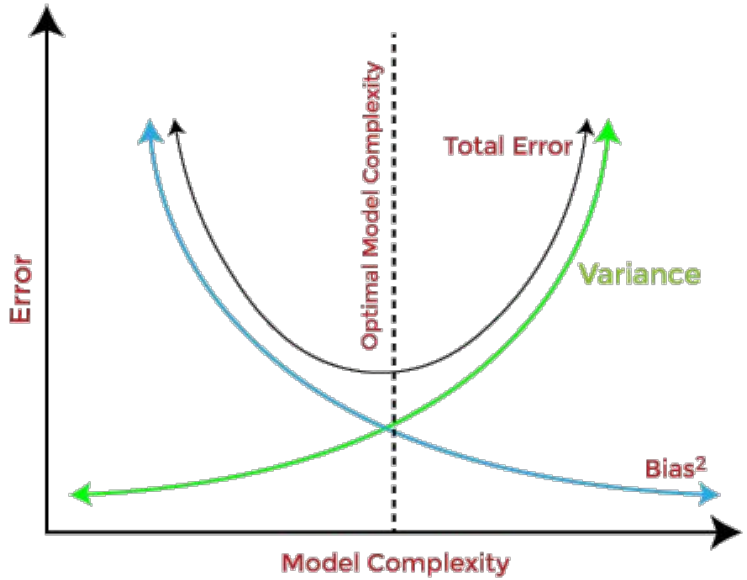


Bias-Variance tradeoff






Y
X
Underfitting
Y
X
Optimal Balance
Y
X
Overfitting


Error
Optimal Model Complexity
Total Error
Variance
Bias²
Model Complexity

Choose Machine Learning model based on data

Low bias + high variance

- Decision tree
- Random Forest
- K-nearest neighbours
- Kernel SVM

-> Phù hợp khi có nhiều data và ít features

High bias + low variance

- Linear regression
- Logistic regression
- Linear SVM

-> Phù hợp khi có ít data và nhiều features

Extra

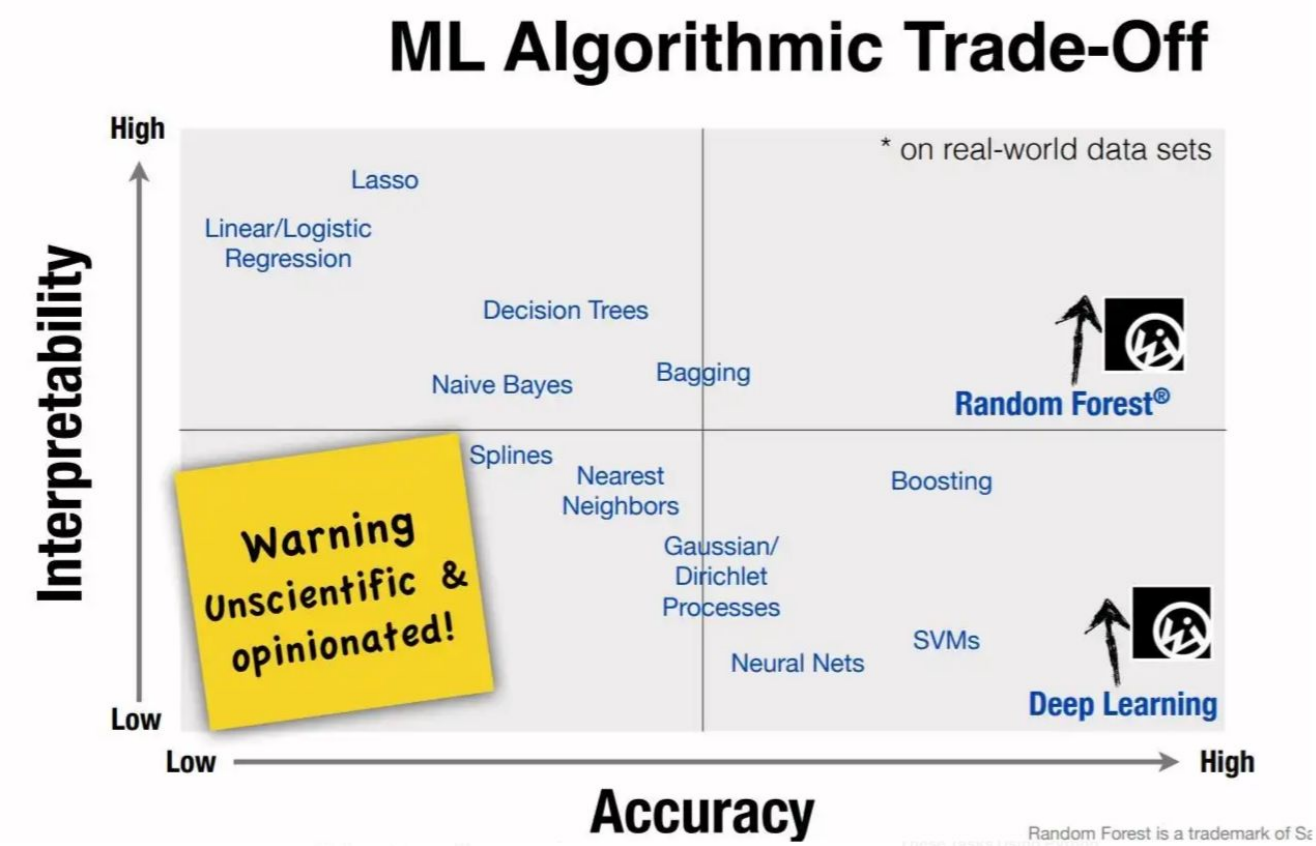
Nếu có nhiều data và nhiều features

- Giảm chiều dữ liệu (e.g. PCA)
- Sử dụng Neural network

Choose Machine Learning model based on business domain

- **Which metric is important?**
  - Accuracy, precision, recall, F1, ...
- **What is the priority?**
  - Speed: self-driving car app need to be realtime -> fast
  - Memory: Model deployed in embedded device needs to be small
  - Accuracy: In medical field, accuracy is the most important factor
- **Interpretability ?**
  - Do you need to explain result?
  - Do you need to find out which features are important?

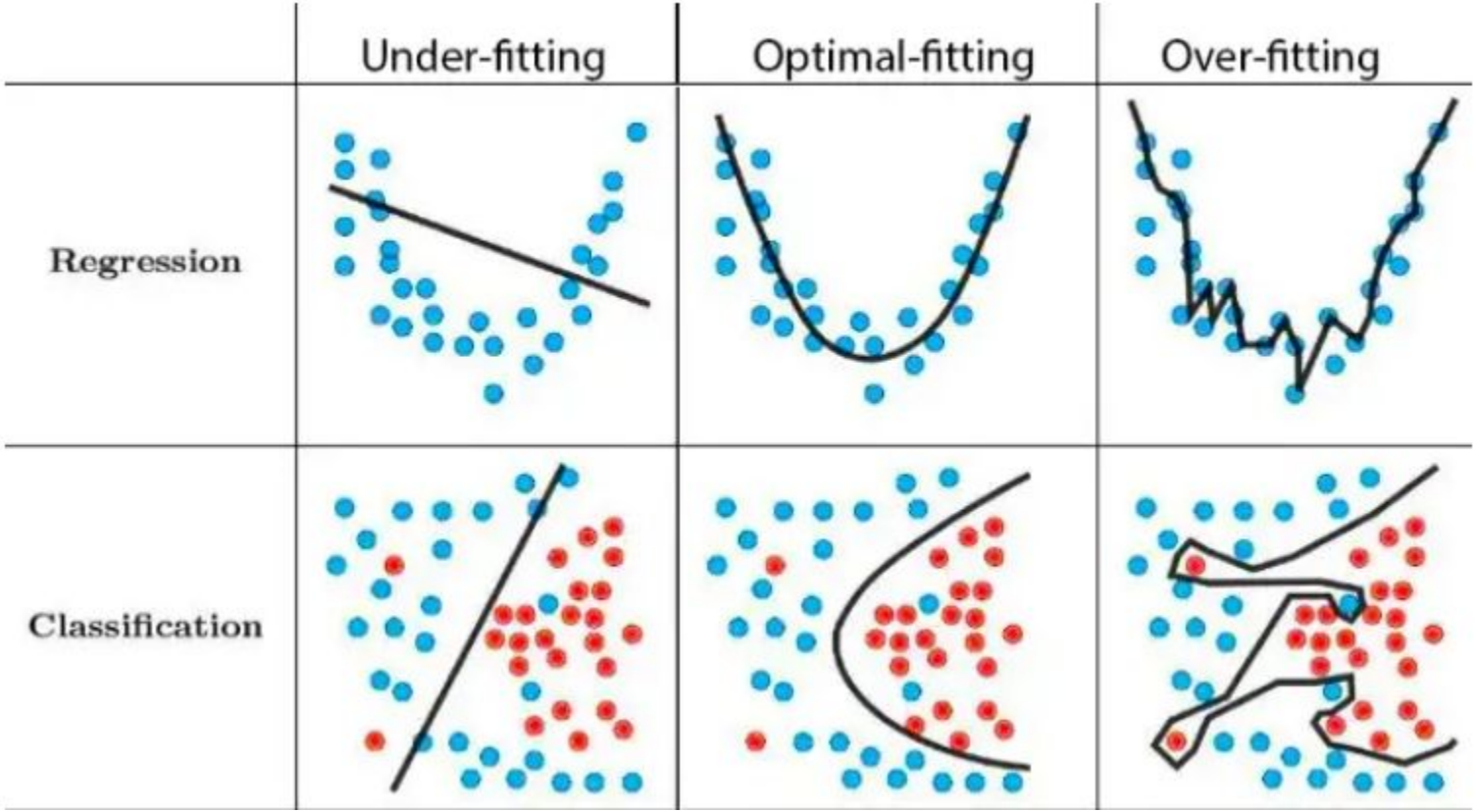
Machine Learning algorithm's interpretability



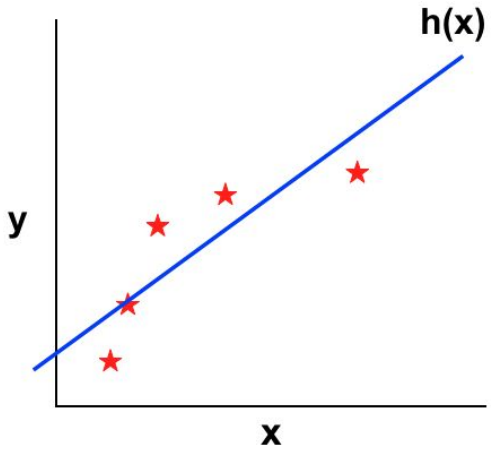
Tips for choosing Machine Learning model

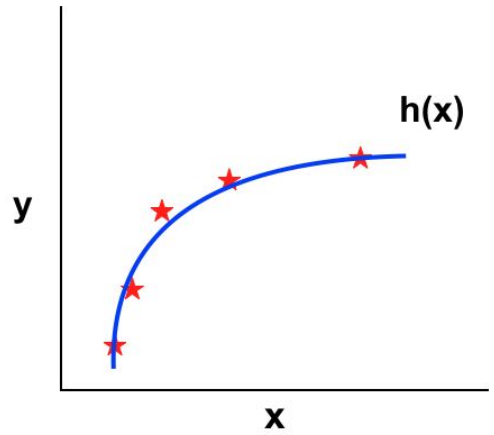
- Start with a simple model
  - Choose the simplest model first
  - If it is good enough, you even do not need to try another model
- Try different models and shortlist the best ones?
- Do Hyperparameter Tuning for each models?
  - GridSearchCV if the number of combinations is small
  - RandomizedSearchCV if the number of combination is large
- Compare amongst the best models with best hyperparameters to pick up the best one

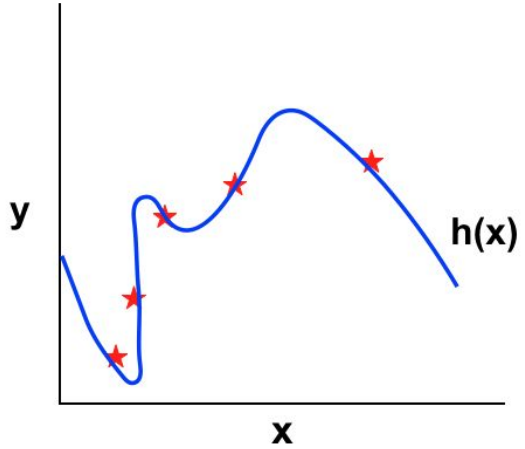
Underfitting and Overfitting



Regularization







$$L(x,y) = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

$$f(x_i) = h_\theta x = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3 + \theta_4 x_4^4$$

$$f(x_i) = h_\theta x = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3 + \theta_4 x_4^4$$

$$f(x_i) = h_\theta x = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2$$

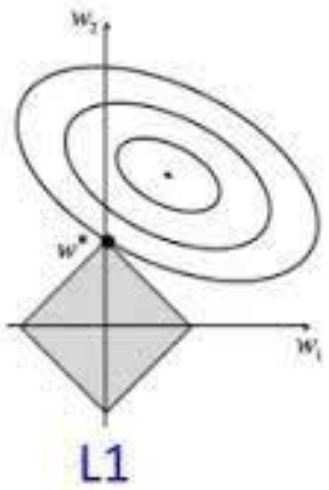
Regularization

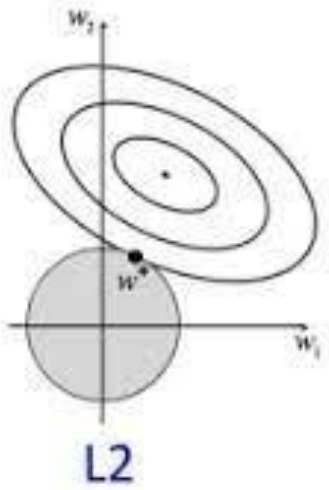
L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij} W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

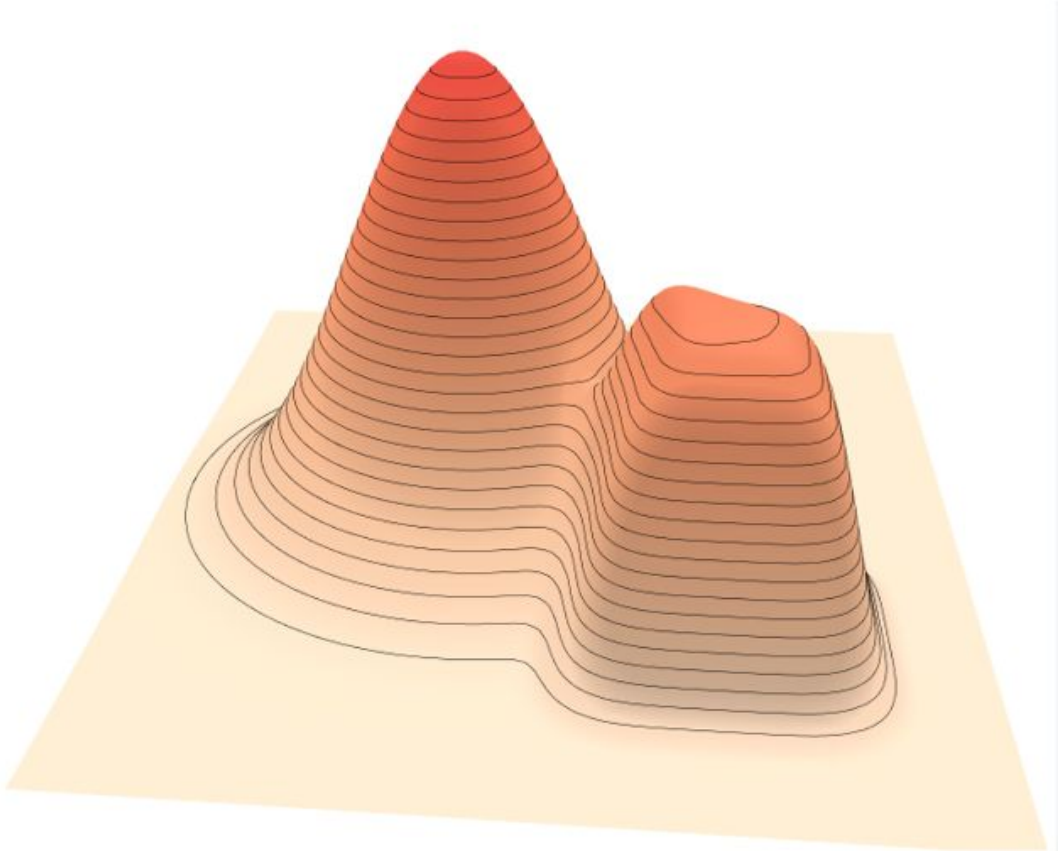
L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij} W_j)^2}_{\text{Loss function}} + \underbrace{\lambda \sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$

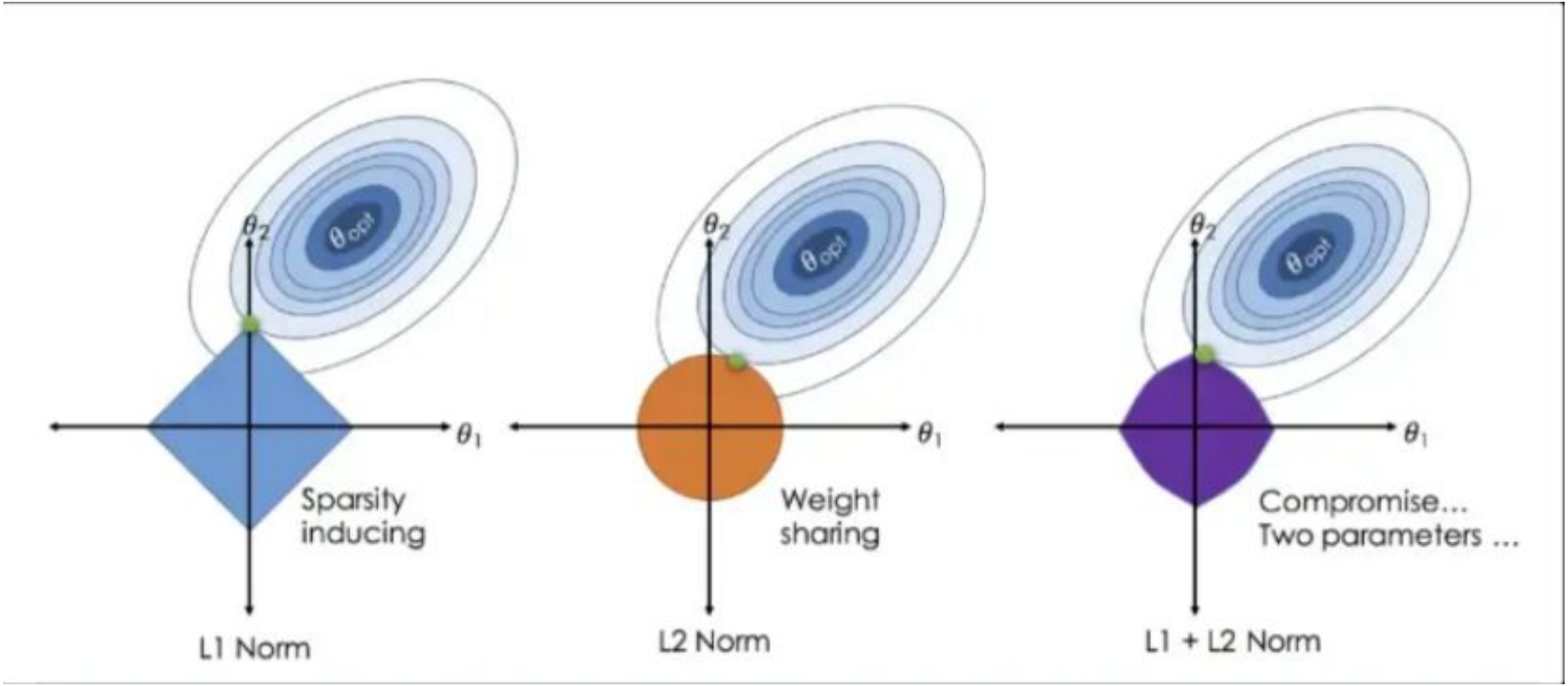




Regularization



Regularization



Regularization Example

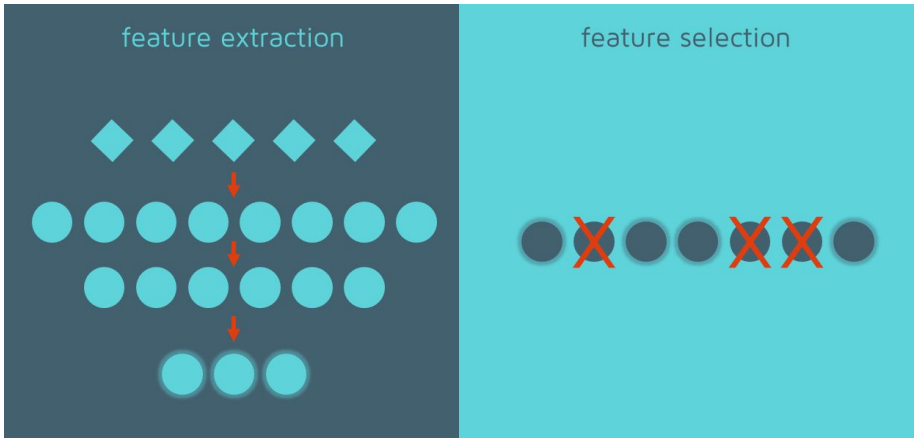
```python
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

data = pd.read_csv("diabetes.csv")
x = data.drop("Outcome", axis=1)
y = data["Outcome"]
cls = Pipeline(steps=[
    ('scaler', StandardScaler()),
    ('model', LogisticRegression(penalty="l1", solver="liblinear"))
])
cls.fit(x, y)
for i, j in zip(x.columns, cls["model"].coef_[0]):
    print("{} is with coefficient of {}".format(i, j))
```

```
fuel_viz
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\Viet\PycharmPr
Pregnancies is with coefficient of 0.4047757715484604
Glucose is with coefficient of 1.1020534150332097
BloodPressure is with coefficient of -0.23905878835657515
SkinThickness is with coefficient of 0.0
Insulin is with coefficient of -0.12014123606887196
BMI is with coefficient of 0.6883338771775394
DiabetesPedigreeFunction is with coefficient of 0.30157785094254597
Age is with coefficient of 0.16766878675897076
```

```python
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

data = pd.read_csv("diabetes.csv")
x = data.drop("Outcome", axis=1)
y = data["Outcome"]
cls = Pipeline(steps=[
    ('scaler', StandardScaler()),
    ('model', LogisticRegression(penalty="l2", solver="lbfgs"))
])
cls.fit(x, y)
for i, j in zip(x.columns, cls["model"].coef_[0]):
    print("{} is with coefficient of {}".format(i, j))
```

```
fuel_viz
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\Viet\PycharmP
Pregnancies is with coefficient of 0.4086468712956544
Glucose is with coefficient of 1.107111971571363
BloodPressure is with coefficient of -0.2508680388517632
SkinThickness is with coefficient of 0.00905045508333392
Insulin is with coefficient of -0.13083627394872085
BMI is with coefficient of 0.6963087173103848
DiabetesPedigreeFunction is with coefficient of 0.3088372414180195
Age is with coefficient of 0.1764978217442397
```

Dimensionality Reduction



01

Feature Selection

Original features are maintained

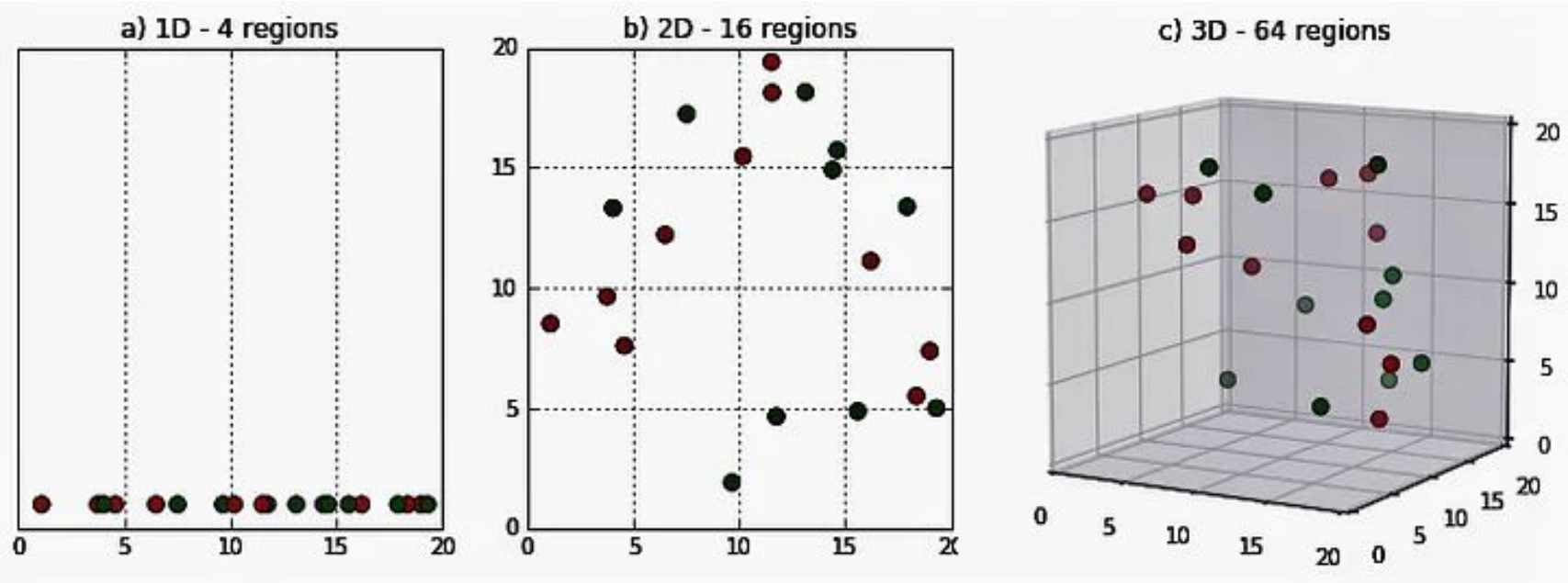
Keep most important features

02

Feature Extraction

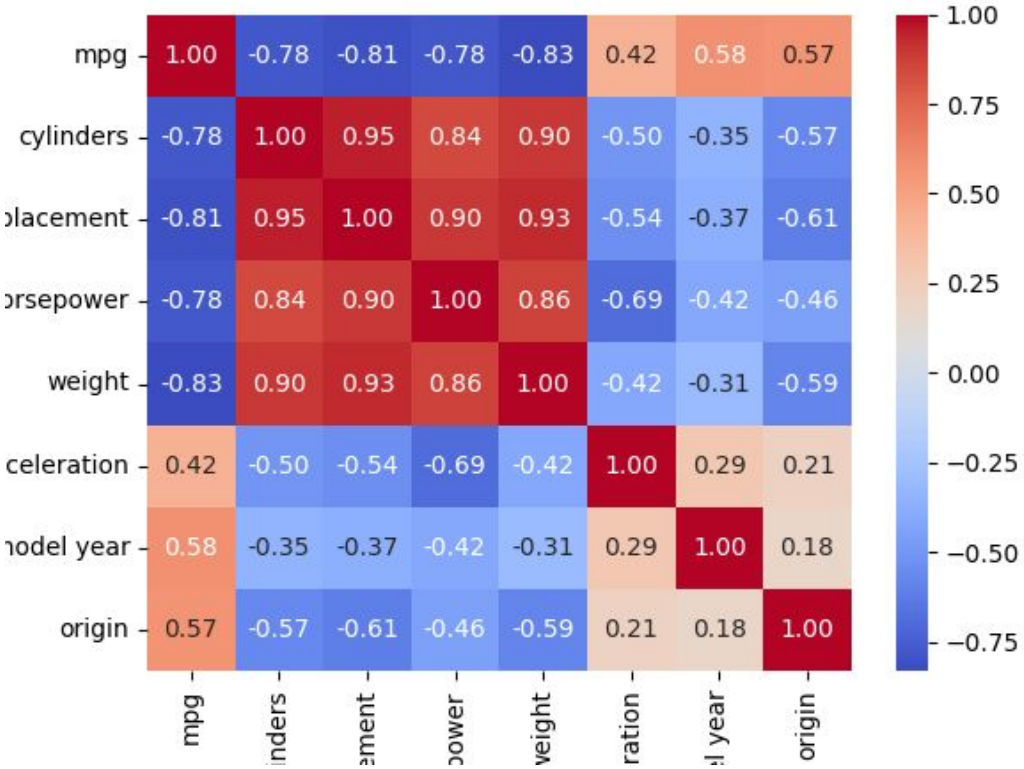
Features are transformed to a new space

Curse of Dimensionality



Feature Selection

Feature Selection - Correlation Coefficient



Feature Selection - Variance Threshold

```python
import pandas as pd
from sklearn.feature_selection import VarianceThreshold

data = pd.read_csv("diabetes.csv", usecols=[i for i in range(8)])
variance_threshold = VarianceThreshold(threshold=1)
variance_threshold.fit(data)
print(variance_threshold.variances_)
print(variance_threshold.get_support())
```

fuel_viz

```
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\Viet\Pycharm
[1.13392724e+01 1.02091726e+03 3.74159449e+02 2.54141900e+02
 1.32638869e+04 6.20790465e+01 1.09635697e-01 1.38122964e+02]
[ True  True  True  True  True  True False  True]
```

Feature Selection - Lasso (L1)

```python
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

data = pd.read_csv("diabetes.csv")
x = data.drop("Outcome", axis=1)
y = data["Outcome"]
cls = Pipeline(steps=[
    ('scaler', StandardScaler()),
    ('model', LogisticRegression(penalty="l1", solver="liblinear"))
])
cls.fit(x, y)
for i, j in zip(x.columns, cls["model"].coef_[0]):
    print("{} is with coefficient of {}".format(i, j))
```

```
fuel_viz
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\Viet\PycharmPr
Pregnancies is with coefficient of 0.4047757715484604
Glucose is with coefficient of 1.1020534150332097
BloodPressure is with coefficient of -0.23905878835657515
SkinThickness is with coefficient of 0.0
Insulin is with coefficient of -0.12014123606887196
BMI is with coefficient of 0.6883338771775394
DiabetesPedigreeFunction is with coefficient of 0.30157785094254597
Age is with coefficient of 0.16766878675897076
```

```python
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

data = pd.read_csv("diabetes.csv")
x = data.drop("Outcome", axis=1)
y = data["Outcome"]
cls = Pipeline(steps=[
    ('scaler', StandardScaler()),
    ('model', LogisticRegression(penalty="l2", solver="lbfgs"))
])
cls.fit(x, y)
for i, j in zip(x.columns, cls["model"].coef_[0]):
    print("{} is with coefficient of {}".format(i, j))
```

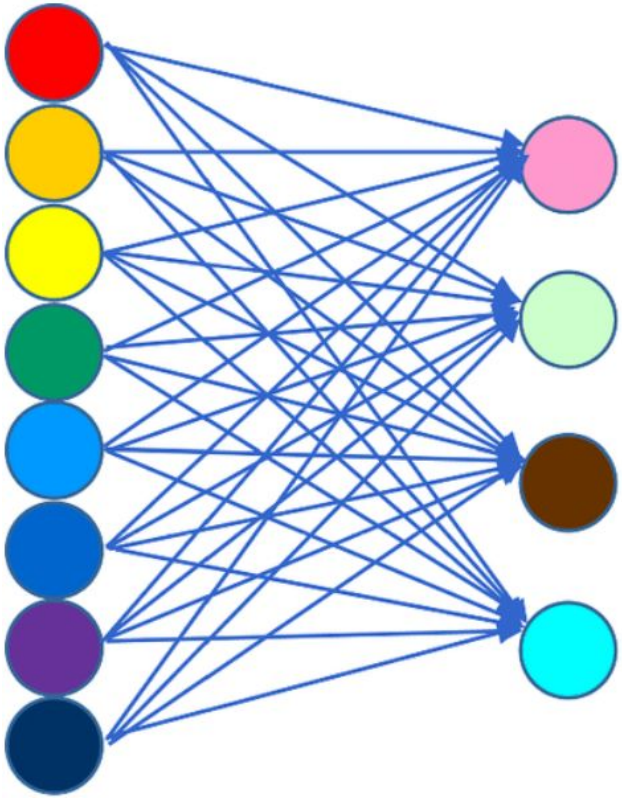
```
fuel_viz
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\Viet\PycharmP
Pregnancies is with coefficient of 0.4086468712956544
Glucose is with coefficient of 1.107111971571363
BloodPressure is with coefficient of -0.2508680388517632
SkinThickness is with coefficient of 0.00905045508333392
Insulin is with coefficient of -0.13083627394872085
BMI is with coefficient of 0.6963087173103848
DiabetesPedigreeFunction is with coefficient of 0.3088372414180195
Age is with coefficient of 0.1764978217442397
```

Feature Selection - RandomForest

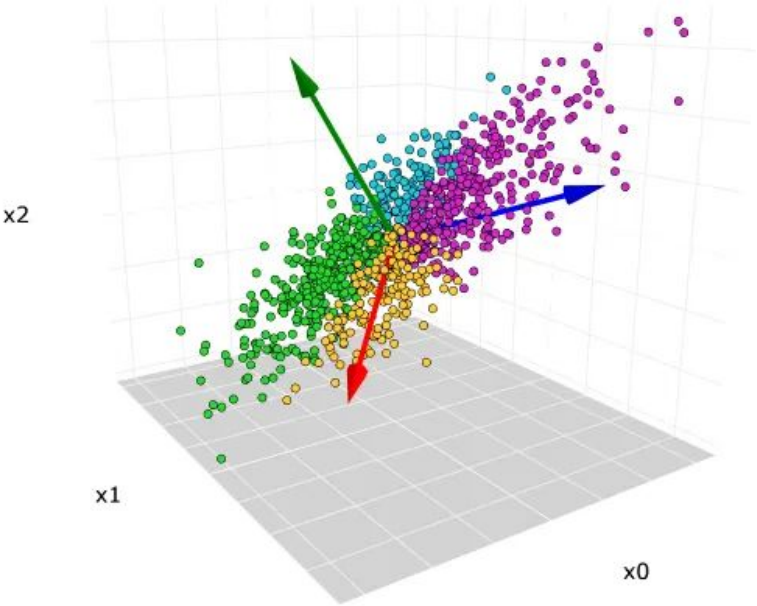
```python
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

data = pd.read_csv("diabetes.csv")
x = data.drop("Outcome", axis=1)
y = data["Outcome"]
cls = Pipeline(steps=[
    ('scaler', StandardScaler()),
    ('model', RandomForestClassifier())
])
cls.fit(x, y)
print(cls["model"].feature_importances_)
```

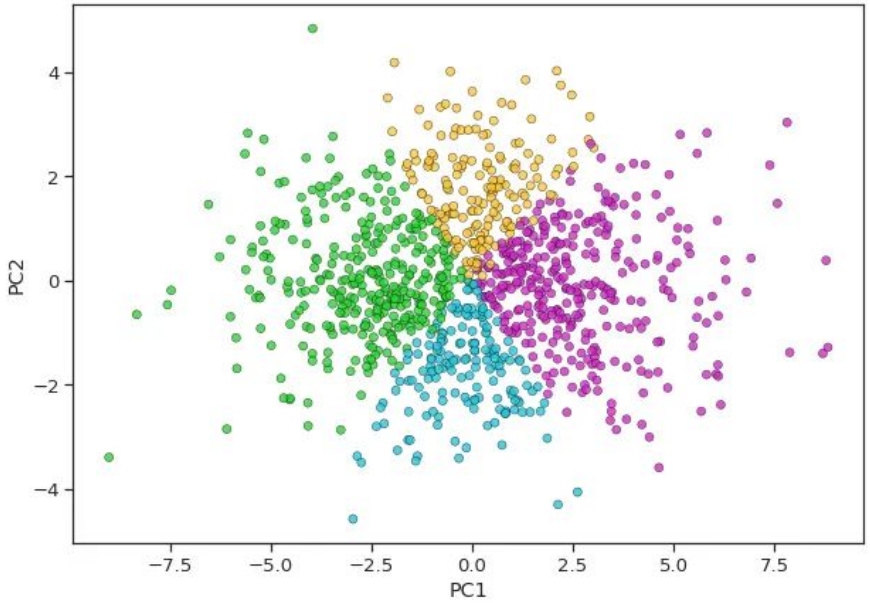
```
fuel_viz
C:\Users\Viet\anaconda3\envs\basic\python.exe C:\Users\Viet\Pycharm
[0.08485069 0.26544363 0.09088871 0.07260613 0.07047188 0.15482689
 0.12826565 0.13264643]
```

Feature Extraction



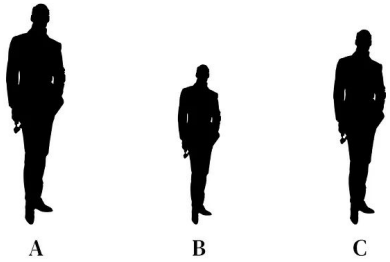
Feature Extraction - Principle Component Analyst



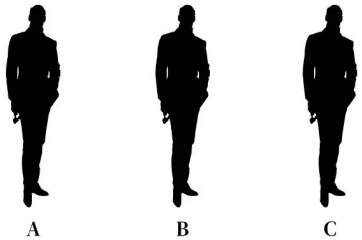


Variance in data

$$\begin{aligned}\operatorname{Var}(X) &= \mathrm{E}\left[(X - \mathrm{E}[X])^2\right] \\ &= \mathrm{E}\left[X^2 - 2X\,\mathrm{E}[X] + \mathrm{E}[X]^2\right] \\ &= \mathrm{E}\left[X^2\right] - 2\,\mathrm{E}[X]\,\mathrm{E}[X] + \mathrm{E}[X]^2 \\ &= \mathrm{E}\left[X^2\right] - \mathrm{E}[X]^2\end{aligned}$$



Person	Height (cm)
Alex	145
Ben	160
Chris	185



Person	Height (cm)
Daniel	172
Elsa	173
Fernandez	171

Covariance vs Correlation

$$\begin{aligned}\operatorname{Var}(X) &= \mathrm{E}\left[(X - \mathrm{E}[X])^2\right] \\ &= \mathrm{E}\left[X^2 - 2X\,\mathrm{E}[X] + \mathrm{E}[X]^2\right] \\ &= \mathrm{E}\left[X^2\right] - 2\,\mathrm{E}[X]\,\mathrm{E}[X] + \mathrm{E}[X]^2 \\ &= \mathrm{E}\left[X^2\right] - \mathrm{E}[X]^2\end{aligned}$$

$$\begin{aligned}\operatorname{cov}(X, Y) &= \mathrm{E}[(X - \mathrm{E}[X])\,(Y - \mathrm{E}[Y])] \\ &= \mathrm{E}[XY - X\,\mathrm{E}[Y] - \mathrm{E}[X]Y + \mathrm{E}[X]\,\mathrm{E}[Y]] \\ &= \mathrm{E}[XY] - \mathrm{E}[X]\,\mathrm{E}[Y] - \mathrm{E}[X]\,\mathrm{E}[Y] + \mathrm{E}[X]\,\mathrm{E}[Y] \\ &= \mathrm{E}[XY] - \mathrm{E}[X]\,\mathrm{E}[Y],\end{aligned}$$

$$\mathrm{Corr}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sigma_x \sigma_y}$$

Covarianced normalized by Standard Deviation

Correlation between X and Y

Standard deviation of X

Standard deviation of Y

$$\textbf{Standard Deviation} = \sqrt{\boldsymbol{Variance}}$$

Covariance vs Correlation

Covariance	Correlation
Indicates the direction of the linear relationship between variables	Indicates both the strength and direction of the linear relationship between two variables
Covariance values are not standard	Correlation values are standardized
Positive number being positive relationship and negative number being negative relationship	1 being strong positive correlation, -1 being strong negative correlation
Value between positive infinity to negative infinity	Value is strictly between -1 to 1

PCA: step 1 - Standardize the dataset

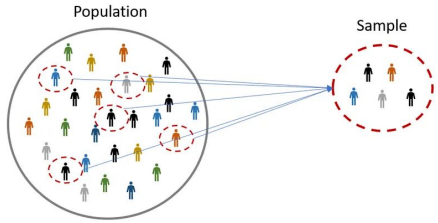
f1	f2	f3	f4
1	2	3	4
5	5	6	7
1	4	2	3
5	3	2	1
8	1	2	2

$$x_{new} = \frac{x - \mu}{\sigma}$$

		f1	f2	f3	f4
μ	=	4	3	3	3.4
σ	=	3	1.58114	1.73205	2.30217

f1	f2	f3	f4
-1	-0.63246	0	0.26062
0.33333	1.26491	1.73205	1.56374
-1	0.63246	-0.57735	-0.17375
0.33333	0	-0.57735	-1.04249
1.33333	-1.26491	-0.57735	-0.60812

PCA: step 2 - Calculate the covariance matrix



f1	f2	f3	f4
-1	-0.63246	0	0.26062
0.33333	1.26491	1.73205	1.56374
-1	0.63246	-0.57735	-0.17375
0.33333	0	-0.57735	-1.04249
1.33333	-1.26491	-0.57735	-0.60812

For Population

$$Cov(x,y) = \frac{\Sigma (x_i - \bar{x}) * (y_i - \bar{y})}{N}$$

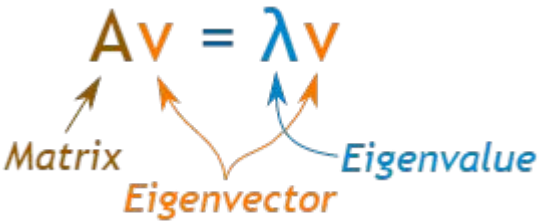
For Sample

$$Cov(x,y) = \frac{\Sigma (x_i - \bar{x}) * (y_i - \bar{y})}{(N-1)}$$

	f1	f2	f3	f4
f1	var(f1)	cov(f1,f2)	cov(f1,f3)	cov(f1,f4)
f2	cov(f2,f1)	var(f2)	cov(f2,f3)	cov(f2,f4)
f3	cov(f3,f1)	cov(f3,f2)	var(f3)	cov(f3,f4)
f4	cov(f4,f1)	cov(f4,f2)	cov(f4,f3)	var(f4)

	f1	f2	f3	f4
f1	0.8	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8	0.51121	0.4945
f3	0.03849	0.51121	0.8	0.75236
f4	-0.14479	0.4945	0.75236	0.8

PCA: step 3 - Calculate eigenvector and eigenvalue



Definition. Let A be an $n \times n$ matrix.

1. An ***eigenvector*** of A is a *nonzero* vector v in R^n such that $Av = \lambda v$, for some scalar λ.
2. An ***eigenvalue*** of A is a scalar λ such that the equation $Av = \lambda v$ has a *nontrivial* solution.

If $Av = \lambda v$ for $v \neq 0$, we say that λ is the ***eigenvalue for*** v, and that v is an ***eigenvector for*** λ.

$$A = \begin{bmatrix} 1 & 4 \\ 3 & 2 \end{bmatrix}$$

$$\det \begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc$$

$$\det(A - \lambda I) = 0$$

$$\det\left(\begin{bmatrix} 1 & 4 \\ 3 & 2 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) = 0$$

$$\det\left(\begin{bmatrix} 1-\lambda & 4 \\ 3 & 2-\lambda \end{bmatrix}\right) = 0$$

$$(1-\lambda)(2-\lambda) - 12 = 0$$

$$\lambda^2 - 3\lambda - 10 = 0$$

$$(\lambda - 5)(\lambda + 2) = 0$$

$$\lambda = 5, -2$$

PCA: step 3 - Calculate eigenvector and eigenvalue

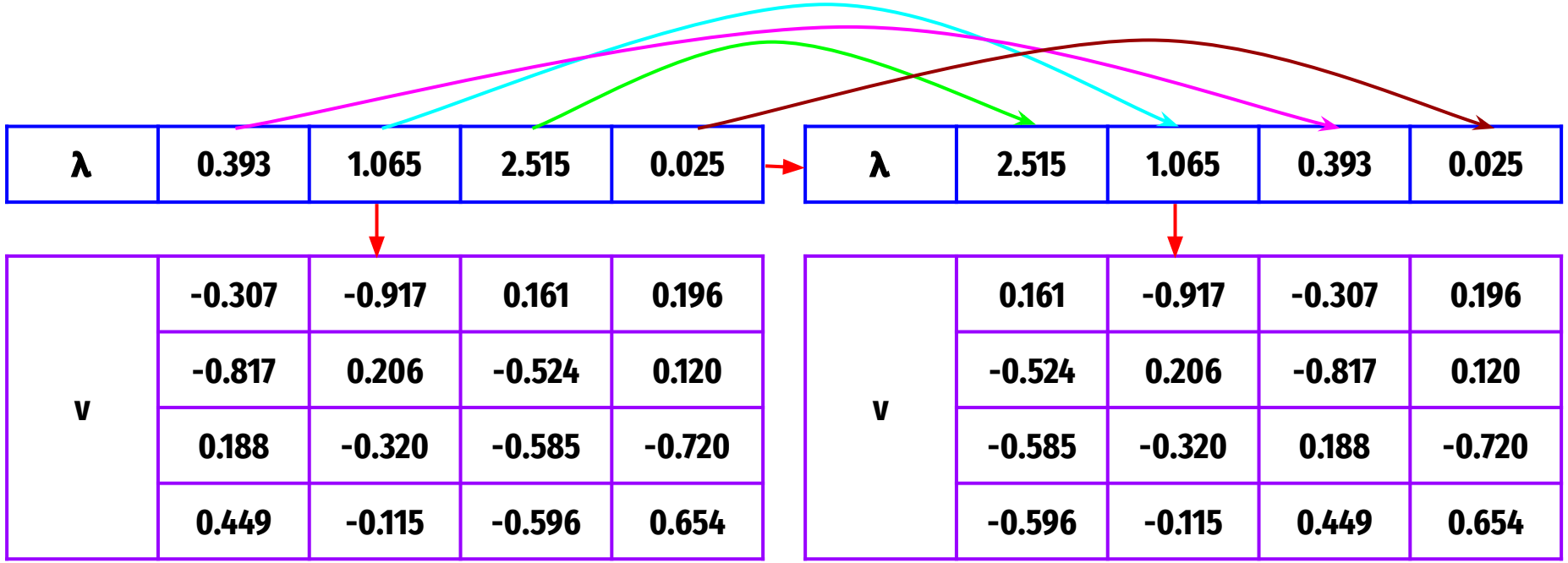
	f1	f2	f3	f4
f1	0.8	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8	0.51121	0.4945
f3	0.03849	0.51121	0.8	0.75236
f4	-0.14479	0.4945	0.75236	0.8

	f1	f2	f3	f4
f1	0.8 - λ	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8- λ	0.51121	0.4945
f3	0.03849	0.51121	0.8 - λ	0.75236
f4	-0.14479	0.4945	0.75236	0.8 - λ

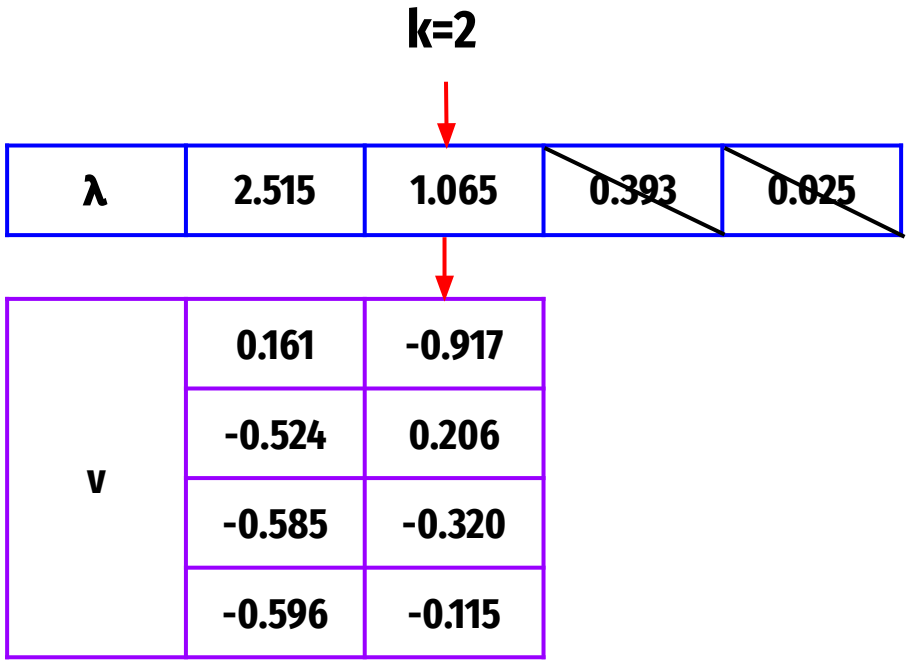
λ	**0.393**	**1.065**	**2.515**	**0.025**

v	**-0.307**	**-0.917**	**0.161**	**0.196**
	-0.817	**0.206**	**-0.524**	**0.120**
	0.188	**-0.320**	**-0.585**	**-0.720**
	0.449	**-0.115**	**-0.596**	**0.654**

PCA: step 4 - Sort eigenvalues & corresponding eigenvectors



PCA: step 5 - Pick k eigenvalues and form a matrix



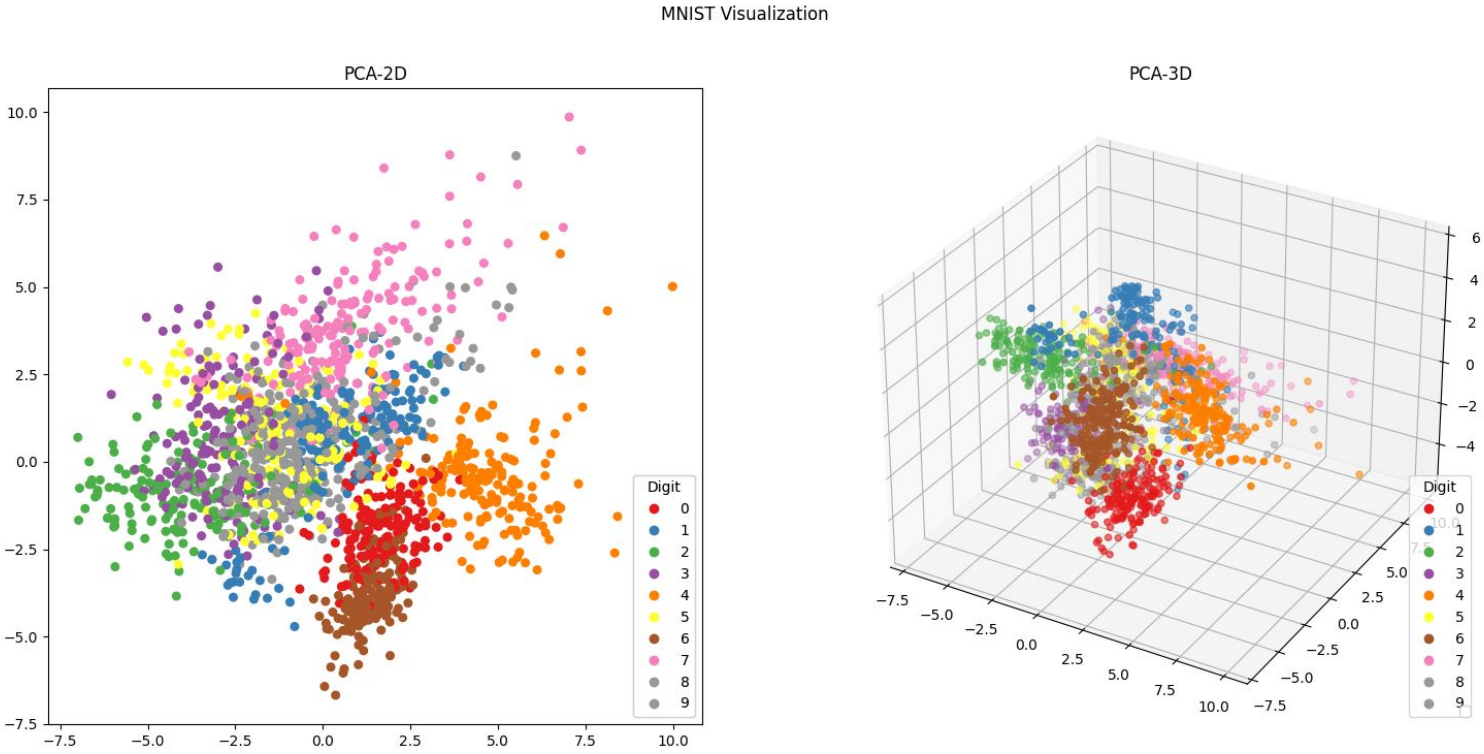
PCA: step 6 - Transform feature matrix

Nx4 matrix x

0.161	-0.917
-0.524	0.206
-0.585	-0.320
-0.596	-0.115

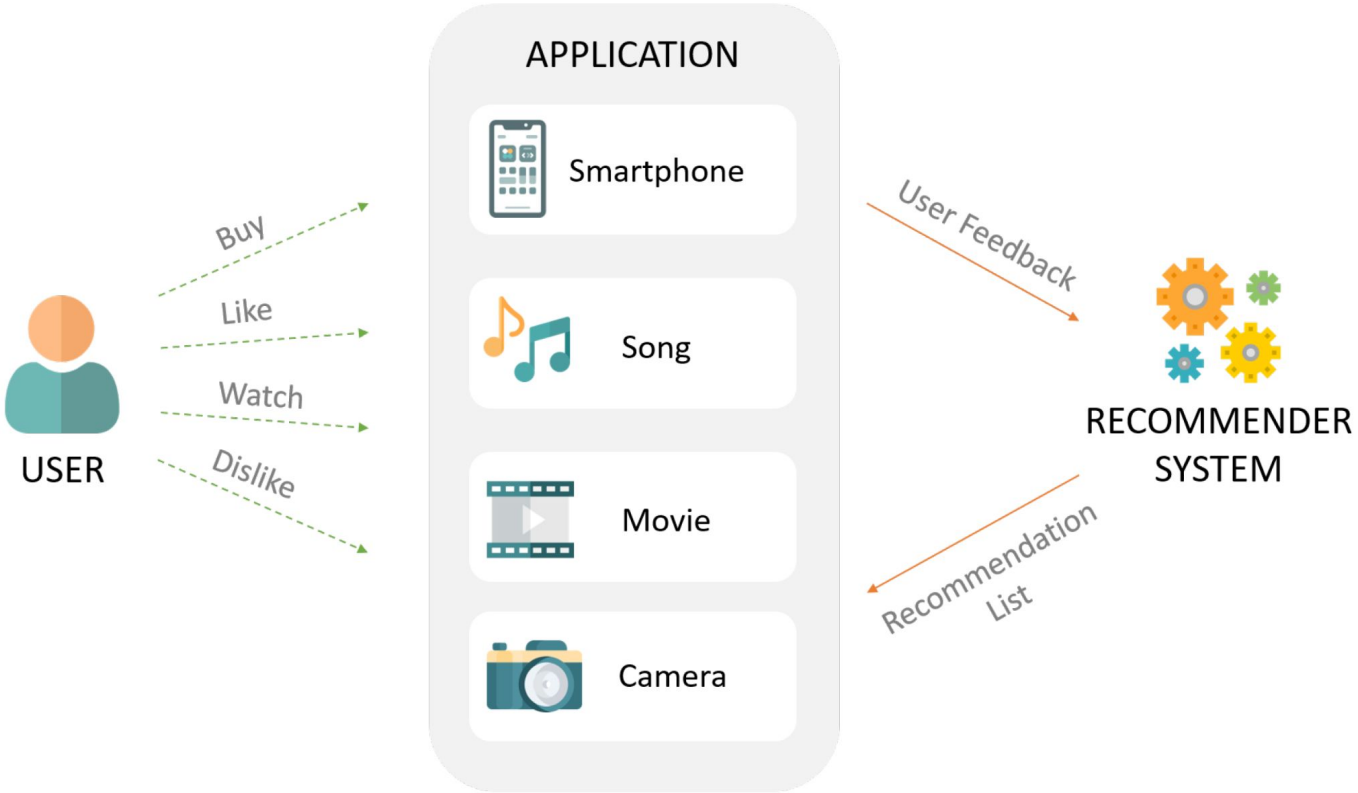
= Nx2 matrix

PCA visualization

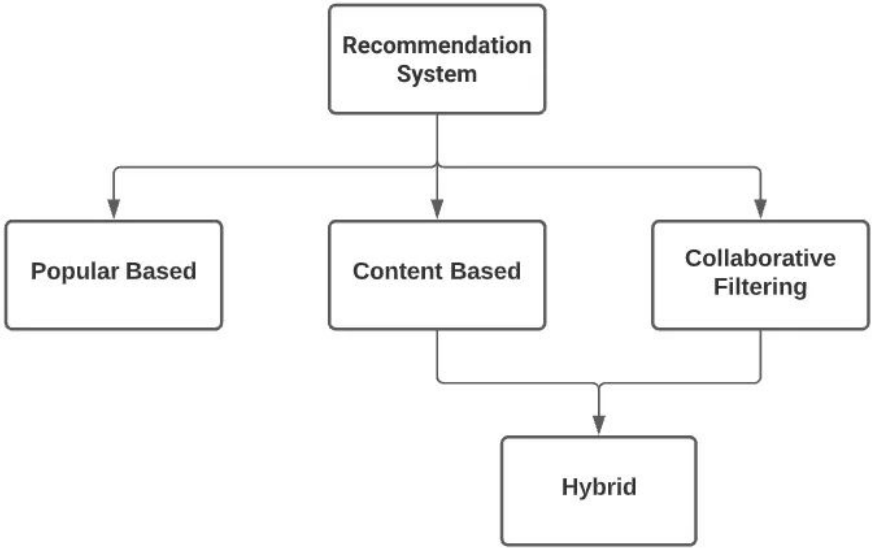


Recommendation systems

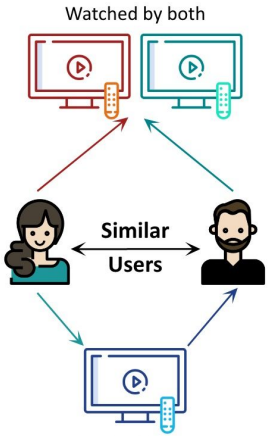
What is recommendation systems



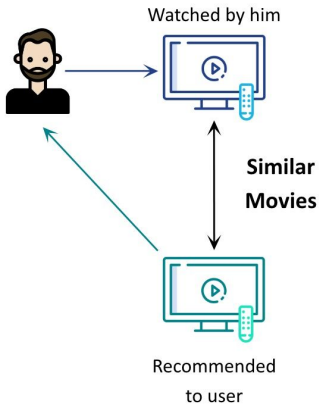
Types of recommendation systems



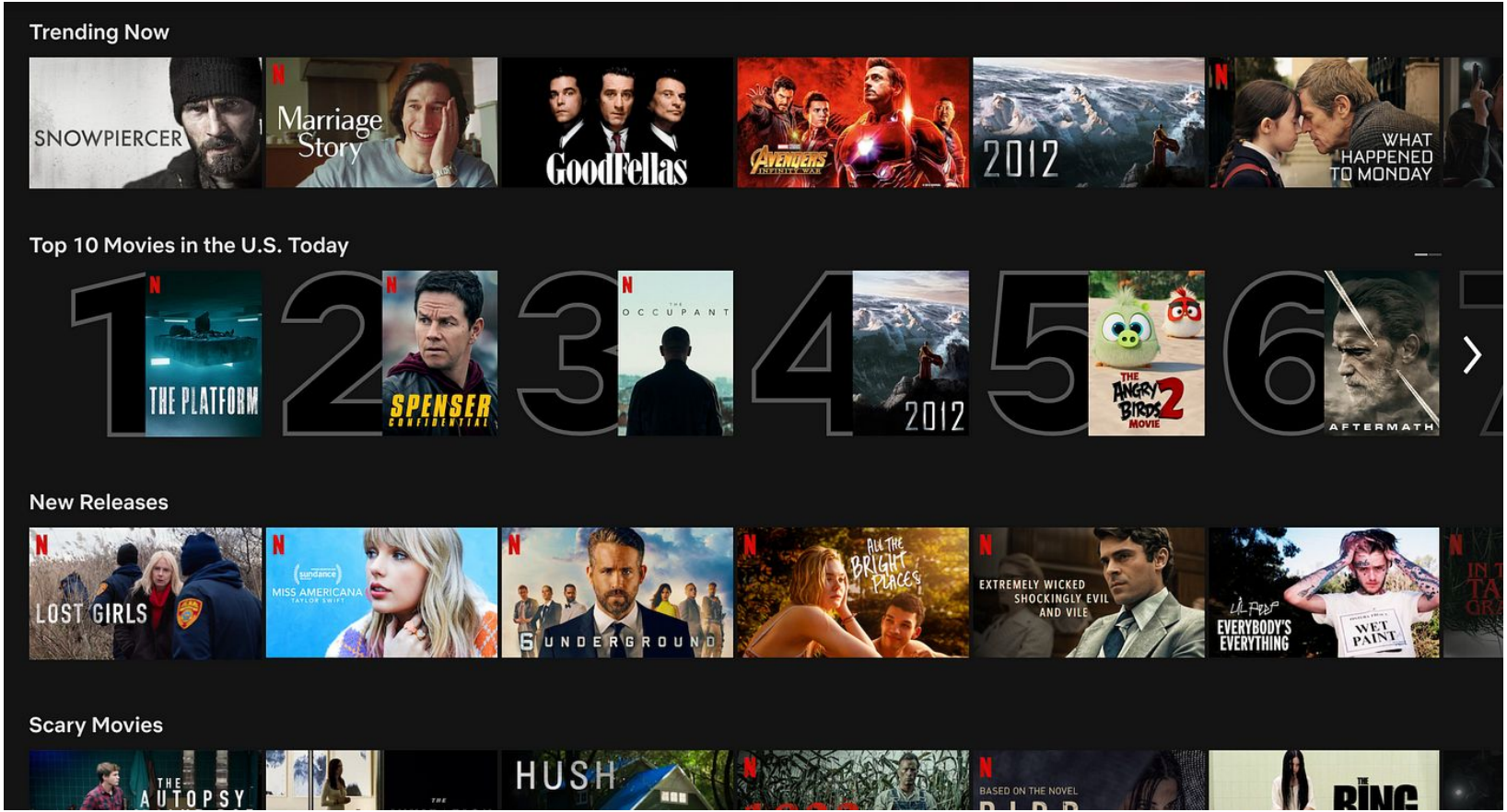
Collaborative Filtering



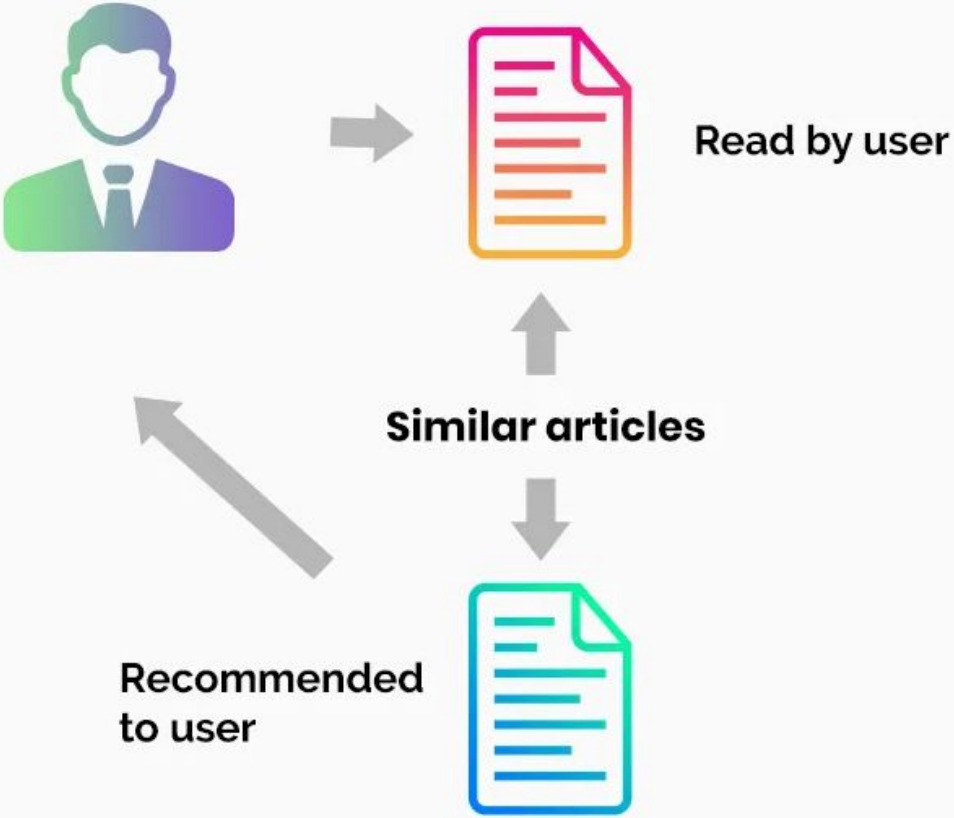
Content-Based Filtering



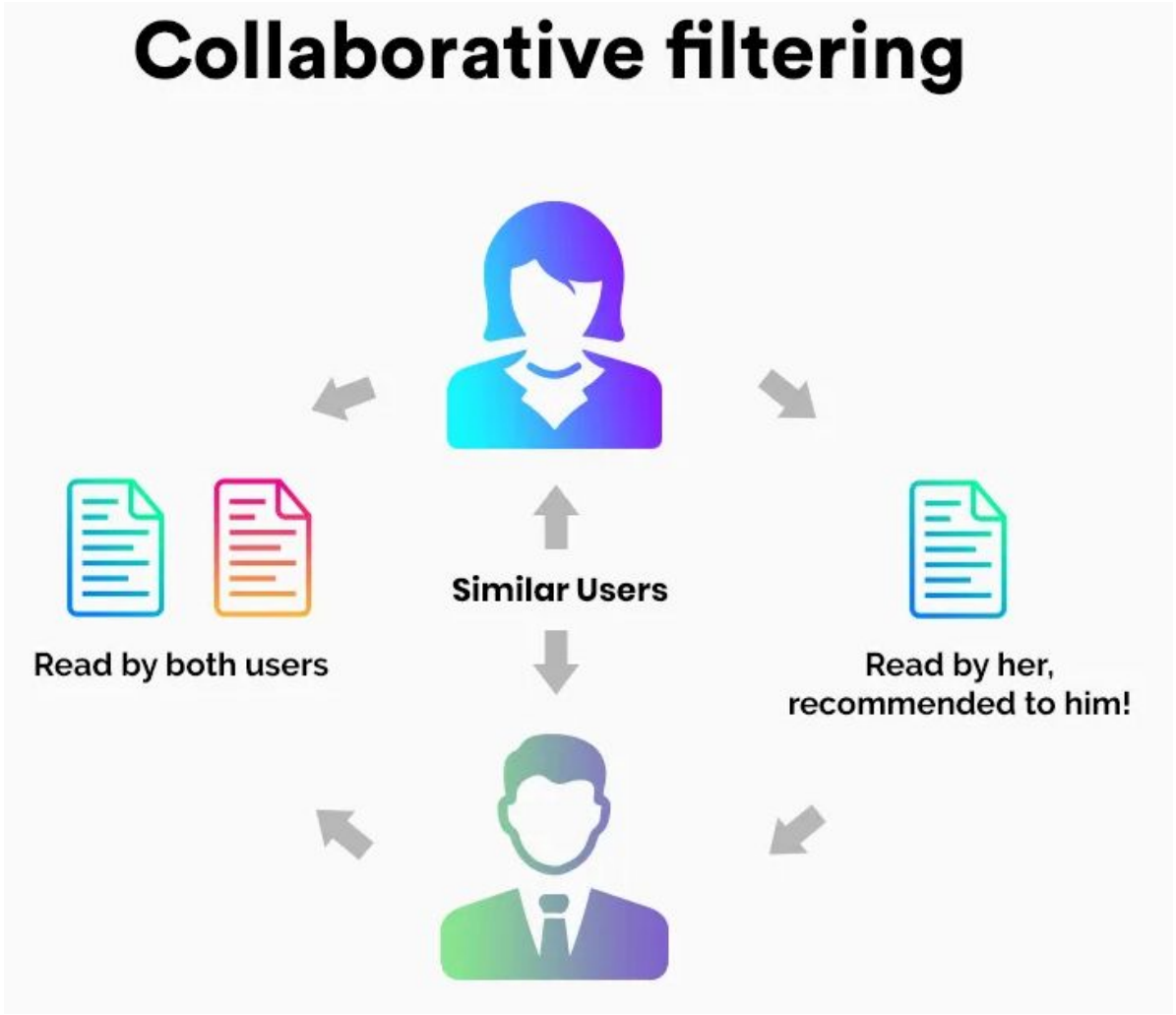
Popularity-based recommendation systems



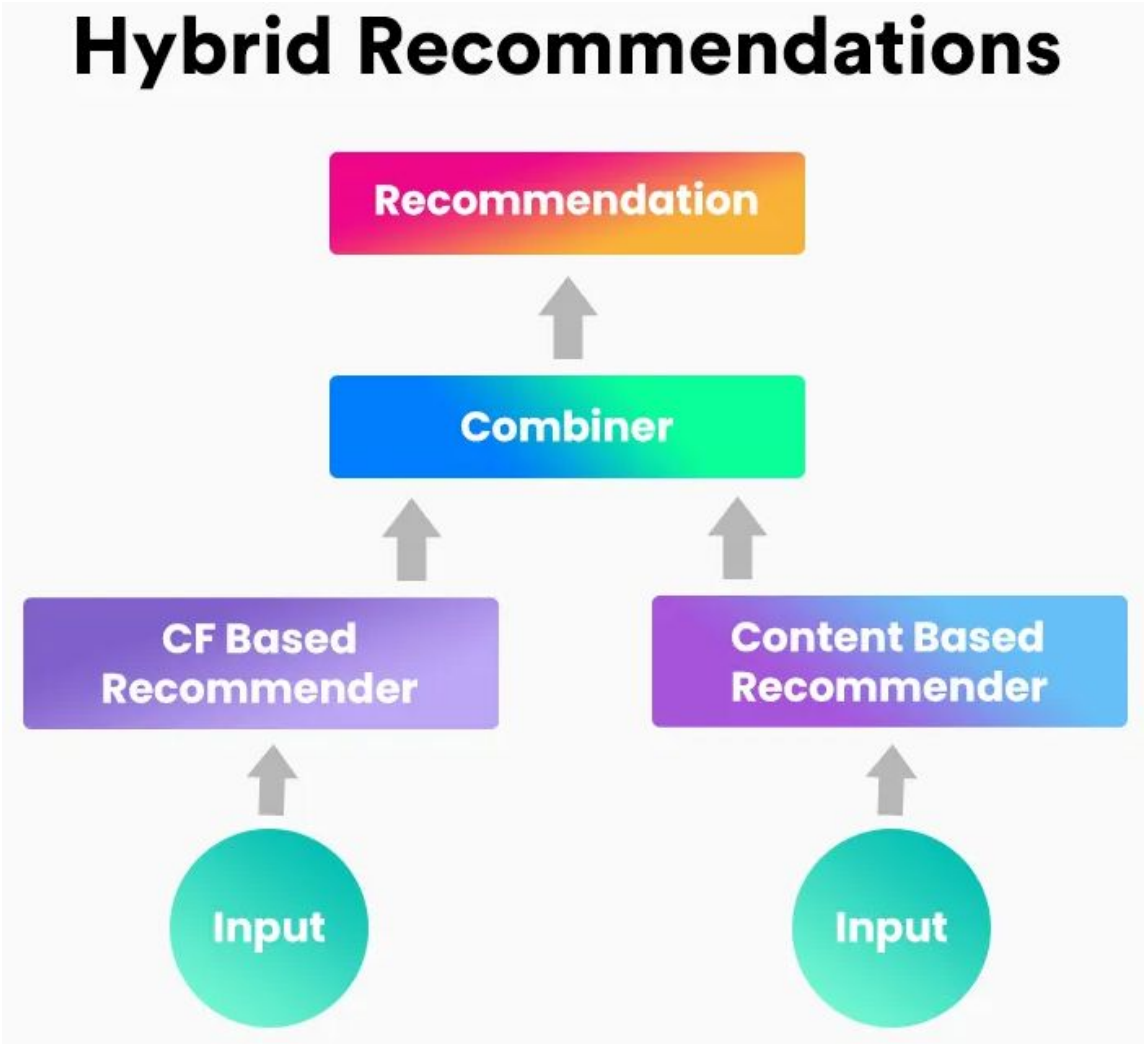
Content-based filtering



Collaborative filtering



Hybrid Recommendations



Utility matrix