Statistical Modelling I (Supervised Learning - Regression)

Hari, Tanggal

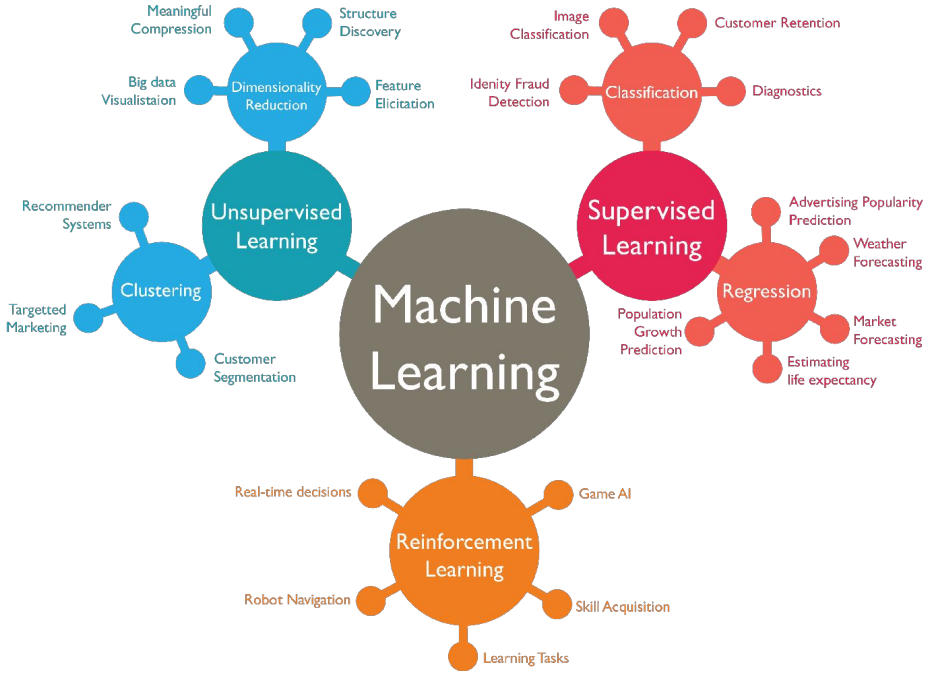
Data Analytics

Program Zenius Studi Independen Bersertifikat Bersama Kampus Merdeka

1. Intro to Machine Learning
2. Supervised Learning
3. Regression
4. Linear Regression
5. Regression : Model Evaluation
6. Important Concepts
7. Machine Learning Algorithms

Intro to Machine Learning

Intro to Machine Learning



Intro to Machine Learning

Supervised Learning :

Input data is called training data and has a known label or result (such as spam/not-spam or a stock price at a time).

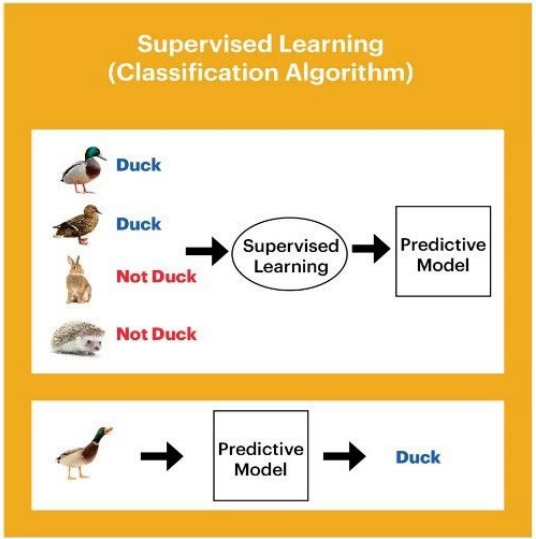
Unsupervised Learning :

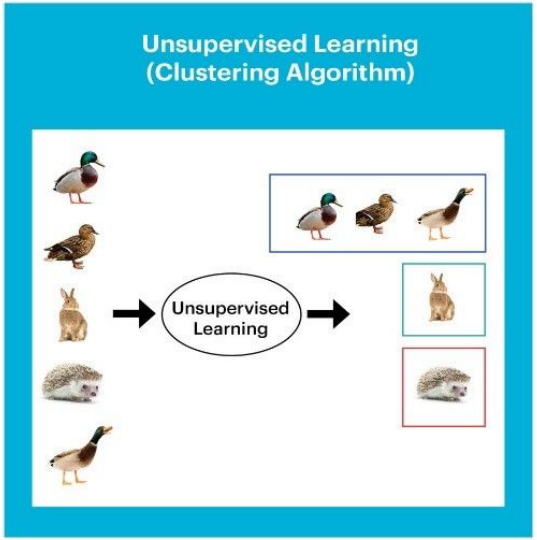
Input data is not labeled and does not have a known result

Reinforcement Learning :

A special type of Machine Learning where the model learns from each action taken. The model is rewarded for any correct decision made and penalized for any wrong decision

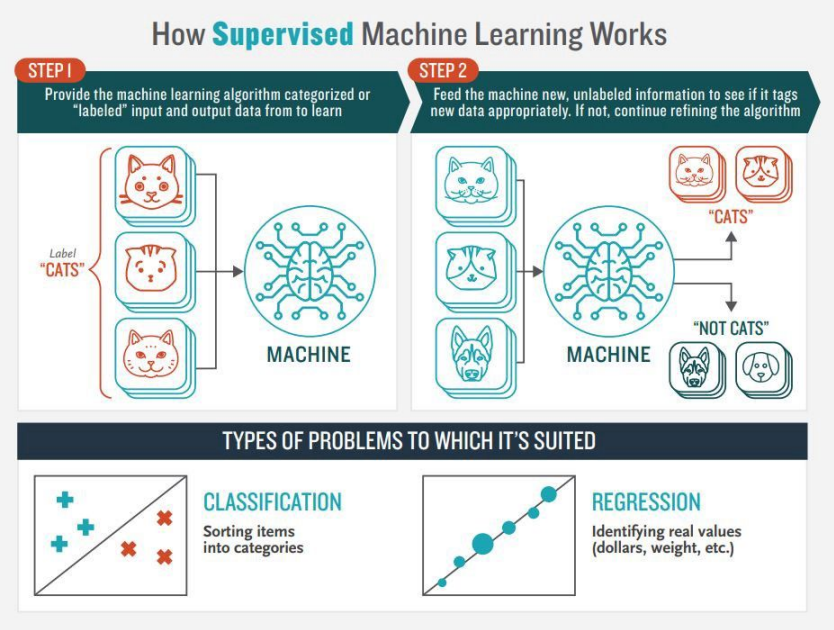
Intro to Machine Learning





Supervised Learning

Supervised Learning



Supervised Learning

price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	s
221900	3	1	1180	5650	1	0	0	3	7	1180	
538000	3	2.25	2570	7242	2	0	0	3	7	2170	
180000	2	1	770	10000	1	0	0	3	6	770	
604000	4	3	1960	5000	1	0	0	5	7	1050	
510000	3	2	1680	8080	1	0	0	3	8	1680	
1225000	4	4.5	5420	101930	1	0	0	3	11	3890	
257500	3	2.25	1715	6819	2	0	0	3	7	1715	
291850	3	1.5	1060	9711	1	0	0	3	7	1060	
229500	3	1	1780	7470	1	0	0	3	7	1050	
323000	3	2.5	1890	6560	2	0	0	3	7	1890	
662500	3	2.5	3560	9796	1	0	0	3	8	1860	
468000	2	1	1160	6000	1	0	0	4	7	860	

Label (Numerical)

Input Data

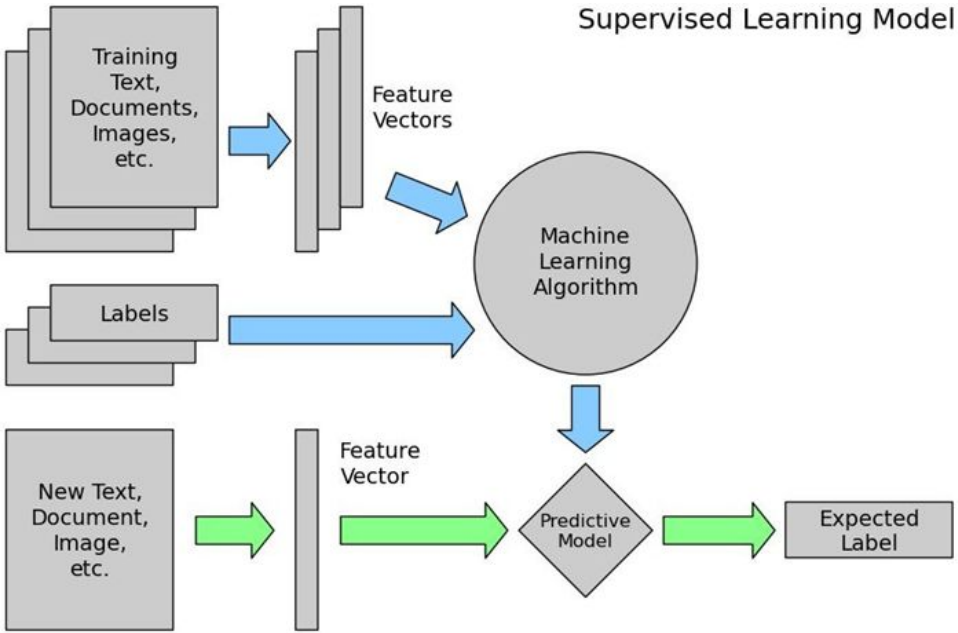
Supervised Learning

is_diabetes	um_pregnant	glucose_concentration	blood_pressure	triceps_thickness	two_hour_insulin	bmi	pedigree_function	age
1	6	148	72	35	0	33.6	0.627	50
0	1	85	66	29	0	26.6	0.351	31
1	8	183	64	0	0	23.3	0.672	32
0	1	89	66	23	94	28.1	0.167	21
1	0	137	40	35	168	43.1	2.288	33
0	5	116	74	0	0	25.6	0.201	30
1	3	78	50	32	88	31	0.248	26
0	10	115	0	0	0	35.3	0.134	29
1	2	197	70	45	543	30.5	0.158	53
1	8	125	96	0	0	0	0.232	54
0	4	110	92	0	0	37.6	0.191	30
1	10	168	74	0	0	38	0.537	34
0	10	139	80	0	0	27.1	1.441	57
1	1	189	60	23	846	30.1	0.398	59

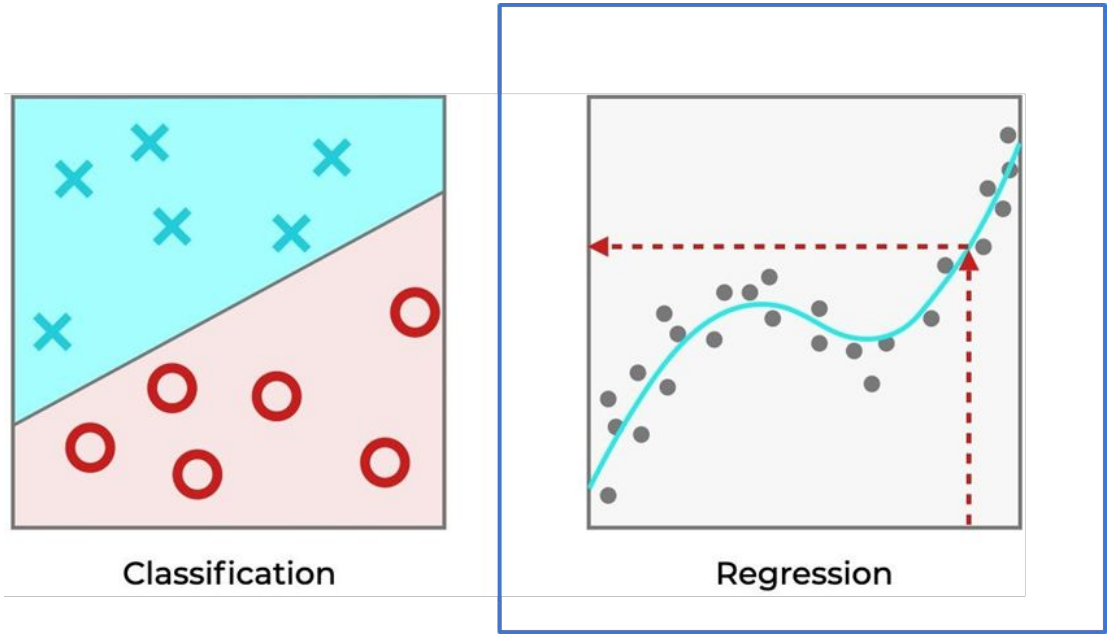
Label (Categorical)

Input Data

Supervised Learning



Supervised Learning Types



Regression

Regression

Regression Concept :

Regression takes a group of random variables, thought to be predicting Y, and tries to find a mathematical relationship between them. This relationship is typically in the form of a straight line (linear regression) that best approximates all the individual data points.

Example : (simple linear regression)

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Linear Regression

Using least-squares to create a linear line that fits the data (Only for numeric data types)

How?

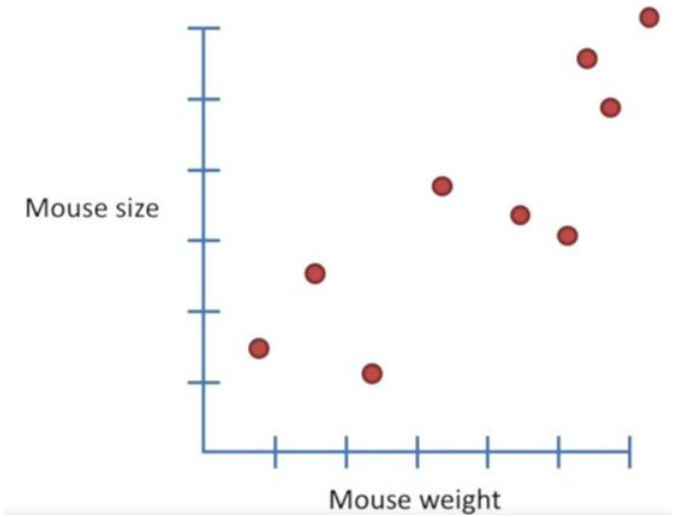
Find the minimum value of :

$$\sum_{i=1}^{M} (y_i - \hat{y}_i)^2$$

for any combination of linear lines

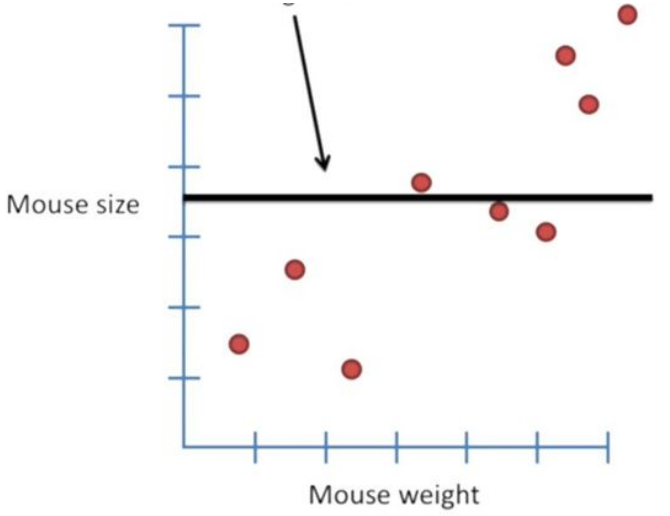
Linear Regression

Goal : Create an optimal linear line for all data points



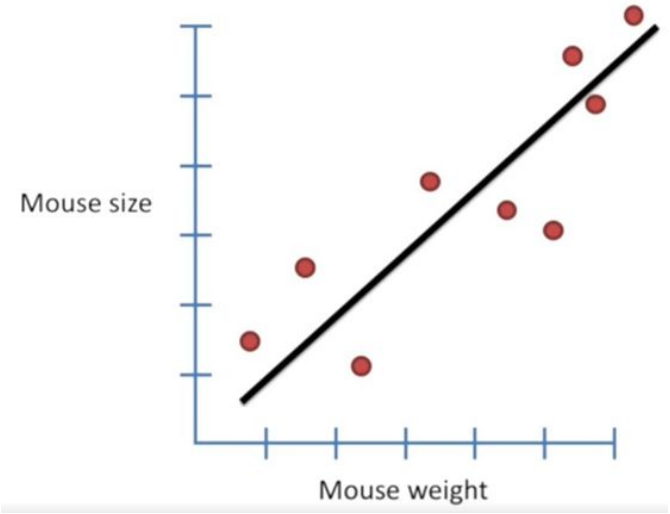
Linear Regression

Goal : Create an optimal linear line for all data points



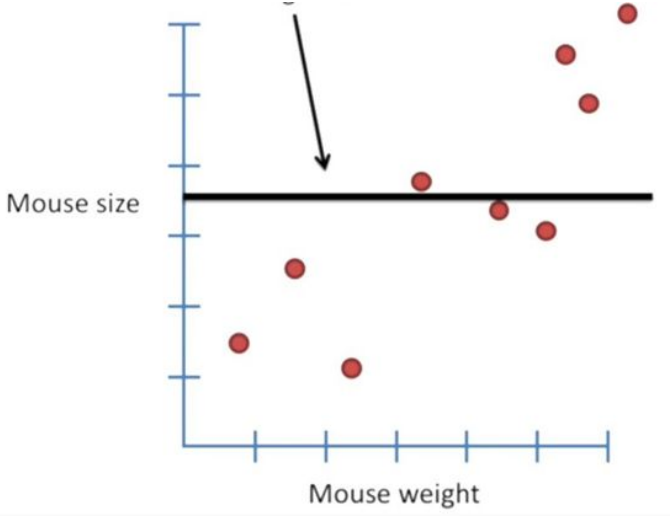
Linear Regression

Goal : Create an optimal linear line for all data points



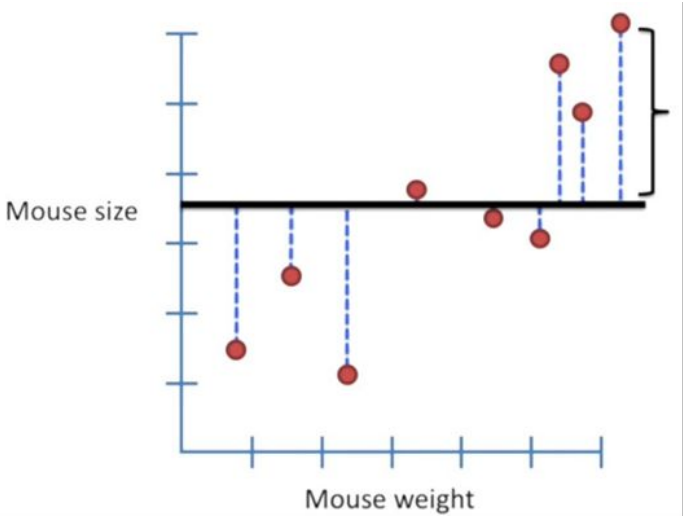
Linear Regression

Step 1 : draw a line (randomly) on the data



Linear Regression

Step 2: Calculate the distance between the line and the data points. Use the distance equation (below). Calculate the square and sum for all data points.

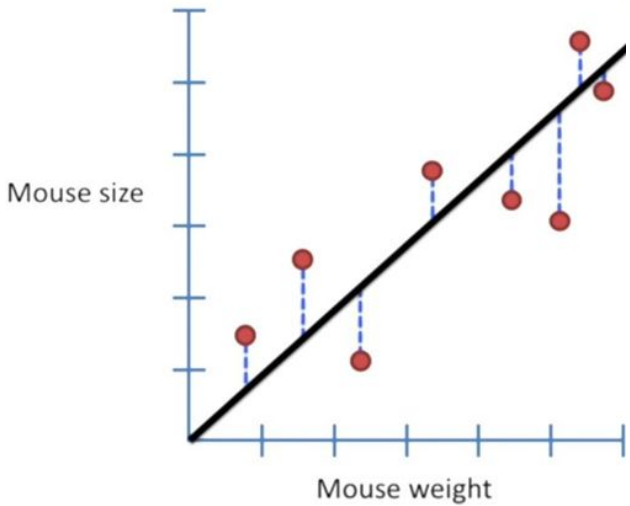


Sum of Squared Errors (Residue)

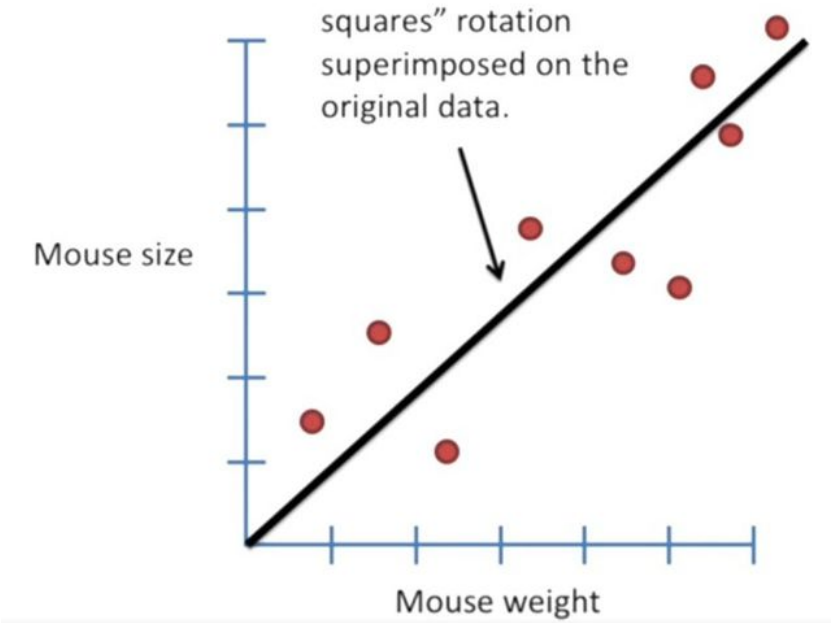
$$\sum_{i=1}^{M} (y_i - \hat{y}_i)^2$$

Linear Regression

Step 3: Rotate the existing line. Then, calculate the sum of squared errors (SSE) again. Repeat for all the possible linear lines.



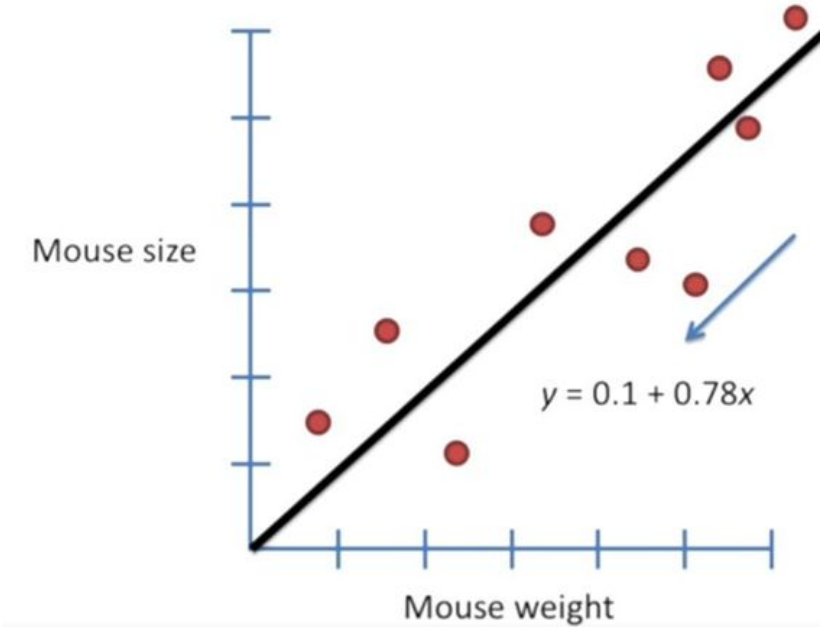
Least Square Method



Therefore, this algorithm is also known as :

Ordinary Least Squares (OLS).

Least Square Method



The equation of the linear line is formed as shown.

There are **2 parameters** that are calculated in the OLS algorithm.
0.1 and 0.78

Linear Regression

Simple Linear Regression

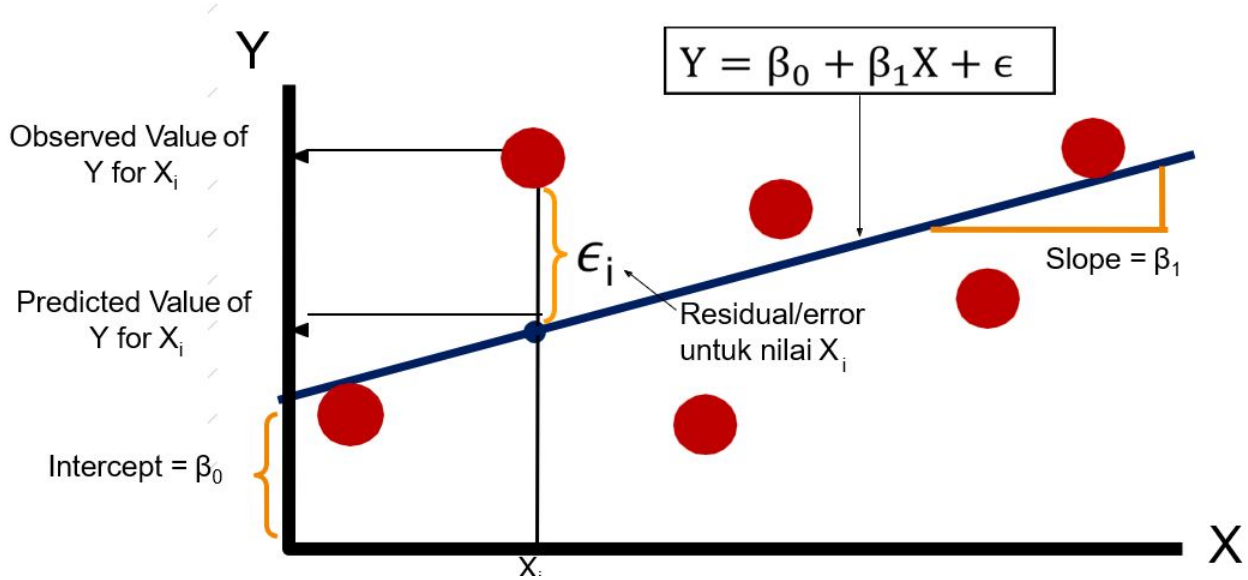
Dependent variable (Y) will change constantly when the value of the independent variable (X) increases or decreases.

Assuming that the relationship between the variables X and Y is linear.

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Y : Dependent variable
β_0 : Intercept
β_1 : Slope
X = Independent variable
ϵ : Random error (Noise)

Simple Linear Regression



Multivariate Linear Regression

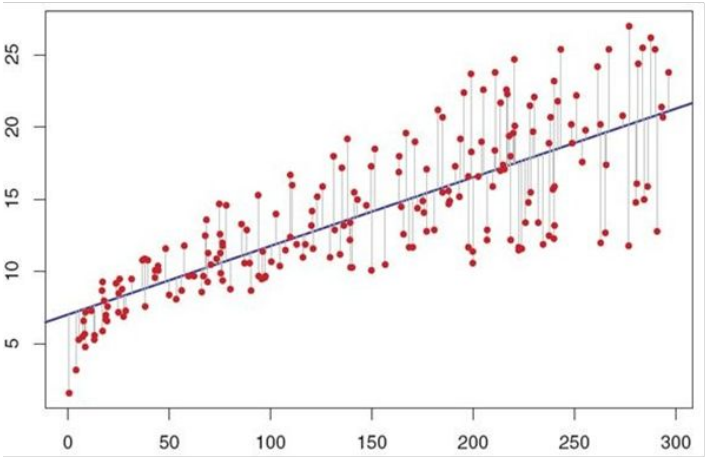
X variables (Independent variables) more than 1 variable.

Assuming that the relationship between the variables X and Y is linear.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ ... + \beta_i X_i$$

Y : Dependent variable
β_0 : Intercept
β_i : Slope for X_i
X = Independent variable

In Python



Fitting Model to Train Set

```
#Fitting simple linear regression to the Training Set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(xtrain, ytrain)
```

executed in 5ms, finished 09:38:27 2019-12-02

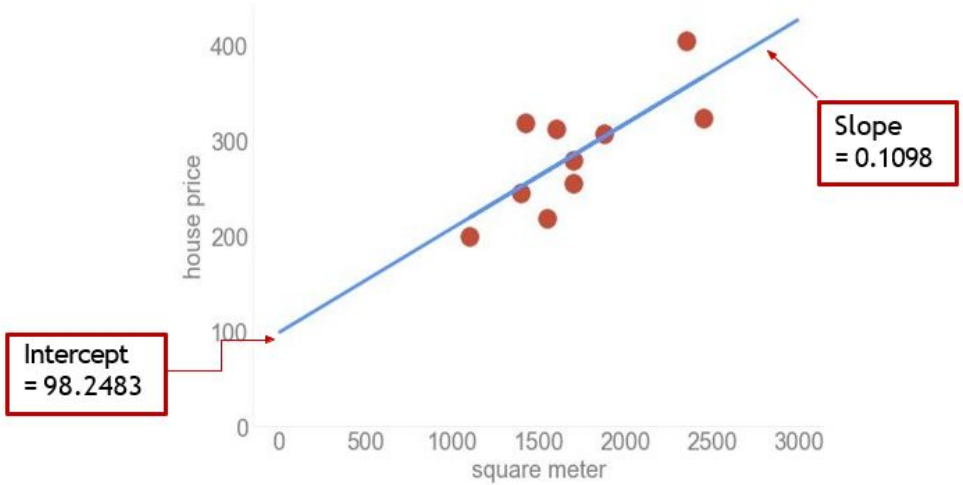
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

Predict the Test Set

```
#Predicting the prices
pred = regressor.predict(xtest)
```

executed in 10ms, finished 09:41:40 2019-12-02

In Python



The coefficients

```
# The coefficients
print('Coefficients: \n', regressor.coef_)
```

Intercept

```
# The Intercept
print('Intercept: \n', regressor.intercept_)
```

In Python

```
4  complex_model_1 = linear_model.LinearRegression()
5  complex_model_1.fit(train_data_dm[features],train_data_dm['price'])
6
7  print('Intercept: {}'.format(complex_model_1.intercept_))
8  print('Coefficients: {}'.format(complex_model_1.coef_))
```

Regression : Model Evaluation

Regression : Model Evaluation

Mean Absolute Error (MAE): measure the average error of the prediction results without taking into account the direction (the smaller the better).

$$\text{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

Root mean squared error (RMSE): is a quadratic scoring rule that also measures the average magnitude of the error.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Regression : Model Evaluation

R-squared: ranged from 0-1, indicating how much the independent variable affects the dependent variable. The closer the value to 1, the better the model.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2}$$

Where,

$\hat{y}$ – *predicted value of y*

$\bar{y}$ – *mean value of y*

Regression : Model Evaluation

CASE 1: Evenly distributed errors

ID	Error	|Error|	Error^2
1	2	2	4
2	2	2	4
3	2	2	4
4	2	2	4
5	2	2	4
6	2	2	4
7	2	2	4
8	2	2	4
9	2	2	4
10	2	2	4

MAE	RMSE
2.000	2.000

CASE 2: Small variance in errors

ID	Error	|Error|	Error^2
1	1	1	1
2	1	1	1
3	1	1	1
4	1	1	1
5	1	1	1
6	3	3	9
7	3	3	9
8	3	3	9
9	3	3	9
10	3	3	9

MAE	RMSE
2.000	2.236

CASE 3: Large error outlier

ID	Error	|Error|	Error^2
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	20	20	400

MAE	RMSE
2.000	6.325

RMSE has the advantage of providing a large error penalty, resulting in precise measurements for some of the more sensitive cases.

Regression : Model Evaluation

```
# MAE
from sklearn.metrics import mean_absolute_error
mean_absolute_error(y_true, y_pred)
```

```
# MSE
from sklearn.metrics import mean_squared_error
mean_squared_error(y_true, y_pred)
```

```
# RMSE
from sklearn.metrics import mean_squared_error
mean_squared_error(y_true, y_pred, squared=False)
```

Important Concepts

One Hot Encoding

Since machine mostly **unable to process categorical data** as these categories have no meaning for them, this information has to be prepared if we want a computer to be able to process it.

This action is called **preprocessing**. A big part of preprocessing is encoding - representing every single piece of data in a way that a computer can understand (the name literally means "**convert to computer code**").

In many branches of computer science, especially machine learning and digital circuit design, **One-Hot Encoding is widely used**.

One Hot Encoding

One-hot Encoding is a type of vector representation in which **all of the elements in a vector are 0, except for one**, **which has 1 as its value**, where **1 represents a boolean specifying a category of the element**.

id	color
1	red
2	blue
3	green
4	blue

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

One Hot Encoding

```
#One Hot Encoding
ids = [11, 22, 33, 44, 55, 66, 77]
countries = ['Spain', 'France', 'Spain', 'Germany', 'France']

df = pd.DataFrame(list(zip(ids, countries)),
                  columns=['Ids', 'Countries'])
df
```

	Ids	Countries
0	11	Spain
1	22	France
2	33	Spain
3	44	Germany
4	55	France

```
y = pd.get_dummies(df["Countries"], prefix='Country')
y.head()
```

	Country_France	Country_Germany	Country_Spain
0	0	0	1
1	1	0	0
2	0	0	1
3	0	1	0
4	1	0	0

Train Test Split

The train-test split is a data preprocessing technique for evaluating the performance **of a machine learning algorithm**.

It can be used for **classification or regression** problems and can be used for any supervised learning algorithm.

The procedure involves taking a dataset and **dividing it into two subsets**.

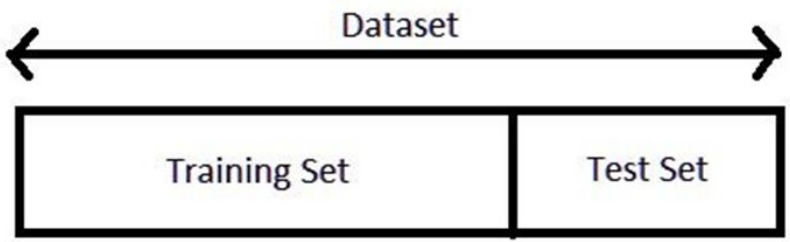
The first subset is used to fit the model and is referred to as the **training dataset**.

The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the **test dataset.**

Train Test Split

Train Dataset: Used to fit the machine learning model.

Test Dataset: Used to evaluate the fit machine learning model.



Train Test Split

```
## Linear Regression
```

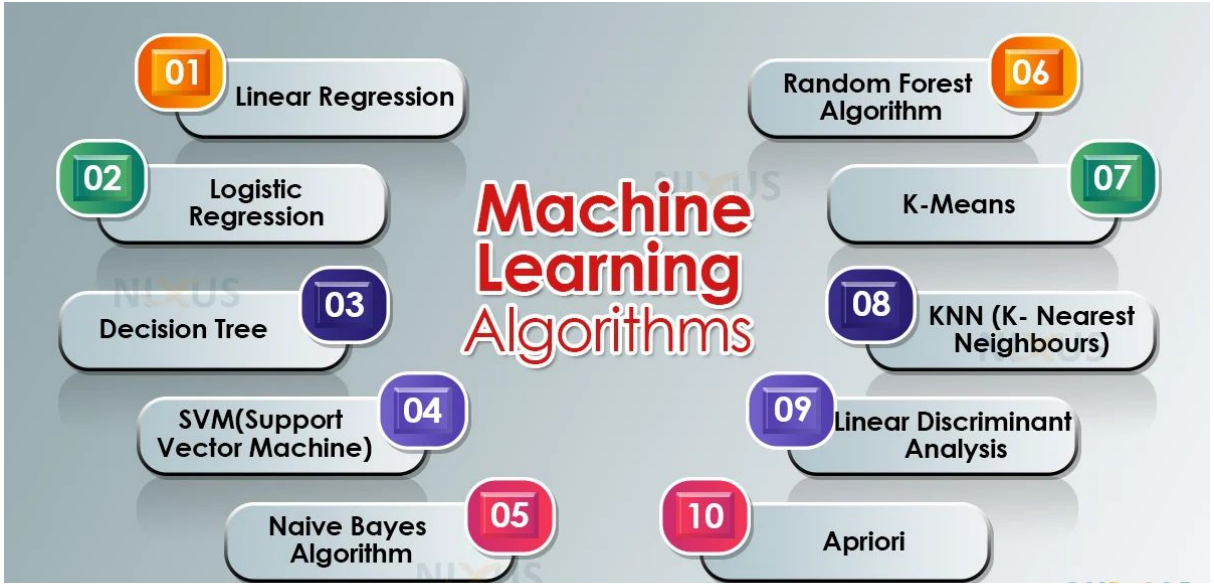
```
# load the boston dataset
boston = datasets.load_boston(return_X_y=False)

# defining feature matrix(X) and response vector(y)
X = boston.data
y = boston.target
```

```
# splitting X and y into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,
                                                    random_state=1)
```

Machine Learning Algorithms

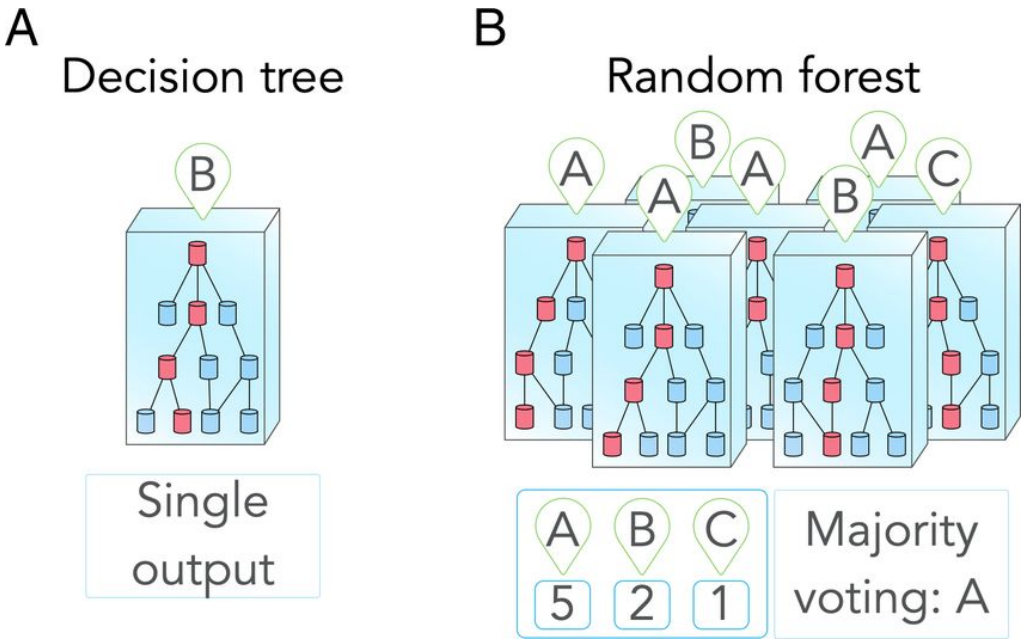
Machine Learning Algorithms



Machine Learning Algorithms

	Regression	Classification
Regression	sklearn.linear_model.LinearRegression	sklearn.linear_model.LogisticRegression
Decision Tree	sklearn.tree.DecisionTreeRegressor	sklearn.tree.DecisionTreeClassifier
Random Forest	sklearn.ensemble.RandomForestRegressor	sklearn.ensemble.RandomForestClassifier

Decision Tree VS Random Forest



Thank you!

Any Questions?