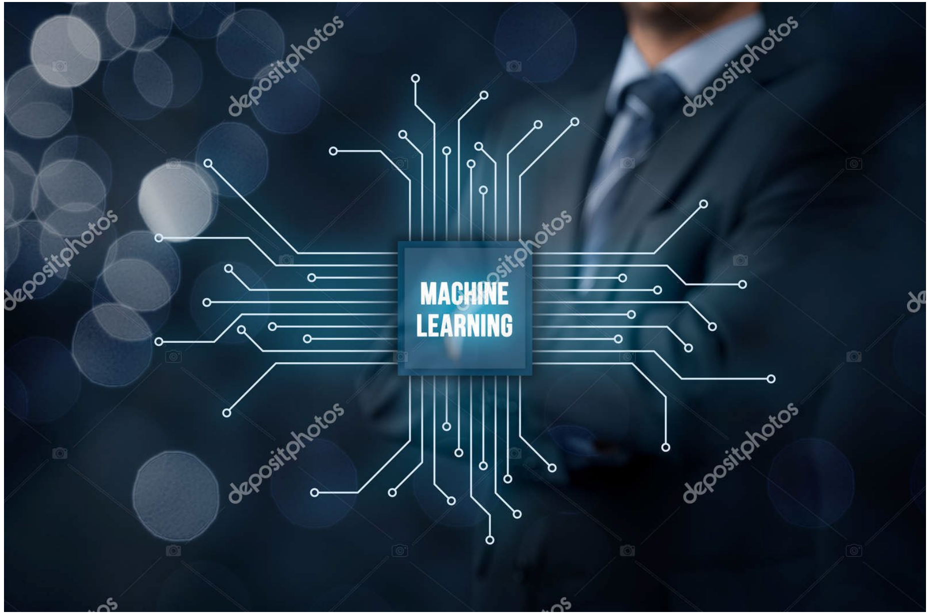
Machine Learning

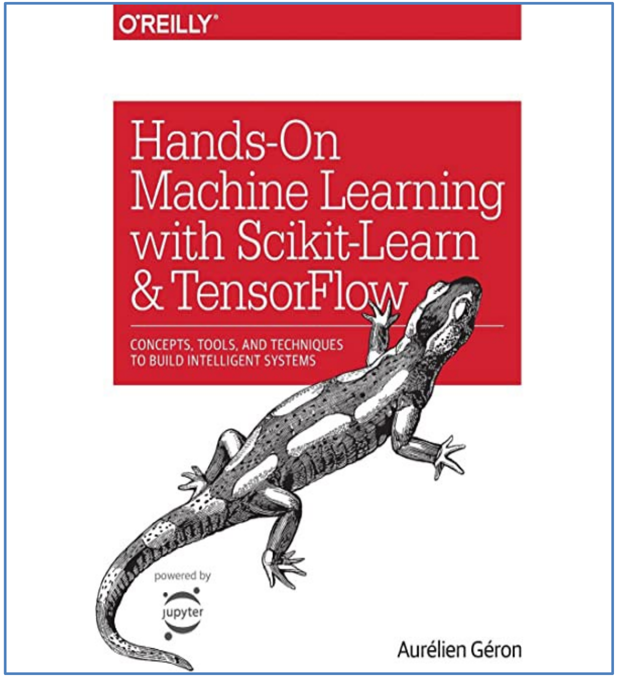


Introduced by

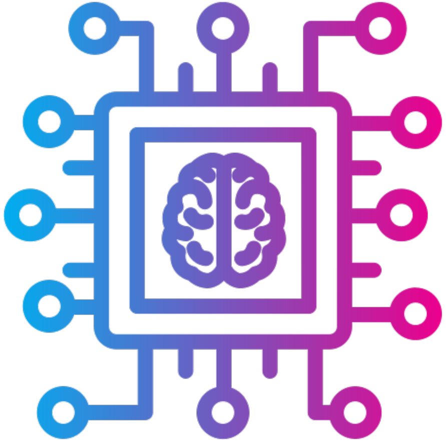
Dr. Ebtsam Adel

**Lecturer of Information Systems,
Information Systems department,
Faculty of computers and information,
Damanhour university**

Materials



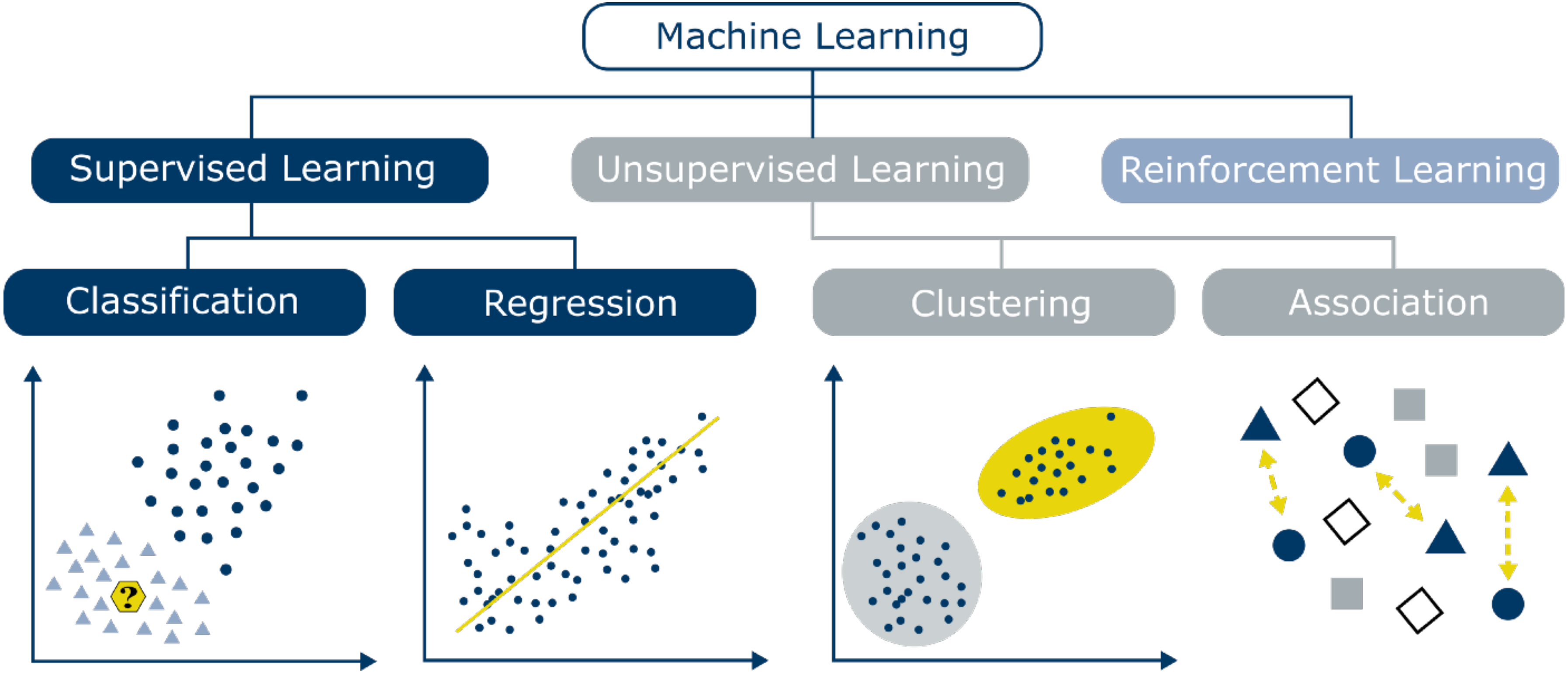
Aurélien Géron



Introduced by

Dr. Ebtsam Adel

Types of Machine Learning



Advantages of Machine Learning

- ✓ **Automation** :repetitive
- ✓ **Enhancing user experience and decision making**: an analyze and gain insights from large datasets for decision making.
- ✓ **Wide Applicability**: wide range of applications.
- ✓ **Continuous Improvement**: improve accuracy and efficiency

Disadvantages of Machine Learning

- **Data acquisition:** the most difficult task in machine learning is collecting data.
- **Inaccurate Results:** credibility of the interpreted result generated by the algorithm.
- **Maintenance:** Machine learning models have to continuously be maintained and monitored to ensure that they remain effective and accurate over time.

Challenges of Machine Learning

Main Challenges of Machine Learning:

- Data Privacy
- Insufficient Quantity of Training Data
- Non_representative Training Data
- Poor-Quality Data
- Irrelevant Features
- Overfitting the Training Data

Machine Learning

Chapter 4

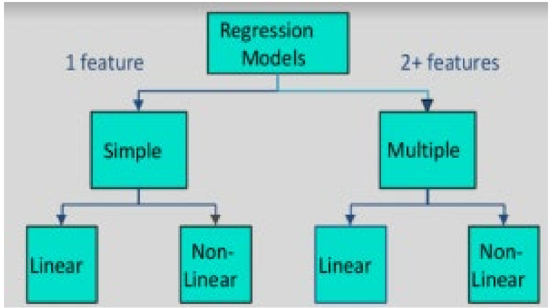
Training Models

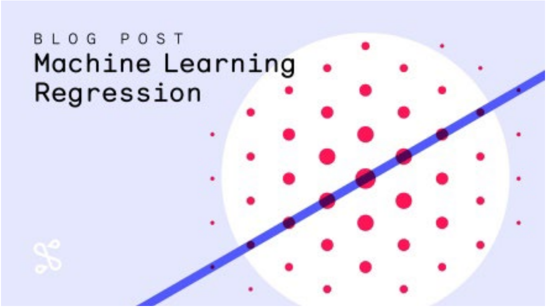
By

Dr. Ebtsam Adel

Outlines

- ➢ **What is Linear Regression?**
- ➢ **Simple Linear Regression**
- ➢ **Gradient Descent**
- ➢ **Multivariable Linear Regression**





What is Regression?

What is Regression

- **Linear regression** can be viewed as a type of supervised ML.

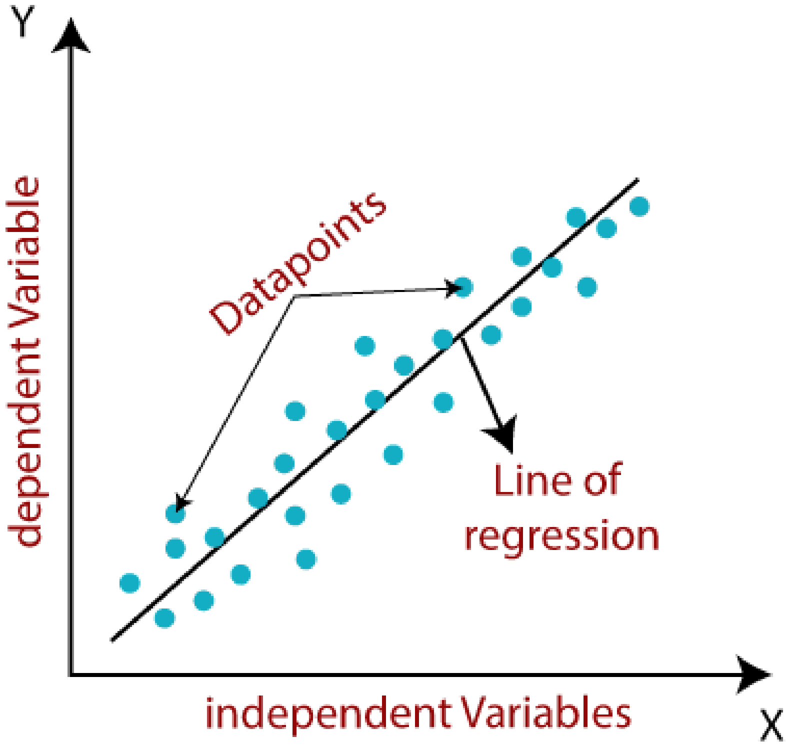
- **Regression analysis :** is a form of predictive modeling technique which investigates the relationship between a dependent (target) and independent variable (s) (predictor). This technique is used for forecasting, time series modeling.

- **Regression** : predict range of continuous values.

Statistical model

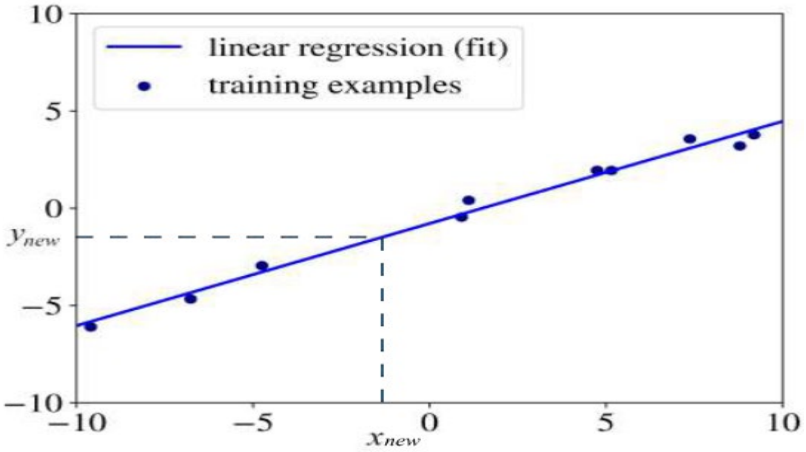
What is Regression

- The **linear regression** model provides a sloped straight line representing the relationship between the variables.



What is Regression

- **Regression**: We have a set of correlated data, and we want to predict new added value.



What is Regression

- **Regression** is classified into two main categories:

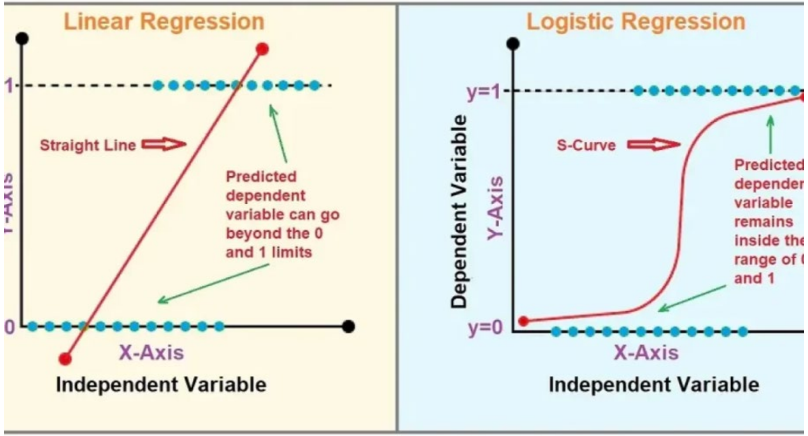
1. **Simple linear regression**: The simple linear regression model handled only a single descriptive feature. It involves **two only variables**, one is independent and the other is dependent. **For example,** house price based on its square feet.

2. **Multivariable linear regression**. Has **more than one** independent variables. **For example,** house prices based on its square feet and location.

What is Regression

What is Regression Analysis?

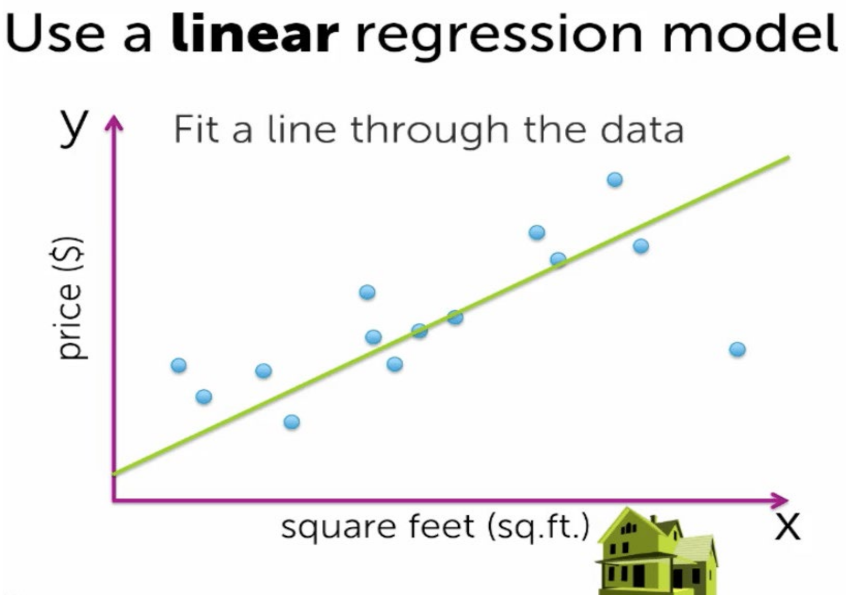
- In machine learning, regression analysis is a statistical technique that predicts continuous numeric values based on the relationship between independent and dependent variables.
- The main **goal** of regression analysis is to plot a line or curve that best fit the data and to estimate how one variable affects another.



Linear Regression – houses prices Example

Square (feet)	House price ($k)
1400	245
1600	312
1700	297
1875	308
1100	199
1550	405
2350	324

Linear regression example data



Terminologies Used In Regression

Terminologies Used In Regression Analysis

- **Independent Variables** – These variables are used to predict the value of the dependent variable, Predictors, features.

- **Dependent Variables** – These are the variables whose values we want to predict, are represented as target variables.

- **Regression line** – It is a straight line or curve that a regressor plots to fit the data points best.

Terminologies Used In Regression Analysis

- **Overfitting and Underfitting**

Overfitting is when the regression model works well with the training dataset but not with the testing dataset. It's also referred to as the problem of high variance.
Underfitting is when the model doesn't work well with training datasets. It's also referred to as the problem of high bias.

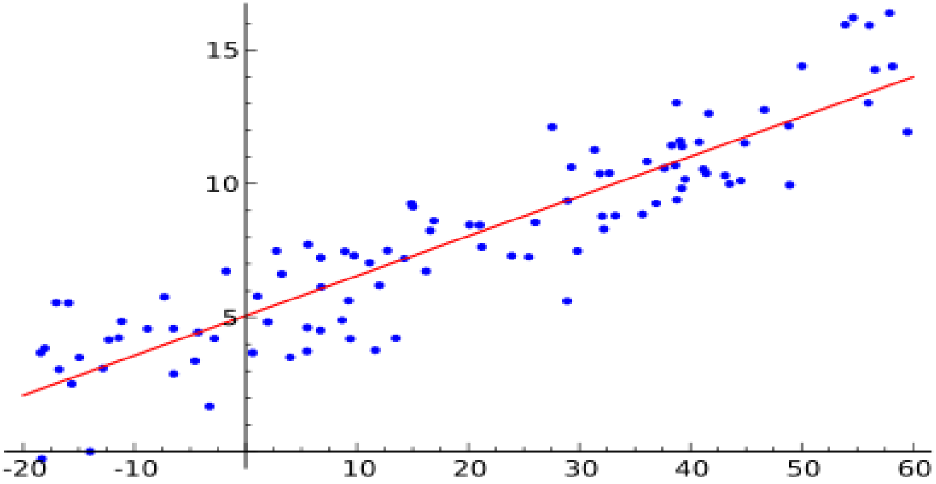
- **Outliers**

These are data points that don't fit the pattern of the rest of the data.

How Does Regression Work?

How Does Regression Work?

- Determine the **straight line (best fit line)** for which the differences between the actual values (Y) and the values that would be predicted from the fitted line of regression $\hat{y}$ (Y-hat) are as small as possible.

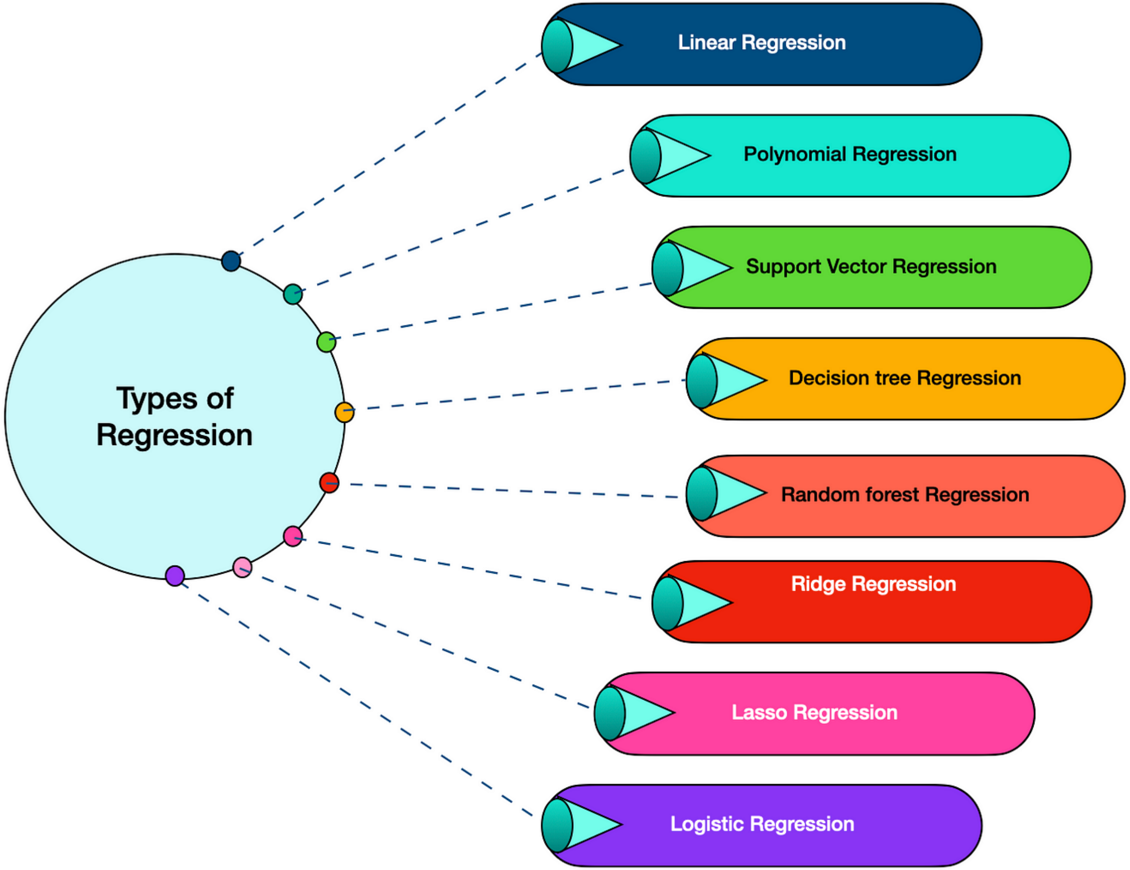


Types of Regression in Machine Learning

- The classification of regression methods is done based on the three metrics: The number of independent variables, type of dependent variables, and shape of the regression line.

- ✓ Linear Regression
- ✓ Logistic Regression
- ✓ Polynomial Regression
- ✓ Lasso Regression
- ✓ Ridge Regression
- ✓ Decision Tree Regression
- ✓ Random Forest Regression
- ✓ Support Vector Regression

Types of Machine Learning - Regression

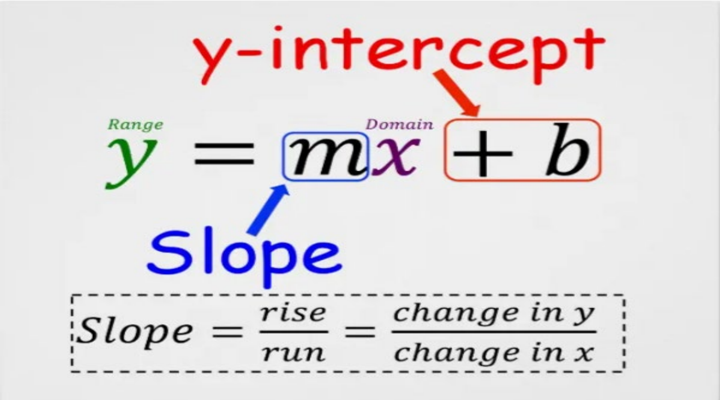


Linear Regression

A line is uniquely determined by two points and we write its equation as

$$y(x) = mx + b$$

where m is the slope of the line and b is the $y-$intercept, i.e., where the line crosses the y-axis. We say that y depends linearly on x. Note that the power of x is 1.



Linear Regression

Equation 4-1. Linear Regression model prediction

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

- $\hat{y}$ is the predicted value.

- n is the number of features.
- x_i is the i^{th} feature value.
- θ_j is the j^{th} model parameter (including the bias term θ_0 and the feature weights $\theta_1, \theta_2, \cdots, \theta_n$).

Linear Regression

Equation 4-2. Linear Regression model prediction (vectorized form)

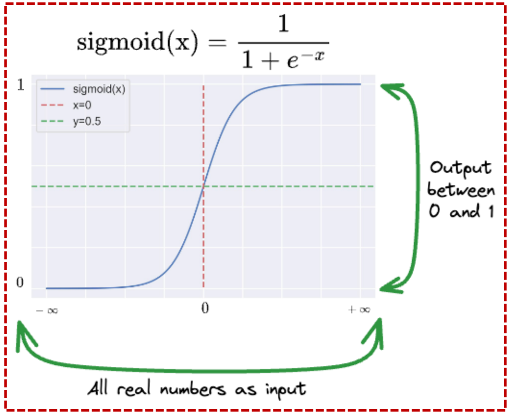
$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta} \cdot \mathbf{x}$$

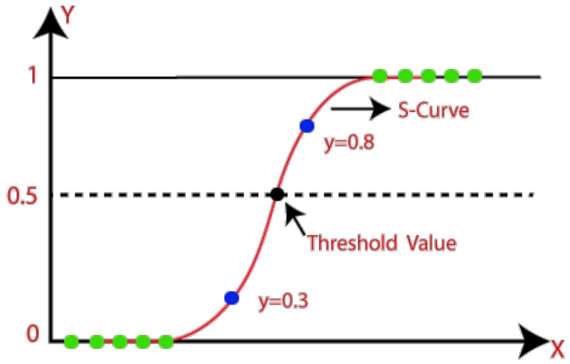
- $\boldsymbol{\theta}$ is the model's *parameter vector*, containing the bias term θ_0 and the feature weights θ_1 to θ_n.
- $\mathbf{x}$ is the instance's *feature vector*, containing x_0 to x_n, with x_0 always equal to 1.
- $\boldsymbol{\theta} \cdot \mathbf{x}$ is the dot product of the vectors $\boldsymbol{\theta}$ and $\mathbf{x}$, which is of course equal to $\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$.
- $h_{\boldsymbol{\theta}}$ is the hypothesis function, using the model parameters $\boldsymbol{\theta}$.

Logistic Regression

Logistic Regression

- **Logistic regression** is a popular machine learning algorithm used for predicting the probability of an event occurring.
- **Logistic regression** maps the dependent variable as a **sigmoid** function of independent variables.
- The sigmoid function produces a probability between 0 and 1. The probability value is used to estimate the dependent variable's value.





Logistic Regression

- ✓ It is mostly used in binary **classification** problems, where the target variable is categorical with two classes.
- ✓ It models the probability of the target variable given the input features and predicts the class with the highest probability.

Polynomial Regression

Polynomial Regression

- What if your data is actually **more complex** than a simple straight line? Surprisingly, you can actually use a linear model to fit nonlinear data.

- A simple way to do this is to **add powers of each feature as new features**, then train a linear model on this extended set of features. This technique is called *Polynomial Regression*

- generate some nonlinear data, based on a simple **quadratic** equation.

Polynomial Regression

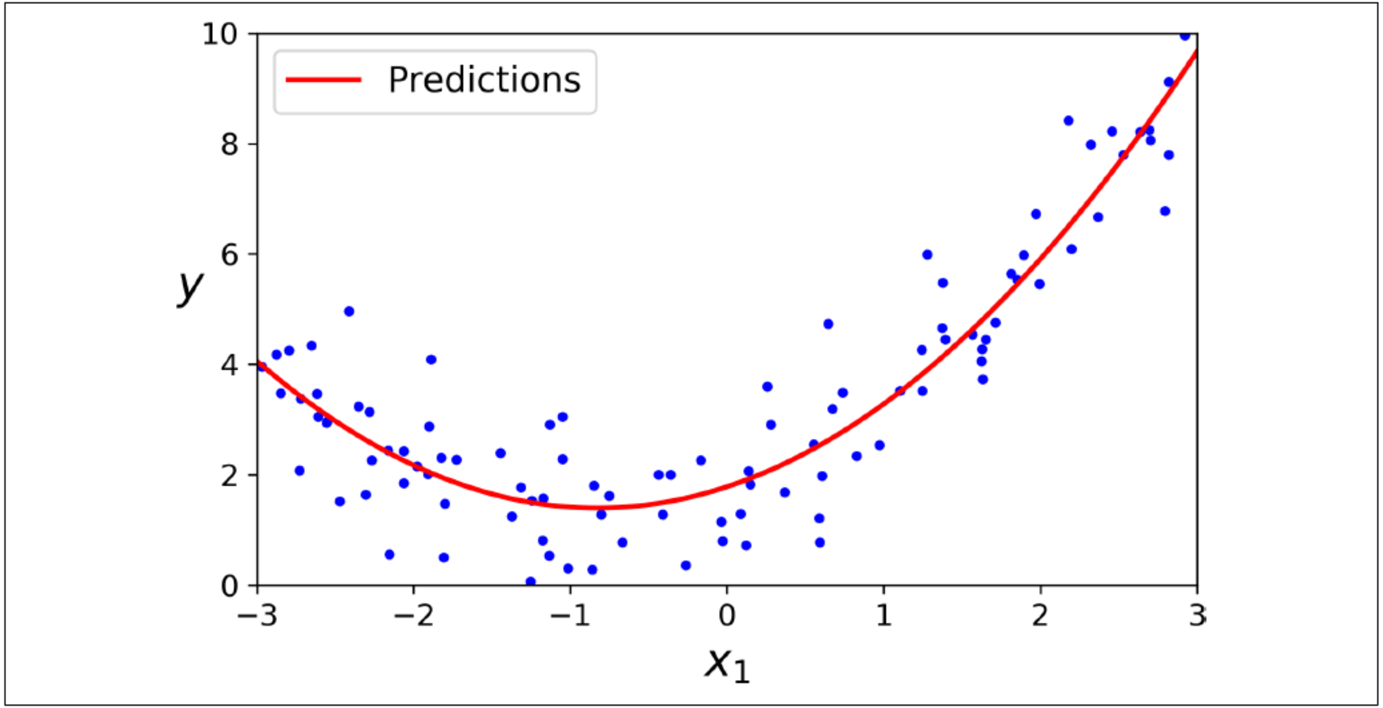


Figure 4-13. Polynomial Regression model predictions

Polynomial Regression

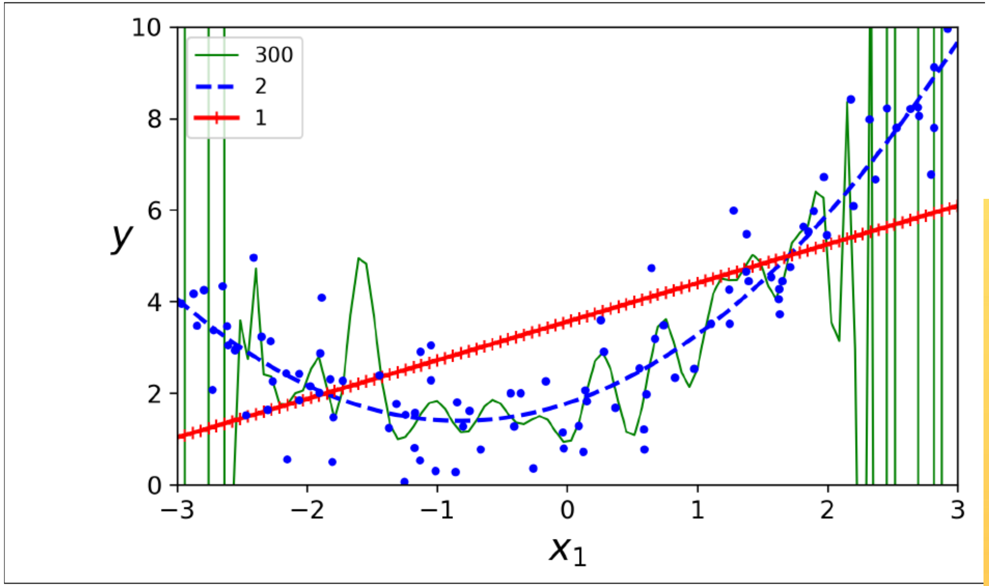
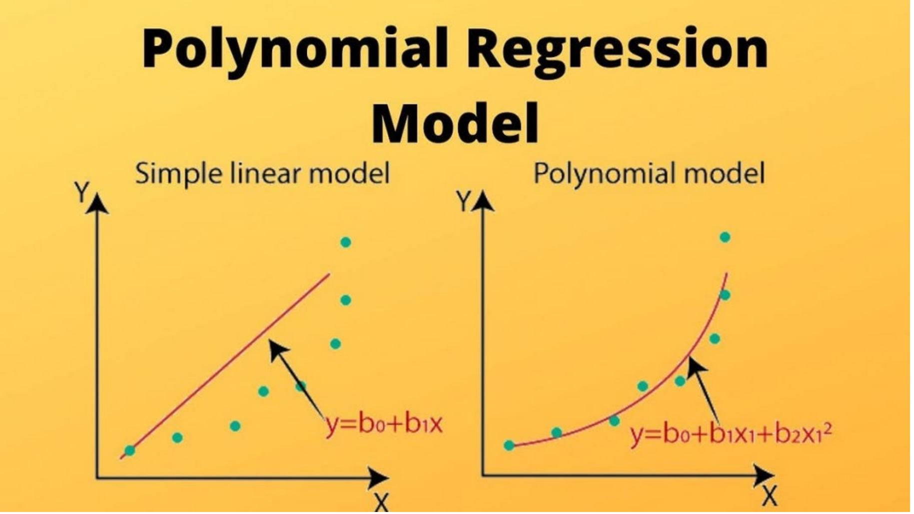


Figure 4-14. High-degree Polynomial Regression



Types of Machine Learning - Regression

Simple Linear Regression

$$y = b_0 + b_1 x_1$$

Multiple Linear Regression

$$y = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$$

Polynomial Linear Regression

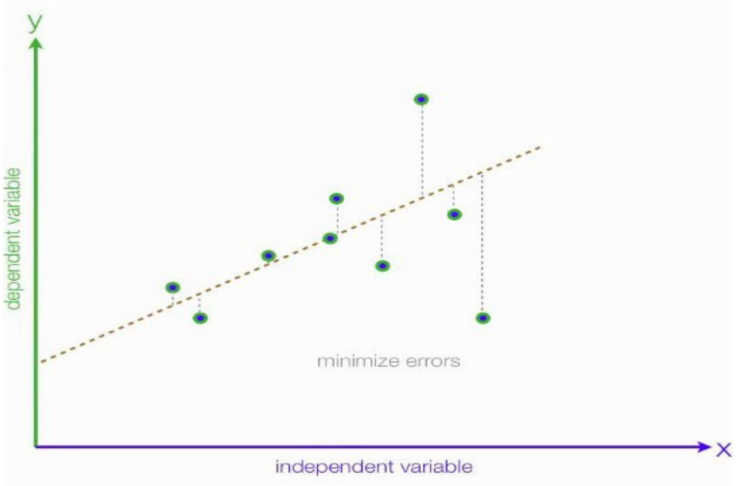
$$y = b_0 + b_1 x_1 + b_2 x_1^2 + \ldots + b_n x_1^n$$

Evaluation Metrics for Regression

Linear Regression- **Error function**

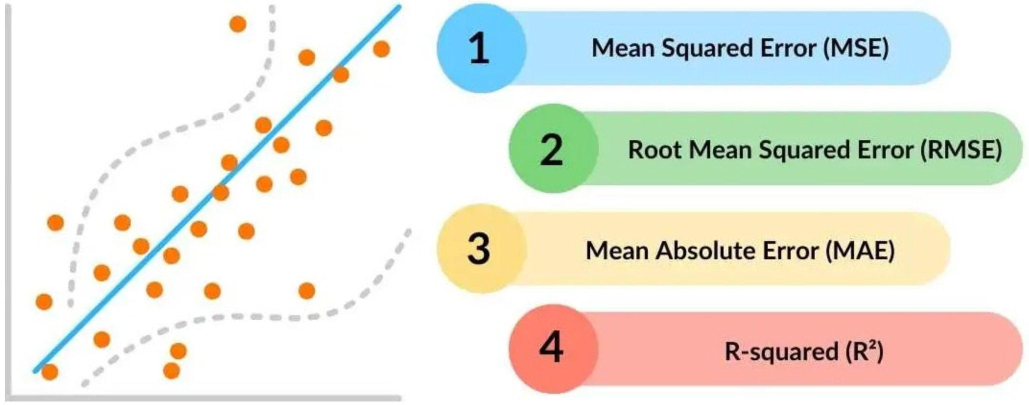
- **Error function.** An error function captures the error between the predictions made by a model and the actual values in a training dataset. There are many **different kinds** of error functions.

- the most commonly used is the sum of squared errors error function.



Evaluation Metrics for Regression

- **Mean Absolute error (MAE)** – It is the average of the absolute difference between predicted values and true values.
- **Mean Squared error (MSE)** – It is the average of the square of the difference between actual and estimated values.
- **Median Absolute error** – It is the median value of the absolute difference between predicted values and true values.
- **Root mean square error (RMSE)** – It is the square root value of the mean squared error (MSE).



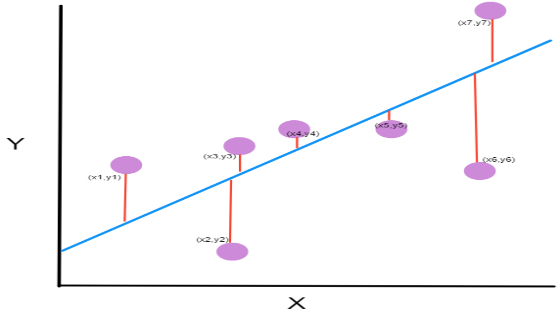
Linear Regression

- the most common performance measure of a regression model is the Root Mean Square Error (RMSE).

Equation 4-3. MSE cost function for a Linear Regression model

$$\mathrm{MSE}(\mathbf{X}, h_\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \mathbf{x}^{(i)} - y^{(i)}\right)^2$$

Predicted value
Actual value



Gradient Descent

Gradient Descent Algorithm

- **Gradient Descent** is a very generic **optimization** algorithm capable of finding optimal solutions to a wide range of problems.

- The general idea of Gradient Descent is to tweak parameters iteratively in order to minimize a cost function.

- **Gradient Descent** used all over the place of machine learning**.**

- **Iterative.**

Gradient Descent

- **Batch**: each step of gradient descent uses all training examples.
- **Stochastic "عشوائي" Gradient Descent (SGD):** random ally instances are picked.
- **Mini-Batch Gradient Descent:** uses small random sets called Mini-Batches.

Gradient Descent Algorithm

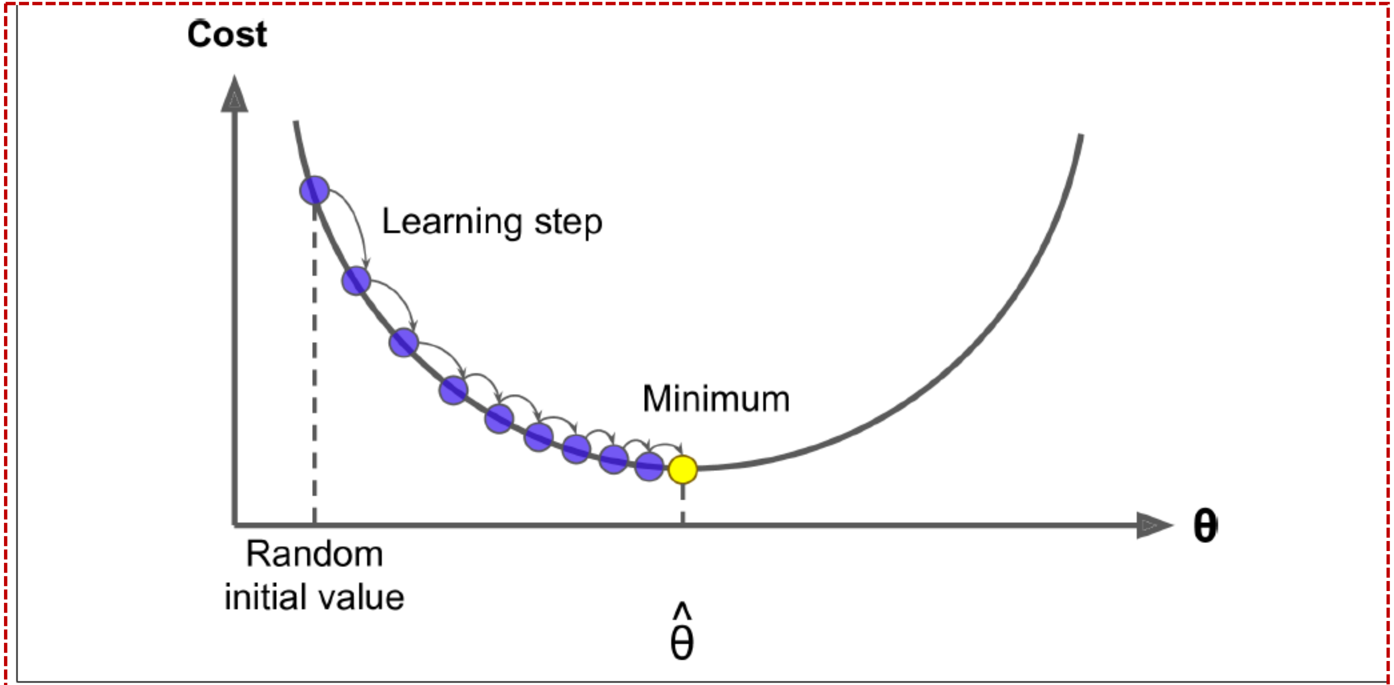
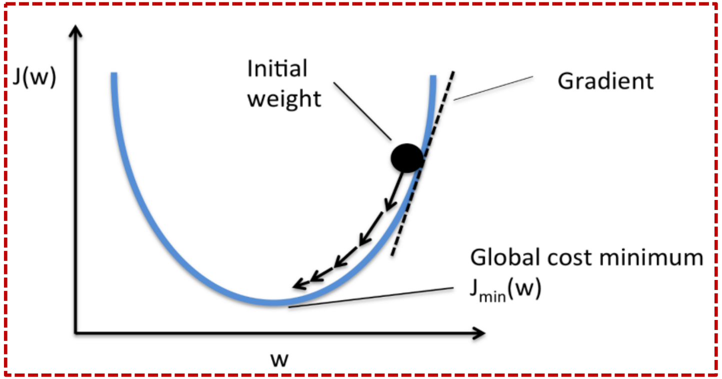
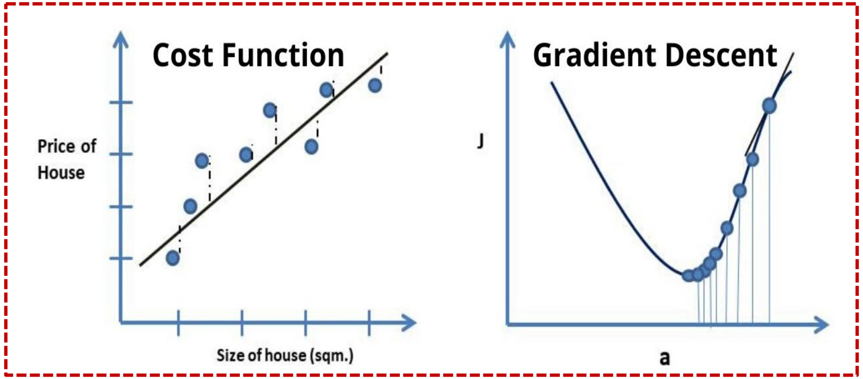


Figure 4-3. Gradient Descent

Gradient Descent Algorithm

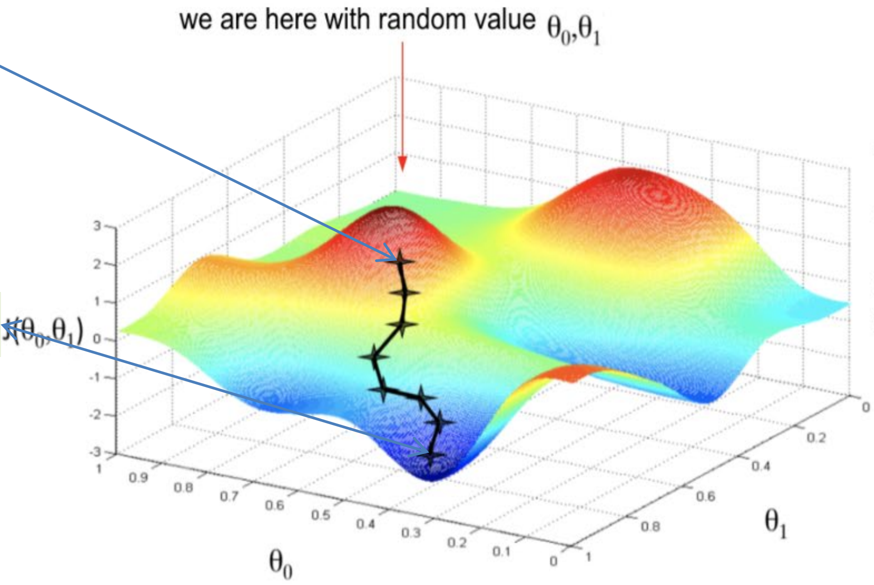




Gradient Descent Algorithm

Starting point

Local minimum

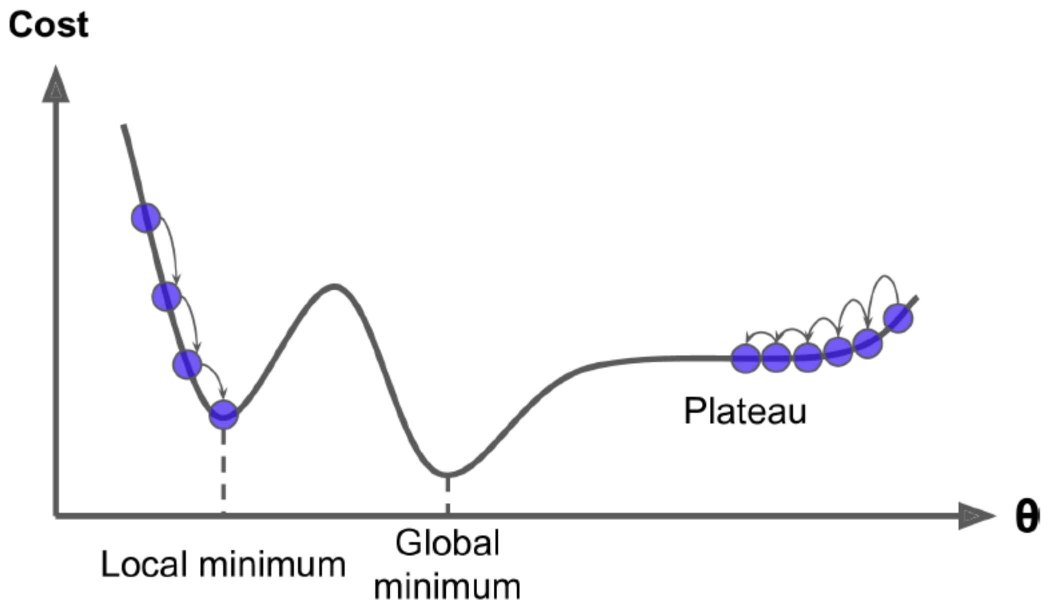


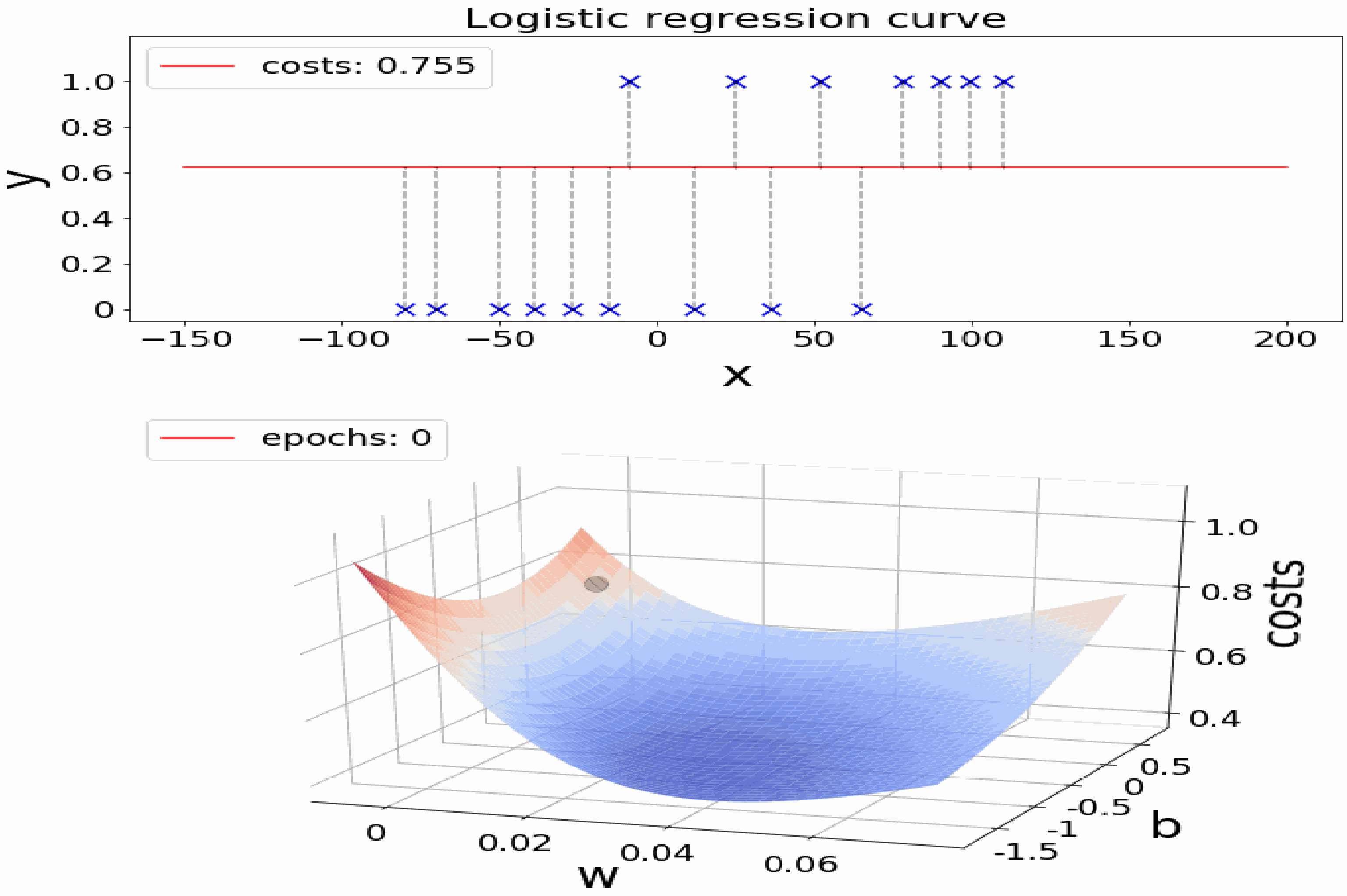
- Start with some θ_0,θ_1
- Keep changing θ_0,θ_1 to reduce $J(\theta_0,\theta_1)$

until we hopefully end up at a minimum

Red: high
Blue: low

Gradient Descent Algorithm




Logistic regression curve
costs: 0.755
1.0
0.8
0.6
0.4
0.2
0
y
−150
−100
−50
0
50
100
150
200
x
epochs: 0
1.0
0.8
0.6
0.4
costs
0.5
0
-0.5
-1
-1.5
b
0
0.02
0.04
0.06
w

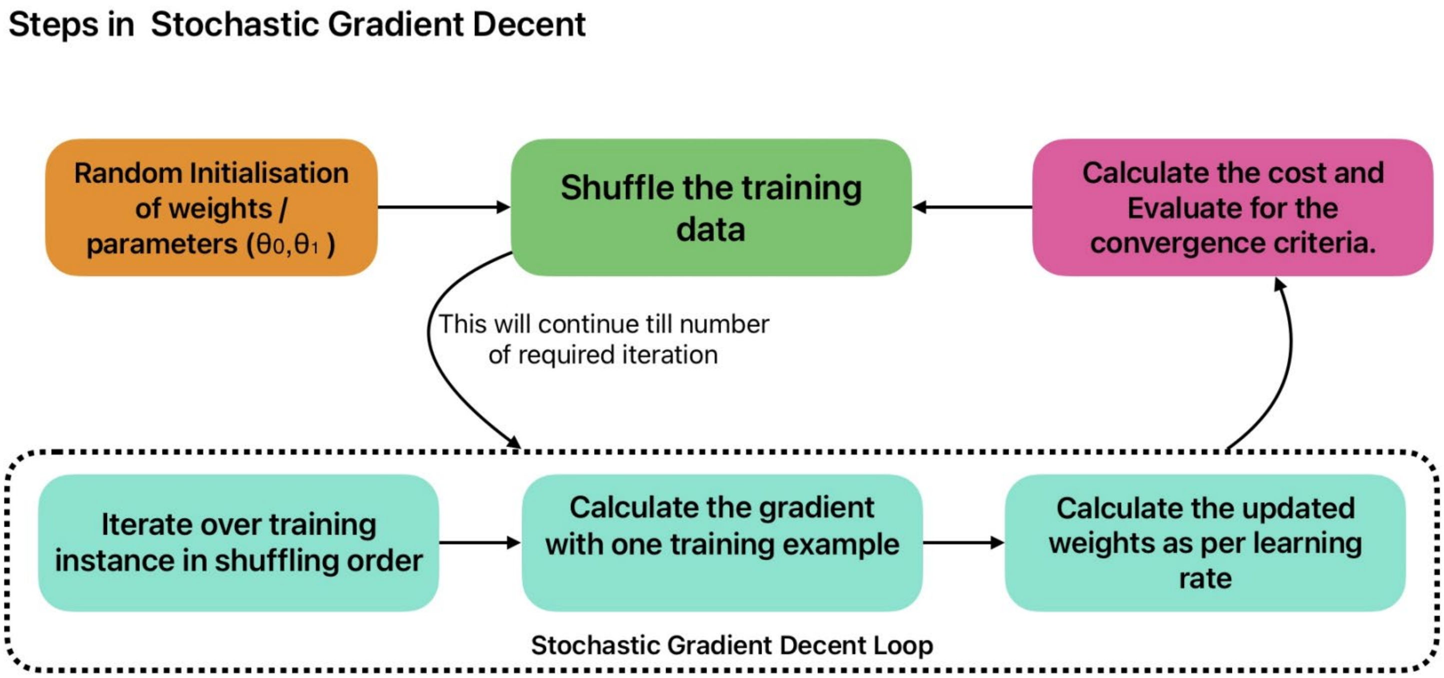
Stochastic Gradient Descent Algorithm

- **Stochastic Gradient Descent** works by randomly selecting a single (or a small mini batch) training example from the dataset and using it to update the model parameters.
- This process is repeated for a fixed number of epochs, or until the model converges to a minimum of the cost function.

Stochastic Gradient Descent Algorithm

- Initialize the model parameters to random values.
- For each epoch, randomly shuffle the training data.
- For each training example:
  - Calculate the gradient of the cost function with respect to the model parameters.
  - Update the model parameters in the opposite direction of the gradient.
- Repeat until convergence

Stochastic Gradient Descent Algorithm



Batch Gradient Descent "Full Gradient Descent"

Equation 4-5. Partial derivatives of the cost function

$$\frac{\partial}{\partial \theta_j} \mathrm{MSE}(\boldsymbol{\theta}) = \frac{2}{m} \sum_{i=1}^{m} \left(\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)} \right) x_j^{(i)}$$

Testing and Validating

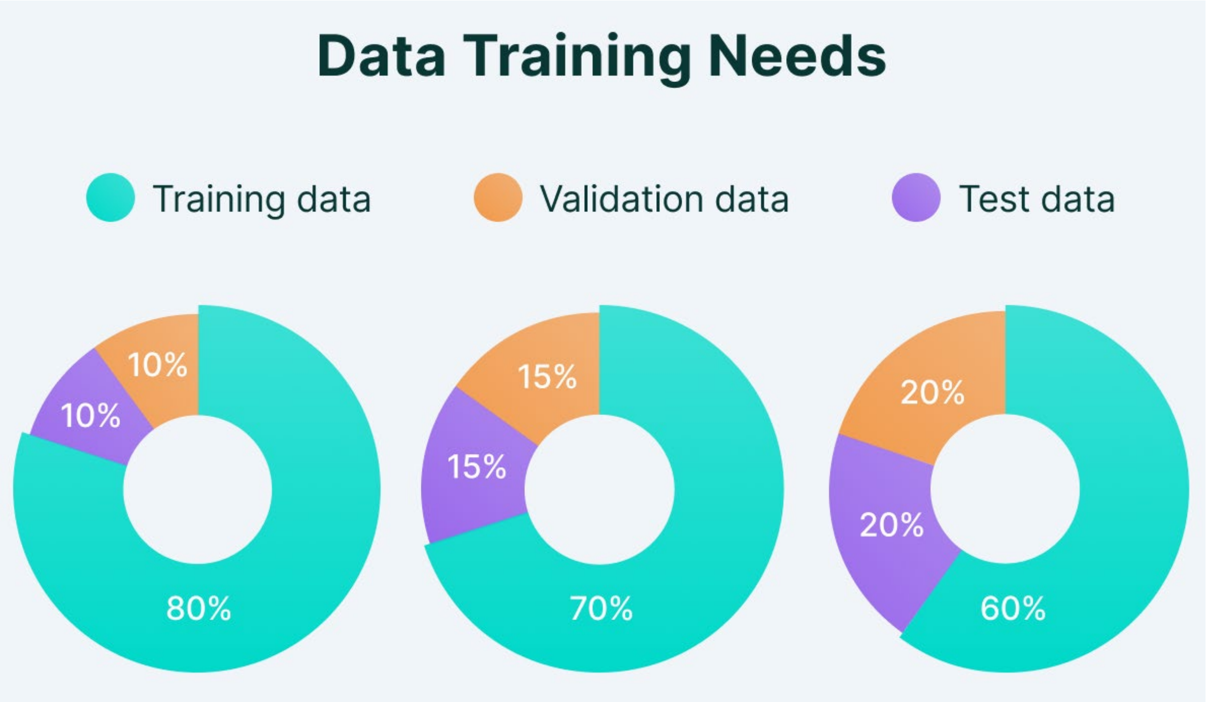
Testing and Validating

Testing and Validating

- A better option is to split your data into two sets: the **training set** and the **test set**. As these names imply, you train your model using the training set, and you test it using the test set.

- The error rate on new cases is called the generalization error (or out-of sample error), and by evaluating your model on the test set, you get an estimate of this error. This value tells you how well your model will perform on instances it has never seen before.

Testing and Validating



Underfitting & Overfitting

Overfitting

Overfitting the Training Data

In Machine Learning this is called **overfitting**: it means that the model performs well on the training data, but it does not **generalize** well.

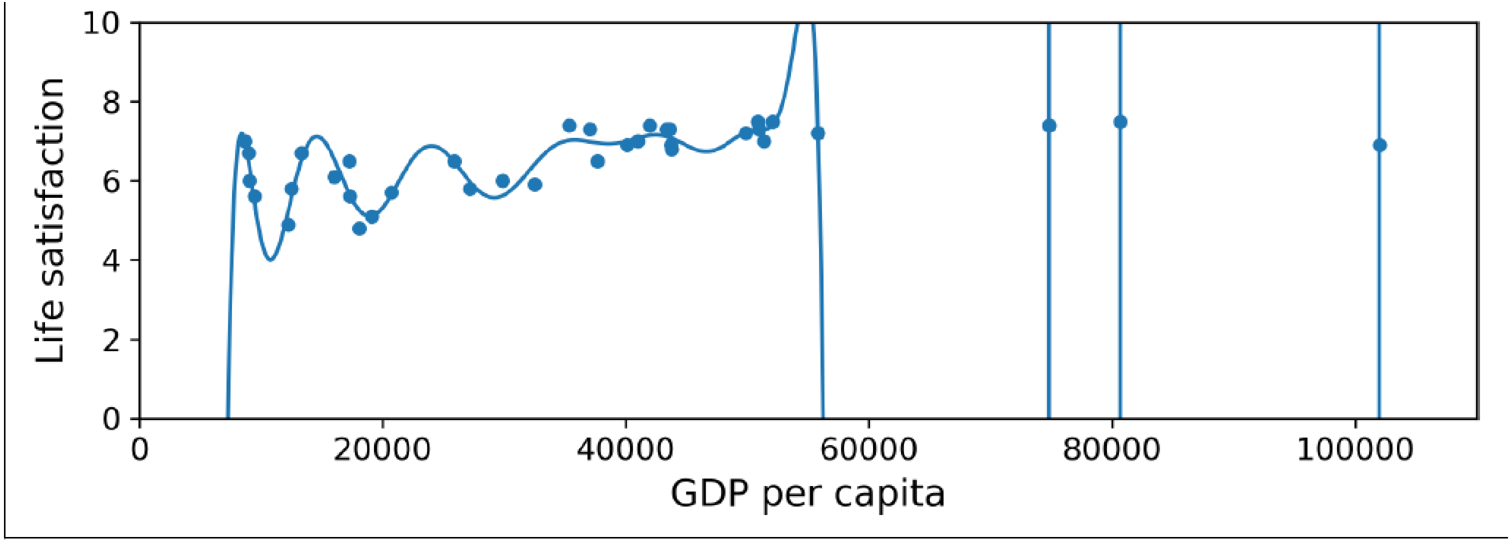


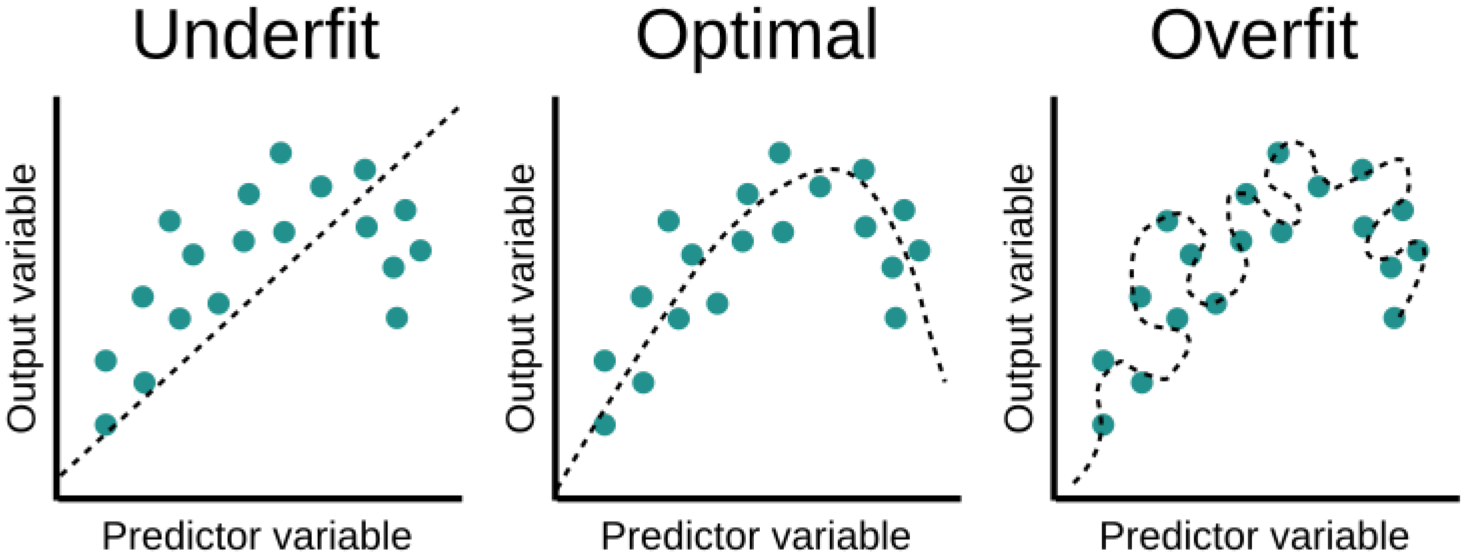
Figure 1-22. Overfitting the training data

More spread

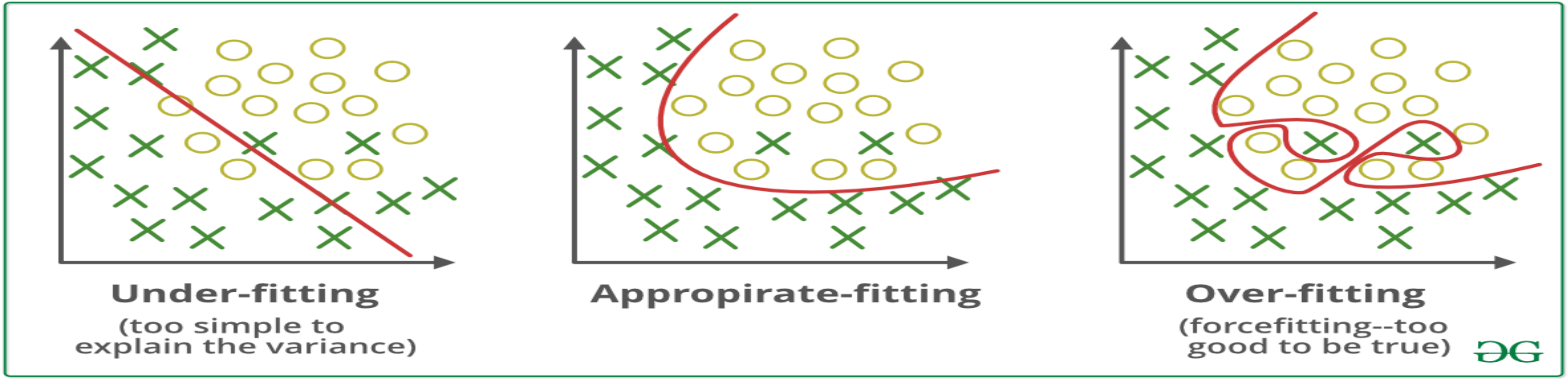
Underfitting

Underfitting the Training Data

- **underfitting** is the opposite of overfitting: it occurs when your model is **too** simple to learn the underlying structure of the data.



Under-fitting & over-fitting



Underfitting

- Poor performance training data
- Poor performance testing data
- High Bias
- Low variance

Overfitting

- Good performance training data
- Poor performance testing data
- Low Bias
- High variance "تفاوت"

❖ **Variance:** the difference in fits between datasets .

Underfitting & Overfitting

What is Bias?

While making predictions, a difference occurs between prediction values made by the model and actual values/expected values, and this difference is known as bias errors or Errors due to bias.

Bias: the inability of ML method to capture the true relationship.

Under-fitting & over-fitting

What is Variance?

- Variance is the measure of spread in data from its mean position.

- In machine learning variance is the amount by which the performance of a predictive model changes when it is trained on different subsets of the training data.
- More specifically, variance is the variability of the model that how much it is **sensitive to another subset** of the training dataset. i.e. how much it can adjust on the new subset of the training dataset.

Overfitting & Underfitting **solutions**

Overfitting happens when the model is too complex relative to the amount and noisiness of the training data.

The possible **solutions** are:

- To simplify the model by selecting fewer parameters
- To gather more training data
- To reduce the noise in the training data.

The main options to fix **Underfitting** problem are:

- Selecting a more powerful model, with more parameters
- Feeding better features to the learning algorithm (feature engineering)
- Reducing the constraints on the model.

Practical Part

Practical part

- **Step 1:** Importing necessary python package

```
import numpy as np from sklearn
import linear_model
import sklearn.metrics as sm
import matplotlib.pyplot as plt
```

- **Step 2:** Importing dataset
- **Step 3:** Organizing data into training & testing sets
- **Step 4:** Model evaluation & prediction

Practical part

- **Step 5:** Plot & visualization

```
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_test, y_test_pred, color = 'black', linewidth = 2)
plt.xticks(())
plt.yticks(())
plt.show()
```

- **Step 6:** Performance computation


Thank You !
Any questions?