|  |
| --- |
|  |
| Scikit Learn |
|  |  |
|  | Python Library for classical machine learning. |
|  |  |
|  | Regression is a statistical measurement used in finance, investing and other disciplines that attempts to determine the strength of the relationship between one dependent variable (usually denoted by Y) and a series of other changing variables (known as independent variables x). |
|  |  |
|  |  |
|  | Dependent Variable OR Output Variable OR Label OR Y |
|  | Independent Variable OR Input Variable OR Feature OR X |
|  |  |
|  |  |
|  | Types of Regressions |
|  |  |
|  | 1. Linear Regression |
|  |  |
|  | A linear regression refers to a regression model that is completely made up of linear variables. Beginning with the simple case, Single Variable Linear Regression is a technique used to model the relationship between a single input independent variable (feature variable) and an output dependent variable using a linear model i.e a line. |
|  |  |
|  |  |
|  | 2. Logistic Regression |
|  |  |
|  | Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression). |
|  |  |
|  |  |
|  | 3. Polynomial Regression |
|  |  |
|  | When we want to create a model that is suitable for handling non-linearly separable data, we will need to use a polynomial regression. In this regression technique, the best fit line is not a straight line. It is rather a curve that fits into the data points. |
|  |  |
|  |  |
|  |  |
|  | #Creating Custom Linear Regression Model Using Straight Line Formula: |
|  |  |
|  | Formula : y = mx + b |
|  |  |
|  |  |
|  | Slope (m) = mean(x).mean(y) - mean(x\*y) / (mean(x))\*\*2 - mean(x\*\*2) |
|  | y Intercept (b) = mean(y) - m\*mean(x) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | Scikit Learn LinearRegression Classifier |
|  | --------------------------------------- |
|  |  |
|  | import numpy as np |
|  | from sklearn.linear\_model import LinearRegression |
|  |  |
|  | x = np.array([1,2,3,4,5]) |
|  | y = np.array([5,4,6,5,6]) |
|  |  |
|  | #Create Classifier Object |
|  | clf = LinearRegression() |
|  |  |
|  | #Train the Classifier |
|  | clf.fit(x,y) |
|  |  |
|  | #Evaluating the Accuracy of the Model |
|  | score=clf.score(x,y) |
|  | print("Accuracy of The Model : ", score) |
|  |  |
|  | #Predictions |
|  | predict\_x=np.arry([6,8,9]) |
|  | predict\_y=clf.predict(predict\_x) |
|  | print(predict\_y) |
|  |  |
|  |  |
| import numpy as np |
|  | from sklearn.linear\_model import LinearRegression |
|  |  |
|  | x = np.array([1,2,3,4,5]) |
|  | x.shape=(-1,1) |
|  | y = np.array([3,4,5,6,7]) |
|  |  |
|  | #Create Classifier Object |
|  | clf = LinearRegression() |
|  |  |
|  | #Train the Classifier |
|  | clf.fit(x,y) |
|  |  |
|  | #Evaluating the Accuracy of the Model |
|  | score=clf.score(x,y) |
|  | print("Accuracy of The Model : ", score) |
|  |  |
|  | #Predictions |
|  | predict\_x=np.array([6,8,9]) |
|  | predict\_x.shape=(-1,1) |
|  | predict\_y=clf.predict(predict\_x) |
|  | print("Prediction Data : ", predict\_y) |