GOVERNMENT POLYTECHNIC CHITRADURGA-577501

5TH SEMESTER

AI/ML WEEK-2

WEEK-6

Data Splitting

Data splitting is when data is divided into two or more subsets. Typically, with a two-part split, one part is used to evaluate or test the data and the other to train the model.

Data splitting is an important aspect of data science, particularly for creating models based on data.

Importance of data splitting

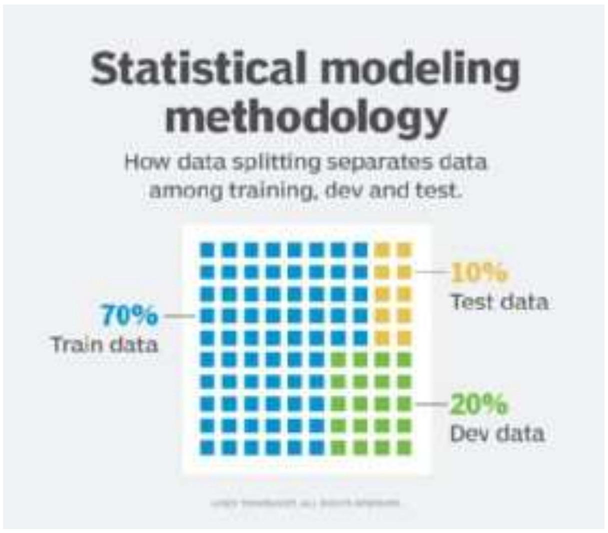
This technique helps ensure the creation of data models and processes that use data models -- such as machine learning -- are accurate.

Data splitting in machine learning

In machine learning, data splitting is typically done to avoid overfitting. That is an instance where a machine learning model fits its training data too well and fails to reliably fit additional data.

The original data in a machine learning model is typically taken and split into three or four sets. The three sets commonly used are the training set, the dev set and the testing set:

1. The **training set** is the portion of data used to train the model. The model should observe and learn from the training set, optimizing any of its parameters.
2. The **dev set** is a data set of examples used to change learning process parameters. It is also called the *crossvalidation* or *model validation set*. This set of data has the goal of ranking the model's accuracy and can help with model selection.
3. The **testing set** is the portion of data that is tested in the final model and is compared against the previous sets of data. The testing set acts as an evaluation of the final mode and algorithm.



How data splitting works

In a basic two-part data split, the training data set is used to train and develop models. Training sets are commonly used to estimate different parameters or to compare different model performance.

The testing data set is used after the training is done. The training and test data are compared to check that the final model works correctly. With machine learning, data is commonly split into three or more sets. With three sets, the additional set is the dev set, which is used to change learning process parameters.

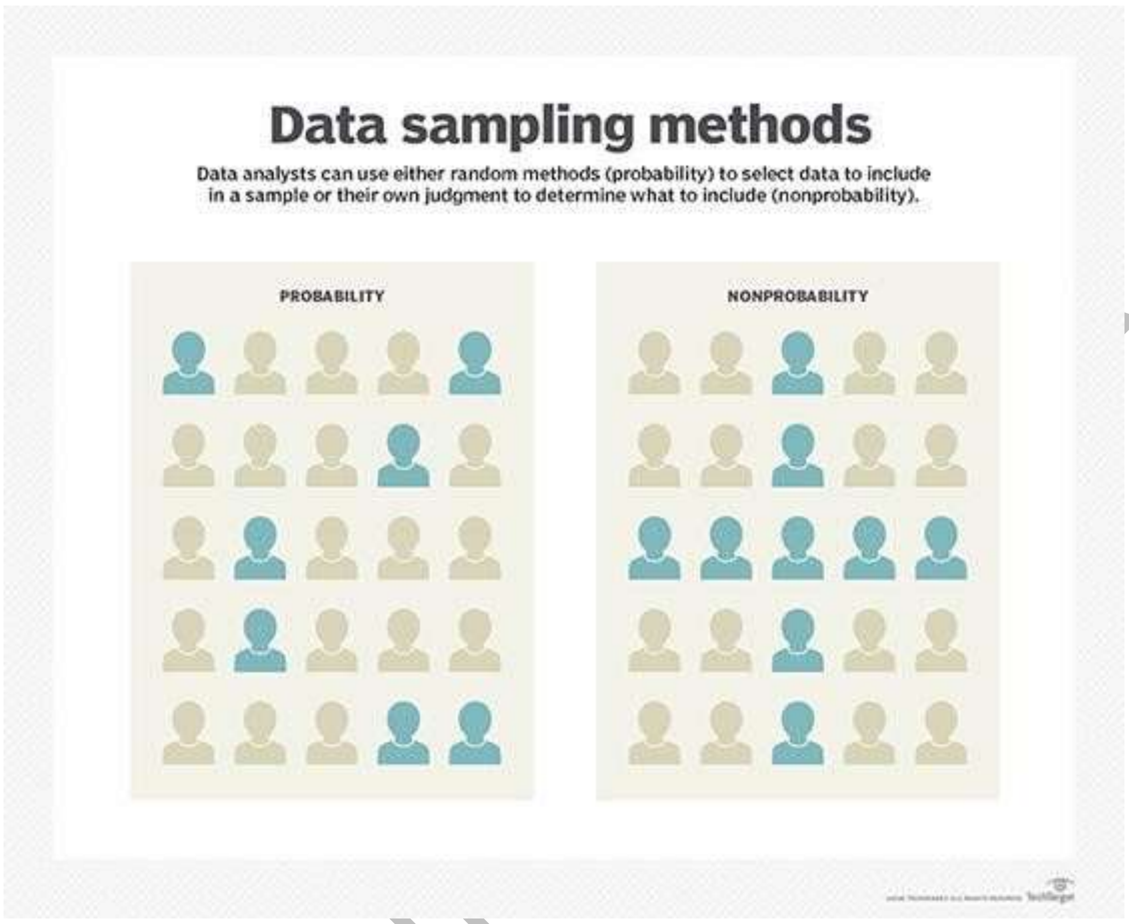
There is no set guideline or metric for how the data should be split; it may depend on the size of the original data pool or the number of predictors in a predictive model. Organizations and data modelers may choose to separate split data based on data sampling methods, such as the following three methods:

1. **Random sampling.** This data sampling method protects the data modeling process from bias toward different possible data characteristics. However, random splitting may have issues regarding the uneven distribution of data.

2. **Stratified random sampling.** This method selects data samples at random within specific parameters. It ensures the data is correctly distributed in training and test sets.

3. **Nonrandom sampling.** This approach is typically used when data modelers want the most recent data as the test set.

With data splitting, organizations don't have to choose between using the data for analytics versus statistical analysis, since the same data can be used in the different processes.


Data sampling methods
Data analysts can use either random methods (probability) to select data to include in a sample or their own judgment to determine what to include (nonprobability).
PROBABILITY
NONPROBABILITY
TechTarget

Machine Learning Pipeline

What is a Machine Learning Pipeline?

A machine learning pipeline is the end-to-end construct that orchestrates the flow of data into, and output from, a machine learning model (or set of multiple models). It includes raw data input, features, outputs, the machine learning model and model parameters, and prediction outputs.

Why is a Machine Learning Pipeline Important?

The design and implementation of a machine learning pipeline is at the core of enterprise AI software applications and fundamentally determines the performance and effectiveness. In addition to the software design, additional factors must be considered, including choice of machine learning libraries and runtime environments (processor requirements, memory, and storage). Many realworld machine learning use cases involve complex, multi-step pipelines. Each step may require different libraries and runtimes and may need to execute on specialized hardware profiles. It is therefore critical to factor in management of libraries, runtimes, and hardware profiles during algorithm development and ongoing maintenance activities. Design choices can have a significant impact on both costs and algorithm performance.

What is Regression and Classification in Machine Learning?

Data scientists use many different kinds of machine learning algorithms to discover patterns in big data that lead to actionable insights. At a high level, these different algorithms can be classified into two groups based on the way they "learn" about data to make predictions: supervised and unsupervised learning.

Supervised Machine Learning: The majority of practical machine learning uses supervised learning. Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output **Y = f(X)** . The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data. Techniques of Supervised Machine Learning algorithms

include **linear** and **logistic regression, multi-class classification, Decision Trees and support vector machines**. Supervised learning requires that the data used to train the algorithm is already labelled with correct answers. For example, a classification algorithm will learn to identify animals after being trained on a dataset of images that are properly labelled with the species of the animal and some identifying characteristics. Supervised learning problems can be further grouped into **Regression** and **Classification** problems. Both problems have a goal of the construction of a succinct model that can predict the value of the dependent attribute from the attribute variables. The difference between the two tasks is the fact that the dependent attribute is nZZZZZZZZZZZZZZZZumerical for regression and categorical for classification.

What is Regression?

consists of a set of *machine learning* methods that allow us to predict a continuous outcome variable (y) based on the value of one or multiple predictor variables (x).
Briefly, the goal of regression model is to build a mathematical equation that defines y as a function of the x variables.

6 Types of Regression Models in Machine Learning You Should Know About

- Linear Regression.
- Logistic Regression.
- Ridge Regression.
- Lasso Regression.
- Polynomial Regression.
- Bayesian Linear Regression.

Formula of linear Regression:

$$Y_i = f(X_i, \beta) + e_i$$

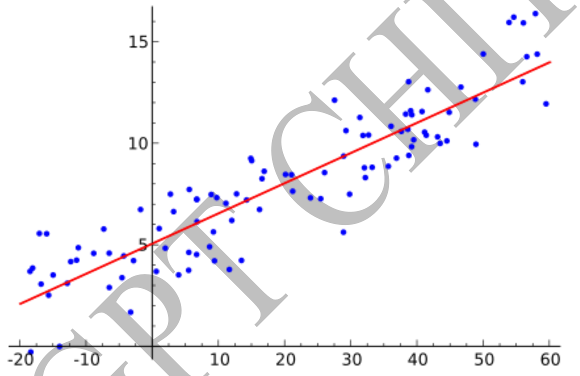
Yi = dependent variable f = function

Xi = independent variable \beta = unknown parameters ei = error terms

Types of Regression

1. Linear Regression

Linear regression is one of the most basic types of regression in machine learning. The linear regression model consists of a predictor variable and a dependent variable related linearly to each other. In case the data involves more than one independent variable, then linear regression is called multiple linear regression models.
The below-given equation is used to denote the linear regression model:
y=mx+c+e where m is the slope of the line, c is an intercept, and e represents the error in the model.



Source
The best fit line is determined by varying the values of m and c. The predictor error is the difference between the observed values and the predicted value. The values of m and c get selected in such a way that it gives the minimum predictor error. It is important to note that a simple linear regression model is susceptible to outliers. Therefore, it should not be used in case of big size data.
There are different types of linear regression. The two major types of linear regression are simple linear regression and multiple linear regression. Below is the formula for simple linear regression.

- Here, **y** is the predicted value of the dependent variable (**y**) for any value of the independent variable (**x**)
- **β0** is the intercepted, aka the value of **y** when **x** is zero
- **β1** is the regression coefficient, meaning the expected change in **y** when **x** increases
- x is the independent variable
- ∈ is the estimated error in the regression Simple linear regression can be used:
- To find the intensity of dependency between two variables. Such as the rate of carbon emission and global warming.
- To find the value of the dependent variable on an explicit value of the independent variable. For example, finding the amount of increase in atmospheric temperature with a certain amount of carbon dioxide emission.

In multiple linear regression, a relationship is established between two or more independent variables and the corresponding dependent variables. Below is the equation for multiple linear regression.

- Here, **y** is the predicted value of the dependent variable
- **β0** = Value of y when other parameters are zero
- **β1X1**= The regression coefficient of the first variable
- …= Repeating the same no matter how many variables you test
- **βnXn=** Regression coefficient of the last independent variable
- ∈ = Estimated error in the regression Multiple linear regression can be used:
- To estimate how strongly two or more independent variables influence the single dependent variable. Such as how location, time, condition, and area can influence the price of a property.
- To find the value of the dependent variables at a definite condition of all the independent variables. For example, finding the price of a property located at a certain place, with a specific area and its condition.

Also visit upGrad's Degree Counselling page for all undergraduate and postgraduate programs.

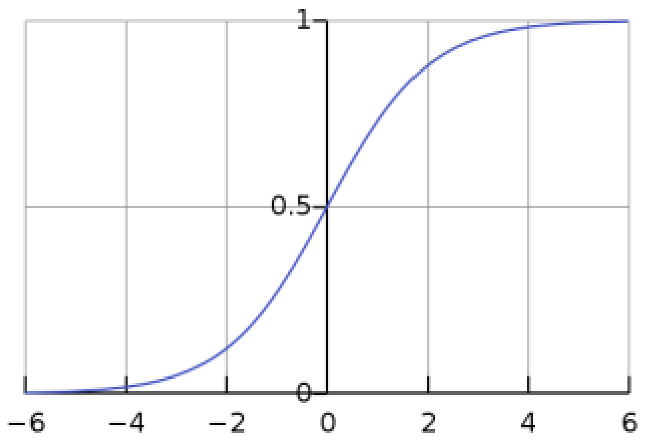
2. Logistic Regression

Logistic regression is one of the types of regression analysis technique, which gets used when the dependent variable is discrete. Example: 0 or 1, true or false, etc. This means the target variable can have only two values, and a sigmoid curve denotes the relation between the target variable and the independent variable.

Logit function is used in Logistic Regression to measure the relationship between the target variable and independent variables. Below is the equation that denotes the logistic regression.

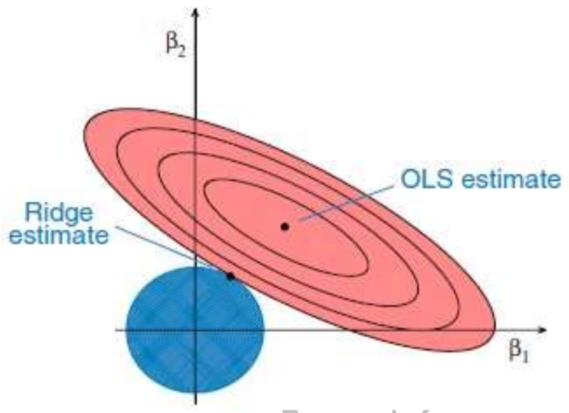
$\text{logit}(p) = \ln(p/(1-p)) = b0+b1X1+b2X2+b3X3\ldots.+bkXk$

where p is the probability of occurrence of the feature.



Source
For selecting logistic regression, as the regression analyst technique, it should be noted, the size of data is large with the almost equal occurrence of values to come in target variables. Also, there should be no multicollinearity, which means that there should be no correlation between independent variables in the dataset.

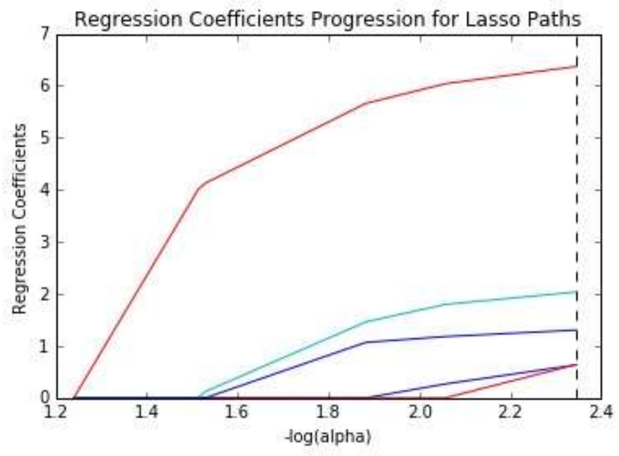
3. Ridge Regression



Source
This is another one of the types of regression in machine learning which is usually used when there is a high correlation between the independent variables. This is because, in the case of multi collinear data, the least square estimates give unbiased values. But, in case the collinearity is very high, there can be some bias value. Therefore, a bias matrix is introduced in the equation of Ridge Regression. This is a powerful regression method where the model is less susceptible to overfitting.
Below is the equation used to denote the Ridge Regression, where the introduction of λ (lambda) solves the problem of multicollinearity:
β = (X^{T}X + λ*I)^{-1}X^{T}y
Check out: 5 Breakthrough Applications of Machine Learning

4. Lasso Regression
Lasso Regression is one of the types of regression in machine learning that performs regularization along with feature selection. It prohibits the absolute size of the regression coefficient. As a result, the coefficient value gets nearer to zero, which does not happen in the case of Ridge Regression.

Due to this, feature selection gets used in Lasso Regression, which allows selecting a set of features from the dataset to build the model. In the case of Lasso Regression, only the required features are used, and the other ones are made zero. This helps in avoiding the overfitting in the model. In case the independent variables are highly collinear, then Lasso regression picks only one variable and makes other variables to shrink to zero.



Source

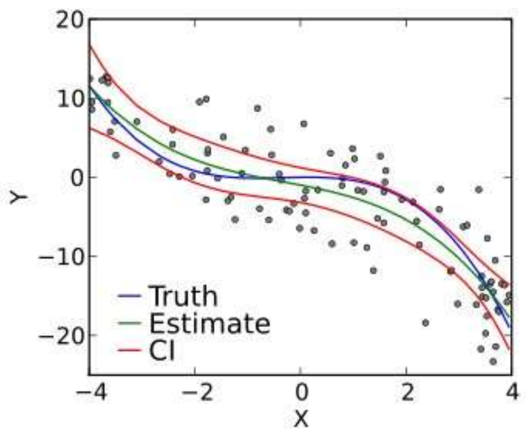
Below is the equation that represents the Lasso Regression method:

N^{-1}Σ^{N}_{i=1}f(x_{i}, y_{I}, α, β)

5. Polynomial Regression

Polynomial Regression is another one of the types of regression analysis techniques in machine learning, which is the same as Multiple Linear Regression with a little modification. In Polynomial Regression, the relationship between independent and dependent variables, that is X and Y, is denoted by the n-th degree.

It is a linear model as an estimator. Least Mean Squared Method is used in Polynomial Regression also. The best fit line in Polynomial Regression that passes through all the data points is not a straight line, but a curved line, which depends upon the power of X or value of n.



Source

While trying to reduce the Mean Squared Error to a minimum and to get the best fit line, the model can be prone to overfitting. It is recommended to analyze the curve towards the end as the higher Polynomials can give strange results on extrapolation.

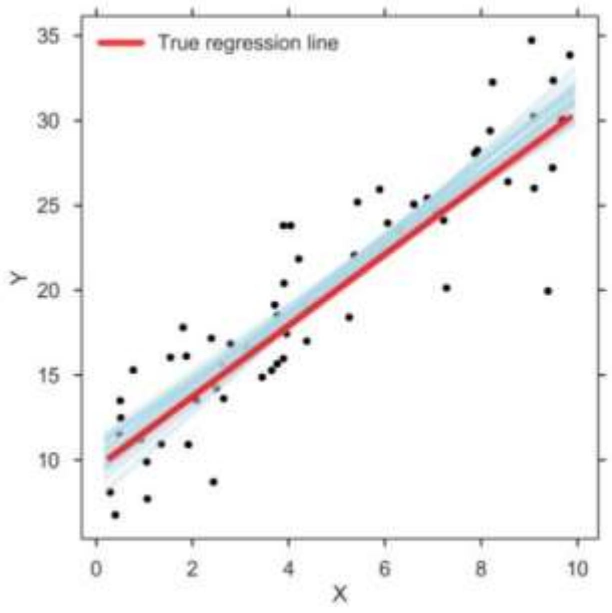
Below equation represents the Polynomial

Regression: l = β0+ β0x1+ε

Read: Machine Learning Project Ideas

6. Bayesian Linear Regression

Bayesian Regression is one of the types of regression in machine learning that uses the Bayes theorem to find out the value of regression coefficients. In this method of regression, the posterior distribution of the features is determined instead of finding the least-squares. Bayesian Linear Regression is like both Linear Regression and Ridge Regression but is more stable than the simple Linear Regression.



Source

People often wonder "what is regression in AI" or "what is regression in machine learning". Machine learning is a subset of AI; hence, both questions have the same answer. In the case of regression in AI, different algorithms are used make a machine learn the relationship between the provided data sets and make predictions accordingly. Hence, regression in AI is mainly used to add a level of automation to the machines. Regression AI is often used in sectors like finance and investment, where establishing a relationship between a single dependent variable and multiple independent variables is a common case. A common example of regression AI will be factors that estimate a house's price based on its location, size, ROI, etc.

Regression plays a vital role in predictive modelling and is found in many machine learning applications. Algorithms from the regressions provide different perspectives regarding the relationship between the variables and their outcomes. These set models could then be used as a guideline for fresh input data or to find missing data.

As the models are trained to understand a variety of relationships between different variables, they are often extremely helpful in predicting the portfolio performance or stocks and trends. These implementations fall under machine learning in finance.

The very common use of regression in AI includes:

- Predicting a company's sales or marketing success
- Generating continuous outcomes like stock prices
- Forecasting different trends or customer's purchase behaviour

Hope this helped to understand what is regression in AI or what is regression in machine learning.

Regularization in ML

Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from overfitting by adding extra information to it. Sometimes the machine learning

model performs well with the training data but does not perform well with the test data. It means the model is not able to predict the output when deals with unseen data by introducing noise in the output, and hence the model is called overfitted. This problem can be deal with the help of a regularization technique.

This technique can be used in such a way that it will allow to maintain all variables or features in the model by reducing the magnitude of the variables. Hence, it maintains accuracy as well as a generalization of the model.

It mainly regularizes or reduces the coefficient of features toward zero. In simple words, "*In regularization technique, we reduce the magnitude of the features by keeping the same number of features.*"

How Does it Work?

Python has methods for finding a relationship between data-points and to draw a line of linear regression. We will show you how to use these methods instead of going through the mathematic formula.

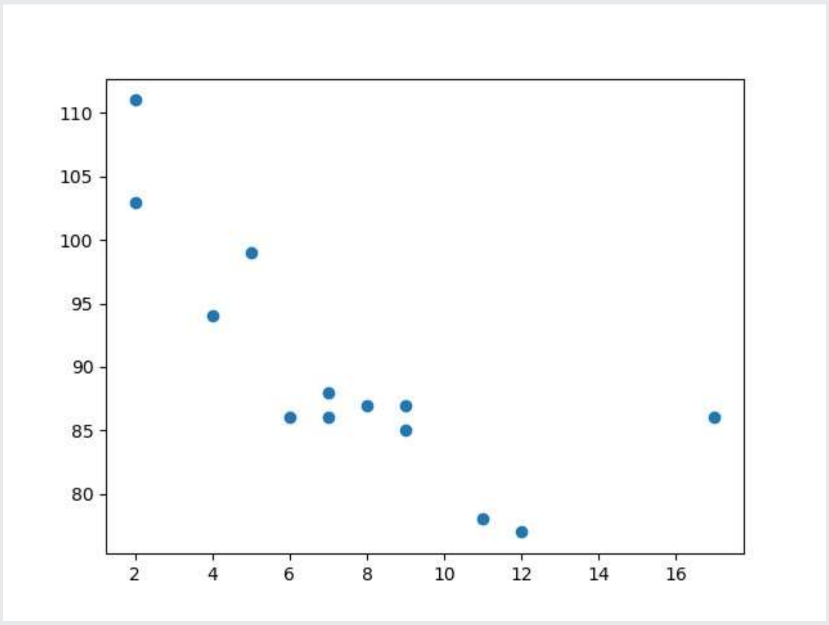
In the example below, the x-axis represents age, and the y-axis represents speed. We have registered the age and speed of 13 cars as they were passing a tollbooth. Let us see if the data we collected could be used in a linear regression:

Example

Start by drawing a scatter plot:

```
import matplotlib.pyplot as plt

x  = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y  = [99,86,87,88,111,86,103,87,94,78,77,85,86]

plt.scatter(x, y) plt.show()
```

Result:



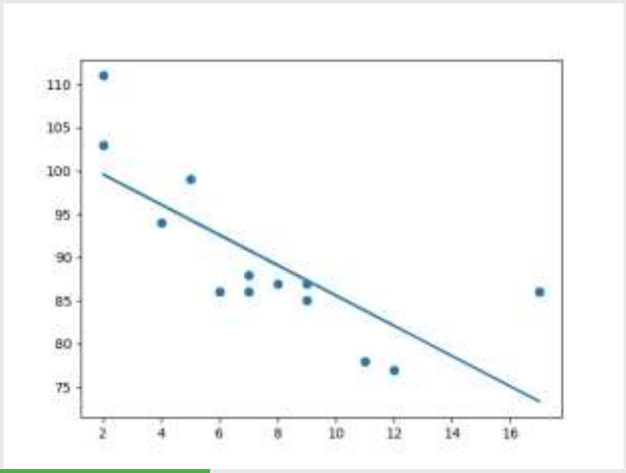
Run example »

Example

Import scipy   d draw the line of Linear Regression:

```
import matplotlib.pyplot as plt
from scipy import stats

x = [5,7,8,7,2,17,2,9,4,11,12,9,6] y =
[99,86,87,88,111,86,103,87,94,78,77,85,86]

slope, intercept, r, p, std_err = stats.linregress(x, y)
 def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))
 plt.scatter(x, y) plt.plot(x,
mymodel) plt.show()
```

Result:



Run example »

Example Explained

Import the modules you need.

You can learn about the Matplotlib module in our Matplotlib Tutorial.

You can learn about the SciPy module in our SciPy Tutorial.

```
import matplotlib.pyplot as plt from scipy import
stats
```

Create the arrays that represent the values of the x and y axis:

```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6] y =
[99,86,87,88,111,86,103,87,94,78,77,85,86]
```

Execute a method that returns some important key values of Linear Regression:

```
slope, intercept, r, p, std_err = stats.linregress(x, y)
```

Create a function that uses the slope and intercept values to return a new value. This new value represents where on the y-axis the corresponding x value will be placed:

```
def myfunc(x):   return slope * x +
intercept
```

Run each value of the x array through the function. This will result in a new array with new values for the y-axis:

```
mymodel = list(map(myfunc, x))
```

Draw the original scatter plot:

```
plt.scatter(x, y)
```

Draw the line of linear regression:

```
plt.plot(x, mymodel)
```

Display the diagram:

```
plt.show()
```

Types

- **Simple linear regression**
- **Multiple linear regression**
- **Polynomial linear regression**

<u>Simple linear regression</u>

Simple linear regression is **a regression model that estimates the relationship between one independent variable and one dependent variable using a straight line**. Both variables should be quantitative.

Formula :

y=\alpha+ \beta x

\beta =slope

\alpha = y-intercept

Y = y- coordinate

X = x-coordinate

<u>Multiple linear regression</u>

Multiple linear regression refers to a statistical technique that uses two or more independent variables to predict the outcome of a dependent variable.

Multiple linear regression (MLR), also known simply as multiple regression, is **a statistical technique that uses several explanatory variables to predict the outcome of a response variable**. Multiple regression is an extension of linear (OLS) regression that uses just one explanatory variable.

Multiple Linear Regression Formula

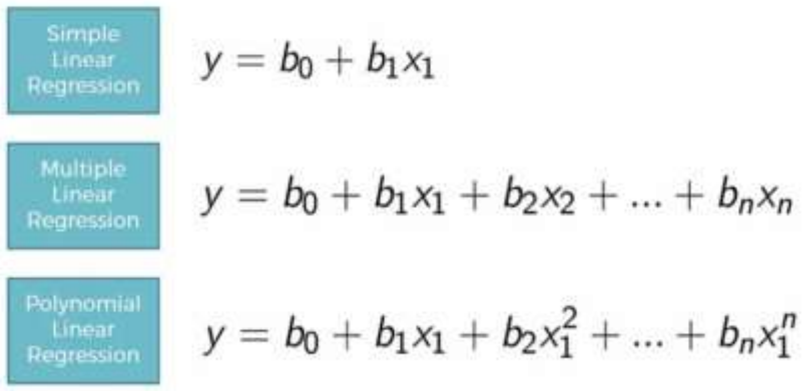
$$yi = \beta 0 + \beta 1xi1 + \beta 2xi2 + \ldots + \beta pxip + \epsilon$$

Where:

- **yi** is the dependent or predicted variable
- **β0** is the y-intercept, i.e., the value of y when both xi and x2 are 0.
- **β1** and **β2** are the regression coefficients representing the change in y relative to a one-unit change in **xi1** and **xi2**, respectively. • **βp** is the slope coefficient for each independent variable
- **ϵ** is the model's random error (residual) term.

Polynomial linear regression:

Polynomial Regression is **a form of Linear regression known as a special case of Multiple linear regression which estimates the relationship as an nth degree polynomial**. Polynomial Regression is sensitive to outliers so the presence of one or two outliers can also badly affect the performance.



Simple Linear Regression	$y = b_0 + b_1x_1$
Multiple Linear Regression	$y = b_0 + b_1x_1 + b_2x_2 + \ldots + b_nx_n$
Polynomial Linear Regression	$y = b_0 + b_1x_1 + b_2x_1^2 + \ldots + b_nx_1^n$

Evaluate regression Model:

Regression is a type of Machine learning which helps in finding the relationship between independent and dependent variable. In simple words, Regression can be defined as a Machine learning problem where we have to predict discrete values like price, Rating, Fees, etc.

Evaluation metrics:

An evaluation metric **quantifies the performance of a predictive model**. This typically involves training a model on a dataset, using the model to make predictions on a holdout dataset not used during training, then comparing the predictions to the expected values in the holdout dataset

- In machine learning, evaluation metrics are used to measure the performance of machine learning models/algorithms.
- Evaluation metrics are crucial. Based on the model performance we are giving decisions.
- We should remember that we are not just only looking for a better model, also looking for our end goal.
- Let's imagine our end goal is to make an application to detect fraud.
- We develop our model based on the data in hand, which contains 99.5% nonfraud cases and %.5 fraud cases.
- Without using the correct evaluation metric on this imbalanced data we will deploy the model with poor performance and prediction on the real data.

linkcode

- In this study we will divide our evaluation metrics into two categories.
  - Classification Evaluation Metrics
  - Regression Evaluation Metrics

- Classification Evalution Metrics
  - Confusion Matrix
  - Accuracy
  - Precision & Recall
  - F Score (F Measure)
  - ROC Curve (AUC)
  - Log Loss

- Regression Evaluation Metrics
  - Mean Absolute Error(MAE)
  - Mean Squared Error (MSE)
  - Root Mean Squared Error (RMSE)
  - R Squared (R2)

Coefficient of Determination or R-squared

What Is R-Squared?

R-squared (R^2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. Whereas correlation explains the strength of the relationship between an independent and dependent variable, Rsquared explains to what extent the variance of one variable explains the variance of the second variable. So, if the R^2 of a model is 0.50, then approximately half of the observed variation can be explained by the model's inputs.

Formula for R-Squared

Formula

$$R^2 = 1 - \frac{RSS}{TSS}$$

R2=coefficient of determination

RSS=sum of squares of residuals

TSS=total sum of squares

R2=1−Unexplained Variation/Total Variation

The actual calculation of R-squared requires several steps. This includes taking the data points (observations) of dependent and independent variables and finding the line of best fit, often from a regression model. From there you would calculate predicted values, subtract actual values and square the results. This yields a list of errors squared, which is then summed and equals the unexplained variance.

To calculate the total variance, you would subtract the average actual value from each of the actual values, square the results and sum them. From there, divide the first sum of errors (explained variance) by the second sum (total variance), subtract the result from one, and you have the R-squared.

Root Mean Squared Error (RMSE)

RMSE is the most popular evaluation metric used in regression problems. It follows an assumption that error are unbiased and follow a normal distribution. Here are the key points to consider on RMSE:

1. The power of 'square root' empowers this metric to show large number deviations.
2. The 'squared' nature of this metric helps to deliver more robust results which prevents cancelling the positive and negative error values. In other words, this metric aptly displays the plausible magnitude of error term.
3. It avoids the use of absolute error values which is highly undesirable in mathematical calculations.
4. When we have more samples, reconstructing the error distribution using RMSE is considered to be more reliable.
5. RMSE is highly affected by outlier values. Hence, make sure you've removed outliers from your data set prior to using this metric.
6. As compared to mean absolute error, RMSE gives higher weightage and punishes large errors.

RMSE metric is given by:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

where, N is Total Number of Observations.

Optimize regression

model Use the

above link to get

model:

https://machinelearningmastery.com/optimizeregressionmodels/#:~:text=Optimize%20

Regression%20Models,R

egression%20models%2C%20like&text=These%20regression%2

0models%

20involve%20the,optimization%20algorithms%20can%20be%20u sed.

Cross-validation

Cross-validation is **a technique for evaluating ML models by training several ML models on subsets of the available input data and evaluating them on the complementary subset of the data**. Use cross-validation to detect overfitting, ie, failing to generalize a pattern.

Cross-validation is a technique in which we train our model using the subset of the data-set and then evaluate using the complementary subset of the data-set.

The three steps involved in cross-validation are as follows :

1. Reserve some portion of sample data-set.
2. Using the rest data-set train the model.
3. Test the model using the reserve portion of the data-set.

Why do we need cross-validation?

Cross-validation is primarily used in applied machine learning **to estimate the skill of a machine learning model on unseen data**. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

The **purpose of cross–validation** is to test the ability of a machine learning model to predict new data. It is also used to flag problems like overfitting or selection bias and gives insights on how the model will generalize to an independent dataset.

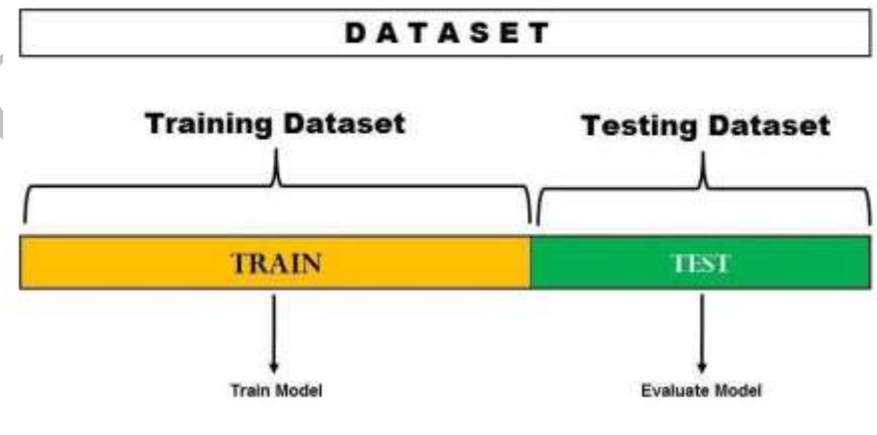
Methods of Cross Validation Hold Out method

Leave One Out cross validation

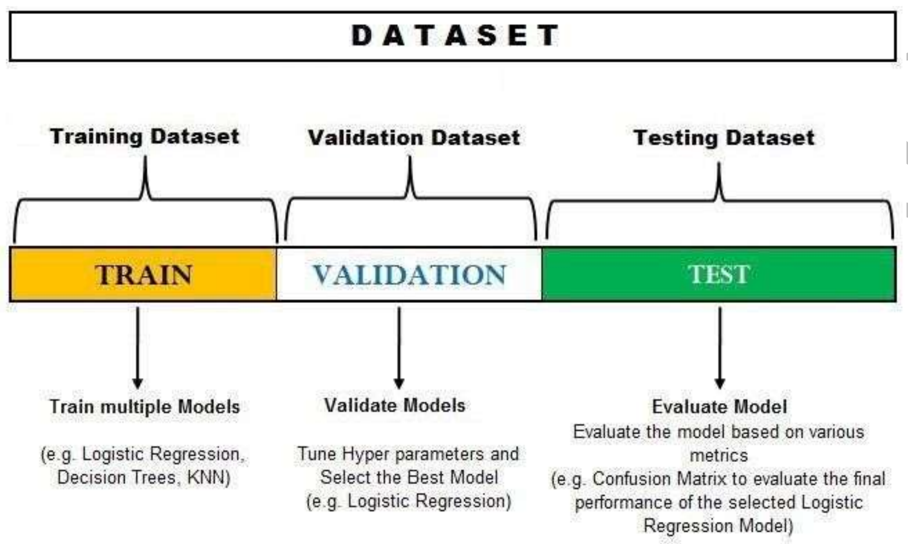
K-fold cross Validation

Holdout method

This is the classic "simplest kind of cross-validation". This method is often classified as a type of "simple validation, rather than a simple or degenerate form of cross-validation".
In this method, we randomly divide our data into two: Training and Test/Validation set i.e. a hold-out set. We then train the model on the training dataset and evaluate the model on the Test/Validation dataset. The model evaluation techniques used on the validation dataset to compute the error depends on the kind of problem we are working with such as MSE being used for Regression problems while various metrics providing with the misclassification rate helping in finding the error for classification problems. Typically the training dataset is bigger than the hold-out dataset. Typical ratios used for splitting the data set include 60:40, 80:20 etc. This method is only used when we only have one model to evaluate and no hyper-parameters to tune.



The limitation of such a method is that the error found in the test dataset can highly depend on the observations included in the train and test dataset. Also if the train or test dataset are not able to represent the actual complete data then the results from the test sets can be skewed. This method is not effective for comparing multiple models and tuning their hyperparameters which leads us to another very popular form of the holdout method which includes the splitting of data into not two, but three separate sets. Here we divide the training dataset into training and validation set thus dividing the original dataset into Training, Validation and Test set.


DATASET
Training Dataset
Validation Dataset
Testing Dataset
TRAIN
VALIDATION
TEST
Train multiple Models
(e.g. Logistic Regression, Decision Trees, KNN)
Validate Models
Tune Hyper parameters and Select the Best Model
(e.g. Logistic Regression)
Evaluate Model
Evaluate the model based on various metrics
(e.g. Confusion Matrix to evaluate the final performance of the selected Logistic Regression Model)

The typical steps to execute model validation here include training the model or commonly multiple models on the training set. The validation set which is a hold-out set from the training set i.e. a portion of training set kept aside is then used to optimize the hyper-parameters of the models and evaluate the model. Thus, the validation set is used to tune the various hyper-parameters and select the best performing algorithm. However, to fully determine that the selected algorithm is correct we apply the model to the training dataset. This is done because as when we tune the hyperparameters based on the validation set, we end up slightly overfitting our model based on the validation set. Thus, the accuracy we receive from the validation set is not considered final and another hold-out dataset which is the test dataset is used to evaluate the final selected model and the error found here is considered as the generalisation error.

Holdout Method is the simplest sort of method to evaluate a classifier. In this method, the data set (a collection of data items or examples) is separated into two sets, called the **Training set and Test set**.
A classifier performs function of assigning data items in a given collection to a target category or class.

Example –

E-mails in our inbox being classified into spam and non-spam. Classifier should be evaluated to find out, it's accuracy, error rate, and error estimates. It can be done using various methods. One of most primitive methods in evaluation of classifier is **'Holdout Method'**.
In the holdout method, data set is partitioned, such that – maximum data belongs to training set and remaining data belongs to test set.

LOOCV (Leave One Out Cross Validation)
In this method, we perform training on the whole data-set but leaves only one data-point of the available data-set and then iterates for each data-point. It has some advantages as well as disadvantages also.
An advantage of using this method is that we make use of all data points and hence it is low bias.
The major drawback of this method is that it leads to higher variation in the testing model as we are testing against one data point. If the data point is an outlier it can lead to higher variation. Another drawback is it takes a lot of execution time as it iterates over 'the number of data points' times.

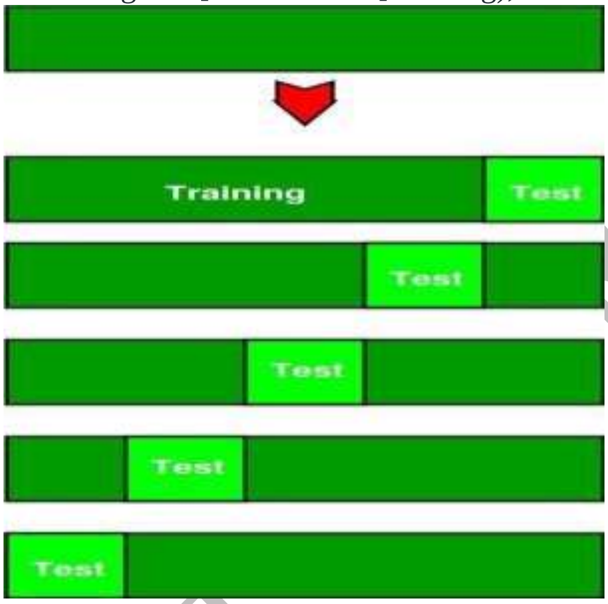
LOOCV(Leave One Out Cross-Validation) is a type of crossvalidation approach in which each observation is considered as the validation set and the rest (N-1) observations are considered as the training set. In LOOCV, fitting of the model is done and predicting using one observation validation set. Furthermore, repeating this for N times for each observation as the validation set. Model is fitted and the model is used to predict a value for observation. This is a special case of **K-fold cross-validation** in which the number of folds is the same as the number of observations(K = N). This method helps to reduce **Bias** and **Randomness.** The method aims at reducing the
Mean-Squared error rate and prevent over fitting. It is very much easy to perform LOOCV in R programming.
Mathematical Expression
LOOCV involves one fold per observation i.e each observation by itself plays the role of the validation set. The (N-1) observations play the role of the training set. With least-squares linear, a single model performance cost is the same as a single model. In LOOCV, refitting of the model can be avoided while implementing the LOOCV method. **MSE(Mean squared error)** is calculated by fitting on the complete dataset.

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - y_i}{1 - h_i}\right)^2$$

In the above formula, ***h***h_i represents how much influence an observation has on its own fit i.e between 0 and 1 that punishes the residual, as it divides by a small number. It inflates the residual.

K fold cross validation

K-Fold Cross Validation
In this method, we split the data-set into k number of subsets(known as folds) then we perform training on the all the subsets but leave one(k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purpose each time.
Note:
It is always suggested that the value of k should be 10 as the lower value of k is takes towards validation and higher value of k leads to LOOCV method.
Example The diagram below shows an example of the training subsets and evaluation subsets generated in k-fold cross-validation. Here, we have total 25 instances. In first iteration we use the first 20 percent of data for evaluation, and the remaining 80 percent for training([1-5] testing and [5-25] training) while in the second iteration we use the second subset of 20 percent for evaluation, and the remaining three subsets of the data for training([5-10] testing and [1-5 and 1025] training), and so on.



Total instances:

25 Value of k :

5

No. Iteration Training set observations Testing set observations

1 [5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24] [0 1 2 3 4]

2 [0 1 2 3 4 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24] [5 6 7 8 9]

3 [0 1 2 3 4 5 6 7 8 9 15 16 17 18 19 20 21 22 23 24] [10 11 12 13 14]

4 [0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 20 21 22 23 24] [15 16 17 18 19]

5 [0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19] [20 21 22 23 24]

Comparison of train/test split to cross-validation

Advantages of train/test split:

1. This runs K times faster than Leave One Out crossvalidation because K-fold cross-validation repeats the train/test split K-times.
2. Simpler to examine the detailed results of the testing process. Advantages of cross-validation:

1. More accurate estimate of out-of-sample accuracy.
2. More “efficient” use of data as every observation is used for both training and testing.

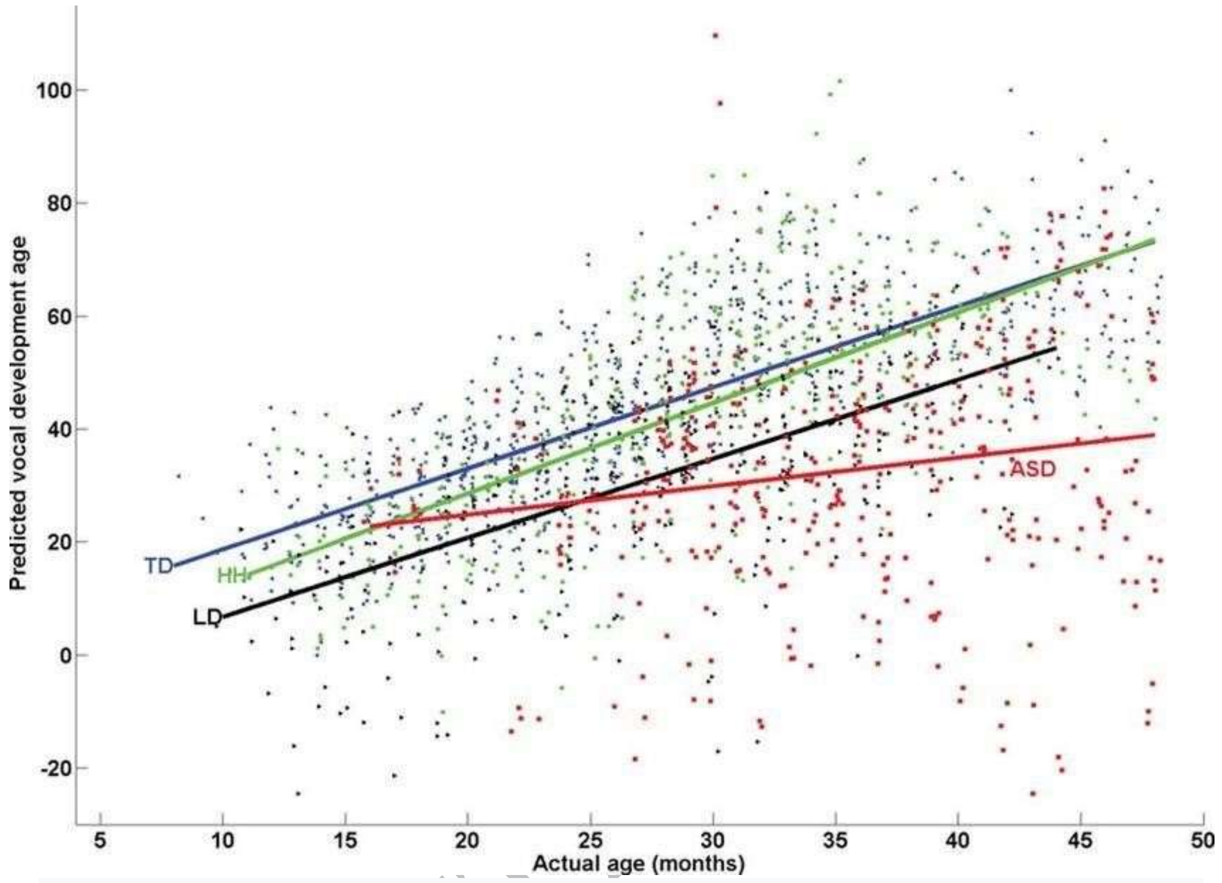
3.

Python code for k fold cross-validation.

```
# This code may not be run on GFG IDE

# as required packages are not found.



# importing cross-validation from sklearn package. from sklearn import

cross_validation



# value of K is 10. data = cross_validation.KFold(len(train_set), n_folds=10,
indices=False)
```

What is Multiple Linear Regression?

Multiple linear regression refers to a statistical technique that is used to predict the outcome of a variable based on the value of two or more variables. It is sometimes known simply as multiple regression, and it is an extension of linear regression. The variable that we want to predict is known as the dependent variable, while the variables we use to predict the value of the dependent variable are known as independent or explanatory variables.



Summary

- Multiple linear regression refers to a statistical technique that uses two or more independent variables to predict the outcome of a dependent variable.
- The technique enables analysts to determine the variation of the model and the relative contribution of each independent variable in the total variance.
- Multiple regression can take two forms, i.e., linear regression and non-linear regression.

Multiple Linear Regression Formula

$$yi = \beta 0 + \beta 1xi1 + \beta 2xi2 + \ldots + \beta pxip + \epsilon$$

Where:

- **yi** is the dependent or predicted variable
- **β0** is the y-intercept, i.e., the value of y when both xi and x2 are 0.
- **β1** and **β2** are the regression coefficients representing the change in y relative to a one-unit change in **xi1** and **xi2**, respectively. • **βp** is the slope coefficient for each independent variable

- ϵ is the model’s random error (residual) term.

Understanding Multiple Linear Regression

Simple linear regression enables statisticians to predict the value of one variable using the available information about another variable. Linear regression attempts to establish the relationship between the two variables along a straight line.

Multiple regression is a type of regression where the dependent variable shows a **linear** relationship with two or more independent variables. It can also be **non-linear**, where the dependent and independent variables do not follow a straight line.

Both linear and non-linear regression track a particular response using two or more variables graphically. However, non-linear regression is usually difficult to execute since it is created from assumptions derived from trial and error.

Assumptions of Multiple Linear Regression

Multiple linear regression is based on the following assumptions:

1. A linear relationship between the dependent and independent variables

The first assumption of multiple linear regression is that there is a linear relationship between the dependent variable and each of the independent variables. The best way to check the linear relationships is to create scatterplots and then visually inspect the scatterplots for linearity. If the relationship displayed in the scatterplot is not linear, then the analyst will need to run a non-linear regression or transform the data using statistical software, such as SPSS.

2. The independent variables are not highly correlated with each other

The data should not show multicollinearity, which occurs when the independent variables (explanatory variables) are highly correlated. When independent variables show multicollinearity, there will be problems figuring out the specific variable that contributes to the variance in the dependent variable. The best method to test for the assumption is the Variance Inflation Factor method.

3. The variance of the residuals is constant

Multiple linear regression assumes that the amount of error in the residuals is similar at each point of the linear model. This scenario is known as homoscedasticity. When analyzing the data, the analyst should plot the standardized residuals against the predicted values to determine if the points are distributed fairly across all the values of independent variables. To test the assumption, the data can be plotted on a scatterplot or by using statistical software to produce a scatterplot that includes the entire model.

4. Independence of observation

The model assumes that the observations should be independent of one another. Simply put, the model assumes that the values of residuals are independent. To test for this assumption, we use the Durbin Watson statistic.

The test will show values from 0 to 4, where a value of 0 to 2 shows positive autocorrelation, and values from 2 to 4 show negative autocorrelation. The mid-point, i.e., a value of 2, shows that there is no autocorrelation.

5. Multivariate normality

Multivariate normality occurs when residuals are normally distributed. To test this assumption, look at how the values of residuals are distributed. It can also be tested using two main methods, i.e., a histogram with a superimposed normal curve or the Normal Probability Plot method.

Some more Assumptions of linear Regression:

- The variables considered for the model should be relevant and the model should be reliable.
- The model should be linear and not non-linear.
- Variables must have a normal distribution
- The variance should be constant for all levels of the predicted variable.

Normal Equation is an analytical approach to Linear Regression with a Least Square Cost Function. We can directly find out the value of θ without using Gradient Descent. Following this approach is an effective and time-saving option when working with a dataset with small features. Normal Equation method is based on the mathematical concept of Maxima & Minima in which the derivative and partial derivative of any function would be zero at the minima and maxima point. So, in Normal Equation method, we get the minimum value of the Cost function by finding its partial derivative w.r.t to each weight and equating it to zero. The normal Equation is as follows:

$$\boxed{\theta = \left(X^T X\right)^{-1} . \left(X^T y\right)}$$

In the above equation, ***θ:*** *hypothesis parameters that define it the best.* ***X:*** *Input feature value of each instance.*
Y: *Output value of each instance.*

Maths Behind the equation:

Given the hypothesis function

$$h(\theta) = \theta_0 x_0 + \theta_1 x_1 + \dots \theta_n x_n$$

where, **n:** the no. of features in the data set. **x_0:** 1 (for vector multiplication) Notice that this is a dot product between θ and x values. So for the convenience to solve we can write it as:

$$h(\theta) = \theta^T x$$

The motive in Linear Regression is to minimize the **cost function**:

J(\Theta) = \frac{1}{2m} \sum_{i = 1}^{m} \frac{1}{2} [h_{\Theta}(x^{(i)}) – y^{(i)}]^{2}

where, **x^i:** the input value of i^{th} training example. **m:** no. of training instances **n:** no. of data-set features **y^i:** the expected result of i^{th} instance

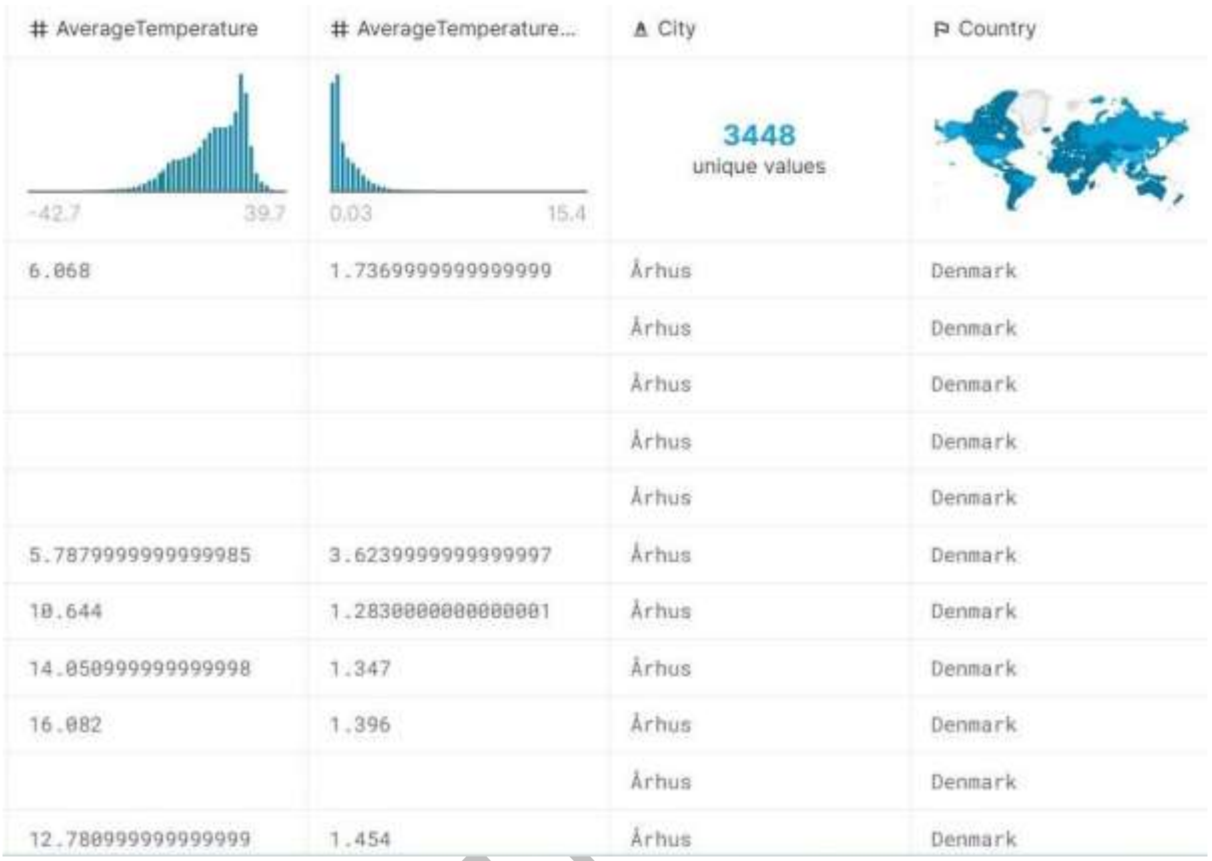
Linear regression datasets for machine learning

- Cancer linear regression. ...
- CDC data: nutrition, physical activity, obesity. ...
- Fish market dataset for regression. ...
- Medical insurance costs. ...
- New York Stock Exchange dataset. ...
- OLS regression challenge. ...
- Real estate price prediction. ...
- Red wine quality.
- Global Temperature and Pollution

Global Temperature and Pollution

Pollution and its impact on the environment are among the most significant concerns globally. Data science and machine learning can help us better understand how to tackle and solve that problem.

You can use multiple datasets to analyze the change in temperature, air pollution, and overall climate throughout the years with linear and other forms of regression. You can find multiple datasets to work with in [this GitHub repository.]



# AverageTemperature	# AverageTemperature...	City	Country
6.068	1.7369999999999999	Århus	Denmark
		Århus	Denmark
		Århus	Denmark
		Århus	Denmark
		Århus	Denmark
5.7879999999999985	3.6239999999999997	Århus	Denmark
10.644	1.2830000000000001	Århus	Denmark
14.050999999999998	1.347	Århus	Denmark
16.082	1.396	Århus	Denmark
		Århus	Denmark
12.780999999999999	1.454	Århus	Denmark

Boston Housing price from sci-kit learn datasets:

Use the above link to
get model:

https://www.kaggle.com/code/jpractice/boston-house-prices-fromsklearndatasets

Regression model to Cricket match Results-past data:

Use the above link to
get model:

https://www.kaggle.com/code/sazid28/cricket-data-analysis-and-scoreprediction

Regression model to check Performance of a cricket player:

Use the above link to
get model:

https://www.kaggle.com/code/aadilmalik94/ipl-winnerpredictorclassification-regression

Regression model to check Crop Yield Prediction:

Use the above link to get model:

https://www.kaggle.com/code/kushagranull/crop-yield-prediction