



# Grid Searching From Scratch using Python

Last Updated : 21 Mar, 2024

Grid searching is a method to find the best possible combination of hyperparameters at which the model achieves the highest accuracy. Before applying Grid Searching on any algorithm, Data is used to be divided into training and validation set, a validation set is used to validate the models. A model with all possible combinations of hyperparameters is tested on the validation set to choose the best combination.

## Implementation:

Grid Searching can be applied to any hyperparameters algorithm whose performance can be improved by tuning hyperparameter. For example, we can apply grid searching on K-Nearest Neighbors by validating its performance on a set of values of K in it. Same thing we can do with Logistic Regression by using a set of values of learning rate to find the best learning rate at which Logistic Regression achieves the best accuracy.

It has 8 features columns like i.e “Age”, “Glucose” e.t.c, and the target variable “Outcome” for 108 patients. So in this, we will train a Logistic Regression Classifier model to predict the presence of diabetes or not for patients with such information.

## Code: Implementation of Grid Searching on Logistic Regression from Scratch

### Python3

```
# Importing libraries
```

```
import numpy as np
```

```
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

Got It !

```

self.learning_rate = learning_rate
self.iterations = iterations

# Function for model training
def fit( self, X, Y ) :
    # no_of_training_examples, no_of_features
    self.m, self.n = X.shape

    # weight initialization
    self.W = np.zeros( self.n )
    self.b = 0
    self.X = X
    self.Y = Y

    # gradient descent learning
    for i in range( self.iterations ) :
        self.update_weights()
    return self

# Helper function to update weights in gradient descent
def update_weights( self ) :
    A = 1 / ( 1 + np.exp( - ( self.X.dot( self.W ) + self.b ) ) )

    # calculate gradients
    tmp = ( A - self.Y.T )
    tmp = np.reshape( tmp, self.m )
    dW = np.dot( self.X.T, tmp ) / self.m
    db = np.sum( tmp ) / self.m

    # update weights
    self.W = self.W - self.learning_rate * dW
    self.b = self.b - self.learning_rate * db
    return self

# Hypothetical function h( x )
def predict( self, X ) :
    Z = 1 / ( 1 + np.exp( - ( X.dot( self.W ) + self.b ) ) )
    Y = np.where( Z > 0.5, 1, 0 )
    return Y

# Driver code

def main() :

    # Importing dataset
    df = pd.read_csv( "diabetes.csv" )
    X = df.iloc[:, :-1].values
    Y = df.iloc[:, -1:].values

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

```

# Model training
max_accuracy = 0

# learning_rate choices
learning_rates = [ 0.1, 0.2, 0.3, 0.4, 0.5,
                   0.01, 0.02, 0.03, 0.04, 0.05 ]

# iterations choices
iterations = [ 100, 200, 300, 400, 500 ]

# available combination of learning_rate and iterations

parameters = []
for i in learning_rates :
    for j in iterations :
        parameters.append( ( i, j ) )

print("Available combinations : ", parameters )

# Applying linear searching in list of available combination
# to achieved maximum accuracy on CV set

for k in range( len( parameters ) ) :
    model = LogisticRegression( learning_rate = parameters[k][0],
                               iterations = parameters[k][1] )

    model.fit( X_train, Y_train )

    # Prediction on validation set
    Y_pred = model.predict( X_valid )

    # measure performance on validation set

    correctly_classified = 0

    # counter
    count = 0

    for count in range( np.size( Y_pred ) ) :
        if Y_valid[count] == Y_pred[count] :
            correctly_classified = correctly_classified + 1

    curr_accuracy = ( correctly_classified / count ) * 100

    if max_accuracy < curr_accuracy :
        max_accuracy = curr_accuracy

print( "Maximum accuracy achieved by our model through grid searching : ",

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

## Output:

```
Available combinations : [(0.1, 100), (0.1, 200), (0.1, 300), (0.1,
400),
(0.1, 500), (0.2, 100), (0.2, 200), (0.2, 300), (0.2, 400), (0.2,
500),
(0.3, 100), (0.3, 200), (0.3, 300), (0.3, 400), (0.3, 500), (0.4,
100),
(0.4, 200), (0.4, 300), (0.4, 400), (0.4, 500), (0.5, 100), (0.5,
200),
(0.5, 300), (0.5, 400), (0.5, 500), (0.01, 100), (0.01, 200), (0.01,
300),
(0.01, 400), (0.01, 500), (0.02, 100), (0.02, 200), (0.02, 300),
(0.02, 400),
(0.02, 500), (0.03, 100), (0.03, 200), (0.03, 300), (0.03, 400),
(0.03, 500),
(0.04, 100), (0.04, 200), (0.04, 300), (0.04, 400), (0.04, 500),
(0.05, 100),
(0.05, 200), (0.05, 300), (0.05, 400), (0.05, 500)]
```

Maximum accuracy achieved by our model through grid searching : 60.0

In the above, we applied grid searching on all possible combinations of learning rates and the number of iterations to find the peak of the model at which it achieves the highest accuracy.

## Code: Implementation of Grid Searching on Logistic Regression of sklearn

### Python3

```
# Importing Libraries
```

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
```

```
# Driver Code
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

```

X = df.iloc[:, :-1].values
Y = df.iloc[:, -1:].values

# Splitting dataset into train and test set
X_train, X_test, Y_train, Y_test = train_test_split(
    X, Y, test_size = 1/3, random_state = 0 )

# Model training
max_accuracy = 0

# grid searching for learning rate
parameters = { 'C' : [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ] }

model = LogisticRegression()
grid = GridSearchCV( model, parameters )
grid.fit( X_train, Y_train )

# Prediction on test set
Y_pred = grid.predict( X_test )

# measure performance
correctly_classified = 0

# counter
count = 0

for count in range( np.size( Y_pred ) ) :
    if Y_test[count] == Y_pred[count] :
        correctly_classified = correctly_classified + 1

accuracy = ( correctly_classified / count ) * 100

print( "Maximum accuracy achieved by sklearn model through grid searching :

if __name__ == "__main__" :
    main()

```

## Output:

Maximum accuracy achieved by sklearn model through grid searching :  
62.86

**Note:** Grid Searching plays a vital role in tuning hyperparameters for the mathematically complex models.

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

[Comment](#)[More info](#)[Advertise with us](#)

## Next Article

[PyTorch v/s Tensorflow](#)

## Similar Reads

### Grid Plot in Python using Seaborn

Grids are general types of plots that allow you to map plot types to grid rows and columns, which helps you to create similar character-separated plots. In this article, we will be using two different data sets (Iri...

4 min read

---

### Python Code Generation Using Transformers

Python's code generation capabilities streamline development, empowering developers to focus on high-level logic. This approach enhances productivity, creativity, and innovation by automating intricate code...

3 min read

---

### Python - seaborn.PairGrid() method

Prerequisite: Seaborn Programming Basics Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics....

3 min read

---

### Create a correlation Matrix using Python

In the field of data science and machine learning, a correlation matrix aids in understanding relationships between variables. Correlation matrix represents how different variables interact with each other. For...

8 min read

---

### Python - seaborn.FacetGrid() method

Prerequisite: Seaborn Programming Basics Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics....

3 min read

---

### Dynamic Visualization using Python

Data visualization in Python refers to the pictorial representation of raw data for better visualization, understanding, and inference. Python provides various libraries containing different features for visualizin...

11 min read

---

### Implementation of neural network from scratch using NumPy

DNN(Deep neural network) in a machine learning algorithm that is inspired by the way the human brain

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

[Policy](#)

## Python - Data visualization using Bokeh

Bokeh is a data visualization library in Python that provides high-performance interactive charts and plots. Bokeh output can be obtained in various mediums like notebook, html and server. It is possible to embed...

3 min read

## Top 50 + Python Interview Questions for Data Science

Python is a popular programming language for Data Science, whether you are preparing for an interview for a data science role or looking to brush up on Python concepts. In this article, we will cover various Top...

15+ min read

## Data Visualization with Python

In today's world, a lot of data is being generated on a daily basis. And sometimes to analyze this data for certain trends, patterns may become difficult if the data is in its raw format. To overcome this data...

14 min read



### Corporate & Communications Address:

A-143, 7th Floor, Sovereign Corporate  
Tower, Sector- 136, Noida, Uttar Pradesh  
(201305)

### Registered Address:

K 061, Tower K, Gulshan Vivante  
Apartment, Sector 137, Noida, Gautam  
Buddh Nagar, Uttar Pradesh, 201305



Advertise with us

### Company

About Us  
Legal  
Privacy Policy  
In Media  
Contact Us

### Languages

Python  
Java  
C++  
PHP  
Go lang

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

## DSA

Data Structures  
Algorithms  
DSA for Beginners  
Basic DSA Problems  
DSA Roadmap  
Top 100 DSA Interview Problems  
DSA Roadmap by Sandeep Jain  
All Cheat Sheets

## Web Technologies

HTML  
CSS  
JavaScript  
TypeScript  
ReactJS  
NextJS  
Bootstrap  
Web Design

## Computer Science

Operating Systems  
Computer Network  
Database Management System  
Software Engineering  
Digital Logic Design  
Engineering Maths  
Software Development  
Software Testing

## System Design

High Level Design  
Low Level Design  
UML Diagrams  
Interview Guide  
Design Patterns  
OOAD  
System Design Bootcamp  
Interview Questions

## School Subjects

Mathematics  
Physics  
Chemistry  
Biology

## Data Science & ML

Data Science With Python  
Data Science For Beginner  
Machine Learning  
ML Maths  
Data Visualisation  
Pandas  
NumPy  
NLP  
Deep Learning

## Python Tutorial

Python Programming Examples  
Python Projects  
Python Tkinter  
Web Scraping  
OpenCV Tutorial  
Python Interview Question  
Django

## DevOps

Git  
Linux  
AWS  
Docker  
Kubernetes  
Azure  
GCP  
DevOps Roadmap

## Interview Preparation

Competitive Programming  
Top DS or Algo for CP  
Company-Wise Recruitment Process  
Company-Wise Preparation  
Aptitude Preparation  
Puzzles

## GeeksforGeeks Videos

DSA  
Python  
Java  
C++

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).



@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved



---

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).