Introduction to Naive Bayes Classification and Its Implementation in Python

What is Naive Bayes Classification? How to implement it in Python?

Tenisha D · Follow
Published in Python in Plain English
4 min read · Nov 12, 2021

Photo by Naser Tamimi on Unsplash

Naive Bayes Classification

The naive Bayes classification algorithm is one of the popularly used **Supervised machine learning** algorithms for classification tasks. It is based on the **Bayes theorem** in probability.

Naive Bayes Classifier is called naive because it considers every input variable as an independent event (Where the outcome of one event does not affect the outcome of other events **Example:** the outcome of rolling the first die does not affect the outcome of rolling the next die). To understand the Naive Bayes algorithm, we need to be familiar with the Bayes theorem.

Applications of Naive Bayes Classification Algorithm

Following are some of the largely used applications of naive Bayes classification Algorithm,

1. Text Classification
2. Multiclass prediction
3. Spam Filteration
4. Recommendation System
5. Sentiment Analysis

Bayes Theorem

Bayes Theorem (or) Bayes law (or) Bayes rule describes the conditional probability of an event, based on prior knowledge of conditions that might be related to the event. Bayes theorem is widely used in machine learning because of its effective way to predict classes with precision and accuracy.

Bayes theorem is mathematically stated as,

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Bayes theorem formula

where,

P(A|B) = Probability of A given that event B (or) Posterior probability of A given B.

P(B|A) = Probability of B given that event A (or) the likelihood of A given B.

P(A) & P(B) = Probability of A and B (or) Prior Probability

Example:

Let's consider an example, where even if 100% of patients with pancreatic cancer have a certain symptom when someone has the same symptom, it does not mean that this person has a 100% chance of getting pancreatic cancer.

Assume the incidence rate of pancreatic cancer is 1/100000, while 10/100000 healthy individuals have the same symptoms worldwide, the probability of having pancreatic cancer given the symptoms is only 9.1%, and the other 90.9% could be “false positives” (that is, falsely said to have cancer; “positive” is a confusing term when, as here, the test gives bad news).

Based on the incidence rate, the following table presents the corresponding numbers per 100,000 people.

Symptom \ Cancer	Yes	No	Total
Yes	1	10	11
No	0	99989	99989
Total	1	99999	100000

Pancreatic cancer & symptom result number per 1,00,00 people

Probability of having cancer when you have the symptoms:

$$
\begin{aligned}
P(\text{Cancer}|\text{Symptoms}) &= \frac{P(\text{Symptoms}|\text{Cancer})P(\text{Cancer})}{P(\text{Symptoms})} \\
&= \frac{P(\text{Symptoms}|\text{Cancer})P(\text{Cancer})}{P(\text{Symptoms}|\text{Cancer})P(\text{Cancer}) + P(\text{Symptoms}|\text{Non-Cancer})P(\text{Non-Cancer})} \\
&= \frac{1 \times 0.00001}{1 \times 0.00001 + (10/99999) \times 0.99999} = \frac{1}{11} \approx 9.1\%
\end{aligned}
$$

Implementation in Python

Let's implement Naive Bayes Classification in Python.

We are using the Advertisement clicking dataset (about users clicking the ads or not)

The dataset has the following features,

1. Daily Time Spent on Site — Amount of time spent on the website
2. Age — User's Age
3. Area Income — Avg revenue of the Users
4. Daily Internet Usage — Avg usage of internet daily
5. Ad Topic Line — Topic text of the advertisement
6. City — City of the Users

7. Male — gender of the users(male or female)
8. Country — Country of the users
9. Timestamp — Time clicked on the Ad
10. Clicked on Ad — 0 or 1, 0-not clicked,1-clicked.

```
#importing libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import sklearn
```

Let's load the dataset

```
#loading the dataset
data =  pd.read_csv('/content/drive/MyDrive/ML_datasets
/advertising.csv')

#head of the dataset
data.head()
```

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0

```
#describing the data
```

```
data.describe()
```

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Male	Clicked on Ad
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000
mean	65.000200	36.009000	55000.000080	180.000100	0.481000	0.50000
std	15.853615	8.785562	13414.634022	43.902339	0.499889	0.50025
min	32.600000	19.000000	13996.500000	104.780000	0.000000	0.00000
25%	51.360000	29.000000	47031.802500	138.830000	0.000000	0.00000
50%	68.215000	35.000000	57012.300000	183.130000	0.000000	0.50000
75%	78.547500	42.000000	65470.635000	218.792500	1.000000	1.00000
max	91.430000	61.000000	79484.800000	269.960000	1.000000	1.00000

Next, we can drop a few columns for a better Naive Bayes model.

```
#drop 'Ad Line Topic' ,'City', 'Country' and Timestamp.

data.drop(['Ad Topic Line','City','Country','Timestamp'],axis =
1,inplace = True)

data.head()
```

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Male	Clicked on Ad
0	68.95	35	61833.90	256.09	0	0
1	80.23	31	68441.85	193.77	1	0
2	69.47	26	59785.94	236.50	0	0
3	74.15	29	54806.18	245.89	1	0
4	68.37	35	73889.99	225.58	0	0

Next, we can then split the dataset into train and test and the test size is fixed as 1/3 or 0.33.

```
#train test split the data
```

```
X = data.iloc[:,0:4].values

Y = data['Clicked on Ad'].values

from sklearn.model_selection import train_test_split

X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size =
1/3,random_state = 0)

print('X_train shape:',X_train.shape)

print('X_test shape:',X_test.shape)

print('Y_train shape:',Y_train.shape)

print('Y_test shape:',X_test.shape)
```

Next, we perform feature scaling for X_train and X_test

```
#feature scaling

from sklearn.preprocessing import StandardScaler

stdscaler = StandardScaler()

X_train = stdscaler.fit_transform(X_train)

X_test = stdscaler.transform(X_test)
```

Let's train the Naive Bayes model on the training data

```
#naive bayes model

from sklearn.naive_bayes import GaussianNB

clf = GaussianNB()

clf.fit(X_train,Y_train
```

Now we can predict the results

```
#predicting the result
```

```
y_pred = clf.predict(X_test)
print(y_pred)
```

Lastly, we can plot a confusion matrix for a better understanding of the model

```
#confusion matrix
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(Y_test, y_pred)
print(cm)
#accuracy score of the model
print('Accuracy score :',accuracy_score(Y_test,y_pred))
```

```
#confusion matrix
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(Y_test, y_pred)
print(cm)

#accuracy score of the model
print('Accuracy score :',accuracy_score(Y_test,y_pred))
```

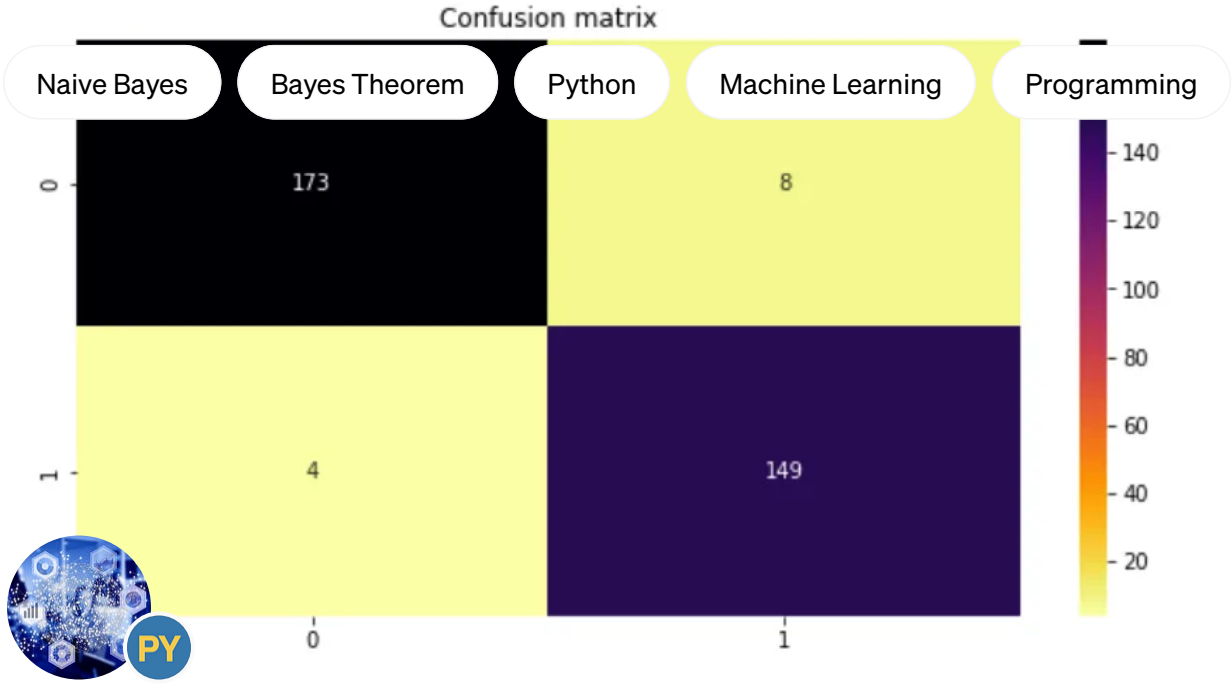
```
[[173   8]
 [  4 149]]
Accuracy score : 0.9640718562874252
```

Let's plot the confusion matrix

```
#plotting the confusion matrix
plt.figure(figsize=(10,5))
plt.title('Confusion matrix')
sns.heatmap(cm,annot=True,fmt='d',cmap='inferno_r')
```

```
plt.figure(figsize=(10,5))
plt.title('Confusion matrix')
sns.heatmap(cm,annot=True,fmt='d',cmap='inferno_r')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f19695eb5d0>
```



Written by Tenisha D

The accuracy of the model for this dataset is 0.96.

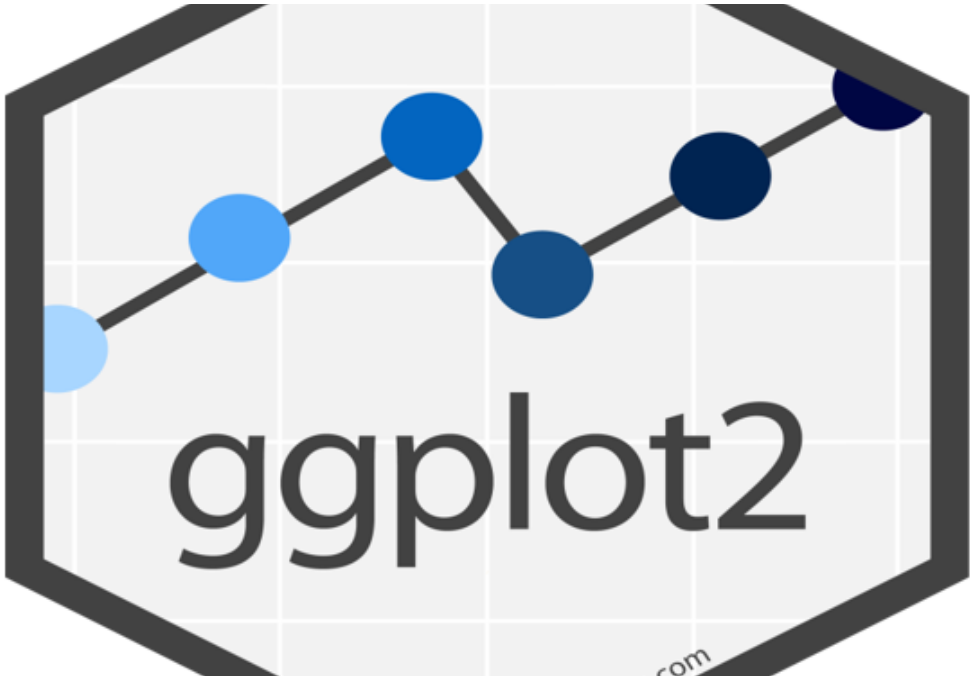
142 Followers · Writer for Python in Plain English

Data Enthusiast | Aspiring Data Scientist |

The Naive Bayes Classification model was performed well for the advertisement dataset.

More content at plainenglish.io

More from Tenisha D and Python in Plain English

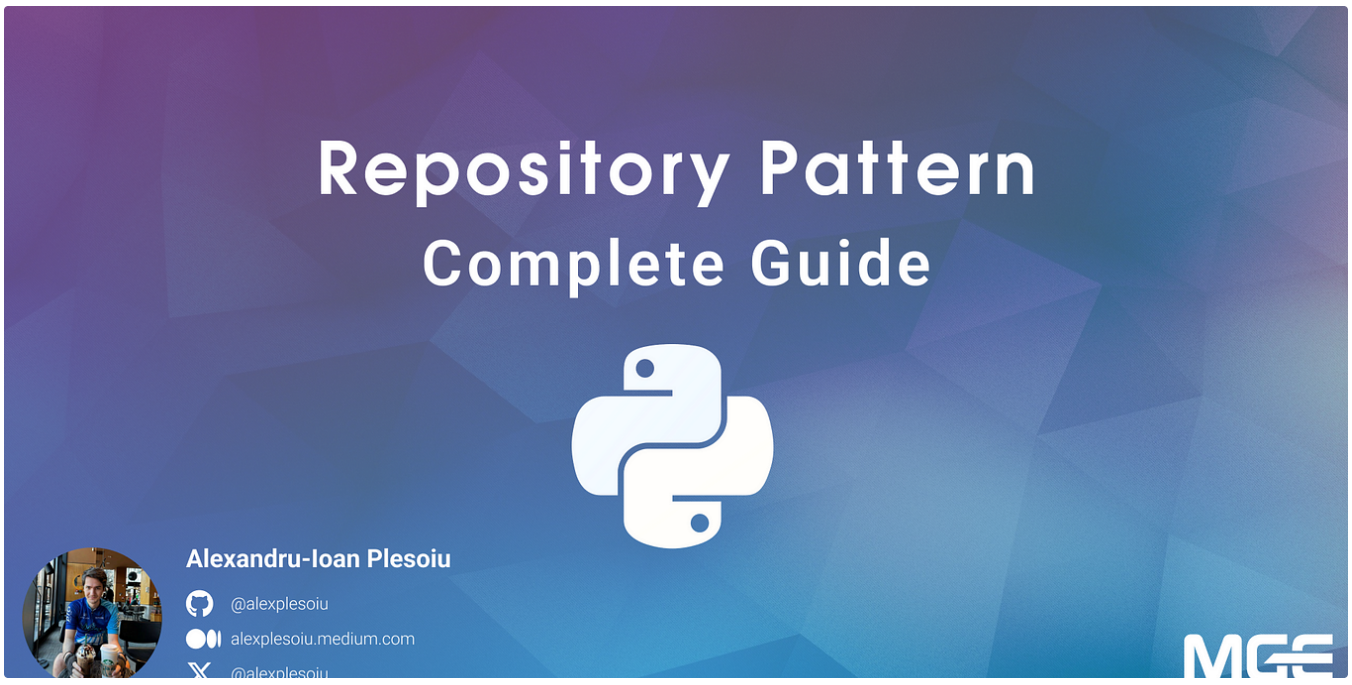


Tenisha D in Python in Plain English

How to Use the ggplot in Python for Visualization?

A guide on using ggplot for data visualization.

3 min read · Feb 6, 2022



Alexandru-Ioan Plesoiu in Python in Plain English

Repository Pattern is INSANE if you know how to use it properly—Python

Repository Pattern is one of the most-known design patterns in the software development industry. The main ideea of this pattern is to ...

· 15 min read · Aug 21

707 6

Delight Olu-Olagbuji in Python in Plain English

Python Optimization Guide: Make Your Code Run 5X Faster

Code optimization involves making your code run faster, use less resources and execute more smoothly, hence increasing its performance

7 min read · Aug 20

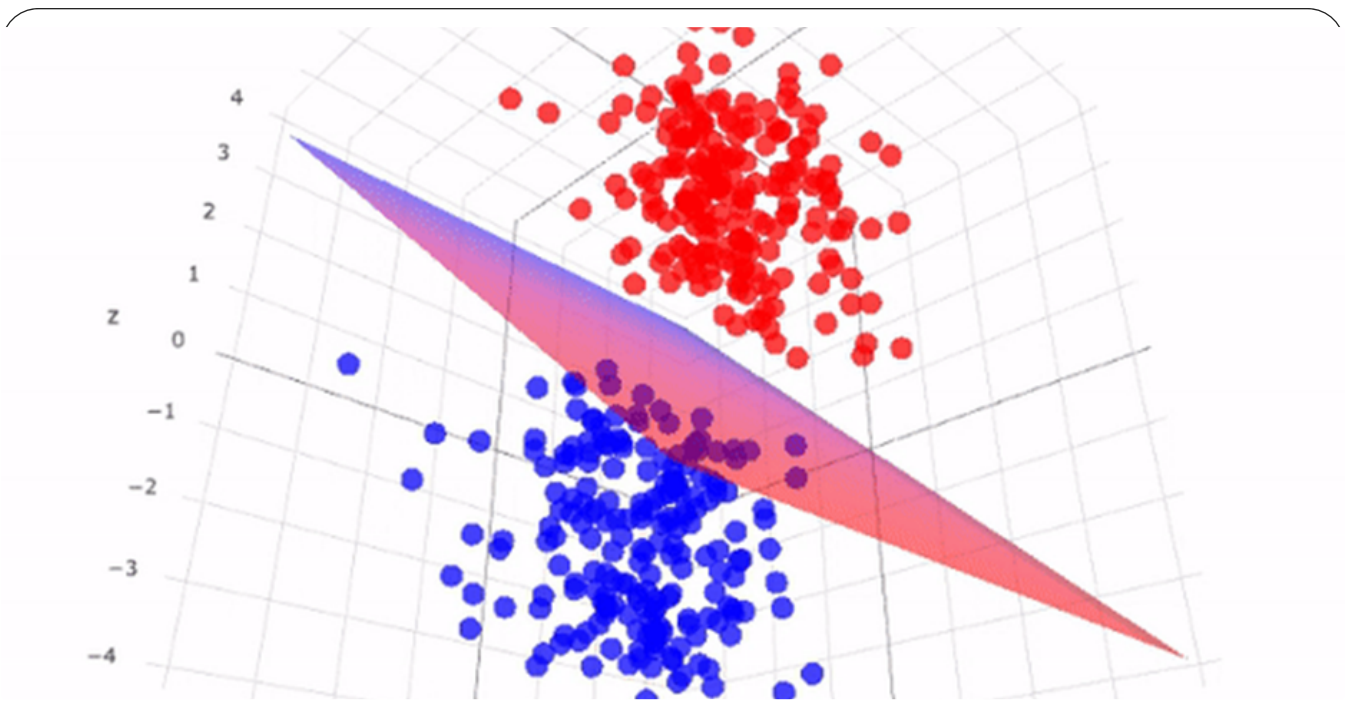
236 5

Tenisha D in Python in Plain English

An Introduction to Classification And Regression Trees (CART) in Python

Part I: Understanding the Classification and Regression Trees (CART) algorithm in Python.

· 7 min read · Nov 20, 2021



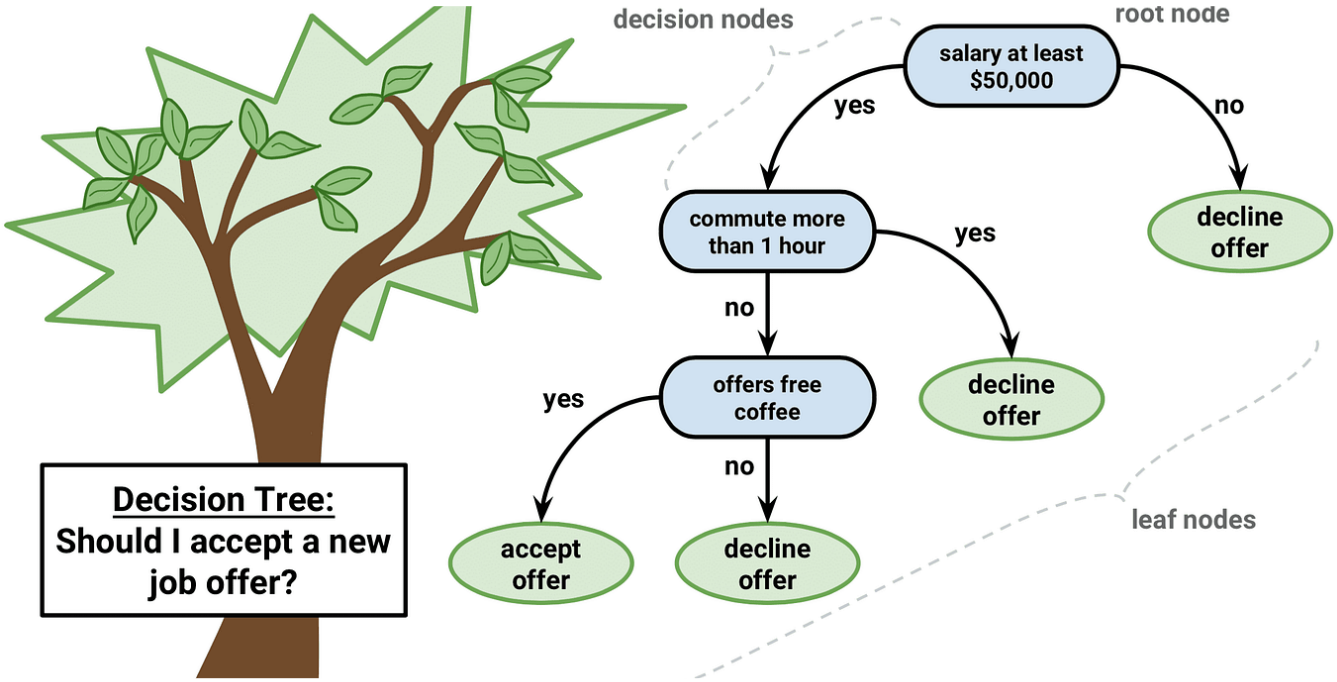
Tasmay Pankaj Tibrewal in Low Code for Data Science

Support Vector Machines (SVM): An Intuitive Explanation

Everything you always wanted to know about this powerful supervised ML algorithm

17 min read · Jul 2

612 2



Abdul4code in GoPenAI

Decision Tree Algorithm

Demystifying Decision Tree Classifier: Unveiling the Inner Workings and Python Implementation

13 min read · May 16

28

Lists

Predictive Modeling w/ Python

20 stories · 391 saves

Practical Guides to Machine Learning

10 stories · 436 saves

Coding & Development

11 stories · 173 saves

It's never too late or early to start something

15 stories · 126 saves

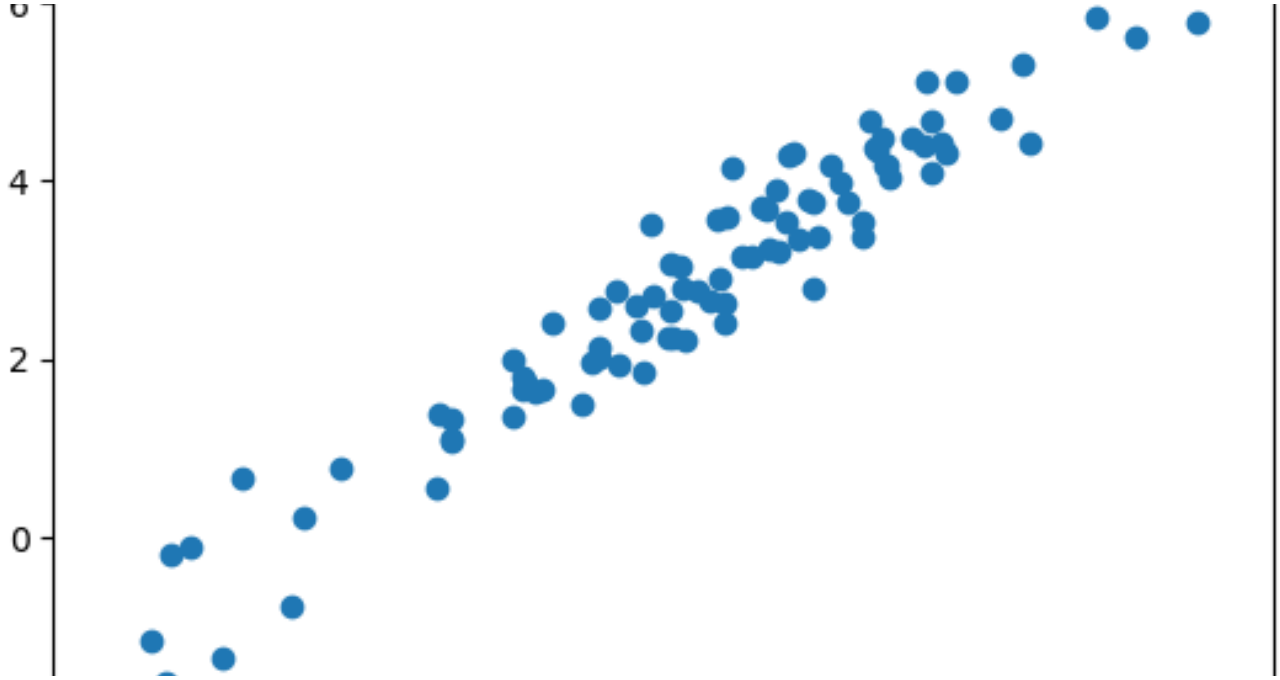
Piero Paialunga in Towards Data Science

From Theory to Practice with Bayesian Neural Network, Using Python

Here's how to incorporate uncertainty in your Neural Networks, using a few lines of code

11 min read · Dec 21, 2022

356 11



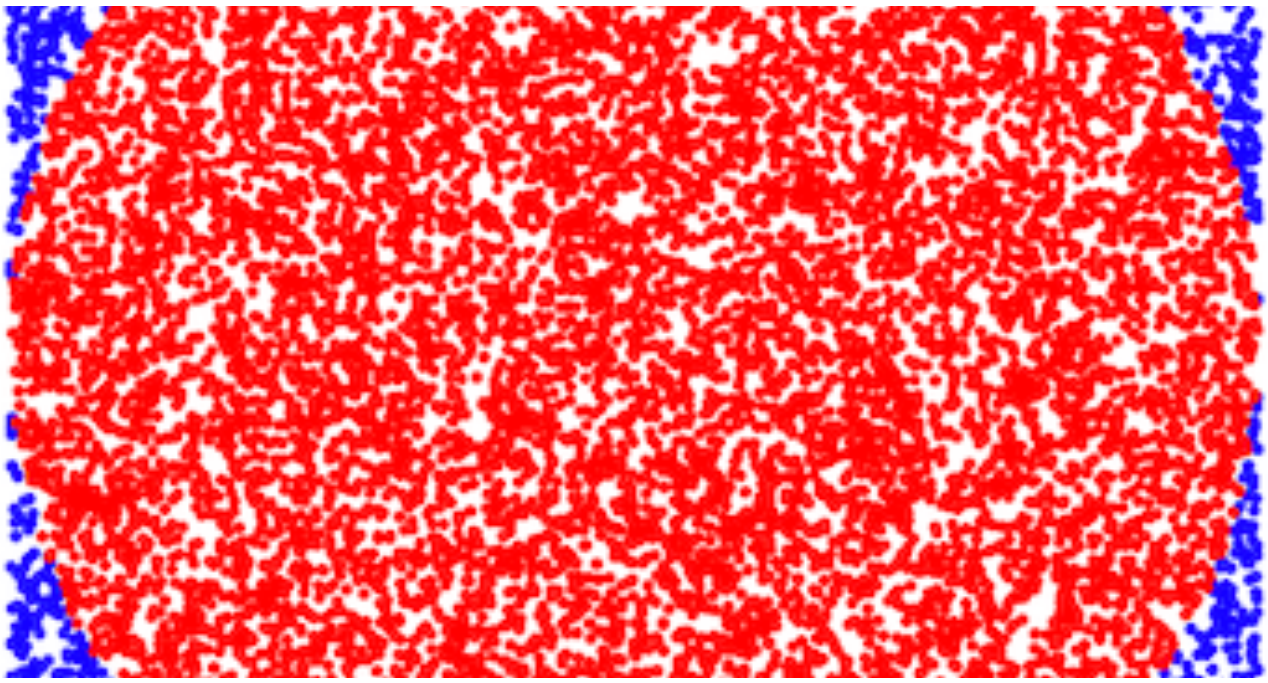
Herman Van Haagen

Backpropagation algorithm part 4

Linear regression

3 min read · Jun 21


6



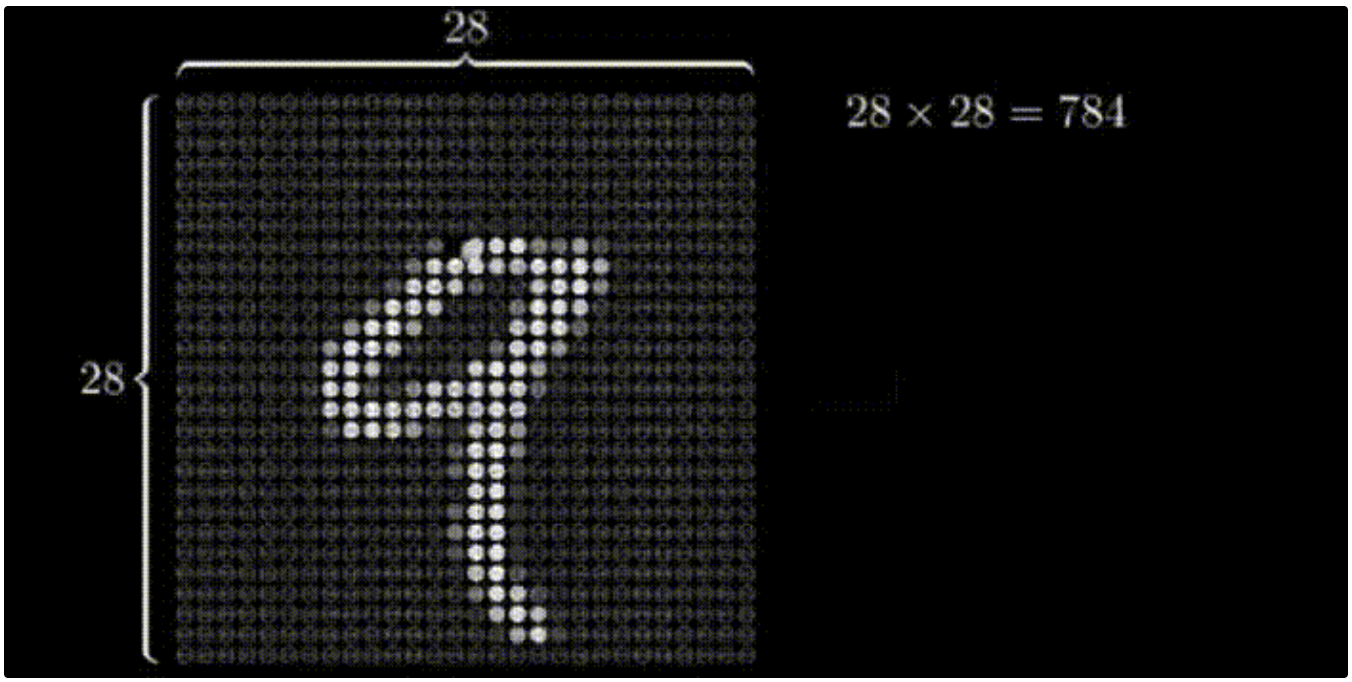
Wendy Hu

Monte Carlo Simulation with Python

Introduction

5 min read · Apr 5

47



Sadaf Saleem

Neural Networks in 10mins. Simply Explained!

What are Neural Networks?

9 min read · May 15

91 1

See more recommendations