

# Representing Neural Networks: Takeaways



by Dataquest Labs, Inc. - All rights reserved © 2020

## Syntax

- Generating data with specific properties using scikit learn:
  - [sklearn.datasets.make\\_regression\(\)](#)
  - [sklearn.datasets.make\\_classification\(\)](#)
  - [sklearn.datasets.make\\_moons\(\)](#)
- Generating a regression data set with 3 features, 1000 observations, and a random seed of 1:

```
from sklearn.datasets import make_regression  
  
data = make_regression(n_samples=1000, n_features=3, random_state=1)
```

- Returning a tuple of two NumPy objects that contain the generated data:

```
print(type(data))  
  
tuple
```

- Retrieving the features of the generated data:

```
print(data[0])  
  
array([[ 0.93514778,  1.81252782,  0.14010988],  
       [-3.06414136,  0.11537031,  0.31742716],  
       [-0.42914228,  1.20845633,  1.1157018 ],  
       ...,  
       [-0.42109689,  1.01057371,  0.20722995],  
       [ 2.18697965,  0.44136444, -0.10015523],  
       [ 0.440956  ,  0.32948997, -0.29257894]])
```

- Retrieving the first row of data:

```
print(data[0][0])  
  
array([ 0.93514778,  1.81252782,  0.14010988])
```

- Retrieving the labels of the data:

```
print(data[1])  
  
array([ 2.55521349e+02, -2.24416730e+02,  1.77695808e+02,  
        1.78288470e+02, -6.31736749e+01, -5.52369226e+01,  
        2.33255554e+01, -8.81410996e+01, -1.75571964e+02,  
        1.06048917e+01,  7.07568627e+01,  2.86371625e+02,  
        ...,  
        7.38320267e+01, -2.38437890e+02, -1.23449719e+02,  
        3.36130733e+01, -2.67823475e+02,  1.21279169e+00,  
        2.62440408e+02,  1.32486453e+02, -1.93414037e+02,  
        2.75702376e+01, -1.00678877e+01,  2.05169507e+02,  
        1.52978767e+02,  1.18361239e+01, -2.97505169e+02,  
        2.40169605e+02,  7.33158364e+01,  2.18888903e+02,  
        3.92751308e+01])
```

- Retrieving the first label of the data:

```
print(data[1][0])  
  
255.52134901495128
```

- Creating a dataframe:

```
features = pd.DataFrame(data[0])
```

## Concepts

- Neural networks are usually represented as **graphs**. A graph is a data structure that consists of nodes (represented as circles) that are connected by edges (represented as lines between the nodes).
- Graphs are a highly flexible data structure; you can even represent a list of values as a graph. Graphs are often categorized by their properties, which act as constraints. You can read about the many different ways graphs can be categorized [on Wikipedia](#).
- Neural network models are represented as a **computational graph**. A computational graph uses nodes to describe variables and edges to describe how variables are combined.

- In a simple neural network:
  - each feature column in a data set is represented as an **input neuron**
  - each weight value is represented as an arrow from the feature column it multiplies to the **output neuron**
- Inspired by biological neural networks, an **activation function** determines if the neuron *fires* or not. In a neural network model, the activation function transforms the weighted sum of the input values.

## Resources

- [Graph Theory on Wikipedia](#)
- [Directed Acyclic Graph on Wikipedia](#)
- [Feedforward Neural Network on Wikipedia](#)
- [Calculus on Computational Graphs](#)
  - Explores how computational graphs can be used to organize derivatives.



Takeaways by Dataquest Labs, Inc. - All rights reserved © 2020