King Abdulaziz University (KAU)
Faculty of Engineering – Girls' Campus
Electrical Engineering Department
EE-495 – Special Topics In Computer Engineering

EE-495

Special Topics In Computer Engineering

Case Study No. 3

How to Generate Test Datasets in Python with scikit-learn

Haneen Alamoudi - 1708436

Submission Date: 6 April 2021 (WEEK 12)

Table of Contents

Introduction

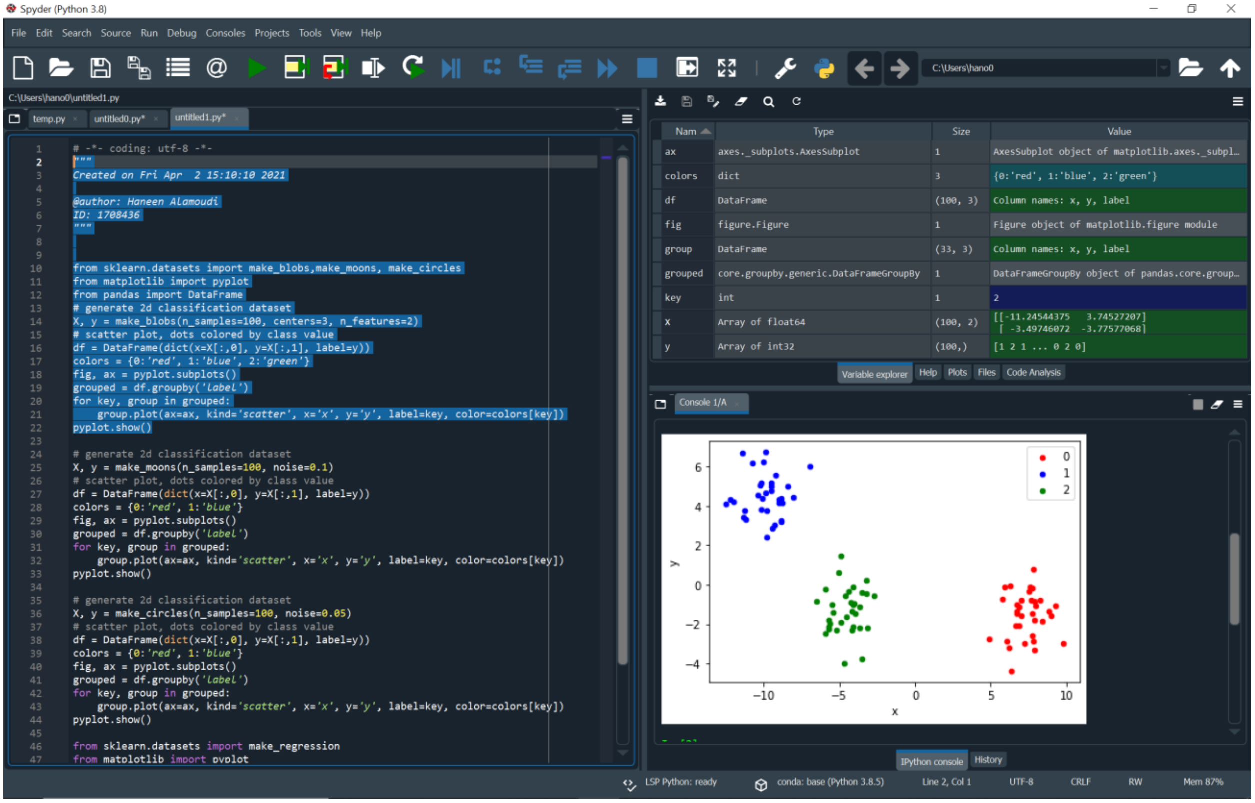
The first step in testing a new machine learning algorithm is to generate test data. Collecting data can be a tedious task, and often the best (and easiest) solution will be to use generated data rather than collecting it. Often, researcher simply want to compare different machine learning algorithms and you don’t care about the origin of the data. The Python library, scikit-learn (sklearn), allows one to create test datasets fit for many different machine learning test problems. Sci-kit learn is a popular library that contains a wide-range of machine-learning algorithms and can be used for data mining and data analysis. This case study is divided into four problems, first blobs classification problem is discussed. Then, moon classification problem, circle classification problem and finally regression test problem is presented.

Blobs Classification Problem

Here, Gaussian distribution with the make_blobs() function can be used to generate blobs of points. In the function you can specify some properties including how many blobs to generate and the number of samples to generate. The problem is suitable for linear classification problems given the linearly separable nature of the blobs.

The figure below shows a 2D dataset of samples with three blobs as a multi-class classification prediction problem.

Output



```
# -*- coding: utf-8 -*-
"""
Created on Fri Apr  2 15:10:10 2021

@author: Haneen Alamoudi
ID: 1708436
"""


from sklearn.datasets import make_blobs,make_moons, make_circles
from matplotlib import pyplot
from pandas import DataFrame
# generate 2d classification dataset
X, y = make_blobs(n_samples=100, centers=3, n_features=2)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue', 2:'green'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_moons(n_samples=100, noise=0.1)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_circles(n_samples=100, noise=0.05)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

from sklearn.datasets import make_regression
from matplotlib import pyplot
```

Nam	Type	Size	Value
ax	axes._subplots.AxesSubplot	1	AxesSubplot object of matplotlib.axes._subpl…
colors	dict	3	{0:'red', 1:'blue', 2:'green'}
df	DataFrame	(100, 3)	Column names: x, y, label
fig	figure.Figure	1	Figure object of matplotlib.figure module
group	DataFrame	(33, 3)	Column names: x, y, label
grouped	core.groupby.generic.DataFrameGroupBy	1	DataFrameGroupBy object of pandas.core.group…
key	int	1	2
x	Array of float64	(100, 2)	[[-11.24544375 3.74527207] [-3.49746072 -3.77577068]
y	Array of int32	(100,)	[1 2 1 ... 0 2 0]

Code

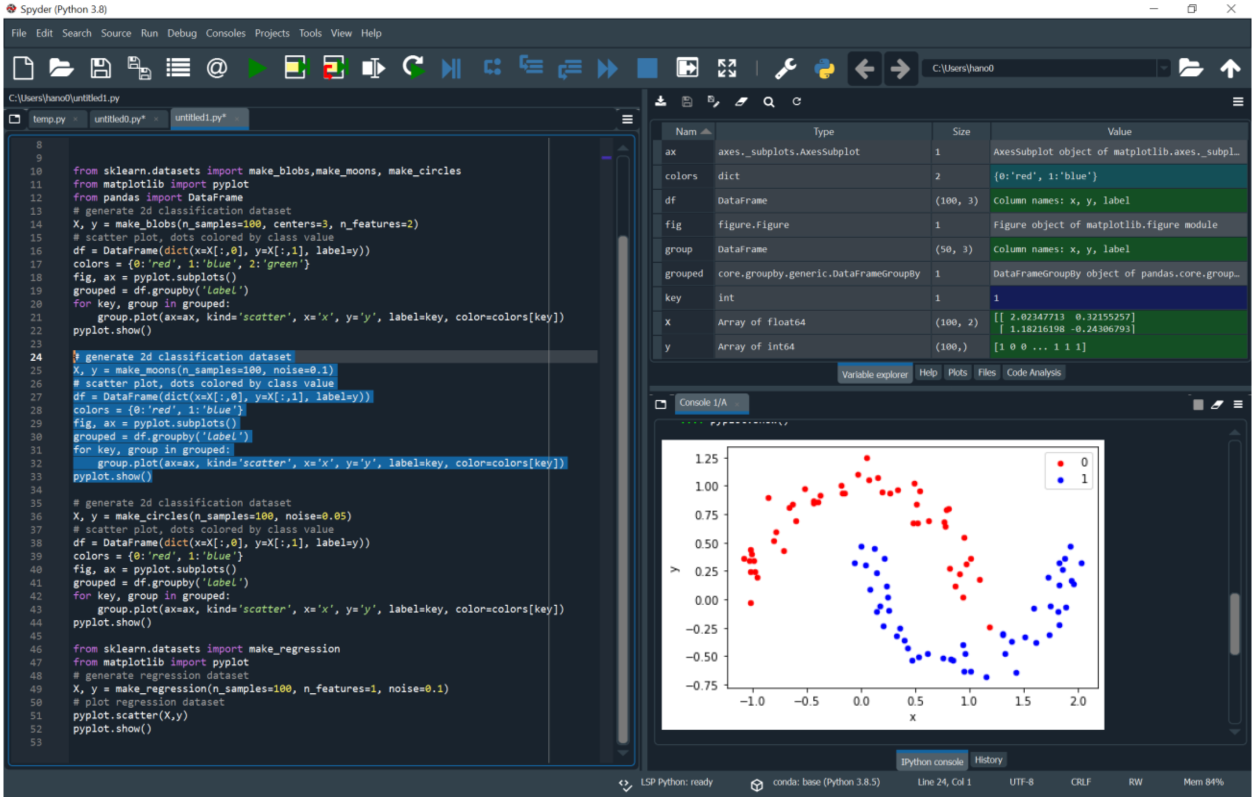
```
"""
Created on Fri Apr  2 15:10:10 2021
@author: Haneen Alamoudi
ID: 1708436
"""
from sklearn.datasets import make_blobs,make_moons, make_circles
from matplotlib import pyplot
from pandas import DataFrame
# generate 2d classification dataset
X, y = make_blobs(n_samples=100, centers=3, n_features=2)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue', 2:'green'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()
```

Moons Classification Problem

Here, the make_moons() function can be used for binary classification and will generate a swirl pattern, or two moons. In the function you can specify some properties including how noisy the moon shapes are and the number of samples to generate. The problem is suitable for algorithms that are capable of learning nonlinear class boundaries.

The figure below shows a 100 sample size moon dataset with 0.1 noise.

Output



Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hano0
C:\Users\hano0\untitled1.py
temp.py
untitled0.py*
untitled1.py*

```
from sklearn.datasets import make_blobs,make_moons, make_circles
from matplotlib import pyplot
from pandas import DataFrame
# generate 2d classification dataset
X, y = make_blobs(n_samples=100, centers=3, n_features=2)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue', 2:'green'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_moons(n_samples=100, noise=0.1)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_circles(n_samples=100, noise=0.05)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

from sklearn.datasets import make_regression
from matplotlib import pyplot
# generate regression dataset
X, y = make_regression(n_samples=100, n_features=1, noise=0.1)
# plot regression dataset
pyplot.scatter(X,y)
pyplot.show()
```

Nam	Type	Size	Value
ax	axes._subplots.AxesSubplot	1	AxesSubplot object of matplotlib.axes._subpl…
colors	dict	2	{0:'red', 1:'blue'}
df	DataFrame	(100, 3)	Column names: x, y, label
fig	figure.Figure	1	Figure object of matplotlib.figure module
group	DataFrame	(50, 3)	Column names: x, y, label
grouped	core.groupby.generic.DataFrameGroupBy	1	DataFrameGroupBy object of pandas.core.group…
key	int	1	1
x	Array of float64	(100, 2)	[[2.02347713 0.32155257] [1.18216198 -0.24306793]
y	Array of int64	(100,)	[1 0 0 ... 1 1 1]

Code

```
"""
Created on Fri Apr  2 15:10:10 2021
@author: Haneen Alamoudi
ID: 1708436
"""

# generate 2d classification dataset
X, y = make_moons(n_samples=100, noise=0.1)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
```

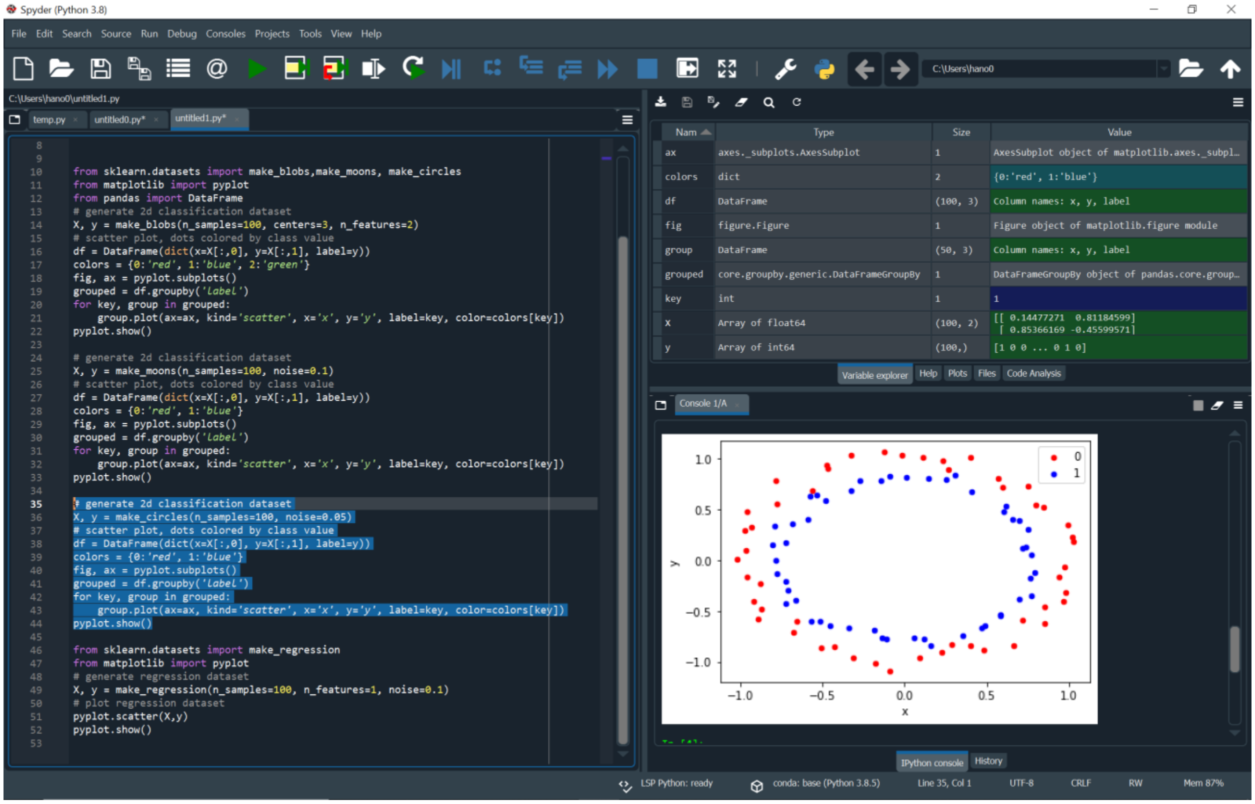
```
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()
```

Circles Classification Problem

Here, make_circles() function can be used to generate a binary classification problem with datasets that fall into concentric circles. In the function you can specify some properties including the noise of the shape. The problem is suitable for algorithms that can learn complex non-linear manifolds.

The figure below shows a 100 sample size circles dataset with 0.05 noise.

Output



```


from sklearn.datasets import make_blobs,make_moons, make_circles
from matplotlib import pyplot
from pandas import DataFrame
# generate 2d classification dataset
X, y = make_blobs(n_samples=100, centers=3, n_features=2)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue', 2:'green'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_moons(n_samples=100, noise=0.1)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_circles(n_samples=100, noise=0.05)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

from sklearn.datasets import make_regression
from matplotlib import pyplot
# generate regression dataset
X, y = make_regression(n_samples=100, n_features=1, noise=0.1)
# plot regression dataset
pyplot.scatter(X,y)
pyplot.show()

```

Nam	Type	Size	Value
ax	axes._subplots.AxesSubplot	1	AxesSubplot object of matplotlib.axes._subpl…
colors	dict	2	{0:'red', 1:'blue'}
df	DataFrame	(100, 3)	Column names: x, y, label
fig	figure.Figure	1	Figure object of matplotlib.figure module
group	DataFrame	(50, 3)	Column names: x, y, label
grouped	core.groupby.generic.DataFrameGroupBy	1	DataFrameGroupBy object of pandas.core.group…
key	int	1	1
x	Array of float64	(100, 2)	[[0.14477271 0.81184599] [0.85366169 -0.45599571]
y	Array of int64	(100,)	[1 0 0 ... 0 1 0]

Code

```
"""
Created on Fri Apr  2 15:10:10 2021
@author: Haneen Alamoudi
ID: 1708436
"""

# generate 2d classification dataset
```

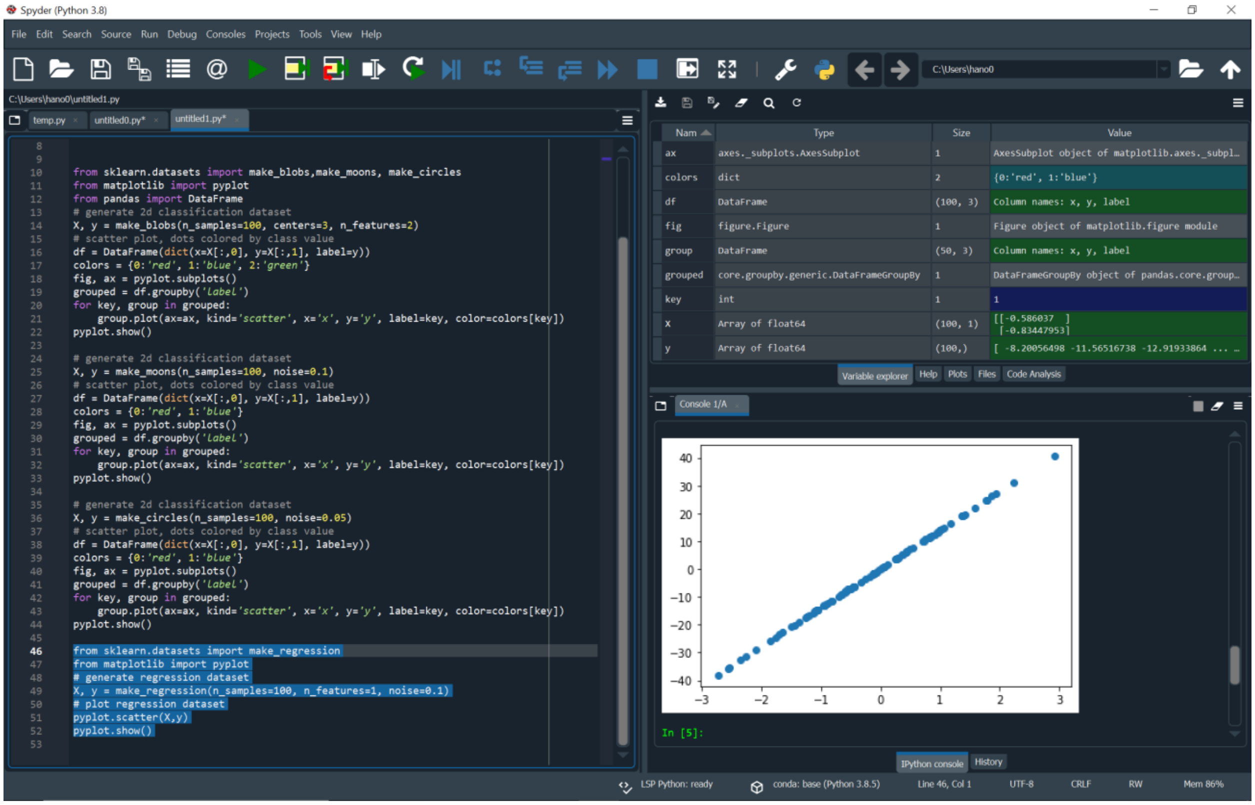
```
X, y = make_circles(n_samples=100, noise=0.05)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()
```

Regression Test Problems

Here, the dataset will be created to have a linear relationship between inputs and outputs by using make_regression() function. . In the function you can specify some properties including the number of samples, number of inputs feature and level of noise. The problem is suitable for algorithms that can learn a linear regression function.

The figure below shows a 100 sample size dataset with 0.1 noise and one input.

Output



```


from sklearn.datasets import make_blobs,make_moons, make_circles
from matplotlib import pyplot
from pandas import DataFrame
# generate 2d classification dataset
X, y = make_blobs(n_samples=100, centers=3, n_features=2)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue', 2:'green'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_moons(n_samples=100, noise=0.1)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

# generate 2d classification dataset
X, y = make_circles(n_samples=100, noise=0.05)
# scatter plot, dots colored by class value
df = DataFrame(dict(x=X[:,0], y=X[:,1], label=y))
colors = {0:'red', 1:'blue'}
fig, ax = pyplot.subplots()
grouped = df.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors[key])
pyplot.show()

from sklearn.datasets import make_regression
from matplotlib import pyplot
# generate regression dataset
X, y = make_regression(n_samples=100, n_features=1, noise=0.1)
# plot regression dataset
pyplot.scatter(X,y)
pyplot.show()

```

Nam	Type	Size	Value
ax	axes._subplots.AxesSubplot	1	AxesSubplot object of matplotlib.axes._subpl…
colors	dict	2	{0:'red', 1:'blue'}
df	DataFrame	(100, 3)	Column names: x, y, label
fig	figure.Figure	1	Figure object of matplotlib.figure module
group	DataFrame	(50, 3)	Column names: x, y, label
grouped	core.groupby.generic.DataFrameGroupBy	1	DataFrameGroupBy object of pandas.core.group…
key	int	1	1
X	Array of float64	(100, 1)	[[-0.586037] [-0.83447953]
y	Array of float64	(100,)	[-8.20056498 -11.56516738 -12.91933864 ... …

```
In [5]:
```

Code

```
"""
Created on Fri Apr  2 15:10:10 2021
@author: Haneen Alamoudi
ID: 1708436
"""
from sklearn.datasets import make_regression
from matplotlib import pyplot
# generate regression dataset
X, y = make_regression(n_samples=100, n_features=1, noise=0.1)
# plot regression dataset
pyplot.scatter(X,y)
pyplot.show()
```

Conclusion

During this case study multiple method were discussed and learned including how to generate a multi-class classification prediction test, this is done through the use of make_blobs() function. Next, how to generate a binary classification prediction test, and this can be achieved using make_moons() and make_circle() functions. Finally, linear regression prediction test. These methods can be used to generate test datasets that can be used to test new machine learning algorithms.