Creating sample data for introductory analysis

We can use scikit-learn to create toy data.

Getting ready

- Very similar to getting built-in datasets, fetching new datasets, and creating sample datasets, the functions that are used follow the naming convention `make_<the data set>`.
- Just to be clear, this data is purely artificial:

```
>>> datasets.make_*?
datasets.make_biclusters
datasets.make_blobs

datasets.make_checkerboard
datasets.make_circles
datasets.make_classification
...
```

```
>>> import sklearn.datasets as d
>>> import numpy as np
```

Implementation

```
>>> reg_data = d.make_regression()
```

- By default, this will generate a tuple with a 100 x 100 matrix : 100 samples by 100 features.
- By default, only 10 features are responsible for the target data generation. The second member of the tuple is the target variable.
- It is also possible to get more involved. For example, to generate a 1000 x 10 matrix with five features responsible for the target creation, an underlying bias factor of 1.0, and 2 targets, the following command will be run:

```
>>> complex_reg_data =
     d.make_regression(1000, 10, 5, 2, 1.0)
>>>
>>> complex_reg_data[0].shape
(1000, 10)
```

Artificial Data sets for Classification

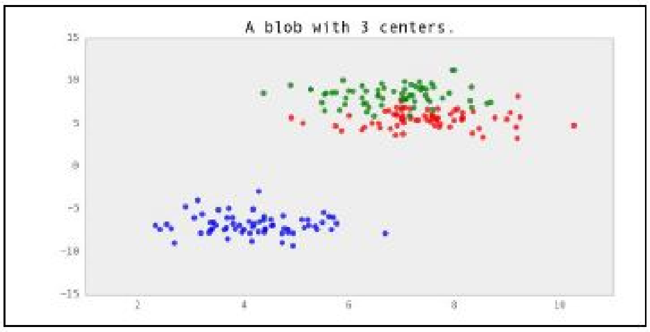
- Classification datasets are also very simple to create.
- It's simple to create a base classification set, but the basic case is rarely experienced in practice—most users don't convert, most transactions aren't fraudulent, and so on.
- Therefore, it's useful to explore classification on unbalanced datasets:

```
>>> classification_set =
   d.make_classification(weights=[0.1])
>>>
>>> np.bincount(classification_set[1])
array([10, 90])
```

Clusters

- There are actually several functions to create datasets that can be modeled by different cluster algorithms.
- For example, blobs are very easy to create and can be modeled by K-Means:

```
>>> blobs = d.make_blobs()
```


A blob with 3 centers.