17.6.4

Scale and Normalize Data

Data scaling and normalization are steps that are sometimes necessary when preparing data for machine learning. Jill explains that when features have different scales, features using larger numbers can have a disproportionate impact on the model. It is therefore important to understand when to scale and normalize data.

Earlier, we worked with a dataset that had already been scaled for us: the values in each column were rescaled to be between 0 and 1. Such scaling is often necessary with models that are sensitive to large numerical values. This is normally the case with models that measure distances between data points. SVM is one model that usually benefits from scaling.

In this section, we'll use Scikit-learn's `StandardScaler` module to scale data. The model -> fit -> predict/transform workflow is also used when scaling data. The standard scaler standardizes the data. This means that each feature will be rescaled so that its mean is 0 and its standard deviation is 1.

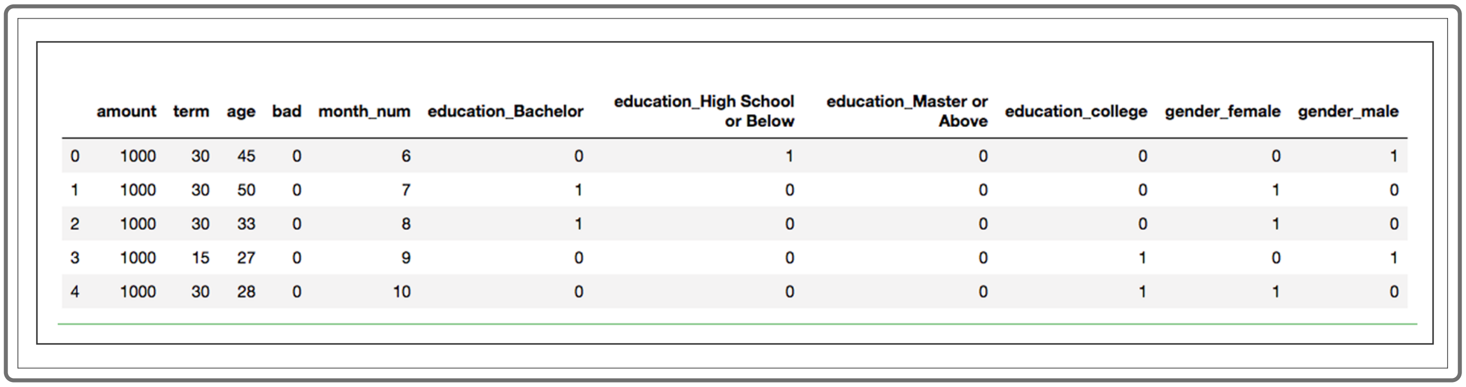
Get started by downloading the files you'll need.

Download 17-6-4-scale.zip (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_17/17-6-4-scale.zip)

Open Jupyter Notebook and read in your saved `loans_data_encoded.csv` file that you created in the previous Skill Drill. If you had trouble creating the encoded data, the dataset is included in the files you downloaded:

```
import pandas as pd
from path import Path

file_path = Path("../Resources/loans_data_encoded.csv")
encoded_df = pd.read_csv(file_path)
encoded_df.head()
```



	amount	term	age	bad	month_num	education_Bachelor	education_High School or Below	education_Master or Above	education_college	gender_female	gender_male
0	1000	30	45	0	6	0	1	0	0	0	1
1	1000	30	50	0	7	1	0	0	0	1	0
2	1000	30	33	0	8	1	0	0	0	1	0
3	1000	15	27	0	9	0	0	0	1	0	1
4	1000	30	28	0	10	0	0	0	1	1	0

To scale the data in this DataFrame, we'll first import the `StandardScaler` module and create an instance of it as `data_scaler`:

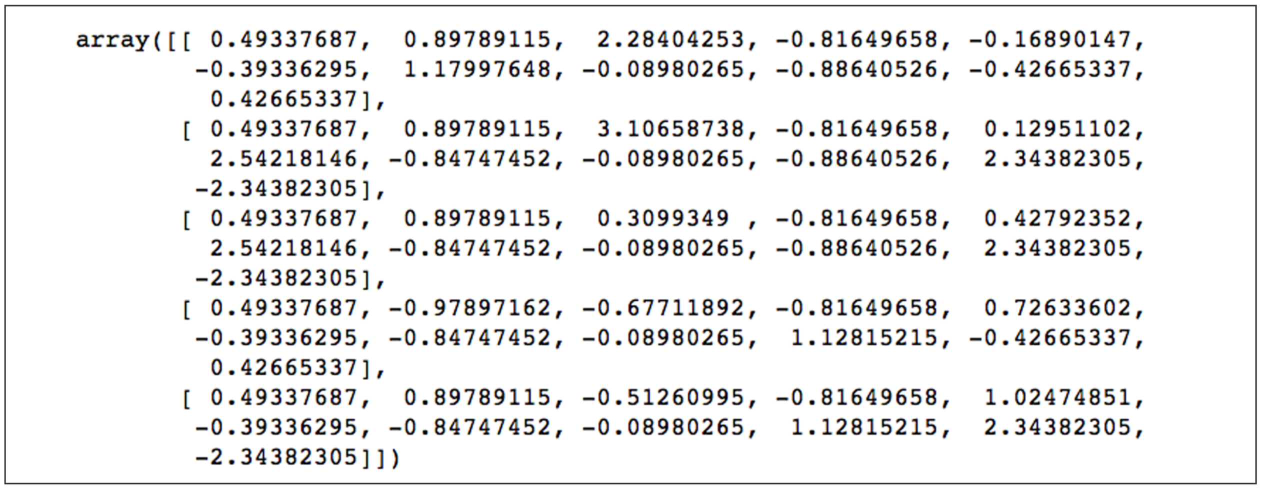
```
from sklearn.preprocessing import StandardScaler
data_scaler = StandardScaler()
```

The next step is to train the scaler and transform the data. Notice that the fit and transform steps are combined into a single `fit_transform()` method. However, they can also be used sequentially with `data_scaler.fit()`, then `data_scaler.transform()`:

```
loans_data_scaled = data_scaler.fit_transform(encoded_df)
```

After transforming the data, we can preview the scaled data:

```
loans_data_scaled[:5]
```



```
array([[ 0.49337687,  0.89789115,  2.28404253, -0.81649658, -0.16890147,
        -0.39336295,  1.17997648, -0.08980265, -0.88640526, -0.42665337,
         0.42665337],
       [ 0.49337687,  0.89789115,  3.10658738, -0.81649658,  0.12951102,
         2.54218146, -0.84747452, -0.08980265, -0.88640526,  2.34382305,
        -2.34382305],
       [ 0.49337687,  0.89789115,  0.3099349 , -0.81649658,  0.42792352,
         2.54218146, -0.84747452, -0.08980265, -0.88640526,  2.34382305,
        -2.34382305],
       [ 0.49337687, -0.97897162, -0.67711892, -0.81649658,  0.72633602,
        -0.39336295, -0.84747452, -0.08980265,  1.12815215, -0.42665337,
         0.42665337],
       [ 0.49337687,  0.89789115, -0.51260995, -0.81649658,  1.02474851,
        -0.39336295, -0.84747452, -0.08980265,  1.12815215,  2.34382305,
        -2.34382305]])
```

After transformation, the data in each column should be standardized. Let's verify that the mean of each column is 0 and its standard deviation is 1:

```
import numpy as np
print(np.mean(loans_data_scaled[:,0]))
print(np.std(loans_data_scaled[:,0]))
```

The following actions are taking place:

- The `np.mean()` and `np.std()` methods return the mean and standard deviation of an array.
- `loans_data_scaled[:,0]` returns all rows and the first column of the dataset.
- The mean of the first column is evaluated as -3.552713678800501e-16, an infinitesimally small number that approximates 0.
- The standard deviation is evaluated as 0.9999999999999999, which is very close to 1.

So the standardization was indeed successful, and all numerical columns should now have a mean of 0 and a standard deviation of 1, reducing the likelihood that large values will unduly influence the model.

SKILL DRILL

`loans_data_scaled.shape` returns (500, 11), indicating that there are 500 rows and 11 columns. Create a `for` loop to verify that all columns of the dataset have been standardized.