This notebook is an exercise in the Data Cleaning (https://www.kaggle.com/learn/data-cleaning) course. You can reference the tutorial at this link (https://www.kaggle.com/alexisbcook/scaling-and-normalization).

In this exercise, you'll apply what you learned in the **Scaling and normalization** tutorial.

Setup

The questions below will give you feedback on your work. Run the following cell to set up the feedback system.

In [1]:

```
from learntools.core import binder
binder.bind(globals())
from learntools.data_cleaning.ex2 import *
print("Setup Complete")
```

```
Setup Complete
```

Get our environment set up

To practice scaling and normalization, we're going to use a dataset of Kickstarter campaigns (https://www.kaggle.com/kemical/kickstarter-projects). (Kickstarter is a website where people can ask people to invest in various projects and concept products.)

The next code cell loads in the libraries and dataset we'll be using.

In [2]:

```
# modules we'll use
import pandas as pd
import numpy as np

# for Box-Cox Transformation
from scipy import stats

# for min_max scaling
from mlxtend.preprocessing import minmax_scaling

# plotting modules
import seaborn as sns
import matplotlib.pyplot as plt

# read in all our data
kickstarters_2017 = pd.read_csv("../input/kickstarter-projects/ks-projects-201

# set seed for reproducibility
np.random.seed(0)
```

Let's start by scaling the goals of each campaign, which is how much money they were asking for. After scaling, all values lie between 0 and 1.

In [3]:

```
# select the usd_goal_real column
original_data = pd.DataFrame(kickstarters_2017.usd_goal_real)

# scale the goals from 0 to 1
scaled_data = minmax_scaling(original_data, columns=['usd_goal_real'])

print('Original data\nPreview:\n', original_data.head())
print('Minimum value:', float(original_data.min()),
      '\nMaximum value:', float(original_data.max()))
print('_'*30)

print('\nScaled data\nPreview:\n', scaled_data.head())
print('Minimum value:', float(scaled_data.min()),
      '\nMaximum value:', float(scaled_data.max()))
```

```
Original data
Preview:
    usd_goal_real
0        1533.95
1       30000.00
2       45000.00
3        5000.00
4       19500.00
Minimum value: 0.01
Maximum value: 166361390.71
______________________________

Scaled data
Preview:
    usd_goal_real
0       0.000009
1       0.000180
2       0.000270
3       0.000030
4       0.000117
Minimum value: 0.0
Maximum value: 1.0
```

1) Practice scaling

We just scaled the "usd_goal_real" column. What about the "goal" column?

Begin by running the code cell below to create a DataFrame `original_goal_data` containing the "goal" column.

In [17]:

```
# select the usd_goal_real column
original_goal_data = pd.DataFrame(kickstarters_2017.goal)
print(original_goal_data)
```

```
           goal
0        1000.0
1       30000.0
2       45000.0
3        5000.0
4       19500.0
...         ...
378656  50000.0
378657   1500.0
378658  15000.0
378659  15000.0
378660   2000.0

[378661 rows x 1 columns]
```

Use `original_goal_data` to create a new DataFrame `scaled_goal_data` with values scaled between 0 and 1. You must use the `minmax_scaling()` function.

In [18]:

```
# TODO: Your code here
scaled_goal_data = minmax_scaling(original_goal_data,columns=["goal"])

# Check your answer
q1.check()
```

Correct

2) Practice normalization

Now you'll practice normalization. We begin by normalizing the amount of money pledged to each campaign.

In [ ]:

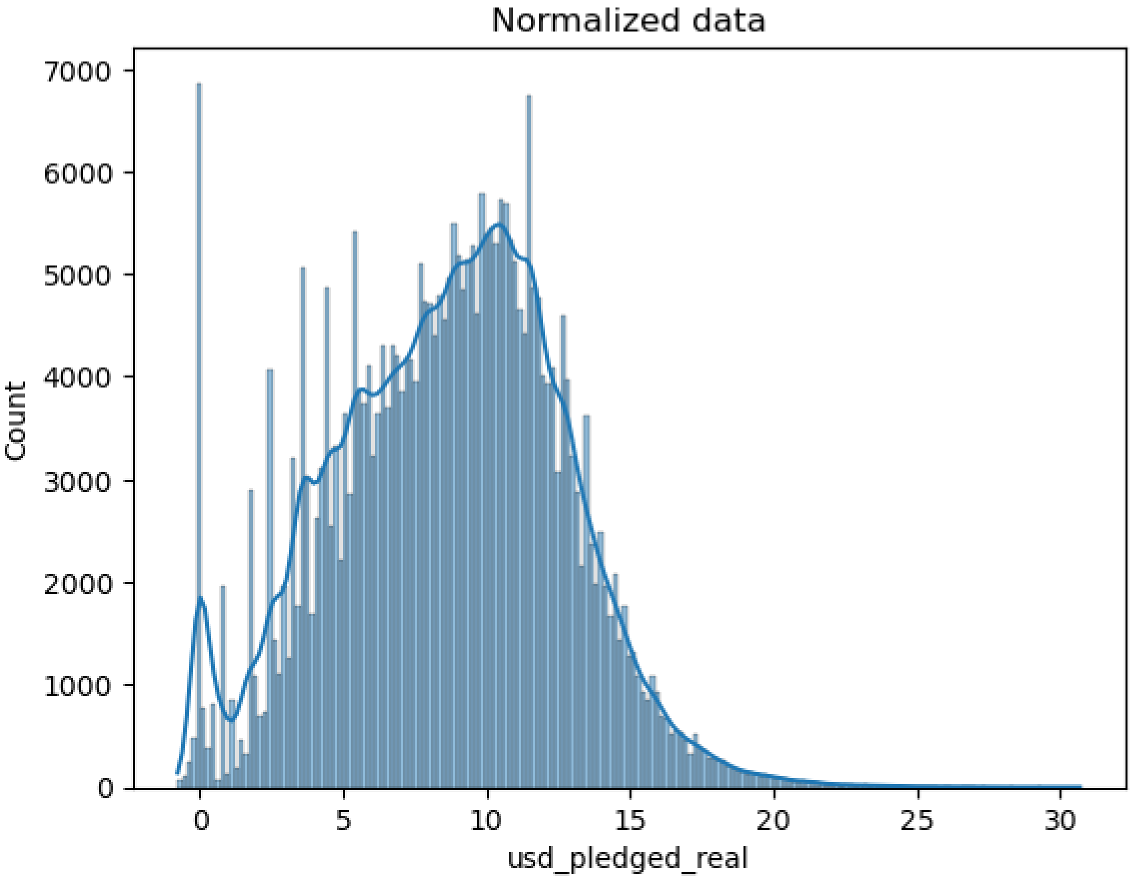
```
For each of the following examples, decide whether scaling or normalization mal

You want to build a linear regression model to predict someone's grades given 
You're still working on your grades study, but you want to include information
Once you have an answer, run the code cell below.

# TODO: Your code here!
# normalized_pledges = pd.Series(stats.boxcox(positive_pledges)[0],
#                                  name='pledged', index=positive_pledges.index)
index_positive_pledges = kickstarters_2017.pledged > 0

# get only positive pledges (using their indexes)
positive_pledges_only = kickstarters_2017.pledged.loc[index_positive_pledges]

# normalize the pledges (w/ Box-Cox)
normalized_values = pd.Series(stats.boxcox(positive_pledges_only)[0],
                              name='pledged', index=positive_pledges_only.inde

# plot both together to compare
fig, ax = plt.subplots(1,2,figsize=(15,3))
sns.distplot(positive_pledges_only, ax=ax[0])
ax[0].set_title("Original Data")
sns.distplot(normalized_values, ax=ax[1])
ax[1].set_title("Normalized data")
```

The values have changed significantly with normalization!

In the next code cell, you'll take a look at the distribution of the normalized data, where it should now resemble a normal distribution.

In [8]:

```
# plot normalized data
ax = sns.histplot(normalized_pledges, kde=True)
ax.set_title("Normalized data")
plt.show()
```



We used the "usd_pledged_real" column. Follow the same process to normalize the "pledged" column.

In [ ]:

```
# TODO: Your code here!
For each of the following examples, decide whether scaling or normalization mal

You want to build a linear regression model to predict someone's grades given
You're still working on your grades study, but you want to include information
Once you have an answer, run the code cell below.

# TODO: Your code here!
# normalized_pledges = pd.Series(stats.boxcox(positive_pledges)[0],
#                                 name='pledged', index=positive_pledges.index)
index_positive_pledges = kickstarters_2017.pledged > 0

# get only positive pledges (using their indexes)
positive_pledges_only = kickstarters_2017.pledged.loc[index_positive_pledges]

# normalize the pledges (w/ Box-Cox)
normalized_values = pd.Series(stats.boxcox(positive_pledges_only)[0],
                              name='pledged', index=positive_pledges_only.inde

# plot both together to compare
fig, ax = plt.subplots(1,2,figsize=(15,3))
sns.distplot(positive_pledges_only, ax=ax[0])
ax[0].set_title("Original Data")
sns.distplot(normalized_values, ax=ax[1])
ax[1].set_title("Normalized data")
```

How does the normalized "usd_pledged_real" column look different from when we normalized the "pledged" column? Or, do they look mostly the same?

Once you have an answer, run the code cell below.

In [10]:

```
# Check your answer (Run this code cell to receive credit!)
q2.check()
```

Correct:

The distributions in the normalized data look mostly the same.

In [11]:

```
# Line below will give you a hint
#q2.hint()
```

(Optional) More practice

Try finding a new dataset and pretend you're preparing to perform a regression analysis (https://www.kaggle.com/rtatman/the-5-day-regression-challenge).

These datasets are a good start! (https://www.kaggle.com/rtatman/datasets-for-regression-analysis)

Pick three or four variables and decide if you need to normalize or scale any of them and, if you think you should, practice applying the correct technique.

Keep going

In the next lesson, learn how to **parse dates** (https://www.kaggle.com/alexisbcook/parsing

---

Have questions or comments? Visit the course discussion forum (https://www.kaggle.com/learn/data-cleaning/discussion) to chat with other learners.