This notebook is an exercise in the Data Cleaning course. You can reference the tutorial at this link.

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

Get our environment set up

To practice scaling and normalization, we're going to use a dataset of Kickstarter campaigns. (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 [ ]:
```
# 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-201801.csv")

# 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. The plots show a histogram of the values in the "usd_goal_real" column, both before and after scaling.

In [ ]:
```
# 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'])

# plot the original & scaled data together to compare
fig, ax=plt.subplots(1,2,figsize=(15,3))
sns.distplot(kickstarters_2017.usd_goal_real, ax=ax[0])
ax[0].set_title("Original Data")
sns.distplot(scaled_data, ax=ax[1])
ax[1].set_title("Scaled data")
```

After scaling, all values lie between 0 and 1 (you can read this in the horizontal axis of the second plot above, and we verify in the code cell below).

In [ ]:
```
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()))
```

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 [ ]:
```
# select the usd_goal_real column
original_goal_data = pd.DataFrame(kickstarters_2017.goal)
original_goal_data.head()
```

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

In [ ]:
```
# TODO: Your code here
scaled_goal_data = scaled_data = minmax_scaling(original_goal_data, col
umns = ['goal'])
# Check your answer
q1.check()
```

In [ ]:
```
# Lines below will give you a hint or solution code
#q1.hint()
#q1.solution()
```

2) Practice normalization

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

In [ ]:
```
# get the index of all positive pledges (Box-Cox only takes positive va
lues)
index_of_positive_pledges = kickstarters_2017.usd_pledged_real > 0

# get only positive pledges (using their indexes)
positive_pledges = kickstarters_2017.usd_pledged_real.loc[index_of_posi
tive_pledges]

# normalize the pledges (w/ Box-Cox)
normalized_pledges = pd.Series(stats.boxcox(positive_pledges)[0],
                               name='usd_pledged_real', index=positive_
pledges.index)

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

It's not perfect (it looks like a lot pledges got very few pledges) but it is much closer to a normal distribution!

In [ ]:
```
print('Original data\nPreview:\n', positive_pledges.head())
print('Minimum value:', float(positive_pledges.min()),
      '\nMaximum value:', float(positive_pledges.max()))
print('_'*30)

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

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

column.

In [ ]:
```
# TODO: Your code here!
```

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

In [ ]:
```
# 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.

These datasets are a good start!

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** in a dataset.

Have questions or comments? Visit the Learn Discussion forum to chat with other Learners.