COMP3430/COMP8430 – Data Wrangling – 2022

Lab 2: Data Cleaning and transformation using Rattle and Pandas Week 4

Overview and Objectives

The objectives of this second lab are to get familiar with the functionalities available in Rattle and the Python Pandas library that can be used to conduct data cleaning and data transformation on smaller example data sets. In this lab we will work on imputing missing values in variables / attributes and learn how to perform data transformation tasks using Rattle and Pandas.

Before you begin, please take a moment to go back over the work from lab 1 and remind yourself about Rattle and the Python Pandas library, and how to start and quit Rattle and Python. Also, for more on R and how to download Rattle and Pandas please see the **Course Resources** link on the COMP3430/COMP8430 Wattle page.

Part 1: Lab Questions using Rattle

This first part of the lab basically consists of working through the `Transform` tab in Rattle. Similar to the previous lab we will use the **weather** data set in this lab. The **weather** data set comes from the Bureau of Meteorology and contains 366 records (observations) and 24 variables (or attributes). Note: If you are using Ubuntu the location where the Rattle example data sets are located is `/usr/local/lib/R/site-library/rattle/csv/`.

If you want to look at other data sets then click on the `Library` radio button, and select one of the many available data sets from the menu available next to `Data Name`. You need to confirm your selection with a click on the `Execute` button.

Before you continue make sure you have selected the **weather** data set (and have confirmed your selection with a click on `Execute`).

Make sure you also untick the `Partition` box (and then again click on `Execute`). Partitioning is only required for supervised machine learning algorithms which we do not cover in this course.

Now go to the `Explore` tab, which offers a large number of possibilities to explore the loaded data set.

1. The first thing we need to explore is the content of the **weather** data set using the various options provided and the number of missing values that occur in each variable. To explore missing values we can use the `Show Missing` option under the type `Summary` in the Explore tab. Which variable has the largest number of missing values?

2. Imputation is the process of filling in missing values in data. Go to the `Transform` tab to impute values for missing data. Try different imputation methods and apply them on the variables with missing values. See the chapter on Transforming Data in the Rattle online documentation for explanations about the various transformation functions available, or the information about transformation from the Help menu.

   Note, in order to perform a data transformation, you need to highlight one or more variables (use Shift-click or Ctrl-click to highlight several variables), and then click on `Execute` to do the actual selected transformation.

   As you will see, Rattle will generate a new variable and append it at the end of the list (scroll down to the bottom of the list). The new variable will have a name based on the transformation performed, concatenated with the original variable name.

   Why do you think Rattle has not overwritten / modified the values in the original variable?

   If you now go back to the `Data` page you will notice that Rattle has set the role of the original variables you just transformed to Ignore, while the newly generated variable has been set as an input variable.

3. Conduct the following three normalisations on one of the temperature variables (MinTemp or MaxTemp): `Recenter`, `Scale [0-1]`, `-Median/MAD`. Then inspect the three new created variables (named for example RRC_MinTemp, R01_MinTemp, and RMD_MinTemp) on the `Explore` page by generating appropriate visualisations. How is the MAD calculated?

   To only inspect these three variables go back to the `Data` page, and set the role for all other variables to `Ignore` (keep the original temperature variable you used as input), click `Execute` and go to the `Explore` page.

4. The `Rank` option will convert each observation's numeric value for the identified variable into a ranking in relation to all other observations in the data set. See how ranking can be applied on different variables in the data sets.

5. The `Interval` option recodes the values of a variable into a rank order between 0 and a certain number (with 100 being the default). A categorical variable can also be identified as part of the transformation. See how ranking can be applied on different variables. An example might be to rank wind speeds within groups defined by the wind direction.

6. Create both `box plots` and `histograms` of the original variables and the variables you transformed using different transformation methods. How do they differ?

Part 2: Lab Questions using Pandas

Before we start the lab questions we need to start Python in *anaconda*. Open a terminal window and type `anaconda` followed by 'Enter' to start the anaconda distribution. This will start the anaconda package and the prompt should have changed to 'yourusername@anaconda:/$'. Now type `python` into the terminal window to start Python. Note that if you start anaconda via the Ubuntu menu then importing the `matplotlib` library might not work.

Now we need to import the necessary Python modules. Type the following code snippet to import pandas, numpy, and matplotlib.pyplot (for plotting functions).

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Next we load the weather data set into a *DataFrame* in Pandas. To read the weather data set into a DataFrame named `df` we use the function *read_csv()* in pandas.

```
df = pd.read_csv("/usr/local/lib/R/site-library/rattle/csv/weather.csv")
```

Note: You may want to copy this weather data set into your working folder and read the file from there rather than reading it from its original location.

1. Now let's try to impute missing values. To view the missing values in a DataFrame `df` we can used the function *isnull()*. To select the rows that have at least one missing value we can use `df[df.isnull().any(axis=1)]`. To select rows that have missing values in all columns we can use `df[df.isnull().all(axis=1)]`. Missing values will be marked as NaN (not-a-number) in the result.

   Note that an axis refers to the dimension of the data frame. `axis=0` means by column or column-wise, and `axis=1` means by row or row-wise. See if you can view the number of missing values in each column in your data frame.

   There are a number of methods to deal with missing values in a data frame, as shown in the following table.

DataFrame.method	description
dropna()	Drop records with missing values
dropna(how='any')	Drop records if any columns contain a missing value
dropna(how='all')	Drop records where values in all columns are missing
dropna (axis=1, how='all')	Drop column if all the values are missing
dropna(thresh = 5)	Drop records that contain less than 5 non-missing values
fillna(0)	Replace missing values with zeros

   The function `fillna` can be used to impute variables / attributes / columns with mean, median, and mode values. We can do this by, for example, using:

   ```
   mean_val = df['column'].mean()
   ```

   Then we can use this mean value to impute any missing values in that column using:

   ```
   df['column'].fillna(mean_val)
   ```

   Now try to impute different variables using mean, median, and mode values of the variables. You can use the functions `median()` and `mode()` to get the median and mode of an variable (column).

2. Data normalisation is another important aspect we can try in Pandas. For example if we want to apply z-score normalisation to the variable *MinTemp* in the weather data set we can use the following code snippet:

   ```
   mean_val = df['MinTemp'].mean()   # Compute the mean value
   std_val =  df['MinTemp'].std()    # Compute the standard deviation value
   MinTempZsc = []                   # Initialise an empty list to hold the normalised values
   for val in df['MinTemp']:         # Calculate the z-score for each row in the data frame
     MinTempZsc.append((val - mean_val) / std_val)
   df['MinTempZsc'] = MinTempZsc     # Add the normalised variable / column to the data frame
   ```

   In Pandas we have to write the normalised values into a new column (variable) in the data frame, while in Rattle a new variable is created automatically when we apply a transformation function. So in the example above we create a new column / variable *MinTempZsc* in the data frame `df`.

   You can view the columns in the data frame by using `df.columns`.

   Programming idea: Similar to the Rattle exercise above, implement and apply min-max [0-1] normalisation, robust normalisation, and log normalisation on different variables in the weather data set.

3. Similar to Rattle, visualisation with Pandas is easy when you want to view your data. To view different plots we need to use the `plot` function as we covered in the first lab (see page 4 of the lab 1 specifications).

   Create both `box plots` and `histograms` of the original variables and the variables you transformed using the different transformation methods. How do they differ?