Outlier Detection and Removal

This lesson will focus on how to detect outliers in the data and what to do with them.

Outlier detection

Detecting outliers is a very important step in data cleaning and exploring. It gives us an idea of the anomalies in the data which can give us valuable insights into the data. So, how can we detect outliers?

Outliers can be detected both visually and mathematically. Some plots are very helpful in visualizing outliers, such as box plots and scatter plots. However, it is sometimes tricky to decide whether or not to remove the outliers. We should remove outliers when we are certain that these outliers were results of some errors.

We will discuss some of the methods to detect and remove outliers. We will be using the Sample Sales Data. The data is in the file *sales_data.csv*.

sales_data.csv

Box plots and Quantile ranges

Box plots, by definition, plot outliers as points and group the rest of the observations. The criteria of a box plot for classifying a point as an outlier is if the point is greater than $Q_3 + (1.5 * IQR)$ or lower than $Q_1 - (1.5 * IQR)$.

where Q_1 = First Quartile

Q_3 = Third Quartile

IQR = Inter Quartile Range = Q_3 - Q_1

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

df = pd.read_csv('sales_data.csv')

# plot box plots
df.plot(kind = 'box',subplots = True,layout = (4,3),sharex = False, sharey=False,figsize=(10,
plt.show()
```

We plot this in **line** 7. We give the size of the figure to the `plot` function as `figsize = (10,10)`. This argument is optional but can be given if the plots are not rendering correctly on the screen. From the plot, we see that `SALES` and `QUANTITYORDERED` have quite a few outliers while `MSRP` has one outlier.

We can also use quartile ranges to filter for outliers. Let's see an example of that below. We will be filtering rows that are outliers in all three variables.

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

df = pd.read_csv('sales_data.csv')
print('Original shape : ',df.shape)

# Retrieve only outlier columns
new_df = df[['QUANTITYORDERED','MSRP','SALES']]

# find max and min using IQR
Q1 = new_df.quantile(0.25)
Q3 = new_df.quantile(0.75)
IQR = Q3-Q1
minimum = Q1 - 1.5*IQR
maximum = Q3 + 1.5*IQR

# condition on which to filter
condition = (new_df <= maximum) & (new_df >= minimum)
print('\n condition in all columns : \n',condition.head())
condition = condition.all(axis=1)
print('\n\n condition combined to a single column \n',condition.head())

# Filter rows that have outliers
df = df[condition]
print('Shape after removing outliers: ',df.shape)
```

We make a new dataframe, `new_df`, with just the three columns having outliers that we identified earlier in **line 8**. We give the list of columns that we need in the new dataframes inside the square brackets appended to `df`. We find the quantiles using the function `quantile` in **lines 11-12**. Then we find `IQR` and the `maximum` and `minimum`.

Our goal is to filter the dataframe `df` and remove those rows that have outlier values in all three columns. To do that, we can provide the dataframe `df` a list of booleans(`True` and `False`) as we learned in the previous chapter.

Therefore, we write our condition on **line 18**. It says that the values should be between the minimum and the maximum boundary values. **Line 18** gives us a dataframe that has `True` or `False` for every cell in `new_df` based on whether or not it satisfies the condition. We can see this by the output of **line 19**. In the next line, we specify that the condition should be true for all three columns by using the `all` function with `axis=1` argument. This gives us a list of `True`/`False` against each row. If a row has all three `True` values, then it gives a `True` value to that row. We can verify this by looking at the output of **line 21**. Then we simply filter the original dataframe `df` in **line 24** using the list we obtained above.

Scatter plots

Scatter plots are a handy way of looking at point and collective outliers. In scatter plots, the values of two variables are plotted along two axes. Let's see an example below.

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

df = pd.read_csv('sales_data.csv')
df = df[['SALES','QUANTITYORDERED','MSRP']]

# Scatter plot
pd.scatter_matrix(df,figsize = (10,10))
```

We choose only three variables that will make sense for the scatter plot and draw scatter plots between all of them using the function `pd.scatter_matrix`. This gives us a table of scatter plots. We can see a few outliers easily using these scatter plots.

Z - Score

Z score is the number of standard deviations an observation is above or below the mean of a variable. Z scores are used in statistics to study variance of data. We can use z-scores to filter outliers easily.

We find z-scores by subtracting the mean and dividing the standard deviation.

```
import pandas as pd
import numpy as np

df = pd.read_csv('sales_data.csv')
print('Shape of originl dataframe : ' , df.shape)

# find the mean and std dev
mean = df['QUANTITYORDERED'].mean()
std_dev = df['QUANTITYORDERED'].std()

# find z scores
z_scores = (df['QUANTITYORDERED'] - mean) / std_dev
z_scores = np.abs(z_scores)

# filter using zscores
df = df[z_scores < 3]
print('Shape after removing outliers : ',df.shape)
```

We removed outliers from the column `QUANTITYORDERED`. We first find mean and the standard deviation using the `mean` and `std` functions. Then we find z-scores in **line 12** by subtracting `mean` (calculated in **line 8**) and dividing by `std_dev` (calculated in **line 9**). We take the absolute values of the z-scores using the function `np.abs`. We imported `numpy` on **line 2** to use this function. In the end, we filter using `z_scores`. We set the condition that the value of `z_scores` must be less than 3. We choose the score of 3 because, in normally distributed data, approximately 97% of the data lies inside 3 standard deviations. So, if a data value has a z-score greater than 3, it is an outlier.

This marks the end of this lesson. In the next lesson, you will be cleaning a dataset yourself as an exercise.