Big Data Analytics
Outliers

Muhammad Affan Alim

Outliers

- An outlier is a data point that is significantly different from the remaining data
- Outliers may also affect the performance of some machine learning models, such as linear regression and AdaBoost

Outliers- cont…

How can we engineer outliers?

- One way to handle outliers is to perform variable discretization
- An alternative way to handle outliers is to assume that the information is missing, treat the outliers together with the remaining missing data, and carry out any of the missing imputation techniques

Outliers –cont…

- We will also discuss how to use the mean and standard deviation for normally distributed variables or the inter quartile range for skewed features or using percentiles, in a process commonly known as **winsorization**

Trimming outliers from the dataset

1. Import the required Python libraries:

```
>> import pandas as pd
>> import numpy as np
>> import matplotlib.pyplot as plt
>> import seaborn as sns
>> from sklearn.datasets import load_boston
```

Trimming outliers from the dataset-cont…

2. Let's load the Boston House Prices dataset from scikit-learn:

```
>> boston_dataset = load_boston()
```

3. Let's capture three of the variables, RM, LSTAT, and CRIM, in a pandas dataframe:

```
>> boston = pd.DataFrame(boston_dataset.data,
columns=boston_dataset.feature_names)[['RM', 'LSTAT', 'CRIM']]
```

Trimming outliers from the dataset-cont…

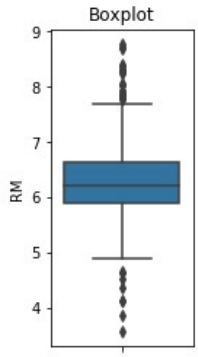
- Let's make a boxplot of the RM variable to visualize outliers:

```
>> sns.distplot(boston['RM])
>> sns.boxplot(boston['RM'])
```

- The outliers are the asterisks sitting outside the whiskers, which delimit the interquartile range proximity rule boundaries:

Trimming outliers from the dataset-cont…

- The outliers are the asterisks sitting outside the whiskers, which delimit the interquartile range proximity rule boundaries:



Trimming outliers from the dataset-cont…

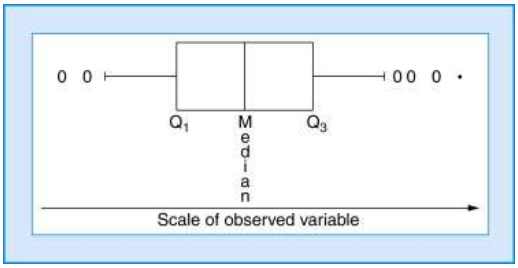
5. Let's create a function to find the boundaries of a variable distribution, using the inter-quartile range proximity rule:

- The IQR describes the middle 50% of values when ordered from lowest to highest. To find the interquartile range (IQR), first find the median (middle value) of the lower and upper half of the data. These values are quartile 1 (Q1) and quartile 3 (Q3). The IQR is the difference between Q3 and Q1.

$$IQR = Q3 - Q1$$

The Box Plot – by interquartile

- The box plot is used to show distributional shapes and to detect unusual observations.



The Box Plot – by interquartile cont…

The features of the plot are as follows:

1. The "box," representing the interquartile range, has a value we denote by *R* and the endpoints *Q1* and *Q3*.

2. A vertical line inside the box indicates the median. If the median is in the centre of the box, the middle portion of the distribution is symmetric.

The Box Plot – by interquartile cont…

3. Horizontal lines extending from the box represent the range of observed values inside the "inner fences," which are located 1.5 times the value of the interquartile range (1.5*R*) beyond *Q1* to the left and *Q3* on the right. The relative lengths of these lines are an indicator of the skewness of the distribution as a whole.

The Box Plot – by interquartile cont…

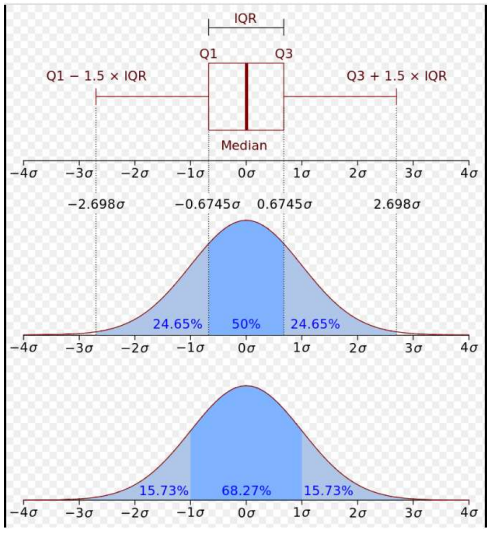
4. Individual symbols O represent "mild" outliers, which are defined as values between the inner and outer fences that are located *3R* units beyond *Q1* and *Q3*.

5. Individual symbols represent the location of extreme outliers, which are defined as being beyond the outer fences. Different computer programs may use different symbols for outliers and may provide options for different formats.

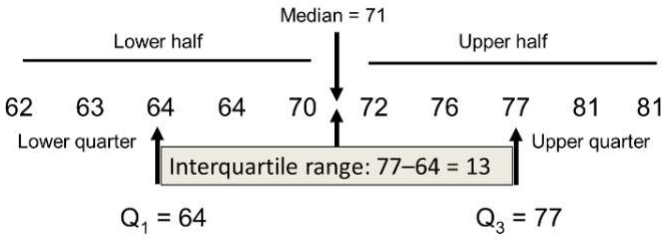
The Box Plot – by interquartile cont…

i	x[i]	Median	Quartile
1	7		
2	7		
3	31		Q_1=31 (median of upper half, from row 1 to 6)
4	31		
5	47		
6	75		
7	87	Q_2=87 (median of whole table)	
8	115		
9	116		
10	119		Q_3=119 (median of lower half, from row 8 to 13)
11	119		
12	155		
13	177		

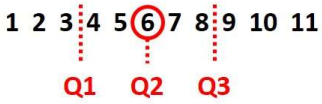
For the data in this table the interquartile range is IQR = $Q_3 - Q_1$ = 119 - 31 = 88.



The Box Plot – by interquartile cont…



$$IQR = Q3 - Q1$$
$$= 8.5 - 3.5$$
$$= 5$$



Trimming outliers from the dataset-cont…

```
>> def find_boundaries(df, variable, distance):
        IQR = df[variable].quantile(0.75) - df[variable].quantile(0.25)
        lower_boundary = df[variable].quantile(0.25) - (IQR * distance)
        upper_boundary = df[variable].quantile(0.75) + (IQR * distance)
        return upper_boundary, lower_boundary
```

Trimming outliers from the dataset-cont…

6. Let's use the function from *step 5* to determine the limits of the RM variable:

```
>> RM_upper_limit, RM_lower_limit = find_boundaries(boston, 'RM', 1.5)
```

7. Let's print those limits beyond which we will consider a value an outlier:

RM_upper_limit, RM_lower_limit The output of the preceding code is as follows:

```
(7.730499999999999, 4.778500000000001)
```

Trimming outliers from the dataset-cont…

- Let's create a Boolean vector to flag the outliers in RM:

```
>> outliers_RM = np.where(boston['RM'] > RM_upper_limit, True,
np.where(boston['RM'] < RM_lower_limit, True,
False))
```

Trimming outliers from the dataset-cont…

- Finally, let's remove the outliers from the dataset:

```
>> boston_trimmed = boston.loc[~(outliers_RM)]
```

Making NaN outliers

```
>> df_new = boston.RM[(boston.RM < RM_upper_limit) & (boston.RM > RM_lower_limit)]
>> boston['RM_new'] = df_new
```

Trimming outliers from the dataset-cont…

- With the pandas' quantile() method, we can calculate the values for the 25th (0.25) and 75th quantiles (0.75).
- We then used this function to return the upper and lower boundaries for the RM variable.
- To find the outliers of RM, we used *np.where*(), which produced a Boolean vector with True if the value was an outlier, that is, if the value was bigger or smaller than the upper or lower boundaries determined for RM.

Trimming outliers from the dataset-cont…

- Briefly, np.where() scanned the rows of the RM variable, and if the value was bigger than the upper boundary, it assigned True; whereas if the value was smaller, the second NumPy's where() method, nested in the first one, checked whether the value was smaller than the lower boundary, in which case, it also assigned True; otherwise, it assigned False.

- Finally, we used the loc[] method from pandas to remove the observations that contained outliers for RM. The ~ symbol used with the pandas' loc[] method removes from the DataFrame the outliers captured in the Boolean vector, outliers_RM.

Trimming outliers – mean and standard deviation

- If instead of using the inter-quartile range proximity rule, we want to use the mean and standard deviation to find the limits, we need to replace the code in the function in *step 5*:

1. Find the outlier boundaries using the mean and standard deviation:

```
>> def find_boundaries(df, variable, distance):
>>     lower_boundary = df[variable].mean() - (df[variable].std() *distance)
>>     upper_boundary = df[variable].mean() + (df[variable].std() *distance)
>>     return upper_boundary, lower_boundary
```

Trimming outliers – mean and standard deviation

To calculate the boundaries for the RM variable with the preceding function, we run the following code.

2. Calculate the boundaries for RM:

```
>> RM_upper_limit, RM_lower_limit = find_boundaries(boston, 'RM', 3)
```

Trimming outliers – alternate method of Quantile

- Alternatively, if we want to use quantiles to calculate the limits, we should write the function like in the next step.

3. Find the outlier boundaries using quantiles:

```
>> def find_boundaries(df, variable):
>>     lower_boundary = df[variable].quantile(0.05)
>>     upper_boundary = df[variable].quantile(0.95)
>>     return upper_boundary, lower_boundary
```

Trimming outliers – mean and standard deviation

4. Calculate the boundaries for RM:

```
>> RM_upper_limit, RM_lower_limit = find_boundaries(boston, 'RM')
```

- The rest of the procedure is identical to the one described in *step 8* and *step 9*, in the *How to do it...* section of the recipe

Trimming outliers – mean and standard deviation

5. Let's calculate the boundaries for the RM, LSTAT, and CRIM variables:

```
>> RM_upper_limit, RM_lower_limit = find_boundaries(boston, 'RM', 1.5)
>> LSTAT_upper_limit, LSTAT_lower_limit = find_boundaries(boston,'LSTAT', 1.5)
>> CRIM_upper_limit, CRIM_lower_limit = find_boundaries(boston, 'CRIM', 1.5)
```

Trimming outliers – mean and standard deviation

6. Let's create Boolean vectors that flag the outliers for each one of RM, LSTAT, and

CRIM:

```
>> outliers_RM = np.where(boston['RM'] > RM_upper_limit, True, np.where(boston['RM'] < RM_lower_limit, True, False))
>> outliers_LSTAT = np.where(boston['LSTAT'] > LSTAT_upper_limit, True, np.where(boston['LSTAT'] < LSTAT_lower_limit, True,False))
>> outliers_CRIM = np.where(boston['CRIM'] > CRIM_upper_limit, True, np.where(boston['CRIM'] < CRIM_lower_limit, True, False))
```

Trimming outliers – mean and standard deviation

7. Finally, let's remove the observations with outliers in any of the variables:

```
>> boston_trimmed = boston.loc[~(outliers_RM + outliers_LSTAT + outliers_CRIM)]
```

Binning

df['horsepower-binned'] = pd.cut(df['horsepower'], bins, labels=group_names, include_lowest=True)

- df['horsepower-binned'] = pd.cut(df['horsepower'], bins, labels=group_names, include_lowest=True)

30