Lecture 5
–
Outliers

Rotkehlchen / European Robin
(*Erithacus rubecula*)

- It is about 12.5–14.0 cm in length, the male and female are similar in colouration, with an orange breast and face lined with grey, brown upper-parts and a whitish belly.

- The distinctive orange breast of both sexes contributed to the European robin's original name of "redbreast", orange as a colour name being unknown in English until the 16th century, by which time the fruit had been introduced.

- The robin produces a fluting, warbling song during the breeding season with 275 different melodies. Both the male and female sing during the winter, when they hold separate territories.

- The avian magnetic compass of the robin has been extensively researched and uses vision-based magnetoreception, in which the robin's ability to sense the magnetic field of the earth for navigation is affected by the light entering the bird's eye. The physical mechanism of the robin's magnetic sense involves quantum entanglement of electron spins in cryptochrome in the bird's eyes.

Sources:
* Photo by Andhoj on pixabay
* https://en.wikipedia.org/wiki/European_robin

Data Science

Outliers

1. Types of Outliers
2. Detecting Outliers
3. Handling Outliers

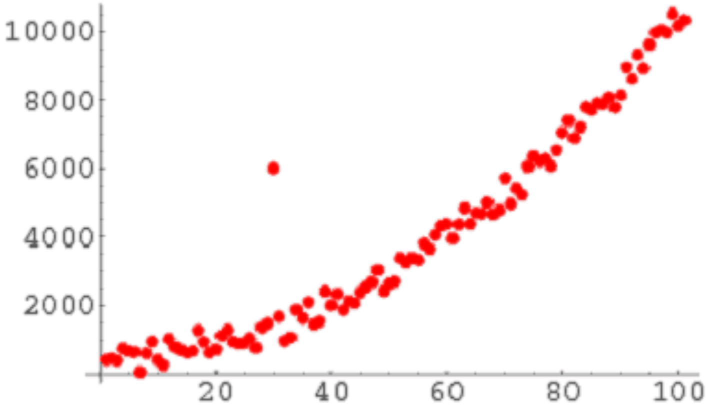
Outliers and Noise

- **Noise** = incorrect values.
  - Causes: many reasons, e.g. human mistakes, the imprecision of a measurement device, …
  - Example: When surveying people about their wealth some will lie
- **Outliers** = values that fall outside the norm in your population
  - These values may be correct or incorrect
  - Example: Bill Gates’ or Jeff Bezos’ wealth are outliers in the wealth of people in the world
- Noise should usually be removed, but is hard/impossible to identify
  - Some noisy data points are Outliers → investigate Outliers to find/remove Noise
  - But not all Outliers are Noise
    - Therefore, we may not want to remove all Outliers
  - Not all noisy data points are Outliers, some are well hidden and pretty much impossible to find
    - Many ML algorithms are resilient towards noise, if it is small and random

Types of Outliers 1: Global Outliers (Point Outliers)

- Global Outlier: individual data point that is anomalous with respect to the rest of the data

- Example:
  - Intrusion detection in computer networks.

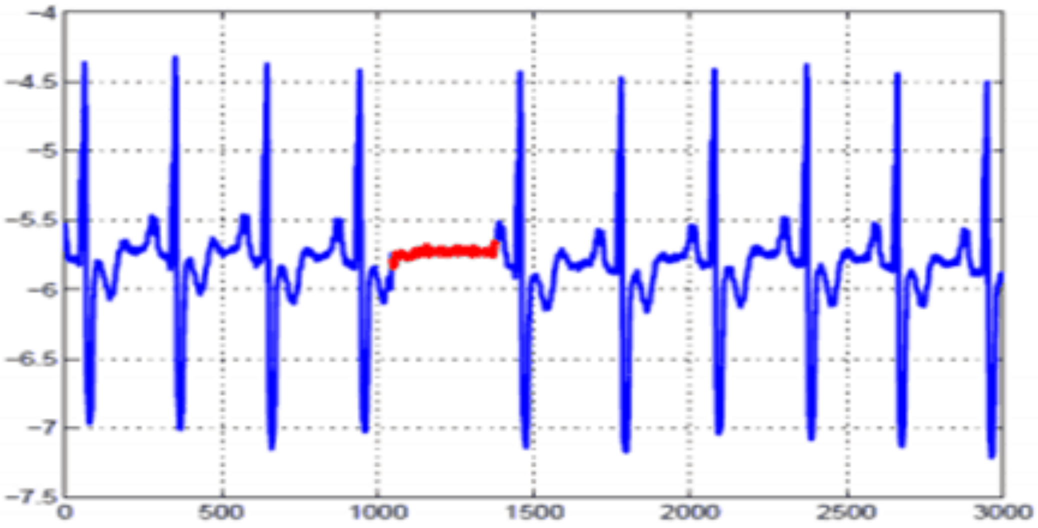


Source: https://www.anblicks.com/blog/an-introduction-to-outliers/)

Types of Outliers 2: Collective Outliers

- Collective Outliers: collection of data points that is anomalous with respect to the entire data set.

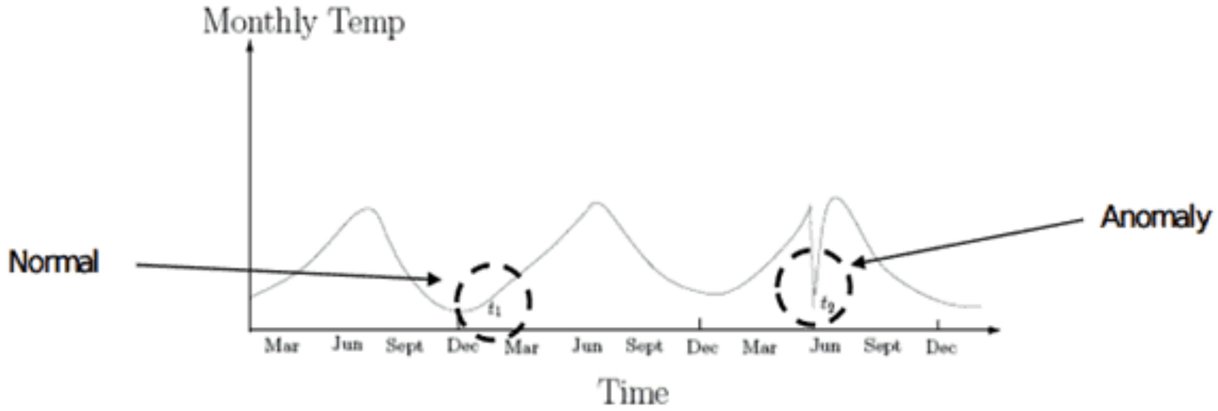
- Example:



Source: https://www.anblicks.com/blog/an-introduction-to-outliers/)

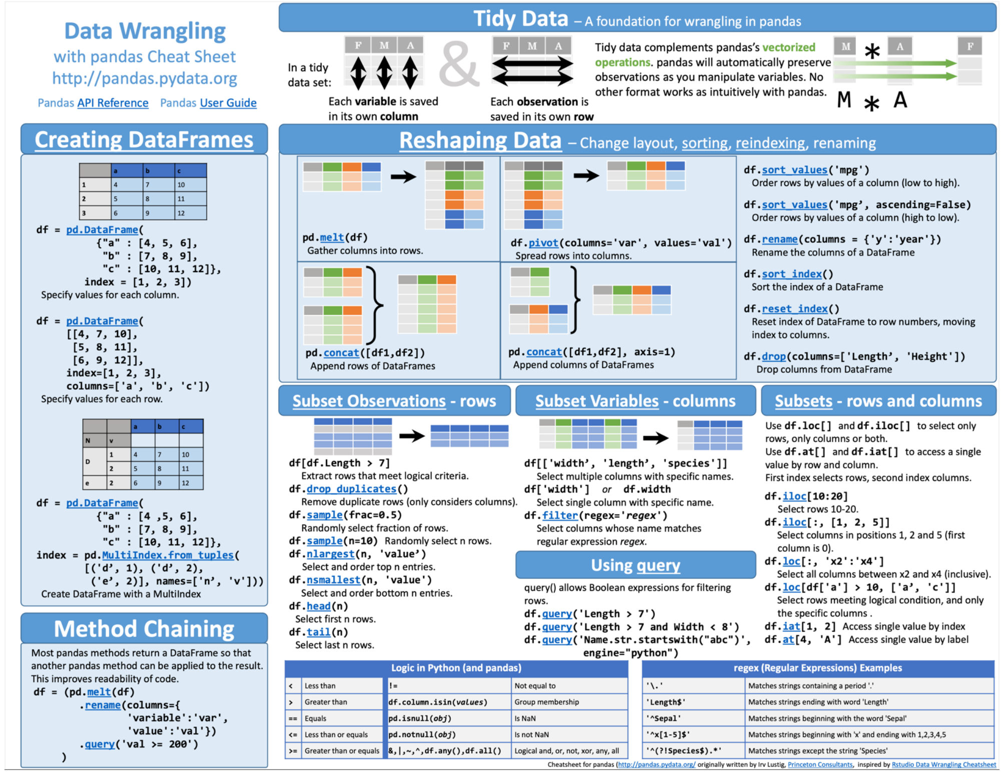
Types of Outliers 2: Contextual Outlier

- Contextual Outlier: data point that is anomalous in a specific context (but not otherwise)
- Attributes of data points should be divided into two groups
  - Contextual attributes: defines the context, e.g., time & location
  - Behavioural attributes: characteristics of the object, used in outlier evaluation, e.g., temperature
- Example:



Source: https://www.anblicks.com/blog/an-introduction-to-outliers/)

Pandas Cheat Sheet: Helpful for the following Exercises



Data Wrangling

with pandas Cheat Sheet
http://pandas.pydata.org

Pandas API Reference   Pandas User Guide

Tidy Data – A foundation for wrangling in pandas

In a tidy data set:

F M A
&
F M A
M * A
F
M * A

Each **variable** is saved in its own **column**

Each **observation** is saved in its own **row**

Tidy data complements pandas's vectorized operations. pandas will automatically preserve observations as you manipulate variables. No other format works as intuitively with pandas.

Creating DataFrames

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

```
df = pd.DataFrame(
        {"a" : [4, 5, 6],
         "b" : [7, 8, 9],
         "c" : [10, 11, 12]},
       index = [1, 2, 3])
```

Specify values for each column.

```
df = pd.DataFrame(
     [[4, 7, 10],
      [5, 8, 11],
      [6, 9, 12]],
     index=[1, 2, 3],
     columns=['a', 'b', 'c'])
```

Specify values for each row.

		a	b	c
N	v			
D	1	4	7	10
	2	5	8	11
e	2	6	9	12

```
df = pd.DataFrame(
        {"a" : [4 ,5, 6],
         "b" : [7, 8, 9],
         "c" : [10, 11, 12]},
index = pd.MultiIndex.from_tuples(
        [('d’, 1), ('d’, 2),
         ('e’, 2)], names=['n’, 'v']))
```

Create DataFrame with a MultiIndex

Method Chaining

Most pandas methods return a DataFrame so that another pandas method can be applied to the result. This improves readability of code.

```
df = (pd.melt(df)
        .rename(columns={
                'variable':'var',
                'value':'val'})
        .query('val >= 200')
     )
```

Reshaping Data – Change layout, sorting, reindexing, renaming

`pd.melt(df)`
Gather columns into rows.

`df.pivot(columns='var', values='val')`
Spread rows into columns.

`pd.concat([df1,df2])`
Append rows of DataFrames

`pd.concat([df1,df2], axis=1)`
Append columns of DataFrames

`df.sort_values('mpg')`
Order rows by values of a column (low to high).

`df.sort_values('mpg’, ascending=False)`
Order rows by values of a column (high to low).

`df.rename(columns = {'y':'year'})`
Rename the columns of a DataFrame

`df.sort_index()`
Sort the index of a DataFrame

`df.reset_index()`
Reset index of DataFrame to row numbers, moving index to columns.

`df.drop(columns=['Length’, 'Height'])`
Drop columns from DataFrame

Subset Observations - rows

`df[df.Length > 7]`
Extract rows that meet logical criteria.

`df.drop_duplicates()`
Remove duplicate rows (only considers columns).

`df.sample(frac=0.5)`
Randomly select fraction of rows.

`df.sample(n=10)` Randomly select n rows.

`df.nlargest(n, 'value’)`
Select and order top n entries.

`df.nsmallest(n, 'value')`
Select and order bottom n entries.

`df.head(n)`
Select first n rows.

`df.tail(n)`
Select last n rows.

Subset Variables - columns

`df[['width’, 'length’, 'species']]`
Select multiple columns with specific names.

`df['width']` *or* `df.width`
Select single column with specific name.

`df.filter(regex='regex')`
Select columns whose name matches regular expression *regex*.

Using query

query() allows Boolean expressions for filtering rows.

```
df.query('Length > 7')
df.query('Length > 7 and Width < 8')
df.query('Name.str.startswith("abc")',
         engine="python")
```

Subsets - rows and columns

Use **df.loc[]** and **df.iloc[]** to select only rows, only columns or both.
Use **df.at[]** and **df.iat[]** to access a single value by row and column.
First index selects rows, second index columns.

`df.iloc[10:20]`
Select rows 10-20.

`df.iloc[:, [1, 2, 5]]`
Select columns in positions 1, 2 and 5 (first column is 0).

`df.loc[:, 'x2':'x4']`
Select all columns between x2 and x4 (inclusive).

`df.loc[df['a'] > 10, ['a’, 'c']]`
Select rows meeting logical condition, and only the specific columns .

`df.iat[1, 2]` Access single value by index

`df.at[4, 'A']` Access single value by label

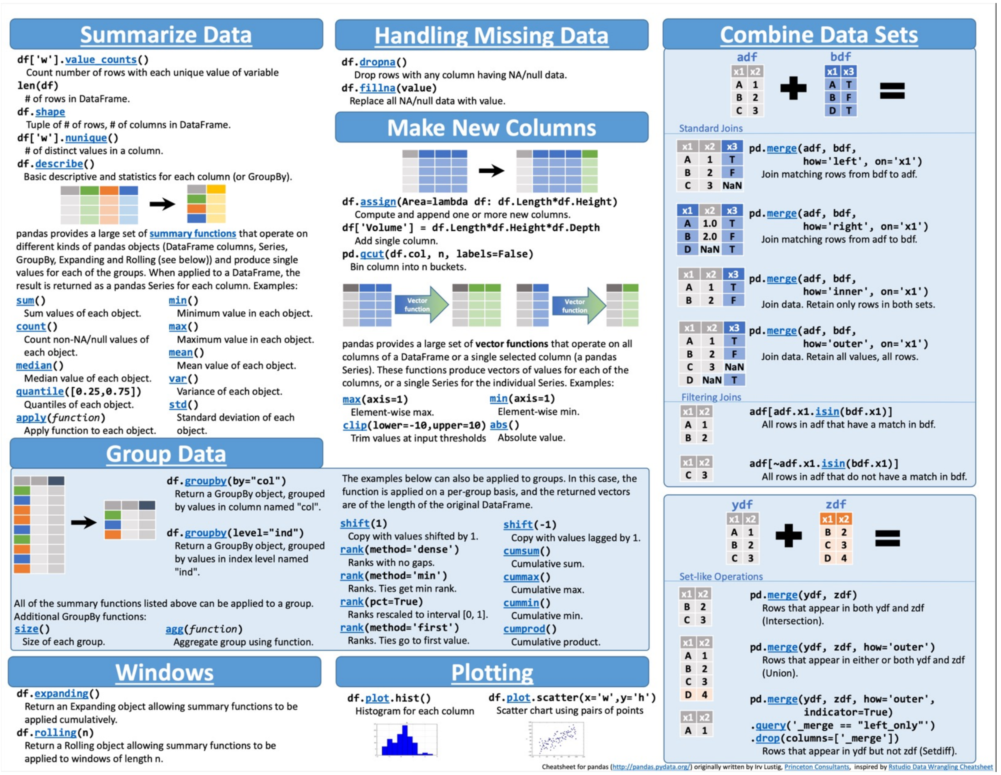
Logic in Python (and pandas)

<	Less than	!=	Not equal to
>	Greater than	df.column.isin(*values*)	Group membership
==	Equals	pd.isnull(*obj*)	Is NaN
<=	Less than or equals	pd.notnull(*obj*)	Is not NaN
>=	Greater than or equals	&,|,~,^,df.any(),df.all()	Logical and, or, not, xor, any, all

regex (Regular Expressions) Examples

'\.'	Matches strings containing a period '.'
'Length$'	Matches strings ending with word 'Length'
'^Sepal'	Matches strings beginning with the word 'Sepal'
'^x[1-5]$'	Matches strings beginning with 'x' and ending with 1,2,3,4,5
'^(?!Species$).*'	Matches strings except the string 'Species'

Cheatsheet for pandas (http://pandas.pydata.org/ originally written by Irv Lustig, Princeton Consultants, inspired by Rstudio Data Wrangling Cheatsheet



Summarize Data

`df['w'].value_counts()`
Count number of rows with each unique value of variable

`len(df)`
# of rows in DataFrame.

`df.shape`
Tuple of # of rows, # of columns in DataFrame.

`df['w'].nunique()`
# of distinct values in a column.

`df.describe()`
Basic descriptive and statistics for each column (or GroupBy).

pandas provides a large set of summary functions that operate on different kinds of pandas objects (DataFrame columns, Series, GroupBy, Expanding and Rolling (see below)) and produce single values for each of the groups. When applied to a DataFrame, the result is returned as a pandas Series for each column. Examples:

`sum()`
Sum values of each object.

`count()`
Count non-NA/null values of each object.

`median()`
Median value of each object.

`quantile([0.25,0.75])`
Quantiles of each object.

`apply(function)`
Apply function to each object.

`min()`
Minimum value in each object.

`max()`
Maximum value in each object.

`mean()`
Mean value of each object.

`var()`
Variance of each object.

`std()`
Standard deviation of each object.

Handling Missing Data

`df.dropna()`
Drop rows with any column having NA/null data.

`df.fillna(value)`
Replace all NA/null data with value.

Make New Columns

`df.assign(Area=lambda df: df.Length*df.Height)`
Compute and append one or more new columns.

`df['Volume'] = df.Length*df.Height*df.Depth`
Add single column.

`pd.qcut(df.col, n, labels=False)`
Bin column into n buckets.

pandas provides a large set of **vector functions** that operate on all columns of a DataFrame or a single selected column (a pandas Series). These functions produce vectors of values for each of the columns, or a single Series for the individual Series. Examples:

`max(axis=1)`
Element-wise max.

`min(axis=1)`
Element-wise min.

`clip(lower=-10,upper=10)`
Trim values at input thresholds

`abs()`
Absolute value.

Group Data

`df.groupby(by="col")`
Return a GroupBy object, grouped by values in column named "col".

`df.groupby(level="ind")`
Return a GroupBy object, grouped by values in index level named "ind".

All of the summary functions listed above can be applied to a group. Additional GroupBy functions:

`size()`
Size of each group.

`agg(function)`
Aggregate group using function.

The examples below can also be applied to groups. In this case, the function is applied on a per-group basis, and the returned vectors are of the length of the original DataFrame.

`shift(1)`
Copy with values shifted by 1.

`rank(method='dense')`
Ranks with no gaps.

`rank(method='min')`
Ranks. Ties get min rank.

`rank(pct=True)`
Ranks rescaled to interval [0, 1].

`rank(method='first')`
Ranks. Ties go to first value.

`shift(-1)`
Copy with values lagged by 1.

`cumsum()`
Cumulative sum.

`cummax()`
Cumulative max.

`cummin()`
Cumulative min.

`cumprod()`
Cumulative product.

Windows

`df.expanding()`
Return an Expanding object allowing summary functions to be applied cumulatively.

`df.rolling(n)`
Return a Rolling object allowing summary functions to be applied to windows of length n.

Plotting

`df.plot.hist()`
Histogram for each column

`df.plot.scatter(x='w',y='h')`
Scatter chart using pairs of points

Combine Data Sets

adf

x1	x2
A	1
B	2
C	3

bdf

x1	x3
A	T
B	F
D	T

Standard Joins

x1	x2	x3
A	1	T
B	2	F
C	3	NaN

```
pd.merge(adf, bdf,
         how='left', on='x1')
```

Join matching rows from bdf to adf.

x1	x2	x3
A	1.0	T
B	2.0	F
D	NaN	T

```
pd.merge(adf, bdf,
         how='right', on='x1')
```

Join matching rows from adf to bdf.

x1	x2	x3
A	1	T
B	2	F

```
pd.merge(adf, bdf,
         how='inner', on='x1')
```

Join data. Retain only rows in both sets.

x1	x2	x3
A	1	T
B	2	F
C	3	NaN
D	NaN	T

```
pd.merge(adf, bdf,
         how='outer', on='x1')
```

Join data. Retain all values, all rows.

Filtering Joins

x1	x2
A	1
B	2

`adf[adf.x1.isin(bdf.x1)]`
All rows in adf that have a match in bdf.

x1	x2
C	3

`adf[~adf.x1.isin(bdf.x1)]`
All rows in adf that do not have a match in bdf.

ydf

x1	x2
A	1
B	2
C	3

zdf

x1	x2
B	2
C	3
D	4

Set-like Operations

x1	x2
B	2
C	3

`pd.merge(ydf, zdf)`
Rows that appear in both ydf and zdf (Intersection).

x1	x2
A	1
B	2
C	3
D	4

`pd.merge(ydf, zdf, how='outer')`
Rows that appear in either or both ydf and zdf (Union).

x1	x2
A	1

```
pd.merge(ydf, zdf, how='outer',
         indicator=True)
.query('_merge == "left_only"')
.drop(columns=['_merge'])
```

Rows that appear in ydf but not zdf (Setdiff).

Cheatsheet for pandas (http://pandas.pydata.org/) originally written by Irv Lustig, Princeton Consultants, inspired by Rstudio Data Wrangling Cheatsheet

Exercise

Exercise 1

Iris Data – Loading, Munging, Missing Values

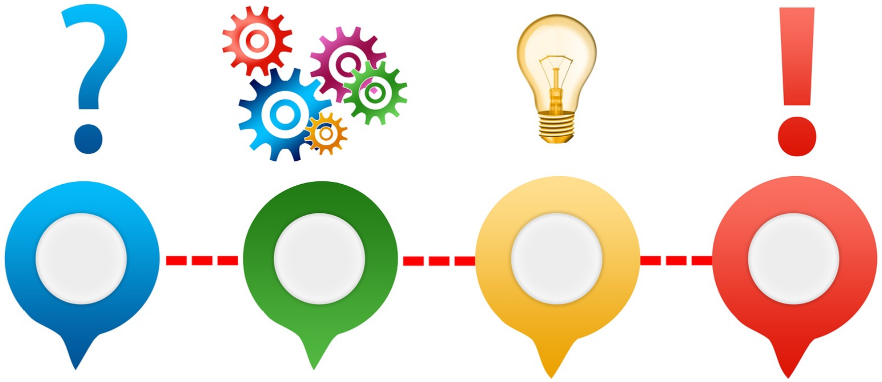


Photo by Gerd Altmann on Pixabay

Data Science

Outliers

1. Types of Outliers

2. Detecting Outliers

3. Handling Outliers

Identifying Global Outliers mathematically

- Mathematical Identification of Outliers

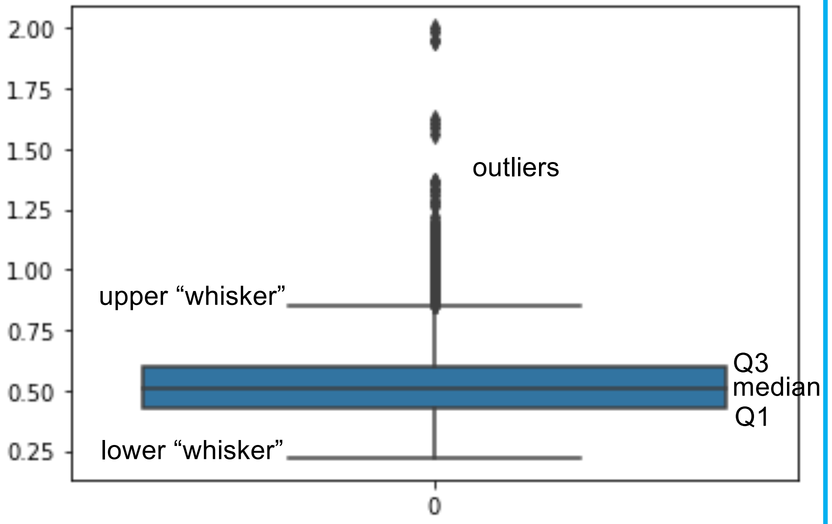
  - Z-Score
    - Definition of Outliers: points that are more than 3 times the standard deviation from the mean
    - Computed directly or via z-score scaling

  - IQR-Score
    - IQR (interquartile range) =
      distance between Q1 = 25%-percentile (=lower quartile) and Q3 = 75%-percentile (=upper quartile)
    - Definition of Outliers: points that are more than1.5*IQR below Q1 or more than 1.5*IQR above Q3
    - Box Plot = visual way to show the IQR values

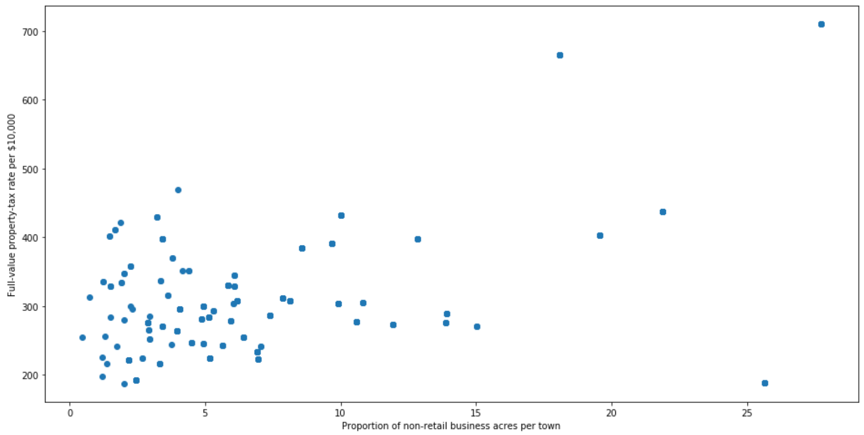
Source: https://towardsdatascience.com/ways-to-detect-and-remove-the-outliers-404d16608dba

Identifying Global Outliers visually

- Visual Identification of Outliers



Box Plot: Single Variable (univariate)

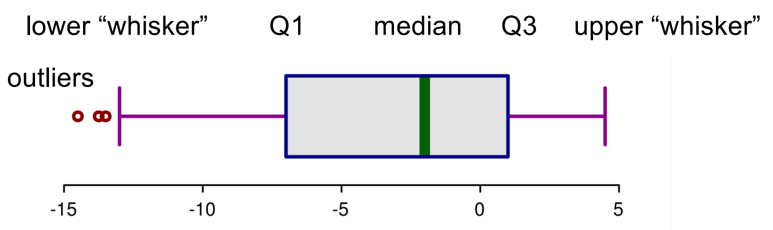


Scatter Plot: Two Variables (multivariate)

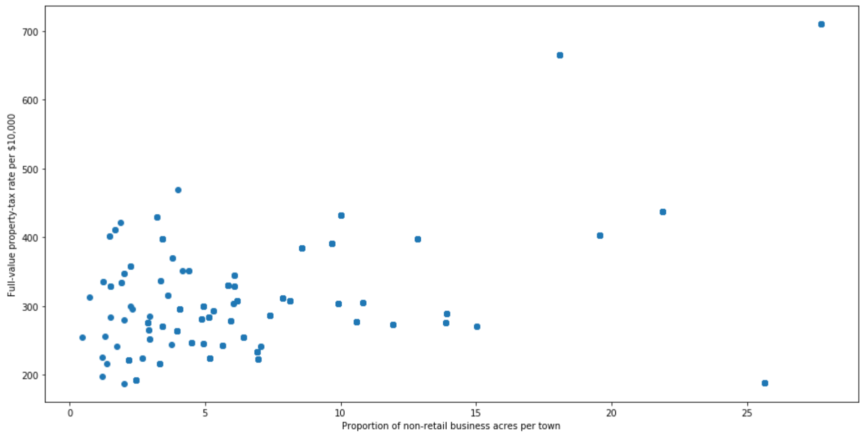
Source: https://towardsdatascience.com/ways-to-detect-and-remove-the-outliers-404d16608dba

Identifying Global Outliers visually

- Visual Identification of Outliers



Box Plot: Single Variable (univariate)



Scatter Plot: Two Variables (multivariate)

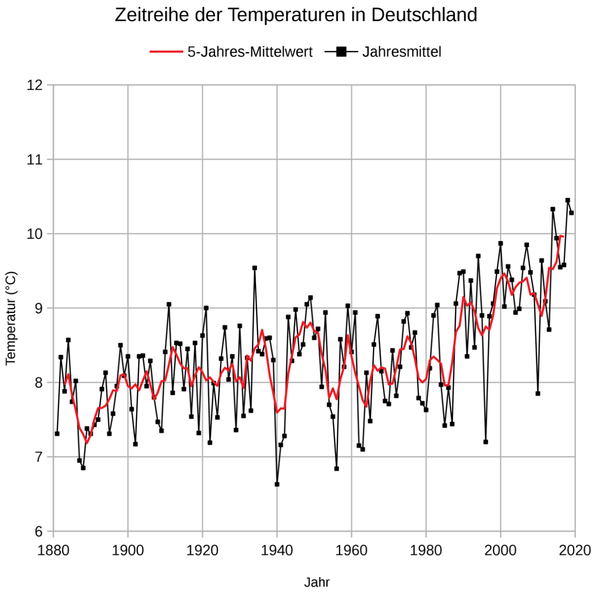
Source: https://towardsdatascience.com/ways-to-detect-and-remove-the-outliers-404d16608dba

Identifying Collective Outliers

- Much harder than Global Outlier Identification
- Often non-noise, but real “signal” – in this case, opportunity to find something important
- Some Clustering Algorithms can be used (e.g. EM-Clustering, DBSCAN)
- Specific Outlier/Anomaly Detection Algorithms exist (e.g. Isolation Forests)

Identifying Contextual Outliers

- Much harder than Global Outlier Identification
- Often important for time-series-data
- May have to model both seasonality and trend (manually or automatically)
  - Example: a temperature of 25° Celcius at noon in Rosenheim
    - Not normal in December 2020 (outlier)
    - normal in August 2020 (no outlier – due to seasonality)
    - Not normal in December 2021 (outlier)
    - normal in December 2040 (no outlier – due to trend)
    - Not normal in August 2040 (outlier – due to trend and seasonality)



Source: https://de.wikipedia.org/wiki/Folgen_der_globalen_Erw%C3%A4rmung_in_Deutschland

Data Science

Outliers

1. Types of Outliers
2. Detecting Outliers
3. Handling Outliers

Handling Outliers

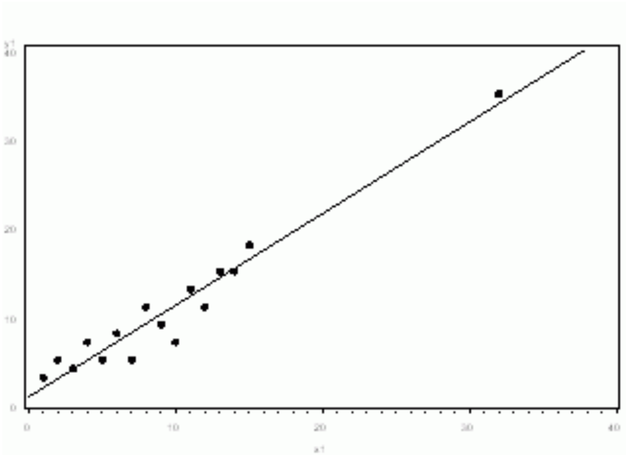
- Three Options
  - Keep Outlier
  - Remove Outlier
  - Correct Outlier (→ cf. Imputation of Missing Values)

- If the Outlier is Noise: Remove Outlier

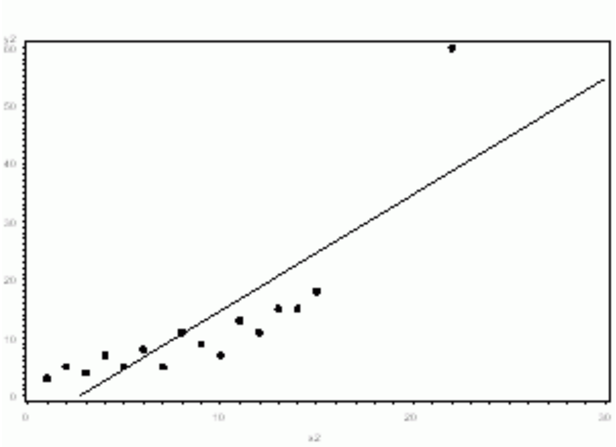
- If the Outlier is not Noise, or if we are unsure: Run analysis with and without Outlier
  - If it does not change the result and assumptions: may remove Outlier and note this
  - If it does change the result, but not the assumptions: show both results and discuss changes
  - It if creates results: drop, note this, and show results without Outlier
    (and maybe investigate this Outlier further)

Example

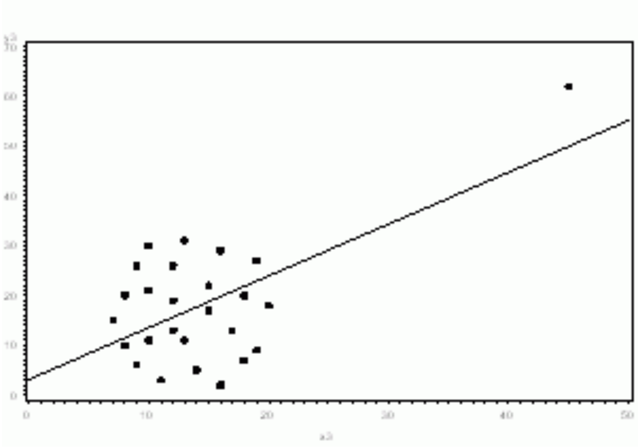
- Regression Analysis with an Outlier for which we are unsure if it is Noise



(a) Outlier does not change the result and assumptions: may remove Outlier and note this



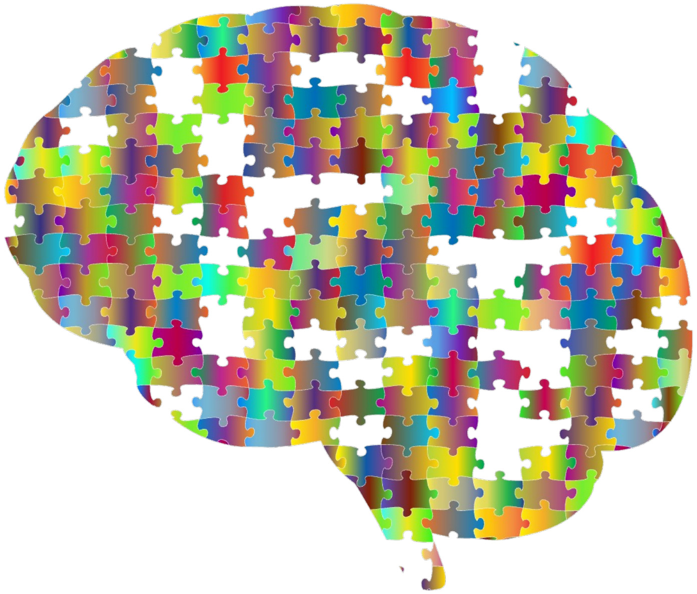
(b) Outlier does change the result, but not the assumptions: show both results and discuss changes



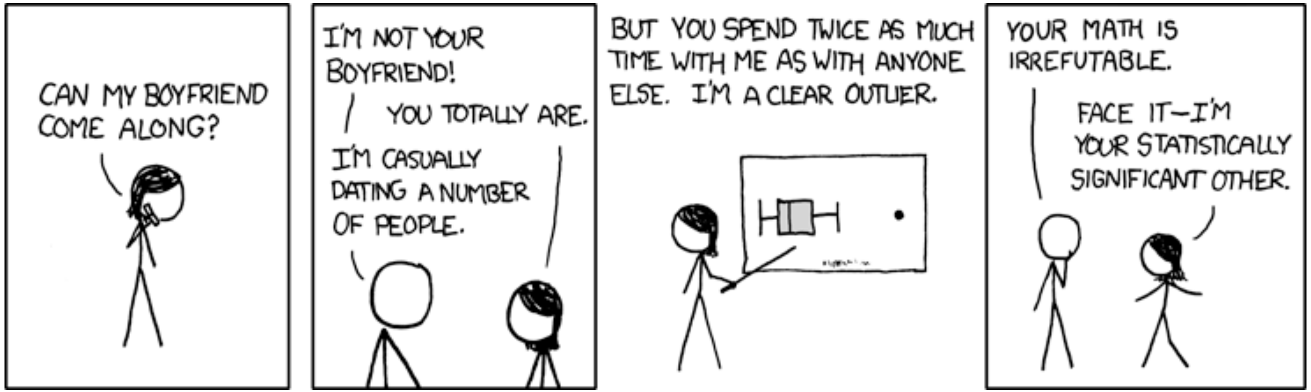
(c) Outlier creates the result: drop, note this, and show result without Outlier (and maybe investigate this Outlier further)

Source: https://www.theanalysisfactor.com/outliers-to-drop-or-not-to-drop/

Key Takeaways



- Noise and Outliers
- Types of Outliers – Global, Collective, Contextual Outliers
- Detecting Outliers
  - Visually (box plots, scatter plots)
  - Mathematically (z-score, IQR score)
- Handling Outliers


CAN MY BOYFRIEND COME ALONG?
I'M NOT YOUR BOYFRIEND!
YOU TOTALLY ARE.
I'M CASUALLY DATING A NUMBER OF PEOPLE.
BUT YOU SPEND TWICE AS MUCH TIME WITH ME AS WITH ANYONE ELSE. I'M A CLEAR OUTLIER.
YOUR MATH IS IRREFUTABLE.
FACE IT—I'M YOUR STATISTICALLY SIGNIFICANT OTHER.

Source: https://imgs.xkcd.com/comics/boyfriend.png

Image by Gordon Johnson on pixabay

Exercise

Exercise 2

Iris Data – Merging, Outlier Detection & Handling

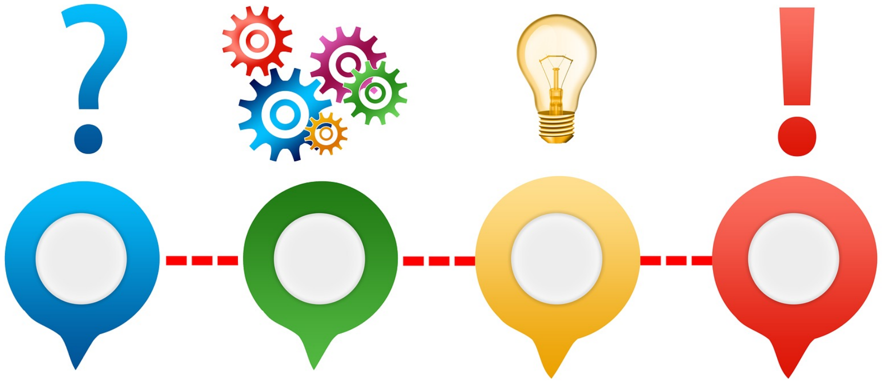


Photo by Gerd Altmann on Pixabay