TechTalent Academy
Safeguarding Policy

*“Protecting an adult’s right to live in **safety, free from abuse and neglect**. It is about people and organisations working together to **prevent and stop both the risks and experience of abuse or neglect**, while at the same time making sure that the **adult’s wellbeing is promoted** including, where appropriate, having regard to their views, wishes, feelings and beliefs in deciding on any action. This must recognise that adults sometimes have complex interpersonal relationships and may be ambivalent, unclear or unrealistic about their personal circumstances.”*

If you have a safeguarding concern, please raise this with your tutor or via the safeguarding link on our website:

https://www.techtalent.co.uk/safeguarding-statement

TechTalent’s safeguarding lead is: **Max Ruddock**

Starter Activity.

Take a quick look at the documentation for the Pandas library

https://pandas.pydata.org/docs/user_guide/10min.html

How can we use this library for data analysis?

techtalent academy

TechTalent Academy

Data Science Course

Pandas

Lesson Objectives.

- What is Pandas?
- Importing Data
- Data Analysis Workflow

What is Pandas?

techtalent academy

Software library for use with Python

Ideal for working with datasets

Library facilitates data manipulation, visualisation and analysis

Created by software developer Wes McKinney in 2008

Why use Pandas?

techtalent academy

You can import, analyse and visualise data easier

Easy for handling large amount of data

Key concepts of Pandas are indexing and dataframes

pandas

Data analysis workflow.

techtalent academy

Data Preparation

- Data acquisition & Importation
- Data Transformation and Cleaning

Data Analysis

- Visualise & Analyse (compare, interpret, predict)
- Ask & Answer questions

Presenting/sharing

- Make reports/dashboard to communicate data

How to Import Pandas.

Pip install pandas

import pandas as pd

You will need the above statement to import the library and by using np you are saying from now on it will be known as np (quicker to type pd than Pandas)

Official documentation: https://pandas.pydata.org/docs/user_guide/index.html techtalent academy

Pandas data structure.

Two types of pandas data structure:

Series
(1D like array)

rows index

	Country
0	United Kingdom
1	France
2	Mexico
3	Canada

Command to create a Series:
pd.Series( )

Dataframe
(2D like array)

columns index

rows index

	0	1
	Country	**Population**
0	United Kingdom	68521968
1	France	65273511
2	Mexico	128932753
3	Canada	37742154

Command to create a dataframe:
pd.DataFrame( )

Data Importation.

Commands to read a CSV or Excel file:

pd.read_csv()
pd.read_excel()

Steps:

1- Place your Jupyter notebook script in the **same folder** as your dataset

2- Create a variable and store the appropriate pandas function to read your dataset:

dataframe= pd.read_csv (‘file_name’)

3-Call your variable to display it: **dataframe**

Data Importation.

techtalent academy

Importing CSV File example:

```
dataframe= pd.read_csv("datacensus.csv")
dataframe
```

✓ 0.1s

With CSV files, you must saved the CSV file in the same folder as your Python script!

output

	Country	Population
0	United Kingdom	68521968.0
1	France	65273511.0
2	Mexico	128932753.0
3	Canada	37742154.0
4	Peru	NaN

Missing value:
Not A Number

Data Importation.

techtalent academy

Importing URL example:

```
import pandas as pd

url = "https://raw.githubusercontent.com/MicrosoftDocs/ml-basics/ee7bccccf5dd1a95f9d547b2e9e5fd68f61fe02e/challenges/data/flights.csv"
df_flights = pd.read_csv(url)
df_flights.head()
```

output

Year	Month	DayofMonth	DayOfWeek	Carrier	OriginAirportID	OriginAirportName	OriginCity	OriginState	DestAirportID	DestAirportName	DestCity	DestState
2013	9	16	1	DL	15304	Tampa International	Tampa	FL	12478	John F. Kennedy International	New York	NY
2013	9	23	1	WN	14122	Pittsburgh International	Pittsburgh	PA	13232	Chicago Midway International	Chicago	IL
2013	9	7	6	AS	14747	Seattle/Tacoma International	Seattle	WA	11278	Ronald Reagan Washington National	Washington	DC
2013	7	22	1	OO	13930	Chicago O'Hare International	Chicago	IL	11042	Cleveland-Hopkins International	Cleveland	OH
2013	5	16	4	DL	13931	Norfolk International	Norfolk	VA	10397	Hartsfield-Jackson Atlanta International	Atlanta	GA

Data Exploration.

techtalent academy

.head() method

Explore the 5 first rows:
df_flights.head()

	Year	Month	DayofMonth	DayOfWeek	Carrier	OriginAirportID	OriginAirportName
0	2013	9	16	1	DL	15304	Tampa International
1	2013	9	23	1	WN	14122	Pittsburgh International
2	2013	9	7	6	AS	14747	Seattle/Tacoma International
3	2013	7	22	1	OO	13930	Chicago O'Hare International
4	2013	5	16	4	DL	13931	Norfolk International

.tail() method

Explore the 5 last rows:
df_flights.tail()

	Year	Month	DayofMonth	DayOfWeek	Carrier	OriginAirportID	OriginAirportName
271935	2013	9	20	5	VX	13204	Orlando International
271936	2013	4	19	5	FL	10397	Hartsfield-Jackson Atlanta International
271937	2013	10	26	6	WN	12191	William P Hobby
271938	2013	5	7	2	HA	13830	Kahului Airport
271939	2013	6	11	2	UA	14771	San Francisco International

The attribute **shape** give the total numbers of rows and columns:
df_flights.shape

output

⬇

(271940, 2)
271940 rows and 2 columns

Try:
df_flights.head (10)
df_flights.tail (22)

Data Exploration.

techtalent academy

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271940 entries, 0 to 271939
Data columns (total 20 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   Year               271940 non-null  int64
 1   Month              271940 non-null  int64
 2   DayofMonth         271940 non-null  int64
 3   DayOfWeek          271940 non-null  int64
 4   Carrier            271940 non-null  object
 5   OriginAirportID    271940 non-null  int64
 6   OriginAirportName  271940 non-null  object
 7   OriginCity         271940 non-null  object
 8   OriginState        271940 non-null  object
 9   DestAirportID      271940 non-null  int64
 10  DestAirportName    271940 non-null  object
 11  DestCity           271940 non-null  object
 12  DestState          271940 non-null  object
 13  CRSDepTime         271940 non-null  int64
 14  DepDelay           271940 non-null  int64
 15  DepDel15           269179 non-null  float64
 16  CRSArrTime         271940 non-null  int64
 17  ArrDelay           271940 non-null  int64
 18  ArrDel15           271940 non-null  int64
 19  Cancelled          271940 non-null  int64
dtypes: float64(1), int64(12), object(7)
memory usage: 41.5+ MB
```

Get a quick summary of the dataframe with the **.info() method** (i.e. # of columns and rows, data type, missing values #):
df_flights.info()

Here we can see that this columns contains some missing value's. We will come back to this later.

Data Selection (columns).

techtalent academy

Select one column
by column name using double backets [[ ]]:
df_flights[['Year']]

The new column can be stored in a new variable:
year=df_flights[['Year']]

	Year	OriginCity
0	2013	Tampa
1	2013	Pittsburgh
2	2013	Seattle
3	2013	Chicago
4	2013	Norfolk

Select multiple columns:
df_flights[['Year', 'OriginCity']]

It is possible to select data with one pair of [ ], but python will return a Series object not a dataframe: try **df_flights[['Year']]**

0	2013	Tampa
1	2013	Pittsburgh
2	2013	Seattle
3	2013	Chicago
4	2013	Norfolk

The method **type()** gives the data type

```
2 type(df_flights['Year'])
✓ 0.7s
pandas.core.series.Series
```

Data Selection (rows).

techtalent academy

.loc and .iloc commands

dataframe.loc[0, 'Country']

dataframe.iloc[2,1]

	Country	Population
0	United Kingdom	68521968
1	France	65273511
2	Mexico	128932753
3	Canada	37742154
4	Peru	32971854

.loc
(primarily label based)
loc[row label, column label]

.iloc
(integer based)
iloc[row position, column position]

Data Selection (rows cont.).

techtalent academy

.loc and .iloc commands

	Year	OriginCity
0	2013	Tampa
1	2013	Pittsburgh
2	2013	Seattle
3	2013	Chicago
4	2013	Norfolk

df_flights.loc[0]
OR
dataframe.iloc[0]

df_flights.loc[2:7]

dataframe.iloc[2:7]
It selects the first row index to n-1

It is possible to **select/slice** a part of the dataframe using a colon or/and a comma

Try:
df_flights.iloc[0:3]
df_flights.iloc[0:3, 0:1]

Data Selection (pattern).

techtalent academy

Searching a particular string pattern: .str.contains() method

df_flights['Country'].str.contains('France')

1. The function evaluate each rows on the Country column for the presence of the string **'**Norfolk**'**. If there is no match, it returns **False**, if there is a match it returns **True**

2. The function returns a pandas series object of Boolean values

3. Selecting the previous command with **df_flights []** will return all data related to the string 'Norfolk'

1

	Year	OriginCity
0	2013	Tampa
1	2013	Pittsburgh
2	2013	Seattle
3	2013	Chicago
4	2013	Norfolk

2

```
0    False
1    False
2    False
3    False
4     True
```

3

Carrier	OriginAirportID	OriginAirportName	OriginCity
DL	13931	Norfolk International	Norfolk
WN	13931	Norfolk International	Norfolk
EV	13931	Norfolk International	Norfolk
US	13931	Norfolk International	Norfolk
EV	13931	Norfolk International	Norfolk

VS Code:

```
df_flights[df_flights['OriginCity'].str.contains('Norfolk')]
```

Data Cleaning.

techtalent academy

Checks:

Remove missing values (NaN) from the dataset

Value replacement: perform the average of other values

Check info(): it will give you the count of non null values

Data uniformity: change type of data/rename variables names

Detect missing values: isnull() function

Drop columns with drop() function

Transform your numbers: absolute number

Remove outliers (can be seen when plotting the data)

lead to make proper analysis

Detecting Missing Data.

techtalent academy

We need to find out how many (if any) missing values we have in our dataset. We can do this by using `df_flights.isnull().sum()`

```
Year                 0
Month                0
DayofMonth           0
DayOfWeek            0
Carrier              0
OriginAirportID      0
OriginAirportName    0
OriginCity           0
OriginState          0
DestAirportID        0
DestAirportName      0
DestCity             0
DestState            0
CRSDepTime           0
DepDelay             0
DepDel15          2761
CRSArrTime           0
ArrDelay             0
ArrDel15             0
Cancelled            0
dtype: int64
```

Sum of missing values

Handling Missing Data.

Depending on the context of your data, you might want to replace missing values by “zero” or leave them. By comparing the DepDelay and DepDel15 columns, we can see they all have a delay of 0.

```
df_flights[df_flights.isnull().any(axis=1)][['DepDelay','DepDel15']]
```

	DepDelay	DepDel15
171	0	NaN
359	0	NaN
429	0	NaN
545	0	NaN
554	0	NaN
...	...	...
271410	0	NaN
271607	0	NaN
271634	0	NaN
271671	0	NaN
271885	0	NaN

2761 rows × 2 columns

```
count    2761.0
mean        0.0
std         0.0
min         0.0
25%         0.0
50%         0.0
75%         0.0
max         0.0
Name: DepDelay, dtype: float64
```

```
df_flights[df_flights.isnull().any(axis=1)].DepDelay.describe()
```

The summary statistics also confirms this. Therefore we can replace of the NaN values with 0.

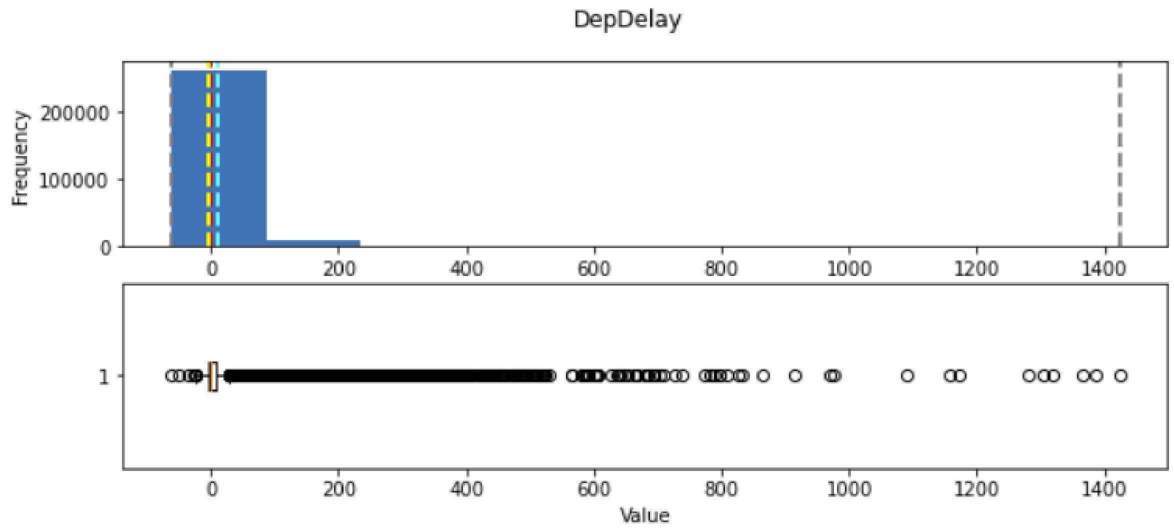
df_flights.DepDel15 = df_flights.DepDel15.fillna(0)

What Are Outliers?

In simple terms, an outlier is an extremely high or extremely low data point relative to the nearest data point and the rest of the neighbouring values in a data graph or dataset.

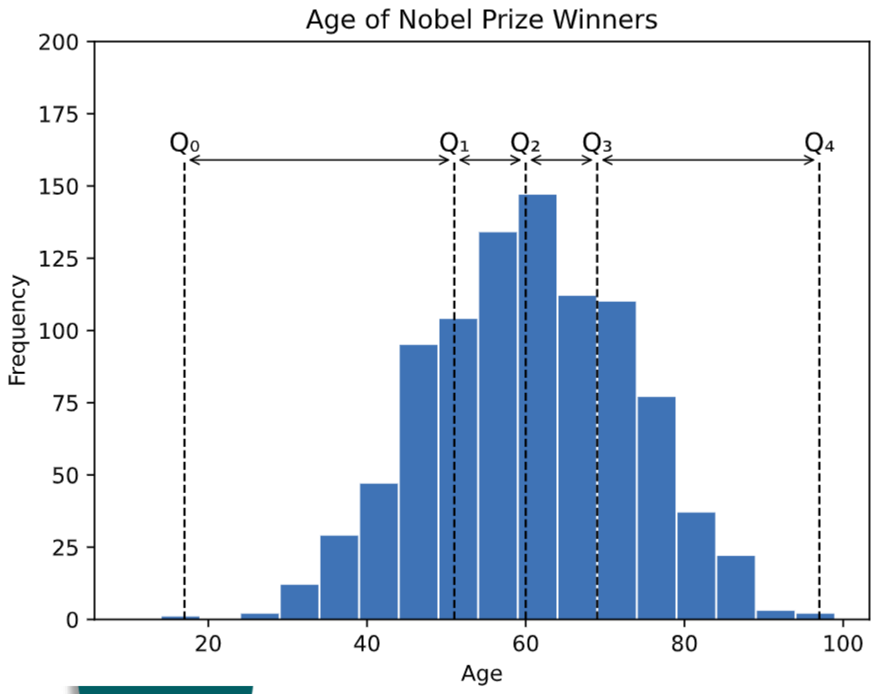
Outliers are extreme values that stand out greatly from the overall pattern of values in a dataset or graph.

Creating visualisations with our dataset can help us to identify outliers.



Quartiles.

Quartiles are values that separate the data into four equal parts. They can help you understand your dataset's central tendency and variability and even help you find outliers. The quartiles (Q_0,Q_1,Q_2,Q_3,Q_4) are the values that separate each quarter.



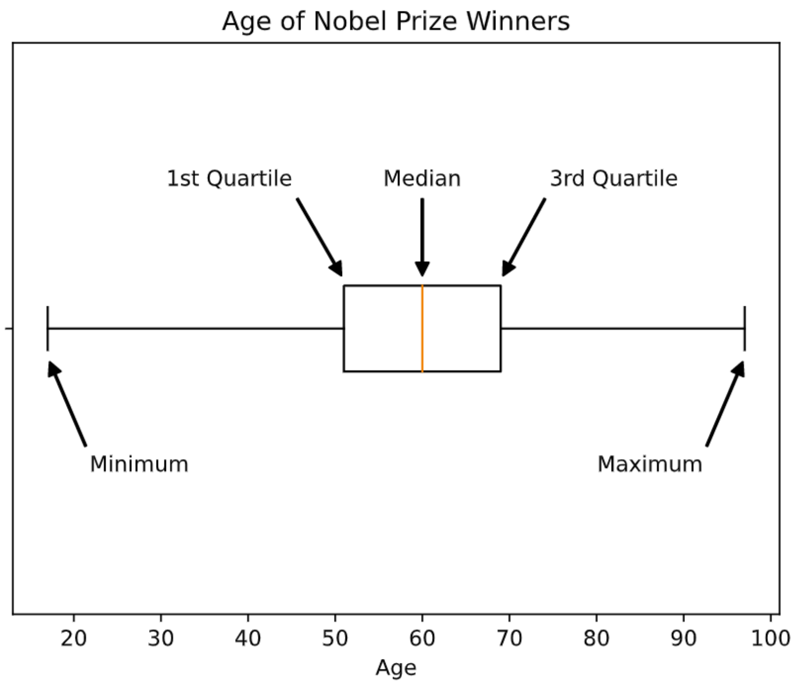
Between Q_0 and Q_1 are the 25% lowest values in the data. Between Q_1 and Q_2 are the next 25%. And so on.

- Q_0 is the smallest value in the data.
- Q_1 is the value separating the first quarter from the second quarter of the data.
- Q_2 is the middle value (median), separating the bottom from the top half.
- Q_3 is the value separating the third quarter from the fourth quarter
- Q_4 is the largest value in the data.

Box Plots.

Box plots are used to show distributions of numeric data values.

They are built to provide high-level information at a glance, offering general information about a group of data's symmetry, skew, variance, and outliers.

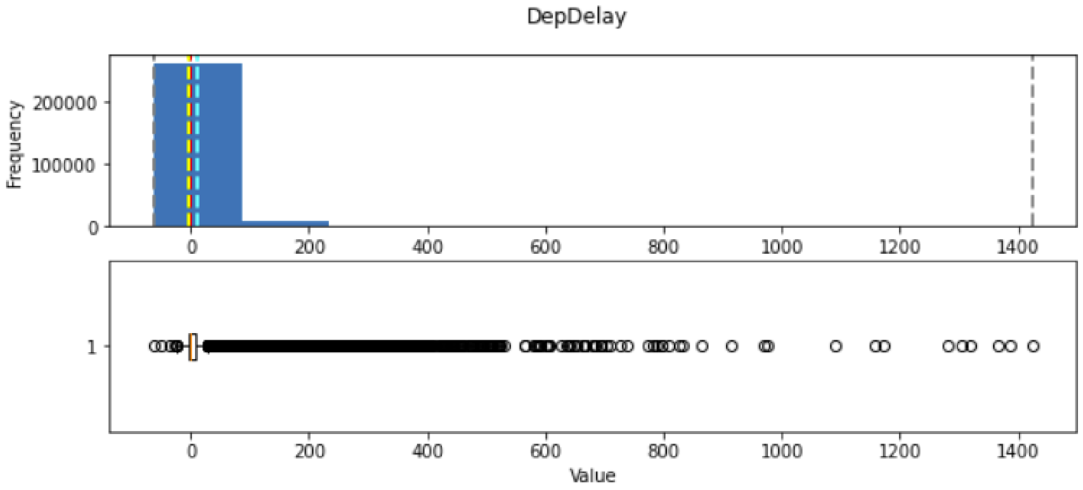


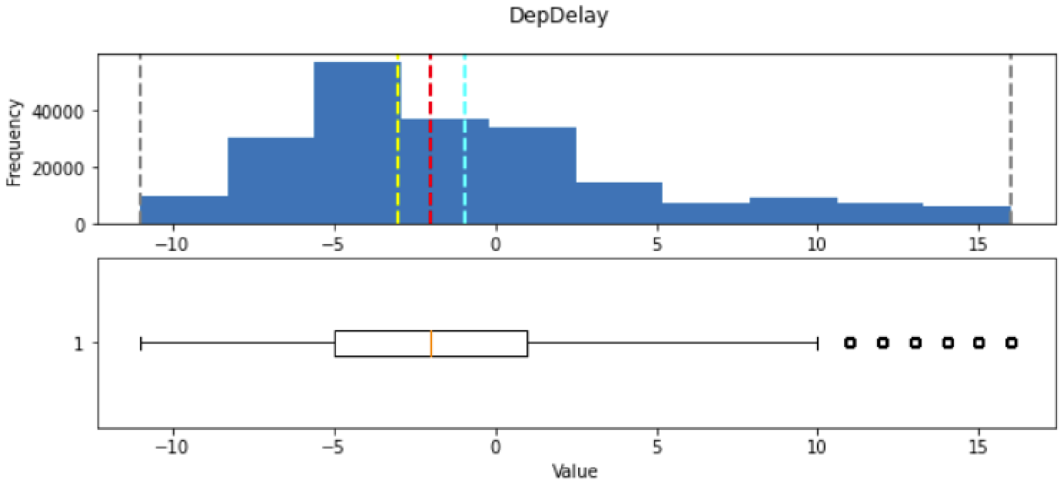
- The **median** is the red line through the middle of the 'box'.
- The left side of the box is the 1st **quartile**. This is the value that separates the first **quarter**, or 25% of the data, from the rest.
- The right side of the box is the 3rd **quartile**. This is the value that separates the first three **quarters**, or 75% of the data, from the rest.
- The distance between the sides of the box is called the **inter-quartile range (IQR)**. This tells us where the 'middle half' of the values are.
- The ends of the lines from the box at the left and the right are the minimum and maximum values in the data. The distance between these is called the **range**.

Cleaning Outliers.

Going back to our dataset. We can see there are a outliers at the lower and upper ends of both variables - particularly at the upper end.

Depending on the context of the dataset, you may want to leave them in or take them out. Let's trim this data so that we include only rows where the values for these fields are within the 1st and 90th percentile.





Asking Questions.

Now that we have cleaned the dataset, we can start to ask and answer questions for data analysis.

- How do the carriers compare in terms of arrival delay performance?
- Are some days of the week more prone to arrival days than others?
- Which departure airport has the highest average departure delay?
- Do late departures tend to result in longer arrival delays than on-time departures?
- Which route (from origin airport to destination airport) has the most late arrivals?
- Which route has the highest average arrival delay?

Data Correlation.

techtalent academy

The corr() method calculates the relationship between each column in your data set.

The correlation statistic is a value between -1 and 1 that indicates the strength of a relationship.
Values above 0 indicate a positive correlation (high values of one variable tend to coincide with high values of the other), while values below 0 indicate a negative correlation (high values of one variable tend to coincide with low values of the other).

For Example:

0.9 is also a good relationship, and if you increase one value, the other will probably increase as well.

-0.9 would be just as good relationship as 0.9, but if you increase one value, the other will probably go down.

0.2 means NOT a good relationship, meaning that if one value goes up does not mean that the other will.

Plenary.

Insert the correct syntax for returning the headers and the first 10 rows of a DataFrame.

```
df.
```

Insert the correct syntax for removing rows with empty cells.

```
df.[        ]()
```

Insert the correct syntax for replacing empty cells with the value "130".

```
df.[        ](130)
```