Introduction to Data Science

Python/pandas commands used

The following commands are used in the lessons. The commands can be copied from this document (ctrl-C) and pasted into your code (ctrl-V).

Lesson 2: Pre-processing and cleaning data

Adding modules

```
import pandas as pd

import matplotlib.pyplot as plt
```

Importing a dataset from a csv file

```
heathrow_2015_data = pd.read_csv("../input/ldsedexcel/heathrow-2015.csv")
```

Displaying the head or tail of the dataset and the shape (number of rows and columns)

```
heathrow_2015_data.head()

heathrow_2015_data.head(10)

heathrow_2015_data.tail()

heathrow_2015_data.shape
```

Displaying the names of fields and data types

This command is particularly useful as the exact field names can be copied into other commands.

```
heathrow_2015_data.dtypes
```

Displaying the basic statistics for a field

```
heathrow_2015_data['Daily Mean Temperature'].describe()
```

Counting the occurrences in a categorical field

```
heathrow_2015_data['Mean Cardinal Direction'].value_counts()
```

Displaying more than one output from a code box

The default is for code boxes will display a single output. To display multiple outputs use `print()`.

```
print(heathrow_2015_data['Daily Mean Temperature'].describe())
print(heathrow_2015_data['Mean Cardinal Direction'].value_counts())
```

Displaying a boxplot

```
heathrow_2015_data.boxplot(column = ['Daily Mean Temperature'])
plt.show()
```

Cleaning data: replacing a value

```
heathrow_2015_data['Daily Total Rainfall'] = heathrow_2015_data['Daily Total Rainfall'].replace({'tr': 0.025})
```

Cleaning data: changing the data type to float

```
heathrow_2015_data['Daily Total Rainfall'] = heathrow_2015_data['Daily Total Rainfall'].astype('float')
```

Introduction to Data Science

Lesson 3: Cleaning, formatting and grouping data

Cleaning data: removing commas

```
travel_2011_data['In employment'] = travel_2011_data['In employment'].str.replace(',', '')
```

Cleaning data: replacing multiple values for a field

```
cars_data['PropulsionTypeId'] = cars_data['PropulsionTypeId'].replace({1:'Petrol',2:'Diesel',3:'Electric',7:'Gas/P',8: 'E/P'})
```

Creating a derived field (e.g. percentage)

```
travel_2011_data['Bicycle percent'] = travel_2011_data['Bicycle']/travel_2011_data['In employment']*100
```

Displaying mean and standard deviation

```
print(travel_2011_data.groupby(['Region'])['Bicycle percent'].mean())
print(travel_2011_data.groupby(['Region'])['Bicycle percent'].std())

print("mass: mean = "+str(petrol_data['Mass'].mean()))
```

Filtering: removing values from a dataset

```
cars_data=cars_data[cars_data['Mass'] >0]
```

Filtering: creating a new dataset based on a subset of an existing one

```
petrol_data = cars_data[cars_data['PropulsionTypeId'] == 'Petrol']
```

Introduction to Data Science

Lesson 4: Data presentation/visualisation

Adding modules

```
import matplotlib.pyplot as plt
```

Displaying a boxplot

```
country_data.boxplot(column = ['life expectancy at birth'])
plt.show()
```

Displaying a horizontal, resized boxplot grouped by a field

```
country_data.boxplot(column = ['life expectancy at birth'],by='Sub region', vert=False, figsize=(12, 8))
plt.show()
```

Displaying a histogram

```
country_data['life expectancy at birth'].plot.hist(bins=[40,50,60,70,80,90,100], density=1)
plt.show()
```

Displaying a density plot

```
country_data['life expectancy at birth'].plot.density()
plt.show()
```

Displaying a scatter diagram

```
country_data.plot.scatter(x='GDP per capita (US$)', y='life expectancy at birth')
plt.show()
```

Displaying a hexagonal bin plot

```
country_data.plot.hexbin(x='GDP per capita (US$)', y='life expectancy at birth', gridsize=10)
plt.show()
```

Displaying a line plot (e.g. for time series)

```
male_le_data.plot(x='Years', y=['North East', 'North West', 'Yorkshire and The Humber', 'East Midlands',], figsize=(10, 10))
plt.show()
```

Lesson 5: Exploring association

Displaying a scatter diagram

```
cars_data.plot.scatter(x='EngineSize', y='CO2') plt.show()
```

Displaying a scatter diagram with a third (numerical) variable controlling the colour

```
cars_data.plot.scatter(x='Mass', y='CO2', c='EngineSize', figsize=(10,10), sharex=False) plt.show()
```

Displaying a scatter diagram with a third (numerical) variable controlling the size

```
cars_data.plot.scatter(x='Mass', y='CO2', s=cars_data['EngineSize']/50, figsize=(10,10)) plt.show()
```

Displaying a scatter diagram with a third (categorical) variable controlling the colour by using a colour map

```
cmap = {'Petrol': 'red', 'Diesel': 'blue'}'
cars_data.plot.scatter(x='Mass', y='CO2', figsize=(10,10), c=[cmap.get(c, 'black') for c in cars_data.PropulsionTypeId])
plt.show()
```