DA2 : Data (tabular) and plotting - live demo

Reading in a CSV (Comma Seperated Variable) file into Python.

We use the pandas **read_csv** function.

CSV is the recommended format to store data for program interoperability.

In [ ]:
```
import pandas as pd
surveys = pd.read_csv("data/surveys.csv")

# ALWAYS take a look at data after reading it in. pandas has a handy .head(0 object method.
surveys.head()
```

In [ ]:
```
# Complementary .tail(0 method is also useful
surveys.tail()
```

What issue is immediately apparent in the output?

In [ ]:
```
# Let us explore our data further with pandas properties
surveys.shape
```

In [ ]:
```
# number of rows?
surveys.shape[0]
```

In [ ]:
```
#number of columns?
surveys.shape[1]
```

In [ ]:
```
# We can use the .describe() method to get some summary statistics
surveys.describe()
```

Hold on!

We had 9 columns and only got 7 from describe - suggests that pandas cannot generate summary statistics for hose columns. What are these 2 columns called?

In [ ]:
```
surveys.columns
```

In [ ]:
```
# looks like those 2 columns are probably some form of categorical data.
# We might find it useful to see what unique values they take. we use the .unique() method.
surveys["species_id"].unique()
```

In [ ]:
```
surveys["sex"].unique()
```

We can find the number of missing values in a column by combining two methods *.isna()* and *.sum()*.

In [ ]:
```
surveys["weight"].isna().sum()
```

Subsetting data objects (pandas datframe example)

Start by listing the record_id column.

In [ ]:
```
surveys.record_id
```

Now listing record_id and weight...

In [ ]:
```
surveys[["record_id","weight"]]
```

Too much data? Let's just look at the first 5 values using the **.iloc()** method.
(Note Pythons indexing & sequencing).

In [ ]:
```
surveys[["record_id","weight"]].iloc[0:5]
```

Simple plots using *plot_nine*

Starting with a scatterplot

In [ ]:
```
# ensure plotnine available in this notebook
from plotnine import *
```

In [ ]:
```
p = (ggplot(surveys, aes(x = "weight", y = "hindfoot_length")) +
  geom_point())

p.show()
```

Whole lot of data - where is it all comng from? let's try colouring by year...

In [ ]:
```
p = (ggplot(surveys, aes(x = "weight", y = "hindfoot_length", colour= "year")) +
  geom_point())

p.show()
```

Facetting plots

That didn't help a lot as we have overplotting - if only there was a way to create a plot for each year... 😀

In [ ]:
```
p = (ggplot(surveys, aes(x = "weight", y = "hindfoot_length", colour= "year")) +
  geom_point()) + facet_wrap("~ year")

p.show()
```

In [ ]:
```

```