Reading CSV Files

Let's practice reading csv files with this toy dataset on student scores. As you've seen a few times already, `read_csv()` is used to load data from csv files into a Pandas dataframe. We just need to specify the filepath of our data. I stored `student_scores.csv` in the same directory as this Jupyter notebook, so we just need to provide the name of the file.

Run each cell as you go through this Jupyter notebook.

In [1]:
```
import pandas as pd

df = pd.read_csv('student_scores.csv')
```

`head()` is a useful function you can call on your dataframe to display the first few rows. Let's use it to see what this data looks like.

In [2]:
```
df.head()
```

Out[2]:

	ID	Name	Attendance	HW	Test1	Project1	Test2	Project2	Final
0	27604	Joe	0.96	0.97	87.0	98.0	92.0	93.0	95.0
1	30572	Alex	1.00	0.84	92.0	89.0	94.0	92.0	91.0
2	39203	Avery	0.84	0.74	68.0	70.0	84.0	90.0	82.0
3	28592	Kris	0.96	1.00	82.0	94.0	90.0	81.0	84.0
4	27492	Rick	0.32	0.85	98.0	100.0	73.0	82.0	88.0

Remember, CSV stands for comma separated values - but they can actually be separated by different characters, tabs, white space, etc. If your file is separated by a colon, let's say, you can still use `read_csv()` with the `sep` parameter.

In [3]:
```
df = pd.read_csv('student_scores.csv', sep=':')
df.head()
```

Out[3]:

	ID,Name,Attendance,HW,Test1,Project1,Test2,Project2,Final
0	27604,Joe,0.96,0.97,87.0,98.0,92.0,93.0,95.0
1	30572,Alex,1.0,0.84,92.0,89.0,94.0,92.0,91.0
2	39203,Avery,0.84,0.74,68.0,70.0,84.0,90.0,82.0
3	28592,Kris,0.96,1.0,82.0,94.0,90.0,81.0,84.0
4	27492,Rick,0.32,0.85,98.0,100.0,73.0,82.0,88.0

This obviously didn't work because there our CSV file is separated by commas. Because there are no colons, nothing was separated and everything was read into one column!

Headers

Another thing you can do with `read_csv` is specify which line of the file is the header, which specifies the column labels. It's usually the first line, but sometimes we'll want to specify a later line if there is extra meta information at the top of the file. We can do that like this.

In [4]:
```
df = pd.read_csv('student_scores.csv', header=2)
df.head()
```

Out[4]:

	30572	Alex	1.0	0.84	92.0	89.0	94.0	92.0.1	91.0
0	39203	Avery	0.84	0.74	68.0	70.0	84.0	90.0	82.0
1	28592	Kris	0.96	1.00	82.0	94.0	90.0	81.0	84.0
2	27492	Rick	0.32	0.85	98.0	100.0	73.0	82.0	88.0

Here, row 2 was used as the the header and everything above that was cut off. By default, `read_csv` uses header=0, which uses the first line for column labels.

If columns labels are not included in your file, you can use `header=None` to prevent your first line of data from being misinterpreted as column labels.

In [5]:
```
df = pd.read_csv('student_scores.csv', header=None)
df.head()
```

Out[5]:

	0	1	2	3	4	5	6	7	8
0	ID	Name	Attendance	HW	Test1	Project1	Test2	Project2	Final
1	27604	Joe	0.96	0.97	87.0	98.0	92.0	93.0	95.0
2	30572	Alex	1.0	0.84	92.0	89.0	94.0	92.0	91.0
3	39203	Avery	0.84	0.74	68.0	70.0	84.0	90.0	82.0
4	28592	Kris	0.96	1.0	82.0	94.0	90.0	81.0	84.0

You can also specify your own column labels like this.

In [6]:
```
labels = ['id', 'name', 'attendance', 'hw', 'test1', 'project1', 'test2'
, 'project2', 'final']
df = pd.read_csv('student_scores.csv', names=labels)
df.head()
```

Out[6]:

	id	name	attendance	hw	test1	project1	test2	project2	final
0	ID	Name	Attendance	HW	Test1	Project1	Test2	Project2	Final
1	27604	Joe	0.96	0.97	87.0	98.0	92.0	93.0	95.0
2	30572	Alex	1.0	0.84	92.0	89.0	94.0	92.0	91.0
3	39203	Avery	0.84	0.74	68.0	70.0	84.0	90.0	82.0
4	28592	Kris	0.96	1.0	82.0	94.0	90.0	81.0	84.0

If you want to tell pandas that there was a header line that you are replacing, you can specify the row of that line like this.

In [7]:
```
labels = ['id', 'name', 'attendance', 'hw', 'test1', 'project1', 'test2'
, 'project2', 'final']
df = pd.read_csv('student_scores.csv', header=0, names=labels)
df.head()
```

Out[7]:

	id	name	attendance	hw	test1	project1	test2	project2	final
0	27604	Joe	0.96	0.97	87.0	98.0	92.0	93.0	95.0
1	30572	Alex	1.00	0.84	92.0	89.0	94.0	92.0	91.0
2	39203	Avery	0.84	0.74	68.0	70.0	84.0	90.0	82.0
3	28592	Kris	0.96	1.00	82.0	94.0	90.0	81.0	84.0
4	27492	Rick	0.32	0.85	98.0	100.0	73.0	82.0	88.0

Index

Instead of using the default index (integers incrementing by 1 from 0), you can specify one or more of your columns to be the index of your dataframe.

In [8]:
```
df = pd.read_csv('student_scores.csv', index_col='Name')
df.head()
```

Out[8]:

	ID	Attendance	HW	Test1	Project1	Test2	Project2	Final
Name								
Joe	27604	0.96	0.97	87.0	98.0	92.0	93.0	95.0
Alex	30572	1.00	0.84	92.0	89.0	94.0	92.0	91.0
Avery	39203	0.84	0.74	68.0	70.0	84.0	90.0	82.0
Kris	28592	0.96	1.00	82.0	94.0	90.0	81.0	84.0
Rick	27492	0.32	0.85	98.0	100.0	73.0	82.0	88.0

In [9]:
```
df = pd.read_csv('student_scores.csv', index_col=['Name', 'ID'])
df.head()
```

Out[9]:

		Attendance	HW	Test1	Project1	Test2	Project2	Final
Name	**ID**							
Joe	**27604**	0.96	0.97	87.0	98.0	92.0	93.0	95.0
Alex	**30572**	1.00	0.84	92.0	89.0	94.0	92.0	91.0
Avery	**39203**	0.84	0.74	68.0	70.0	84.0	90.0	82.0
Kris	**28592**	0.96	1.00	82.0	94.0	90.0	81.0	84.0
Rick	**27492**	0.32	0.85	98.0	100.0	73.0	82.0	88.0

There are many other things you can do with this function alone, such as parsing dates, filling null values, skipping rows, etc. A lot of these can be done in different steps after `read_csv()`. We're going to modify our data in other ways, but you can always look up how to do some steps with this function here (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_csv.html).

Quiz #1

Use `read_csv()` to read in `cancer_data.csv` and use an appropriate column as the index. Then, use `.head()` on your dataframe to see if you've done this correctly. *Hint: First call `read_csv()` **without parameters** and then `head()` to see what the data looks like.*

In [11]:
```
df_cancer = pd.read_csv('cancer_data.csv', index_col='id')
df_cancer.head()
```

Out[11]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothn
id						
842302	M	17.99	NaN	122.80	1001.0	0.11840
842517	M	20.57	17.77	132.90	1326.0	0.08474
84300903	M	19.69	21.25	130.00	1203.0	0.10960
84348301	M	11.42	20.38	77.58	386.1	NaN
84358402	M	20.29	14.34	135.10	1297.0	0.10030

5 rows × 31 columns

Quiz #2

Use `read_csv()` to read in `powerplant_data.csv` with more descriptive column names based on the description of features on this website (http://archive.ics.uci.edu/ml/datasets/combined+cycle+power+plant). Then, use `.head()` on your dataframe to see if you've done this correctly. *Hint: Like in the previous quiz, first call `read_csv()` without parameters and then `head()` to see what the data looks like.*

In [13]:
```
new_column_labels = ['temperature', 'exhaust_vacuum', 'pressure', 'humid
ity', 'energy_output']
df_powerplant = pd.read_csv('powerplant_data.csv', names=new_column_labe
ls, header=0)
df_powerplant.head()
```

Out[13]:

	temperature	exhaust_vacuum	pressure	humidity	energy_output
0	8.34	40.77	1010.84	90.01	480.48
1	23.64	58.49	1011.40	74.20	445.75
2	29.74	56.90	1007.15	41.91	438.76
3	19.07	49.69	1007.22	76.79	453.09
4	11.80	40.66	1017.13	97.20	464.43

Writing CSV Files

Awesome! Now, we'll save your second dataframe with power plant data into a csv file for the next section.

In [14]:
```
df_powerplant.to_csv('powerplant_data_edited.csv')
```

Let's see if that worked the way we wanted.

In [15]:
```
df = pd.read_csv('powerplant_data_edited.csv')
df.head()
```

Out[15]:

	Unnamed: 0	temperature	exhaust_vacuum	pressure	humidity	energy_output
0	0	8.34	40.77	1010.84	90.01	480.48
1	1	23.64	58.49	1011.40	74.20	445.75
2	2	29.74	56.90	1007.15	41.91	438.76
3	3	19.07	49.69	1007.22	76.79	453.09
4	4	11.80	40.66	1017.13	97.20	464.43

What's this `Unnamed:0`? `to_csv()` will store our index unless we tell it not to. To make it ignore the index, we have to provide the parameter `index=False`

In [16]:
```
df_powerplant.to_csv('powerplant_data_edited.csv', index=False)
```

In [17]:
```
df = pd.read_csv('powerplant_data_edited.csv')
df.head()
```

Out[17]:

	temperature	exhaust_vacuum	pressure	humidity	energy_output
0	8.34	40.77	1010.84	90.01	480.48
1	23.64	58.49	1011.40	74.20	445.75
2	29.74	56.90	1007.15	41.91	438.76
3	19.07	49.69	1007.22	76.79	453.09
4	11.80	40.66	1017.13	97.20	464.43