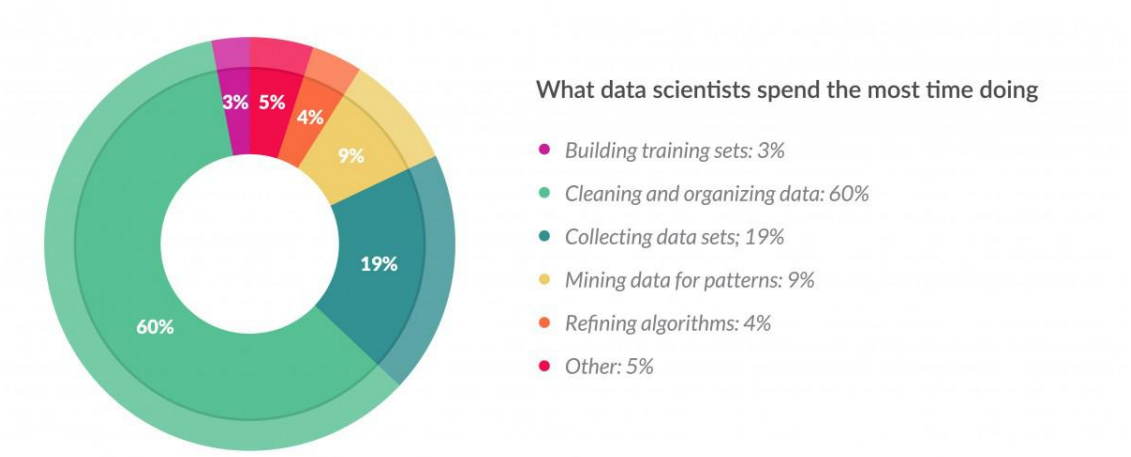
18. Pandas - Part 1

October 27, 2022

1 Data Wrangling

- Data wrangling is the process of cleaning and unifying messy and complex data sets for easy access and analysis.
- This process typically includes manually converting and mapping data from one raw form into another format to allow for more convenient consumption and organization of the data.



2 Pandas

- `pandas` is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

3 Install Pandas

- **Recommended** to create virtual environment and install packages in it.
- To install pandas, on treminal/Command Prompt enter > `pip install pandas`
- Also install matplotlib, required for plotting charts. > `pip install matplotlib`

4 Import pandas

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

5 How to read a tabular data file into pandas?

[ ]:
```
# read a dataset of ufo
df = pd.read_csv('data/ufo.csv')
```

[ ]:
```
df
```

[ ]:
```
type(df)
```

[ ]:
```
# examine the first 5 rows
df.head()
```

5.1 Reading with seperators other than comma

[ ]:
```
chip = pd.read_csv('data/chipotle.tsv',sep='\t')
```

[ ]:
```
chip.head()
```

6 How to select a Column from a DataFrame?

[ ]:
```
# examine the first 5 rows
df.head(1)
```

[ ]:
```
# select the 'City' Column using bracket notation
# Column also called as Series
type(df['City'])
```

[ ]:
```
# or equivalently, use dot notation
df.City
```

Square Bracket notation will always work, whereas **dot notation** has limitations:

- Dot notation doesn't work if there are **spaces** in the Column name
- Dot notation doesn't work if the Column has the same name as a **DataFrame method or attribute** (like 'head' or 'shape')
- Dot notation can't be used to define the name of a **new Column**

[ ]:
```
# create a new 'Location' Column (must use bracket notation to define the␣
↪Column name)
df['Location'] = df.City + ', ' + df.State
```

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

7 Some more basic commands and attributes

[ ]:
```
# read a dataset of top-rated IMDb movies into a DataFrame
movies = pd.read_csv('data/imdb_1000.csv')
```

[ ]:
```
movies.describe()
```

Methods end with parentheses, while **attributes** don't:

[ ]:
```
# example method: show the first 5 rows
df.head()
```

[ ]:
```
# get number of rows and columns in df
df.shape
```

[ ]:
```
# method to find summary statistics
df.describe()
```

[ ]:
```
# get data type of each column
df.dtypes
```

[ ]:
```
movies.dtypes
```

8 How to rename columns in a pandas DataFrame?

[ ]:
```
# dataset of UFO reports
df
```

[ ]:
```
# examine the column names
df.columns
```

8.1 rename two of the columns by using the 'rename' method

[ ]:
```
df.rename(columns={'Colors Reported':'Colors_Reported',
                   'Shape Reported':'Shape_Reported'}, inplace=True)
```

[ ]:
```
df.columns
```

8.2 Replace all of the column names by overwriting the 'columns' attribute

[ ]:
```
df_cols = ['city', 'colors reported', 'shape reported', 'state', 'time',␣
 ↪'location']
```

[ ]:
```
df.columns = df_cols
```

[ ]:
```
df.columns
```

8.3 Replace the column names during the file reading process

[ ]:
```
df_cols = ['city', 'colors reported', 'shape reported', 'state', 'time']
```

[ ]:
```
df = pd.read_csv('data/ufo.csv', header=0, names=df_cols)
```

[ ]:
```
df.columns
```

[ ]:
```
# replace all spaces with underscores in the column names by using
# the 'str.replace' method
df.columns = df.columns.str.replace(' ', '_')
```

[ ]:
```
df.columns
```

9 How to remove columns from a pandas DataFrame?

[ ]:
```
df.tail(3) # shows last 5 rows of df
```

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

[ ]:
```
# remove a single column
# axis=1 refers to columns, axis=0 refers to rows(index), default is 0
df.drop('colors_reported', axis=1)
```

[ ]:
```
df.drop('colors_reported', axis=1, inplace=True)
```

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

[ ]:
```
# remove multiple columns at once
df.drop(['city', 'state'], axis=1, inplace=True)
```

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

[ ]:
```
# remove multiple rows at once (axis=0 refers to rows)
df.drop([0, 1], axis=0, inplace=True)
```

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