Week 2

Pre-Processing Data in Python

1. Id / handle missing values
2. Data formatting
3. Data normalization
4. Data binning
5. Categorial values → numeric vars

- Python: operations along columns

---

1. Missing value can be represented as "?", "N/A", "0", a blank cell, or "NaN"

- Check w/ data source to see if data should exist
- Drop value or entire var (column)
- Replace data w/ a guess
  - Use avg, mode, or context clues

In Pandas, use dataframes.dropna()
- axis = 0 drops whole row; 1 drops whole column
- df.dropna(subset=["row name"], axis=#, inplace=(T)/F)

This part won't actually make any changes — will modify

Replaces NaN's w/ mean

or mode

```
1  mean = df["row name"].mean()
2  df["row name"].replace(np.nan, mean)
```

np.nan = formatted NaN

missing val — new val

---

2. Units:

Replaces mpg column w/ L/100km column and accordingly edits vals

```
1  df["mpg"] = 235/df["mpg"] yields L/100km
2  df.rename(columns={"mpg": "L/100km"}, inplace=True)
```

- df.~~dtype~~dtypes() → check data type (str, obj, int)
- df.astype() → change data type

↳
```
1  df["price"] = df["price"].astype("int")
```

3. With numbers that are larger depending on the variable (ie. price > year), data normalization allows each variable to similarly influence statistical models.

For example,

	Price	Year
range	$0 to $1 000 000	1980 to 2018
1	$30 000	1990
2	$40 000	1995
3	$100 000	2010

→

	Price	Year
range		
1	0.03	0.33
2	0.04	0.5
3	0.1	1.0

3 methods:

1. Simple feature scaling: $x_{new} = \frac{x_{old}}{x_{max}}$: $0 \leq x \leq 1$
2. Min-Max: $x_{new} = \frac{x_{old} - x_{min}}{x_{max} - x_{min}\ (rng)}$: $0 \leq x \leq 1$
3. Z-score: $x_{new} = \frac{x_{old} - \mu}{\sigma}$

```
df["var"] = df["var"] / df["var"].max()
df["var"] = (df["var"] - df["var"].min()) / (df["var"].max() / df["var"].min())
df["var"] = (df["var"] - df["var"].mean()) / df["var"].std()
```

4. Binning converts data into bins (ie. histograms)

```
1 bins = np.linspace(min(df["var"]), max(df["var"]), #)   # of sections
2 group_names = ["1", "2", "3"]
3 df["var-binned"] = pd.cut(df["var"], bins, labels = group_names, include_lowest = True)
```

5. Most statistical models can't use obj or str as input

Car	Fuel	...	Gas	Diesel	Elec
1	Gas		1	0	0
2	Diesel	→	0	1	0
3	Elec		0	0	1

"one-hot encoding"

- pd.get_dummies(df["var"])