Cleaning Practice

Let's first practice handling missing values and duplicate data with `cancer_data_means.csv`.

In [1]:
```
# import pandas and load cancer data
import pandas as pd

df = pd.read_csv('cancer_data_means.csv')

# check which columns have missing values with info()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 11 columns):
id                     569 non-null int64
diagnosis              569 non-null object
radius_mean            569 non-null float64
texture_mean           548 non-null float64
perimeter_mean         569 non-null float64
area_mean              569 non-null float64
smoothness_mean        521 non-null float64
compactness_mean       569 non-null float64
concavity_mean         569 non-null float64
concave_points_mean    569 non-null float64
symmetry_mean          504 non-null float64
dtypes: float64(9), int64(1), object(1)
memory usage: 49.0+ KB
```

In [2]:
```
# use means to fill in missing values
df.fillna(df.mean(), inplace=True)

# confirm your correction with info()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 11 columns):
id                     569 non-null int64
diagnosis              569 non-null object
radius_mean            569 non-null float64
texture_mean           569 non-null float64
perimeter_mean         569 non-null float64
area_mean              569 non-null float64
smoothness_mean        569 non-null float64
compactness_mean       569 non-null float64
concavity_mean         569 non-null float64
concave_points_mean    569 non-null float64
symmetry_mean          569 non-null float64
dtypes: float64(9), int64(1), object(1)
memory usage: 49.0+ KB
```

In [3]:
```
# check for duplicates in the data
sum(df.duplicated())
```

Out[3]: 5

In [4]:
```
# drop duplicates
df.drop_duplicates(inplace=True)
```

In [5]:
```
# confirm correction by rechecking for duplicates in the data
sum(df.duplicated())
```

Out[5]: 0

Renaming Columns

Since we also previously changed our dataset to only include means of tumor features, the "_mean" at the end of each feature seems unnecessary. It just takes extra time to type in our analysis later. Let's come up with a list of new labels to assign to our columns.

In [6]:
```
# remove "_mean" from column names
new_labels = []
for col in df.columns:
    if '_mean' in col:
        new_labels.append(col[:-5])  # exclude last 6 characters
    else:
        new_labels.append(col)

# new labels for our columns
new_labels
```

Out[6]:
```
['id',
 'diagnosis',
 'radius',
 'texture',
 'perimeter',
 'area',
 'smoothness',
 'compactness',
 'concavity',
 'concave_points',
 'symmetry']
```

In [7]:
```
# assign new labels to columns in dataframe
df.columns = new_labels

# display first few rows of dataframe to confirm changes
df.head()
```

Out[7]:

	id	diagnosis	radius	texture	perimeter	area	smoothness	compactness
0	842302	M	17.99	19.293431	122.80	1001.0	0.118400	0.27760
1	842517	M	20.57	17.770000	132.90	1326.0	0.084740	0.07864
2	84300903	M	19.69	21.250000	130.00	1203.0	0.109600	0.15990
3	84348301	M	11.42	20.380000	77.58	386.1	0.096087	0.28390
4	84358402	M	20.29	14.340000	135.10	1297.0	0.100300	0.13280

In [8]:
```
# save this for later
df.to_csv('cancer_data_edited.csv', index=False)
```