# Why do missing values exist?

# How gaps in data occur

- Data not being collected properly
- Collection and management errors
- Data intentionally being omitted
- Could be created due to transformations of the data

# Why we care?

- Some models cannot work with missing data (Nulls/NaNs)
- Missing data may be a sign of a wider data issue
- Missing data can be a useful feature

# Missing value discovery

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 999 entries, 0 to 998
Data columns (total 12 columns):
SurveyDate                    999 non-null object
...
StackOverflowJobsRecommend    487 non-null float64
VersionControl                999 non-null object
Gender                        693 non-null object
RawSalary                     665 non-null object
dtypes: float64(2), int64(2), object(8)
memory usage: 93.7+ KB
```

# Finding missing values

```
print(df.isnull())
```

```
   StackOverflowJobsRecommend  VersionControl  ... \
0                        True           False  ...
1                       False           False  ...
2                       False           False  ...
3                        True           False  ...
4                       False           False  ...

   Gender  RawSalary
0   False       True
1   False      False
2    True       True
3   False      False
4   False      False
```

# Finding missing values

```
print(df['StackOverflowJobsRecommend'].isnull().sum())
```

```
512
```

# Finding non-missing values

```
print(df.notnull())
```

```
   StackOverflowJobsRecommend  VersionControl  ... \
0                       False            True  ...
1                        True            True  ...
2                        True            True  ...
3                       False            True  ...
4                        True            True  ...

   Gender  RawSalary
0    True      False
1    True       True
2   False      False
3    True       True
4    True       True
```

# Go ahead and find missing values!

# Dealing with missing values (I)

# Listwise deletion

```
      SurveyDate      ConvertedSalary     Hobby ... \
0  2/28/18 20:20                 NaN       Yes ...
1  6/28/18 13:26             70841.0       Yes ...
2    6/6/18 3:37                 NaN        No ...
3    5/9/18 1:06             21426.0       Yes ...
4  4/12/18 22:41             41671.0       Yes ...
```

# Listwise deletion in Python

```
# Drop all rows with at least one missing values
df.dropna(how='any')
```

# Listwise deletion in Python

```
# Drop rows with missing values in a specific column
df.dropna(subset=['VersionControl'])
```

# Issues with deletion

- It deletes valid data points
- Relies on randomness
- Reduces information

# Replacing with strings

```
# Replace missing values in a specific column
# with a given string
df['VersionControl'].fillna(
    value='None Given', inplace=True
)
```

# Recording missing values

```
# Record where the values are not missing
df['SalaryGiven'] = df['ConvertedSalary'].notnull()
```

```
# Drop a specific column
df.drop(columns=['ConvertedSalary'])
```

# Practice time

# Fill continuous missing values

# Deleting missing values

- Can't delete rows with missing values in the test set

# What else can you do?

- **Categorical columns**: Replace missing values with the most common occurring value or with a string that flags missing values such as 'None'
- **Numeric columns**: Replace missing values with a suitable value

# Measures of central tendency

- Mean
- Median

# Calculating the measures of central tendency

```
print(df['ConvertedSalary'].mean())
print(df['ConvertedSalary'].median())
```

```
92565.16992481203
55562.0
```

# Fill the missing values

```
df['ConvertedSalary'] = df['ConvertedSalary'].fillna(
    df['ConvertedSalary'].mean()
)
```

```
df['ConvertedSalary'] = df['ConvertedSalary']\
                        .astype('int64')
```

# Rounding values

```
df['ConvertedSalary'] = df['ConvertedSalary'].fillna(
    round(df['ConvertedSalary'].mean())
)
```

# Let's Practice!

# Dealing with other data issues

# Bad characters

```
print(df['RawSalary'].dtype)
```

```
dtype('O')
```

# Bad characters

```
print(df['RawSalary'].head())
```

```
0          NaN
1    70,841.00
2          NaN
3    21,426.00
4    41,671.00
Name: RawSalary, dtype: object
```

# Dealing with bad characters

```
df['RawSalary'] = df['RawSalary'].str.replace(',', '')
```

```
df['RawSalary'] = df['RawSalary'].astype('float')
```

# Finding other stray characters

```
print(df[coerced_vals.isna()].head())
```

```
0           NaN
2           NaN
4     $51408.00
Name: RawSalary, dtype: object
```

# Chaining methods

```
df['column_name'] = df['column_name'].method1()
df['column_name'] = df['column_name'].method2()
df['column_name'] = df['column_name'].method3()
```

Same as:

```
df['column_name'] = df['column_name']\
                    .method1().method2().method3()
```

# Go ahead and fix bad characters!