Cleaning Data

Reindert-Jan Ekker

@rjekker http://nl.linkedin.com/in/rjekker

Overview

Missing data

- Detect and inspect
- Remove
- Fill or interpolate

Unwanted data

- Outliers
- Duplicates

Type conversions

Fixing indices

Demo

Missing data

- Detect and inspect
- Remove
- Fill or interpolate

Detecting Missing Data

```
# isnull() returns True for every cell that is NaN
# any() returns True if a column is True at least once

# Which columns have missing values?
df.isnull().any()
df[df.isnull().any(axis=1)] # Use axis=1 for rows

# notnull() and all() work similar to isnull() and any()
```

Removing Missing Values

```
# You can use df.drop() to remove items
# But df.dropna() is more powerful in this case
df.dropna()            # drops all rows with null values
df.dropna(axis=1)      # drop columns
df.dropna(thresh=4)    # keep only rows with 4 values or more
df.dropna(how=‘all’) # only drop if all values are NaN
df.dropna(how=‘any’) # drop if any values are NaN
df.dropna(inplace=True)
```

Filling Missing Values

```
# Replace all NaN values with a specific value
df.fillna(5)


# fillna() accepts a Series of values
# Per column: replace missing data with mean
df.fillna(df.mean())
```

Interpolation

```
# Fill missing values with previous value
df.fillna(method='ffill')
# Use 'bfill' to fill backwards


# fillna() also accepts options inplace and columns


# For advanced interpolations use df.interpolate()
```

Demo

Handling unwanted data

- Outliers
- Duplicates

Removing Duplicates

```
# duplicated() returns a Series of Booleans
# which is True whenever a row is a duplicate
df[df.duplicated()]  # shows all duplicates


# removing all duplicates
df.drop_duplicates()
# unique() does the same but returns a numpy array
df.unique()   # you usually don't want this
```

Demo

Type conversions

Fixing indices

Converting types

```
# We can use the astype() function
# And pass it the type we want to convert to
df['some_column'].astype(int)

# Or pass a dict with a type per column
df.astype({'name': str, 'age': int})
# Note: All values have to fit into the new data type
```

Data Types

Strings (nullable)

Python: str

Numpy: np.object

Floats (nullable)

Python: float

Numpy: np.float64

Integers (non-nullable)

Python: int

Numpy: np.int64

Others:

bool/np.bool

complex/np.complex64

Fixing Indices

```
# Set the index to a simple range 0..n
df.reset_index()
df.reset_index(drop=True)  # Don't keep the original index

# Set index from a column
df.set_index('id', drop=True)
```

Rename

```
# rename columns
df.rename(columns={‘a’: ‘Ann’, ‘b’: ‘Bob’})

# Or rename some rows
df.rename(index={‘a’: ‘Ann’, ‘b’: ‘Bob’})
```

Summary

Missing data

- Detect and inspect
- Remove
- Fill or interpolate

Unwanted data

- Outliers
- Duplicates

Type conversions

Fixing indices