Pandas

METASNAKE

Loading Data

`pd.read_csv(file)`	Read CSV file
`pd.read_sql(sql, con)`	Read from database
`pd.read_excel(file)`	Read from Excel

Inspecting Data

`df.columns`	Columns
`df.dtypes`	Types of columns
`df.info()`	Information
`df.head(10)`	First 10 rows
`df.head(10).T`	Transposed
`df.sample(10)`	Sample
`df.shape`	Tuple of (rows, columns) count
`df.index`	Index
`pd.options.display.max_columns`	View display options
`pd.option_context('display.max_rows', 3)`	Change display options in context manager

Tweaking Data

`df.rename(columns=fn)`	`fn` can be function or dictionary
`df['Make'].value_counts()`	Frequency of items
`df.assign(col=new_col)`	Create new column `new_col` can be scalar, function, or series
`df['Make'].str`	Access string methods on string columns
`df['Year'].apply(fn)`	Apply function to column
`df['Year'].astype(str)`	Convert column to string
`pd.to_numeric(s)`	Convert column to number type
`pd.to_datetime(s)`	Convert column to date type
`date_series.dt`	Access date methods on date columns
`df['CPU'] * 2`	Vectorized math operation
`df.drop(columns=['Name'])`	Remove column

Stats

`df.describe()`	Summary statistics
`df.describe(include='all')`	Include non-numeric columns
`df['Ram'].max()`	Aggregate column (the are a bunch of these)
`df['Ram'].corr(df['Year'])`	Correlate two columns
`df.corr()`	Correlate all numeric columns

Plotting

`df['Ram'].plot.hist()`	Histogram
`df['Make'].value_counts().plot.bar()`	Bar plot
`df.plot.scatter(x='Year', y='Ram')`	Scatter plot

Filtering

`m = df['Make'] == 'MBP'`	Make a boolean array where Make equals MBP
`y = df['Year'] > 2017`	Make a boolean array for numeric condition
`df.loc[m]`	Pull out rows where array is True
`df.loc[m & y]`	Combine arrays & (and) | (or) ^ (not)
`df.loc[m, ['CPU', 'Make']]`	Pull out rows and two columns by *label*
`df.iloc[[0,-1], [0,2,3]]`	Pull out rows and columns by *position*
`df.iloc[:, [0,2,3]]`	Pull out *all* rows

Missing Data

`df.isna()`	DataFrame where every cell is True/False if missing `.isnull` is a synonym
`df.isna().any(axis=1)`	Boolean array where rows have missing values
`df[df.isna()['Ram']]`	Filter out to show rows where Ram is missing
`df.isna().any(axis=0)`	Columns that are missing
`df.isna().sum(axis=0)`	Count of missing in each column
`df.isna().mean(axis=0).mul(100)`	Percent missing in each column
`df['Ram'].ffill()`	Forward fill missing numbers
`df['Ram'].interpolate()`	Interpolate to fill missing numbers
`df.dropna()`	Drop rows with missing values
`df['Ram'].fillna(v)`	Fill missing numbers v is scalar, dict, series, or dataframe
`df['Ram'].where(c, v)`	Keep where c is true, v otherwise. c is boolean array or function, v is scalar or series

Grouping

`df.groupby('Make').mean()`	Group by make and take mean of each numeric column for group
`df.groupby('Make')['Ram'].mean()`	Group by make and take mean of Ram column for group
`df.groupby(['Year','Make'])['Ram'].mean()`	Group by year and make and take mean of Ram column for group (*hierachical index*)
`df.groupby(['Year','Make']).agg(['min', 'max'])`	Group by year and make and take min and max of every numeric column for group (*hierachical column*)
`df_hcols.columns = ['_'.join(c) for c in df_hcols.columns.to_flat_index()]`	Flatten columns (*mutates!*)
`df.groupby(pd.Grouper(key='Year', freq='2w')).mean()`	Use `Grouper` to aggregate at 2 week *offset alias*
`df.pivot_table(index='Make', aggfunc='mean')`	Group by make and take mean of each numeric column for group
`df.pivot_table(index='Make', values=['Ram'], aggfunc='mean')`	Group by make and take mean of Ram column for group

Join Data

`df.merge(df2, how='left')`	Join DataFrames by common columns with left join
`df.merge(df2, left_on=c1, right_on=c2)`	Join DataFrames by indicated columns with inner join