Pandas - Basics

- df.head()
- df.info() - To display information regarding all the attributes of the data frame
- df.shape - An attribute which Returns number of rows and columns
- df.describe() - Gives a mathematical overview of all the numerical columns - count, mean, standard deviation, min, 25%, 50%, 75% and max
- df.values() - Returns a 2D list of all the values in the data frame
- df.columns - Returns the list of columns of the data frame
- df.index - Returns the information about the indexes

Code Snippets

```
# Print the head of the homelessness data
print(homelessness.head())

# Print information about homelessness
print(homelessness.info())

# Print the shape of homelessness
print(homelessness.shape)

# Print a description of homelessness
print(homelessness.describe())

# Import pandas using the alias pd
import pandas as pd

# Print the values of homelessness
print(homelessness.values)

# Print the column index of homelessness
print(homelessness.columns)

# Print the row index of homelessness
print(homelessness.index)
```

———————————————————————————————

Sorting and Subsetting

- df.sort_values(col_name, ascending = False) - To sort by one column in the descending order
- df.sort_values(['col_name1', 'col_name2']) - First we sort by col_name1

and then by col_name2

Subsetting the columns

- df[‘col_name’]
- df[[‘col_name1’, ‘col_name2’]] - To select two columns
- We can provide a subset of column names as a list and then pass it to the dataframe to get the values.
- dogs[‘height’] > 50 - This returns True in the rows dogs where this condition is True and False elsewhere
- We can use the above statement inside dataframe to filter out only the values that satisfy the condition
- dogs[dogs[‘height’] > 50] - This returns only the rows that satisfy the condition given
- dogs[dogs[‘name’] == “Labrador”]
- To check for multiple conditions:
- dogs[(dogs[“breed”] == “Labrador”) & (dogs[“color”] == “Brown”)]
- we can also store these conditions as variables and pass these variables to the dataframe

To filter on the categorical variable we will make use of .isin()
is_black_or_brown = dogs[“color”].isin([“Black”, “Brown”])
dogs[is_black_or_brown]

Code Snippets

```
# Sort homelessness by region, then descending family members
homelessness_reg_fam = homelessness.sort_values(['region', 'family_members'],
ascending = [True, False])

# Print the top few rows
print(homelessness_reg_fam.head())

# Subset for rows in South Atlantic or Mid-Atlantic regions
south_mid_atlantic = homelessness[(homelessness['region'].isin(['South Atlantic',
'Mid-Atlantic'] ))]
#south_mid_atlantic = homelessness[(homelessness['region'] == 'South Atlantic')
| (homelessness['region'] == 'Mid-Atlantic')]

# See the result
print(south_mid_atlantic)

# The Mojave Desert states
canu = ["California", "Arizona", "Nevada", "Utah"]
```

```
# Filter for rows in the Mojave Desert states
mojave_homelessness = homelessness[homelessness['state'].isin(canu)]
```

———————————————————————————————

Adding/Mutating DataFrame

```
dogs[‘new_col’] = dogs[‘old’] / 100
```