Claude Shyaka

ID: 801326243

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

```
# Load energy data
energy_data = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/data/energy_dataset.csv", delimiter=",", \
                          index_col="time")
energy_data.head()
```

time	generation biomass	generation fossil brown coal/lignite	generation fossil coal-derived gas	generation fossil gas	generation fossil hard coal	generation fossil oil
2015-01-01 00:00:00+01:00	447.0	329.0	0.0	4844.0	4821.0	162.0
2015-01-01 01:00:00+01:00	449.0	328.0	0.0	5196.0	4755.0	158.0
2015-01-01 02:00:00+01:00	448.0	323.0	0.0	4857.0	4581.0	157.0
2015-01-01 03:00:00+01:00	438.0	254.0	0.0	4314.0	4131.0	160.0
2015-01-01 04:00:00+01:00	428.0	187.0	0.0	4130.0	3840.0	156.0

5 rows × 28 columns

```
energy_data.shape
```

```
(35064, 28)
```

```
# Filter energy data from relevant features
energy_data_filters = ['total load forecast', 'total load actual']
energy_data_filtered = energy_data[energy_data_filters]
energy_data_filtered.head()
```

time	total load forecast	total load actual
2015-01-01 00:00:00+01:00	26118.0	25385.0
2015-01-01 01:00:00+01:00	24934.0	24382.0
2015-01-01 02:00:00+01:00	23515.0	22734.0
2015-01-01 03:00:00+01:00	22642.0	21286.0
2015-01-01 04:00:00+01:00	21785.0	20264.0

```
# Print shape of energy
energy_data_filtered.shape
```

```
(35064, 2)
```

```
# Load weather data
weather_data = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/data/weather_features.csv", delimiter=",", \
                           index_col="dt_iso")
weather_data.head()
```

dt_iso	city_name	temp	temp_min	temp_max	pressure	humidity	wind_spe
2015-01-01 00:00:00+01:00	Valencia	270.475	270.475	270.475	1001	77	
2015-01-01 01:00:00+01:00	Valencia	270.475	270.475	270.475	1001	77	

```
weather_data.shape
```

```
(178396, 16)
```

```
# Filter data for relevant features
weather_data_filter = ['city_name', 'temp', 'pressure', 'humidity', \
                       'rain_1h', 'rain_3h', 'snow_3h']
weather_data_filtered = weather_data[weather_data_filter]
weather_data_filtered.head()
```

dt_iso	city_name	temp	pressure	humidity	rain_1h	rain_3h	snow_3h
2015-01-01 00:00:00+01:00	Valencia	270.475	1001	77	0.0	0.0	0.0
2015-01-01 01:00:00+01:00	Valencia	270.475	1001	77	0.0	0.0	0.0
2015-01-01 02:00:00+01:00	Valencia	269.686	1002	78	0.0	0.0	0.0
2015-01-01							

```
# Get shape of weather data
weather_data_filtered.shape
```

```
(178396, 7)
```

```
# Destructure data into individual cities
valencia_df = weather_data_filtered[weather_data_filtered['city_name'].str.strip() == 'Valencia']
madrid_df = weather_data_filtered[weather_data_filtered['city_name'].str.strip() == 'Madrid']
bilbao_df = weather_data_filtered[weather_data_filtered['city_name'].str.strip() == 'Bilbao']
barcelona_df = weather_data_filtered[weather_data_filtered['city_name'].str.strip() == 'Barcelona']
seville_df = weather_data_filtered[weather_data_filtered['city_name'].str.strip() == 'Seville']
valencia_df.shape, madrid_df.shape, bilbao_df.shape, barcelona_df.shape, seville_df.shape
```

```
((35145, 7), (36267, 7), (35951, 7), (35476, 7), (35557, 7))
```

```
# Merge data from individual cities into one dataset
merged_on_time = pd.merge(valencia_df, madrid_df, on=["dt_iso"], how="inner")
merged_on_time = merged_on_time.rename({
    "city_name_x": "city_name_v",
    "temp_x": "temp_v",
    "pressure_x": "pressure_v",
    "humidity_x": "humidity_v",
    "rain_1h_x": "rain_1h_v",
    "rain_3h_x": "rain_3h_v",
    "snow_3h_x": "snow_3h_v",
    "city_name_y": "city_name_m",
    "temp_y": "temp_m",
    "pressure_y": "pressure_m",
    "humidity_y": "humidity_m",
    "rain_1h_y": "rain_1h_m",
    "rain_3h_y": "rain_3h_m",
    "snow_3h_y": "snow_3h_m",
    }, axis=1)

# Merge and rename columns
merged_on_time = pd.merge(merged_on_time, bilbao_df, \
                          on=["dt_iso"], how="inner")
merged_on_time = merged_on_time.rename({
    "city_name": "city_name_b",
    "temp": "temp_b",
    "pressure": "pressure_b",
    "humidity": "humidity_b"},
    axis=1)
```

```
# Merge and rename columns
merged_on_time = pd.merge(merged_on_time, barcelona_df, \
                          on=["dt_iso"], how="inner")
merged_on_time = merged_on_time.rename({
    "city_name": "city_name_bn",
    "temp": "temp_bn",
    "pressure": "pressure_bn",
    "humidity": "humidity_bn"},
    axis=1)

# Merge and rename columns
merged_on_time = pd.merge(merged_on_time, seville_df, \
                          on=["dt_iso"], how="inner")
merged_on_time = merged_on_time.rename({
    "city_name": "city_name_s",
    "temp": "temp_s",
    "pressure": "pressure_s",
    "humidity": "humidity_s"},
    axis=1)

# View new dataset
merged_on_time.head()
```

dt_iso	city_name_v	temp_v	pressure_v	humidity_v	rain_1h_v	rain_3h_v
2015-01-01 00:00:00+01:00	Valencia	270.475	1001	77	0.0	0.0
2015-01-01 01:00:00+01:00	Valencia	270.475	1001	77	0.0	0.0
2015-01-01 02:00:00+01:00	Valencia	269.686	1002	78	0.0	0.0
2015-01-01 03:00:00+01:00	Valencia	269.686	1002	78	0.0	0.0
2015-01-01 04:00:00+01:00	Valencia	269.686	1002	78	0.0	0.0

5 rows × 35 columns

```
# Filters to average over
temp_cols = ['temp_v', 'temp_m', 'temp_b', 'temp_bn', 'temp_s']
pressure_cols = ['pressure_v', 'pressure_m', 'pressure_b', \
                 'pressure_bn', 'pressure_s']
humidity_cols = ['humidity_v', 'humidity_m', 'humidity_b', \
                 'humidity_bn', 'humidity_s']
rain1h_cols = ['rain1h_v', 'rain1h_m', 'rain1h_b', 'rain1h_bn', \
               'rain1h_s']
rain3h_cols = ['rain3h_v', 'rain3h_m', 'rain3h_b', 'rain3h_bn', \
               'rain3h_s']
snow3h_cols = ['snow3h_v', 'snow3h_m', 'snow3h_b', 'snow3h_bn', \
               'snow3h_s']

# Compute mean for all features to build a df for overall cities
merged_on_time['avg_temp'] = merged_on_time[temp_cols].mean(axis=1)
merged_on_time['avg_pressure'] = merged_on_time[pressure_cols].mean(axis=1)
merged_on_time['avg_humidity'] = merged_on_time[humidity_cols].mean(axis=1)
merged_on_time['avg_rain1h'] = merged_on_time[humidity_cols].mean(axis=1)
merged_on_time['avg_rain3h'] = merged_on_time[humidity_cols].mean(axis=1)
merged_on_time['avg_snow3h'] = merged_on_time[humidity_cols].mean(axis=1)
merged_on_time.shape
```

```
(38568, 41)
```

```
# Filter the mean columns values
new_filter = ['avg_temp', 'avg_pressure', 'avg_humidity', 'avg_rain1h', \
              'avg_rain3h', 'avg_snow3h']
weather_data_filtered = merged_on_time[new_filter]

# Rename columns for more descriptive names
weather_data_filtered = weather_data_filtered.rename(
    {"avg_temp": "Average temperature in K",
     "avg_pressure": "Average pressure in hPa",
```

```
      "avg_humidity": "Average humidity in %",
      "avg_rain1h":"Average rain in last 1 hour in mm",
      "avg_rain3h":"Average rain in last 3 hours in mm",
      "avg_snow3h":"Average snow in last 3 hours in mm"
      }, axis=1)
weather_data_filtered.index.names = ['time']
weather_data_filtered.columns
```

```
Index(['Average temperature in K', 'Average pressure in hPa',
       'Average humidity in %', 'Average rain in last 1 hour in mm',
       'Average rain in last 3 hours in mm',
       'Average snow in last 3 hours in mm'],
      dtype='object')
```

```
# merge load and weather data
df = pd.merge(weather_data_filtered, energy_data_filtered, on="time", \
              how="inner")

# Check for any missing values function
df.isna().any()
```

```
Average temperature in K              False
Average pressure in hPa               False
Average humidity in %                 False
Average rain in last 1 hour in mm     False
Average rain in last 3 hours in mm    False
Average snow in last 3 hours in mm    False
total load forecast                   False
total load actual                      True
dtype: bool
```

```
# Fill missing values by linear interpolation
df = df.interpolate(method='linear', limit_direction='forward')
# recheck for missing values
df.isna().any()
```

```
Average temperature in K              False
Average pressure in hPa               False
Average humidity in %                 False
Average rain in last 1 hour in mm     False
Average rain in last 3 hours in mm    False
Average snow in last 3 hours in mm    False
total load forecast                   False
total load actual                     False
dtype: bool
```

```
# View current state of the complete dataset
df.head()
```

time	Average temperature in K	Average pressure in hPa	Average humidity in %	Average rain in last 1 hour in mm	Average rain in last 3 hours in mm	Average snow in last 3 hours in mm	tota loa forecas
2015-01-01 00:00:00+01:00	272.491463	1016.4	82.4	82.4	82.4	82.4	26118.
2015-01-01 01:00:00+01:00	272.512700	1016.2	82.4	82.4	82.4	82.4	24934.
2015-01-01 02:00:00+01:00	272.099137	1016.8	82.0	82.0	82.0	82.0	23515.
2015-01-01 03:00:00+01:00	272.089469	1016.6	82.0	82.0	82.0	82.0	22642.

```
df.tail()
```

time	Average temperature in K	Average pressure in hPa	Average humidity in %	Average rain in last 1 hour in mm	Average rain in last 3 hours in mm	Average snow in last 3 hours in mm	tota loa forecas
2018-12-31 19:00:00+01:00	284.470	1029.2	73.6	73.6	73.6	73.6	30619.

```
# Filter dataset for features
feature = ["Average temperature in K", "Average pressure in hPa", \
           "Average humidity in %", "Average rain in last 1 hour in mm", \
           "Average rain in last 3 hours in mm", \
           "Average snow in last 3 hours in mm", "total load actual"]
df_dataset = df[feature]
df_dataset.head()
```

time	Average temperature in K	Average pressure in hPa	Average humidity in %	Average rain in last 1 hour in mm	Average rain in last 3 hours in mm	Average snow in last 3 hours in mm	total load actual
2015-01-01 00:00:00+01:00	272.491463	1016.4	82.4	82.4	82.4	82.4	25385.0
2015-01-01 01:00:00+01:00	272.512700	1016.2	82.4	82.4	82.4	82.4	24382.0

```
df_dataset.shape
```

```
(38568, 7)
```

```
# Save to file
df_dataset.to_csv("/content/drive/MyDrive/Colab Notebooks/data/power_dataset.csv")
```