

```

{"username": "lucasrudd", "key": "e361954e02971eed5dfa1e70dc8b1ea8"}
import os
os.environ['KAGGLE_USERNAME'] = 'lucasrudd'
os.environ['KAGGLE_KEY'] = 'e361954e02971eed5dfa1e70dc8b1ea8'

import kaggle
kaggle.api.authenticate()

import pandas as pd
import zipfile

!kaggle datasets download -d hnavrodiev/london-bike-sharing-dataset

Dataset URL: https://www.kaggle.com/datasets/hnavrodiev/london-bike-sharing-dataset
License(s): other
london-bike-sharing-dataset.zip: Skipping, found more recently modified local copy (use --force to force download)

zipfile_name = 'london-bike-sharing-dataset.zip'
with zipfile.ZipFile(zipfile_name, 'r') as file:
    file.extractall()

bikes = pd.read_csv("london_merged.csv")

bikes.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17414 entries, 0 to 17413
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   timestamp             17414 non-null  object
 1   cnt                   17414 non-null  int64
 2   t1                    17414 non-null  float64
 3   t2                    17414 non-null  float64
 4   hum                   17414 non-null  float64
 5   wind_speed            17414 non-null  float64
 6   weather_code          17414 non-null  float64
 7   is_holiday            17414 non-null  float64
 8   is_weekend            17414 non-null  float64
 9   season                17414 non-null  float64
dtypes: float64(8), int64(1), object(1)
memory usage: 1.3+ MB

bikes.shape

(17414, 10)

bikes

```

	weather_code \	timestamp	cnt	t1	t2	hum	wind_speed
0	2015-01-04	00:00:00	182	3.0	2.0	93.0	6.0
3.0							
1	2015-01-04	01:00:00	138	3.0	2.5	93.0	5.0
1.0							
2	2015-01-04	02:00:00	134	2.5	2.5	96.5	0.0
1.0							
3	2015-01-04	03:00:00	72	2.0	2.0	100.0	0.0
1.0							
4	2015-01-04	04:00:00	47	2.0	0.0	93.0	6.5
1.0							
...	
...							
17409	2017-01-03	19:00:00	1042	5.0	1.0	81.0	19.0
3.0							
17410	2017-01-03	20:00:00	541	5.0	1.0	81.0	21.0
4.0							
17411	2017-01-03	21:00:00	337	5.5	1.5	78.5	24.0
4.0							
17412	2017-01-03	22:00:00	224	5.5	1.5	76.0	23.0
4.0							
17413	2017-01-03	23:00:00	139	5.0	1.0	76.0	22.0
2.0							

	is_holiday	is_weekend	season
0	0.0	1.0	3.0
1	0.0	1.0	3.0
2	0.0	1.0	3.0
3	0.0	1.0	3.0
4	0.0	1.0	3.0
...
17409	0.0	0.0	3.0
17410	0.0	0.0	3.0
17411	0.0	0.0	3.0
17412	0.0	0.0	3.0
17413	0.0	0.0	3.0

[17414 rows x 10 columns]

bikes['weather_code'].value_counts()

weather_code	
1.0	6150
2.0	4034
3.0	3551
7.0	2141
4.0	1464
26.0	60

```

10.0      14
Name: count, dtype: int64

bikes['season'].value_counts()

season
0.0      4394
1.0      4387
3.0      4330
2.0      4303
Name: count, dtype: int64

# specifying the column names that I want to use
new_cols_dict = {
    'timestamp': 'time',
    'cnt': 'count',
    't1': 'temp_real_C',
    't2': 'temp_feels_like_C',
    'hum': 'humidity_percent',
    'wind_speed': 'wind_speed_kph',
    'weather_code': 'weather',
    'is_holiday': 'is_holiday',
    'is_weekend': 'is_weekend',
    'season': 'season'
}

# Renaming the columns to the specified column names
bikes.rename(new_cols_dict, axis=1, inplace=True)

# changing the humidity values to percentage (i.e. a value between 0
and 1)
bikes.humidity_percent = bikes.humidity_percent / 100

# creating a season dictionary so that we can map the integers 0-3 to
the actual written values
season_dict = {
    '0.0': 'spring',
    '1.0': 'summer',
    '2.0': 'autumn',
    '3.0': 'winter'
}

# creating a weather dictionary so that we can map the integers to the
actual written values
weather_dict = {
    '1.0': 'Clear',
    '2.0': 'Scattered clouds',
    '3.0': 'Broken clouds',
    '4.0': 'Cloudy',
    '7.0': 'Rain',
    '10.0': 'Rain with thunderstorm',

```

```

    '26.0': 'Snowfall'
}

# changing the seasons column data type to string
bikes.season = bikes.season.astype('str')
# mapping the values 0-3 to the actual written seasons
bikes.season = bikes.season.map(season_dict)

# changing the weather column data type to string
bikes.weather = bikes.weather.astype('str')
# mapping the values to the actual written weathers
bikes.weather = bikes.weather.map(weather_dict)

# checking our dataframe to see if the mappings have worked
bikes.head()

```

	time	count	temp_real_C	temp_feels_like_C	\
0	2015-01-04 00:00:00	182	3.0	2.0	
1	2015-01-04 01:00:00	138	3.0	2.5	
2	2015-01-04 02:00:00	134	2.5	2.5	
3	2015-01-04 03:00:00	72	2.0	2.0	
4	2015-01-04 04:00:00	47	2.0	0.0	

is_weekend	humidity_percent	wind_speed_kph	weather	is_holiday
0	0.930	6.0	Broken clouds	0.0
1.0				
1	0.930	5.0	Clear	0.0
1.0				
2	0.965	0.0	Clear	0.0
1.0				
3	1.000	0.0	Clear	0.0
1.0				
4	0.930	6.5	Clear	0.0
1.0				

season
0 winter
1 winter
2 winter
3 winter
4 winter

```

# writing the final dataframe to an excel file that we will use in our
Tableau visualisations. The file will be the 'london_bikes_final.xlsx'
file and the sheet name is 'Data'
bikes.to_excel('london_bikes_final.xlsx', sheet_name='Data')

```