Assignment 1

Data Wrangling I Perform the following operations using Python on any open source dataset (eg. data.csv)

1. Import all the required Python Libraries.
2. Locate an open source data from the web (eg. https://www.kaggle.com (https://www.kaggle.com)). Provide a clear description of the data and its source (i.e. URL of the web site).
3. Load the Dataset into pandas dataframe.
4. Data Preprocessing: check for missing values in the data using pandas isnull(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.
5. Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions.
6. Turn categorical variables into quantitative variables in Python

1.Import all the required Python Libraries.

In [58]:
```
import pandas as pd
import numpy as np
```

In [24]:
```
pwd
```

Out[24]:
```
'C:\\Users\\Admin'
```

2.Locate an open source data from the web (eg. https://www.kaggle.com (https://www.kaggle.com)).

Provide a clear description of the data and its source (i.e. URL of the web site).

In [59]:
```
df=pd. read_csv("C:\\Users\\Admin\\Desktop\\weather_data.csv")
```

In [60]:
```
print(df)
```

```
          day  temperature  windspeed   event  duration
0  01-01-2017         32.0        6.0    Rain         2
1  01-04-2017          NaN        9.0   Sunny         3
2  01-05-2017         28.0        NaN    Snow         4
3  01-06-2017          NaN        7.0     NaN         5
4  01-07-2017         32.0        NaN    Rain         7
5  01-08-2017          NaN        NaN   Sunny         5
6  01-09-2017          NaN        NaN     NaN         6
7  01-10-2017         34.0        8.0  Cloudy         4
8  01-11-2017         40.0       12.0   Sunny         2
```

In [83]:
```
df.head()
```

Out[83]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2.0
1	01-04-2017	33.2	9.0	Sunny	3.0
2	01-05-2017	28.0	33.2	Snow	4.0
3	01-06-2017	33.2	7.0	NaN	5.0
4	01-07-2017	32.0	33.2	Rain	7.0

In [84]:
```
df.tail()
```

Out[84]:

	day	temperature	windspeed	event	duration
4	01-07-2017	32.0	33.2	Rain	7.0
5	01-08-2017	33.2	33.2	Sunny	5.0
6	01-09-2017	33.2	33.2	NaN	6.0
7	01-10-2017	34.0	8.0	Cloudy	4.0
8	01-11-2017	40.0	12.0	Sunny	2.0

4.Data Preprocessing: check for missing values in the data using pandas isnull(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.

In [ ]:
```
4.a)Function: Describe():
        The  Describe function returns the statistical summary of the data frame or series.
```

In [85]:
```
df.describe()
```

Out[85]:

	temperature	windspeed	duration
count	9.000000	9.000000	9.000000
mean	33.200000	19.422222	4.222222
std	3.098387	13.171729	1.715938
min	28.000000	6.000000	2.000000
25%	32.000000	8.000000	3.000000
50%	33.200000	12.000000	4.000000
75%	33.200000	33.200000	5.000000
max	40.000000	33.200000	7.000000

4.b)Stepes for working with missing data 3.b.1. Identify misssing data 3.b.2. deal with missing data 3.b.3.correct data

4.b.1 identify missing data

A) convert the "?","blackspace" into NaN(non numerical data) function: replace() The replace () method replaces the specified value with another specified value. The replace() method searches the entire Data Frame and replaces every case of specified value.

In [61]:
```
df.replace("  ", np.nan, inplace = True)
df.head(9)
```

Out[61]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2
1	01-04-2017	NaN	9.0	Sunny	3
2	01-05-2017	28.0	NaN	Snow	4
3	01-06-2017	NaN	7.0	NaN	5
4	01-07-2017	32.0	NaN	Rain	7
5	01-08-2017	NaN	NaN	Sunny	5
6	01-09-2017	NaN	NaN	NaN	6
7	01-10-2017	34.0	8.0	Cloudy	4
8	01-11-2017	40.0	12.0	Sunny	2

B) Evaluating missing data The missing values are converted to python's default. Built in function: isnull(),.notnull() 1.isnull()-->we can use isnull() method to check whether a cell contains a numeric value(false) or if data is missing(true)

1. notnull()-->otnull() function detects existing/ non-missing values in the dataframe.

In [51]:
```
missingdata=df.isnull()
```

count missing values in each column

In [63]:
```
df.isnull().sum()
```

Out[63]:
```
day            0
temperature    4
windspeed      4
event          2
duration       0
dtype: int64
```

4.b.2 Deal with missing Data a. Drop data Functions:dropna() b. replace data calculate the mean()

DROPPING NULL OR MISSING VALUES This is the fastest and easiest step to handle missing values. However, it is not generally advised. This method reduces the quality of our model as it reduces sample size because it works by deleting all other observations where any of the variable is missing. The process can be done by: data_name.dropna()

In [64]:
```
df.dropna()
```

Out[64]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2
7	01-10-2017	34.0	8.0	Cloudy	4
8	01-11-2017	40.0	12.0	Sunny	2

It will be observed that of 10 entries will be reduced to 3 just by dropping NaN values.!!! Dropping is only advised to be used if missing values are few (say 0.01–0.5% of our data).

In [65]:
```
#original dataset is not changed
df
```

Out[65]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2
1	01-04-2017	NaN	9.0	Sunny	3
2	01-05-2017	28.0	NaN	Snow	4
3	01-06-2017	NaN	7.0	NaN	5
4	01-07-2017	32.0	NaN	Rain	7
5	01-08-2017	NaN	NaN	Sunny	5
6	01-09-2017	NaN	NaN	NaN	6
7	01-10-2017	34.0	8.0	Cloudy	4
8	01-11-2017	40.0	12.0	Sunny	2

In [67]:
```
#We can drop columns that have at least one NaN in any row by setting the axis argument
df.dropna(axis=1)
```

Out[67]:

	day	duration
0	01-01-2017	2
1	01-04-2017	3
2	01-05-2017	4
3	01-06-2017	5
4	01-07-2017	7
5	01-08-2017	5
6	01-09-2017	6
7	01-10-2017	4
8	01-11-2017	2

In [68]:
```
#The dropna() method has several additional parameters:
df.dropna(how='all')
```

Out[68]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2
1	01-04-2017	NaN	9.0	Sunny	3
2	01-05-2017	28.0	NaN	Snow	4
3	01-06-2017	NaN	7.0	NaN	5
4	01-07-2017	32.0	NaN	Rain	7
5	01-08-2017	NaN	NaN	Sunny	5
6	01-09-2017	NaN	NaN	NaN	6
7	01-10-2017	34.0	8.0	Cloudy	4
8	01-11-2017	40.0	12.0	Sunny	2

In [69]:
```
df.dropna(subset=['event'])
```

Out[69]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2
1	01-04-2017	NaN	9.0	Sunny	3
2	01-05-2017	28.0	NaN	Snow	4
4	01-07-2017	32.0	NaN	Rain	7
5	01-08-2017	NaN	NaN	Sunny	5
7	01-10-2017	34.0	8.0	Cloudy	4
8	01-11-2017	40.0	12.0	Sunny	2

Replace Data Calculate the mean and replace the null value by mean by using fillna() function

In [70]:
```
mean_value = df['temperature'].mean()
```

In [71]:
```
mean_value
```

Out[71]:
```
33.2
```

In [72]:
```
df['temperature'] = df['temperature'].fillna(mean_value)
```

In [73]:
```
df
```

Out[73]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2
1	01-04-2017	33.2	9.0	Sunny	3
2	01-05-2017	28.0	NaN	Snow	4
3	01-06-2017	33.2	7.0	NaN	5
4	01-07-2017	32.0	NaN	Rain	7
5	01-08-2017	33.2	NaN	Sunny	5
6	01-09-2017	33.2	NaN	NaN	6
7	01-10-2017	34.0	8.0	Cloudy	4
8	01-11-2017	40.0	12.0	Sunny	2

In [74]:
```
mean_v=df['windspeed'].mean()
```

In [44]:
```
mean_v
```

Out[44]:
```
8.4
```

In [75]:
```
df['windspeed'] = df['windspeed'].fillna(mean_value)
```

In [46]:
```
df
```

Out[46]:

	day	temperature	windspeed	event
0	01-01-2017	32.0	8.4	Rain
1	01-04-2017	33.2	8.4	Sunny
2	01-05-2017	28.0	8.4	Snow
3	01-06-2017	33.2	8.4	NaN
4	01-07-2017	32.0	8.4	Rain
5	01-08-2017	33.2	8.4	Sunny
6	01-09-2017	33.2	8.4	NaN
7	01-10-2017	34.0	8.4	Cloudy
8	01-11-2017	40.0	8.4	Sunny

let's say I want to replace all any values with zero and in my day not day but wind speed column I want to replace it with again zero but my event I want to say "No Event" and then print new data frame now as you can see here the temperature and wind speed is replaced with zero as you can see here but the event now I have no event

In [76]:
```
new_df = df.fillna({'temperature':0,
'windspeed':0,
'event':'no_event'})
```

In [77]:
```
new_df
```

Out[77]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2
1	01-04-2017	33.2	9.0	Sunny	3
2	01-05-2017	28.0	33.2	Snow	4
3	01-06-2017	33.2	7.0	no_event	5
4	01-07-2017	32.0	33.2	Rain	7
5	01-08-2017	33.2	33.2	Sunny	5
6	01-09-2017	33.2	33.2	no_event	6
7	01-10-2017	34.0	8.0	Cloudy	4
8	01-11-2017	40.0	12.0	Sunny	2

5.Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions. Function: dtypes()--> to check the data type astype() to change the data type

In [78]:
```
df.dtypes
```

Out[78]:
```
day             object
temperature    float64
windspeed      float64
event           object
duration         int64
dtype: object
```

In [79]:
```
df[["duration"]]=df[["duration"]].astype("float")
```

In [80]:
```
df.head()
```

Out[80]:

	day	temperature	windspeed	event	duration
0	01-01-2017	32.0	6.0	Rain	2.0
1	01-04-2017	33.2	9.0	Sunny	3.0
2	01-05-2017	28.0	33.2	Snow	4.0
3	01-06-2017	33.2	7.0	NaN	5.0
4	01-07-2017	32.0	33.2	Rain	7.0

6.Turn categorical variables into quantitative variables in Python Under this approach, we deploy the simplest way to perform the conversion of all possible Categorical Columns in a data frame to Dummy Columns by using the get_dummies() method of the pandas library.

We can either specify the columns to get the dummies by default it will convert all the possible categorical columns to their dummy columns.

In [81]:
```
# creating a copy of the original data frame
df3 = df.copy()
```

In [82]:
```
# calling the get_dummies method
# the first parameter mentions the
# the name of the data frame to store the
# new data frame in
# the second parameter is the list of
# columns which if not mentioned
# returns the dummies for all
# categorical columns

df3 = pd.get_dummies(df3,
                     columns = ['event'])

display(df3)
```

	day	temperature	windspeed	duration	event_Cloudy	event_Rain	event_Snow	event_Sunny
0	01-01-2017	32.0	6.0	2.0	0	1	0	0
1	01-04-2017	33.2	9.0	3.0	0	0	0	1
2	01-05-2017	28.0	33.2	4.0	0	0	1	0
3	01-06-2017	33.2	7.0	5.0	0	0	0	0
4	01-07-2017	32.0	33.2	7.0	0	1	0	0
5	01-08-2017	33.2	33.2	5.0	0	0	0	1
6	01-09-2017	33.2	33.2	6.0	0	0	0	0
7	01-10-2017	34.0	8.0	4.0	1	0	0	0
8	01-11-2017	40.0	12.0	2.0	0	0	0	1

In [ ]:
```
```