DataAnalysis1-Copy1

August 19, 2022

```
[1]: import pandas as pd
```

```
[2]: # data = pd.read_excel('Sample_sheet.xlsx')
     data = pd.read_csv('EmpDetails.csv')
     data
```

```
[2]:    Serial number      Name  Age  Gender
     0              1      John   34    Male
     1              2     Smith   25    Male
     2              3     Sarah   39  Female
     3              4  Angelina   42  Female
     4              5   Krishna   45    Male
     5              6      Jack   32    Male
     6              7      Lisa   67  Female
     7              8   Govinda   49    Male
     8              9     Diana   51  Female
     9             10     Trump   60    Male
```

```
[3]: print('Shape of data :', data.shape)
```

```
Shape of data : (10, 4)
```

```
[4]: print('Size of data :', data.size)
```

```
Size of data : 40
```

```
[5]: print('Data types : \n', data.dtypes)
```

```
Data types :
 Serial number     int64
Name             object
Age               int64
Gender           object
dtype: object
```

```
[6]: print('Statistical data analysis :\n',data.describe())
```

```
Statistical data analysis :
        Serial number        Age
count       10.00000  10.000000
mean         5.50000  44.400000
std          3.02765  12.877198
min          1.00000  25.000000
25%          3.25000  35.250000
50%          5.50000  43.500000
75%          7.75000  50.500000
max         10.00000  67.000000
```

[7]:

```
#Drop the column
data.drop('Serial number', axis=1)
```

[7]:

```
       Name  Age  Gender
0      John   34    Male
1     Smith   25    Male
2     Sarah   39  Female
3  Angelina   42  Female
4   Krishna   45    Male
5      Jack   32    Male
6      Lisa   67  Female
7   Govinda   49    Male
8     Diana   51  Female
9     Trump   60    Male
```

[8]:

```
#Drop the row
data.drop(0, axis=0)
```

[8]:

```
   Serial number      Name  Age  Gender
1              2     Smith   25    Male
2              3     Sarah   39  Female
3              4  Angelina   42  Female
4              5   Krishna   45    Male
5              6      Jack   32    Male
6              7      Lisa   67  Female
7              8   Govinda   49    Male
8              9     Diana   51  Female
9             10     Trump   60    Male
```

[9]:

```
# Permanently drop the column
data.drop('Serial number', axis = 1, inplace = True)
data
```

[9]:

```
       Name  Age  Gender
0      John   34    Male
1     Smith   25    Male
```

```
2      Sarah   39  Female
3   Angelina   42  Female
4    Krishna   45    Male
5       Jack   32    Male
6       Lisa   67  Female
7    Govinda   49    Male
8      Diana   51  Female
9      Trump   60    Male
```

[10]:

```
#List all values and unique values
print('All the values :', data['Gender'].values)
print('All the unique values :', data['Gender'].unique())
```

```
All the values : ['Male' 'Male' 'Female' 'Female' 'Male' 'Male' 'Female' 'Male'
'Female'
 'Male']
All the unique values : ['Male' 'Female']
```

[11]:

```
print('All the values :', data['Age'].values)
print('All the unique values :', data['Age'].unique())
```

```
All the values : [34 25 39 42 45 32 67 49 51 60]
All the unique values : [34 25 39 42 45 32 67 49 51 60]
```

[12]:

```
#Correlation of dataframe
data.corr()
```

[12]:

```
     Age
Age  1.0
```

[13]:

```
data
```

[13]:

```
        Name  Age  Gender
0       John   34    Male
1      Smith   25    Male
2      Sarah   39  Female
3   Angelina   42  Female
4    Krishna   45    Male
5       Jack   32    Male
6       Lisa   67  Female
7    Govinda   49    Male
8      Diana   51  Female
9      Trump   60    Male
```

Exercise : Add a column to your dataframe which will have negative correlation with Income and then find correlation

```
[14]: data['Salary'] = [10000, 7000, 12000, 18800, 20000, 12000, 0, 17690, 45000,␣
       ↪15000]
      data
```

```
[14]:        Name  Age  Gender  Salary
      0      John   34    Male   10000
      1     Smith   25    Male    7000
      2     Sarah   39  Female   12000
      3  Angelina   42  Female   18800
      4   Krishna   45    Male   20000
      5      Jack   32    Male   12000
      6      Lisa   67  Female       0
      7   Govinda   49    Male   17690
      8     Diana   51  Female   45000
      9     Trump   60    Male   15000
```

```
[15]: correlation = data.corr()
      correlation
```

```
[15]:              Age    Salary
      Age     1.000000  0.091306
      Salary  0.091306  1.000000
```

```
[16]: #Saving data to csv file
      correlation.to_csv('correlation.csv')
```

0.0.1 Missing Values

```
[17]: data = pd.read_csv('Sample_sheet.csv')
      data
```

```
[17]:    S. No.     Names  Years of Experience         Domain  Relevant Experience  \
      0       1      John                    8     Automotive                  6.0
      1       2     Jason                    5  Entertainment                  4.0
      2       3     Maria                   10        Banking                  3.0
      3       4     Jacob                   12      Insurance                 10.0
      4       5     Sarah                   15      Logistics                  5.0
      5       6  Angelina                    3         Travel                  3.0
      6       7   Krishna                    0            NaN                  NaN
      7       8      Adam                    8           Food                  5.0
      8       9   Deepika                   15             IT                 13.0
      9      10      Alan                    2          Space                  1.0

         Income(USD) Marital Status  Number of siblings
      0      20000.0         Single                   3
      1      15000.0        Married                   3
```

```
2      18000.0         Single                   3
3      24000.0         Single                   3
4       8000.0        Married                   3
5       9500.0        Married                   3
6          NaN         Single                   3
7       7500.0         Single                   3
8      19500.0        Married                   3
9      12500.0        Married                   3
```

[18]:

```
# Detection of missing values

data.isna().any()
```

[18]:

```
S. No.                 False
Names                  False
Years of Experience    False
Domain                  True
Relevant Experience     True
Income(USD)             True
Marital Status         False
Number of siblings     False
dtype: bool
```

[19]:

```
# Number of missing values across columns
data.isna().sum()
```

[19]:

```
S. No.                 0
Names                  0
Years of Experience    0
Domain                 1
Relevant Experience    1
Income(USD)            1
Marital Status         0
Number of siblings     0
dtype: int64
```

Treatment of missing values

[20]:

```
#Using a constant value
```

[21]:

```
data['Domain'].fillna('Government')
```

[21]:

```
0       Automotive
1    Entertainment
2          Banking
3        Insurance
4        Logistics
```

```
5           Travel
6       Government
7             Food
8               IT
9            Space
Name: Domain, dtype: object
```

[22]:
```
data['Income(USD)'].fillna(3000)
```

[22]:
```
0    20000.0
1    15000.0
2    18000.0
3    24000.0
4     8000.0
5     9500.0
6     3000.0
7     7500.0
8    19500.0
9    12500.0
Name: Income(USD), dtype: float64
```

[23]:
```
#Using a mean of the series

data['Income(USD)'].fillna(data['Income(USD)'].mean())
```

[23]:
```
0    20000.000000
1    15000.000000
2    18000.000000
3    24000.000000
4     8000.000000
5     9500.000000
6    14888.888889
7     7500.000000
8    19500.000000
9    12500.000000
Name: Income(USD), dtype: float64
```

[24]:
```
#Using a median of the series

data['Income(USD)'].fillna(data['Income(USD)'].median())
```

[24]:
```
0    20000.0
1    15000.0
2    18000.0
3    24000.0
4     8000.0
5     9500.0
```

```
6    15000.0
7     7500.0
8    19500.0
9    12500.0
Name: Income(USD), dtype: float64
```

[25]:

```
from sklearn.impute import SimpleImputer
imp_constant = SimpleImputer(strategy='constant', fill_value=12345)
imp_mean = SimpleImputer(strategy='mean')
imp_median = SimpleImputer(strategy='median')
imp_mode = SimpleImputer(strategy='most_frequent')
```

[26]:

```
imp_constant.fit_transform(data[['Income(USD)']])
```

[26]:

```
array([[20000.],
       [15000.],
       [18000.],
       [24000.],
       [ 8000.],
       [ 9500.],
       [12345.],
       [ 7500.],
       [19500.],
       [12500.]])
```

[27]:

```
imp_mean.fit_transform(data[['Income(USD)']])
```

[27]:

```
array([[20000.        ],
       [15000.        ],
       [18000.        ],
       [24000.        ],
       [ 8000.        ],
       [ 9500.        ],
       [14888.88888889],
       [ 7500.        ],
       [19500.        ],
       [12500.        ]])
```

[28]:

```
imp_median.fit_transform(data[['Income(USD)']])
```

[28]:

```
array([[20000.],
       [15000.],
       [18000.],
       [24000.],
       [ 8000.],
       [ 9500.],
       [15000.],
       [ 7500.],
       [19500.],
       [12500.]])
```

```
[29]: imp_mode.fit_transform(data[['Income(USD)']])
```

```
[29]: array([[20000.],
             [15000.],
             [18000.],
             [24000.],
             [ 8000.],
             [ 9500.],
             [ 7500.],
             [ 7500.],
             [19500.],
             [12500.]])
```

```
[30]: # Permanent replacement of missing values
      data['Income(USD)'] = imp_mode.fit_transform(data[['Income(USD)']])
      data
```

```
[30]:    S. No.     Names  Years of Experience         Domain  Relevant Experience  \
      0       1      John                    8     Automotive                  6.0
      1       2     Jason                    5  Entertainment                  4.0
      2       3     Maria                   10        Banking                  3.0
      3       4     Jacob                   12      Insurance                 10.0
      4       5     Sarah                   15      Logistics                  5.0
      5       6  Angelina                    3         Travel                  3.0
      6       7   Krishna                    0            NaN                  NaN
      7       8      Adam                    8           Food                  5.0
      8       9   Deepika                   15             IT                 13.0
      9      10      Alan                    2          Space                  1.0

         Income(USD) Marital Status  Number of siblings
      0      20000.0         Single                   3
      1      15000.0        Married                   3
      2      18000.0         Single                   3
      3      24000.0         Single                   3
      4       8000.0        Married                   3
      5       9500.0        Married                   3
      6       7500.0         Single                   3
      7       7500.0         Single                   3
      8      19500.0        Married                   3
      9      12500.0        Married                   3
```

```
[31]: # Row with the missing values
      data[data['Domain'].isna()]
```

```
[31]:    S. No.    Names  Years of Experience Domain  Relevant Experience  \
      6       7  Krishna                    0    NaN                  NaN

         Income(USD) Marital Status  Number of siblings
      6       7500.0         Single                   3
```

```
[32]: data
```

```
[32]:    S. No.     Names  Years of Experience         Domain  Relevant Experience  \
      0       1      John                    8     Automotive                  6.0
      1       2     Jason                    5  Entertainment                  4.0
      2       3     Maria                   10        Banking                  3.0
      3       4     Jacob                   12      Insurance                 10.0
      4       5     Sarah                   15      Logistics                  5.0
      5       6  Angelina                    3         Travel                  3.0
      6       7   Krishna                    0            NaN                  NaN
      7       8      Adam                    8           Food                  5.0
      8       9   Deepika                   15             IT                 13.0
      9      10      Alan                    2          Space                  1.0

         Income(USD) Marital Status  Number of siblings
      0      20000.0         Single                   3
      1      15000.0        Married                   3
      2      18000.0         Single                   3
      3      24000.0         Single                   3
      4       8000.0        Married                   3
      5       9500.0        Married                   3
      6       7500.0         Single                   3
      7       7500.0         Single                   3
      8      19500.0        Married                   3
      9      12500.0        Married                   3
```

```
[33]: data_ = pd.DataFrame({'Name':['Krishna', 'Adam', 'Adam', 'Alan', 'Krishna'],
                            'DOB':['January', 'March', 'March', 'May', 'December'],
                   'Age':[29, 44, 45, 12, 39]})
```

```
[34]: data_
```

```
[34]:       Name       DOB  Age
      0  Krishna   January   29
      1     Adam     March   44
      2     Adam     March   45
      3     Alan       May   12
      4  Krishna  December   39
```

```
[35]: data_['Name']
```

```
[35]: 0    Krishna
      1       Adam
      2       Adam
      3       Alan
      4    Krishna
      Name: Name, dtype: object
```

```
[36]: data_['Name'].duplicated()
```

```
[36]: 0    False
      1    False
      2     True
      3    False
      4     True
      Name: Name, dtype: bool
```

```
[37]: data_['Name'].drop_duplicates()
```

```
[37]: 0    Krishna
      1       Adam
      3       Alan
      Name: Name, dtype: object
```

```
[38]: data_.drop_duplicates()
```

```
[38]:       Name       DOB  Age
      0  Krishna   January   29
      1     Adam     March   44
      2     Adam     March   45
      3     Alan       May   12
      4  Krishna  December   39
```

```
[39]: data_
```

```
[39]:       Name       DOB  Age
      0  Krishna   January   29
      1     Adam     March   44
      2     Adam     March   45
      3     Alan       May   12
      4  Krishna  December   39
```

```
[40]: data_.drop_duplicates(subset=['Name','DOB'])
```

```
[40]:       Name       DOB  Age
      0  Krishna   January   29
      1     Adam     March   44
      3     Alan       May   12
```

```
4  Krishna  December   39
```

[41]:
```
data.nunique()
```

[41]:
```
S. No.                  10
Names                   10
Years of Experience      8
Domain                   9
Relevant Experience      7
Income(USD)              9
Marital Status           2
Number of siblings       1
dtype: int64
```

[42]:
```
#Loading the sklearn inbuilt datasets

from sklearn.datasets import load_iris
```

[43]:
```
iris_data = load_iris()
```

[44]:
```
#Description of the data
print(iris_data.DESCR)
```

```
.. _iris_dataset:

Iris plants dataset
--------------------

**Data Set Characteristics:**

    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, predictive attributes and the class
    :Attribute Information:
        - sepal length in cm
        - sepal width in cm
        - petal length in cm
        - petal width in cm
        - class:
                - Iris-Setosa
                - Iris-Versicolour
                - Iris-Virginica

    :Summary Statistics:

    ============== ==== ==== ======= ===== ====================
                    Min  Max   Mean    SD   Class Correlation
    ============== ==== ==== ======= ===== ====================
    sepal length:   4.3  7.9   5.84   0.83    0.7826
    sepal width:    2.0  4.4   3.05   0.43   -0.4194
    petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
    ============== ==== ==== ======= ===== ====================

    :Missing Attribute Values: None
    :Class Distribution: 33.3% for each of 3 classes.
    :Creator: R.A. Fisher
    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
    :Date: July, 1988

The famous Iris database, first used by Sir R.A. Fisher. The dataset is taken
from Fisher's paper. Note that it's the same as in R, but not as in the UCI
Machine Learning Repository, which has two wrong data points.

This is perhaps the best known database to be found in the
pattern recognition literature.  Fisher's paper is a classic in the field and
is referenced frequently to this day.  (See Duda & Hart, for example.)  The
data set contains 3 classes of 50 instances each, where each class refers to a
type of iris plant.  One class is linearly separable from the other 2; the
latter are NOT linearly separable from each other.

.. topic:: References

   - Fisher, R.A. "The use of multiple measurements in taxonomic problems"
     Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to
     Mathematical Statistics" (John Wiley, NY, 1950).
   - Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysis.
     (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.
   - Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System
     Structure and Classification Rule for Recognition in Partially Exposed
     Environments".  IEEE Transactions on Pattern Analysis and Machine
     Intelligence, Vol. PAMI-2, No. 1, 67-71.
   - Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule".  IEEE Transactions
     on Information Theory, May 1972, 431-433.
   - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et al"s AUTOCLASS II
     conceptual clustering system finds 3 classes in the data.
   - Many, many more …
```

[47]:

```
#Actual data
pd.DataFrame(iris_data.data, columns= iris_data.feature_names)
```

[47]:

```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0                1.4               0.2
2                  4.7               3.2                1.3               0.2
```

```
3                  4.6                3.1                1.5               0.2
4                  5.0                3.6                1.4               0.2
..                 …                …                …               …
145                6.7                3.0                5.2               2.3
146                6.3                2.5                5.0               1.9
147                6.5                3.0                5.2               2.0
148                6.2                3.4                5.4               2.3
149                5.9                3.0                5.1               1.8

[150 rows x 4 columns]
```

[ ]:

```
#Feature Names
iris_data.feature_names
```

[ ]:

```
#Target data
iris_data.target
```

[ ]:

```
#Target name
iris_data.target_names
```

Exercise : Perform the reading, exploration, detection, treatment and saving of the iris data