15Z332 Ex1 - DescriptiveAnalysis

October 17, 2018

1 Exercise 1

1.1 Perform descriptive analysis on EmployeeAttrition dataset

The data set contains 1470 records and 35 attributes. The Target attribute is Attrition, which contains two values ('Yes' and 'No')

In [22]:
```
import pandas as pd

#Importing the dataset
df = pd.read_csv('EmployeeAttrition.csv')
print(df.shape)
df.head(5)
```

```
(1470, 35)
```

Out[22]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipSatisfaction	StandardHours
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	1	80
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	4	80
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	2	80
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	3	80
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	4	80

```
   StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  \
0                 0                  8                      0               1
1                 1                 10                      3               3
2                 0                  7                      3               3
3                 0                  8                      3               3
4                 1                  6                      3               3

   YearsAtCompany YearsInCurrentRole  YearsSinceLastPromotion  \
0               6                  4                        0
1              10                  7                        1
2               0                  0                        0
3               8                  7                        3
4               2                  2                        2

   YearsWithCurrManager
0                     5
1                     7
2                     0
3                     0
4                     2

[5 rows x 35 columns]
```

describe() function gives the aggregate analysis of the dataset, such as count, max, min and mean.

```
In [23]: df.describe()
```

Out[23]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0
mean	36.923810	802.485714	9.192517	2.912925	1.0
std	9.135373	403.509100	8.106864	1.024165	0.0
min	18.000000	102.000000	1.000000	1.000000	1.0
25%	30.000000	465.000000	2.000000	2.000000	1.0
50%	36.000000	802.000000	7.000000	3.000000	1.0
75%	43.000000	1157.000000	14.000000	4.000000	1.0
max	60.000000	1499.000000	29.000000	5.000000	1.0

	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement
count	1470.000000	1470.000000	1470.000000	1470.000000
mean	1024.865306	2.721769	65.891156	2.729932
std	602.024335	1.093082	20.329428	0.711561
min	1.000000	1.000000	30.000000	1.000000
25%	491.250000	2.000000	48.000000	2.000000
50%	1020.500000	3.000000	66.000000	3.000000
75%	1555.750000	4.000000	83.750000	3.000000
max	2068.000000	4.000000	100.000000	4.000000

```
          JobLevel          ...          RelationshipSatisfaction  \
count  1470.000000          ...                       1470.000000
mean      2.063946          ...                          2.712245
std       1.106940          ...                          1.081209
min       1.000000          ...                          1.000000
25%       1.000000          ...                          2.000000
50%       2.000000          ...                          3.000000
75%       3.000000          ...                          4.000000
max       5.000000          ...                          4.000000

       StandardHours  StockOptionLevel  TotalWorkingYears  \
count         1470.0       1470.000000        1470.000000
mean            80.0          0.793878          11.279592
std              0.0          0.852077           7.780782
min             80.0          0.000000           0.000000
25%             80.0          0.000000           6.000000
50%             80.0          1.000000          10.000000
75%             80.0          1.000000          15.000000
max             80.0          3.000000          40.000000

       TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
count            1470.000000      1470.000000     1470.000000
mean                2.799320         2.761224        7.008163
std                 1.289271         0.706476        6.126525
min                 0.000000         1.000000        0.000000
25%                 2.000000         2.000000        3.000000
50%                 3.000000         3.000000        5.000000
75%                 3.000000         3.000000        9.000000
max                 6.000000         4.000000       40.000000

       YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
count         1470.000000              1470.000000           1470.000000
mean             4.229252                 2.187755              4.123129
std              3.623137                 3.222430              3.568136
min              0.000000                 0.000000              0.000000
25%              2.000000                 0.000000              2.000000
50%              3.000000                 1.000000              3.000000
75%              7.000000                 3.000000              7.000000
max             18.000000                15.000000             17.000000

[8 rows x 26 columns]
```

We count the number of missing values in each attribute. If there were any empty cells, appropriate methods to handle them should be administered (in case if the data type is numerical, we can fill with mean or median, and if the data type is categorical, we can fill with mode).

```
In [24]: print(len(df))
         df.isnull().sum()
```

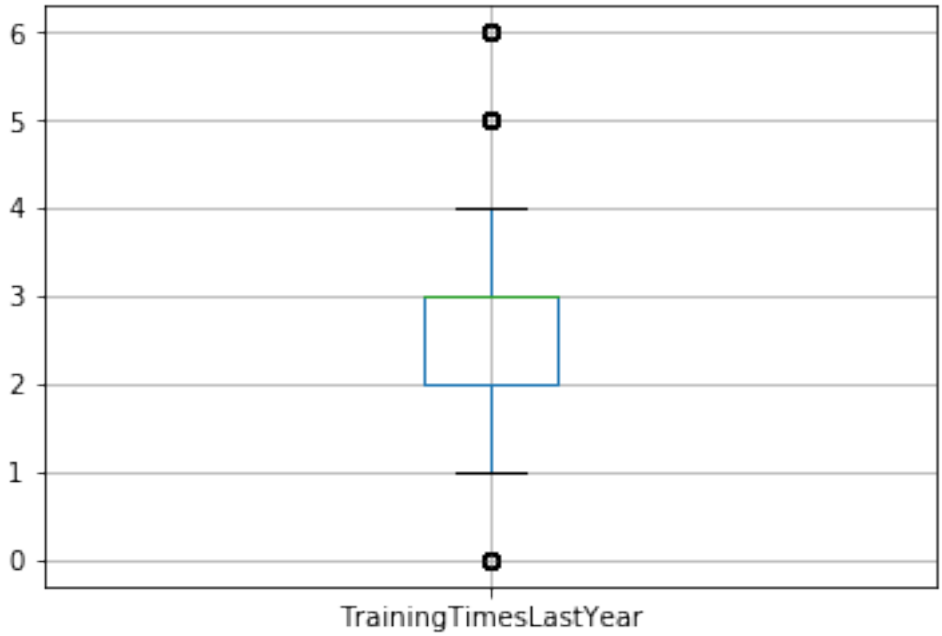
```
1470
```

```
Out[24]: Age                         0
         Attrition                   0
         BusinessTravel              0
         DailyRate                   0
         Department                  0
         DistanceFromHome            0
         Education                   0
         EducationField              0
         EmployeeCount               0
         EmployeeNumber              0
         EnvironmentSatisfaction     0
         Gender                      0
         HourlyRate                  0
         JobInvolvement              0
         JobLevel                    0
         JobRole                     0
         JobSatisfaction             0
         MaritalStatus               0
         MonthlyIncome               0
         MonthlyRate                 0
         NumCompaniesWorked          0
         Over18                      0
         OverTime                    0
         PercentSalaryHike           0
         PerformanceRating           0
         RelationshipSatisfaction    0
         StandardHours               0
         StockOptionLevel            0
         TotalWorkingYears           0
         TrainingTimesLastYear       0
         WorkLifeBalance             0
         YearsAtCompany              0
         YearsInCurrentRole          0
         YearsSinceLastPromotion     0
         YearsWithCurrManager        0
         dtype: int64
```

EmployeeAttrition.csv doesn't have any empty vales. But this doesn't mean that the dataset is clean; it may contain noise data. The behaviour of each data can be analysed using graphs and plot.

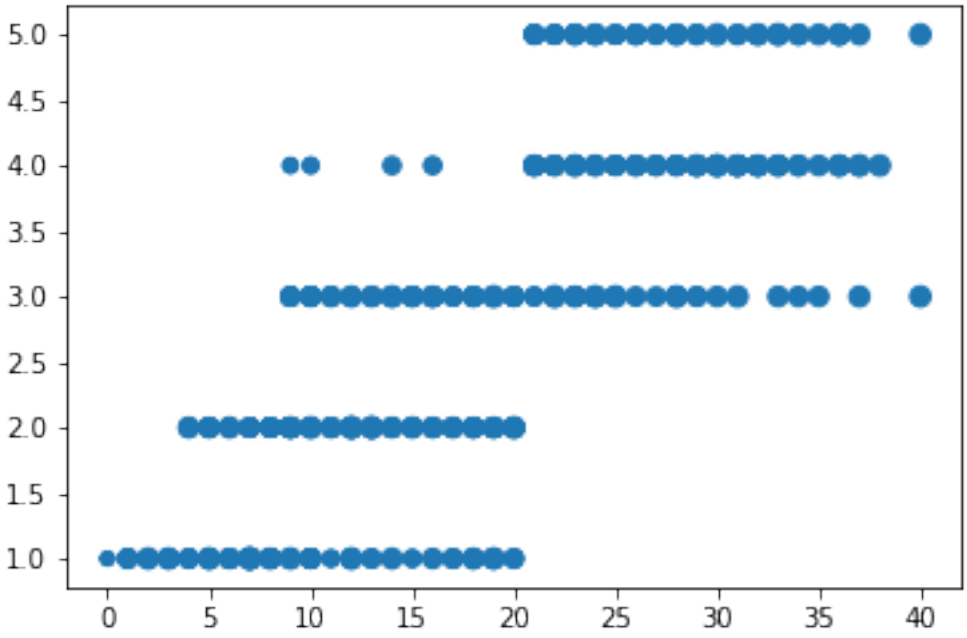
For example, a box plot is drawn for the attribute 'TrainingTimesLastYear'. Box plots can only be drawn for numerical datatype. This box plot denotes that the average range of training times for an employee is 1 to 4. Some employees lie beyond this range, which can be considered as outliers. With this, we may conclude that some employee with more training (like 5 or 6 times) may be cost inefficient to the company, hence these employees may not be valuable.

```
In [25]: import matplotlib.pyplot as plt
         df.boxplot('TrainingTimesLastYear')

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x11422c9b0>
```



A scatter plot between 'TotalWorkingYears' and 'JobLevel' is drawn. We can see that there is a positive correlation between the two attributes. If an employee has been working for more number of years, they are higher up in the job level. These employes may also be more valuable to the company.

```
In [26]: plt.scatter(df.TotalWorkingYears,df.JobLevel,s=df.Age)

Out[26]: <matplotlib.collections.PathCollection at 0x1146d5080>
```


5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0
5
10
15
20
25
30
35
40