In [1]:
```
##Import packages needed for data loading and processing

import pandas as pd
import numpy as np
```

In [2]:
```
## 0. Acquire a dataset of your interest.
# Human Resource Analytics Dataset Kaggle HR Analytics Dataset with Missing Values and is_smoker Feature

hrdb = pd.read_csv("hr_data.csv")

hrdb.head(3)
```

Out[2]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	work_accident	left	promotion_last_5years
0	0.38	0.53	2	157.0	3.0	0	yes	(
1	0.80	0.86	5	262.0	6.0	0	yes	(
2	0.11	0.88	7	272.0	4.0	0	yes	(

1. Write up the Abstract of your homework topic and a concise description the dataset to be analized. • Title: Human Resource Analytics Dataset The dataset is about human resources. Aim is to answer an interesting question of a company such as "Why are our best and most experienced employees leaving prematurely?" However, for this excercise, will stick to the scope of homework to deal with missing data. This dataset is generated by analyzing answers of the employees to the job satisfaction survey and their work related records. The dataset is formed by the Human Resources (HR) department after conducting a survey on their employees. In this study we first run an Explanatory Data Analysis (EDA) on tha data to make it more meaningfull, • Resource links and the download address of the dataset: https://www.kaggle.com/cezarschroeder/human-resource-analytics-dataset • A copy of the dataset file used is to be submitted with the Jupyter Notebook report: will be attched while uploading homework

1. Create a full list of fields and the description for each attribute

This HR data set is obtained from the results of a satisfaction survey the company has carried out on their employees in combination with other HR related records. It consists of 14999 rows and 11 columns. Each row is dedicated for a different employee. Out of 11, 5 columns are in numeric type, while the remaining 6 are non-numeric values. Below you can find columns and their explanations, respectively.

Our dataset contains the following observations: satisfaction_level: A numeric evaluation by the employee on the scale of 0 to 1 being 1 is highly statified and 0 is not staisfied. last_evaluation: A numeric evaluation graded by the employee's manager. number_project: A integer - the number of projects the employee has been involved. average_monthly_hours: The number of hours they work (billed) in the month. time_spend_company: An integer value, length of service of the employee. Work_accident: Boolean value, whether or not they had an accident. left: if the employee leave or not. promoted_last_5years: boolean value, if the employee was promoted at least once in last 5 years. is_smoker: if employee was smoker. department: Name of the department where the employees are assign. salary: A 3-level pay grade indicator (low, medium, high).

In [3]:
```
hrdb.describe()
```

Out[3]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	work_accident	promotion_last_5years
count	14999.000000	14999.000000	14999.000000	14631.000000	14848.000000	14999.000000	14999.000000
mean	0.612834	0.716102	3.803054	200.958376	3.494141	0.144610	0.021268
std	0.248631	0.171169	1.232592	50.002307	1.458976	0.351719	0.144281
min	0.090000	0.360000	2.000000	96.000000	2.000000	0.000000	0.000000
25%	0.440000	0.560000	3.000000	156.000000	3.000000	0.000000	0.000000
50%	0.640000	0.720000	4.000000	200.000000	3.000000	0.000000	0.000000
75%	0.820000	0.870000	5.000000	245.000000	4.000000	0.000000	0.000000
max	1.000000	1.000000	7.000000	310.000000	10.000000	1.000000	1.000000

In [4]:
```
# Checking data types

hrdb.dtypes
```

Out[4]:
```
satisfaction_level       float64
last_evaluation          float64
number_project             int64
average_montly_hours     float64
time_spend_company       float64
work_accident              int64
```

```
left                        object
promotion_last_5years        int64
is_smoker                   object
department                  object
salary                      object
dtype: object
```

In [5]:

```
## Looking for missing values in the dataset

count_NA = hrdb.isna().sum()
count_NA
```

Out[5]:

```
satisfaction_level           0
last_evaluation              0
number_project               0
average_montly_hours       368
time_spend_company         151
work_accident                0
left                         0
promotion_last_5years        0
is_smoker                14764
department                   0
salary                       0
dtype: int64
```

1. Handle Missing Data There are 3 columns, average_montly_hours, time_spend_company and is_smoker having missing values.

'average_montly_hours': Normally, average working days of a month are 21, and working hrs of a day are 8, so average monthly hrs will be 21*8 = 168 hrs. These are the minimum hrs every person is expected to work. Hence, we will fill 168 hrs in case of missing values in this column. 'time_spend_company': Mode is the frequently occurring number of the dataset. For missing values in this column, it is logical to say that employee must have been stayed with the company for those many years. So will replace missing value with the mode of this column (which is 3 years). 'is_smoker': we do not need this column, as it is not a determining factor to find out why employees are leaving the company, hence we will delete it.

In [6]:

```
# Creating a copy of a database to proceed further

hrdb1 = hrdb.copy()
hrdb1
```

Out[6]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	work_accident	left	promotion_last_5
0	0.38	0.53	2	157.0	3.0	0	yes	
1	0.80	0.86	5	262.0	6.0	0	yes	
2	0.11	0.88	7	272.0	4.0	0	yes	
3	0.72	0.87	5	223.0	5.0	0	yes	
4	0.37	0.52	2	NaN	NaN	0	yes	
...	...	...	...	...	...	...	...	
14994	0.40	0.57	2	151.0	3.0	0	yes	
14995	0.37	0.48	2	160.0	3.0	0	yes	
14996	0.37	0.53	2	143.0	3.0	0	yes	
14997	0.11	0.96	6	280.0	4.0	0	yes	
14998	0.37	0.52	2	158.0	3.0	0	yes	

14999 rows × 11 columns

In [7]:

```
# Removing 'is_smoker'
del hrdb1['is_smoker']
```

In [8]:

```
# checking if the column 'is_smoker' is dropped.
hrdb1.columns

# we dont see column 'is_smoker' in the dataset.
```

Out[8]:

```
Index(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_montly_hours', 'time_spend_company', 'work_accident', 'left',
       'promotion_last_5years', 'department', 'salary'],
      dtype='object')
```

In [9]:
```python
# Filling 'time_spend_company' missing values with its Mode

#calculate mode

md = hrdb.mode()['time_spend_company'][0]
md
print("Mode of column 'time_spend_company' is", md, "years.")
```

```
Mode of column 'time_spend_company' is 3.0 years.
```

In [10]:
```python
# replacing missing values with mode = 3 years

hrdb1.time_spend_company = hrdb1.time_spend_company.fillna(md)
```

In [11]:
```python
# checking if there are any missing values in column 'time_spend_company'

count_timespent = hrdb1['time_spend_company'].isna().sum()

print("There are", count_timespent, "missing values in column 'time_spend_company'.")
```

```
There are 0 missing values in column 'time_spend_company'.
```

In [12]:
```python
hrdb1.time_spend_company.describe()

# check below result, now the count is 14999, earlier it was 14848 with 151 missing values
```

Out[12]:
```
count    14999.000000
mean         3.489166
std          1.452451
min          2.000000
25%          3.000000
50%          3.000000
75%          4.000000
max         10.000000
Name: time_spend_company, dtype: float64
```

In [13]:
```python
# replacing missing values of column 'average_montly_hours' with 168 hrs

hrdb1.average_montly_hours = hrdb1.average_montly_hours.fillna(168)
```

In [14]:
```python
count_monthlyhrs = hrdb1['average_montly_hours'].isna().sum()

print("There are", count_monthlyhrs, "missing values in column 'average_montly_hours'.")
```

```
There are 0 missing values in column 'average_montly_hours'.
```

In [15]:
```python
hrdb1.average_montly_hours.describe()
# check below result, now the count is 14999, earlier it was 14631 with 368 missing values
```

Out[15]:
```
count    14999.000000
mean       200.149743
std         49.647584
min         96.000000
25%        156.000000
50%        197.000000
75%        244.000000
max        310.000000
Name: average_montly_hours, dtype: float64
```

In [16]:
```python
# Converting categorical features to binary integer representation and to one-hot encoding
# making a copy for further analysis

hrdb2 = hrdb1.copy()
hrdb2.shape
```

Out[16]:
```
(14999, 10)
```

In [17]:
```python
# validating if there are still any missing values

count_MV = hrdb2.isna().sum()
count_MV
```

Out[17]:
```
satisfaction_level       0
last_evaluation          0
number_project           0
```

```
average_montly_hours     0
time_spend_company       0
work_accident            0
left                     0
promotion_last_5years    0
department               0
salary                   0
```

In [18]:

```
## Converting categorical features to binary integer representation and to one-hot encoding

hrdb2.left = hrdb2.left.map({'no': 0, 'yes': 1})
hrdb2 = pd.get_dummies(hrdb2)
hrdb2
```

Out[18]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	work_accident	left	promotion_last_5
0	0.38	0.53	2	157.0	3.0	0	1	
1	0.80	0.86	5	262.0	6.0	0	1	
2	0.11	0.88	7	272.0	4.0	0	1	
3	0.72	0.87	5	223.0	5.0	0	1	
4	0.37	0.52	2	168.0	3.0	0	1	
...	...	...	...	...	...	...	...	
14994	0.40	0.57	2	151.0	3.0	0	1	
14995	0.37	0.48	2	160.0	3.0	0	1	
14996	0.37	0.53	2	143.0	3.0	0	1	
14997	0.11	0.96	6	280.0	4.0	0	1	
14998	0.37	0.52	2	158.0	3.0	0	1	

14999 rows × 21 columns

1. Handle Outliners

Outliers are decided based on 2 criteria, first by specification limits which are determined by experts and second by statistics using standard deviation. In this case we will use specification limits as explained below:

As explained earlier, average person can work only 168 hrs per month (21 days per month * 8 hrs per day). So, we need to investigate why employees have logged their time more than 168 hrs per month or even more than 200 hrs. These are the cases of overutilization of the resources resulting into burnout.

Also, data to be investigated where average monthly hrs are less that 150 hrs. Which is a case of underutilization as company is paying full month salary but resource is not utilized up to minimum expected hrs.

Any values in the column 'average_montly_hours' less than 150 hrs or greater than 200 hrs are the outliers. Both are the reasons for employees to leave the company being underutilized and over utilized. Here USL (uppser specification limit) = 200 and LSL (lower specification limit) = 150, anything beyond these values are outliers.

In [19]:

```
# Filter all rows with column 'average_montly_hours' has value more than 150 and less than or equal to 200 hrs

filter_rows = hrdb2[(hrdb2.average_montly_hours > 150) & (hrdb2.average_montly_hours <=200)]

filter_rows.average_montly_hours.describe()

# From below summary statistics we can see that Max value is 200 and min is 151 which is this dataset no longer conta
```

Out[19]:

```
count    4727.000000
mean      172.534165
std        14.281688
min       151.000000
25%       160.000000
50%       169.000000
75%       185.000000
max       200.000000
Name: average_montly_hours, dtype: float64
```

1. Summary and Conclusion

After filtering out employees those were underutilized and over utilized, we are left with 4727 data point. Now we will find out how many of these are happy to be part of this company. So, we create the bins of satisfaction as below: 0.20, 0.60, 0.80 and 1.00. We group them with names Dissatisfied, Somewhat Satisfied, Satisfied and Highly Satisfied. Company needs to take detailed feedback on these respective groups on what is going right and what is going wrong to improve employee retention.

In [20]:
```
# Bining

bins = [0.2, 0.4, 0.6, 0.8, 1.0]

cats = pd.cut(filter_rows['satisfaction_level'], bins)
cats.value_counts()
```

Out[20]:
```
(0.6, 0.8]    1383
(0.4, 0.6]    1290
(0.8, 1.0]    1271
(0.2, 0.4]     567
Name: satisfaction_level, dtype: int64
```

In [21]:
```
## Display the staisfaction rank of all the employees in resulting dataset

bins = [0.2, 0.4, 0.6, 0.8, 1.0]
group_names = ['Dissatisfied', 'Somewhat Satisfied', 'Satisfied', 'Highly Satisfied']
grp_names = pd.cut(filter_rows['satisfaction_level'], 4, labels=group_names)
grp_names
```

Out[21]:
```
0        Somewhat Satisfied
4        Somewhat Satisfied
5        Somewhat Satisfied
16       Somewhat Satisfied
18       Somewhat Satisfied
                ...
14989    Somewhat Satisfied
14992    Somewhat Satisfied
14994    Somewhat Satisfied
14995    Somewhat Satisfied
14998    Somewhat Satisfied
Name: satisfaction_level, Length: 4727, dtype: category
Categories (4, object): ['Dissatisfied' < 'Somewhat Satisfied' < 'Satisfied' < 'Highly Satisfied']
```

In [ ]: