#Project HR Analytics Project.#

#Describe:# This case study aims to model the probability of attrition of each employee from the HR Analytics Dataset, available on Kaggle. Its conclusions will allow the management to understand which factors urge the employees to leave the company and which changes should be made to avoid their departure.

All the files of this project are saved in a GitHub repository. link: https://github.com/sameerCoder/DATA_ANALYST_DATASETS/tree/main/HrAnalytics

```
# from google.colab import drive
#drive.mount('/content/drive')
```

Importing Libraries and Dataset

```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
```

```
# reading the data

train = 
pd.read_csv('https://raw.githubusercontent.com/sameerCoder/DATA_ANALYS
T_DATASETS/main/HrAnalytics/HrAnalytics_train.csv')
test = 
pd.read_csv('https://raw.githubusercontent.com/sameerCoder/DATA_ANALYS
T_DATASETS/main/HrAnalytics/HrAnalytics_test.csv')

# getting their shapes
print("Shape of train :", train.shape)
print("Shape of test :", test.shape)
```

```
Shape of train : (54808, 14)
Shape of test : (23490, 13)
```

Data Analysis

```
# print head of train and test df
print(train.head())
```

```
   employee_id         department     region         education gender 
\
0        65438  Sales & Marketing   region_7  Master's & above      f   

1        65141         Operations  region_22        Bachelor's      m   

2         7513  Sales & Marketing  region_19        Bachelor's      m   

3         2542  Sales & Marketing  region_23        Bachelor's      m   
```

```
4        48945         Technology  region_26        Bachelor's      m


  recruitment_channel  no_of_trainings  age  previous_year_rating  \
0            sourcing                1   35                   5.0
1               other                1   30                   5.0
2            sourcing                1   34                   3.0
3               other                2   39                   1.0
4               other                1   45                   3.0

   length_of_service  KPIs_met >80%  awards_won?
avg_training_score  \
0                  8              1            0                  49

1                  4              0            0                  60

2                  7              0            0                  50

3                 10              0            0                  50

4                  2              0            0                  73


   is_promoted
0            0
1            0
2            0
3            0
4            0
```

```
print(test.head())
```

```
   employee_id         department     region   education gender  \
0         8724         Technology  region_26  Bachelor's      m
1        74430                 HR   region_4  Bachelor's      f
2        72255  Sales & Marketing  region_13  Bachelor's      m
3        38562        Procurement   region_2  Bachelor's      f
4        64486            Finance  region_29  Bachelor's      m

  recruitment_channel  no_of_trainings  age  previous_year_rating  \
0            sourcing                1   24                   NaN
1               other                1   31                   3.0
2               other                1   31                   1.0
3               other                3   31                   2.0
4            sourcing                1   30                   4.0

   length_of_service  KPIs_met >80%  awards_won?  avg_training_score
0                  1              1            0                  77
```

```
1                  5              0            0                  51
2                  4              0            0                  47
3                  9              0            0                  65
4                  7              0            0                  61
```

```
# describing the training set . all columns should display in result.
print(train.describe(include = 'all'))
```

```
         employee_id         department    region   education  
gender  \
count   54808.000000              54808     54808       52399  54808

unique           NaN                  9        34           3      2

top              NaN  Sales & Marketing  region_2  Bachelor's      m

freq             NaN              16840     12343       36669  38496

mean    39195.830627                NaN       NaN         NaN    NaN

std     22586.581449                NaN       NaN         NaN    NaN

min         1.000000                NaN       NaN         NaN    NaN

25%     19669.750000                NaN       NaN         NaN    NaN

50%     39225.500000                NaN       NaN         NaN    NaN

75%     58730.500000                NaN       NaN         NaN    NaN

max     78298.000000                NaN       NaN         NaN    NaN


       recruitment_channel  no_of_trainings           age  \
count                54808     54808.000000  54808.000000   
unique                   3              NaN           NaN   
top                  other              NaN           NaN   
freq                 30446              NaN           NaN   
mean                   NaN         1.253011     34.803915   
std                    NaN         0.609264      7.660169   
min                    NaN         1.000000     20.000000   
25%                    NaN         1.000000     29.000000   
50%                    NaN         1.000000     33.000000   
75%                    NaN         1.000000     39.000000   
max                    NaN        10.000000     60.000000   

        previous_year_rating  length_of_service  KPIs_met >80%  
awards_won?  \
count           50684.000000       54808.000000   54808.000000
```

```
54808.000000
unique                   NaN                NaN            NaN
NaN
top                      NaN                NaN            NaN
NaN
freq                     NaN                NaN            NaN
NaN
mean                3.329256           5.865512       0.351974
0.023172
std                 1.259993           4.265094       0.477590
0.150450
min                 1.000000           1.000000       0.000000
0.000000
25%                 3.000000           3.000000       0.000000
0.000000
50%                 3.000000           5.000000       0.000000
0.000000
75%                 4.000000           7.000000       1.000000
0.000000
max                 5.000000          37.000000       1.000000
1.000000

        avg_training_score   is_promoted
count         54808.000000  54808.000000
unique                 NaN           NaN
top                    NaN           NaN
freq                   NaN           NaN
mean             63.386750      0.085170
std              13.371559      0.279137
min              39.000000      0.000000
25%              51.000000      0.000000
50%              60.000000      0.000000
75%              76.000000      0.000000
max              99.000000      1.000000
```

```
#print info of train and test
print(train.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 54808 entries, 0 to 54807
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   employee_id           54808 non-null  int64
 1   department            54808 non-null  object
 2   region                54808 non-null  object
 3   education             52399 non-null  object
 4   gender                54808 non-null  object
 5   recruitment_channel   54808 non-null  object
 6   no_of_trainings       54808 non-null  int64
 7   age                   54808 non-null  int64
 8   previous_year_rating  50684 non-null  float64
 9   length_of_service     54808 non-null  int64
 10  KPIs_met >80%         54808 non-null  int64
 11  awards_won?           54808 non-null  int64
 12  avg_training_score    54808 non-null  int64
 13  is_promoted           54808 non-null  int64
dtypes: float64(1), int64(8), object(5)
memory usage: 5.9+ MB
```

```
# checking if there is any NULL value in the dataset
print(train.isnull().any())
```

```
employee_id             False
department              False
region                  False
education                True
gender                  False
recruitment_channel     False
no_of_trainings         False
age                     False
previous_year_rating     True
length_of_service       False
KPIs_met >80%           False
awards_won?             False
avg_training_score      False
is_promoted             False
dtype: bool
```

```
print(test.isnull().sum())
```

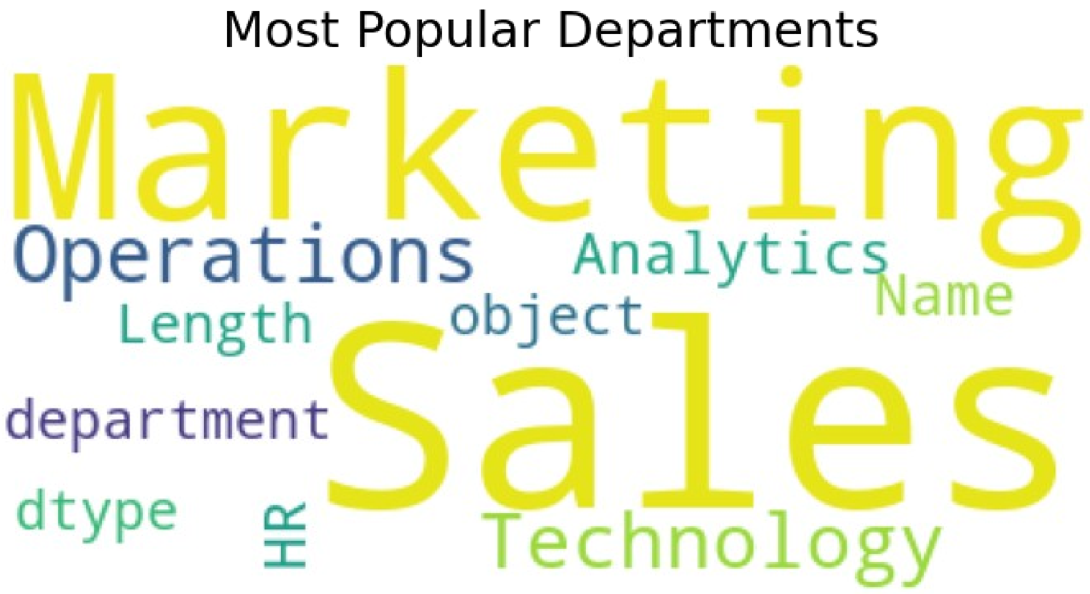
```
employee_id                0
department                 0
region                     0
education               1034
gender                     0
recruitment_channel        0
no_of_trainings            0
age                        0
previous_year_rating    1812
length_of_service          0
KPIs_met >80%              0
awards_won?                0
avg_training_score         0
dtype: int64
```

UNi-variate Data Visualization

```
# looking at the most popular departments with tittle Most Popular
Departments
# use and import matplotlib, wordcloud & stopwords
```

```
from wordcloud import WordCloud
from wordcloud import STOPWORDS
```

```
<wordcloud.wordcloud.WordCloud object at 0x7fc163e4d710>
```



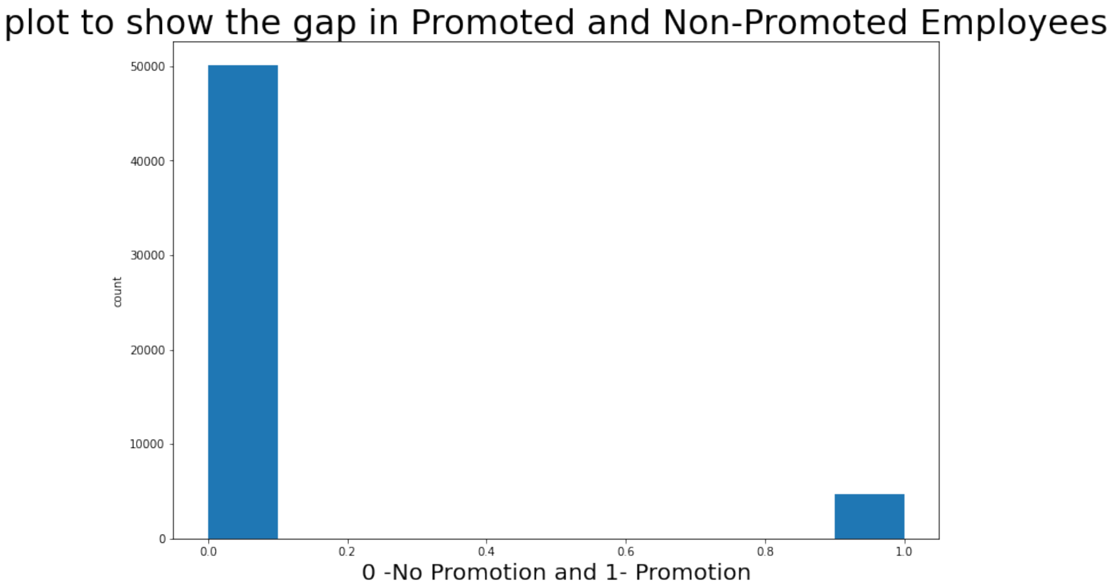
```
# checkig the no. of Employees Promoted
print(train['is_promoted'].value_counts())
```

```
0    50140
1     4668
Name: is_promoted, dtype: int64
```

```
# finding the %age of people promoted

promoted = (4668/54808)*100
print("Percentage of Promoted Employees is {:.2f}%".format(promoted))
```

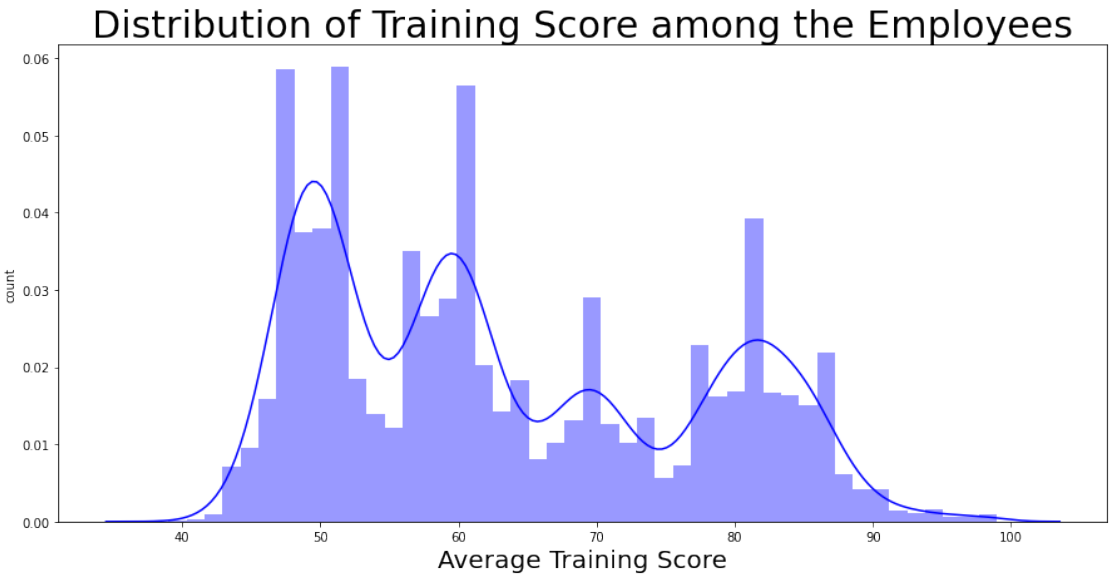
```
Percentage of Promoted Employees is 8.52%
```

```
#plotting a scatter plot
#title - plot to show the gap in Promoted and Non-Promoted Employees
#use train['is_promoted']

plt.hist(train['is_promoted'])
plt.title('plot to show the gap in Promoted and Non-Promoted
Employees', fontsize = 30)
plt.xlabel('0 -No Promotion and 1- Promotion', fontsize = 20)
plt.ylabel('count')
plt.show()
```



```
# checking the distribution of the avg_training score of the Employees
# title - Distribution of Training Score among the Employees
#use train['avg_training_score']
plt.rcParams['figure.figsize'] = (15, 7)
sns.distplot(train['avg_training_score'], color = 'blue')
plt.title('Distribution of Training Score among the Employees',
fontsize = 30)
plt.xlabel('Average Training Score', fontsize = 20)
plt.ylabel('count')
plt.show()
```

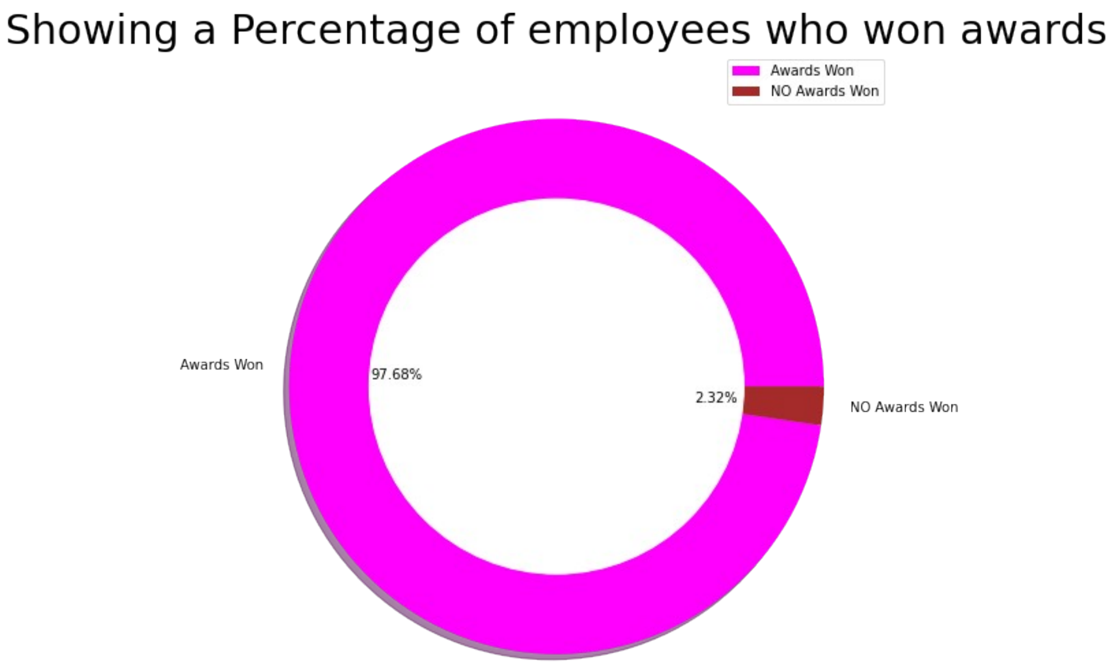
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an
axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



```
print(train['awards_won?'].value_counts())
```

```
0    53538
1     1270
Name: awards_won?, dtype: int64
```

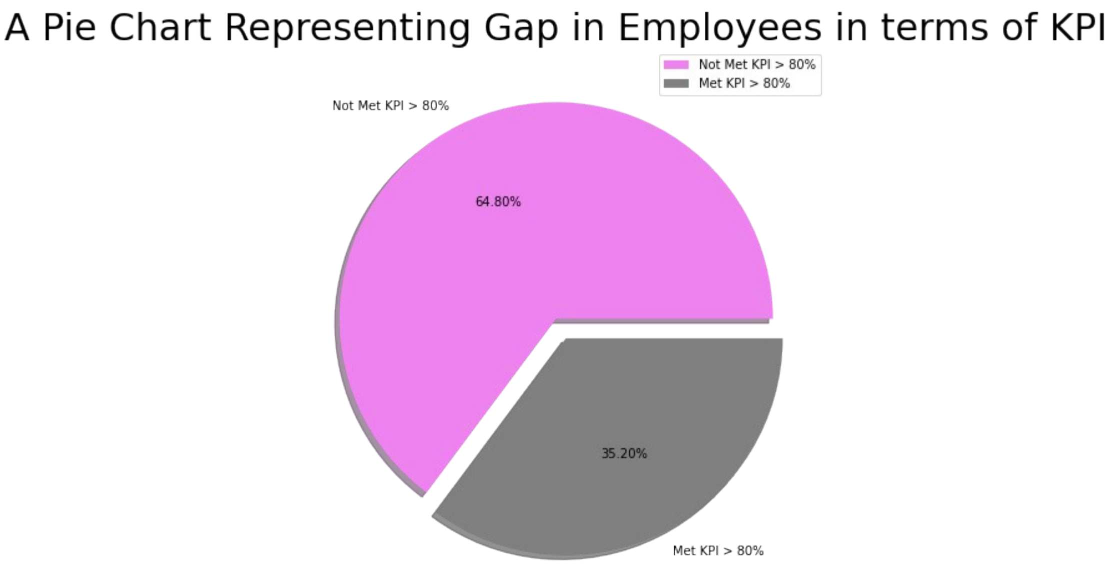
```
# plotting a donut chart for visualizing each of the recruitment
channel's share
# title 'Showing a Percentage of employees who won awards'
# use plt.Circle
# labels = "Awards Won", "NO Awards Won"
# add legend (Awards Won and NO Awards Won)
size = [53538, 1270]
colors = ['magenta', 'brown']
labels = "Awards Won", "NO Awards Won"

my_circle = plt.Circle((0, 0), 0.7, color = 'white')

plt.rcParams['figure.figsize'] = (9, 9)
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct
= '%.2f%%')
plt.title('Showing a Percentage of employees who won awards', fontsize
= 30)
p = plt.gcf()
p.gca().add_artist(my_circle)
plt.legend()
plt.show()
```



```
# find the counts whose 'KPIs_met >80%'
print(train['KPIs_met >80%'].value_counts())
```

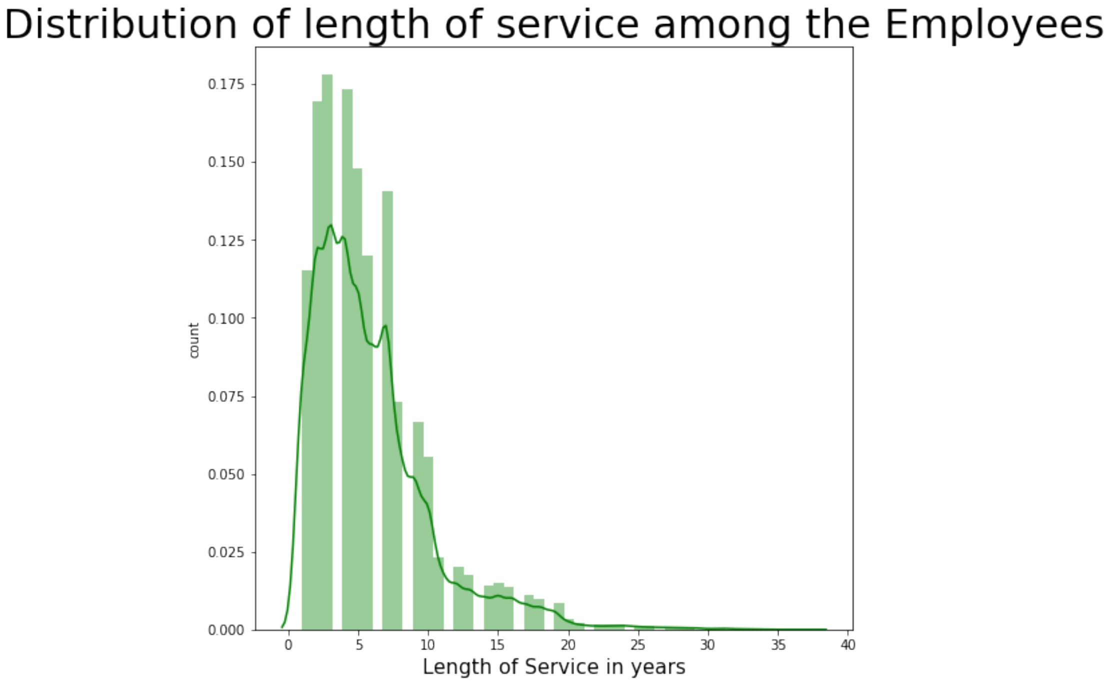
```
0    35517
1    19291
Name: KPIs_met >80%, dtype: int64
```

```
# plotting a pie chart
# labels = "Not Met KPI > 80%", "Met KPI > 80%"
# title 'A Pie Chart Representing Gap in Employees in terms of KPI'
# display legend

size = [35517, 19291]
labels = "Not Met KPI > 80%", "Met KPI > 80%"
colors = ['violet', 'grey']
explode = [0, 0.1]

plt.rcParams['figure.figsize'] = (8, 8)
plt.pie(size, labels = labels, colors = colors, explode = explode,
shadow = True, autopct = "%.2f%%")
plt.title('A Pie Chart Representing Gap in Employees in terms of KPI',
fontsize = 30)
plt.axis('off')
plt.legend()
plt.show()
```

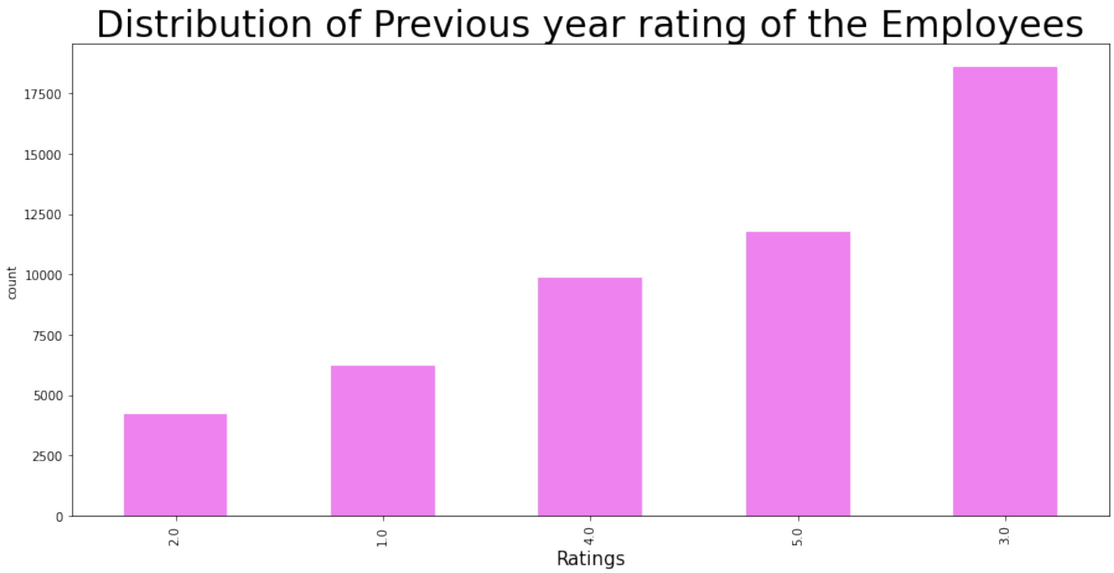


```
# checking the distribution of length of service
# title 'Distribution of length of service among the Employees'

sns.distplot(train['length_of_service'], color = 'green')
plt.title('Distribution of length of service among the Employees',
fontsize = 30)
plt.xlabel('Length of Service in years', fontsize = 15)
plt.ylabel('count')
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an
axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```


Distribution of length of service among the Employees

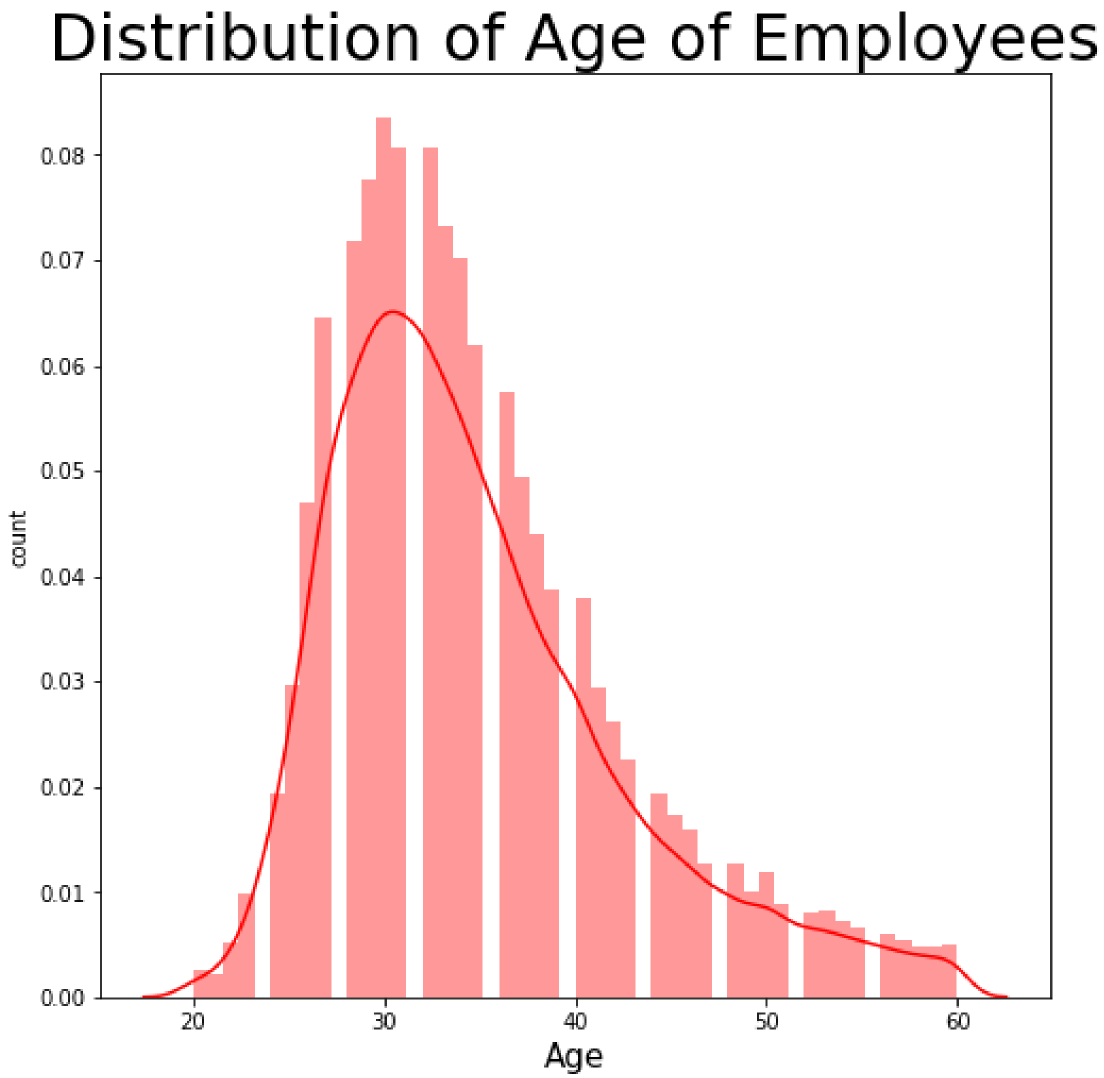
```
# 'Distribution of Previous year rating of the Employees'
train['previous_year_rating'].value_counts().sort_values().plot.bar(color = 'violet', figsize = (15, 7))
plt.title('Distribution of Previous year rating of the Employees', fontsize = 30)
plt.xlabel('Ratings', fontsize = 15)
plt.ylabel('count')
plt.show()
```


Distribution of Previous year rating of the Employees

```
# checking the distribution of age of Employees in the company

sns.distplot(train['age'], color = 'red')
```

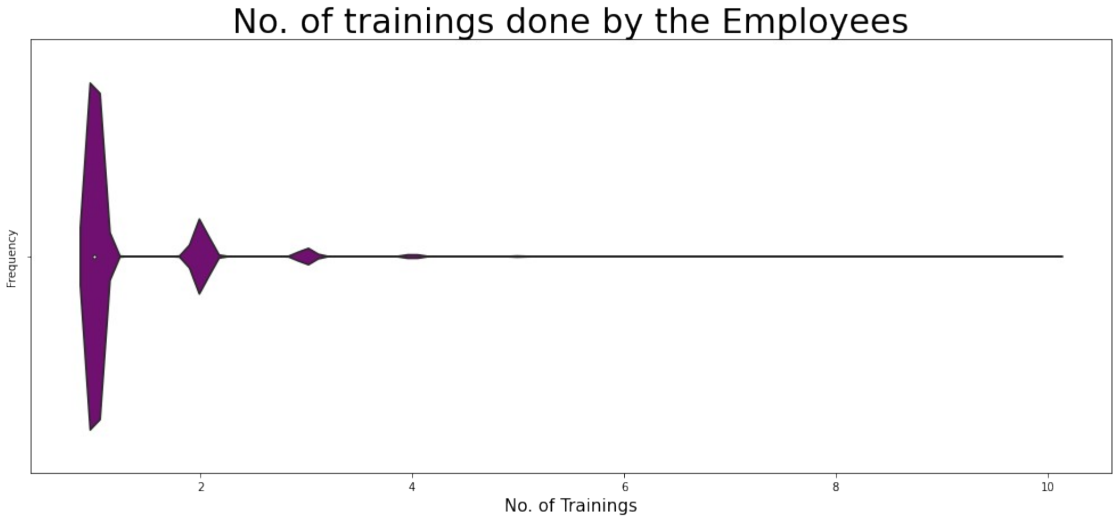
```
plt.title('Distribution of Age of Employees', fontsize = 30)
plt.xlabel('Age', fontsize = 15)
plt.ylabel('count')
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an
axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



```
# checking the different no. of training done by the employees
# use Violinplot for the train['no_of_trainings'] column
# title 'No. of trainings done by the Employees'
plt.rcParams['figure.figsize'] = (17, 7)
sns.violinplot(train['no_of_trainings'], color = 'purple')
plt.title('No. of trainings done by the Employees', fontsize = 30)
plt.xlabel('No. of Trainings', fontsize = 15)
plt.ylabel('Frequency')
plt.show()
```

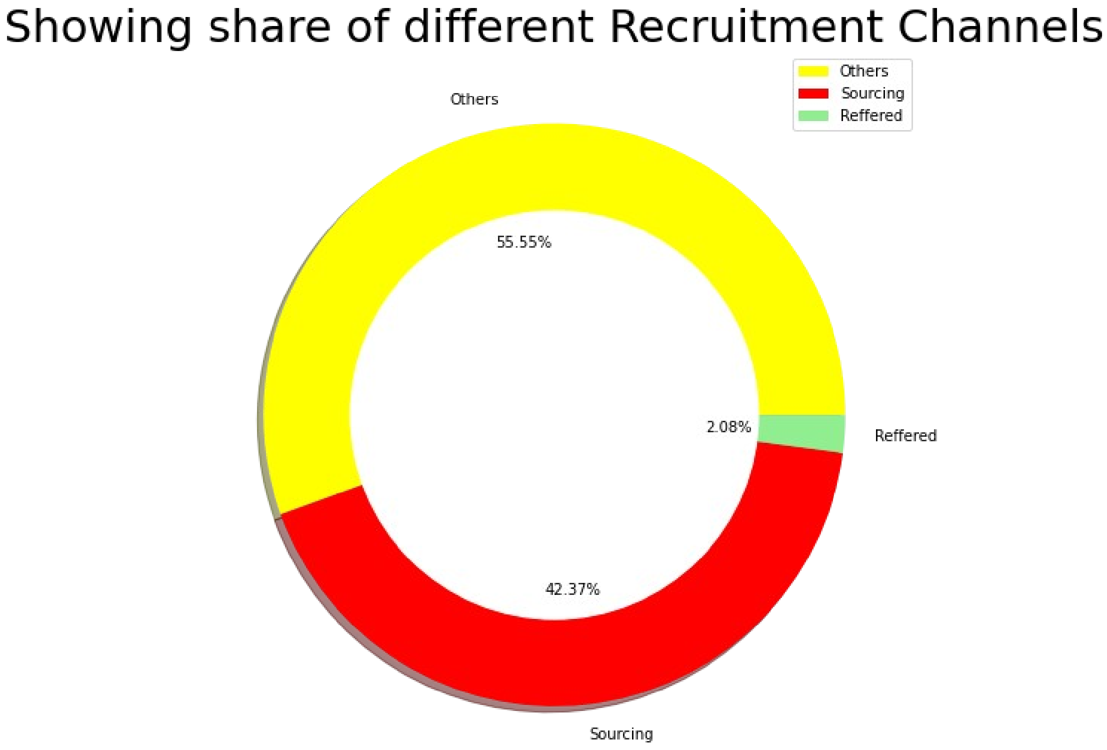
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```



```
# checking the different types of recruitment channels for the company
# use value_counts()

print(train['recruitment_channel'].value_counts())
```

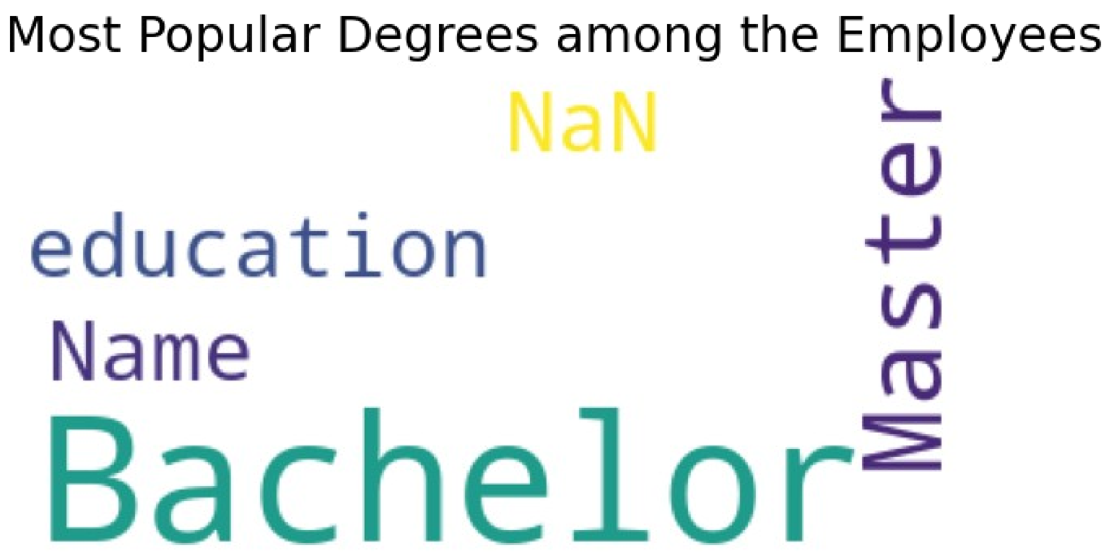
```
other       30446
sourcing    23220
referred     1142
Name: recruitment_channel, dtype: int64
```

```
# plotting a donut chart for visualizing each of the recruitment
channel's share
# use plt.Circle and plt.pie
#labels = "Others", "Sourcing", "Reffered"
size = [30446, 23220, 1142]
colors = ['yellow', 'red', 'lightgreen']
labels = "Others", "Sourcing", "Reffered"

my_circle = plt.Circle((0, 0), 0.7, color = 'white')

plt.rcParams['figure.figsize'] = (9, 9)
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct
= '%.2f%%')
plt.title('Showing share of different Recruitment Channels', fontsize
= 30)
p = plt.gcf()
p.gca().add_artist(my_circle)
plt.legend()
plt.show()
```



```
# checking the most popular education degree among the employees
# title 'Most Popular Degrees among the Employees'

from wordcloud import WordCloud
from wordcloud import STOPWORDS
```

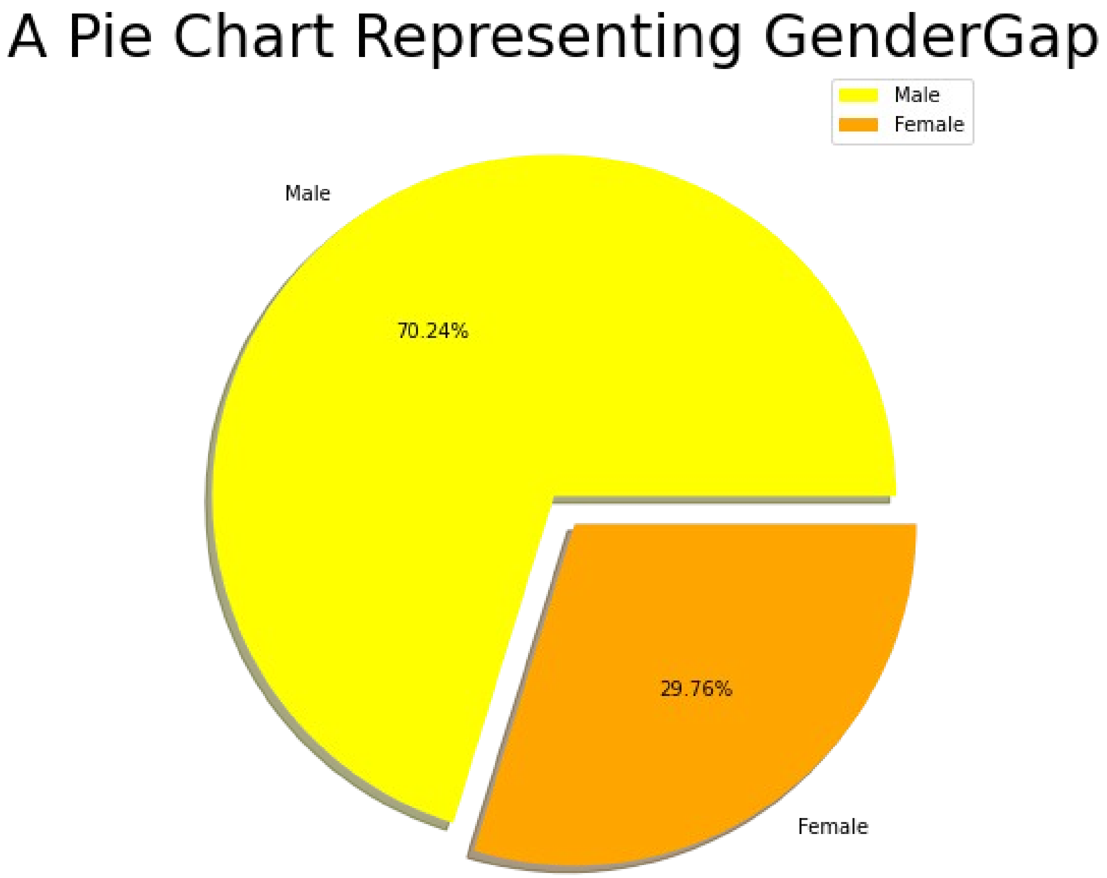
```
<wordcloud.wordcloud.WordCloud object at 0x7fc15c9ea710>
```



```
# checking the gender gap
# count male and female

print(train['gender'].value_counts())
```

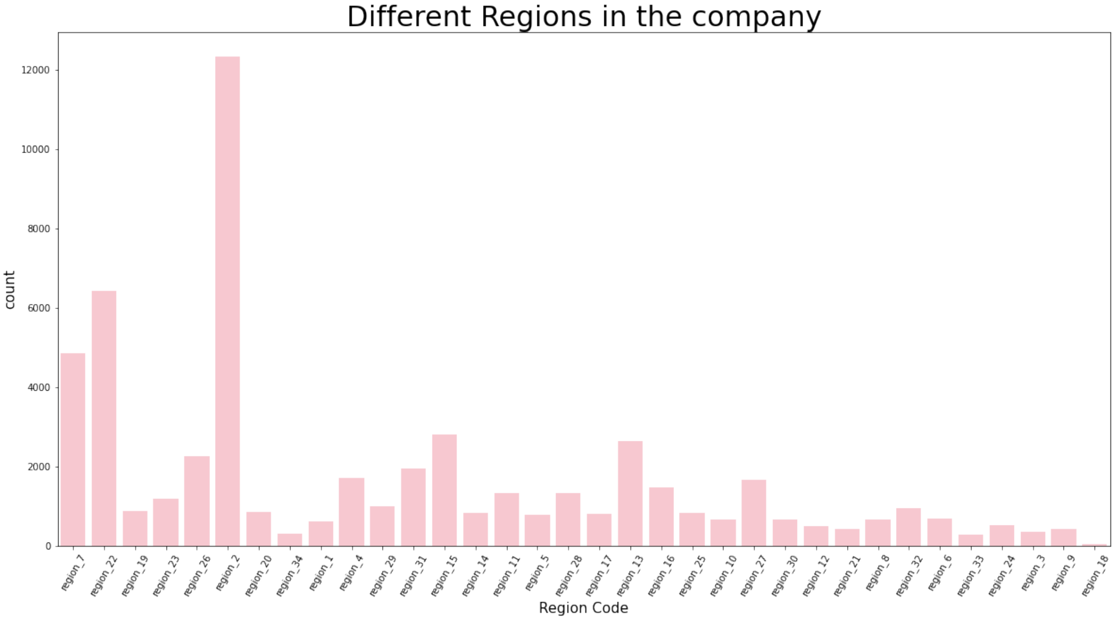
```
m    38496
f    16312
Name: gender, dtype: int64
```

```
# plotting a pie chart
# title A Pie Chart Representing GenderGap
# legend Male & Female

size = [38496, 16312]
labels = "Male", "Female"
colors = ['yellow', 'orange']
explode = [0, 0.1]

plt.rcParams['figure.figsize'] = (8, 8)
plt.pie(size, labels = labels, colors = colors, explode = explode,
shadow = True, autopct = "%.2f%%")
plt.title('A Pie Chart Representing GenderGap', fontsize = 30)
plt.axis('off')
plt.legend()
plt.show()
```



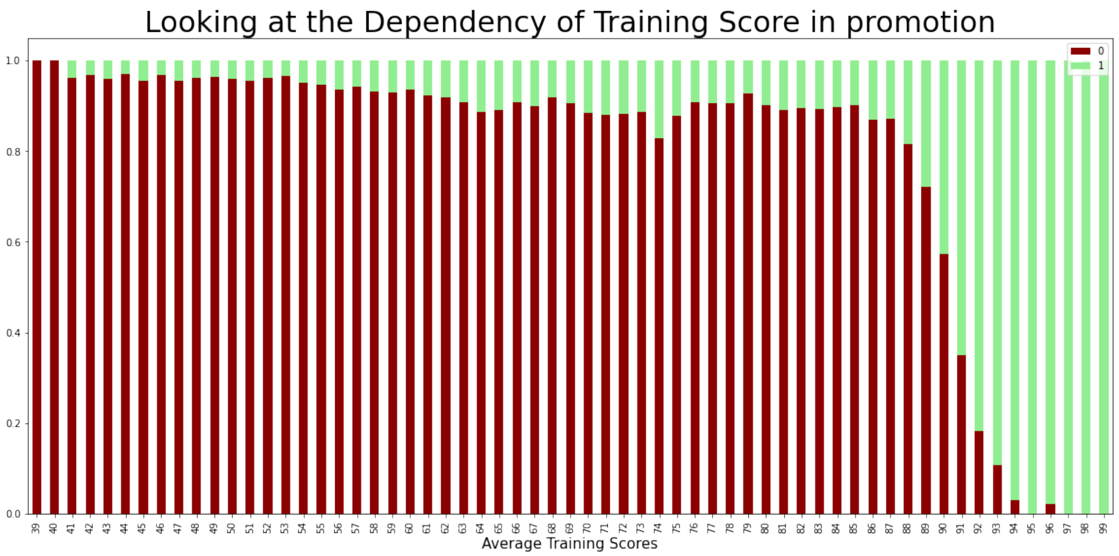
```
# checking the different regions of the company
# title 'Different Regions in the company'

plt.rcParams['figure.figsize'] = (20, 10)
sns.countplot(train['region'], color = 'pink')
plt.title('Different Regions in the company', fontsize = 30)
plt.xticks(rotation = 60)
plt.xlabel('Region Code', fontsize = 15)
plt.ylabel('count', fontsize = 15)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```



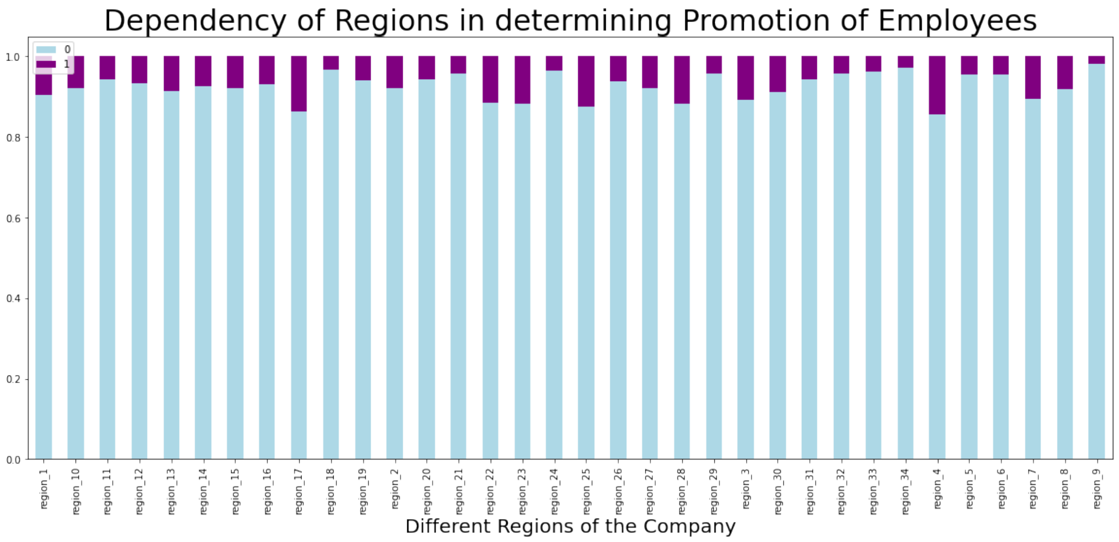
Bi-varaiate Data Visualization

```
# scatter plot between average training score and is_promoted
# use crosstab in two columns train['avg_training_score'],
train['is_promoted']

data = pd.crosstab(train['avg_training_score'], train['is_promoted'])
data.div(data.sum(1).astype(float), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (20, 9), color = ['darkred', 'lightgreen'])

plt.title('Looking at the Dependency of Training Score in promotion',
fontsize = 30)
plt.xlabel('Average Training Scores', fontsize = 15)
plt.legend()
plt.show()
```



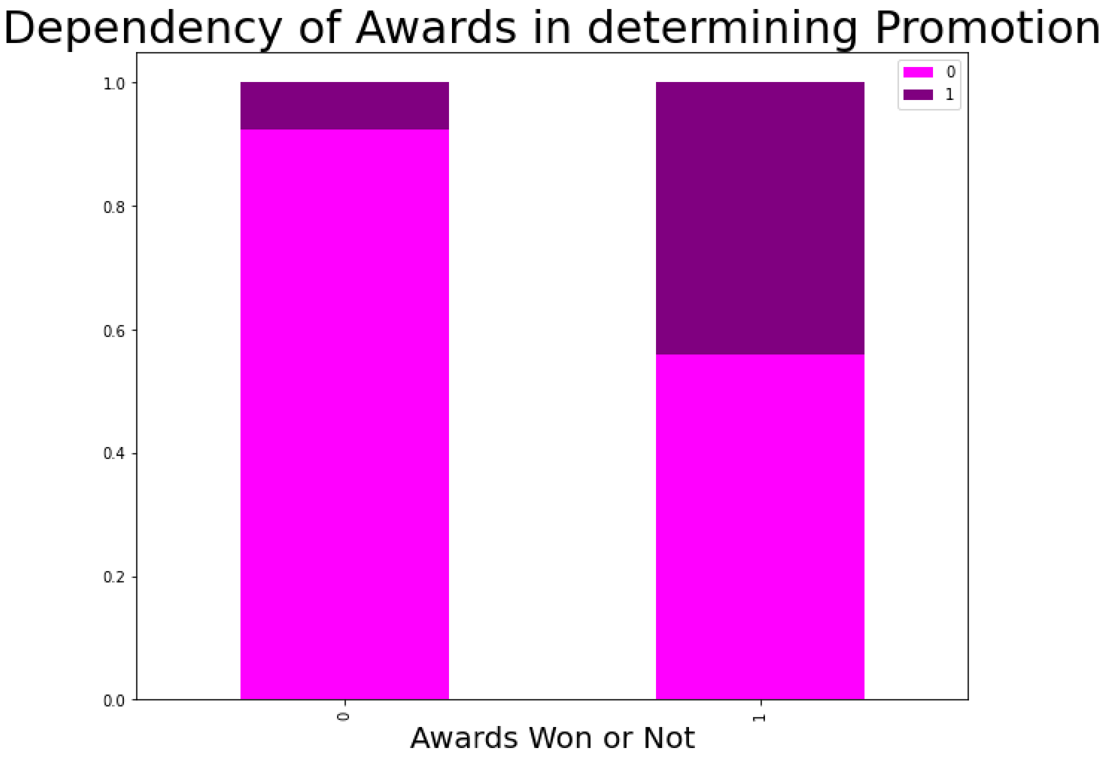
As, the Training Scores Increases, the chances of Promotion Increases Highly

```
# checking dependency of different regions in promotion
# use pd.crosstab train['region'], train['is_promoted']
# title 'Dependency of Regions in determining Promotion of Employees'

data = pd.crosstab(train['region'], train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (20, 8), color = ['lightblue', 'purple'])

plt.title('Dependency of Regions in determining Promotion of
Employees', fontsize = 30)
plt.xlabel('Different Regions of the Company', fontsize = 20)
plt.legend()
plt.show()
```



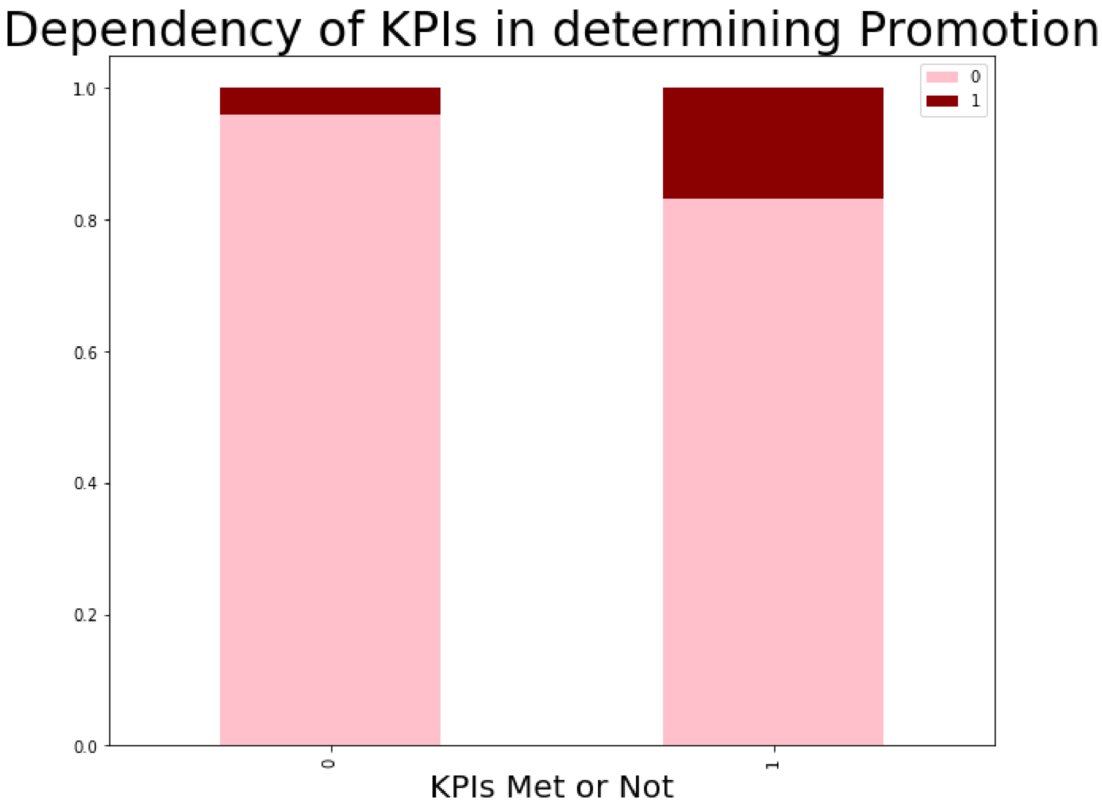
The above graph shows that there is no biasedness over regions in terms of Promotion as all the regions share promotions almost equally.

```
# dependency of awards won on promotion
# pd.crosstab train['awards_won?'], train['is_promoted']

data = pd.crosstab(train['awards_won?'], train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (10, 8), color = ['magenta', 'purple'])

plt.title('Dependency of Awards in determining Promotion', fontsize =
30)
plt.xlabel('Awards Won or Not', fontsize = 20)
plt.legend()
plt.show()
```



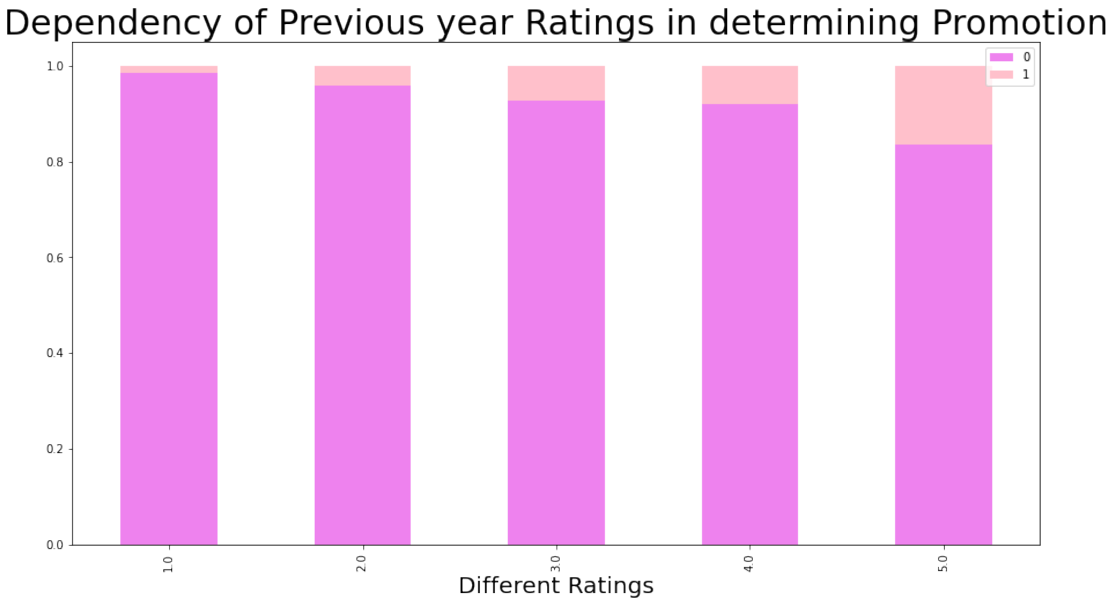
There is a very good chance of getting promoted if the employee has won an award

```
#dependency of KPIs with Promotion

data = pd.crosstab(train['KPIs_met >80%'], train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (10, 8), color = ['pink', 'darkred'])

plt.title('Dependency of KPIs in determining Promotion', fontsize =
30)
plt.xlabel('KPIs Met or Not', fontsize = 20)
plt.legend()
plt.show()
```



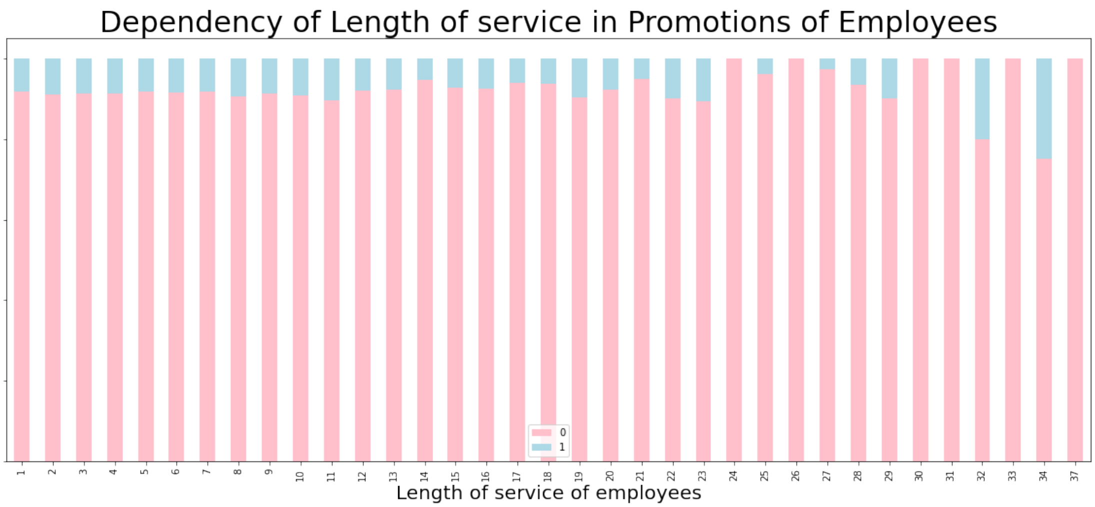
Again Having a good KPI score increases the chances of getting promoted in the company.

```
# checking dependency on previous years' ratings
# pd.crosstab(train['previous_year_rating'], train['is_promoted'])

data = pd.crosstab(train['previous_year_rating'],
train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (15, 8), color = ['violet', 'pink'])

plt.title('Dependency of Previous year Ratings in determining
Promotion', fontsize = 30)
plt.xlabel('Different Ratings', fontsize = 20)
plt.legend()
plt.show()
```

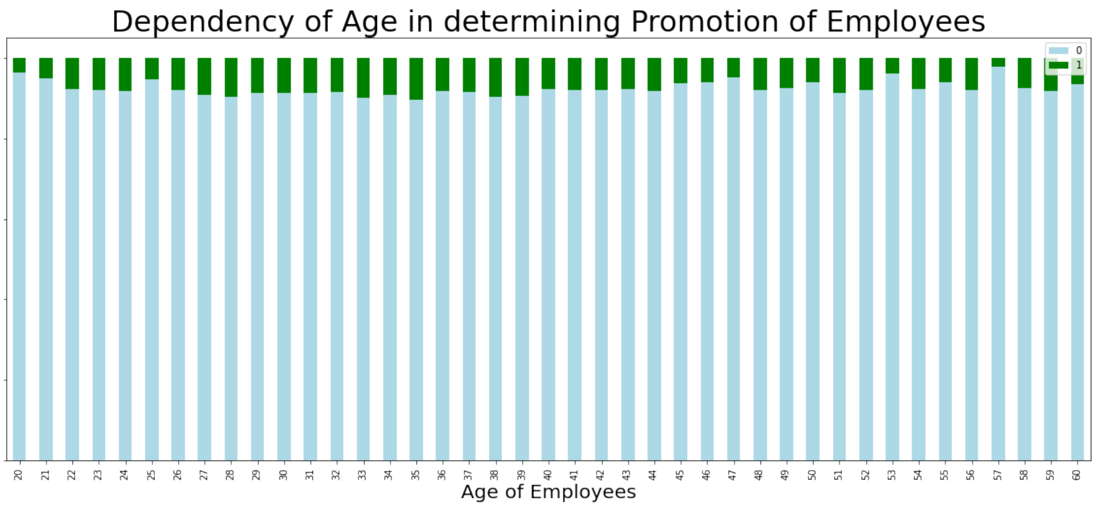


The Above Graph clearly suggests that previous ratings matter a lot, if the ratings are high, the chances of being promoted in the company increases and there is completely no promotion for the employees with previous year ratings = 0

```
# checking how length of service determines the promotion of employees

#data = pd.crosstab(train['length_of_service'], train['is_promoted'])

data = pd.crosstab(train['length_of_service'], train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (20, 8), color = ['pink', 'lightblue'])

plt.title('Dependency of Length of service in Promotions of
Employees', fontsize = 30)
plt.xlabel('Length of service of employees', fontsize = 20)
plt.legend()
plt.show()
```



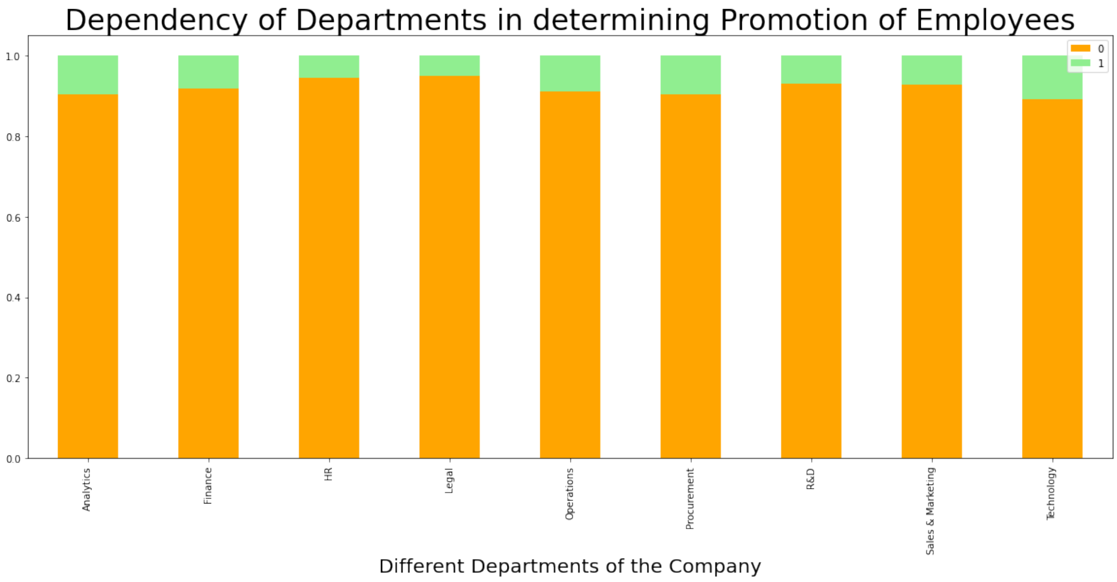
```
# checking dependency of age factor in promotion of employees


data = pd.crosstab(train['age'], train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (20, 8), color = ['lightblue', 'green'])

plt.title('Dependency of Age in determining Promotion of Employees',
fontsize = 30)
plt.xlabel('Age of Employees', fontsize = 20)
plt.legend()
plt.show()
```



This is Very Impressive that the company promotes employees of all the ages equally even the freshers have equal share of promotion and also the senior citizen employees are getting the equal share of Promotion in the Company

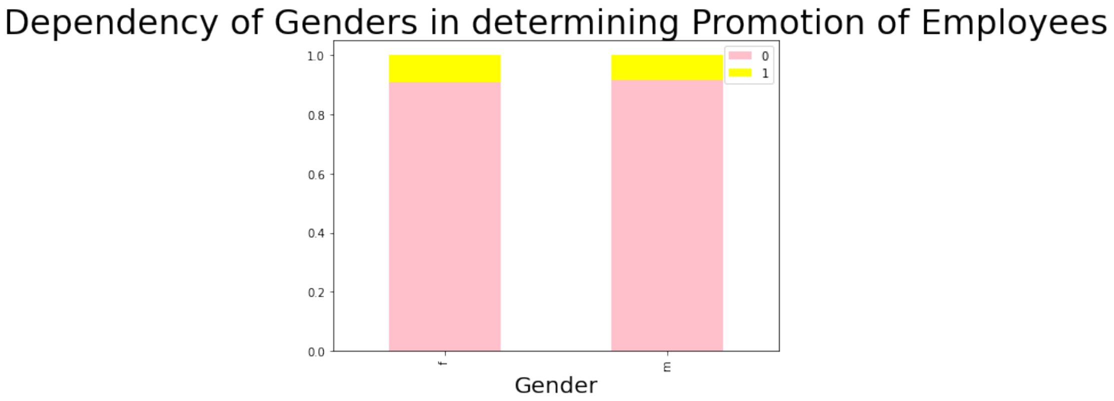
```
# checking which department got most number of promotions
```

```
#data = pd.crosstab(train['department'], train['is_promoted'])
#data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (20, 8), color = ['orange', 'lightgreen'])

data = pd.crosstab(train['department'], train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (20, 8), color = ['orange', 'lightgreen'])

plt.title('Dependency of Departments in determining Promotion of
Employees', fontsize = 30)
plt.xlabel('Different Departments of the Company', fontsize = 20)
plt.legend()
plt.show()
```



Again, Each of the departments have equal no. of promotions showing an equal developement in each of the departments of the company.

```
# checking dependency of gender over promotion


data = pd.crosstab(train['gender'], train['is_promoted'])
data.div(data.sum(1).astype('float'), axis = 0).plot(kind = 'bar',
stacked = True, figsize = (7, 5), color = ['pink', 'yellow'])

plt.title('Dependency of Genders in determining Promotion of
Employees', fontsize = 30)
plt.xlabel('Gender', fontsize = 20)
plt.legend()
plt.show()
```



The above plot shows that there is no partiality between males and females in terms of promotion

Data Pre-processing

```
# filling missing values

print(train['education'].fillna(train['education'].mode()[0], inplace
= True))
print(train['previous_year_rating'].fillna(1, inplace = True))

# again checking if there is any Null value left in the data
print(train.isnull().sum().sum())
```

```
----------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
c:\Users\Hemendra yadav\Documents\cs II\python project\
Data_Analysis_Question_of_HRAnalytics (1).ipynb Cell 52' in <cell
line: 3>()
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000051?line=0'>1</a> # filling missing values
----> <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000051?line=2'>3</a>
print(train['education'].fillna(train['education'].mode()[0], inplace
= True))
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000051?line=3'>4</a>
print(train['previous_year_rating'].fillna(1, inplace = True))
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
```

```
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000051?line=5'>6</a> # again checking if there is
any Null value left in the data

NameError: name 'train' is not defined
```

```
# filling missing values

test['education'].fillna(test['education'].mode()[0], inplace = True)
test['previous_year_rating'].fillna(1, inplace = True)

# again checking if there is any Null value left in the data
test.isnull().sum().sum()
```

```
----------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
c:\Users\Hemendra yadav\Documents\cs II\python project\
Data_Analysis_Question_of_HRAnalytics (1).ipynb Cell 53' in <cell
line: 3>()
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000052?line=0'>1</a> # filling missing values
----> <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000052?line=2'>3</a>
test['education'].fillna(test['education'].mode()[0], inplace = True)
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000052?line=3'>4</a>
test['previous_year_rating'].fillna(1, inplace = True)
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000052?line=5'>6</a> # again checking if there is
any Null value left in the data

NameError: name 'test' is not defined
```

```
# removing the employee_id column


train = train.drop(['employee_id'], axis = 1)

print(train.columns)
```

```
Index(['department', 'region', 'education', 'gender',
'recruitment_channel',
       'no_of_trainings', 'age', 'previous_year_rating',
'length_of_service',
       'KPIs_met >80%', 'awards_won?', 'avg_training_score',
'is_promoted'],
      dtype='object')
```

```
# saving the employee_id

emp_id = test['employee_id']

# removing the employee_id column

test = test.drop(['employee_id'], axis = 1)

print(test.columns)


# print all the columns
```

```
Index(['department', 'region', 'education', 'gender',
'recruitment_channel',
       'no_of_trainings', 'age', 'previous_year_rating',
'length_of_service',
       'KPIs_met >80%', 'awards_won?', 'avg_training_score'],
      dtype='object')
```

```
# defining the test set

x_test = test
print(x_test.columns)
```

```
----------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
c:\Users\Hemendra yadav\Documents\cs II\python project\
Data_Analysis_Question_of_HRAnalytics (1).ipynb Cell 56' in <cell
line: 3>()
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000055?line=0'>1</a> # defining the test set
----> <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000055?line=2'>3</a> x_test = test
      <a
href='vscode-notebook-cell:/c%3A/Users/Hemendra%20yadav/Documents/cs
```

```
%20II/python%20project/Data_Analysis_Question_of_HRAnalytics
%20%281%29.ipynb#ch0000055?line=3'>4</a> print(x_test.columns)

NameError: name 'test' is not defined
```

```
# one hot encoding for the test set
# use pd.get_dummies in x_test


# print all the columns
```

```
x = train.iloc[:, :-1]
y = train.iloc[:, -1]

print("Shape of x:", x.shape)
print("Shape of y:", y.shape)
```

```
Index(['no_of_trainings', 'age', 'previous_year_rating',
'length_of_service',
       'KPIs_met >80%', 'awards_won?', 'avg_training_score',
       'department_Analytics', 'department_Finance', 'department_HR',
       'department_Legal', 'department_Operations',
'department_Procurement',
       'department_R&D', 'department_Sales & Marketing',
       'department_Technology', 'region_region_1', 'region_region_10',
       'region_region_11', 'region_region_12', 'region_region_13',
       'region_region_14', 'region_region_15', 'region_region_16',
       'region_region_17', 'region_region_18', 'region_region_19',
       'region_region_2', 'region_region_20', 'region_region_21',
       'region_region_22', 'region_region_23', 'region_region_24',
       'region_region_25', 'region_region_26', 'region_region_27',
       'region_region_28', 'region_region_29', 'region_region_3',
       'region_region_30', 'region_region_31', 'region_region_32',
       'region_region_33', 'region_region_34', 'region_region_4',
       'region_region_5', 'region_region_6', 'region_region_7',
       'region_region_8', 'region_region_9', 'education_Bachelor's',
       'education_Below Secondary', 'education_Master's & above',
'gender_f',
       'gender_m', 'recruitment_channel_other',
'recruitment_channel_referred',
       'recruitment_channel_sourcing'],
      dtype='object')
```

```
# splitting the train set into dependent and independent sets

x = # all rows and last columns  in train dataframe
y = # all rows and last columns  in train dataframe

# print the shape of x & y
```

```
Shape of x: (54808, 12)
Shape of y: (54808,)
```

```
# Do one hot encoding for the train set
# pd.get_dummies(x)

#print all columns

x = pd.get_dummies(x)

print(x.columns)
```

```
Index(['no_of_trainings', 'age', 'previous_year_rating',
'length_of_service',
       'KPIs_met >80%', 'awards_won?', 'avg_training_score',
       'department_Analytics', 'department_Finance', 'department_HR',
       'department_Legal', 'department_Operations',
'department_Procurement',
       'department_R&D', 'department_Sales & Marketing',
       'department_Technology', 'region_region_1', 'region_region_10',
       'region_region_11', 'region_region_12', 'region_region_13',
       'region_region_14', 'region_region_15', 'region_region_16',
       'region_region_17', 'region_region_18', 'region_region_19',
       'region_region_2', 'region_region_20', 'region_region_21',
       'region_region_22', 'region_region_23', 'region_region_24',
       'region_region_25', 'region_region_26', 'region_region_27',
       'region_region_28', 'region_region_29', 'region_region_3',
       'region_region_30', 'region_region_31', 'region_region_32',
       'region_region_33', 'region_region_34', 'region_region_4',
       'region_region_5', 'region_region_6', 'region_region_7',
       'region_region_8', 'region_region_9', 'education_Bachelor's',
       'education_Below Secondary', 'education_Master's & above',
'gender_f',
       'gender_m', 'recruitment_channel_other',
'recruitment_channel_referred',
       'recruitment_channel_sourcing'],
      dtype='object')
```

```
# Thank you!!!
```