In [5]:

```
### Table of Contents:
Importing the Library
Loading the Dataset
Check the Null Values using Visualization
Exploratory Data Analysis
Numerical Value Visualization
Pairplot
Heatmap
Countplot
Hist Plot ....
Categorical Variable Visualization
Data Preprocessing
Handling Categorical Variables
Data Modeling
RandomForest CLassifier
XGBoost Classifier



import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns


%matplotlib inline
sns.set_style("whitegrid")
plt.style.use("fivethirtyeight")
```

In [2]:

```
import os
```

In [6]:

```
train = pd.read_csv("HR_train.zip")
```

In [7]:

```
train
```

Out[7]:

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length
0	65438	Sales & Marketing	region_7	Master's & above	f	sourcing	1	35	5.0	
1	65141	Operations	region_22	Bachelor's	m	other	1	30	5.0	
2	7513	Sales & Marketing	region_19	Bachelor's	m	sourcing	1	34	3.0	
3	2542	Sales & Marketing	region_23	Bachelor's	m	other	2	39	1.0	
4	48945	Technology	region_26	Bachelor's	m	other	1	45	3.0	
...	...	...	...	...	...	...	...	...	...	
54803	3030	Technology	region_14	Bachelor's	m	sourcing	1	48	3.0	
54804	74592	Operations	region_27	Master's & above	f	other	1	37	2.0	
54805	13918	Analytics	region_1	Bachelor's	m	other	1	27	5.0	
54806	13614	Sales & Marketing	region_9	NaN	m	sourcing	1	29	1.0	
54807	51526	HR	region_22	Bachelor's	m	other	1	27	1.0	

54808 rows × 14 columns

In [12]:

```
train.tail(10)
```

Out[12]:

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length
54798	40257	Sales & Marketing	region_2	Master's & above	f	other	2	40	5.0	
54799	68093	Procurement	region_2	Master's & above	f	other	1	50	5.0	
54800	39227	HR	region_11	Bachelor's	m	other	2	34	5.0	
54801	12431	Technology	region_26	Bachelor's	f	sourcing	1	31	NaN	
54802	6915	Sales & Marketing	region_14	Bachelor's	m	other	2	31	1.0	
54803	3030	Technology	region_14	Bachelor's	m	sourcing	1	48	3.0	
54804	74592	Operations	region_27	Master's & above	f	other	1	37	2.0	
54805	13918	Analytics	region_1	Bachelor's	m	other	1	27	5.0	
54806	13614	Sales & Marketing	region_9	NaN	m	sourcing	1	29	1.0	
54807	51526	HR	region_22	Bachelor's	m	other	1	27	1.0	

In [11]:

```
train.head(10)
```

Out[11]:

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length_of_
0	65438	Sales & Marketing	region_7	Master's & above	f	sourcing	1	35	5.0	
1	65141	Operations	region_22	Bachelor's	m	other	1	30	5.0	
2	7513	Sales & Marketing	region_19	Bachelor's	m	sourcing	1	34	3.0	
3	2542	Sales & Marketing	region_23	Bachelor's	m	other	2	39	1.0	
4	48945	Technology	region_26	Bachelor's	m	other	1	45	3.0	
5	58896	Analytics	region_2	Bachelor's	m	sourcing	2	31	3.0	
6	20379	Operations	region_20	Bachelor's	f	other	1	31	3.0	
7	16290	Operations	region_34	Master's & above	m	sourcing	1	33	3.0	
8	73202	Analytics	region_20	Bachelor's	m	other	1	28	4.0	
9	28911	Sales & Marketing	region_1	Master's & above	m	sourcing	1	32	5.0	

In [15]:

```
test = pd.read_csv("HR_test.zip")
```

In [16]:

```
test
```

Out[16]:

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length
0	8724	Technology	region_26	Bachelor's	m	sourcing	1	24	NaN	
1	74430	HR	region_4	Bachelor's	f	other	1	31	3.0	
2	72255	Sales & Marketing	region_13	Bachelor's	m	other	1	31	1.0	
3	38562	Procurement	region_2	Bachelor's	f	other	3	31	2.0	
4	64486	Finance	region_29	Bachelor's	m	sourcing	1	30	4.0	
...	...	...	...	...	...	...	...	...	...	
23485	53478	Legal	region_2	Below Secondary	m	sourcing	1	24	3.0	
23486	25600	Technology	region_25	Bachelor's	m	sourcing	1	31	3.0	
23487	45409	HR	region_16	Bachelor's	f	sourcing	1	26	4.0	
23488	1186	Procurement	region_31	Bachelor's	m	sourcing	3	27	NaN	
23489	5973	Technology	region_17	Master's & above	m	other	3	40	5.0	

23490 rows × 13 columns

In [17]:

```
train.shape
```

Out[17]:

```
(54808, 14)
```

In [18]:

```
test.shape
```

Out[18]:

```
(23490, 13)
```

In [19]:

```
train.info
```

Out[19]:

```
<bound method DataFrame.info of        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
...             ...                ...        ...               ...    ...
54803          3030         Technology  region_14        Bachelor's      m
54804         74592         Operations  region_27  Master's & above      f
54805         13918          Analytics   region_1        Bachelor's      m
54806         13614  Sales & Marketing   region_9               NaN      m
54807         51526                 HR  region_22        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
...                  ...              ...  ...                   ...
54803           sourcing                1   48                   3.0
54804              other                1   37                   2.0
54805              other                1   27                   5.0
54806           sourcing                1   29                   1.0
54807              other                1   27                   1.0

       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
...                ...            ...         ...                   ...
54803               17              0           0                    78
54804                6              0           0                    56
54805                3              1           0                    79
54806                2              0           0                    45
54807                5              0           0                    49

       is_promoted
0                0
1                0
2                0
3                0
4                0
...            ...
54803            0
54804            0
54805            0
54806            0
54807            0

[54808 rows x 14 columns]>
```

In [20]:

```
test.info
```

Out[20]:

```
<bound method DataFrame.info of        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
...            ...                ...        ...               ...    ...
23485        53478              Legal   region_2   Below Secondary      m
23486        25600         Technology  region_25        Bachelor's      m
23487        45409                 HR  region_16        Bachelor's      f
23488         1186        Procurement  region_31        Bachelor's      m
23489         5973         Technology  region_17  Master's & above      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
...                  ...              ...  ...                   ...
23485           sourcing                1   24                   3.0
23486           sourcing                1   31                   3.0
23487           sourcing                1   26                   4.0
23488           sourcing                3   27                   NaN
23489              other                3   40                   5.0

       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
...                  ...            ...          ...                 ...
23485                  1              0            0                  61
23486                  7              0            0                  74
23487                  4              0            0                  50
23488                  1              0            0                  70
23489                  5              1            0                  89

[23490 rows x 13 columns]>
```

In [21]:

```
##### checking Null values ###########

train.isnull().sum()
```

Out[21]:

```
employee_id                0
department                 0
region                     0
education               2409
gender                     0
recruitment_channel        0
no_of_trainings            0
age                        0
previous_year_rating    4124
length_of_service          0
KPIs_met >80%              0
awards_won?                0
avg_training_score         0
is_promoted                0
dtype: int64
```

In [31]:

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

Out[31]:

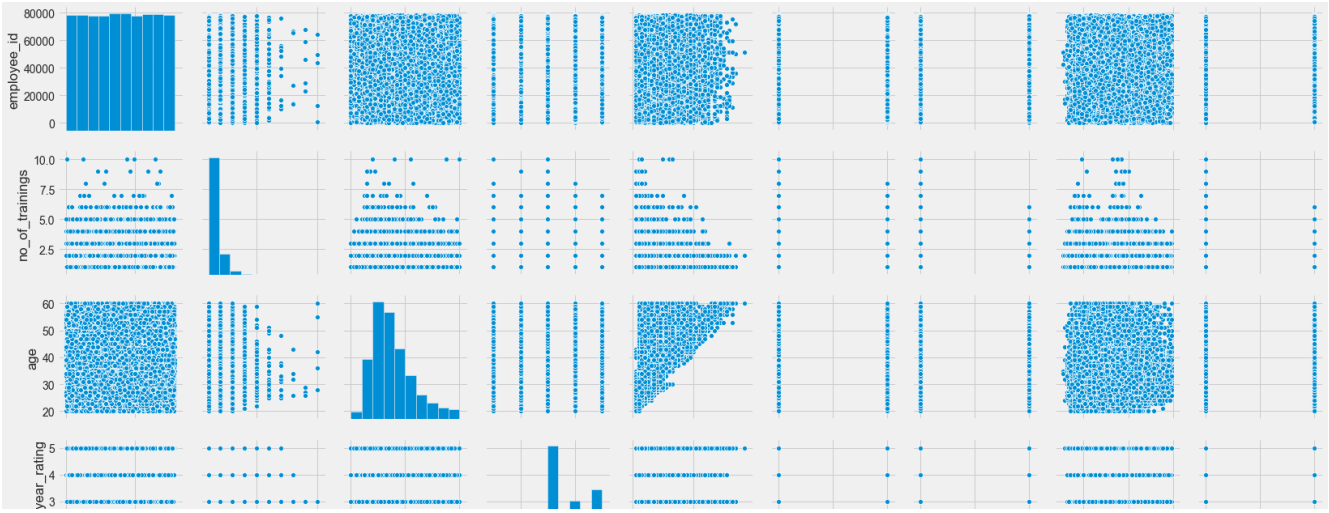
```
employee_id                0
department                 0
region                     0
education               1034
gender                     0
recruitment_channel        0
no_of_trainings            0
age                        0
previous_year_rating    1812
length_of_service          0
KPIs_met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

In [33]:

```
############# exploratory data analysis #############

sns.pairplot(train)
```

Out[33]:

```
<seaborn.axisgrid.PairGrid at 0x261d537c988>
```

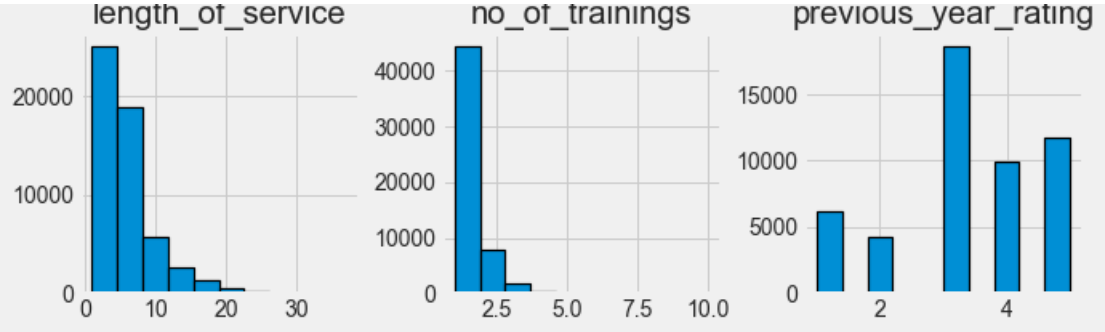


In [34]:

```
######### visualizing the distribution of the data for every feature ##############

train.hist(edgecolor = "black",linewidth = 1.2, figsize=(10,10))
```

Out[34]:

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x00000261DD2BDFC8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000261E3E981C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000261E3ED84C8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x00000261E32B7208>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000261E32F49C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000261E3336EC8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x00000261E3374A88>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000261E33B8EC8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000261E33BFAC8>]],
      dtype=object)
```

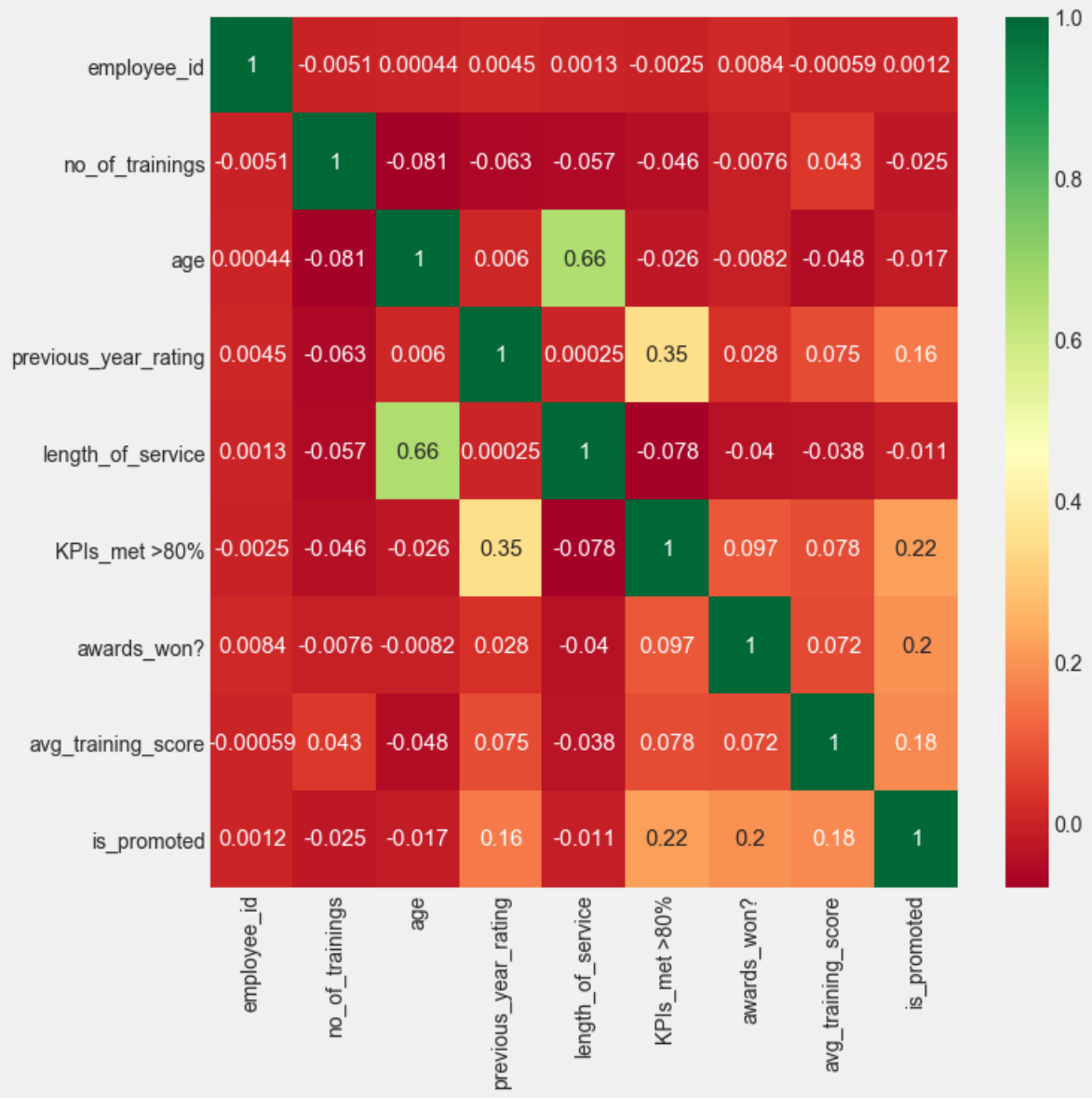


In [35]:

```
######## using heatmap #########

plt.figure(figsize=(10,10))
sns.heatmap(train.corr(),annot=True,cmap="RdYlGn",annot_kws={"size":15})
```

Out[35]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x261e51ff988>
```



In [43]:

```
list(train.columns)
```

Out[43]:

```
['employee_id',
 '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']
```

In [44]:

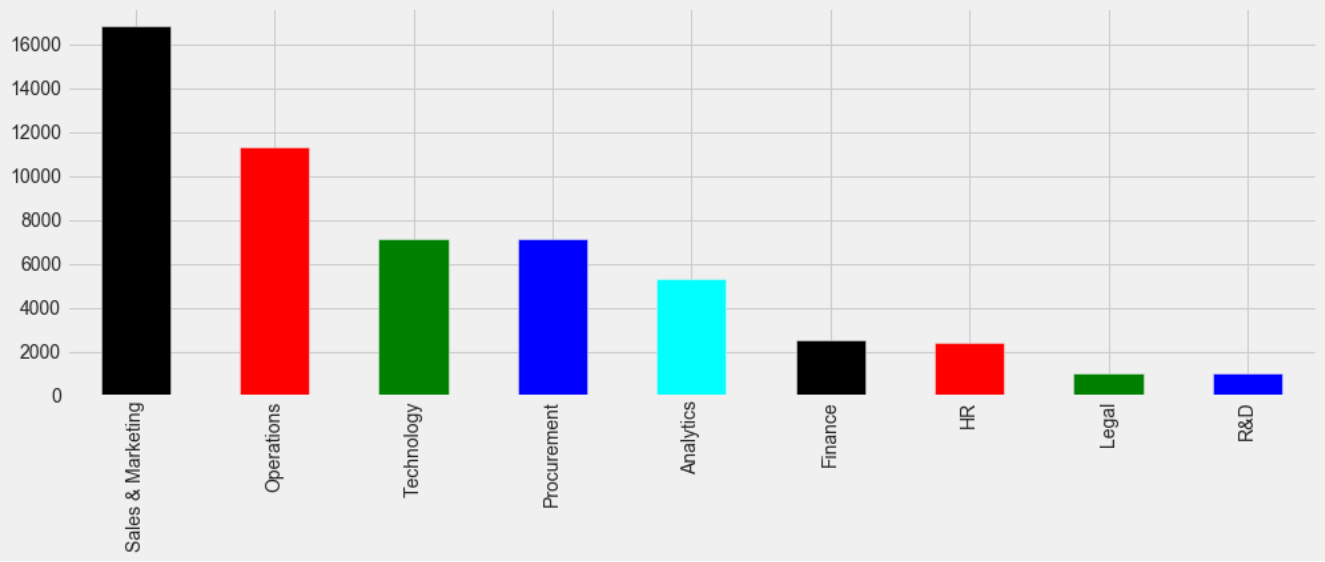
```
train["department"].value_counts()
```

Out[44]:

```
Sales & Marketing    16840
Operations           11348
Technology            7138
Procurement           7138
Analytics             5352
Finance               2536
HR                    2418
Legal                 1039
R&D                    999
Name: department, dtype: int64
```

In [48]:

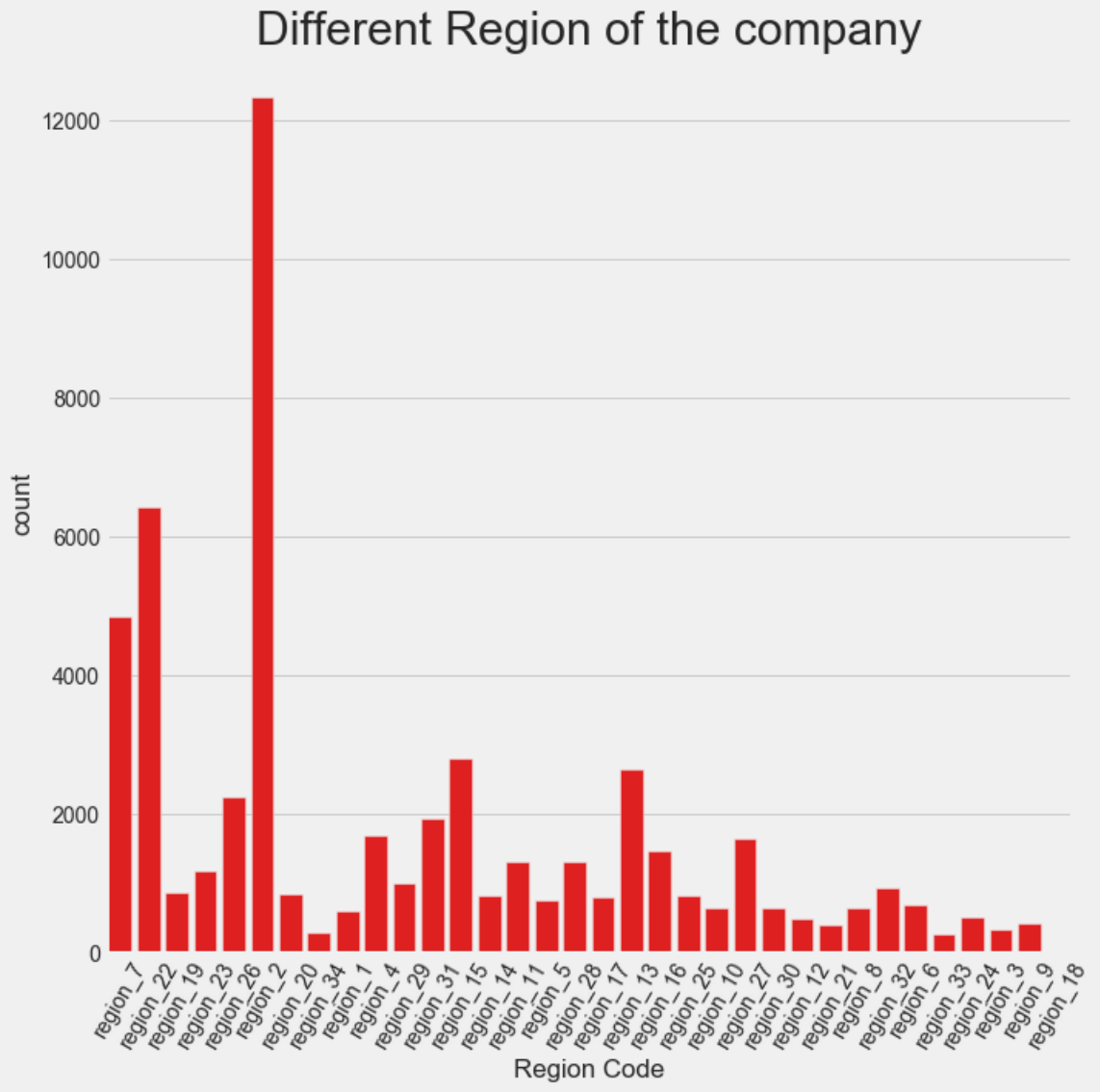
```
########## visualizing the different group in the data set ############

plt.subplots(figsize=(15,5))
train['department'].value_counts(normalize = True)
train['department'].value_counts(dropna = False).plot.bar(color=['black', 'red', 'green', 'blue', '
cyan'])
plt.show()
```



In [49]:

```
########## checking the different region of the company ##########

plt.subplots(figsize=(10,10))
sns.countplot(train['region'], color = 'red')
plt.title('Different Region of the company',fontsize = 30)
plt.xticks(rotation=60)
plt.xlabel('Region Code')
```

```
plt.ylabel('count')
plt.show()
```



In [54]:

```
!pip install wordcloud
from wordcloud import WordCloud
from wordcloud import STOPWORDS
```

```
Collecting wordcloud
  Downloading wordcloud-1.8.0-cp37-cp37m-win_amd64.whl (157 kB)
Requirement already satisfied: pillow in c:\users\dipsikha\anaconda3\lib\site-packages (from
wordcloud) (7.0.0)
Requirement already satisfied: numpy>=1.6.1 in c:\users\dipsikha\anaconda3\lib\site-packages (from
wordcloud) (1.18.1)
Requirement already satisfied: matplotlib in c:\users\dipsikha\anaconda3\lib\site-packages (from
wordcloud) (3.1.3)
Requirement already satisfied: cycler>=0.10 in c:\users\dipsikha\anaconda3\lib\site-packages (from
matplotlib->wordcloud) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
c:\users\dipsikha\anaconda3\lib\site-packages (from matplotlib->wordcloud) (2.4.6)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\dipsikha\anaconda3\lib\site-
packages (from matplotlib->wordcloud) (2.8.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\dipsikha\anaconda3\lib\site-packages
(from matplotlib->wordcloud) (1.1.0)
Requirement already satisfied: six in c:\users\dipsikha\anaconda3\lib\site-packages (from
cycler>=0.10->matplotlib->wordcloud) (1.14.0)
Requirement already satisfied: setuptools in c:\users\dipsikha\anaconda3\lib\site-packages (from
kiwisolver>=1.0.1->matplotlib->wordcloud) (45.2.0.post20200210)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.8.0
```

In [57]:

```
stopword = set(STOPWORDS)
wordcloud = WordCloud(stopwords = stopword).generate(str(train['department']))

plt.rcParams['figure.figsize'] = (15, 8)
print(wordcloud)
plt.imshow(wordcloud)
plt.title('Most Popular Departments', fontsize = 10)
plt.axis('off')
plt.show()
```

```
<wordcloud.wordcloud.WordCloud object at 0x00000261E5395DC8>
```

In [61]:

```
train['education'].value_counts()
```

Out[61]:

```
Bachelor's          36669
Master's & above    14925
Below Secondary       805
Name: education, dtype: int64
```

In [62]:

```
######### pie chart of different education ###########

####### prepare data #####

df = train.groupby('education').size()
```

In [63]:

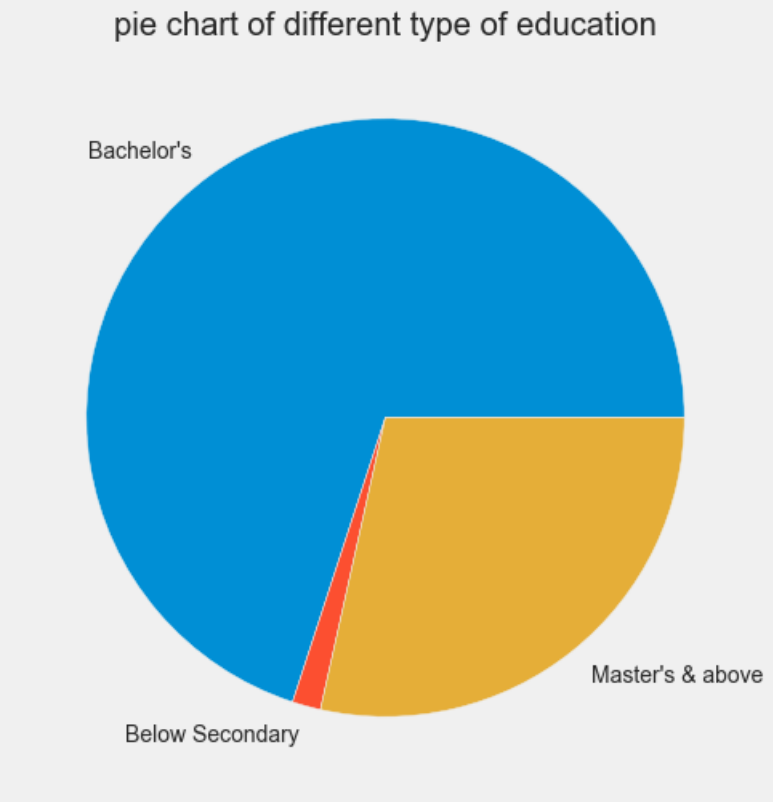
```
df
```

Out[63]:

```
education
Bachelor's          36669
Below Secondary       805
Master's & above    14925
dtype: int64
```

In [64]:

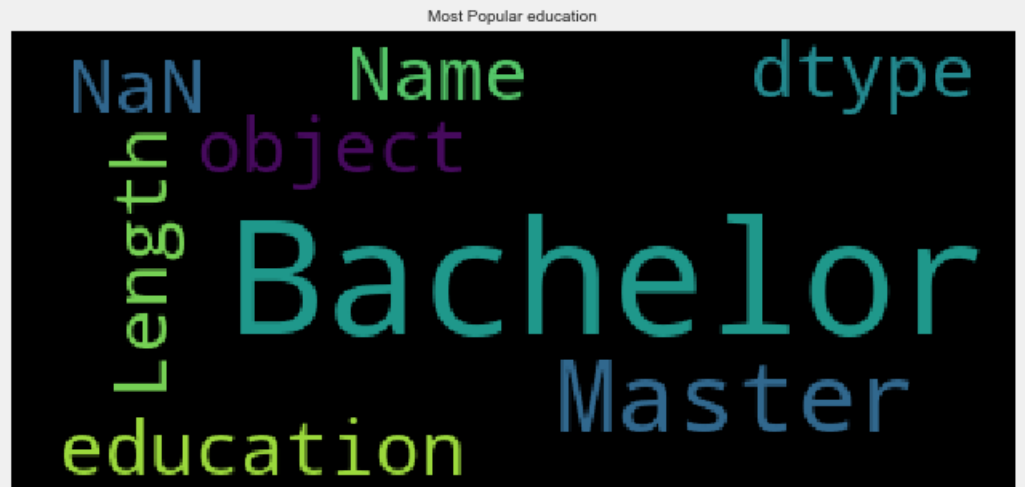
```
###### make plot with pandas ############

df.plot(kind = 'pie',subplots = True,figsize = (10,8))
plt.title('pie chart of different type of education')
plt.ylabel("")
plt.show()
```



In [67]:

```
###### most popular education degree among the employees ###########

stopword = set(STOPWORDS)
wordcloud = WordCloud(stopwords = stopword).generate(str(train['education']))

plt.rcParams['figure.figsize'] = (10, 8)
print(wordcloud)
plt.imshow(wordcloud)
plt.title('Most Popular education', fontsize = 10)
plt.axis('off')
plt.show()
```

```
<wordcloud.wordcloud.WordCloud object at 0x00000261E6CD6348>
```



In [68]:

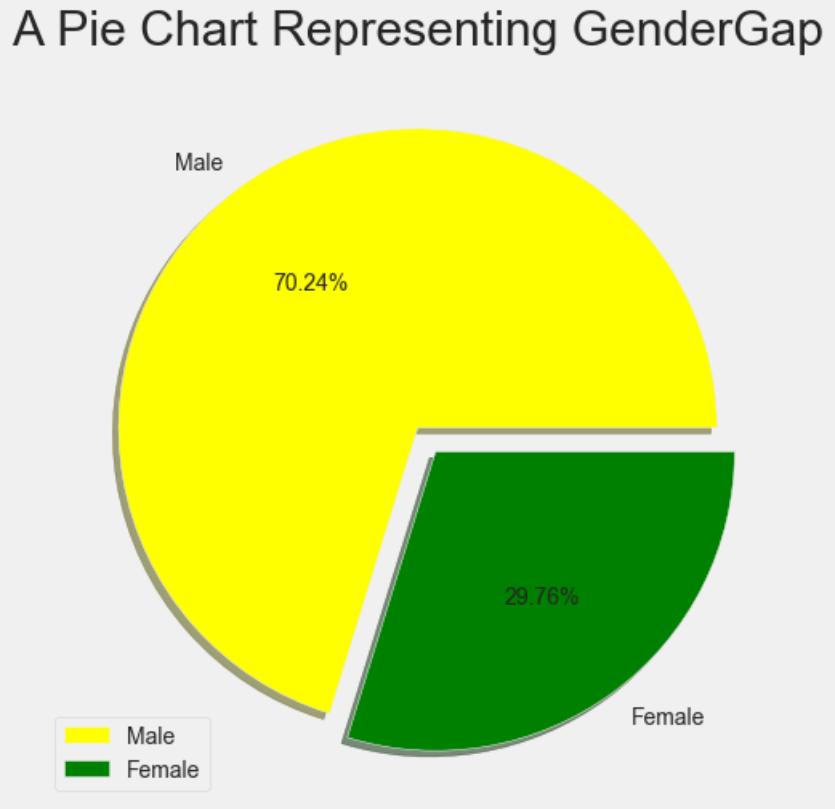
```
###### checking gender gap ###########

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

Out[68]:

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

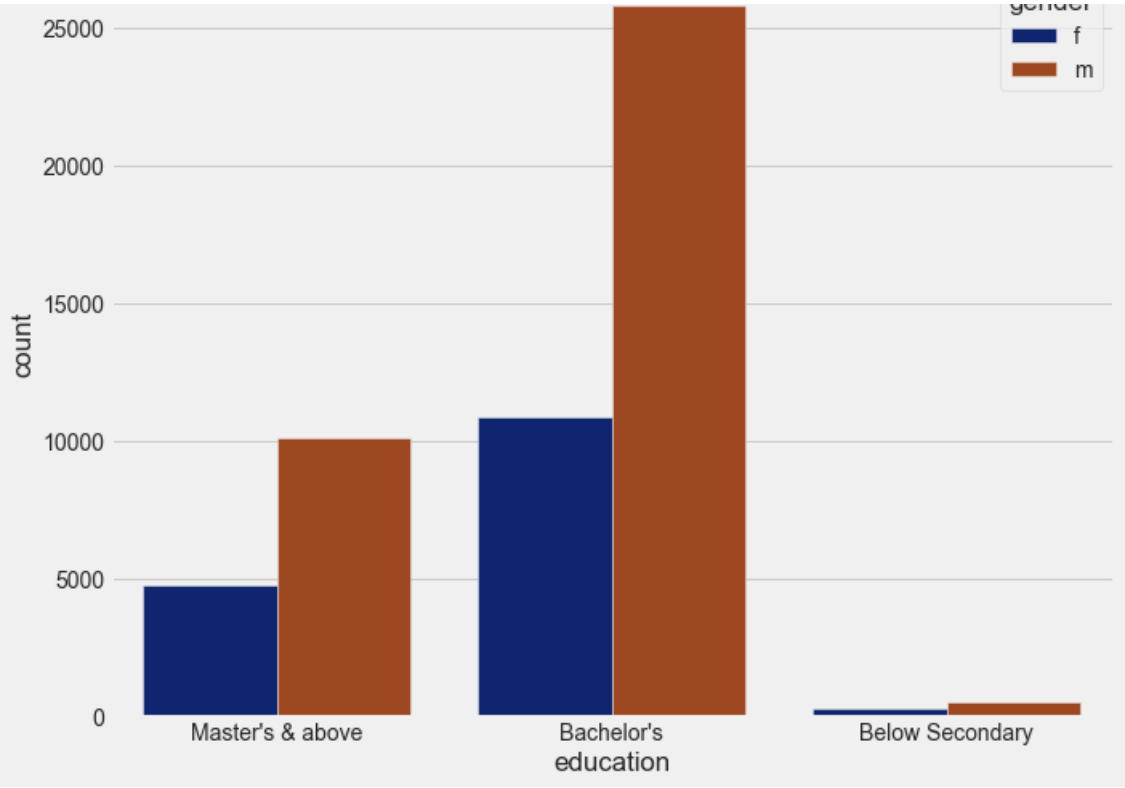
In [69]:

```
####### plotting the pie chart ###############

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

plt.subplots(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()
```



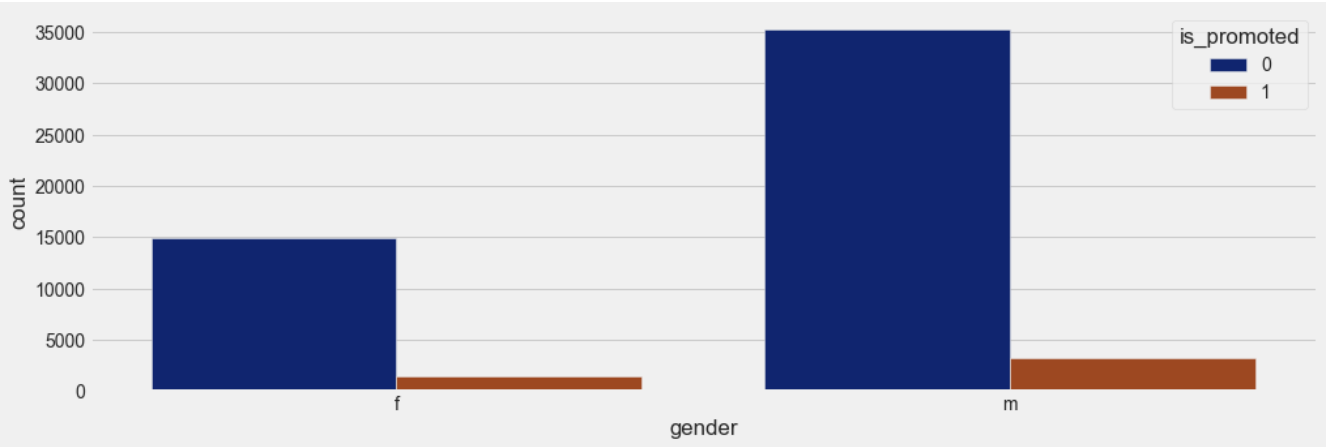
In [70]:

```
######## comparison of  gender male and female based on education  ############

plt.subplots(figsize = (10,8))
sns.countplot(x = 'education', data = train, hue = 'gender',palette = 'dark')
plt.show()
```



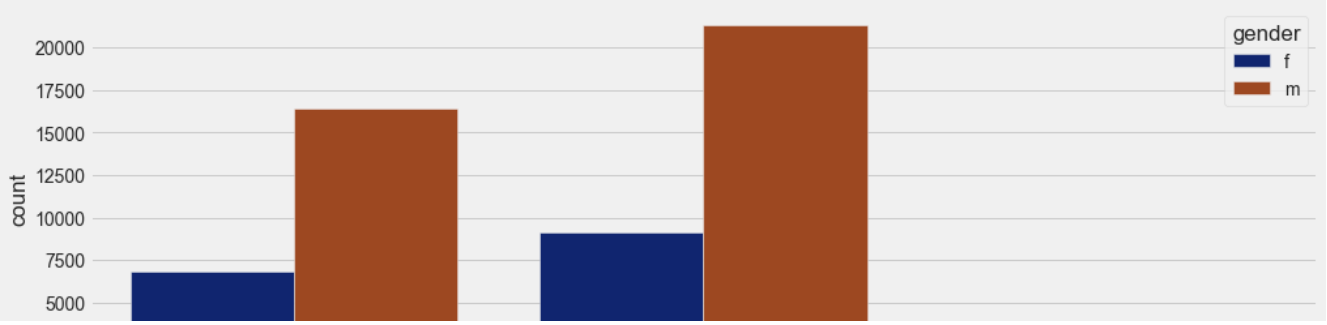
In [71]:

```
####### comparison of permoted male and female ############

plt.subplots(figsize=(15,5))
sns.countplot(x = 'gender', data = train, hue = 'is_promoted', palette = 'dark')
plt.show()
```



In [72]:

```
# comparison of permoted gender male & female based on recruitment channel ############
plt.subplots(figsize=(15,5))
sns.countplot(x = 'recruitment_channel', data = train, hue = 'gender', palette = 'dark')
plt.show()
```



In [73]:

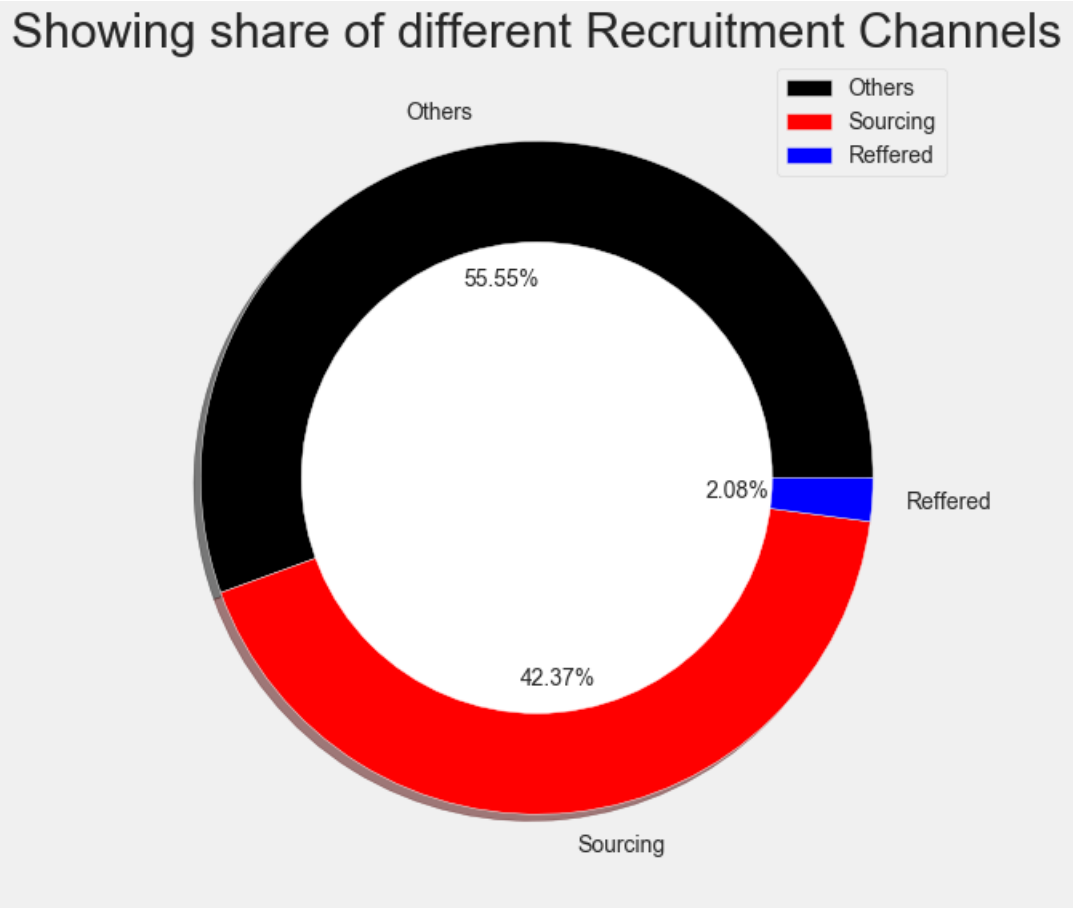
```
train['recruitment_channel'].value_counts()
```

Out[73]:

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

In [74]:

```
# plotting a donut chart for visualizing each of the recruitment channel's share

size = [30446, 23220, 1142]
colors = ['black', 'red', 'blue']
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()
```



In [75]:

```
######## distribution of age of employees ##########
```

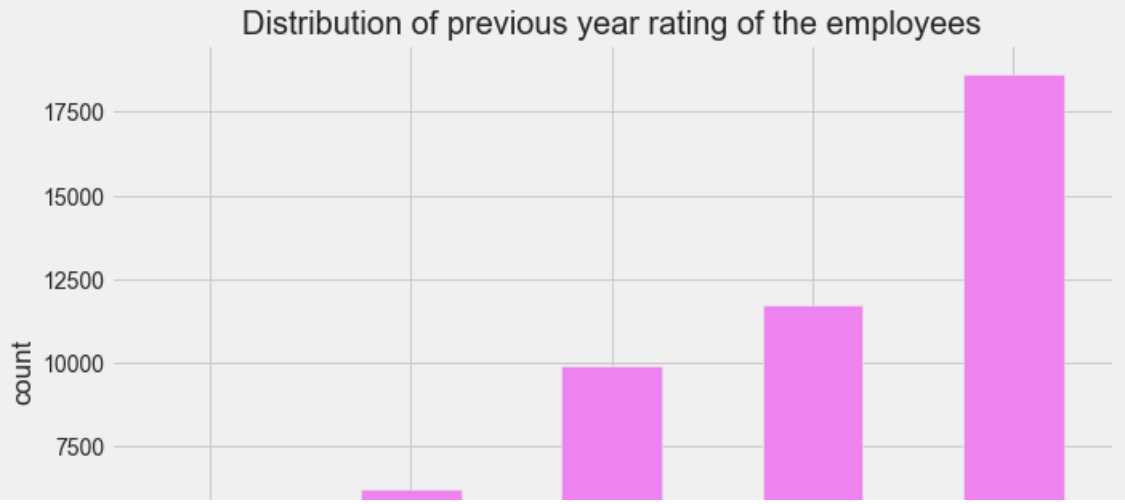
```
plt.subplots(figsize = (10,10))
sns.distplot(train['age'])
plt.title('Distribution of age of employees', fontsize = 20)
```

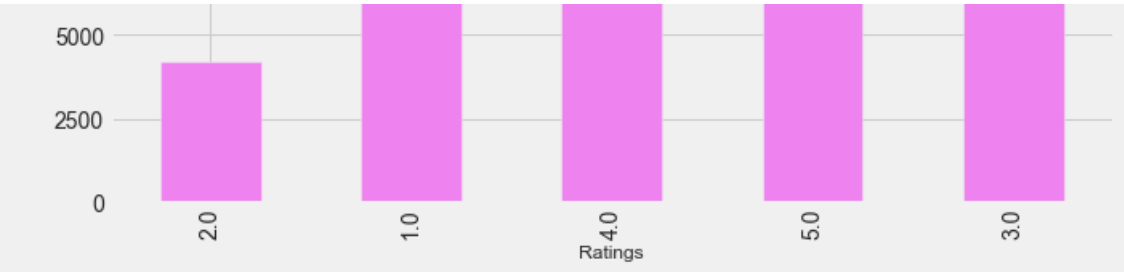
Out[75]:

```
Text(0.5, 1.0, 'Distribution of age of employees')
```

In [76]:

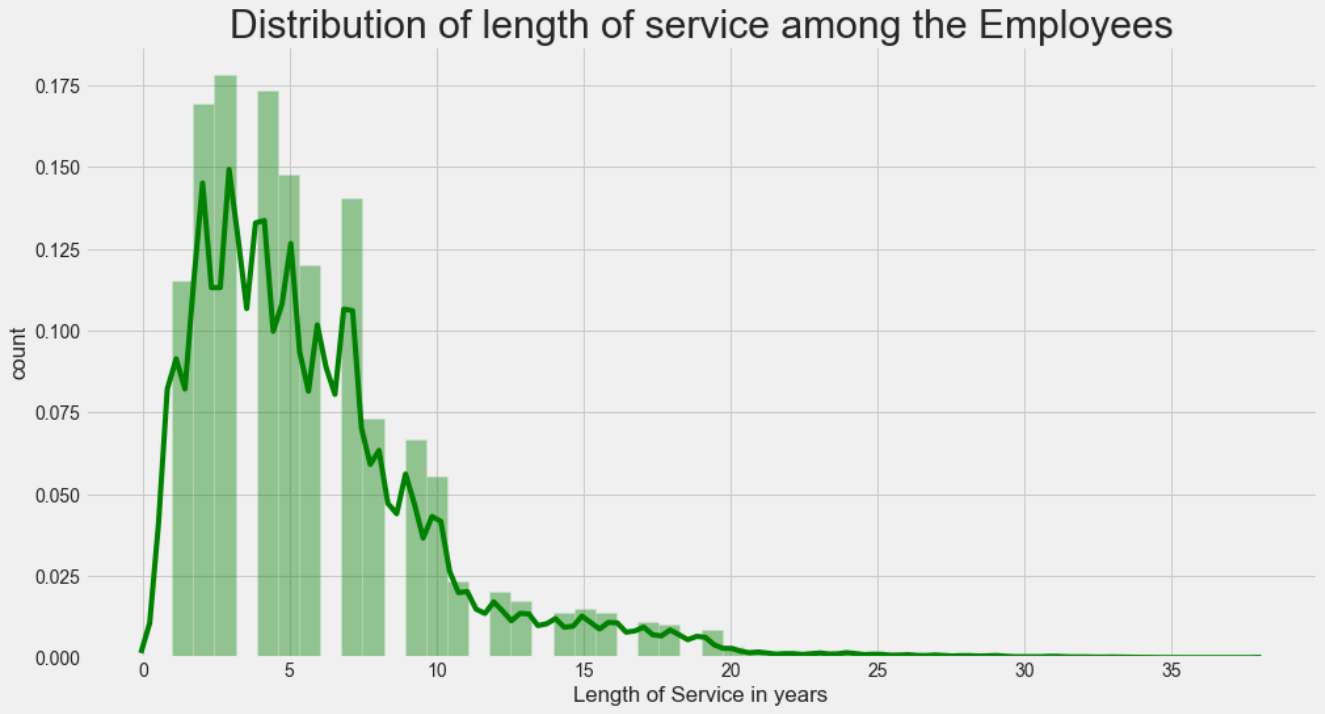
```
train['previous_year_rating'].value_counts().sort_values().plot.bar(color = 'violet',figsize = (10,
7))
plt.title('Distribution of previous year rating of the employees', fontsize = 20)
plt.xlabel('Ratings',fontsize =12)
plt.ylabel('count')
plt.show()
```





In [77]:

```
plt.subplots(figsize=(15,8))
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')
plt.ylabel('count')
plt.show()
```



In [78]:

```
train['KPIs_met >80%'].value_counts()
```

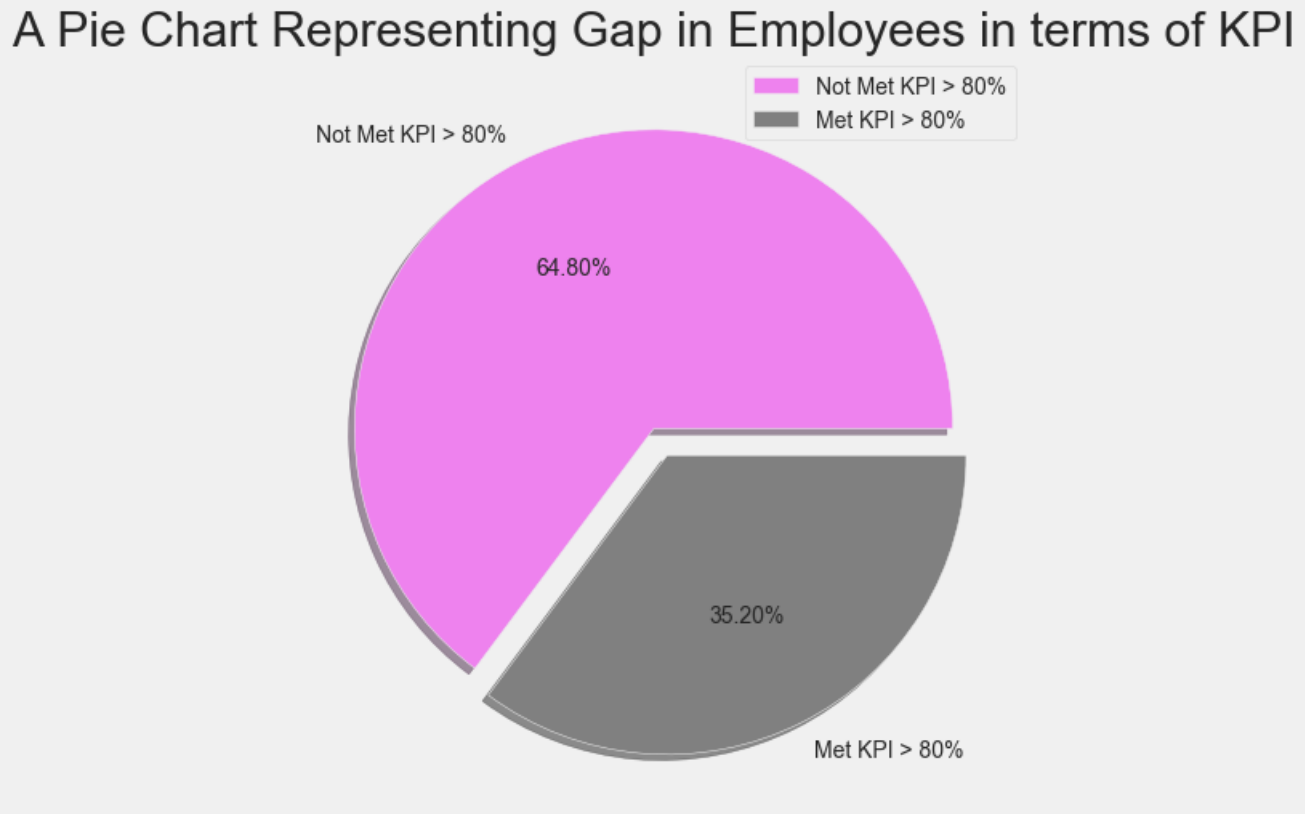
Out[78]:

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

In [79]:

```
# plotting a pie chart


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()
```



In [80]:

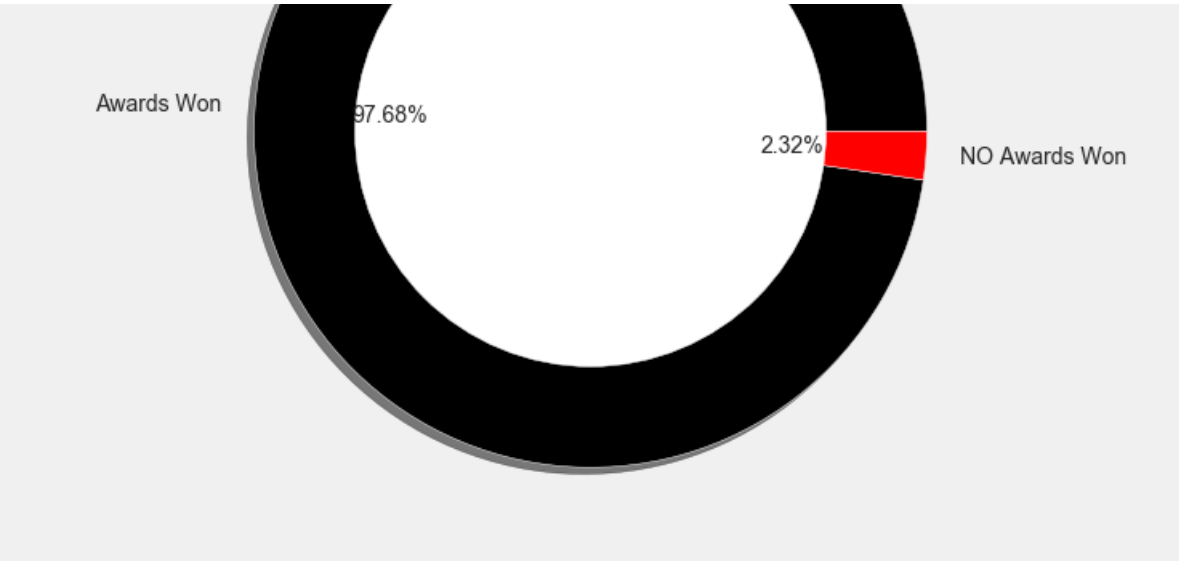
```
train['awards_won?'].value_counts()
```

Out[80]:

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

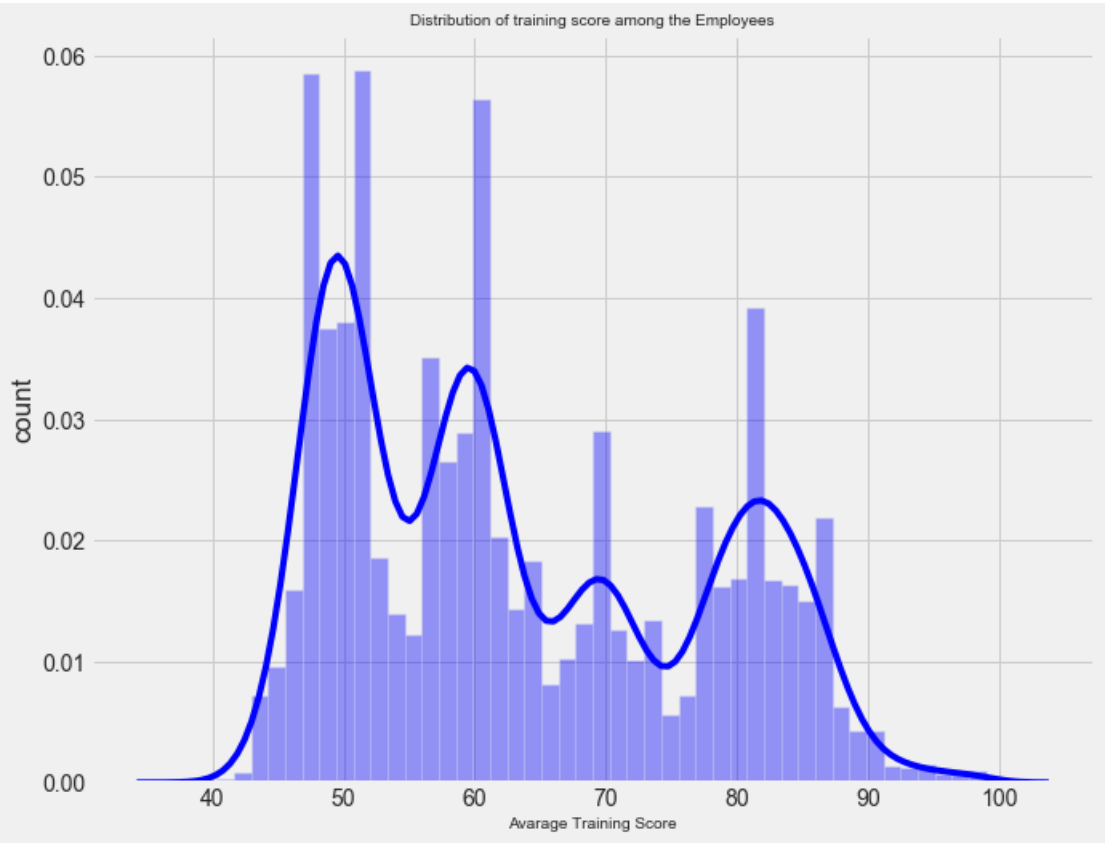
In [81]:

```
# plotting a donut chart for visualizing each of the recruitment channel's share

size = [53538, 1270]
colors = ['black', 'red']
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()
```



In [83]:

```
####### checking the distribution of avarage training score of the employees ##########

plt.subplots(figsize = (10,8))
sns.distplot(train['avg_training_score'],color = 'blue')
plt.title('Distribution of training score among the Employees', fontsize = 10)
plt.xlabel('Avarage Training Score', fontsize = 10)
plt.ylabel('count')
plt.show()
```



In [84]:

```
# checkig the no. of Employees Promoted

train['is_promoted'].value_counts()
```

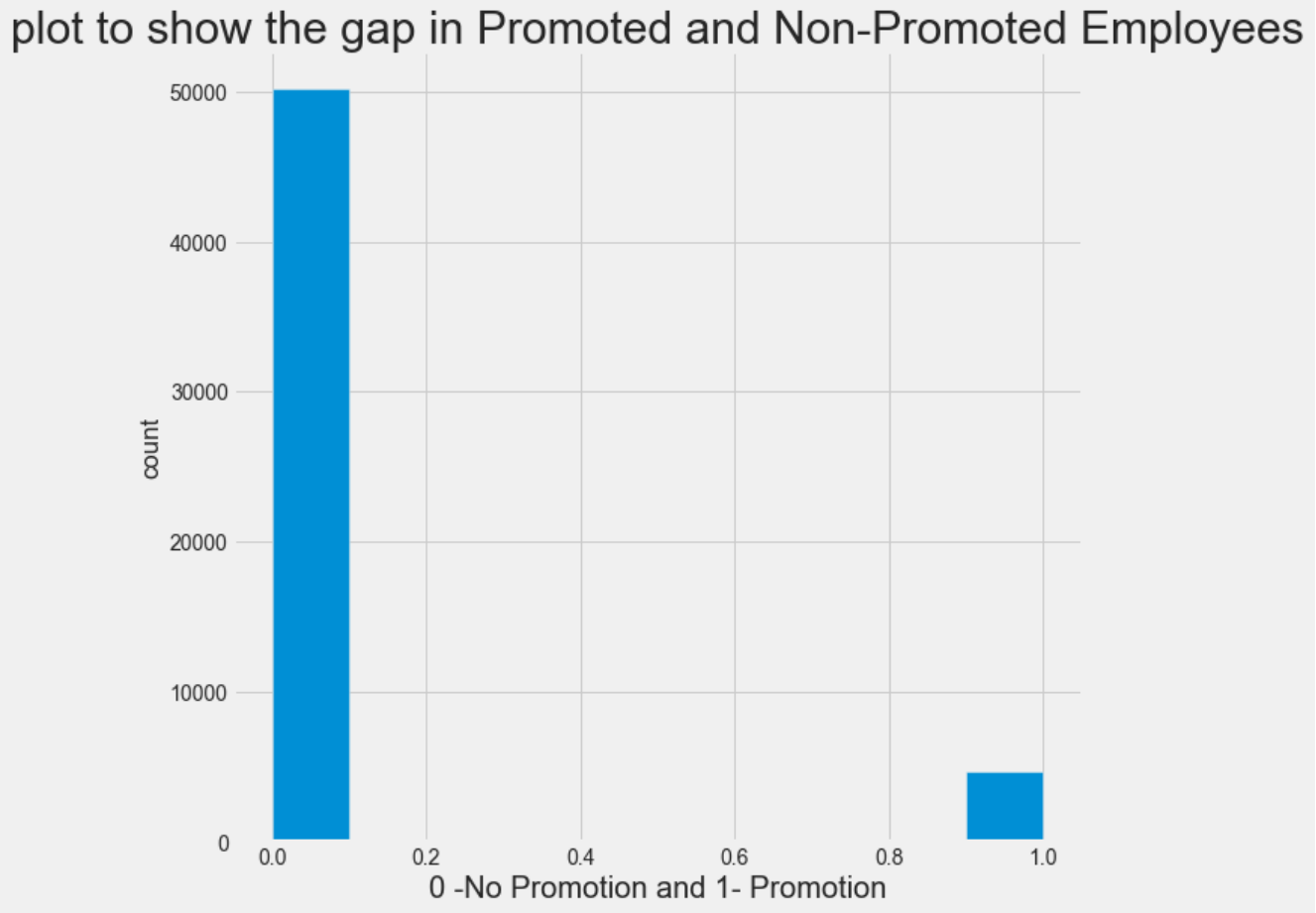
Out[84]:

```
0    50140
1     4668
```

```
Name: is_promoted, dtype: int64
```

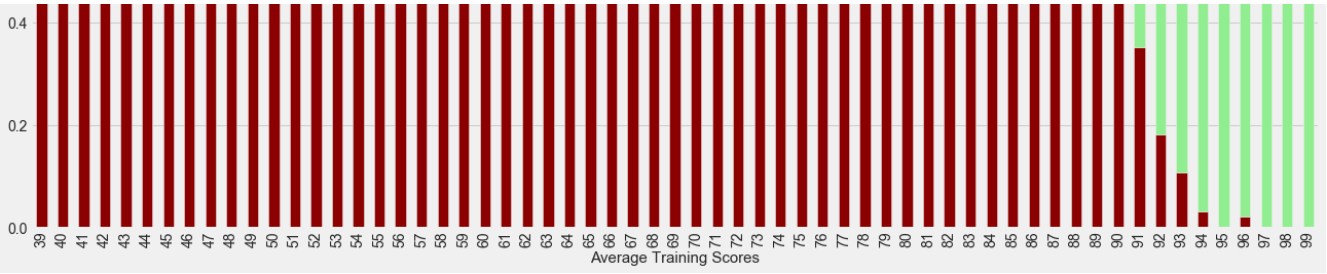
In [85]:

```
#plotting a scatter plot

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()
```



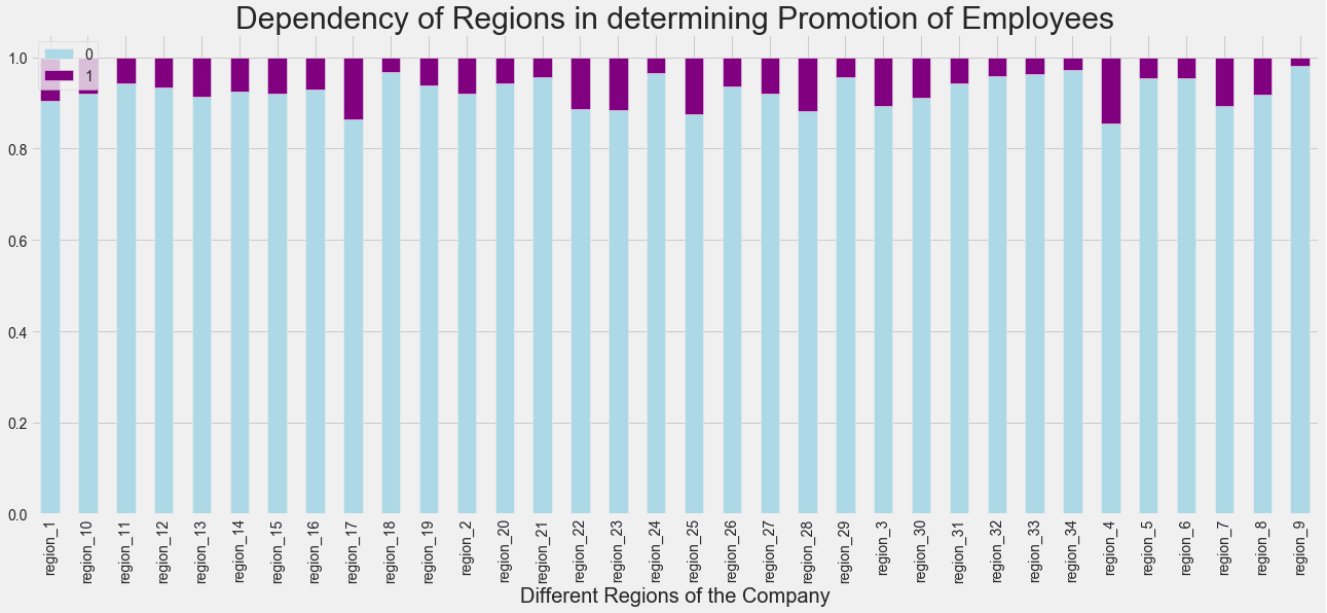
In [86]:

```
# scatter plot between average training score and 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 ##########
```



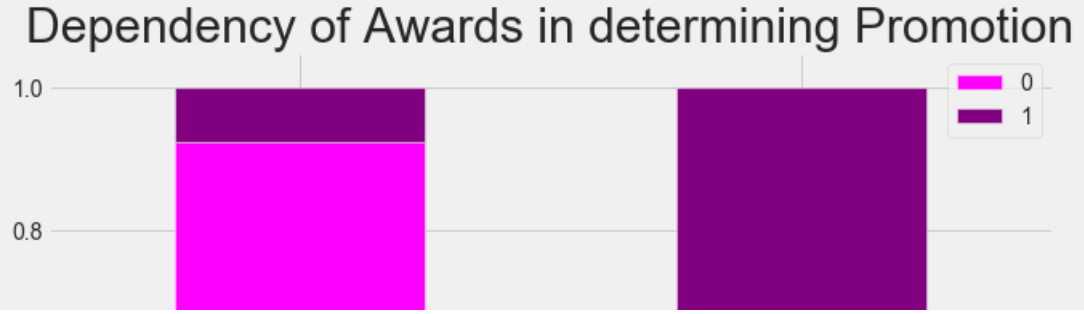
In [87]:

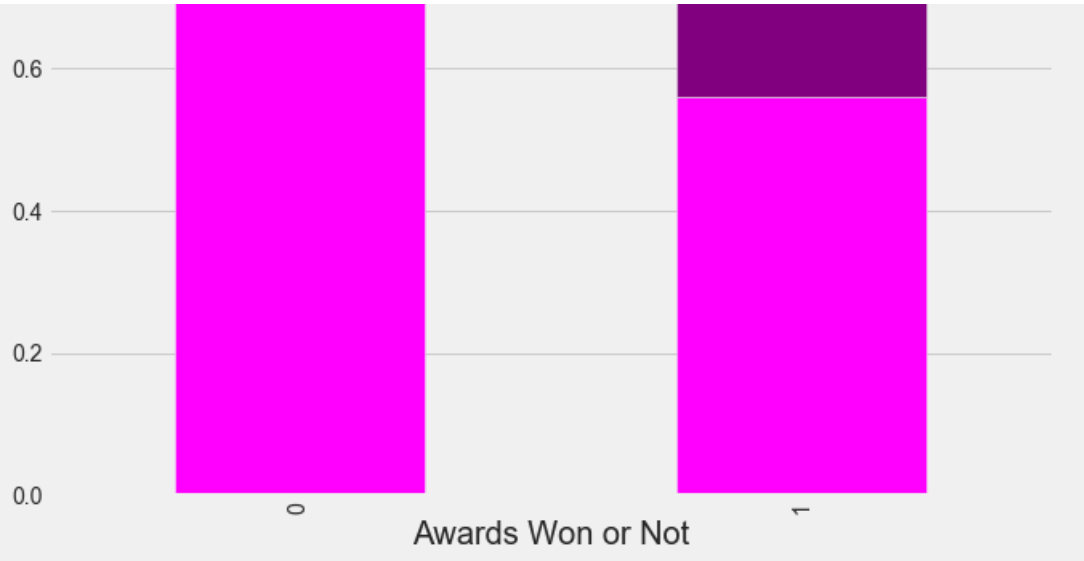
```
# checking dependency of different regions in promotion

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.###########
```



In [88]:

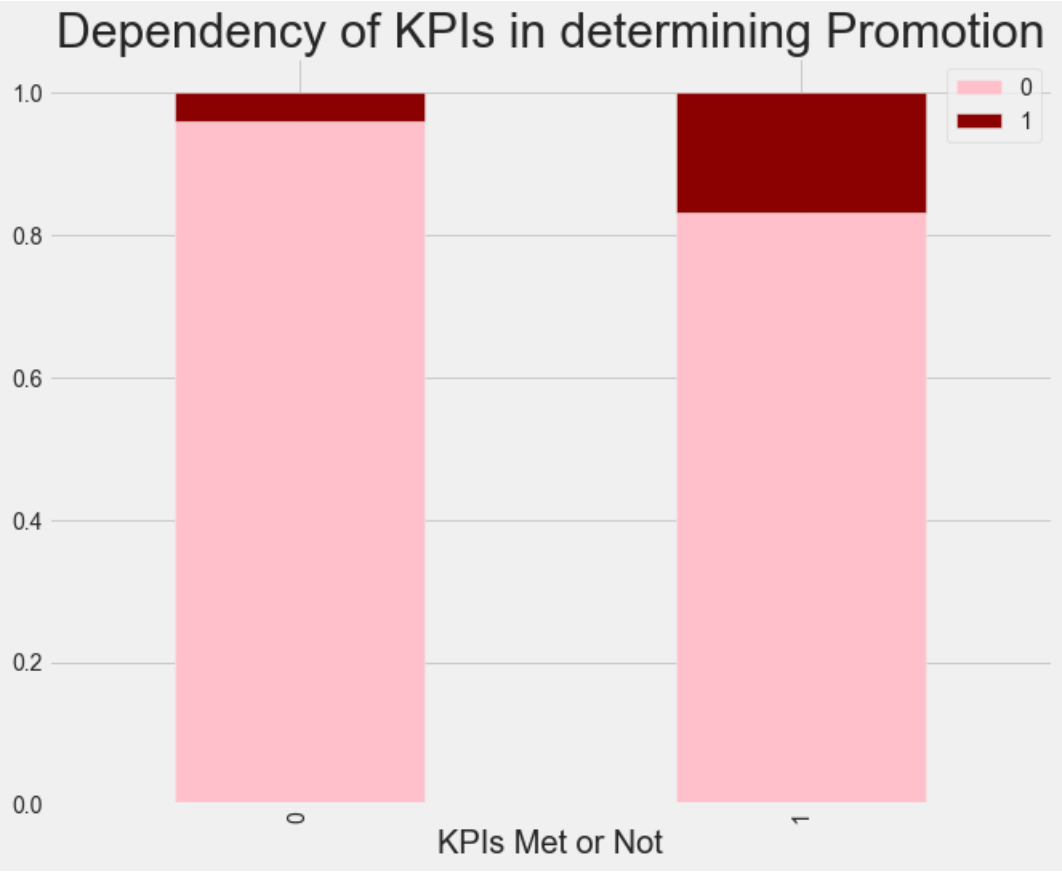
```
# dependency of awards won on promotion

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 aw
ard ###########
```





In [89]:

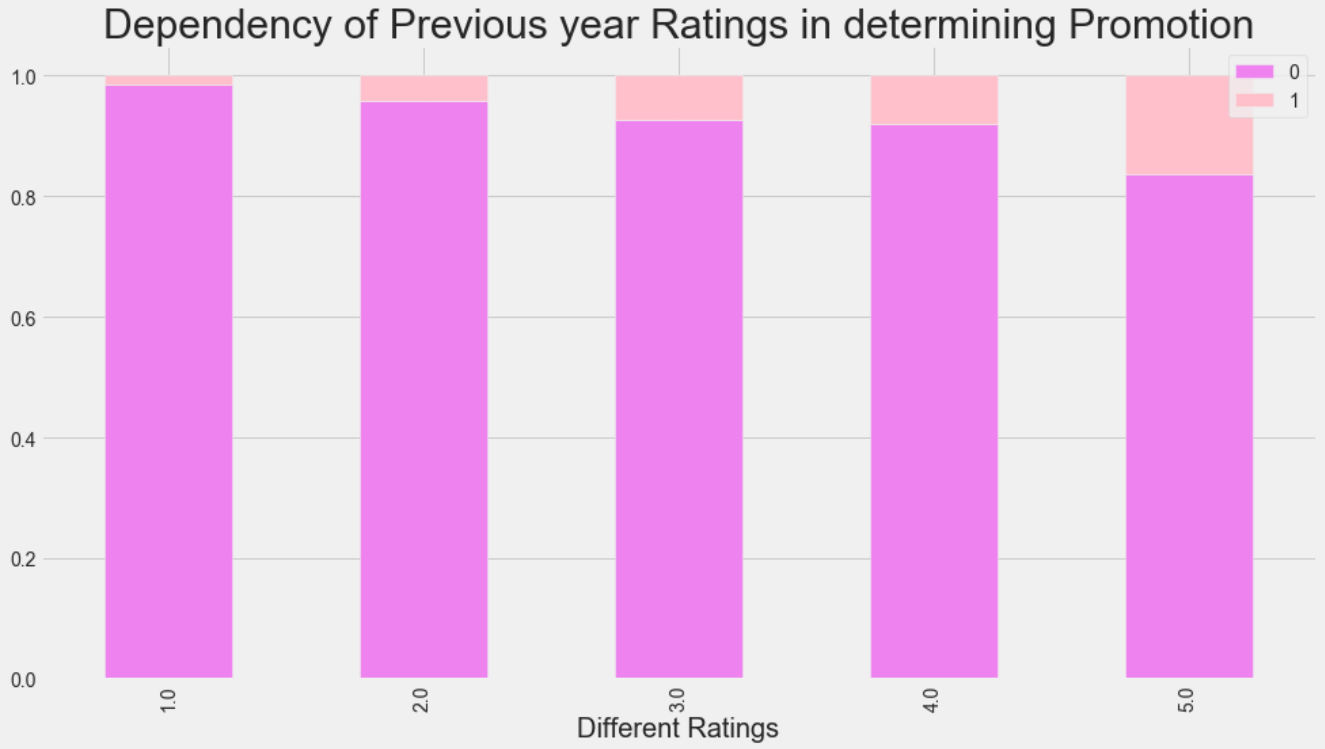
```
#dependency of KPIs with Promotion

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 i
n the company.  ##########
```



In [90]:

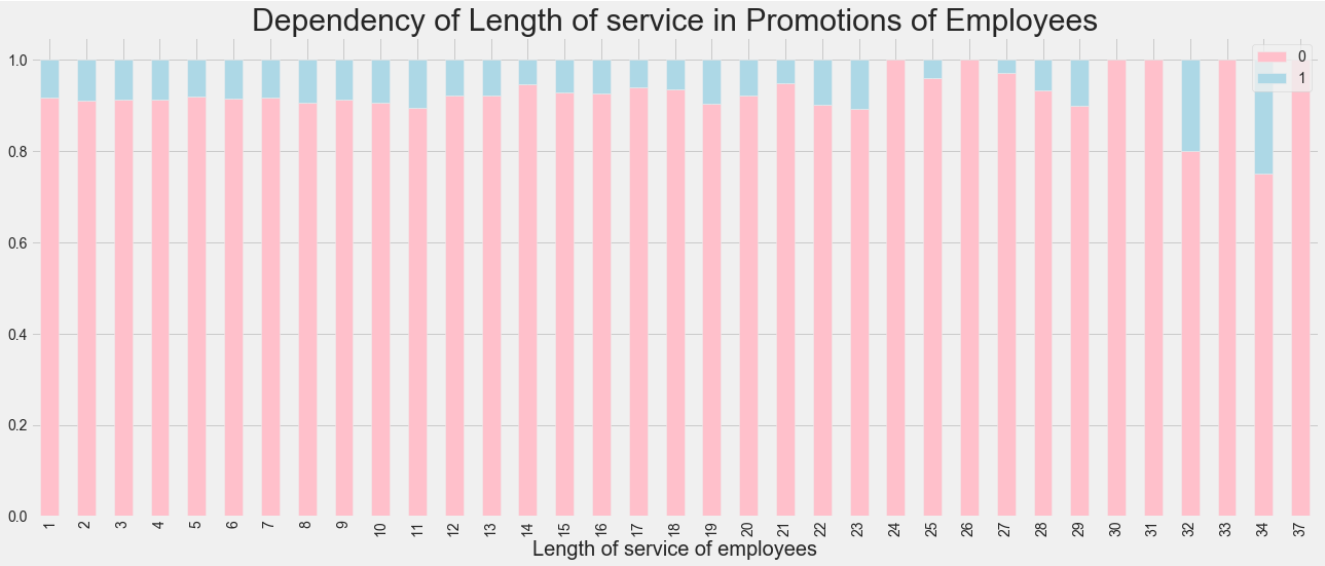
```
# checking dependency on previous years' ratings

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 ra
tings 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#################
```



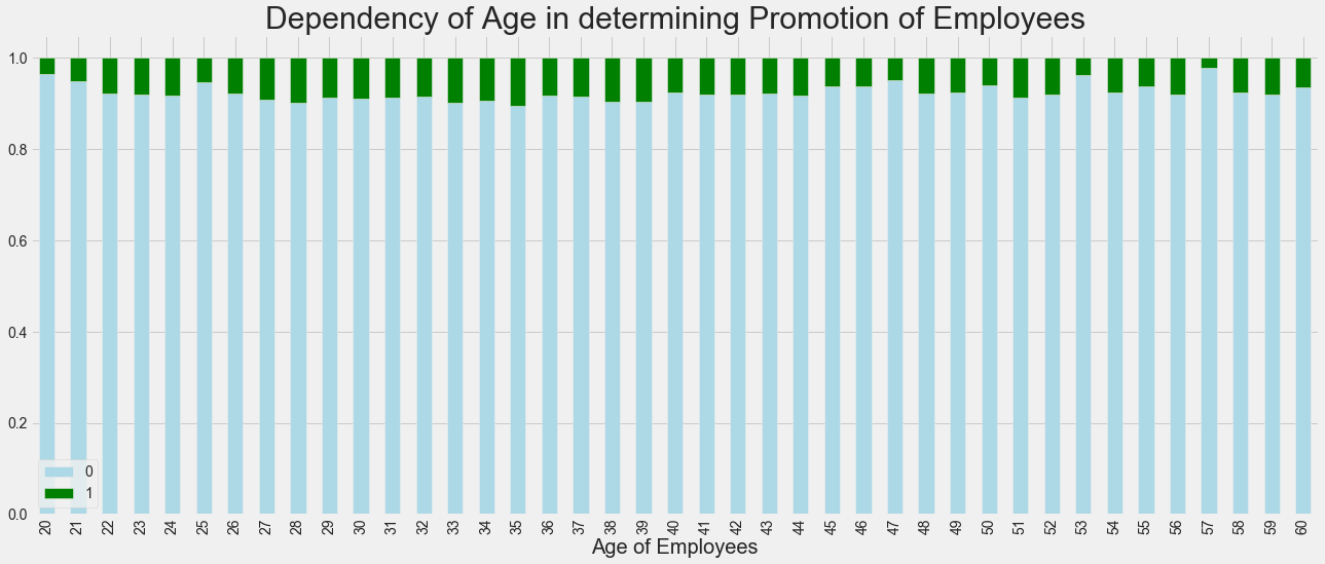
In [91]:

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

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()
```



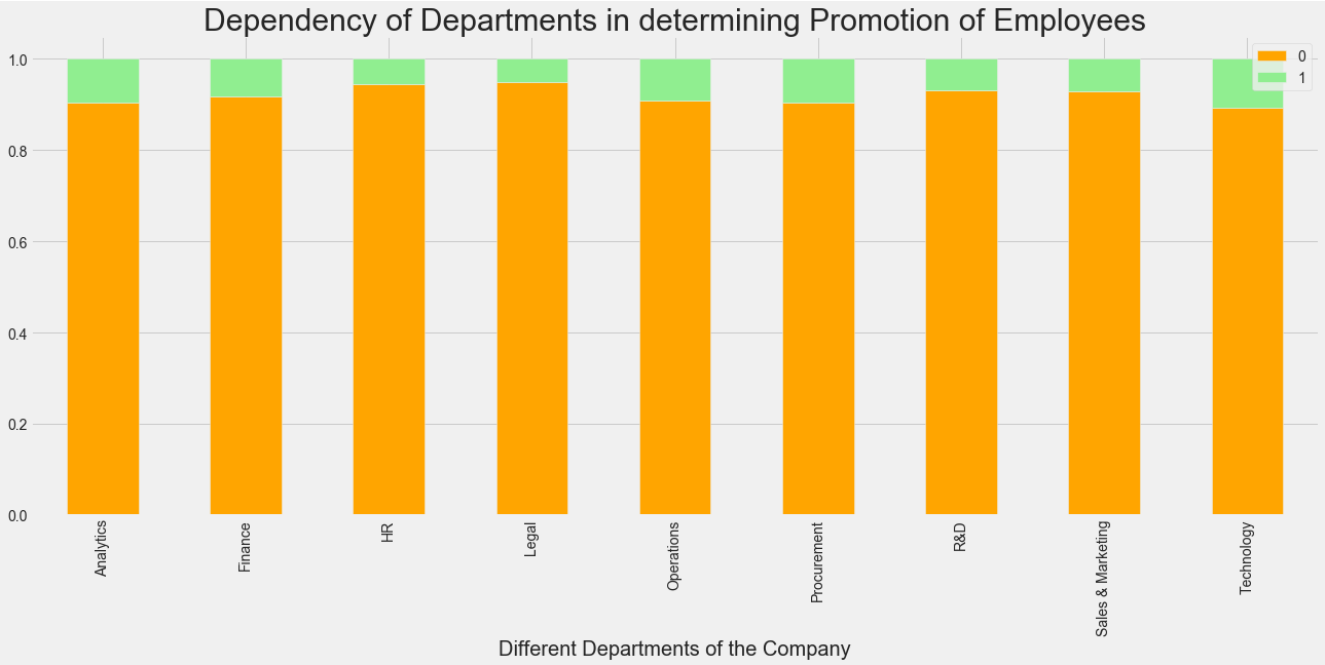
In [92]:

```
# 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####################
```



In [93]:

```
# 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'])

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.###############
```

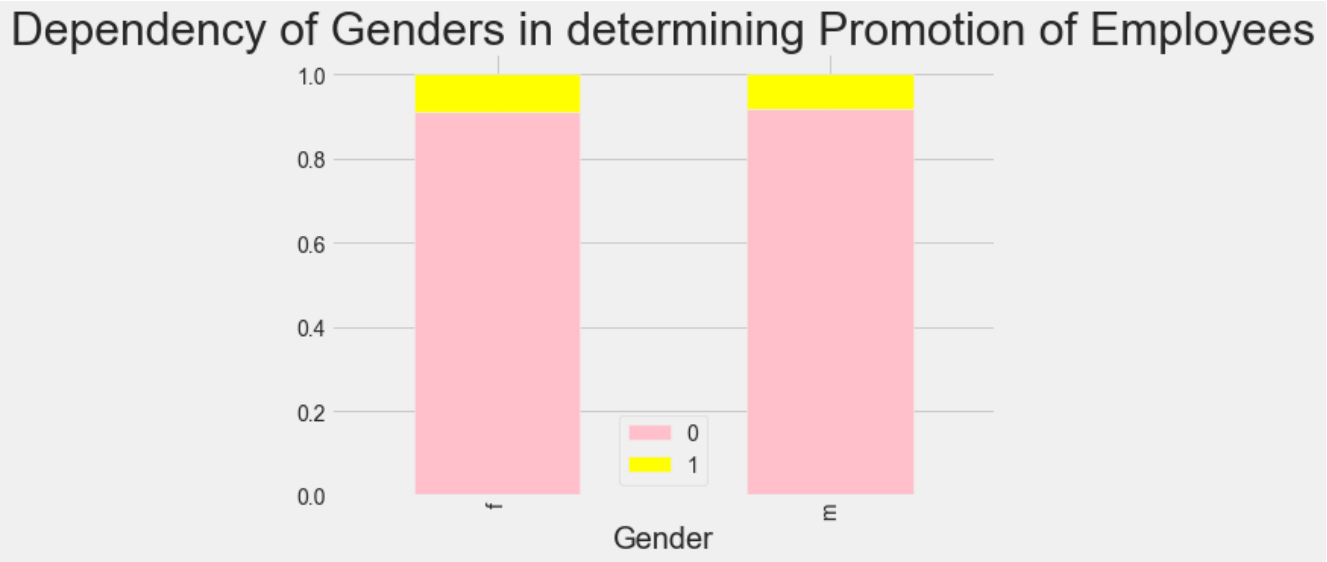


In [94]:

```
# 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()  ####### The above plot shows that there is no partiality between males and females i
n terms of promotion ##########
```

Out[94]:

```
<matplotlib.legend.Legend at 0x261e8d79f88>
```



In [95]:

```
######### data pre processing ##########

# filling missing values

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

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

Out[95]:

```
0
```

In [97]:

```
 # 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()
```

Out[97]:

```
0
```

In [98]:

```
# removing the employee_id column
```

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

train.columns
```

Out[98]:

```
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')
```

In [99]:

```
# saving the employee_id

emp_id = test['employee_id']

# removing the employee_id column

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

test.columns
```

Out[99]:

```
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')
```

In [100]:

```
# defining the test set

x_test = test

x_test.columns
```

Out[100]:

```
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')
```

In [101]:

```
# one hot encoding for the test set

x_test = pd.get_dummies(x_test)

x_test.columns
```

Out[101]:

```
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')
```

In [102]:

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

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

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

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

In [103]:

```
# one hot encoding for the train set

x = pd.get_dummies(x)

x.columns
```

Out[103]:

```
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')
```

In [107]:

```
########### model building ###########

!pip install imblearn

from imblearn.over_sampling import SMOTE

x_sample, y_sample = SMOTE().fit_sample(x, y.values.ravel())

x_sample = pd.DataFrame(x_sample)
y_sample = pd.DataFrame(y_sample)

# checking the sizes of the sample data
print("Size of x-sample :", x_sample.shape)
print("Size of y-sample :", y_sample.shape)
```

```
Requirement already satisfied: imblearn in c:\users\dipsikha\anaconda3\lib\site-packages (0.0)
Requirement already satisfied: imbalanced-learn in c:\users\dipsikha\anaconda3\lib\site-packages (
from imblearn) (0.7.0)
Requirement already satisfied: scikit-learn>=0.23 in c:\users\dipsikha\anaconda3\lib\site-packages
(from imbalanced-learn->imblearn) (0.23.2)
Requirement already satisfied: joblib>=0.11 in c:\users\dipsikha\anaconda3\lib\site-packages (from
```

```
imbalanced-learn->imblearn) (0.14.1)
Requirement already satisfied: numpy>=1.13.3 in c:\users\dipsikha\anaconda3\lib\site-packages
(from imbalanced-learn->imblearn) (1.18.1)
Requirement already satisfied: scipy>=0.19.1 in c:\users\dipsikha\anaconda3\lib\site-packages
(from imbalanced-learn->imblearn) (1.4.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\dipsikha\anaconda3\lib\site-
packages (from scikit-learn>=0.23->imbalanced-learn->imblearn) (2.1.0)
Size of x-sample : (100280, 58)
Size of y-sample : (100280, 1)
```

In [108]:

```
# splitting x and y into train and validation sets

from sklearn.model_selection import train_test_split

x_train, x_valid, y_train, y_valid = train_test_split(x_sample, y_sample, test_size = 0.2, random_s
tate = 0)

print("Shape of x_train: ", x_train.shape)
print("Shape of x_valid: ", x_valid.shape)
print("Shape of y_train: ", y_train.shape)
print("Shape of y_valid: ", y_valid.shape)
```

```
Shape of x_train:  (80224, 58)
Shape of x_valid:  (20056, 58)
Shape of y_train:  (80224, 1)
Shape of y_valid:  (20056, 1)
```

In [109]:

```
# standard scaling

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test  = sc.transform(x_test)
x_valid = sc.transform(x_valid)
```

In [111]:

```
######### random forest classifier ############

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import average_precision_score

rfc = RandomForestClassifier()
rfc.fit(x_train, y_train)

rfc_pred = rfc.predict(x_test)

print("Training Accuracy :", rfc.score(x_train, y_train))
```

```
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:9: DataConversionWarning: A co
lumn-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
  if __name__ == '__main__':
```

```
Training Accuracy : 0.9998254886318308
```

In [114]:

```
######### XG Boost classifier ###########

!pip install xgboost
from xgboost.sklearn import XGBClassifier
xgb = XGBClassifier()
xgb.fit(x_train, y_train)
```

```
xgb_pred = xgb.predict(x_test)

print("Training Accuracy :", xgb.score(x_train, y_train))
```

```
Requirement already satisfied: xgboost in c:\users\dipsikha\anaconda3\lib\site-packages (1.2.0)
Requirement already satisfied: scipy in c:\users\dipsikha\anaconda3\lib\site-packages (from
xgboost) (1.4.1)
Requirement already satisfied: numpy in c:\users\dipsikha\anaconda3\lib\site-packages (from
xgboost) (1.18.1)
Training Accuracy : 0.9651475867570801
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```