MAJOR PROJECT

September 5, 2022

1 EMPLOYEE PROMOTION AND STARTUP CASE STUDY

1.1 Importing Modules

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
```

1.2 Reading and Analysing the Data Set

[2]:
```
df=pd.read_csv("train.csv")
```

[3]:
```
df
```

[3]:

	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  awards_won?  avg_training_score  is_promoted
0                      8            0                  49            0
1                      4            0                  60            0
2                      7            0                  50            0
3                     10            0                  50            0
4                      2            0                  73            0
…                    …            …                   …            …
54803                 17            0                  78            0
54804                  6            0                  56            0
54805                  3            0                  79            0
54806                  2            0                  45            0
54807                  5            0                  49            0

[54808 rows x 13 columns]
```

[4]:
```
df=df.drop("employee_id",axis=1)
```

[5]:
```
df['is_promoted'].unique()
```

[5]:
```
array([0, 1])
```

[6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 54808 entries, 0 to 54807
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   department            54808 non-null  object
 1   region                54808 non-null  object
 2   education             52399 non-null  object
 3   gender                54808 non-null  object
 4   recruitment_channel   54808 non-null  object
 5   no_of_trainings       54808 non-null  int64
 6   age                   54808 non-null  int64
 7   previous_year_rating  50684 non-null  float64
 8   length_of_service     54808 non-null  int64
 9   awards_won?           54808 non-null  int64
 10  avg_training_score    54808 non-null  int64
 11  is_promoted           54808 non-null  int64
dtypes: float64(1), int64(6), object(5)
memory usage: 5.0+ MB
```

[7]:
```
df.isna().sum()
```

[7]:
```
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
awards_won?               0
avg_training_score        0
is_promoted               0
dtype: int64
```

[8]:
```
df.education[54806]
```

[8]:
```
nan
```

[9]:
```
df.education.describe()
```

[9]:
```
count          52399
unique             3
top       Bachelor's
freq           36669
Name: education, dtype: object
```

[10]:
```
df.education.isna().sum()
```

[10]:
```
2409
```

[11]:
```
df.education.unique()
```

[11]:
```
array(["Master's & above", "Bachelor's", nan, 'Below Secondary'],
      dtype=object)
```

[12]:
```
df.previous_year_rating.unique()
```

[12]:
```
array([ 5.,  3.,  1.,  4., nan,  2.])
```

[13]:
```
df.columns
```

[13]:
```
Index(['department', 'region', 'education', 'gender', 'recruitment_channel',
       'no_of_trainings', 'age', 'previous_year_rating', 'length_of_service',
       'awards_won?', 'avg_training_score', 'is_promoted'],
      dtype='object')
```

[14]:
```
df
```

[14]:
```
              department     region         education gender  \
0      Sales & Marketing   region_7  Master's & above      f
1             Operations  region_22        Bachelor's      m
2      Sales & Marketing  region_19        Bachelor's      m
3      Sales & Marketing  region_23        Bachelor's      m
4             Technology  region_26        Bachelor's      m
…                    …        …                …    …
54803         Technology  region_14        Bachelor's      m
54804         Operations  region_27  Master's & above      f
54805          Analytics   region_1        Bachelor's      m
54806  Sales & Marketing   region_9               NaN      m
54807                 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  awards_won?  avg_training_score  is_promoted
0                      8            0                  49            0
1                      4            0                  60            0
2                      7            0                  50            0
3                     10            0                  50            0
4                      2            0                  73            0
…                    …          …                  …          …
54803                 17            0                  78            0
54804                  6            0                  56            0
54805                  3            0                  79            0
54806                  2            0                  45            0
54807                  5            0                  49            0

[54808 rows x 12 columns]
```

[15]:
```
df.recruitment_channel.unique()
```

[15]:
```
array(['sourcing', 'other', 'referred'], dtype=object)
```

[16]:
```
df.avg_training_score.unique()
```

[16]:
```
array([49, 60, 50, 73, 85, 59, 63, 83, 54, 77, 80, 84, 51, 46, 75, 57, 70,
       68, 79, 44, 72, 61, 48, 58, 87, 47, 52, 88, 71, 65, 62, 53, 78, 91,
       82, 69, 55, 74, 86, 90, 92, 67, 89, 56, 76, 81, 45, 64, 39, 94, 93,
       66, 95, 42, 96, 40, 99, 43, 97, 41, 98])
```

[17]:
```
df.education.describe()
```

[17]:
```
count          52399
unique             3
top       Bachelor's
freq           36669
Name: education, dtype: object
```

[18]:
```
df["education"]=df['education'].fillna("Bachelor's")#Filling the null values␣
↪with the mode,if we fill with any other value,it might be irrelevant
```

[19]:
```
df["previous_year_rating"]=df['previous_year_rating'].fillna(0)
```

[20]:
```
df0=df.copy(deep=True)#Making a copy of test.csv instead of df0=df, because df␣
↪will be also changed if df0 is changed
```

[21]:
```
df.isna().sum()
```

[21]:
```
department              0
region                  0
education               0
gender                  0
recruitment_channel     0
no_of_trainings         0
age                     0
previous_year_rating    0
length_of_service       0
awards_won?             0
avg_training_score      0
is_promoted             0
dtype: int64
```

1.3 Data Preprocessing

1.3.1 Label Encoding for train dataset

[22]:
```
from sklearn.preprocessing import LabelEncoder
```

[23]:
```
le=LabelEncoder()
```

[24]:
```
cat=[]
for i in df.columns:
    if type(df[i][100])==str:
        cat.append(i)
```

[25]:
```
cat
```

[25]:
```
['department', 'region', 'education', 'gender', 'recruitment_channel']
```

[26]:
```
for cols in cat:
    le=LabelEncoder()
    df[cols]=le.fit_transform(df[cols])
```

[27]:
```
for i in df.columns:
    if type(df[i][100])==str:
        print(i)
```

1.3.2 Removing Unwanted columns

[28]:
```
X_train=df.drop("is_promoted",axis=1)
```

[29]:
```
Y_train=df.is_promoted
```

1.4 Reading the test dataset

[30]:
```
df2=pd.read_csv("test.csv")
```

[31]:
```
df2=df2.drop("employee_id",axis=1)
```

[ ]:
```

```

1.4.1 Filling the null Values

[32]:
```
df2.education.describe()
```

[32]:
```
count          22456
unique             3
top       Bachelor's
freq           15578
Name: education, dtype: object
```

[33]:
```
df2["education"]=df2['education'].fillna("Bachelor's")#Filling the null values␣
↪with the mode,if we fill with any other value,it might be irrelevant
```

[34]:
```

```

```
df2["previous_year_rating"]=df2['previous_year_rating'].fillna(0)#If the␣
↪previous year rate is not given, then most probably the employee's length oh␣
↪service might be less than 1 year
```

[35]:
```
X_test=df2
```

1.4.2 Label encoding for test dataset

[36]:
```
le2=LabelEncoder()
```

[37]:
```
cat2=[]
for i in df2.columns:
    if type(df2[i][100])==str:
        cat2.append(i)
```

[38]:
```
cat2
```

[38]:
```
['department', 'region', 'education', 'gender', 'recruitment_channel']
```

[39]:
```
for cols in cat2:
    le2=LabelEncoder()
    df2[cols]=le.fit_transform(df2[cols])
```

[ ]:
```

```

1.5 Predicting whether the employee is promoted or not

1.6 Decision Tree

[40]:
```
from sklearn.tree import DecisionTreeClassifier
```

[41]:
```
model=DecisionTreeClassifier()
```

[42]:
```
model=model.fit(X_train,Y_train)
```

[43]:
```
ypred=model.predict(X_test)
```

[44]:
```
ypred
```

[44]:
```
array([0, 0, 0, …, 0, 0, 1])
```

[45]:
```
df2
```

[45]:

	department	region	education	gender	recruitment_channel	\
0	8	18	0	1	2	
1	2	28	0	0	0	
2	7	4	0	1	0	

```
3               5      11           0       0                    0
4               1      21           0       1                    2
…             …      …          …       …                  …
23485           3      11           1       1                    2
23486           8      17           0       1                    2
23487           2       7           0       0                    2
23488           5      24           0       1                    2
23489           8       8           2       1                    0

       no_of_trainings  age  previous_year_rating  length_of_service  \
0                    1   24                   0.0                  1
1                    1   31                   3.0                  5
2                    1   31                   1.0                  4
3                    3   31                   2.0                  9
4                    1   30                   4.0                  7
…                  …  …                    …                …
23485                1   24                   3.0                  1
23486                1   31                   3.0                  7
23487                1   26                   4.0                  4
23488                3   27                   0.0                  1
23489                3   40                   5.0                  5

       awards_won?  avg_training_score
0                0                  77
1                0                  51
2                0                  47
3                0                  65
4                0                  61
…              …                  …
23485            0                  61
23486            0                  74
23487            0                  50
23488            0                  70
23489            0                  89

[23490 rows x 11 columns]
```

[46]:
```
len(ypred)
```

[46]:
```
23490
```

[47]:
```
ypred=np.array(ypred)
```

[48]:
```
np.unique(ypred,return_counts=True)
```

[48]:
```
(array([0, 1]), array([21130,  2360]))
```

[49]:
```
df3=df2.copy(deep=True)#Making a copy of test.csv instead of df3=df2, because
↪df2 will be also changed if df3 is changed
```

[50]:
```
df3["is_promoted"]=ypred
```

[51]:
```
df3
```

[51]:

	department	region	education	gender	recruitment_channel	\
0	8	18	0	1	2	
1	2	28	0	0	0	
2	7	4	0	1	0	
3	5	11	0	0	0	
4	1	21	0	1	2	
…	…	…	…	…	…	
23485	3	11	1	1	2	
23486	8	17	0	1	2	
23487	2	7	0	0	2	
23488	5	24	0	1	2	
23489	8	8	2	1	0	

	no_of_trainings	age	previous_year_rating	length_of_service	\
0	1	24	0.0	1	
1	1	31	3.0	5	
2	1	31	1.0	4	
3	3	31	2.0	9	
4	1	30	4.0	7	
…	…	…	…	…	
23485	1	24	3.0	1	
23486	1	31	3.0	7	
23487	1	26	4.0	4	
23488	3	27	0.0	1	
23489	3	40	5.0	5	

	awards_won?	avg_training_score	is_promoted
0	0	77	0
1	0	51	0
2	0	47	0
3	0	65	0
4	0	61	0
…	…	…	…
23485	0	61	0
23486	0	74	0
23487	0	50	0
23488	0	70	0
23489	0	89	1

[23490 rows x 12 columns]

[52]:
```
df3.tail()
```

[52]:

	department	region	education	gender	recruitment_channel	\
23485	3	11	1	1	2	
23486	8	17	0	1	2	
23487	2	7	0	0	2	
23488	5	24	0	1	2	
23489	8	8	2	1	0	

	no_of_trainings	age	previous_year_rating	length_of_service	\
23485	1	24	3.0	1	
23486	1	31	3.0	7	
23487	1	26	4.0	4	
23488	3	27	0.0	1	
23489	3	40	5.0	5	

	awards_won?	avg_training_score	is_promoted
23485	0	61	0
23486	0	74	0
23487	0	50	0
23488	0	70	0
23489	0	89	1

[ ]:
```

```

1.7 Random Forest

[53]:
```
from sklearn.ensemble import RandomForestClassifier
```

[54]:
```
model=RandomForestClassifier()
```

[55]:
```
model=model.fit(X_train,Y_train)#9 seconds
```

[56]:
```
ypred1=model.predict(X_test)
```

[57]:
```
ypred1
```

[57]:
```
array([0, 0, 0, …, 0, 0, 1])
```

[58]:
```
len(ypred1)
```

[58]:
```
23490
```

[59]:
```
ypred1=np.array(ypred1)
```

[60]:
```
np.unique(ypred1,return_counts=True)
```

[60]:
```
(array([0, 1]), array([22828,   662]))
```

[ ]:
```
```

[61]:
```
df4=df2.copy(deep=True)
```

[62]:
```
df4["is_promoted"]=ypred1
```

[63]:
```
df4
```

[63]:

	department	region	education	gender	recruitment_channel	\
0	8	18	0	1	2	
1	2	28	0	0	0	
2	7	4	0	1	0	
3	5	11	0	0	0	
4	1	21	0	1	2	
…	…	…	…	…	…	
23485	3	11	1	1	2	
23486	8	17	0	1	2	
23487	2	7	0	0	2	
23488	5	24	0	1	2	
23489	8	8	2	1	0	

	no_of_trainings	age	previous_year_rating	length_of_service	\
0	1	24	0.0	1	
1	1	31	3.0	5	
2	1	31	1.0	4	
3	3	31	2.0	9	
4	1	30	4.0	7	
…	…	…	…	…	
23485	1	24	3.0	1	
23486	1	31	3.0	7	
23487	1	26	4.0	4	
23488	3	27	0.0	1	
23489	3	40	5.0	5	

	awards_won?	avg_training_score	is_promoted
0	0	77	0
1	0	51	0
2	0	47	0
3	0	65	0
4	0	61	0
…	…	…	…
23485	0	61	0
23486	0	74	0
23487	0	50	0
23488	0	70	0

```
23489            0                  89            1

[23490 rows x 12 columns]
```

[64]:
```
df4.tail()
```

[64]:
```
       department  region  education  gender  recruitment_channel  \
23485           3      11          1       1                    2
23486           8      17          0       1                    2
23487           2       7          0       0                    2
23488           5      24          0       1                    2
23489           8       8          2       1                    0

       no_of_trainings  age  previous_year_rating  length_of_service  \
23485                1   24                   3.0                  1
23486                1   31                   3.0                  7
23487                1   26                   4.0                  4
23488                3   27                   0.0                  1
23489                3   40                   5.0                  5

       awards_won?  avg_training_score  is_promoted
23485            0                  61            0
23486            0                  74            0
23487            0                  50            0
23488            0                  70            0
23489            0                  89            1
```

[ ]:
```

```

1.8 Logistic Regression

[65]:
```
from sklearn.linear_model import LogisticRegression
```

[66]:
```
model=LogisticRegression(max_iter=1000)
```

[67]:
```
model=model.fit(X_train,Y_train)#to avoid this error -> run the min max scaler first
```

[68]:
```
ypred2=model.predict(X_test)
```

[69]:
```
ypred2
```

[69]:
```
array([0, 0, 0, …, 0, 0, 0])
```

[70]:
```
ypred2=np.array(ypred2)
```

[71]:
```
len(ypred2)
```

[71]:
```
23490
```

[72]:
```
np.unique(ypred2,return_counts=True)
```

[72]:
```
(array([0, 1]), array([23287,   203]))
```

[ ]:
```

```

[73]:
```
df4=df2.copy(deep=True)
```

[74]:
```
df4["is_promoted"]=ypred2
```

[75]:
```
df4
```

[75]:

	department	region	education	gender	recruitment_channel	\
0	8	18	0	1	2	
1	2	28	0	0	0	
2	7	4	0	1	0	
3	5	11	0	0	0	
4	1	21	0	1	2	
…	…	…	…	…	…	
23485	3	11	1	1	2	
23486	8	17	0	1	2	
23487	2	7	0	0	2	
23488	5	24	0	1	2	
23489	8	8	2	1	0	

	no_of_trainings	age	previous_year_rating	length_of_service	\
0	1	24	0.0	1	
1	1	31	3.0	5	
2	1	31	1.0	4	
3	3	31	2.0	9	
4	1	30	4.0	7	
…	…	…	…	…	
23485	1	24	3.0	1	
23486	1	31	3.0	7	
23487	1	26	4.0	4	
23488	3	27	0.0	1	
23489	3	40	5.0	5	

	awards_won?	avg_training_score	is_promoted
0	0	77	0
1	0	51	0
2	0	47	0
3	0	65	0

```
4                0                   61            0
…              …                   …             …
23485            0                   61            0
23486            0                   74            0
23487            0                   50            0
23488            0                   70            0
23489            0                   89            0

[23490 rows x 12 columns]
```

[76]:
```
df4.tail()
```

[76]:
```
       department  region  education  gender  recruitment_channel  \
23485           3      11          1       1                    2
23486           8      17          0       1                    2
23487           2       7          0       0                    2
23488           5      24          0       1                    2
23489           8       8          2       1                    0

       no_of_trainings  age  previous_year_rating  length_of_service  \
23485                1   24                   3.0                  1
23486                1   31                   3.0                  7
23487                1   26                   4.0                  4
23488                3   27                   0.0                  1
23489                3   40                   5.0                  5

       awards_won?  avg_training_score  is_promoted
23485            0                  61            0
23486            0                  74            0
23487            0                  50            0
23488            0                  70            0
23489            0                  89            0
```

1.8.1 Using all three models for prediction

[77]:
```
yfinal=[]
for i in range(len(ypred)):
    if (ypred[i]==1 or ypred1[i]==1 or ypred2[i]==1):
        yfinal.append(1)
    else:
        yfinal.append(0)
```

[78]:
```
yfinal=np.array(yfinal)
```

[79]:
```
yfinal
```

[79]:
```
array([0, 0, 0, …, 0, 0, 1])
```

[80]:
```
np.unique(yfinal,return_counts=True)
```

[80]:
```
(array([0, 1]), array([21019,  2471]))
```

[81]:
```
df2["is_promoted"]=yfinal
```

[82]:
```
df2 #Final predicted data set
```

[82]:

	department	region	education	gender	recruitment_channel	\
0	8	18	0	1	2	
1	2	28	0	0	0	
2	7	4	0	1	0	
3	5	11	0	0	0	
4	1	21	0	1	2	
…	…	…	…	…	…	
23485	3	11	1	1	2	
23486	8	17	0	1	2	
23487	2	7	0	0	2	
23488	5	24	0	1	2	
23489	8	8	2	1	0	

	no_of_trainings	age	previous_year_rating	length_of_service	\
0	1	24	0.0	1	
1	1	31	3.0	5	
2	1	31	1.0	4	
3	3	31	2.0	9	
4	1	30	4.0	7	
…	…	…	…	…	
23485	1	24	3.0	1	
23486	1	31	3.0	7	
23487	1	26	4.0	4	
23488	3	27	0.0	1	
23489	3	40	5.0	5	

	awards_won?	avg_training_score	is_promoted
0	0	77	0
1	0	51	0
2	0	47	0
3	0	65	0
4	0	61	0
…	…	…	…
23485	0	61	0
23486	0	74	0
23487	0	50	0
23488	0	70	0
23489	0	89	1

```
[23490 rows x 12 columns]
```

[83]:
```
df2.tail()
```

[83]:
```
       department  region  education  gender  recruitment_channel  \
23485           3      11          1       1                    2
23486           8      17          0       1                    2
23487           2       7          0       0                    2
23488           5      24          0       1                    2
23489           8       8          2       1                    0

       no_of_trainings  age  previous_year_rating  length_of_service  \
23485                1   24                   3.0                  1
23486                1   31                   3.0                  7
23487                1   26                   4.0                  4
23488                3   27                   0.0                  1
23489                3   40                   5.0                  5

       awards_won?  avg_training_score  is_promoted
23485            0                  61            0
23486            0                  74            0
23487            0                  50            0
23488            0                  70            0
23489            0                  89            1
```

1.9 Checking the Accuracy

[84]:
```
X_train=df2.drop("is_promoted",axis=1)
```

[85]:
```
Y_train=df2.is_promoted
```

[86]:
```
X_train
```

[86]:
```
       department  region  education  gender  recruitment_channel  \
0               8      18          0       1                    2
1               2      28          0       0                    0
2               7       4          0       1                    0
3               5      11          0       0                    0
4               1      21          0       1                    2
…             …      …         …      …                    …
23485           3      11          1       1                    2
23486           8      17          0       1                    2
23487           2       7          0       0                    2
23488           5      24          0       1                    2
23489           8       8          2       1                    0

       no_of_trainings  age  previous_year_rating  length_of_service  \
```

```
0                    1   24                   0.0                    1
1                    1   31                   3.0                    5
2                    1   31                   1.0                    4
3                    3   31                   2.0                    9
4                    1   30                   4.0                    7
…                  …   …                    …                  …
23485                1   24                   3.0                    1
23486                1   31                   3.0                    7
23487                1   26                   4.0                    4
23488                3   27                   0.0                    1
23489                3   40                   5.0                    5

       awards_won?  avg_training_score
0                0                   77
1                0                   51
2                0                   47
3                0                   65
4                0                   61
…              …                   …
23485            0                   61
23486            0                   74
23487            0                   50
23488            0                   70
23489            0                   89

[23490 rows x 11 columns]
```

[87]:
```
Y_train
```

[87]:
```
0        0
1        0
2        0
3        0
4        0
        ..
23485    0
23486    0
23487    0
23488    0
23489    1
Name: is_promoted, Length: 23490, dtype: int64
```

[88]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X_train,Y_train,test_size=0.
↪4,random_state=10)
```

[89]:
```
model=LogisticRegression(max_iter=1000)
```

[ ]:

[90]:
```
model=model.fit(X_train,Y_train)
```

[91]:
```
ypred5=model.predict(X_test)
```

[92]:
```
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

[93]:
```
accuracy_score(Y_test,ypred5)#90% Accuracy
```

[93]:
```
0.9024052788420605
```

[94]:
```
print(classification_report(Y_test,ypred5))
```

```
              precision    recall  f1-score   support

           0       0.90      1.00      0.95      8398
           1       0.80      0.11      0.19       998

    accuracy                           0.90      9396
   macro avg       0.85      0.55      0.57      9396
weighted avg       0.89      0.90      0.87      9396
```

[95]:
```
#Now do EDA with df0(the train dataset)
```

2 VISUALIZATION

2.1 UNIVARIATE ANALYSIS

[96]:
```
df0
```

[96]:
```
              department     region         education gender  \
0      Sales & Marketing   region_7  Master's & above      f
1             Operations  region_22         Bachelor's      m
2      Sales & Marketing  region_19         Bachelor's      m
3      Sales & Marketing  region_23         Bachelor's      m
4             Technology  region_26         Bachelor's      m
…                    …         …                  …    …
54803         Technology  region_14         Bachelor's      m
54804         Operations  region_27  Master's & above      f
54805          Analytics   region_1         Bachelor's      m
54806  Sales & Marketing   region_9         Bachelor's      m
54807                 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  awards_won?  avg_training_score  is_promoted
0                      8            0                  49            0
1                      4            0                  60            0
2                      7            0                  50            0
3                     10            0                  50            0
4                      2            0                  73            0
…                    …            …                  …           …
54803                 17            0                  78            0
54804                  6            0                  56            0
54805                  3            0                  79            0
54806                  2            0                  45            0
54807                  5            0                  49            0

[54808 rows x 12 columns]
```

[97]:
```
df0.columns
```

[97]:
```
Index(['department', 'region', 'education', 'gender', 'recruitment_channel',
       'no_of_trainings', 'age', 'previous_year_rating', 'length_of_service',
       'awards_won?', 'avg_training_score', 'is_promoted'],
      dtype='object')
```

[98]:
```
df0["gender"].value_counts()
```

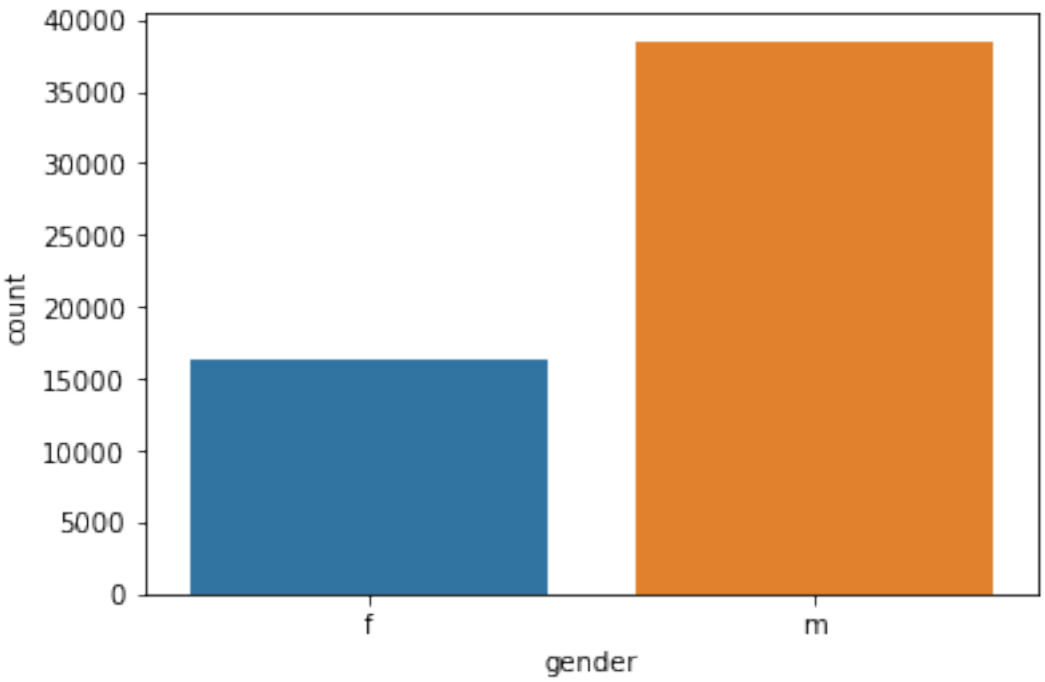
[98]:
```
m    38496
f    16312
Name: gender, dtype: int64
```

[99]:
```
df0["gender"].describe()
```

[99]:
```
count     54808
unique        2
top           m
freq      38496
Name: gender, dtype: object
```

2.1.1 GENDER

[100]:
```
sb.countplot(x=df0["gender"])
plt.show()
#Male employees are more in number than the female employees
```



[101]:
```
df0["is_promoted"].value_counts()
```

[101]:
```
0    50140
1     4668
Name: is_promoted, dtype: int64
```

[102]:
```
df0["is_promoted"].describe()
```

[102]:
```
count    54808.000000
mean         0.085170
std          0.279137
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max          1.000000
Name: is_promoted, dtype: float64
```

[103]:
```
df0['avg_training_score'].max()
```

[103]:
```
99
```

[104]:
```
df0['avg_training_score'].min()
```

[104]:
```
39
```

[105]:
```
df0['avg_training_score'].median()
```

[105]:
```
60.0
```

[ ]:
```

```

[106]:
```
df0.groupby(['is_promoted']).count()
```

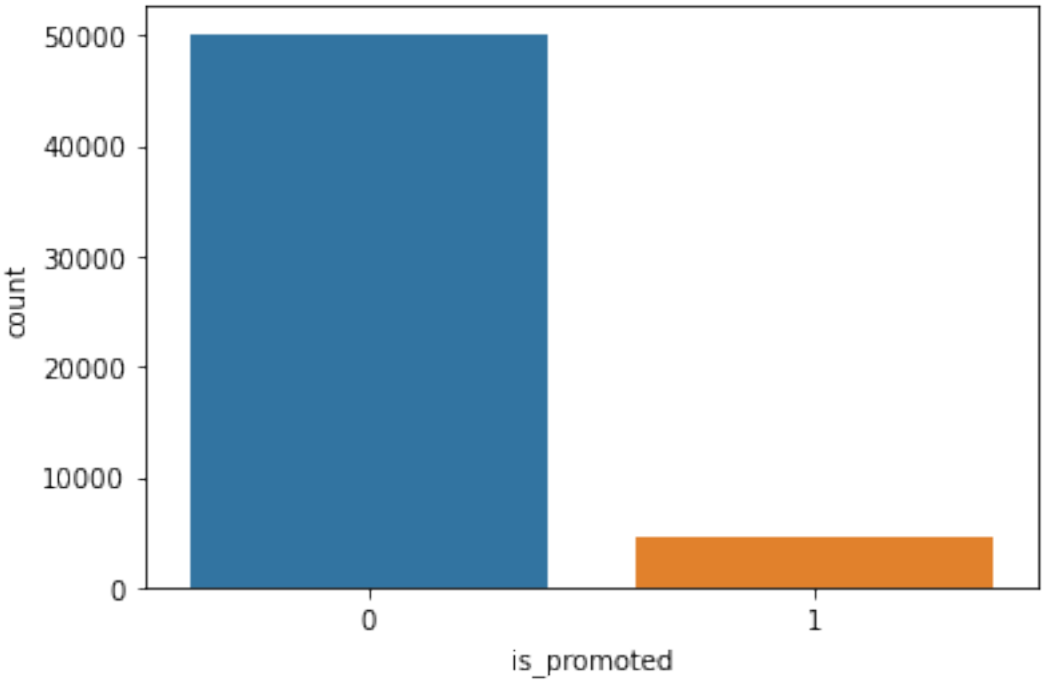
[106]:
```
              department  region  education  gender  recruitment_channel  \
is_promoted
0                  50140   50140      50140   50140                50140
1                   4668    4668       4668    4668                 4668

              no_of_trainings    age  previous_year_rating  length_of_service  \
is_promoted
0                       50140  50140                 50140              50140
1                        4668   4668                  4668               4668

              awards_won?  avg_training_score
is_promoted
0                   50140               50140
1                    4668                4668
```

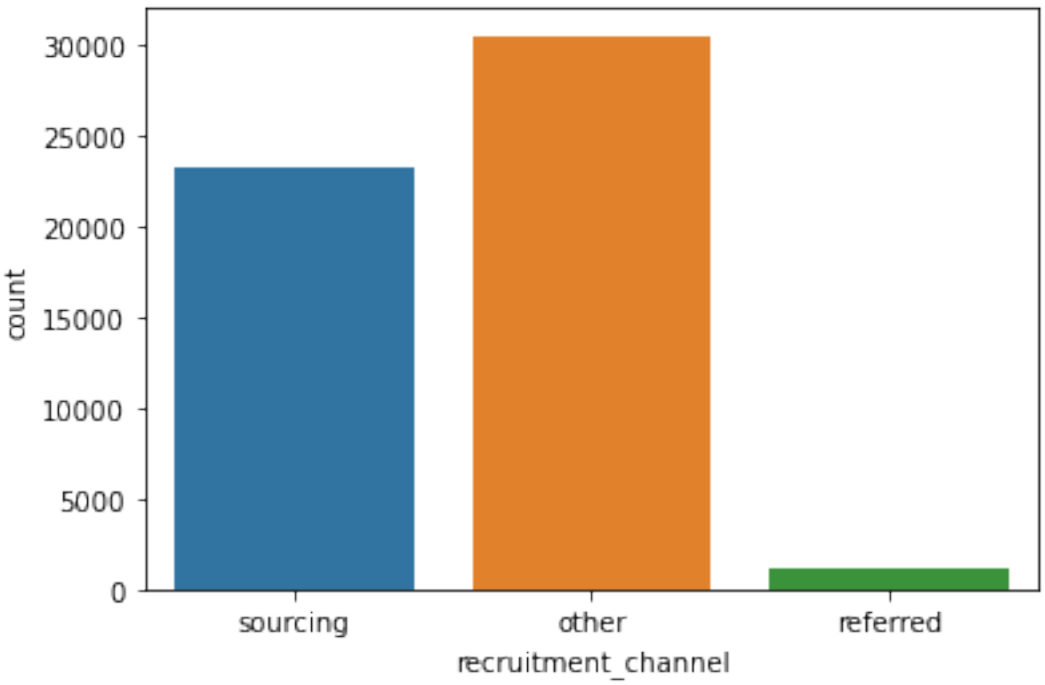
2.1.2 IS PROMOTED

[107]:
```
sb.countplot(x=df0["is_promoted"])
plt.show()
#Only less number of people are promoted in the company
```



2.1.3 RECRUITMENT CHANNEL

[108]:
```
sb.countplot(x=df0["recruitment_channel"])
plt.show()
#Some people are employed through sourcing and very few are referred , most of␣
 ↪the employees joined the company through other recruitment channels
```



2.1.4 DEPARTMENT

[109]:
```
df0['department'].value_counts()
```

[109]:
```
Sales & Marketing    16840
Operations           11348
Procurement           7138
Technology            7138
Analytics             5352
Finance               2536
HR                    2418
Legal                 1039
R&D                    999
Name: department, dtype: int64
```

[110]:
```
df0.department.describe()
```

[110]:
```
count                54808
unique                   9
top       Sales & Marketing
freq                 16840
Name: department, dtype: object
```

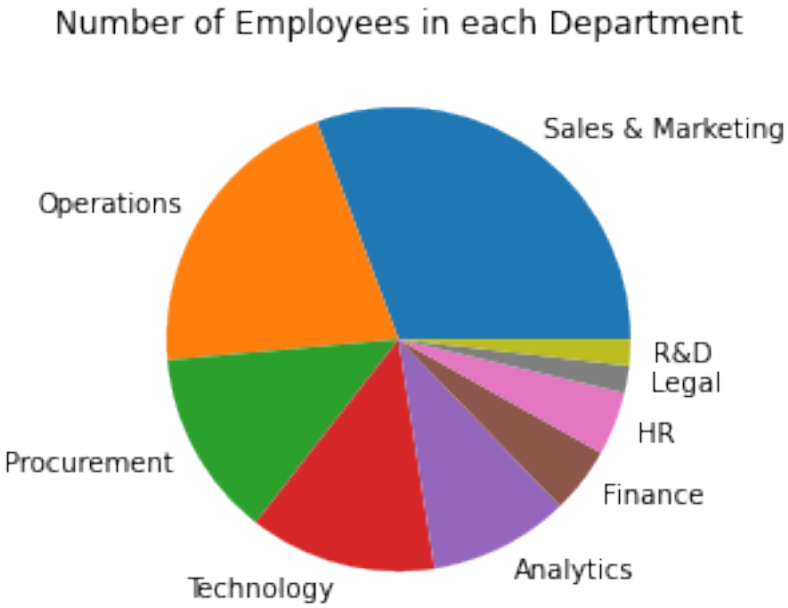
[111]:
```
name=df0.department.value_counts().index
```

[112]:
```
count=df0.department.value_counts().values
```

[ ]:
```

```

[113]:
```
plt.pie(count,labels=name)
plt.title('Number of Employees in each Department')
plt.show()
#The company gives more importance to Sales and Marketing and also covering all␣
 ↪Departments with adequate number of employees
```



2.1.5 NUMBER OF TRAININGS

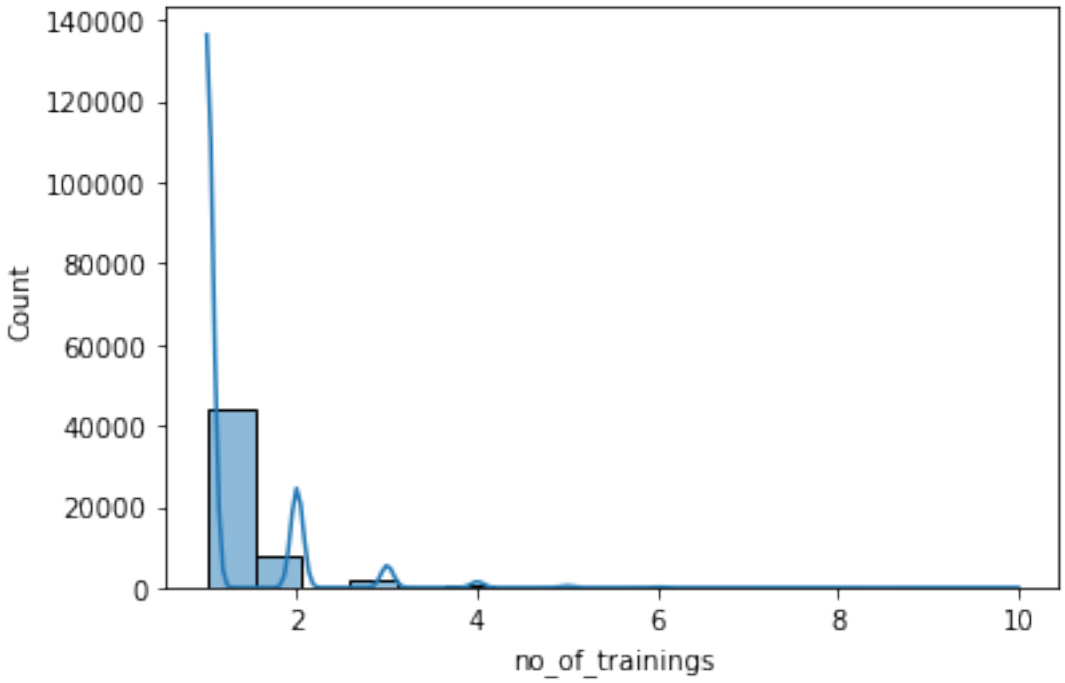
[114]:
```
df0.no_of_trainings.value_counts()
```

[114]:
```
1     44378
2      7987
3      1776
4       468
5       128
6        44
7        12
10        5
9         5
8         5
Name: no_of_trainings, dtype: int64
```

[115]:
```
df0.no_of_trainings.describe()
```

[115]:
```
count    54808.000000
mean         1.253011
std          0.609264
min          1.000000
25%          1.000000
50%          1.000000
75%          1.000000
max         10.000000
Name: no_of_trainings, dtype: float64
```

[116]:
```
sb.histplot(data = df0, x = "no_of_trainings", kde = True)
plt.show()
#Most of the employees have received less than 2 training sessions
```



2.1.6 AGE

[117]:
```
df0.age.value_counts()
```

[117]:
```
30    3665
31    3534
32    3534
29    3405
```

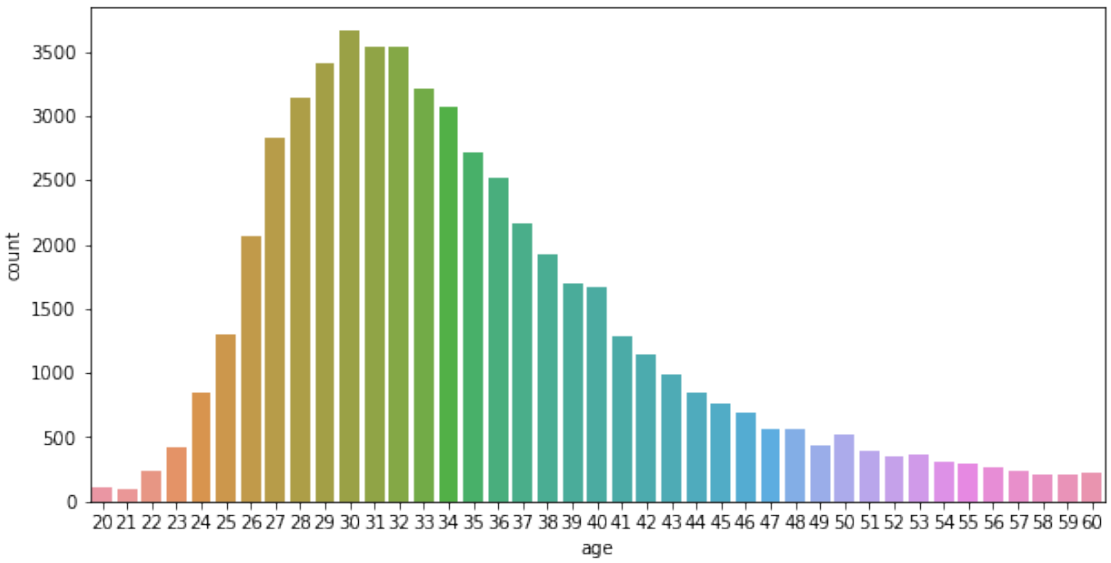
```
33    3210
28    3147
34    3076
27    2827
35    2711
36    2517
37    2165
26    2060
38    1923
39    1695
40    1663
25    1299
41    1289
42    1149
43     992
44     847
24     845
45     760
46     697
47     557
48     557
50     521
49     441
23     428
51     389
53     364
52     351
54     313
55     294
56     264
57     238
22     231
60     217
58     213
59     209
20     113
21      98
Name: age, dtype: int64
```

[118]:
```
df0.age.describe()
```

[118]:
```
count    54808.000000
mean        34.803915
std          7.660169
min         20.000000
25%         29.000000
50%         33.000000
```

```
75%          39.000000
max          60.000000
Name: age, dtype: float64
```

[119]:
```
plt.figure(figsize=(10,5))
sb.countplot(x=df0["age"])
plt.show()
# Most of the people working here are between the age of 25-35
```

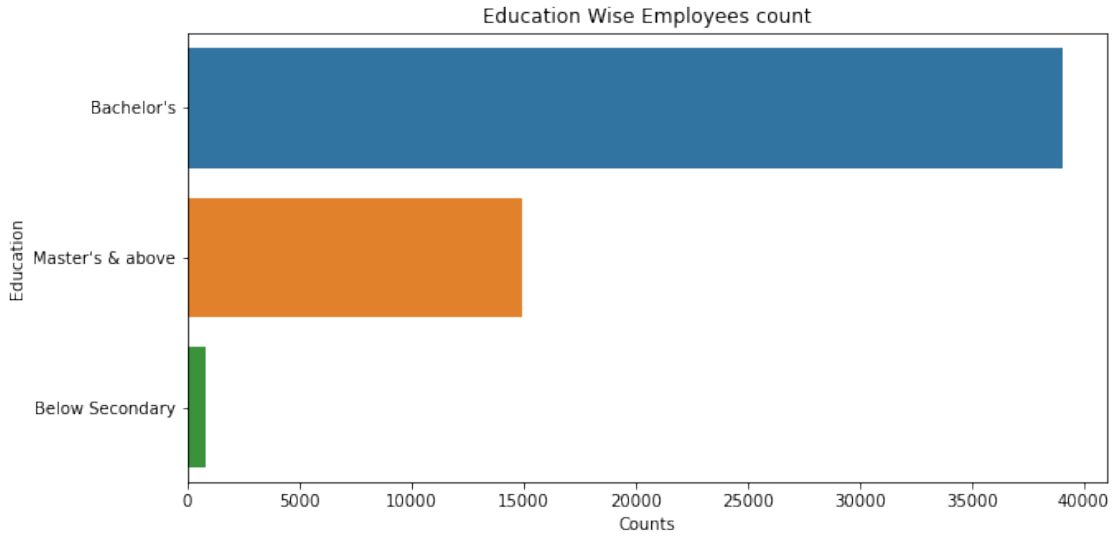


2.1.7 EDUCATION

[120]:
```
counts_education = df0["education"].value_counts()
counts_education = counts_education.reset_index()
counts_education.columns=["Education","Counts"]
counts_education
```

[120]:
```
           Education  Counts
0         Bachelor's   39078
1  Master's & above   14925
2   Below Secondary     805
```

[121]:
```
plt.figure(figsize=(10,5))
sb.barplot(y="Education",x="Counts",data = counts_education)
plt.title("Education Wise Employees count")
plt.show()
#Most of the employees here have joined the company after they had finished␣
 ↪their Bachelor's degree
```



[122]:
```
df0.head()
```

[122]:

	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length_of_service	awards_won?	avg_training_score	is_promoted
0	Sales & Marketing	region_7	Master's & above	f	sourcing	1	35	5.0	8	0	49	0
1	Operations	region_22	Bachelor's	m	other	1	30	5.0	4	0	60	0
2	Sales & Marketing	region_19	Bachelor's	m	sourcing	1	34	3.0	7	0	50	0
3	Sales & Marketing	region_23	Bachelor's	m	other	2	39	1.0	10	0	50	0
4	Technology	region_26	Bachelor's	m	other	1	45	3.0	2	0	73	0

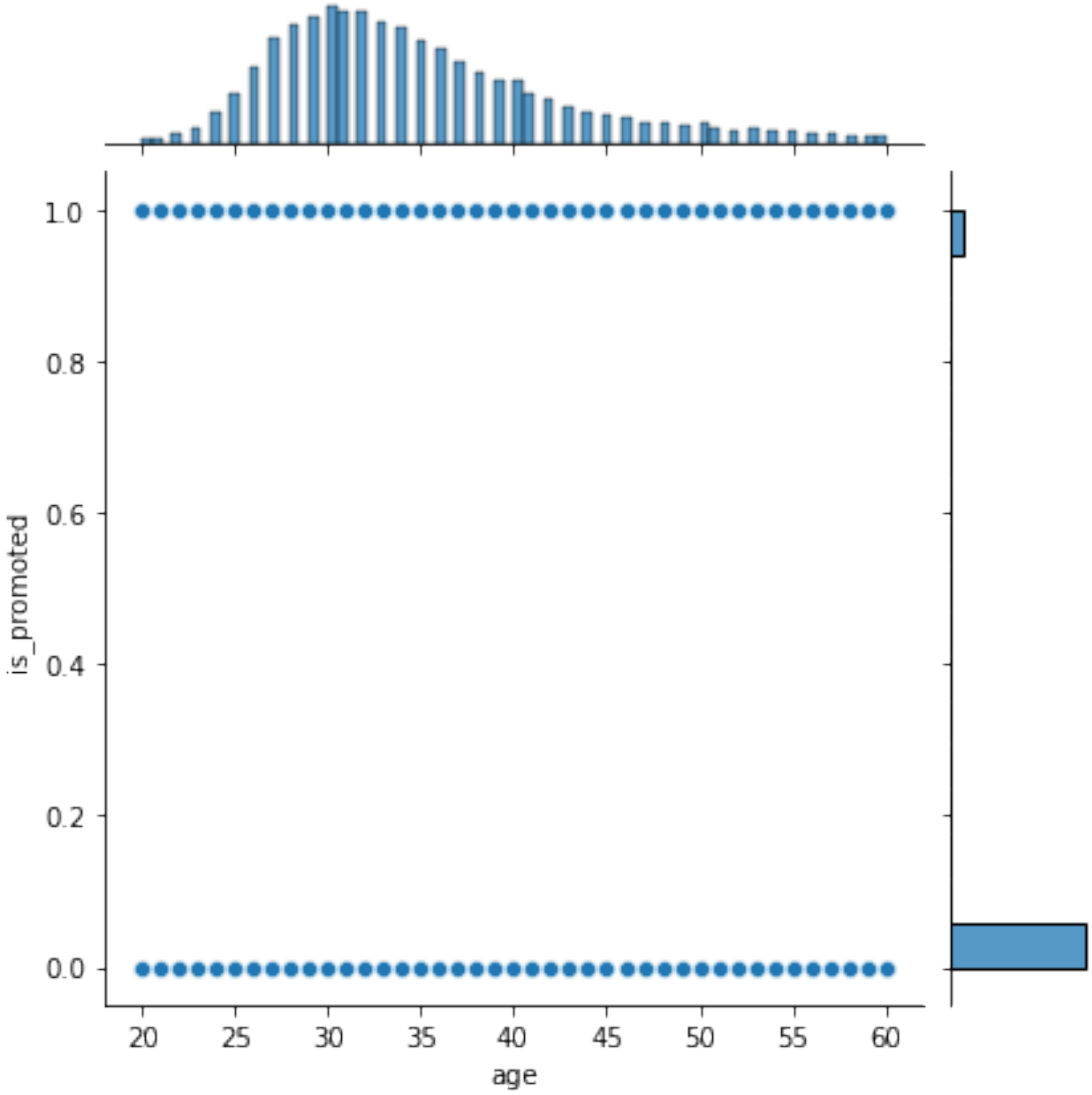
[ ]:

[ ]:

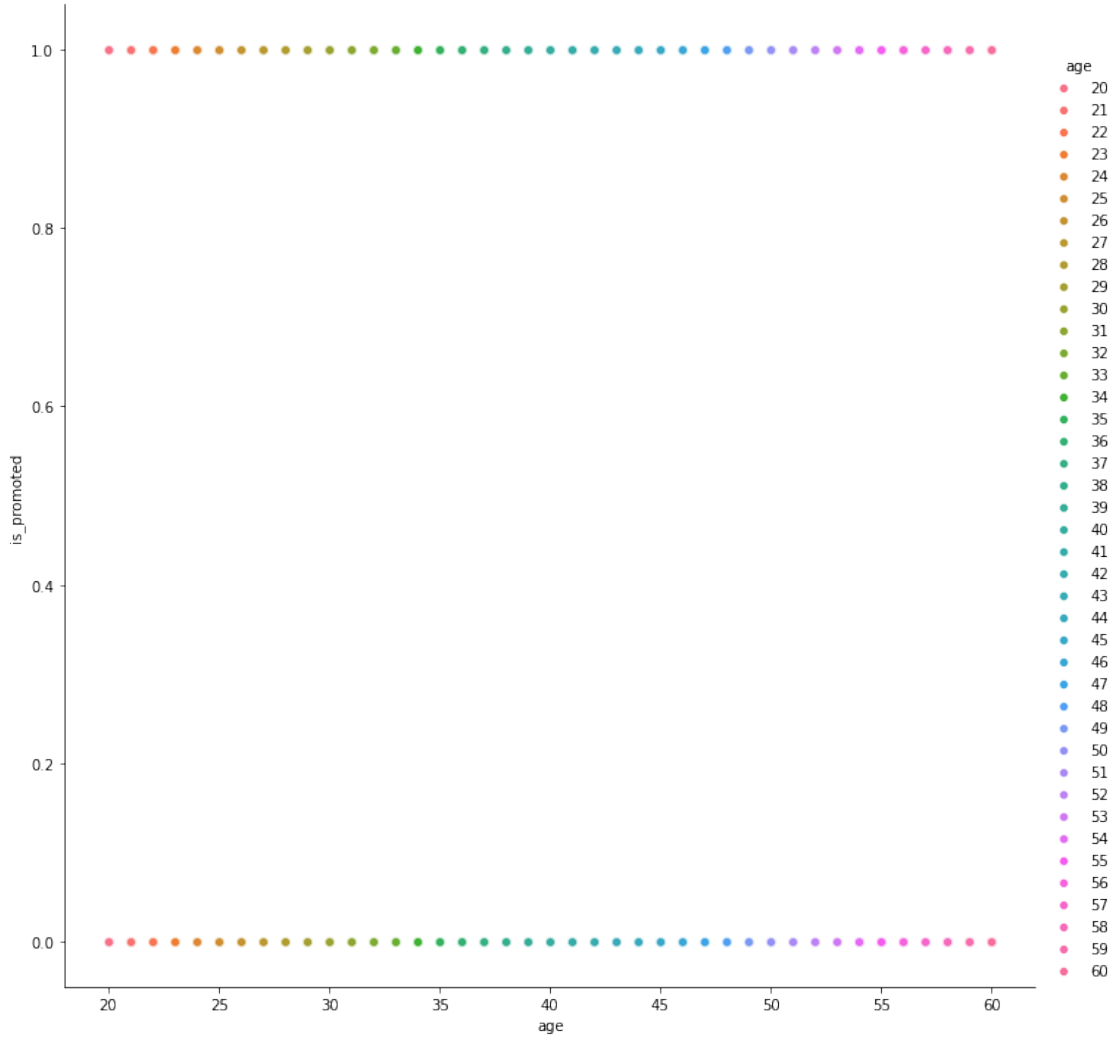
2.2 BIVARIATE

2.2.1 AGE VS IS_PROMOTED

[123]:
```
sb.jointplot(x=df0["age"],y=df0["is_promoted"])
plt.show()
#People from every age are promoted , but most of the employees promoted are␣
 ↪25-35 years old
```



[124]:
```
#For more colourful representation
sb.FacetGrid(df,hue='age',height=10).map(sb.
 ↪scatterplot,'age','is_promoted',edgecolor='w').add_legend()
plt.show()
```



2.2.2 GENDER VS IS_PROMOTED

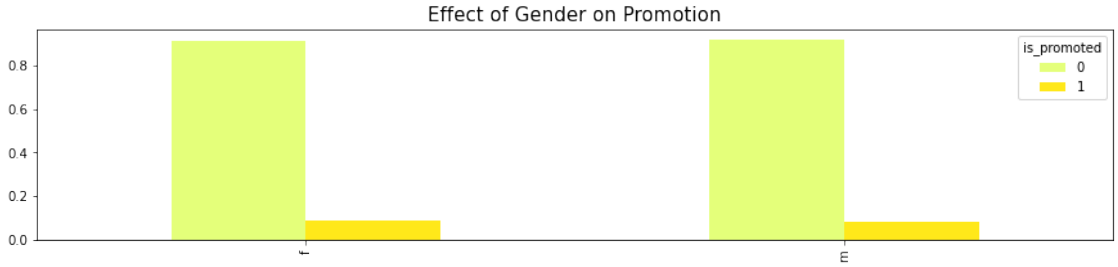
[125]:
```
pd.crosstab(df0['gender'], df0['is_promoted'])
```

[125]:
```
is_promoted      0     1
gender
f            14845  1467
m            35295  3201
```

[ ]:
```
```

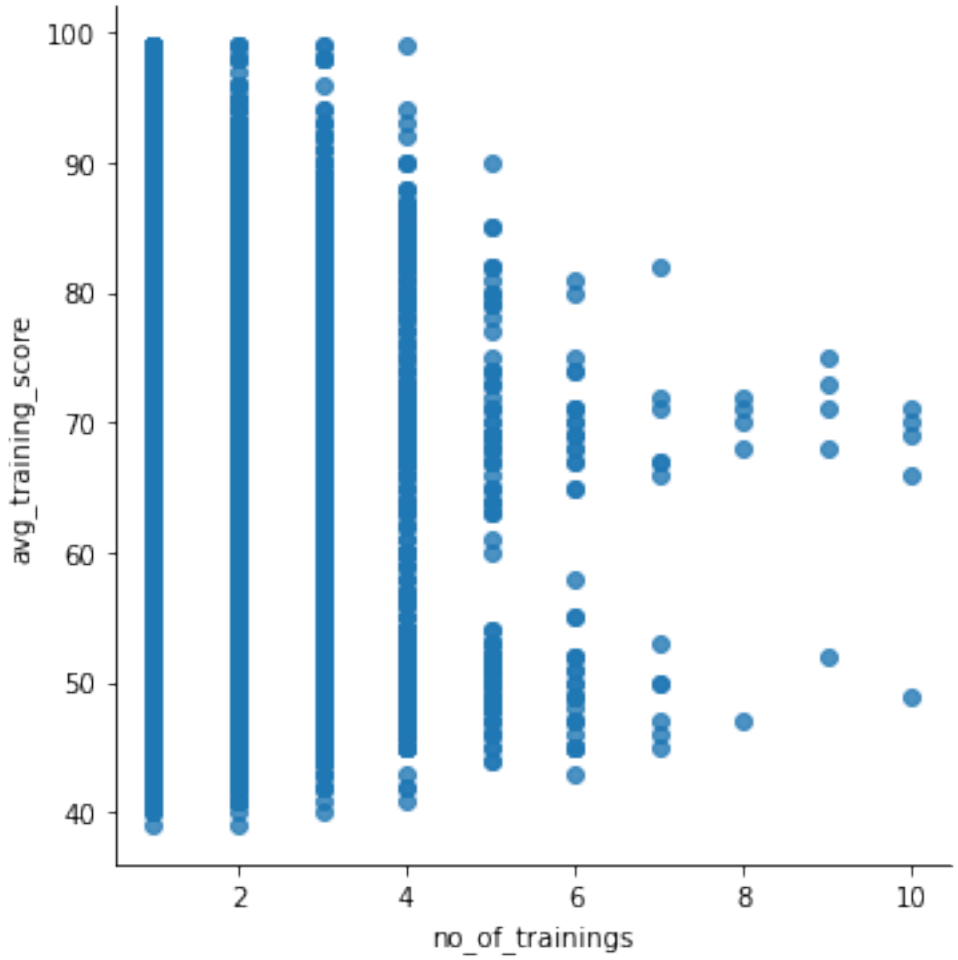
[126]:
```
plt.rcParams['figure.figsize'] = (15, 3)
x = pd.crosstab(df0['gender'], df0['is_promoted'])
color = plt.cm.Wistia(np.linspace(0, 1, 5))
```

```
x.div(x.sum(1).astype(float), axis = 0).plot(kind = 'bar', stacked = False,␣
 ↪color = color)
plt.title('Effect of Gender on Promotion', fontsize = 15)
plt.xlabel(' ')
plt.show()
#This shows that employees are not rejected promotion based on their gender,it␣
 ↪depends on their performance only(This shows that this company is a good␣
 ↪company)
```



2.2.3 NO OF TRAININGS VS AVERAGE SCORE

[127]:
```
#Scatterplot
sb.
 ↪lmplot(x="no_of_trainings",y="avg_training_score",fit_reg=False,data=df0,palette="BrBG")
plt.show()
#Those who have attended more than 8 training sessions most probably have an␣
 ↪average(or decent) score
#Those who have received less than that , can get any score from 0-100
```



[128]:
```
df0.head()
```

[128]:

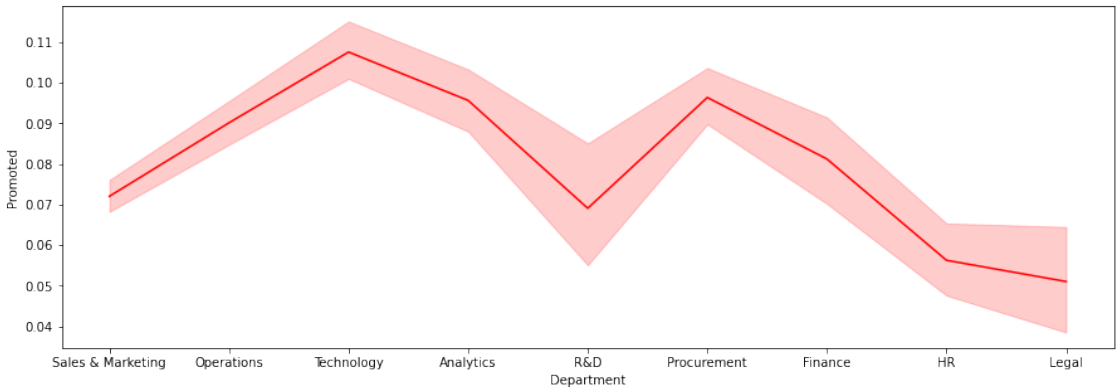
	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length_of_service	awards_won?	avg_training_score	is_promoted
0	Sales & Marketing	region_7	Master's & above	f	sourcing	1	35	5.0	8	0		
1	Operations	region_22	Bachelor's	m	other	1	30	5.0	4	0		
2	Sales & Marketing	region_19	Bachelor's	m	sourcing	1	34	3.0	7	0		
3	Sales & Marketing	region_23	Bachelor's	m	other	2	39	1.0	10	0		
4	Technology	region_26	Bachelor's	m	other	1	45	3.0	2	0		

0	49	0
1	60	0
2	50	0
3	50	0
4	73	0

[129]:
```
#awards_won? vs length_of_service
```

2.2.4 DEPARTMENT VS IS_PROMOTED

[130]:
```
plt.figure(figsize=(15,5))
sb.lineplot(x=df0['department'],y=df0['is_promoted'],color='red')
plt.xlabel("Department")
plt.ylabel("Promoted")
plt.show()
#Most of the promoted employees are from the Technology Department(The company␣
 ↪gives more important to Technology)
#The employees in the Legal Department are promoted less often than the␣
 ↪employees in the other departments
```



[131]:
```
df0.head()
```

[131]:

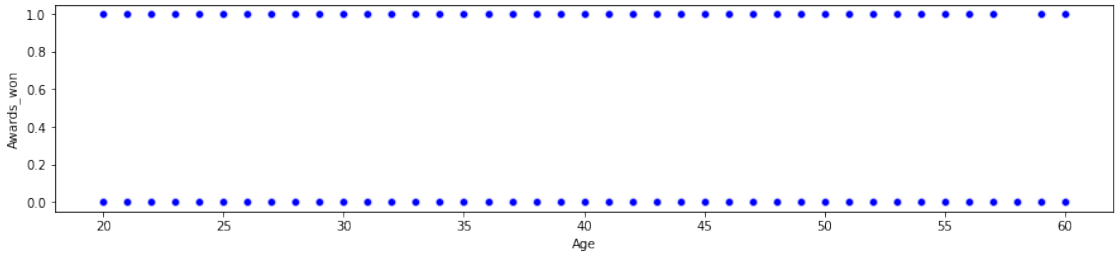
	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length_of_service	awards_won?
0	Sales & Marketing	region_7	Master's & above	f	sourcing	1	35	5.0	8	0
1	Operations	region_22	Bachelor's	m	other	1	30	5.0	4	0
2	Sales & Marketing	region_19	Bachelor's	m	sourcing					
3	Sales & Marketing	region_23	Bachelor's	m	other					
4	Technology	region_26	Bachelor's	m	other					

2	1	34	3.0	7	0
3	2	39	1.0	10	0
4	1	45	3.0	2	0

	avg_training_score	is_promoted
0	49	0
1	60	0
2	50	0
3	50	0
4	73	0

2.2.5 AGE VS AWARDS WON

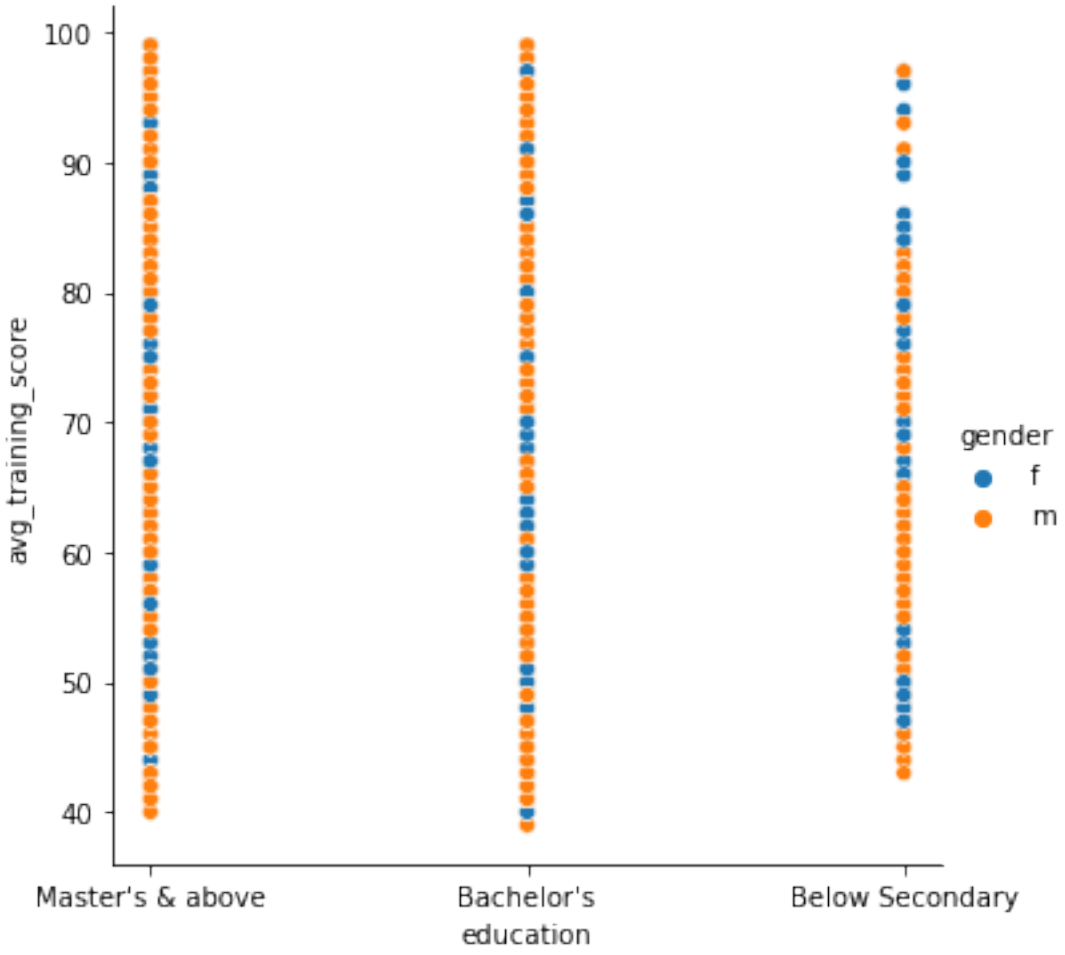
[132]:
```
sb.scatterplot(x=df0['age'],y=df0['awards_won?'],color="blue")
plt.xlabel("Age")
plt.ylabel("Awards_won")
plt.show()
#This shows that the Awards are not given based on age.
#Or Age is a least influencing factor to determine if the employee should get␣
 ↪an Award
```



2.2.6 EDUCATION VS AVERAGE TRAINING SCORE

[ ]:
```
```

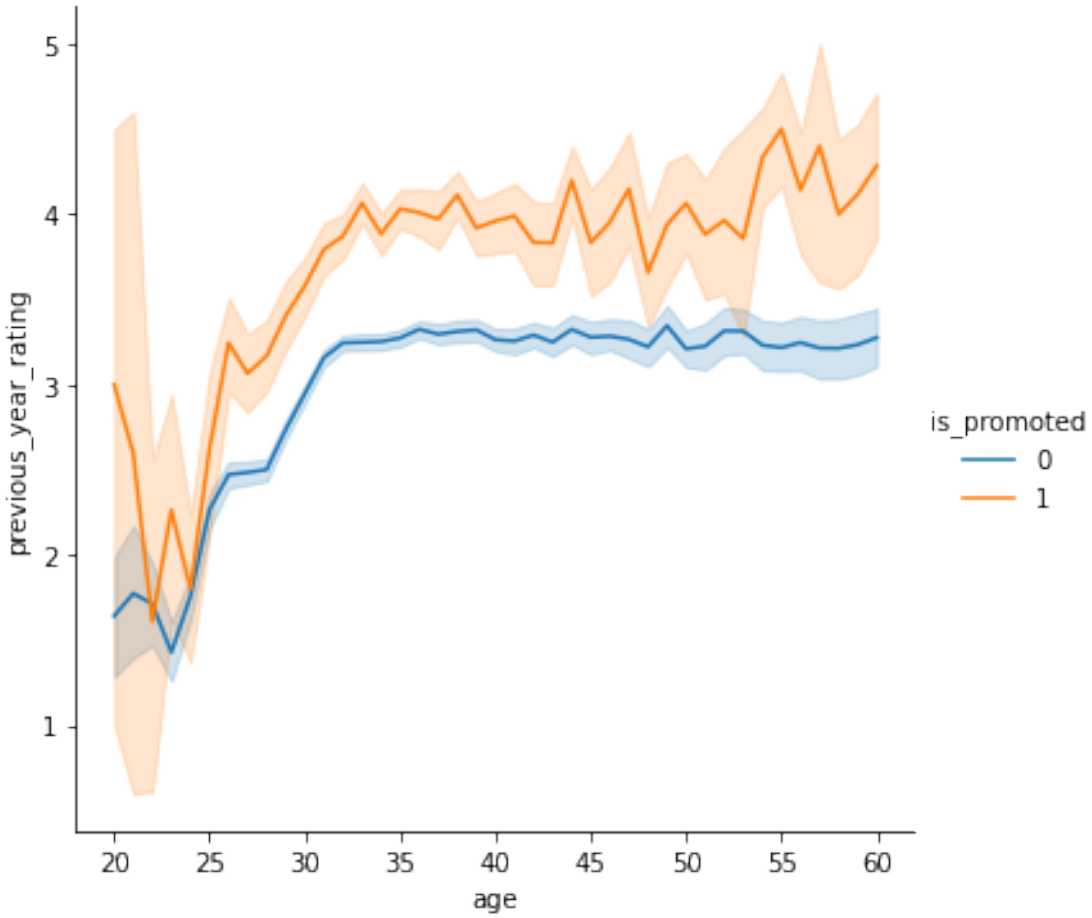
[133]:
```
sb.relplot(x="education",y="avg_training_score",hue='gender',data=df0)
plt.show()
#This shows that employees having education level as below secondary , dont get␣
 ↪a very less or very high training score(They might be in departments where␣
 ↪education level is not a factor to influence their performance)
#But generally the education might not determine their training score(Employees␣
 ↪might be assigned departments based on their education level)
```



2.2.7 PREVIOUS YEAR RATING VS AGE (Based on IS_PROMOTED)

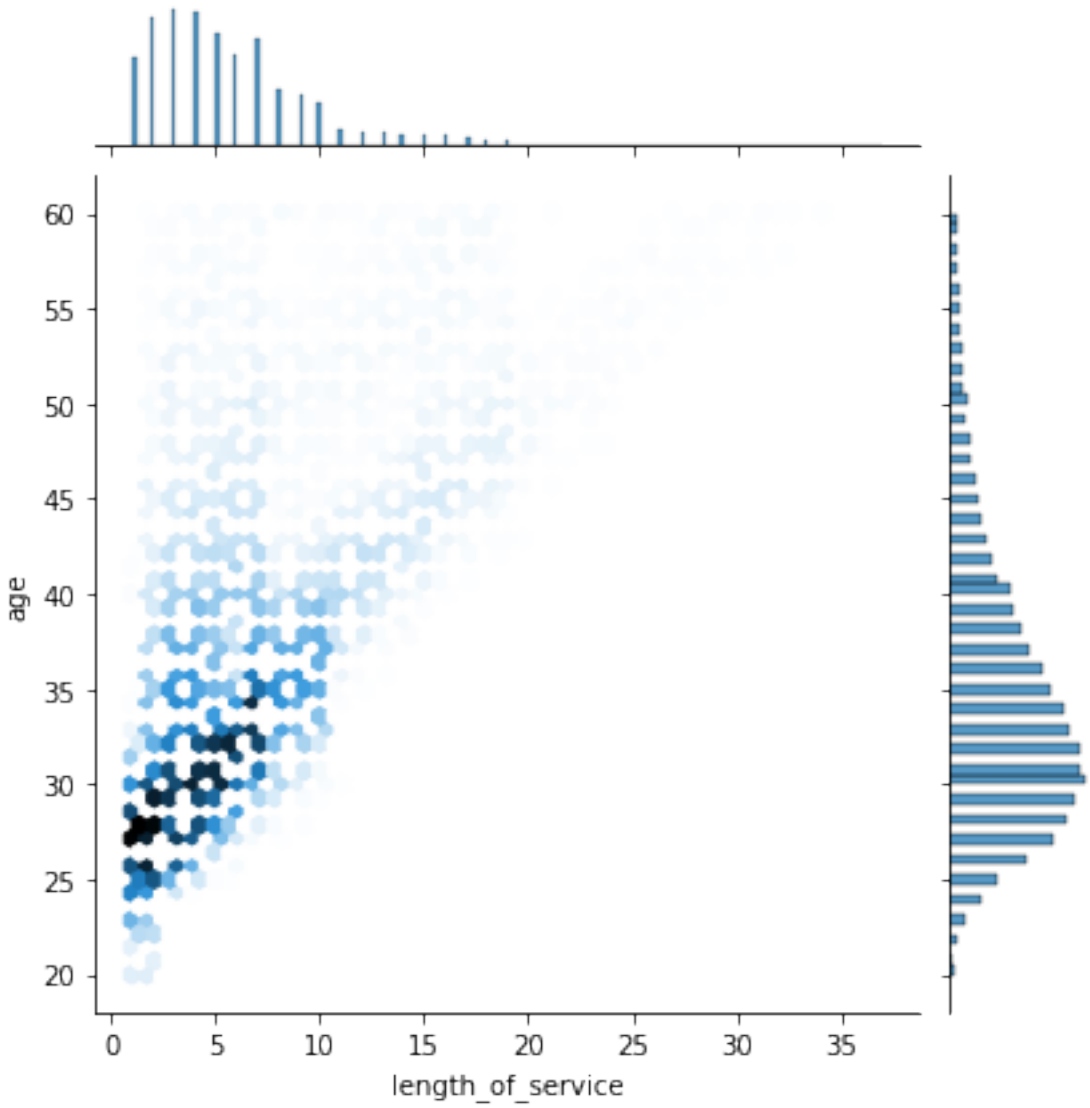
[134]:
```
plt.figure(figsize=(15,8))
sb.
 ↪relplot(x="age",y="previous_year_rating",hue="is_promoted",kind="line",data=df0)
plt.show()
#Generally employees with a higher previous year rating are promoted more often␣
 ↪than people who have lesser rating(with some exceptions)
```

```
<Figure size 1080x576 with 0 Axes>
```



2.2.8 LENGTH OF SERVICE VS AGE

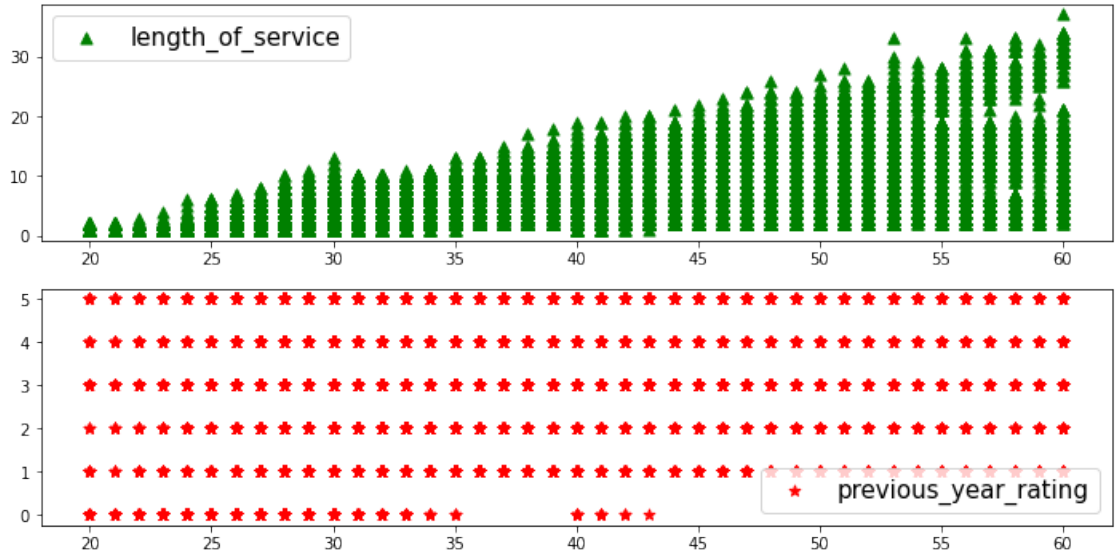
[135]:
```
sb.jointplot(x="length_of_service", y="age",kind='hex', data=df0)
plt.show()
#Most of the freshers here belong to the age group of 25-35
#Only few employees who are around 50 years are working here(They may be people␣
 ↪with higher length of service)
```



2.2.9 LENGTH OF SERVICE AND PREVIOUS YEAR RATING VS AGE

[136]:
```
fig = plt.figure(figsize = (12,6))
fig1 = fig.add_subplot(211) # 2 rows, 1 column, 1st figure(subplot)
fig2 = fig.add_subplot(212) # 2nd figure(subplot) in the same figure(fig)
fig1.scatter(df0['age'],df0['length_of_service'],color = "g",marker = "^",s =␣
↪50,label = 'length_of_service')
fig2.scatter(df0['age'],df0['previous_year_rating'],color = "r",marker = "*",s␣
↪= 50,label = 'previous_year_rating')
fig1.legend(fontsize = 15)
fig2.legend(fontsize = 15)
plt.show()
```

```
#Employees who have many years of experience dont get bad ratings generally
```



3 MULTIVARIATE

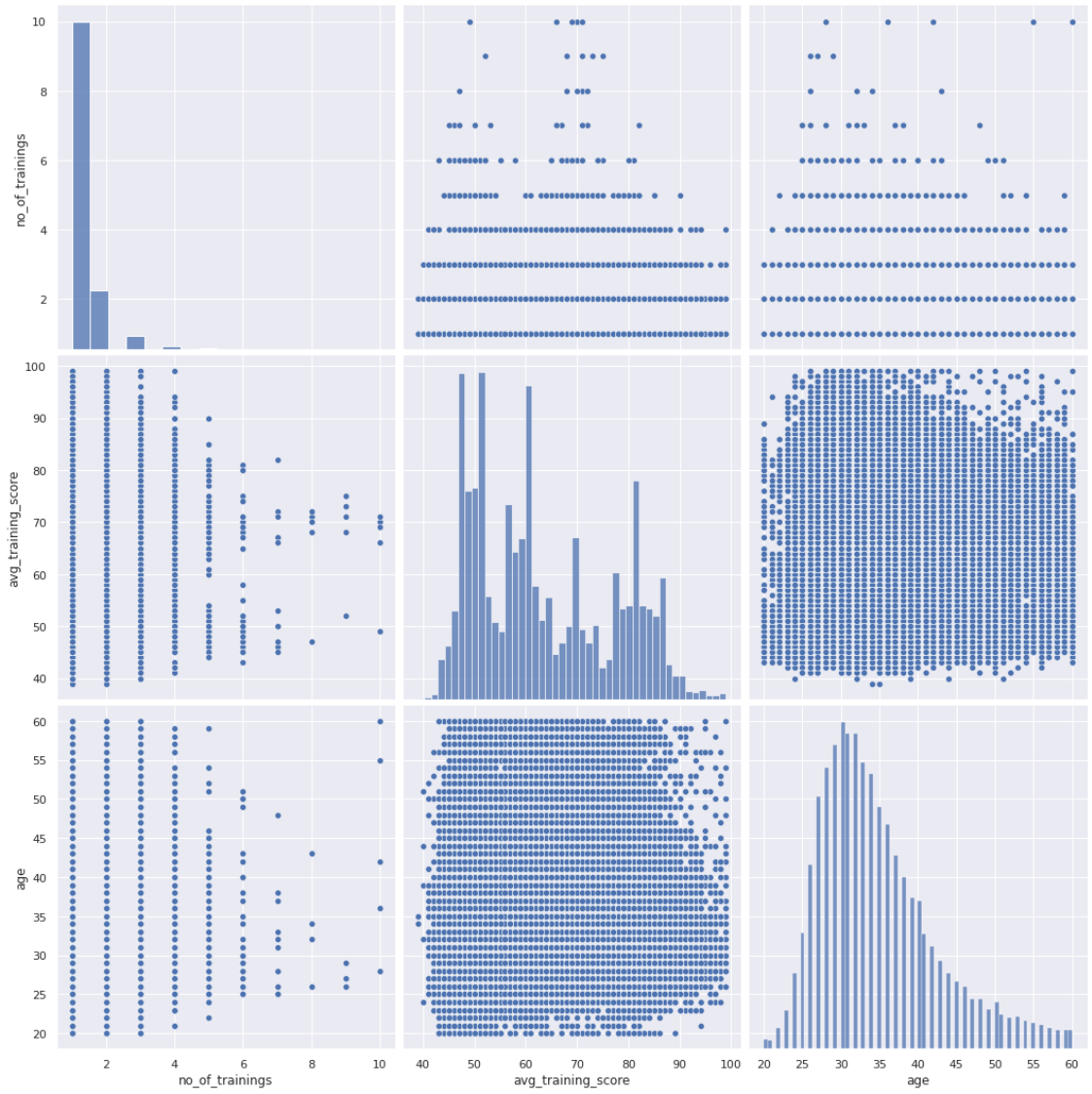
3.0.1 NUMBER OF TRAININGS VS AVG TRAINING SCORE VS AGE

[137]:
```
df[['no_of_trainings','avg_training_score','age']].corr()
```

[137]:

	no_of_trainings	avg_training_score	age
no_of_trainings	1.000000	0.042517	-0.081278
avg_training_score	0.042517	1.000000	-0.048380
age	-0.081278	-0.048380	1.000000

[138]:
```
sb.set()
col = ['no_of_trainings','avg_training_score','age']
sb.pairplot(df0[col],height = 5)
plt.savefig("pairplot.png")
plt.show()
#The highest correlation between these 9 pairs is for the number of trainings␣
 ↪and age pair
#The age influences the no of trainings an employee has attended(It maybe␣
 ↪positive or negative)
#Either an employee who is older has got many number of training sessions or␣
 ↪the experienced people might have attended less number of training␣
 ↪sessions(They might be employed based on their experience in another company)
```



[ ]:

[ ]:

[139]:
```
result = pd.pivot_table(data=df0,␣
↪index='education',columns='no_of_trainings',values='avg_training_score')
result
#This shows that number of trainings and education does not affect their␣
↪training score , generally
```

[139]:
```
no_of_trainings          1          2          3          4          5   \
education
Bachelor's        62.653576  65.149719  64.696856  63.850136  59.693069
```

```
Below Secondary   64.111455  67.416667  68.051282  76.166667        NaN
Master's & above  63.764411  65.800797  64.713217  64.089888  61.518519

no_of_trainings          6          7     8     9          10
education
Bachelor's        58.108108  56.777778  62.0  67.8  68.666667
Below Secondary         NaN        NaN   NaN   NaN        NaN
Master's & above  59.857143  68.333333  71.0   NaN  59.500000
```

3.0.2 CORRELATION BETWEEN ANY TWO FEATURES

[140]:
```
df0.corr()
#We can find the correlation between any two features
```

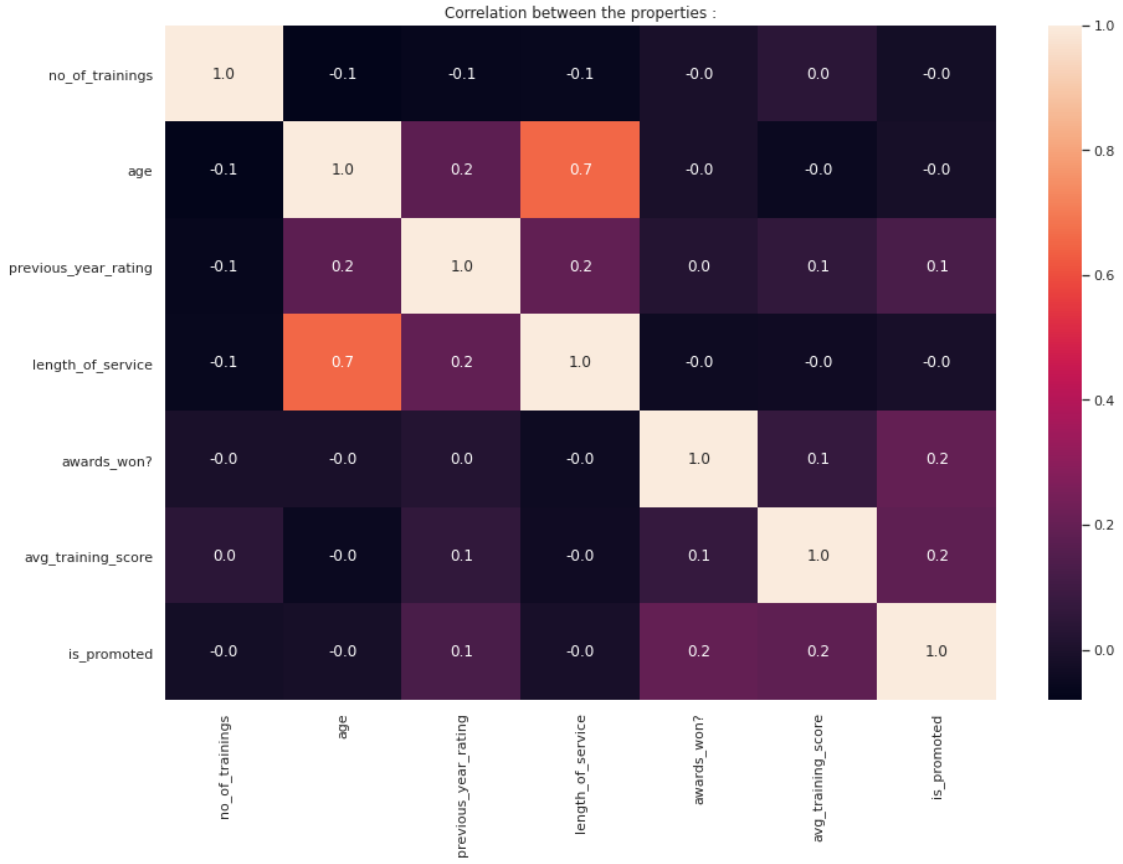
[140]:
```
                     no_of_trainings       age  previous_year_rating  \
no_of_trainings             1.000000 -0.081278             -0.064119
age                        -0.081278  1.000000              0.177935
previous_year_rating       -0.064119  0.177935              1.000000
length_of_service          -0.057275  0.657111              0.191163
awards_won?                -0.007628 -0.008169              0.021475
avg_training_score          0.042517 -0.048380              0.058718
is_promoted                -0.024896 -0.017166              0.125991

                     length_of_service  awards_won?  avg_training_score  \
no_of_trainings              -0.057275    -0.007628            0.042517
age                           0.657111    -0.008169           -0.048380
previous_year_rating          0.191163     0.021475            0.058718
length_of_service             1.000000    -0.039927           -0.038122
awards_won?                  -0.039927     1.000000            0.072138
avg_training_score           -0.038122     0.072138            1.000000
is_promoted                  -0.010670     0.195871            0.181147

                     is_promoted
no_of_trainings        -0.024896
age                    -0.017166
previous_year_rating    0.125991
length_of_service      -0.010670
awards_won?             0.195871
avg_training_score      0.181147
is_promoted             1.000000
```

[141]:
```
plt.figure(figsize = (15, 10))
sb.heatmap(df0.corr(), annot = True, fmt = '.1f')
plt.title("Correlation between the properties :")
plt.savefig("heatmap.png")
plt.show()
#Age and length of service have the highest correlation
```

```
#It can mean any of the two things:
#i)Most have the employees spend most of their career here
#ii)Most of the employees might be freshers
```



Thus the data set is analysed and the prediction is done with 90% accuracy

[ ]: