In [1]:

```
import pandas as pd
import numpy as np
import sklearn
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
df=pd.read_csv('train (1).csv')
df.head()
```

Out[2]:

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings
0	65438	Sales & Marketing	region_7	Master's & above	f	sourcing	1
1	65141	Operations	region_22	Bachelor's	m	other	1
2	7513	Sales & Marketing	region_19	Bachelor's	m	sourcing	1
3	2542	Sales & Marketing	region_23	Bachelor's	m	other	2
4	48945	Technology	region_26	Bachelor's	m	other	1

In [3]:

```
df.shape
```

Out[3]:

```
(54808, 14)
```

In [4]:

```
1 df.info()
```

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

In [5]:

```
1 df.describe()
```

Out[5]:

	employee_id	no_of_trainings	age	previous_year_rating	length_of_service	
count	54808.000000	54808.000000	54808.000000	50684.000000	54808.000000	548
mean	39195.830627	1.253011	34.803915	3.329256	5.865512	
std	22586.581449	0.609264	7.660169	1.259993	4.265094	
min	1.000000	1.000000	20.000000	1.000000	1.000000	
25%	19669.750000	1.000000	29.000000	3.000000	3.000000	
50%	39225.500000	1.000000	33.000000	3.000000	5.000000	
75%	58730.500000	1.000000	39.000000	4.000000	7.000000	
max	78298.000000	10.000000	60.000000	5.000000	37.000000	

In [6]:

```
1 df.isnull().sum()
```

Out[6]:

```
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 [7]:

```
1 df['previous_year_rating']=df['previous_year_rating'].fillna(df['previous_year_rati
2 df['education']=df['education'].fillna(df['education'].mode()[0])
```

In [8]:

```
1 df.isnull().sum()
```

Out[8]:

```
employee_id             0
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
KPIs_met >80%           0
awards_won?             0
avg_training_score      0
is_promoted             0
dtype: int64
```

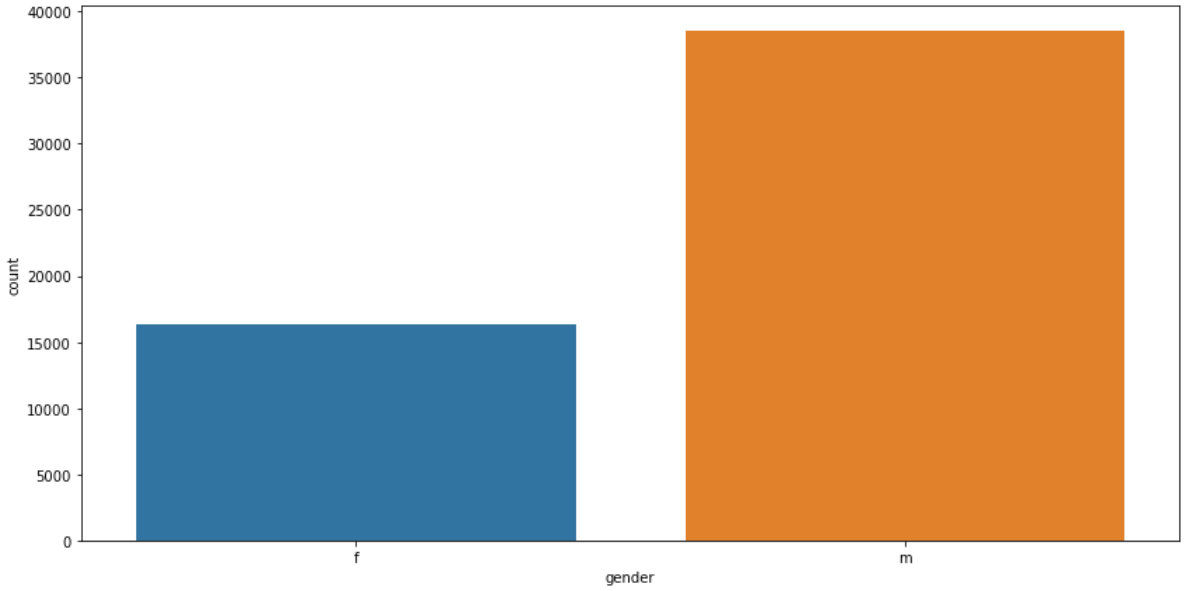
In [9]:

```
1 df.info()
```

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

In [10]:

```
1 plt.figure(figsize=(14,7))
2 sns.countplot(x=df['gender'])
3 plt.show()
```



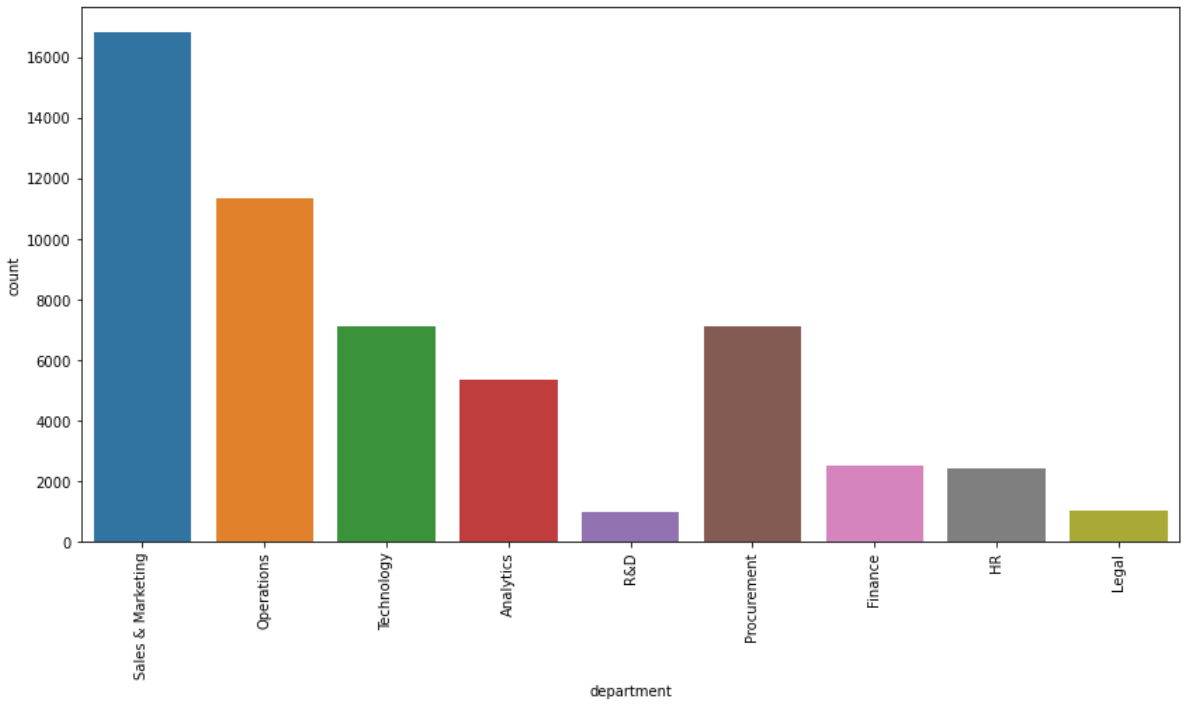
In [11]:

```
#percentage of male and females
males=df.gender.value_counts()['m']/len(df)
print("percentage of male :",round(males*100,2))

females=df.gender.value_counts()['f']/len(df)
print("percentage of female :",round(females*100,2))
```

```
percentage of male : 70.24
percentage of female : 29.76
```

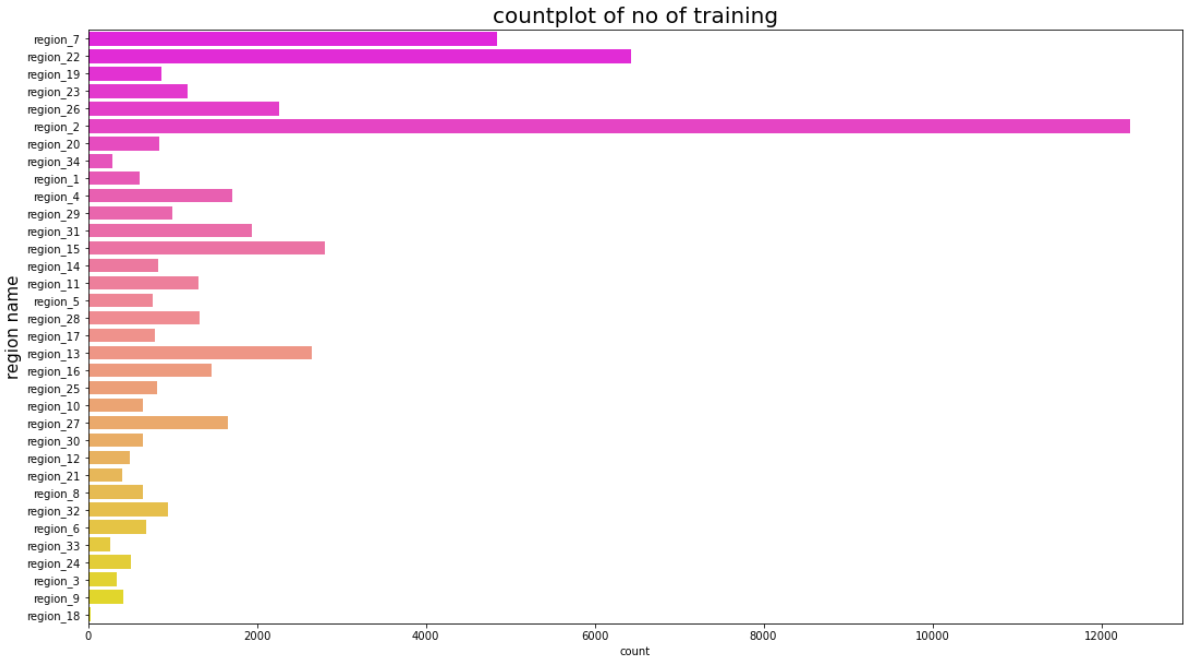
In [12]:

```
#department wise distribution
plt.figure(figsize=(14,7))
sns.countplot(x=df['department'])
plt.xticks(rotation=90)
plt.show()
```



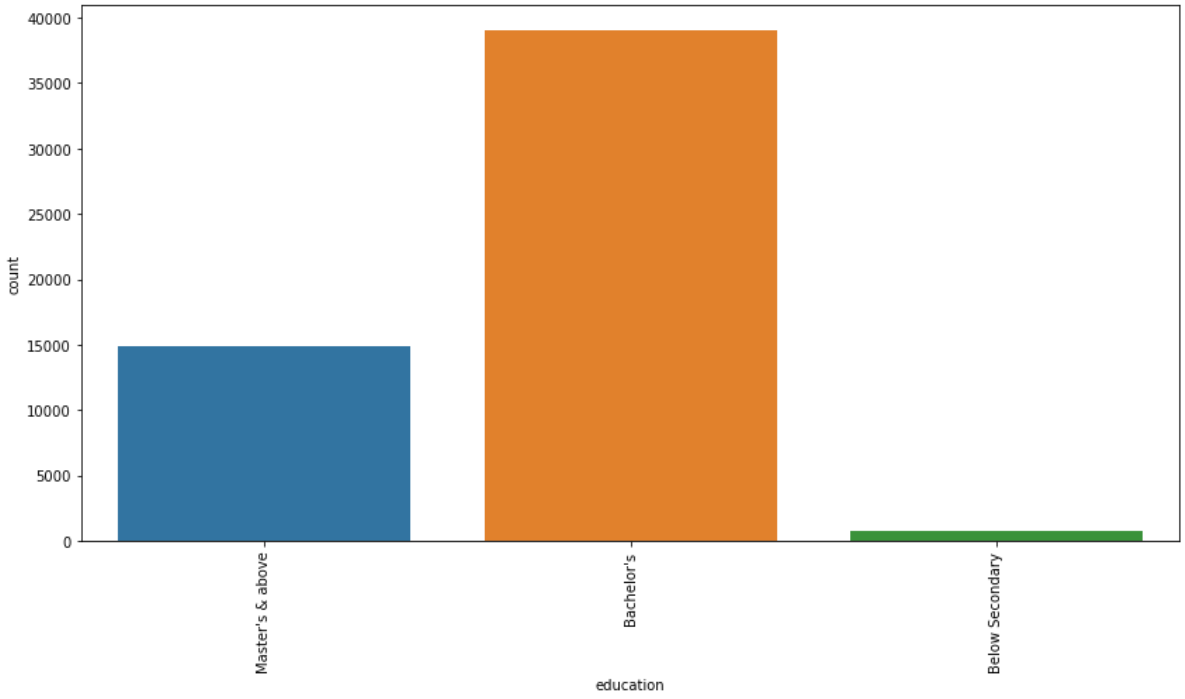
In [13]:

```
plt.rcParams['figure.figsize']=(18,10)

sns.countplot(y=df['region'],palette='spring',orient='v')
#y= .... and orient = v means count plot(bins) as vertically

plt.ylabel('region name',fontsize='15')

plt.title('countplot of no of training',fontsize=20)

plt.show()
```



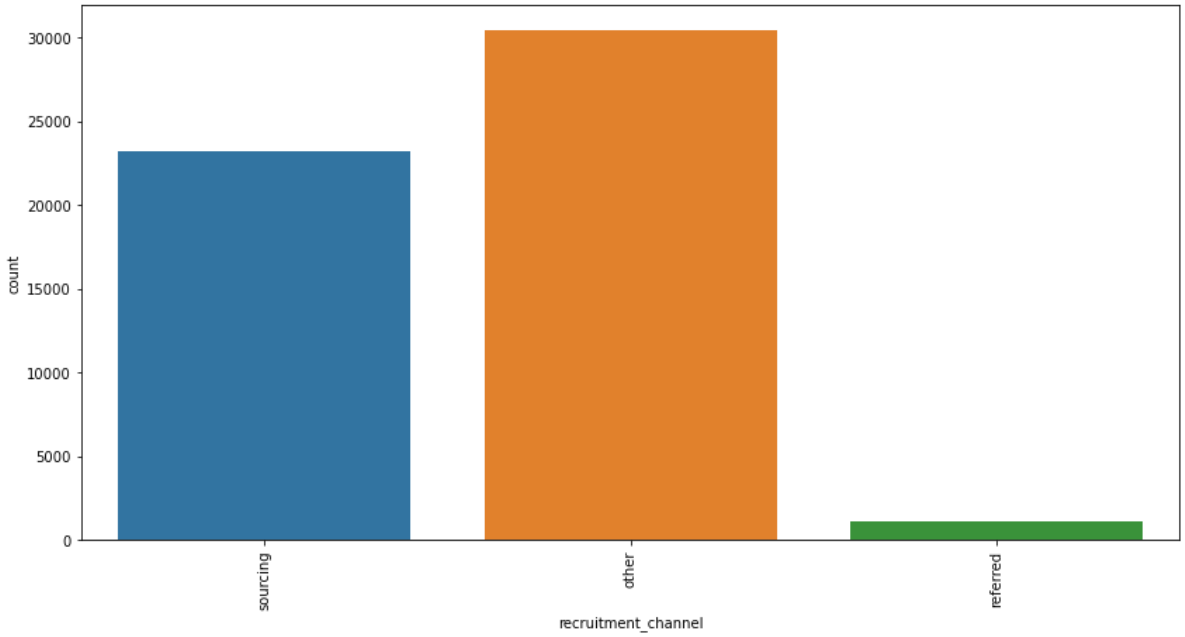
In [14]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['education'])
plt.xticks(rotation=90)
plt.show()
```



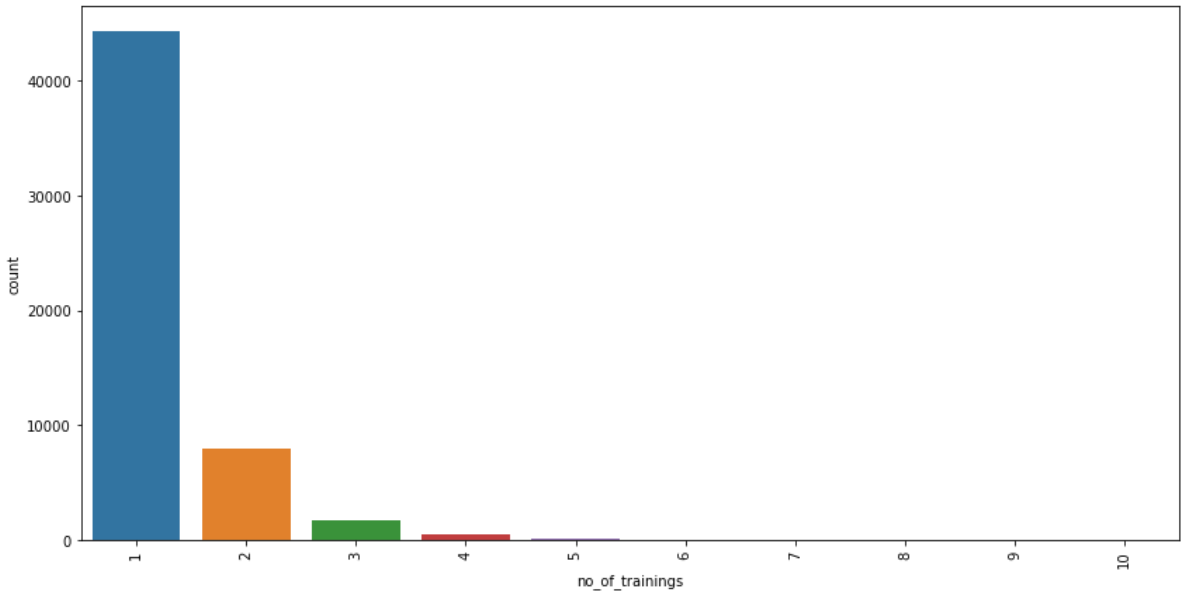
In [15]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['recruitment_channel'])
plt.xticks(rotation=90)
plt.show()
```



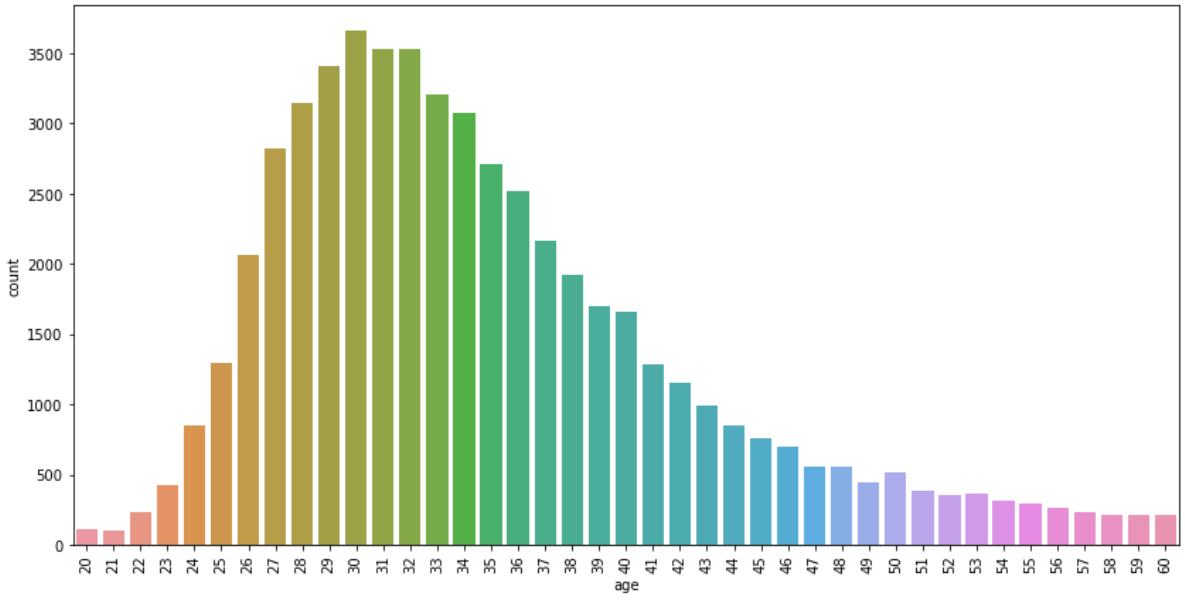
In [16]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['no_of_trainings'])
plt.xticks(rotation=90)
plt.show()
```



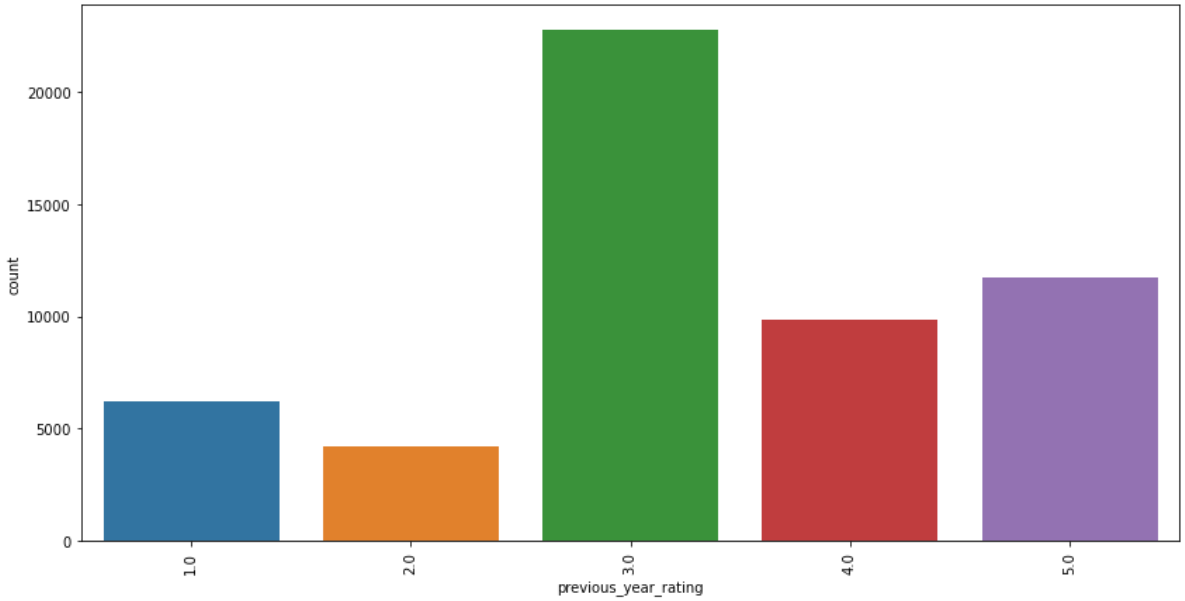
In [17]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['age'])
plt.xticks(rotation=90)
plt.show()
```



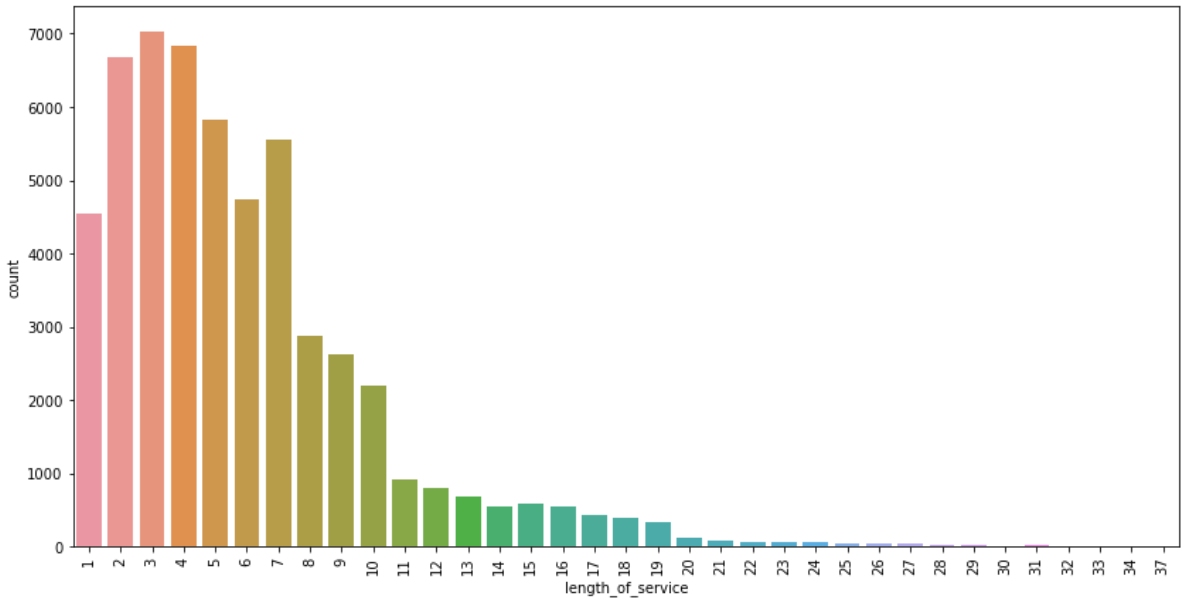
In [18]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['previous_year_rating'])
plt.xticks(rotation=90)
plt.show()
```



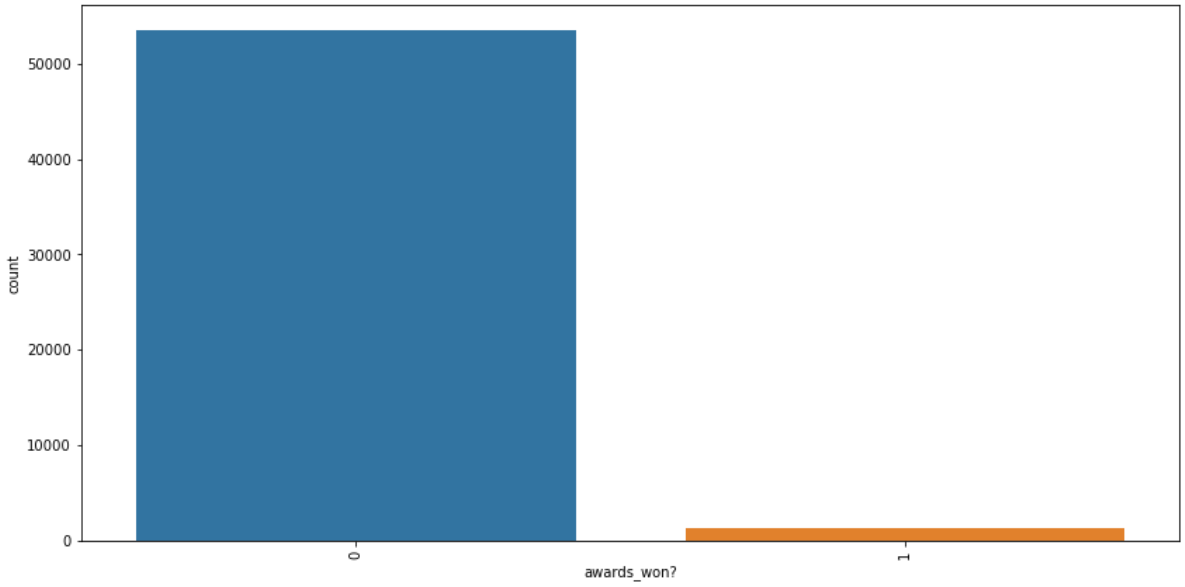
In [19]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['length_of_service'])
plt.xticks(rotation=90)
plt.show()
```



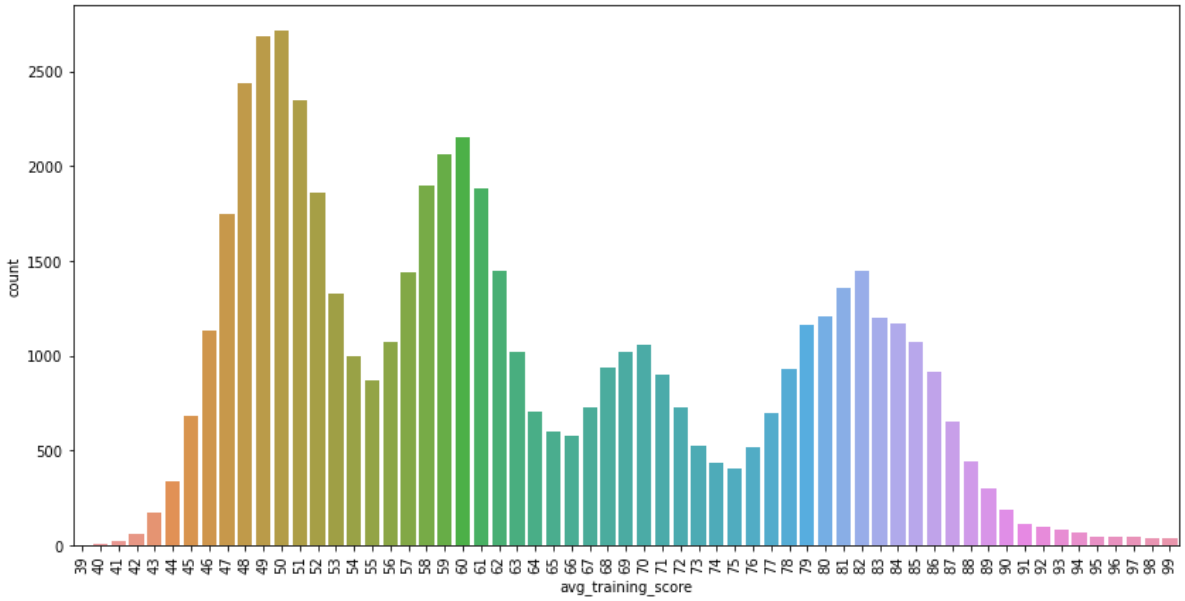
In [20]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['awards_won?'])
plt.xticks(rotation=90)
plt.show()
```



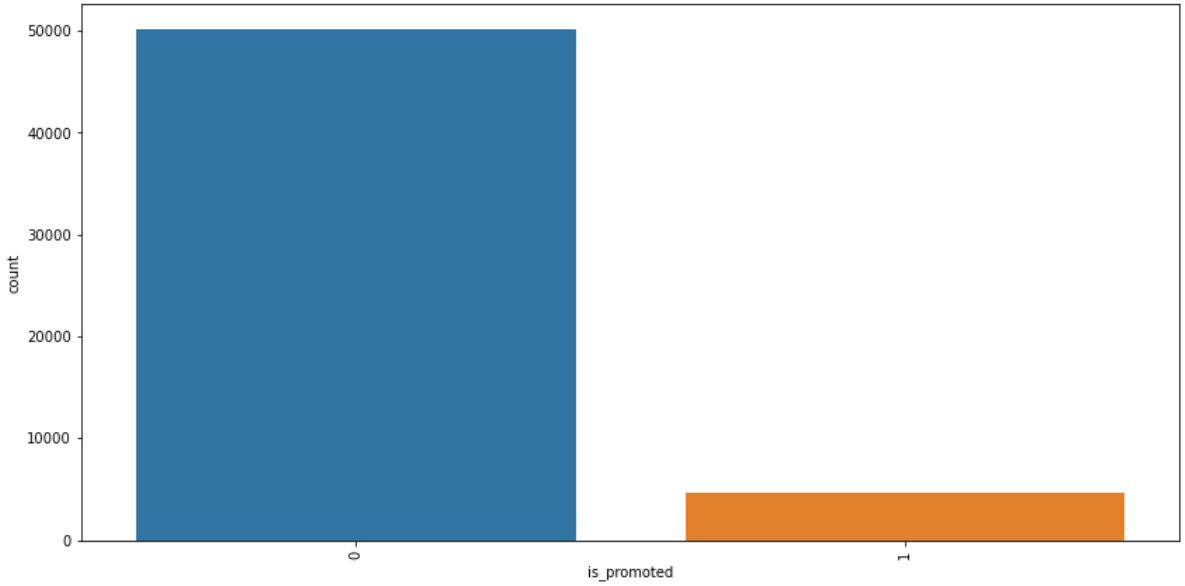
In [21]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['avg_training_score'])
plt.xticks(rotation=90)
plt.show()
```



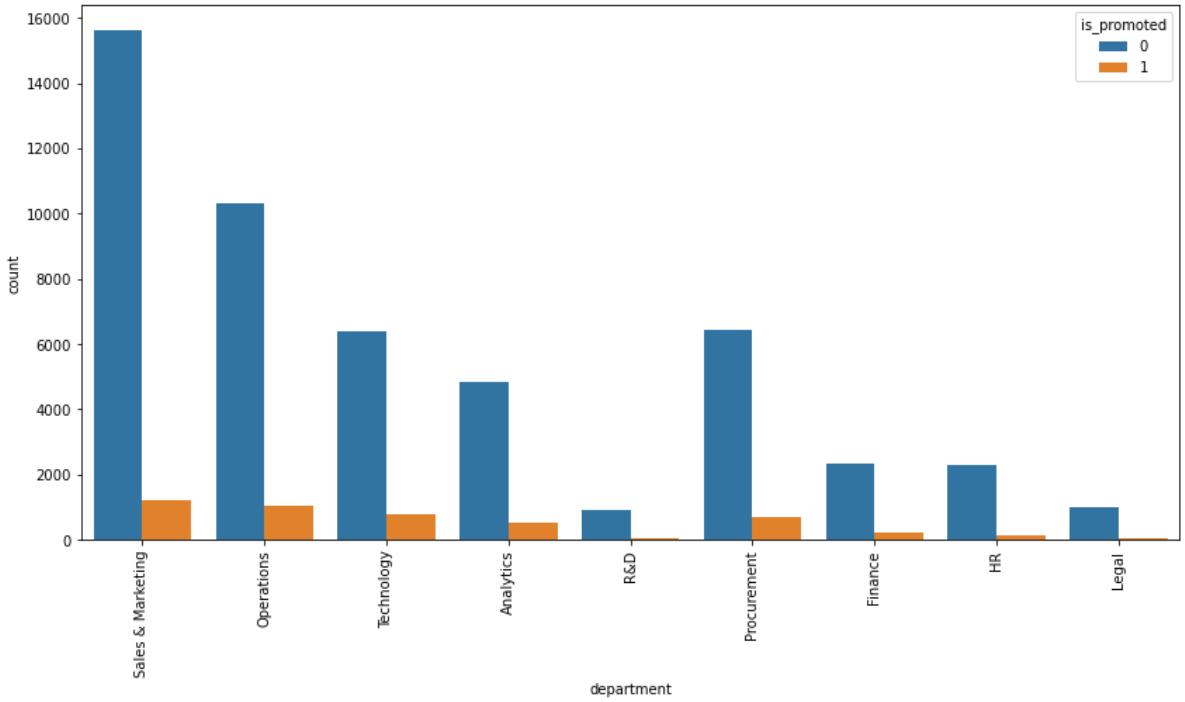
In [22]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



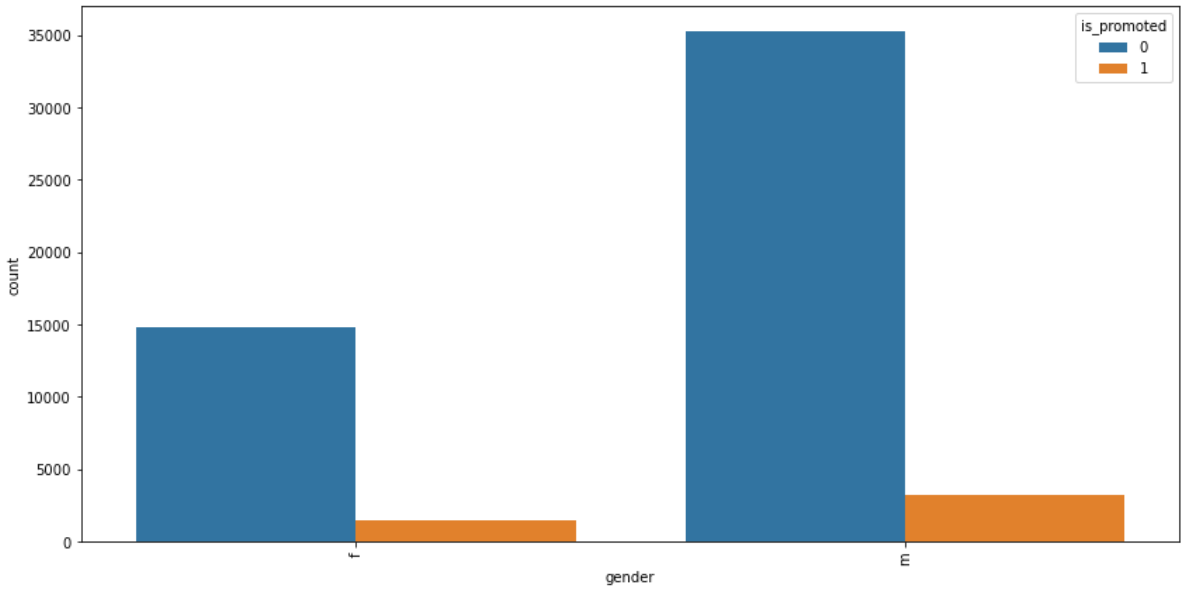
In [23]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['department'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



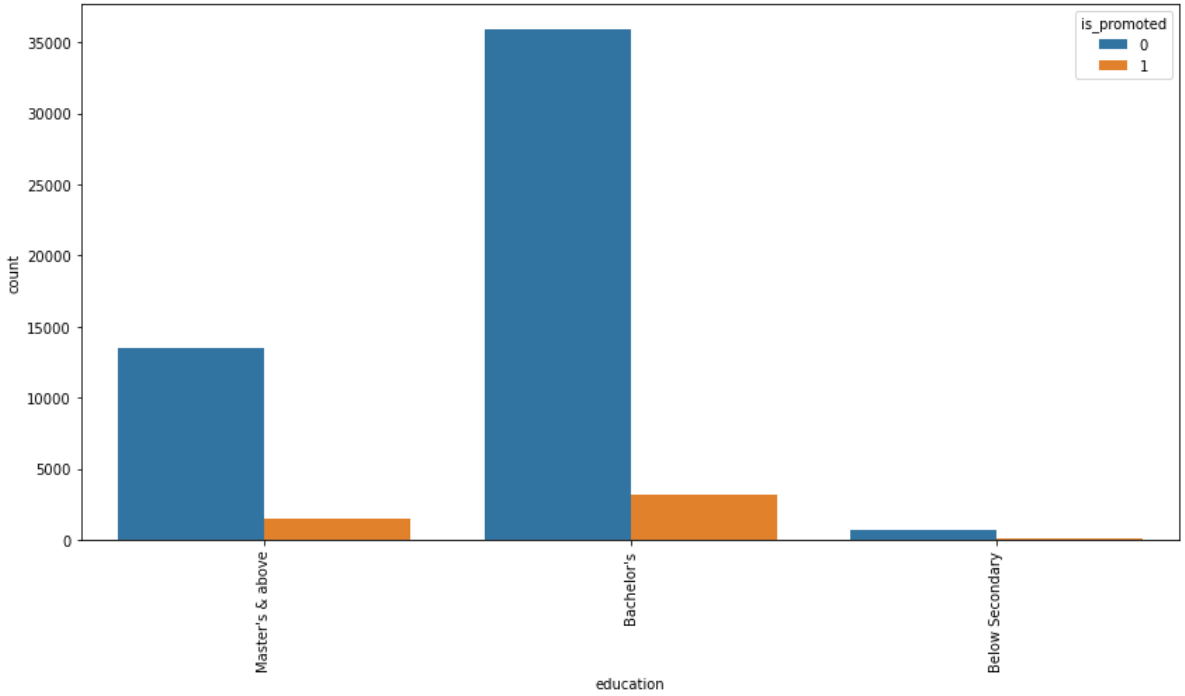
In [24]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['gender'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



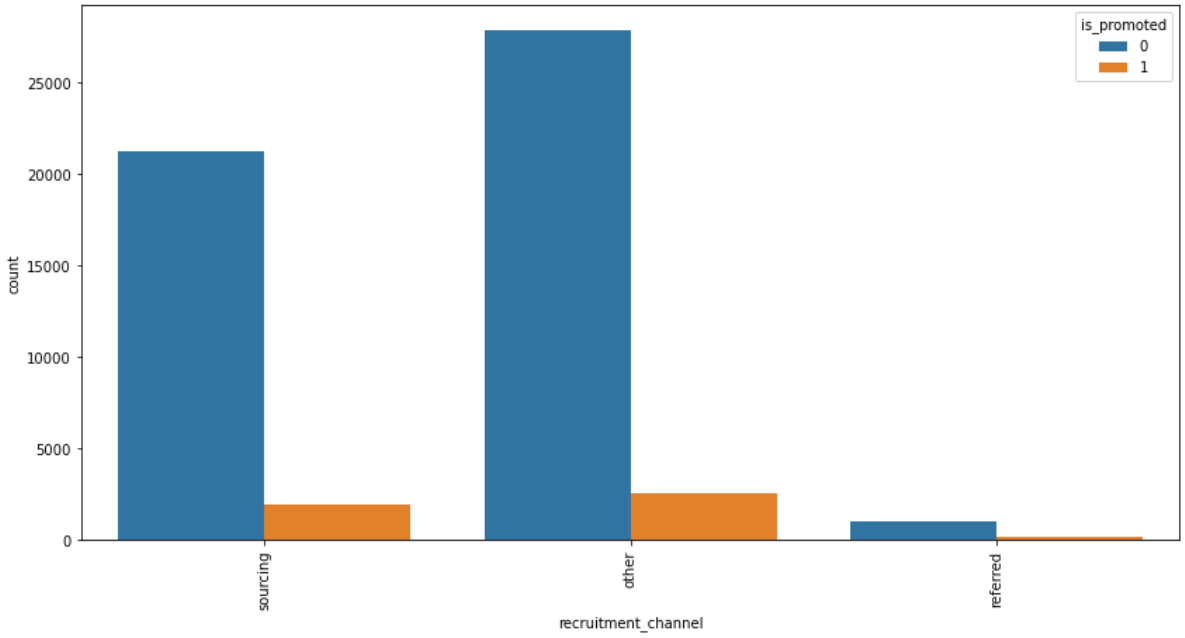
In [25]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['education'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



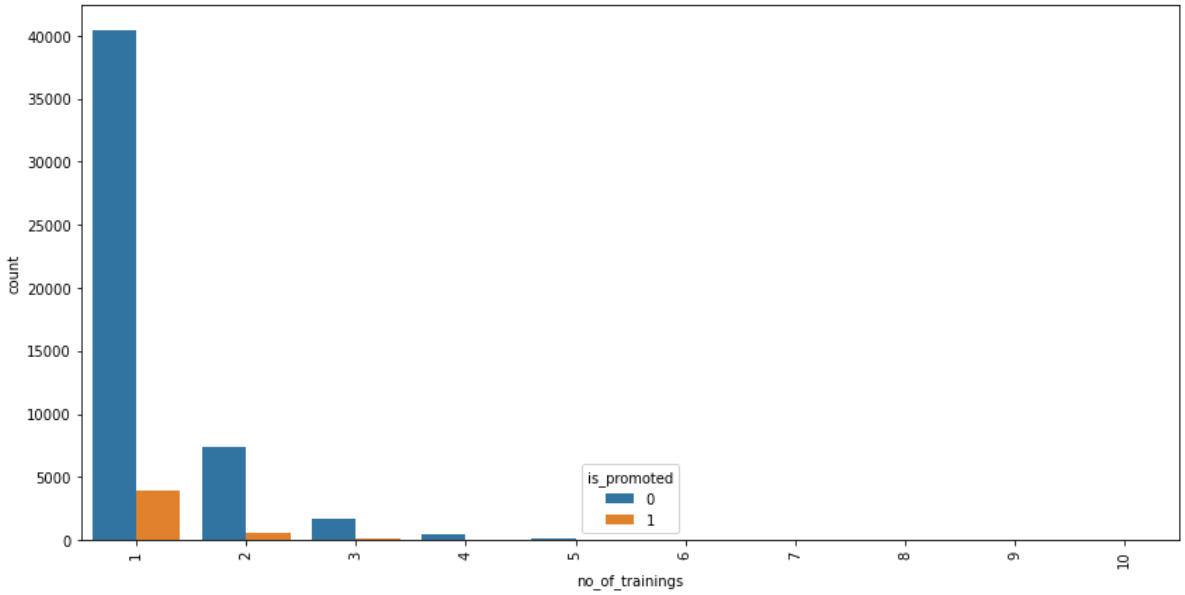
In [26]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['recruitment_channel'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



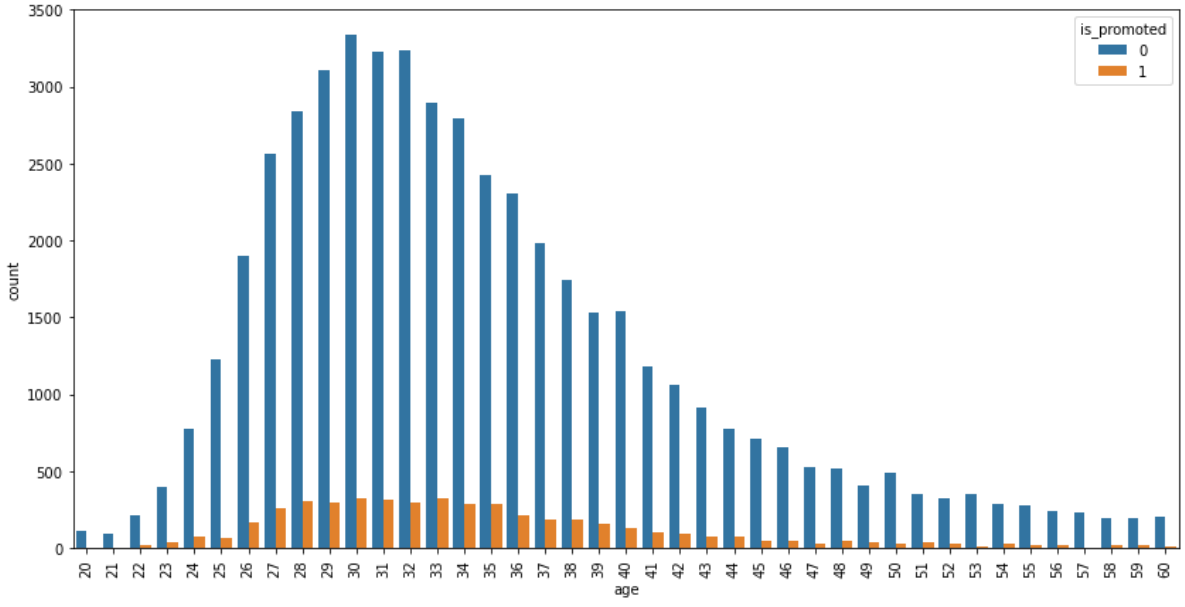
In [27]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['no_of_trainings'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



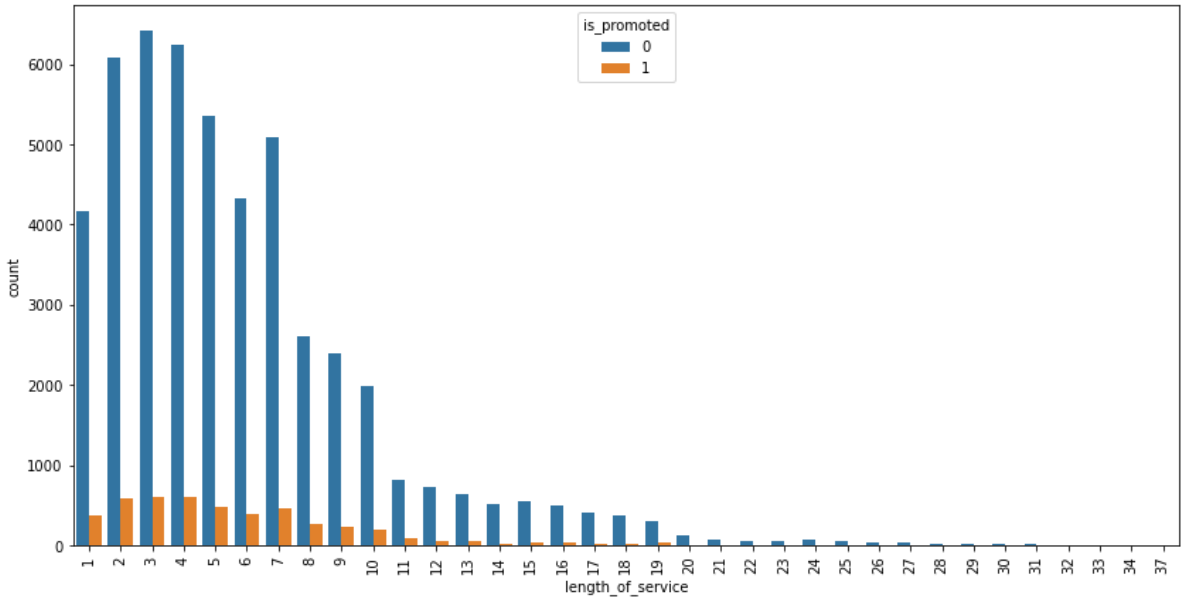
In [28]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['age'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



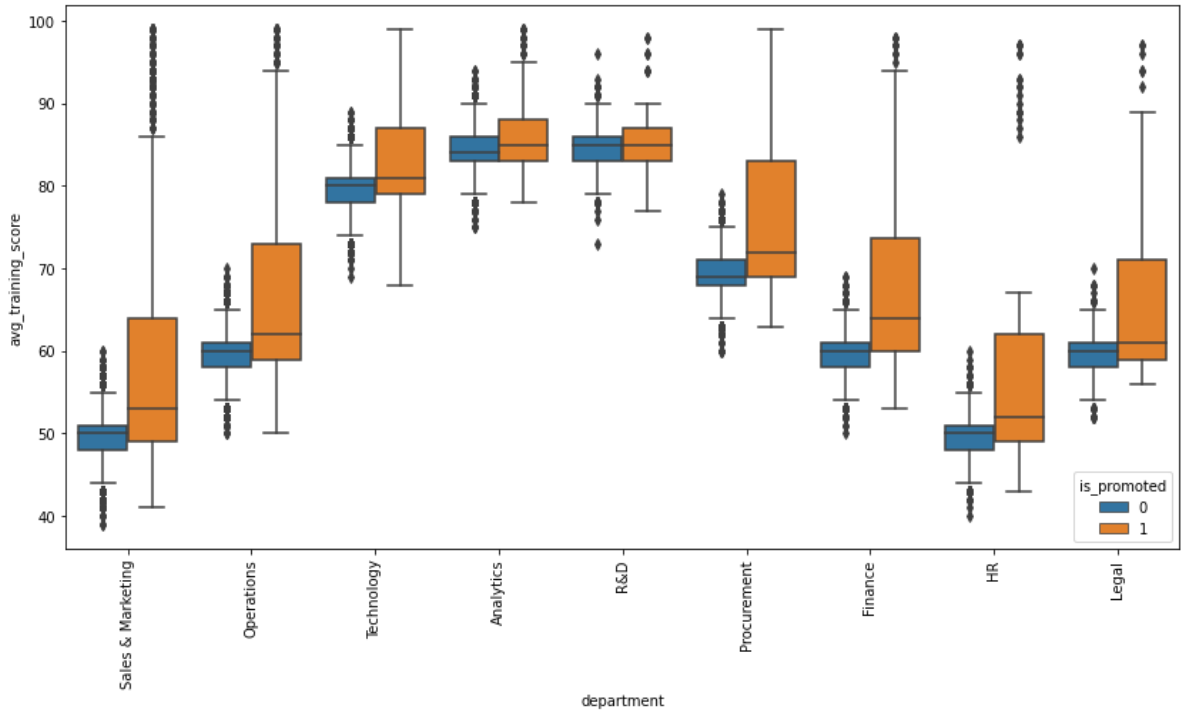
In [29]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['length_of_service'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



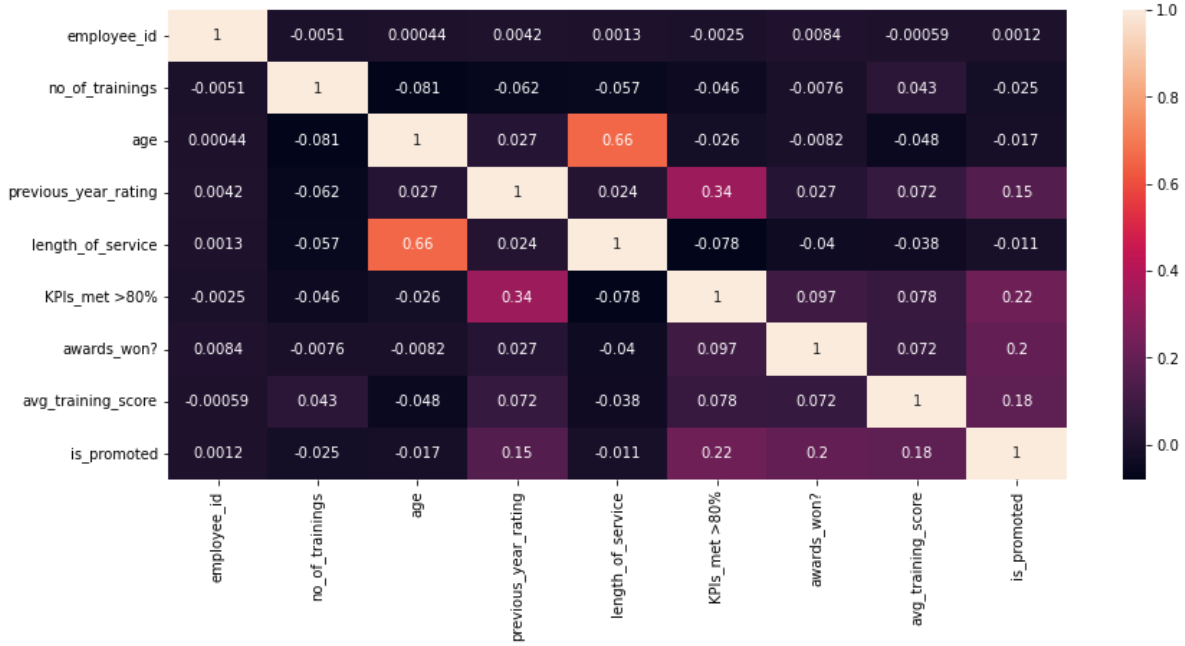
In [30]:

```
plt.figure(figsize=(14,7))
sns.boxplot(x=df['department'],y = df['avg_training_score'], hue = df['is_promoted'
plt.xticks(rotation=90)
plt.show()
```



In [31]:

```
plt.figure(figsize=(14,6))
sns.heatmap(df.corr(),annot=True)
plt.show()
```



In [32]:

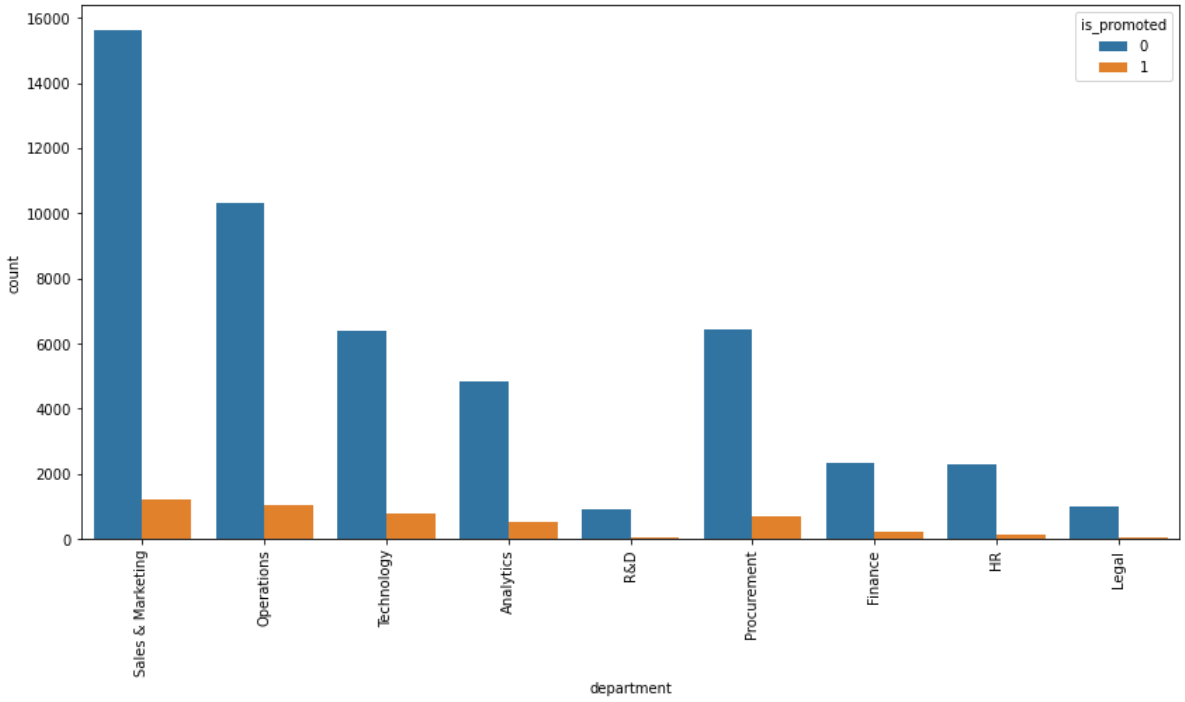
```
df['Sum_of_kpa']=df['KPIs_met >80%']+df['previous_year_rating']+df['awards_won?']
df.head()
```

Out[32]:

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings
0	65438	Sales & Marketing	region_7	Master's & above	f	sourcing	1
1	65141	Operations	region_22	Bachelor's	m	other	1
2	7513	Sales & Marketing	region_19	Bachelor's	m	sourcing	1
3	2542	Sales & Marketing	region_23	Bachelor's	m	other	2
4	48945	Technology	region_26	Bachelor's	m	other	1

In [33]:

```
plt.figure(figsize=(14,7))
sns.countplot(x=df['department'], hue=df['is_promoted'])
plt.xticks(rotation=90)
plt.show()
```



In [34]:

```
#df=df.drop(['employee_id','recruitment_channel','region'],axis=1,inplace=True)
df=df.drop(['recruitment_channel','avg_training_score','region','employee_id', 'no_
```

In [35]:

```
1 df.head()
```

Out[35]:

	department	education	gender	age	length_of_service	is_promoted	Sum_of_kpa
0	Sales & Marketing	Master's & above	f	35	8	0	6.0
1	Operations	Bachelor's	m	30	4	0	5.0
2	Sales & Marketing	Bachelor's	m	34	7	0	3.0
3	Sales & Marketing	Bachelor's	m	39	10	0	1.0
4	Technology	Bachelor's	m	45	2	0	3.0

In [36]:

```
1 df.columns
```

Out[36]:

```
Index(['department', 'education', 'gender', 'age', 'length_of_service',
       'is_promoted', 'Sum_of_kpa'],
      dtype='object')
```

from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test= train_test_split(df.iloc[:,
[0,1,2,3,4,6]],df.iloc[:,5:6],test_size=0.2,random_state=10)

x_train.head()

In [ ]:

```
1
```

In [37]:

```
1 from sklearn.preprocessing import OneHotEncoder
2
3 from sklearn.preprocessing import OrdinalEncoder
4
5 from sklearn.compose import ColumnTransformer
6
7 from sklearn.preprocessing import StandardScaler
```

In [38]:

```
#Data preprocessing

#normalizing alll the numerical features and encoding all the categorical features
```

In [39]:

```
cat_cols=['gender','education','department']
```

In [40]:

```
enc=OneHotEncoder(sparse=False)
```

In [41]:

```
gender=enc.fit_transform(df['gender'].values.reshape(-1,1))
gender_df=pd.DataFrame(gender,columns=list(enc.categories_[0]))

education=enc.fit_transform(df['education'].values.reshape(-1,1))
education_df=pd.DataFrame(education,columns=list(enc.categories_[0]))

department=enc.fit_transform(df['department'].values.reshape(-1,1))
department_df=pd.DataFrame(department,columns=list(enc.categories_[0]))

cat_df=pd.concat([gender_df,education_df,department_df],axis=1)
cat_df
```

Out[41]:

	f	m	Bachelor's	Below Secondary	Master's & above	Analytics	Finance	HR	Legal	Operations	Pr
0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
1	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	
2	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	
54803	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
54804	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	
54805	0.0	1.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	
54806	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
54807	0.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	

54808 rows × 14 columns

standard scaling

In [42]:

```
num_cols=['age','length_of_service','Sum_of_kpa']
scale=StandardScaler()
df[num_cols]=scale.fit_transform(df[num_cols])

num_df=df[num_cols]
num_df.head()
```

Out[42]:

	age	length_of_service	Sum_of_kpa
0	0.025598	0.500460	1.585631
1	-0.627135	-0.437395	0.902279
2	-0.104948	0.265996	-0.464424
3	0.547785	0.969387	-1.831127
4	1.331064	-0.906322	-0.464424

In [43]:

```
num_df.shape
```

Out[43]:

```
(54808, 3)
```

In [44]:

```
cat_df.shape
```

Out[44]:

```
(54808, 14)
```

In [45]:

```
features=pd.concat([num_df,cat_df],axis=1)
target=df['is_promoted']
```

In [46]:

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test= train_test_split(features,target,test_size=0.2,rando
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(43846, 17)
(43846,)
(10962, 17)
(10962,)
```

In [47]:

```
learn.linear_model import LogisticRegression
learn.tree import DecisionTreeClassifier
learn.metrics import accuracy_score
learn.ensemble import RandomForestClassifier,AdaBoostClassifier,GradientBoostingClassifie
```

In [48]:

```
model1=LogisticRegression().fit(x_train,y_train)
```

In [ ]:

```
#model2=DecisionTreeClassifier().fit(x_train,y_train)
```

In [50]:

```
model3=RandomForestClassifier().fit(x_train,y_train)
```

In [51]:

```
#model4=AdaBoostClassifier().fit(x_train,y_train)
```

In [ ]:

```
a
```

In [53]:

```
y_pred1=model1.predict(x_test)
#y_pred2=model2.predict(x_test)
y_pred3=model3.predict(x_test)
#y_pred4=model4.predict(x_test)


```

In [54]:

```
print("Logisticregression", accuracy_score(y_test,y_pred1))
#print("Decision Tree", accuracy_score(y_test,y_pred2))
print("RandomForest", accuracy_score(y_test,y_pred3))
#print("Adaboost", accuracy_score(y_test,y_pred4))


```

```
Logisticregression 0.9176245210727969
RandomForest 0.8986498814085021
```

In [ ]:

```
You can use different models also...for best results
```

In [ ]:

```

```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
#Trans=ColumnTransformer(transformers=[
#    ('tnf1',OrdinalEncoder(categories=[['Below Secondary',"Bachelor's","Master's &
#    ('tnf3',OneHotEncoder(sparse=False,drop='first'),['gender','department'])
#
#],remainder='drop')
```

tnf1= ColumnTransformer([('OHE_gender',OneHotEncoder(sparse=False,handle_unknown='ignore'), ['gender','department'])],remainder='passthrough')

tnf2= ColumnTransformer([('Ord_education',OrdinalEncoder(categories=[['Below Secondary',"Bachelor's","Master's & above"]]),['education'])],remainder='passthrough')

tnf3= ColumnTransformer([('scale',MinMaxScaler(),slice(0,6))],remainder='passthrough')

from sklearn.feature_selection import SelectKBest,chi2

tnf4=ColumnTransformer([('feature',SelectKBest(score_func=chi2,k=4))])

```
from sklearn.tree import DecisionTreeClassifier tnf5=DecisionTreeClassifier()

from sklearn.pipeline import Pipeline,make_pipeline

pipe1 = Pipeline([ ('tnf1',tnf1), ('tnf2',tnf2), ('tnf3',tnf3), ('tnf4',tnf4), ('tnf5',tnf5) ])

from sklearn import set_config set_config(display='diagram')

pipe1.fit(x_train,y_train)
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```