Importing basic library

In [2]:

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

importing data

In [3]:

```
df = pd.read_csv('/home/rajan/Desktop/EDA task/data0/aug_test.csv')
```

Data can be downloaded from the given link:

In [4]:

```
df.head()
```

Out[4]:

	enrollee_id	city	city_development_index	gender	relevent_experience	enrolled_universit
0	32403	city_41	0.827	Male	Has relevent experience	Full time cours
1	9858	city_103	0.920	Female	Has relevent experience	no_enrollmer
2	31806	city_21	0.624	Male	No relevent experience	no_enrollmer
3	27385	city_13	0.827	Male	Has relevent experience	no_enrollmer
4	27724	city_103	0.920	Male	Has relevent experience	no_enrollmer

About data:

```
dataset name: Predict who will move to a new job
enrollee_id : Unique ID for enrollee
city: City code
citydevelopmentindex: Developement index of the city (scaled)
gender: Gender of enrolee
relevent_experience: Relevent experience of enrolee
enrolled_university: Type of University course enrolled if any
education_level: Education level of enrolee
major_discipline :Education major discipline of enrolee
experience: Enrolee total experience in years
company_size: No of employees in current employer's company
company_type : Type of current employer
```

```
lastnewjob: Difference in years between previous job and current job
training_hours: training hours completed
```

In [5]:

```
df.columns
```

Out[5]:

```
Index(['enrollee_id', 'city', 'city_development_index', 'gender',
       'relevent_experience', 'enrolled_university', 'education_leve
l',
       'major_discipline', 'experience', 'company_size', 'company_typ
e',
       'last_new_job', 'training_hours'],
      dtype='object')
```

In [6]:

```
len(df.columns)
```

Out[6]:

```
13
```

In [7]:

```
df.shape
```

Out[7]:

```
(2129, 13)
```

In [ ]:

```

```

In [8]:

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2129 entries, 0 to 2128
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   enrollee_id             2129 non-null   int64  
 1   city                    2129 non-null   object 
 2   city_development_index  2129 non-null   float64
 3   gender                  1621 non-null   object 
 4   relevent_experience     2129 non-null   object 
 5   enrolled_university     2098 non-null   object 
 6   education_level         2077 non-null   object 
 7   major_discipline        1817 non-null   object 
 8   experience              2124 non-null   object 
 9   company_size            1507 non-null   object 
 10  company_type            1495 non-null   object 
 11  last_new_job            2089 non-null   object 
 12  training_hours          2129 non-null   int64  
dtypes: float64(1), int64(2), object(10)
memory usage: 216.4+ KB
```

In [9]:

```
1 df.describe()   #only categorical data are included on it
```

Out[9]:

	enrollee_id	city_development_index	training_hours
count	2129.000000	2129.000000	2129.000000
mean	16861.614843	0.824984	64.983091
std	9576.846029	0.125074	60.238660
min	3.000000	0.448000	1.000000
25%	8562.000000	0.698000	23.000000
50%	16816.000000	0.903000	47.000000
75%	25129.000000	0.920000	86.000000
max	33353.000000	0.949000	334.000000

In [10]:

```
df.isnull().sum()  #we are checking about  missing values
```

Out[10]:

```
enrollee_id                0
city                       0
city_development_index     0
gender                   508
relevent_experience        0
enrolled_university       31
education_level           52
major_discipline         312
experience                 5
company_size             622
company_type             634
last_new_job              40
training_hours             0
dtype: int64
```

```
OR
```

In [11]:

```
[i for i in df.columns if  df[i].isnull().sum()>0]
```

Out[11]:

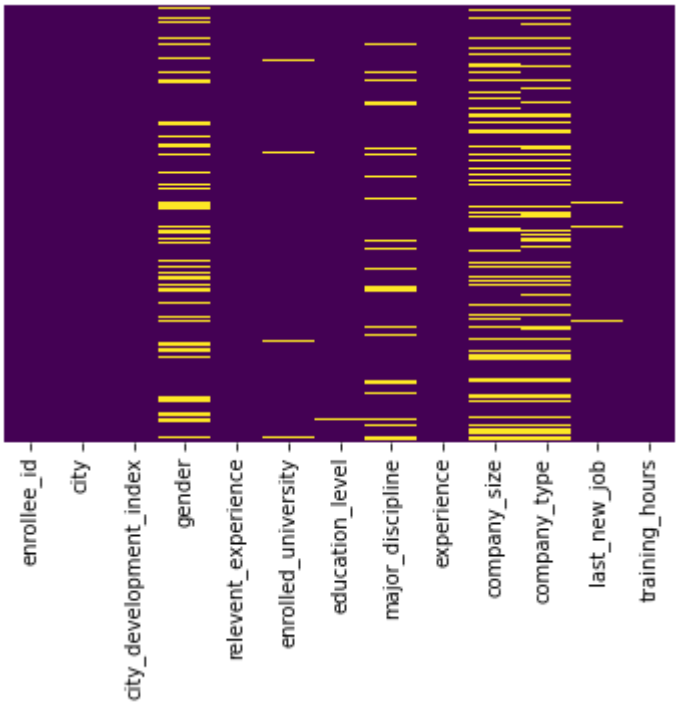
```
['gender',
 'enrolled_university',
 'education_level',
 'major_discipline',
 'experience',
 'company_size',
 'company_type',
 'last_new_job']
```

In [12]:

```
1 sns.heatmap(df.isnull(), yticklabels=False,cbar = False, cmap='viridis')
```

Out[12]:

<AxesSubplot:>



From this observation, we can see so many null values at gender,major-discipline,company size,company-type,and few in enrolled-university and last_new_job

to check data types

In [13]:

```
df.dtypes
```

Out[13]:

```
enrollee_id                int64
city                      object
city_development_index    float64
gender                    object
relevent_experience       object
enrolled_university       object
education_level           object
major_discipline          object
experience                object
company_size              object
company_type              object
last_new_job              object
training_hours             int64
dtype: object
```

In [14]:

```
df.gender.value_counts()
```

Out[14]:

```
Male      1460
Female     137
Other       24
Name: gender, dtype: int64
```

In [15]:

```
gender_count = df.gender.value_counts()
```

In [16]:

```
gender_count
```

Out[16]:

```
Male      1460
Female     137
Other       24
Name: gender, dtype: int64
```

In [ ]:

```

```

In [17]:

```
education_qualification = df.education_level.value_counts()

```

In [18]:

```
1 df.education_level.value_counts()
```

Out[18]:

```
Graduate          1269
Masters            496
High School        222
Phd                 54
Primary School      36
Name: education_level, dtype: int64
```

In [19]:

```
1 education_level_count = df.education_level.value_counts().index
```

In [20]:

```
1 gender_wise = df.gender.value_counts().index
```

In [21]:

```
1 gender_wise= df.groupby('gender').describe()
```

In [ ]:

```
1
```

Pie chart

In [22]:

```
1 plt.pie(gender_count,labels = gender_wise)
```

Out[22]:

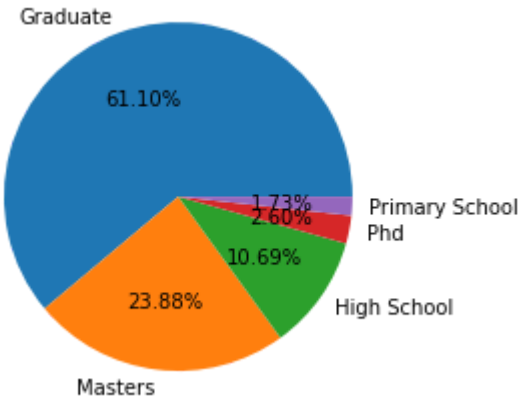
```
([<matplotlib.patches.Wedge at 0x7faf5a956340>,
  <matplotlib.patches.Wedge at 0x7faf5a956820>,
  <matplotlib.patches.Wedge at 0x7faf5a956d00>],
 [Text(-1.0468844310156573, 0.33768770794481606, "('enrollee_id', 'cou
nt')"),
  Text(1.0300508029733733, -0.38599914934350477, "('enrollee_id', 'mea
n')"),
  Text(1.0988102845567398, -0.05114644222560084, "('enrollee_id', 'st
d')")])
```

In [23]:

```
plt.pie(education_qualification,labels =education_level_count,autopct=
        '%1.2f%%')
```

Out[23]:

```
([<matplotlib.patches.Wedge at 0x7faf5a8ab760>,
  <matplotlib.patches.Wedge at 0x7faf5a8abeb0>,
  <matplotlib.patches.Wedge at 0x7faf5a8b9610>,
  <matplotlib.patches.Wedge at 0x7faf5a8b9d30>,
  <matplotlib.patches.Wedge at 0x7faf5a8c6490>],
 [Text(-0.37578776144341286, 1.0338198868029909, 'Graduate'),
  Text(-0.13525821404468816, -1.0916525159286912, 'Masters'),
  Text(0.9028396790461523, -0.6283951893035473, 'High School'),
  Text(1.0800833442792397, -0.20837458916713844, 'Phd'),
  Text(1.098369615839224, -0.05986807998587772, 'Primary School')],
 [Text(-0.2049751426054979, 0.5639017564379949, '61.10%'),
  Text(-0.07377720766073899, -0.5954468268701951, '23.88%'),
  Text(0.4924580067524466, -0.34276101234738937, '10.69%'),
  Text(0.5891363696068579, -0.11365886681843913, '2.60%'),
  Text(0.5991106995486676, -0.03265531635593329, '1.73%')])
```



Observation: Most of the enrollee are Graduated. In second, most of them are with master degree and then high school.

In [24]:

```
new_data=df.groupby(['experience','training_hours']).size().reset_index().renam
```

In [25]:

```
1 new_data.head(20)
```

Out[25]:

	experience	training_hours	Repeat Count
0	1	3	1
1	1	8	1
2	1	9	2
3	1	10	2
4	1	11	1
5	1	12	1
6	1	13	1
7	1	14	2
8	1	15	3
9	1	17	1
10	1	18	2
11	1	20	1
12	1	22	2
13	1	23	1
14	1	28	1
15	1	29	1
16	1	34	1
17	1	44	1
18	1	48	4
19	1	53	2

In [26]:

```
1 new_data.tail(20)
```

Out[26]:

	experience	training_hours	Repeat Count
1349	>20	176	1
1350	>20	178	1
1351	>20	182	1
1352	>20	192	1
1353	>20	196	1
1354	>20	200	2
1355	>20	212	2
1356	>20	220	1
1357	>20	248	2
1358	>20	256	1
1359	>20	266	1
1360	>20	270	1
1361	>20	280	1
1362	>20	282	1
1363	>20	290	1
1364	>20	292	1
1365	>20	298	1
1366	>20	304	1
1367	>20	322	1
1368	>20	328	1

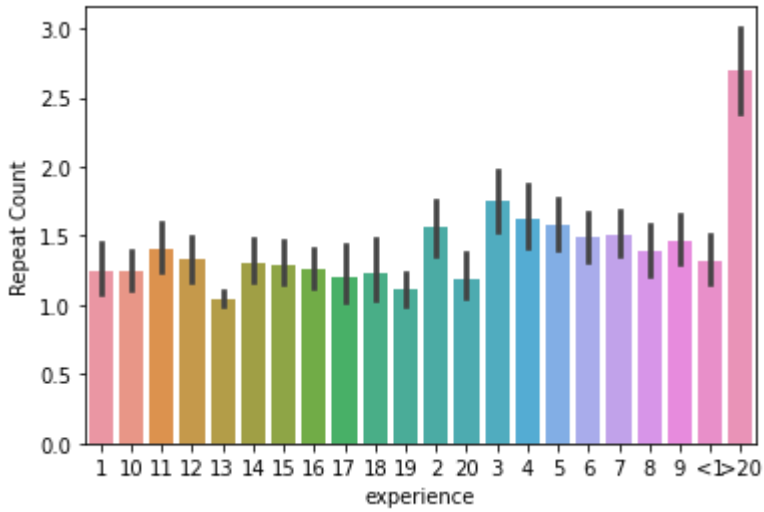
Observation: With increase in experience, the training hour completed alos increased.

In [27]:

```
1 sns.barplot(y ='Repeat Count',x ='experience', data = new_data)
```

Out[27]:

```
<AxesSubplot:xlabel='experience', ylabel='Repeat Count'>
```



Observation: Most of Enrolle are with experience greater than 20 are there

In [ ]:

```
1
```

In [28]:

```
1 df.major_discipline.value_counts()
```

Out[28]:

```
STEM                1621
Humanities            80
Other                 40
Business Degree       37
No Major              22
Arts                  17
Name: major_discipline, dtype: int64
```

Observation: Most of the enrollee are having major in Science Technology Engineering and Mathematics.

In [29]:

```
df2 = df.groupby(['city','city_development_index']).size().reset_index().rename
```

In [ ]:

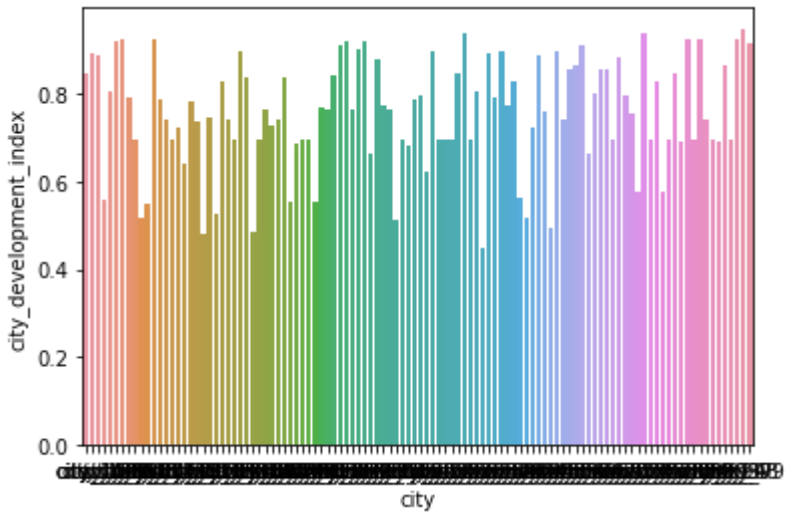
```

```

In [30]:

```
sns.barplot(x ='city' , y = 'city_development_index', data = df2)
```

Out[30]:

```
<AxesSubplot:xlabel='city', ylabel='city_development_index'>
```



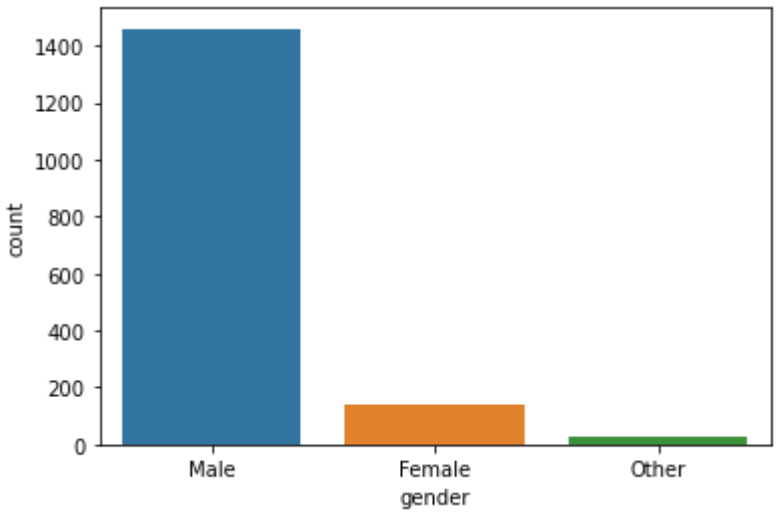
Observation: There are very few city with low city development indexes between 0.4-0.6

In [31]:

```
sns.countplot(x = 'gender',data = df)
```

Out[31]:

```
<AxesSubplot:xlabel='gender', ylabel='count'>
```



Observation: Most of them are male

In [32]:

```
1 df.duplicated().sum()
```

Out[32]:

```
0
```

Observation: No duplicate value

Statistical Analysis

In [33]:

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

Out[33]:

	enrollee_id	city_development_index	training_hours
count	2129.000000	2129.000000	2129.000000
mean	16861.614843	0.824984	64.983091
std	9576.846029	0.125074	60.238660
min	3.000000	0.448000	1.000000
25%	8562.000000	0.698000	23.000000
50%	16816.000000	0.903000	47.000000
75%	25129.000000	0.920000	86.000000
max	33353.000000	0.949000	334.000000

In [34]:

```
1 df.skew()
```

```
/tmp/ipykernel_19825/1665899112.py:1: FutureWarning: Dropping of nuisa
nce columns in DataFrame reductions (with 'numeric_only=None') is depr
ecated; in a future version this will raise TypeError.  Select only va
lid columns before calling the reduction.
  df.skew()
```

Out[34]:

```
enrollee_id              -0.015213
city_development_index   -0.923030
training_hours            1.876451
dtype: float64
```

Observation: city development index is left skwed

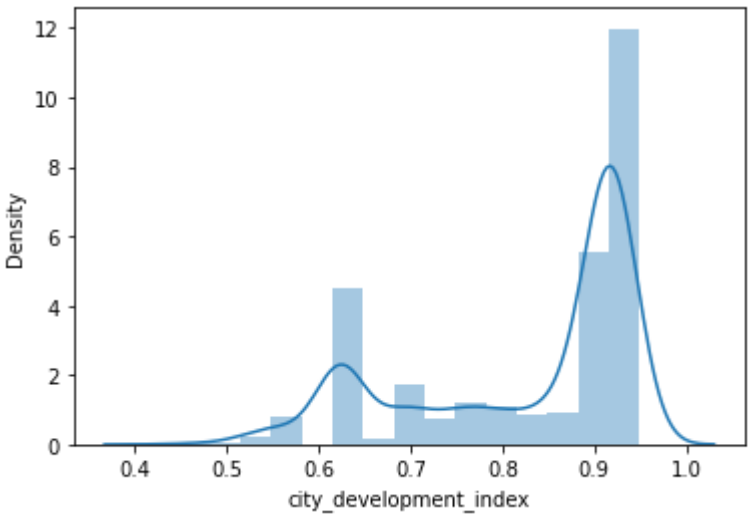
In [35]:

```
1 sns.distplot(df['city_development_index'])
2
```

```
/home/rajan/anaconda3/lib/python3.9/site-packages/seaborn/distribution
s.py:2619: FutureWarning: `distplot` is a deprecated function and will
be removed in a future version. Please adapt your code to use either `
displot` (a figure-level function with similar flexibility) or `histpl
ot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[35]:

```
<AxesSubplot:xlabel='city_development_index', ylabel='Density'>
```



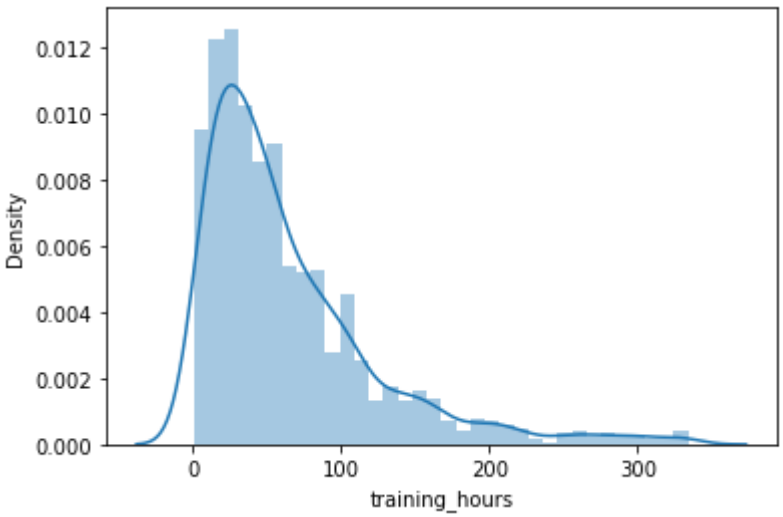
In [36]:

```
1 sns.distplot(df['training_hours'])
2
```

```
/home/rajan/anaconda3/lib/python3.9/site-packages/seaborn/distribution
s.py:2619: FutureWarning: `distplot` is a deprecated function and will
be removed in a future version. Please adapt your code to use either `
displot` (a figure-level function with similar flexibility) or `histpl
ot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[36]:

```
<AxesSubplot:xlabel='training_hours', ylabel='Density'>
```



Observation: Training hours data is left skewed. Outlier lies at the right side.

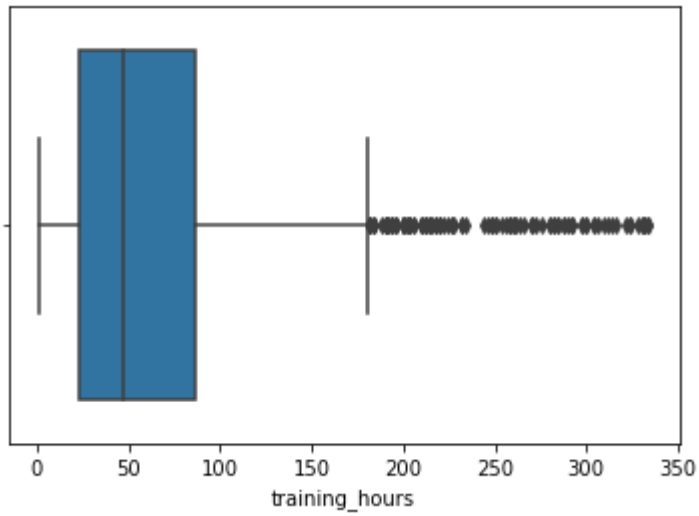
Checking Outliers

In [37]:

```
sns.boxplot(x = 'training_hours', data = df)
```

Out[37]:

<AxesSubplot:xlabel='training_hours'>



Observation: As previously mentioned outliers lies on the left side of the graph.

Lets count:

In [38]:

```
def lowerfence_higherfence(variable):
    q1= df[variable].quantile(0.25)
    q3 = df[variable].quantile(0.75)
    IQR = q3-q1
    lowerfence = q1- 1.5*IQR
    higherfence  = q3 + 1.5*IQR
    return lowerfence,higherfence
```

In [39]:

```
1 lowerfence_higherfence('training_hours')
```

Out[39]:

```
(-71.5, 180.5)
```

Observation: Training hours more than 180.5 is outliers.

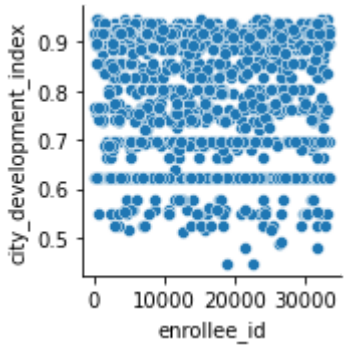
graph analysis

In [40]:

```
1 sns.pairplot(y_vars = 'city_development_index', x_vars = 'enrollee_id', data =
```

Out[40]:

```
<seaborn.axisgrid.PairGrid at 0x7faf59c24ee0>
```



In [41]:

```
1 df.groupby('education_level').size()
```

Out[41]:

```
education_level
Graduate          1269
High School        222
Masters            496
Phd                 54
Primary School      36
dtype: int64
```

Observation: Very few people from low city development index city

Q. City with high number of Phd

In [42]:

```
1 df[df['education_level']=='Phd'].groupby('city').size()
```

Out[42]:

```
city
city_100     1
city_103    17
city_104     2
city_114     6
city_123     1
city_134     1
city_136     2
city_16      9
city_160     1
city_165     3
city_28      2
city_45      1
city_61      1
city_65      1
city_67      1
city_71      1
city_75      3
city_77      1
dtype: int64
```

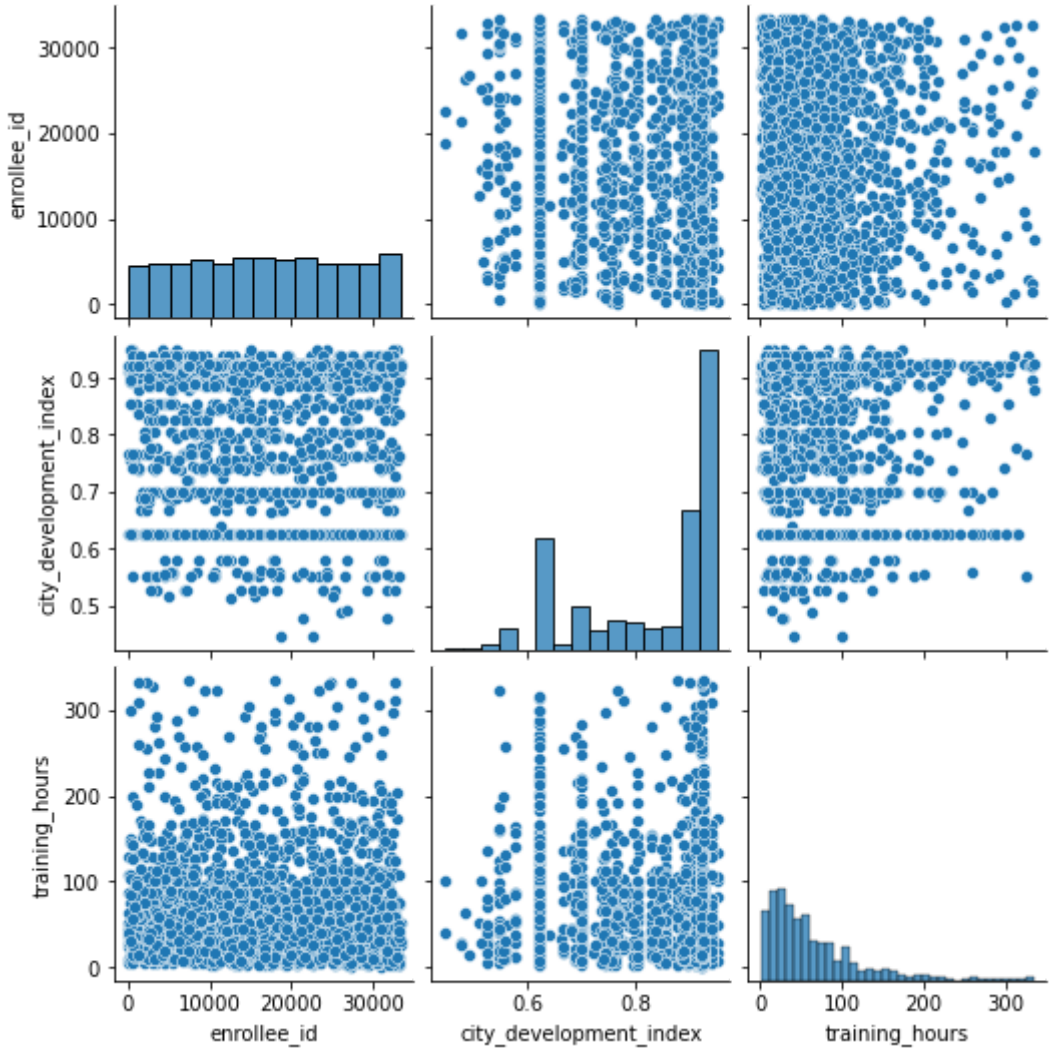
Observation: City 103 has a most number of phd

In [43]:

```
1 sns.pairplot(df)
```

Out[43]:

```
<seaborn.axisgrid.PairGrid at 0x7faf5a9c2280>
```



In [44]:

```
#now let us save the file for the further feature enginering
```

In [46]:

```
df.to_csv('final_data.csv', index=False)

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```