دانشکده مهندسی کامپیوتر و فناوری اطلاعات

Kasra Khalafi: 9531306

Home Work 1

Data Mining

Dr.Nazerfard

1 –

a) Supervised:Supervised data are those data which have label, which means that we know that which input results in what out put and also in supervised learning the main job is to map these inputs to the outputs based on input-output pairs.

b) Unsupervised:in Unsupervised data, unlikely to supervised, data have no labal and there isn't prepattern and the two main methods used in unsupervised learning are principal component analyst and cluster analyst.

c) Semi-Supervised: it is a combination of small amout of labeled data with a large amount of unlabeled data. So as it is written, semi-supervies data are between supervised and unsupervised data.

d) Outlier: an outlier is a data point that differs significantly from other observations.

e) Missing value Vs. Noise: these two parameters are different from each other. Missing value are those values that are not recorded. This problem could be due to equipment malfunction,not register history or change of data,data not entered due to misunderstanding and so on. And also another reason for this could be that data in not always available. But noise is random error or variance in measured variable. This noise problem could be due to data transmission problems, technology limitation, incomplete data, etc.

2- There are several different methods to manipulate this problem.when the class label is missing we could ignore the tuple when doing classification but it isn't effective when the percentage of missing value per feature varies. Other method is to fill the missing value manually which could be tedious. And the other way is to fill it automatically with the feature mean, a global constant, or the most probable value with respect to Bayesian formula.

3-The minsup is 60% and TID is 5 so the support count should be greater or rqual to 3.

Apriori:

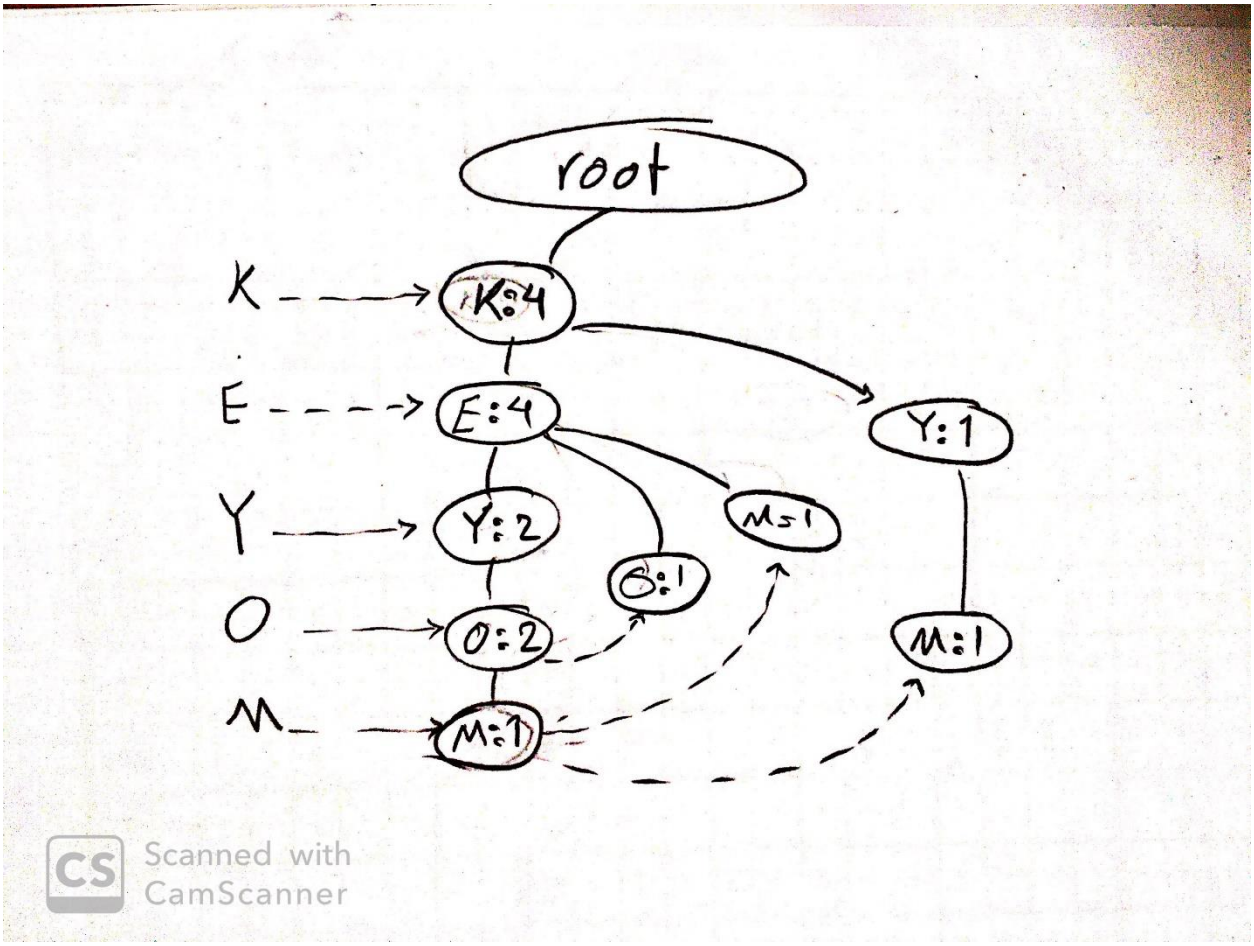
Item	Count
M	3
O	3
N	2
K	5
E	4
Y	3
D	1
A	1
U	1
C	1
I	1

Item	Count
M,O	1
M,K	3
M,E	2
M,Y	2
O,K	3
O,E	3
O,Y	2
K,E	4
K,Y	3
E,Y	2

Item	Count
M,K,O	1
M,O,E	1
O,K,E	3
O,K,Y	1
K,Y,E	2

FP-Growth:

TID	Item Sets	Ordered Item Sets
T100	{M,O,N,K,E,Y}	K,E,Y,O,M
T200	{D,O,N,K,E,Y}	K,E,Y,O
T300	{M,A,K,E}	K,E,M
T400	{M,U,C,K,Y}	K,Y,M
T500	{C,O,O,K,I,E}	K,E,O



4-

A:The maximum association rules that can be extracted is 2^d which is 2^6=64,so in this example association rules are less or equal to 64.

{MILK},{BEER},{DIAPERS},[BREAD},{BUTTER}, {COOKIES}

{MILK,BEER}, {MILK,DIAPERS}, {BEER,DIAPERS}, {BREAD,BUTTER}, {BREAD,MILK}, {BUTTER,MILK}, {MILK,COOKIE}, {DIAPER,COOKIES}, {BREAD,COOKIES}, {BUTTER,COOKIES}, {BEER,COOKIES}, {DIAPERS,BREAD}, {DIAPERS,BUTTER}

{MILK,BEER,DIAPERS}, {BREAD,BUTTER,MILK}, {BREAD,BUTTER,COOKIES}, {BEER,COOKIES,DIAPRES}, {MILK,BREAD,BUTTER}, {DIAPERS,BREAD,BUTTER}

{MILK,DIAPERS,BUTTER,BREAD}

As calculated above, number of associative rules are 25(if I didn't make mistake in counting).

B: The maximum Frequent Itemset that can be extracted is 2^d (d here is 6) so 2^6=64 is the maximum frequent itemset that can be extracted

C:

{MILK,BEER,DIAPERS} (support count = 1)

{BREAD,BUTTER,MILK} (support count = 3)

{BREAD,BUTTER,COOKIES} (support count = 1)

{BEER,COOKIES,DIAPRES} (support count = 1)

{DIAPERS,BREAD,BUTTER} (support count = 3)

D:

{MILK,BEER} = 1

{MILK,DIAPERS} = 4

{BEER,DIAPERS} = 3

{BREAD,BUTTER} = 5

{BREAD,MILK} = 3

{BUTTER,MILK} = 3

{MILK,COOKIE} = 1

{DIAPER,COOKIES} = 2

{BREAD,COOKIES} = 1

{BUTTER,COOKIES} = 1

{BEER,COOKIES} = 2

{DIAPERS,BREAD} = 3

{DIAPERS,BUTTER} = 3

As it is written above, the largest support belongs to Bread & Butter itemset with a support of 5 (and due to Apriori, itemsets with size of 2, has the same or larger supports rather than other itemsets with larger size like 3 or 4)

E:

{BREAD} → {MILK} && {MILK} → {BREAD} : have the same confidence of 0.6

{BUTTER} → {MILK} && {MILK} → {BUTTER} : have the same confidence of 0.6

{BREAD} → {BUTTER} && {BUTTER} → {BREAD} : have the same confidence of 1

5-A database/data warehouse may store terabytes of data. Complex data analysis may take a very long time to run on the complete data set. So we could speed up our calculations and reduce mass of unefficient data.

Data reduction strategies are:

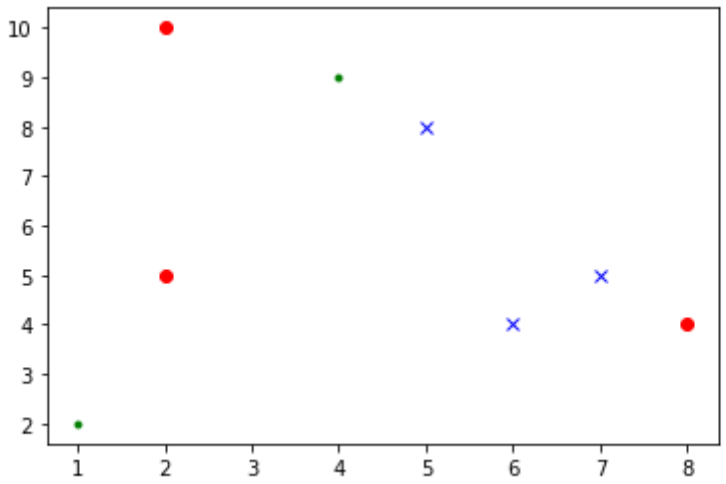
- Dimensionality reduction, e.g., remove unimportant features
    - Wavelet transforms
    - Principal Components Analysis (PCA)
    - Feature subset selection, feature creation
- Numerosity reduction (Data Reduction)
    - Regression and Log-Linear Models
    - Histograms, clustering, sampling
    - Data cube aggregation
- Data compression

6- A zero correlation suggests that the correlation statistic did not indicate a relationship between the two variables. It's important to note that this does not mean that there is not a relationship at all; it simply means that there is not a linear relationship

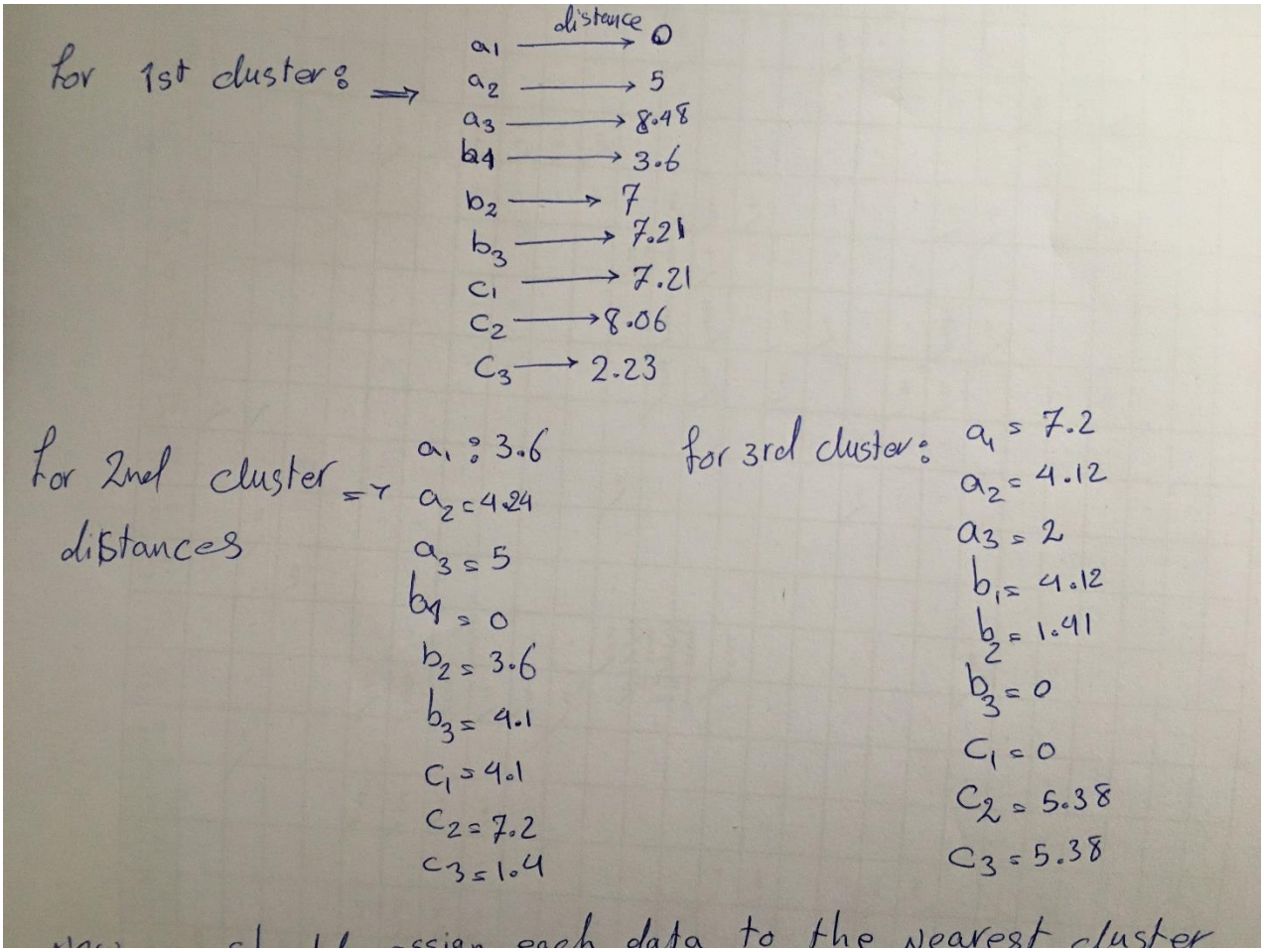
---

7- first we plot the figure with python. Code and plot of the data are :

```
import matplotlib.pyplot as plt
a_x = [2, 2, 8]
a_y = [10, 5, 4]
b_x = [5, 7, 6]
b_y = [8, 5, 4]
c_x = [1, 4]
c_y = [2, 9]
plt.plot(a_x,a_y,'ro')
plt.plot(b_x,b_y,'bx')
plt.plot(c_x,c_y,'g.')

plt.show()
```



Here is the calculated distance each data to the clusters:



Now we should assign each data to the nearest cluster :

A1 ======➔ 1

A2 ======➔ 3

A3 ======➔ 3

B1 ======➔ 2

B2 ======➔ 3

B3 ======➔ 3

C1 ======➔ 3

C2 ======➔ 2

And the new centers are :

Center 1 = A1 = (2, 10)

Center 2 = (B1 + C2)/2 = (4.5, 8.5)

Center 3 = (4.8, 4)

We continue this method till the assigning the data to cluster does not change

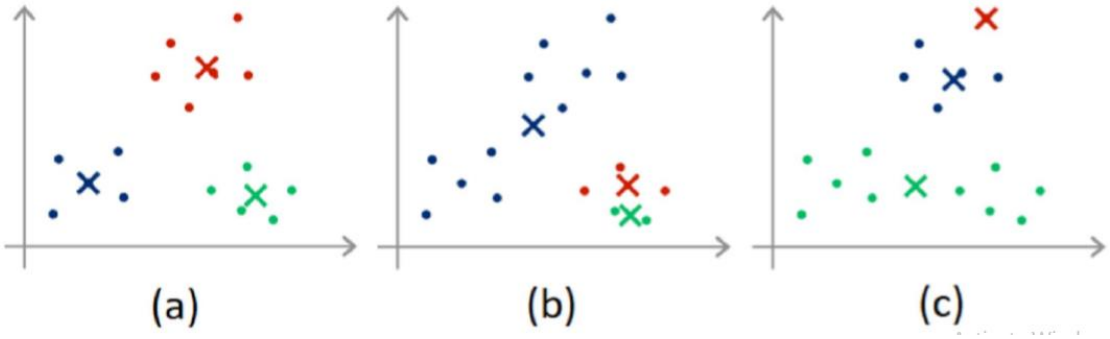
After recalculating, it will be considered that the clusters does not change so here are the final clusters:

1 ==========➔ (2,10)

2 ==========➔ (4.5, 8.5)

3 ==========➔ (4.8, 4)

8- K-means algorithm does not always find the global optimum and it depends on the initialization so it could find different optimum and clusters depended on the initialization. Bellow is an example of finding different clusters and as shown bellow (a) is a good clustring but (b) and (c) are stuck in a local optimum.



Implementation:

A-In the first part we use csv_read to read the data with the help of pandas library, and use index_col = ‘id’ to write the columns and rows with been shown with id in the left and sorted, I also used parse_date to return the dates to the formal way.

```
import pandas
df = pandas.read_csv('data.csv',
                     index_col='id',
                     parse_dates=['confirmed_date'],
                     header=0)
print(df)
```

```
        sex  birth_year country  ... infected_by confirmed_date     state
id                               ...
1    female      1984.0   China  ...         NaN     2020-01-20  released
2      male      1964.0   Korea  ...         NaN     2020-01-24  released
3      male      1966.0   Korea  ...         NaN     2020-01-26  released
4      male      1964.0   Korea  ...         NaN     2020-01-27  released
5      male      1987.0   Korea  ...         NaN     2020-01-30  released
..      ...         ...     ...  ...         ...            ...       ...
172  female      1997.0   Korea  ...         NaN     2020-02-24  isolated
173    male      1949.0   Korea  ...         NaN     2020-02-24  deceased
174  female      1958.0   Korea  ...         NaN     2020-02-24  isolated
175    male      1997.0   Korea  ...         NaN     2020-02-24  isolated
176  female      1950.0   Korea  ...         NaN     2020-02-24  isolated

[176 rows x 8 columns]
```

B-In this part we have to write the columns name and also size.we use the df.size() for the size (df is the pandas.csv_read()) and we read each columns name with df.columns.values which is an array so we read each element.

```
------------------------------------------------
The columns of file are :

1: sex
2: birth_year
3: country
4: region
5: infection_reason
6: infected_by
7: confirmed_date
8: state
------------------------------------------------

The size of data is :
1408
------------------------------------------------
```

C-In this part we have to calculate max, mean and std for birth day.

We use `df['birth_year'].std()for std and like this for min and mean.`

```
----------------------------------------
The min is: 1936.0
The mean is: 1973.3855421686746
The std is: 17.032824869574775
----------------------------------------
```

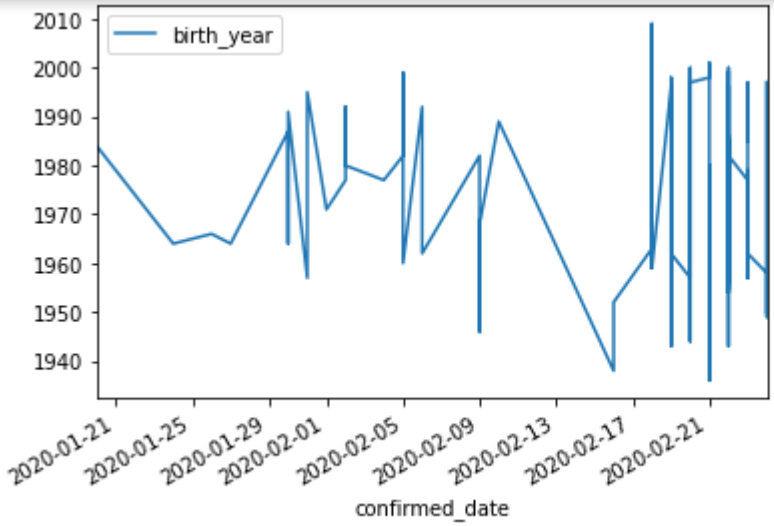
D-I chose infected_by and showed it as bellow. The empety ones have been named as NaN to handle better,and filled one datas are 42 of the all columns

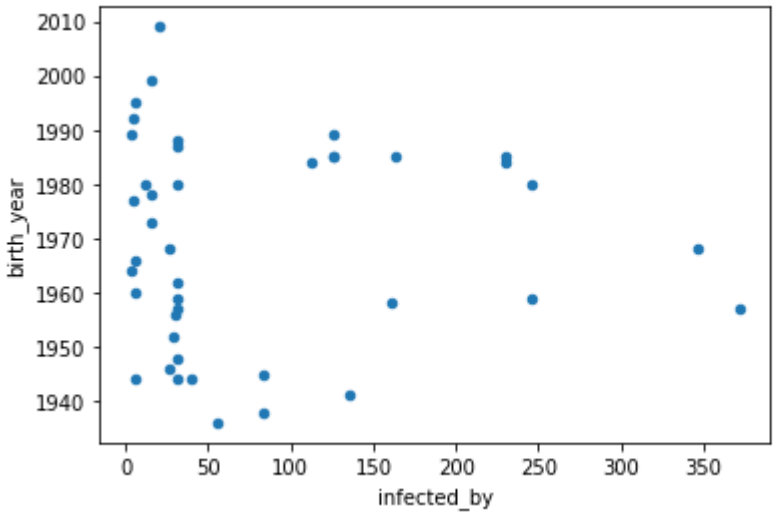
```
----------------------------------------
The Infected_by column is as shown bellow
id
1      NaN
2      NaN
3      NaN
4      NaN
5      NaN
       ..
172    NaN
173    NaN
174    NaN
175    NaN
176    NaN
Name: infected_by, Length: 176, dtype: float64

number of not empety fields also is :
42

----------------------------------------
```

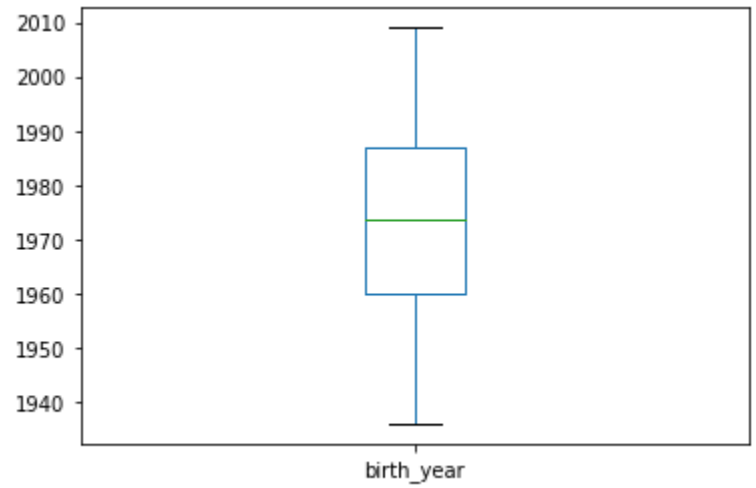
E-First of all I checked that do we have null or not and I realised as there was discrepancy beetwen the filled one and all one there is some null datas so by adding `na_values={'birth_year':['-1']}` to fill empety birth_years by NaN

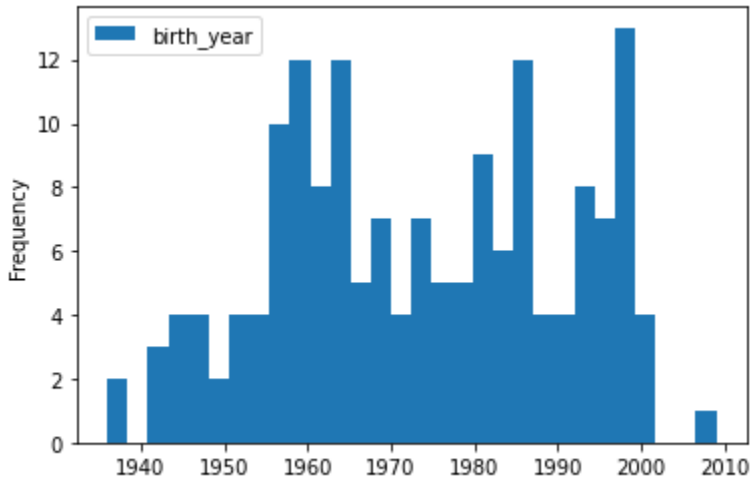
```
----------------------------------------
There is null in the birth year
empety datas are replaced by NaN
----------------------------------------
```

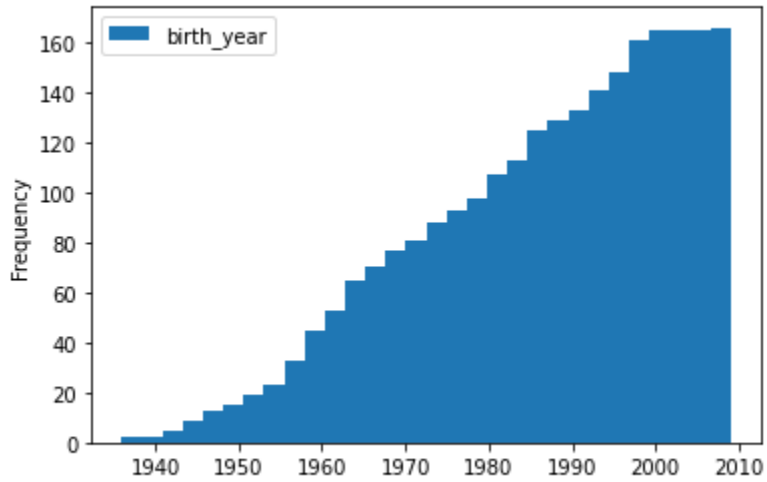
F-Here are the plots as written :


birth_year
2010
2000
1990
1980
1970
1960
1950
1940
2020-01-21
2020-01-25
2020-01-29
2020-02-01
2020-02-05
2020-02-09
2020-02-13
2020-02-17
2020-02-21
confirmed_date


birth_year
2010
2000
1990
1980
1970
1960
1950
1940
0
50
100
150
200
250
300
350
infected_by


2010
2000
1990
1980
1970
1960
1950
1940
birth_year


birth_year
Frequency
0
2
4
6
8
10
12
1940
1950
1960
1970
1980
1990
2000
2010


birth_year
Frequency
0
20
40
60
80
100
120
140
160
1940
1950
1960
1970
1980
1990
2000
2010

G- This part is described in persian in the material support file