In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Workshop 1: Data Analysis with Pandas

In [2]:
```python
data = pd.read_csv('adult.csv')
```

In [3]:
```python
data.head()
```

Out[3]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	class labe
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50k
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50k
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50k
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50k
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50k

Question no 1

In [4]:
```python
data.head(2)
```

Out[4]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	class-label
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K

This command shows the first two rows of the data set

In [5]:
```python
data.head(10)
```

Out[5]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	class labe
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50k
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50k
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50k
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50k
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50k
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	0	0	40	United-States	<=50k
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0	0	16	Jamaica	<=50k
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	45	United-States	>50k
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female	14084	0	50	United-States	>50k
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	5178	0	40	United-States	>50k

this shows the first 10 rows of the data set

In [6]:
```
data.tail(2)
```

Out[6]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	c
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Male	0	0	20	United-States	<
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Female	15024	0	40	United-States	

This commands shows the last two rows of the data set

In [7]:
```
data.shape
```

Out[7]:
```
(32561, 15)
```

The data frame shape property tells you the dimensionalty of the data set in the form of number of rows and columns.

This data has 32561 rows and 15 columns.

Unique Data Set

In [39]:
```
data = data.sample(n=30000, random_state = 70)
```

In [40]:
```
data.shape
```

Out[40]:
```
(30000, 14)
```

In [41]:
```
data.describe()
```

Out[41]:

	age	education-num	capital-gain	capital-loss	hours-per-week
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.00000
mean	38.589133	10.080400	1086.949933	87.256033	40.43530
std	13.635182	2.572396	7459.713916	403.036258	12.34685
min	17.000000	1.000000	0.000000	0.000000	1.00000
25%	28.000000	9.000000	0.000000	0.000000	40.00000
50%	37.000000	10.000000	0.000000	0.000000	40.00000
75%	48.000000	12.000000	0.000000	0.000000	45.00000
max	90.000000	16.000000	99999.000000	4356.000000	99.00000

This command gives the description of the data. It shows the mean, standard deviation, count, minimum value maximum value and percentiles.

In [11]:
```
data['education-num'].value_counts()
```

Out[11]:
```
9     9676
10    6714
13    4955
14    1578
11    1267
7     1081
12     984
6      861
4      598
15     524
5      483
8      393
16     383
3      304
2      153
1       46
Name: education-num, dtype: int64
```

In [12]:
```
data = data.drop(['fnlwgt'], axis=1)
```

This command drops the column 'fnlwgt' and I have used axis=1 to drop the first row.

In [13]:
```
data.shape
```

Out[13]:
```
(30000, 14)
```

As I have dropped one coulumn 'fnlwgt' from the data so now the data has 14 columns.

In [14]:
```
data.describe(include='all')
```

Out[14]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	ho
count	30000.000000	30000	30000	30000.000000	30000	30000	30000	30000	30000	30000.000000	30000.000000	300
unique	NaN	9	16	NaN	7	15	6	5	2	NaN	NaN	
top	NaN	Private	HS-grad	NaN	Married-civ-spouse	Prof-specialty	Husband	White	Male	NaN	NaN	
freq	NaN	20927	9676	NaN	13791	3818	12166	25615	20049	NaN	NaN	
mean	38.589133	NaN	NaN	10.080400	NaN	NaN	NaN	NaN	NaN	1086.949933	87.256033	
std	13.635182	NaN	NaN	2.572396	NaN	NaN	NaN	NaN	NaN	7459.713916	403.036258	
min	17.000000	NaN	NaN	1.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	
25%	28.000000	NaN	NaN	9.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	
50%	37.000000	NaN	NaN	10.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	
75%	48.000000	NaN	NaN	12.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	
max	90.000000	NaN	NaN	16.000000	NaN	NaN	NaN	NaN	NaN	99999.000000	4356.000000	

This commands shows the description of all the variables we have in data set.

In [15]:
```
data['education'].nunique()
```

Out[15]:
```
16
```

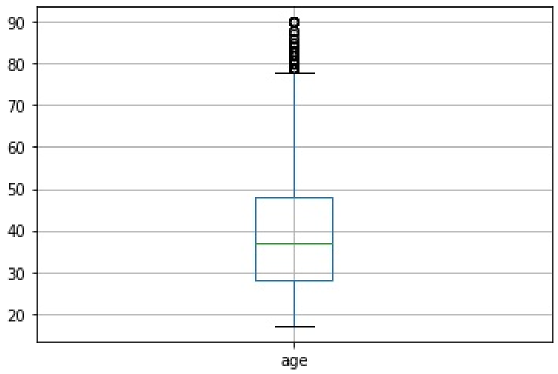
This commands tell about the unique values we have in education column.

In [16]:
```
data['age'].value_counts()
```

Out[16]:
```
31    824
35    821
36    812
23    807
28    804
     ...
83      5
85      3
88      3
86      1
87      1
Name: age, Length: 73, dtype: int64
```

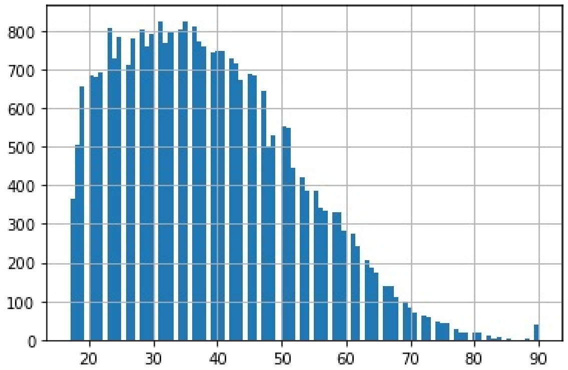
In [17]:
```
data.boxplot(column='age')
```

Out[17]:
```
<AxesSubplot:>
```



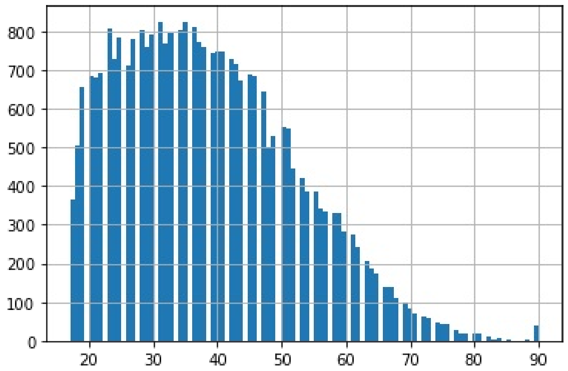
In [18]:
```
data['age'].hist(bins=100)
```

Out[18]:
```
<AxesSubplot:>
```



In [19]:
```
data.age.hist(bins=100)
```

Out[19]:
```
<AxesSubplot:>
```



In [20]:
```
data['sex'].value_counts()
```

Out[20]:
```
 Male      20049
 Female     9951
Name: sex, dtype: int64
```

The data is collected from 20049 males and 9951 females.

In [21]:
```
data.columns
```

Out[21]:
```
Index(['age', 'workclass', 'education', 'education-num', 'marital-status',
       'occupation', 'relationship', 'race', 'sex', 'capital-gain',
       'capital-loss', 'hours-per-week', 'native-country', 'class-label'],
      dtype='object')
```

In [22]:
```
data['workclass'].value_counts()
```

Out[22]:
```
 Private             20927
 Self-emp-not-inc     2336
 Local-gov            1925
 ?                    1694
 State-gov            1191
 Self-emp-inc         1020
 Federal-gov           888
 Without-pay            13
 Never-worked            6
Name: workclass, dtype: int64
```

Question no 2

In [23]:
```
data['sex'].value_counts()
```

Out[23]:
```
 Male      20049
 Female     9951
Name: sex, dtype: int64
```

Applying groupby functions in order to summarise the data.

Groupby functions are usually used with aggregate functions, which are useful to summarise the dataset and make observations. Some common functions are SUM, MEAN, MAX, MIN and COUNT. Using groupby, we can answer questions such as:

Question: What is the average age of each gender in the given population?

In [24]:
```
data['age'].groupby([data['sex']]).mean()
```

Out[24]:
```
sex
 Female    36.851271
 Male      39.451693
Name: age, dtype: float64
```

This shows that average age of female is 36 and average age of male is 39 in the adult data.

Question. What is the average age of male and female across different eduction categories?

In [25]:
```
data['age'].groupby([data['sex'],data['education']]).mean()
```

Out[25]:
```
sex      education
 Female   10th            35.319703
          11th            30.348148
          12th            30.150376
          1st-4th         49.976190
          5th-6th         45.285714
          7th-8th         50.165563
          9th             41.789855
          Assoc-acdm      36.413265
          Assoc-voc       37.823276
          Bachelors       35.619906
          Doctorate       45.120482
          HS-grad         38.593172
          Masters         42.932515
          Preschool       42.266667
          Prof-school     40.716049
          Some-college    33.788454
 Male     10th            38.094595
          11th            33.331361
          12th            32.826923
          1st-4th         45.684685
          5th-6th         41.656388
          7th-8th         48.255034
          9th             40.492754
          Assoc-acdm      38.064189
          Assoc-voc       39.022416
          Bachelors       40.395604
          Doctorate       48.160000
          HS-grad         39.178997
          Masters         44.525253
          Preschool       42.322581
          Prof-school     45.584650
          Some-college    37.012582
Name: age, dtype: float64
```

In the above code, we group by 'sex' and 'education' and computed mean 'age' in the given population.

NOTE: groupby can be applied on numeric attributes only.

Question no 3

What is the average contribution to capital-gain of each sex and occupation category?

In [28]:
```
#Answer
data['capital-gain'].groupby([data['sex'],data['occupation']]).mean()
```

Out[28]:
```
sex      occupation
 Female   ?                     351.420716
          Adm-clerical          508.543497
          Craft-repair          807.793269
          Exec-managerial      1022.757263
          Farming-fishing      1293.019231
          Handlers-cleaners     151.421769
          Machine-op-inspct     149.511583
          Other-service         160.582691
          Priv-house-serv       302.651163
          Prof-specialty       1304.731568
          Protective-serv      1734.301370
          Sales                 281.543199
          Tech-support          658.773292
          Transport-moving      455.589744
 Male     ?                     877.041394
          Adm-clerical          480.800352
          Armed-Forces            0.000000
          Craft-repair          659.414846
          Exec-managerial      2778.056962
          Farming-fishing       504.397390
          Handlers-cleaners     286.047748
          Machine-op-inspct     397.674191
          Other-service         253.938672
          Priv-house-serv        74.250000
          Prof-specialty       3485.083850
          Protective-serv       606.676864
          Sales                1951.053906
          Tech-support          724.552876
          Transport-moving      494.525706
Name: capital-gain, dtype: float64
```

In the above code, we group by 'sex' and 'occupation' and computed mean 'capital-gain' in the given population

Question no 4

Identify the average capital-gain by males and females accross different marital-status.

In [30]:
```
#Answer
data['capital-gain'].groupby([data['sex'],data['marital-status']]).mean()
```

Out[30]:
```
sex      marital-status
 Female   Divorced                   454.577590
          Married-AF-spouse          204.076923
          Married-civ-spouse        1615.607662
          Married-spouse-absent      373.540404
          Never-married              335.807964
          Separated                  366.775891
          Widowed                    493.536137
 Male     Divorced                  1157.684535
          Married-AF-spouse          810.888889
          Married-civ-spouse        1791.060031
          Married-spouse-absent     1037.455026
          Never-married              434.198822
          Separated                  872.103825
          Widowed                    925.869281
Name: capital-gain, dtype: float64
```

In the above code, we group by 'sex' and 'marital-status' and computed mean 'capital-gain' in the given population

Question. What is the maximum age accross differnt races?

Let's first see what are the different races and then apply groupby.

In [32]:
```
data['race'].value_counts()
```

Out[32]:
```
 White                 25615
 Black                  2890
 Asian-Pac-Islander      961
 Amer-Indian-Eskimo      281
 Other                   253
Name: race, dtype: int64
```

In [33]:
```python
data['age'].groupby([data['race']]).max()
```

Out[33]:
```
race
 Amer-Indian-Eskimo    82
 Asian-Pac-Islander    90
 Black                 90
 Other                 77
 White                 90
Name: age, dtype: int64
```

It reflects that maximum adult of age 82 is amer-indian-eskimo Maximum age of Asian-Pac-islander in the data is 90 Maximum age of Black person in the data is 90 Maximum age of White person in the data is 90

Question no 5

Are minimum and maximum age by sex same?

Minimum age by sex

In [35]:
```python
data['age'].groupby([data['sex']]).min()
```

Out[35]:
```
sex
 Female    17
 Male      17
Name: age, dtype: int64
```

In [36]:
```python
data['age'].groupby([data['sex']]).max()
```

Out[36]:
```
sex
 Female    90
 Male      90
Name: age, dtype: int64
```

Yes, the minimum and maximum age by sex is same

Data Visualisation

Matplotlib is python library for visualising data in the form of graphs such as histograms, scatter, box plot, line plots, heat plots, etc.

In [37]:
```python
import matplotlib.pyplot as plt
%matplotlib inline
```

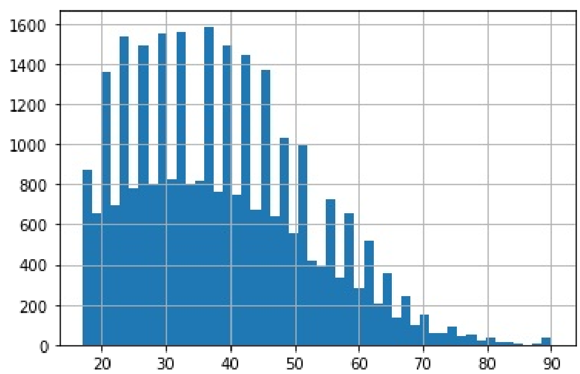
In [42]:
```python
data.describe()
```

Out[42]:

	age	education-num	capital-gain	capital-loss	hours-per-week
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.00000
mean	38.589133	10.080400	1086.949933	87.256033	40.43530
std	13.635182	2.572396	7459.713916	403.036258	12.34685
min	17.000000	1.000000	0.000000	0.000000	1.00000
25%	28.000000	9.000000	0.000000	0.000000	40.00000
50%	37.000000	10.000000	0.000000	0.000000	40.00000
75%	48.000000	12.000000	0.000000	0.000000	45.00000
max	90.000000	16.000000	99999.000000	4356.000000	99.00000

In [43]:
```python
data['age'].hist(bins=50)
```

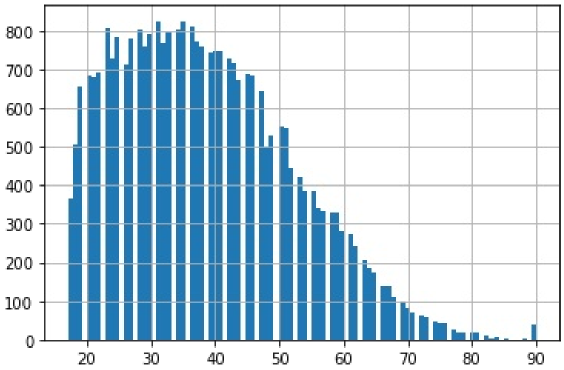
Out[43]:
```
<AxesSubplot:>
```



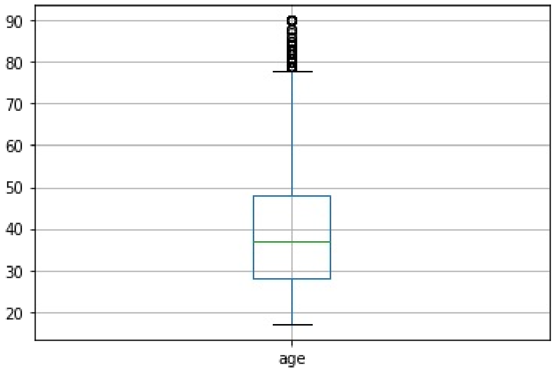
Histograms is used to represent the distribution of dataset. The bars of the histograms are known as bins or "bucket" - the range of

Histograms is used to represent the distribution of dataset. The bars of the histograms are known as bins or "bucket" – the range of values. Bins are of same width. Width of the bins can be calculated as (max value of data – min value of data) / total number of bins. The bins are usually specified as continuous, non-overlapping intervals of a variable.

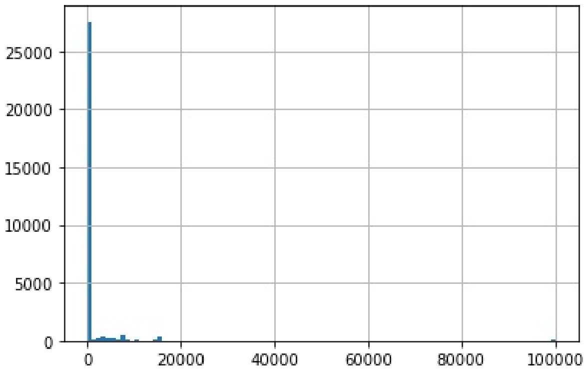
In the above figure, histogram with bins = 50 is used to show number of peolpe belongs to different age-groups. Here, x-axis represents 'age' and y-axis represents the 'count'. **Try-it-yourself:** change bins = 100 and run the cell to observe the difference for your own understanding.

In [46]:
```
data['age'].hist(bins=100)
```
Out[46]:
```
<AxesSubplot:>
```


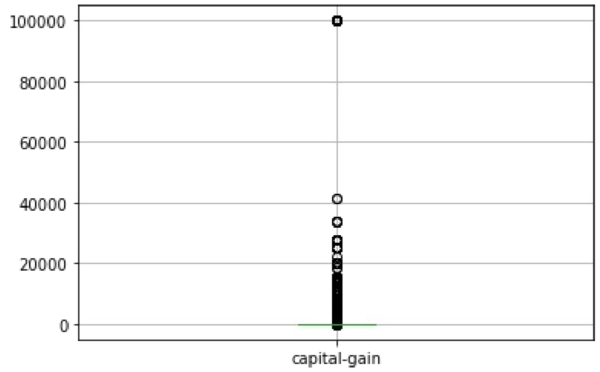
if we increase the bin size, the grouping is histogram is more clearly visibile

In [48]:
```
data.boxplot(column='age')
```
Out[48]:
```
<AxesSubplot:>
```


In the above figure, boxplot is used to find the average number of people belongs to which age-range group. The mean is around 38 age. and there are outliers after 78 age. the minimum age we can see from box plot is 17 and maxium age is 78. After 78 age there are outliers.

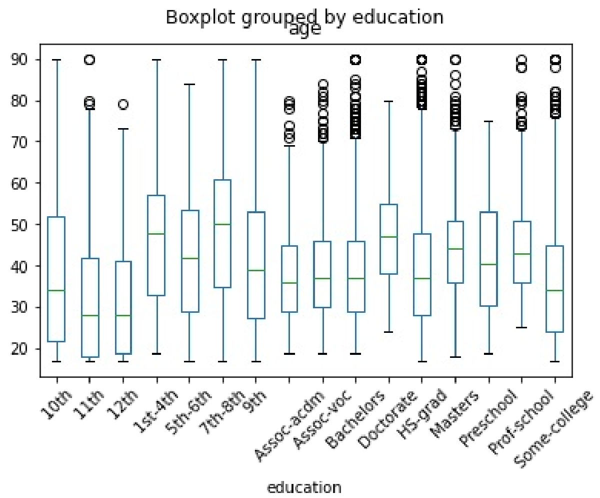
In [49]:
```
data['capital-gain'].hist(bins=100)
```
Out[49]:
```
<AxesSubplot:>
```


In [50]:
```
data.boxplot(column='capital-gain')
```
Out[50]:
```
<AxesSubplot:>
```



In [51]:
```
data.boxplot(column='age', by = 'education', grid=False, rot = 45, fontsize = 10)
```

Out[51]:
```
<AxesSubplot:title={'center':'age'}, xlabel='education'>
```

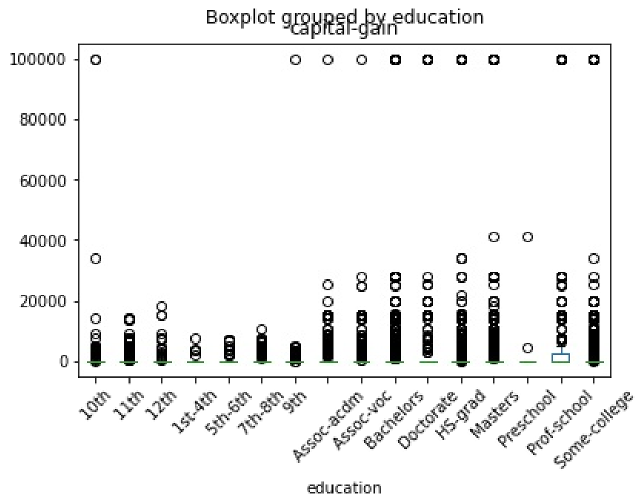


In [52]:
```
data['education'].value_counts()
```

Out[52]:
```
 HS-grad         9676
 Some-college    6714
 Bachelors       4955
 Masters         1578
 Assoc-voc       1267
 11th            1081
 Assoc-acdm       984
 10th             861
 7th-8th          598
 Prof-school      524
 9th              483
 12th             393
 Doctorate        383
 5th-6th          304
 1st-4th          153
 Preschool         46
Name: education, dtype: int64
```

In [53]:
```
data.boxplot(column='capital-gain', by = 'education', grid=False, rot = 45, fontsize = 10)
```

Out[53]:
```
<AxesSubplot:title={'center':'capital-gain'}, xlabel='education'>
```



After performing some basic data analysis, let's look at data pre-processing to improve the quality of the dataset.

Data pre-processing is an important step in the process. Raw data can be unstructured and full of noise. Aim of this phase is to clean the raw data, reduce noise and to prepare the dataset that can be accepted by the algorithm as an input. Remember garbage in, garbage out!

In [54]:
```
data['marital-status'].value_counts()
```

Out[54]:
```
 Married-civ-spouse       13791
 Never-married             9827
 Divorced                  4104
 Separated                  955
 Widowed                    914
 Married-spouse-absent      387
 Married-AF-spouse           22
Name: marital-status, dtype: int64
```

Checking NULL values in the dataset

In [55]:
```
data.apply(lambda x: sum(x.isnull()), axis = 0)
```

Out[55]:
```
age               0
workclass         0
education         0
education-num     0
marital-status    0
occupation        0
relationship      0
race              0
sex               0
capital-gain      0
capital-loss      0
hours-per-week    0
native-country    0
class-label       0
dtype: int64
```

As the missing values in this data is already replaces by ?.

Data Transformation

Label encoding:

Some attributes are categorical, therefore (statistical) analysis on those variables is not possible. We need to convert all categorical variables (string labels) into numeric by encoding the categories. Package 'sklearn' provides 'LabelEncoder' library for encoding labels between 0 to n-1 discrete values/labels, where n is the number of values/labels. E.g.: Male -> 0 Female -> 1

In [56]:
```
from sklearn.preprocessing import LabelEncoder
```

In [57]:
```
data.head()
```

Out[57]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	class-label
31113	28	Private	Assoc-voc	11	Married-civ-spouse	Prof-specialty	Wife	White	Female	0	0	5	United-States	<=50K
12788	24	State-gov	Doctorate	16	Never-married	Prof-specialty	Not-in-family	White	Female	0	0	99	England	<=50K
27524	38	Private	HS-grad	9	Separated	Sales	Not-in-family	White	Male	0	0	60	United-States	<=50K
30497	39	Self-emp-not-inc	10th	6	Married-spouse-absent	Other-service	Not-in-family	White	Female	0	1721	15	United-States	<=50K
9118	23	Private	10th	6	Never-married	Handlers-cleaners	Other-relative	Other	Male	0	0	40	United-States	<=50K

In [58]:
```
data.dtypes
```

Out[58]:
```
age                int64
workclass         object
education         object
education-num      int64
marital-status    object
occupation        object
relationship      object
race              object
sex               object
capital-gain       int64
capital-loss       int64
hours-per-week     int64
native-country    object
class-label       object
dtype: object
```

In [59]:
```
columns = list(data.select_dtypes(exclude=['int64']))
```

As we do not need to convert the integers they are already in numbers. So, I drop all the integer columns in the data.

In [60]:
```
columns
```

Out[60]:
```
['workclass',
 'education',
 'marital-status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native-country',
 'class-label']
```

In [61]:
```
data['class-label'].value_counts()
```

Out[61]:
```
 <=50K    22768
 >50K      7232
Name: class-label, dtype: int64
```

In [108…
```
le = LabelEncoder()
for i in columns:
    #print(i)
    data[i] = le.fit_transform(data[i])
data.dtypes
```

Out[108]:
```
age               int64
workclass         int64
education         int64
education-num     int64
marital-status    int64
occupation        int64
relationship      int64
race              int64
sex               int64
capital-gain      int64
capital-loss      int64
hours-per-week    int64
native-country    int64
class-label       int64
dtype: object
```

In [109…
```
data.head()
```

Out[109]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	class-label
31113	28	4	14	11	2	2	5	4	0	0	0	5	33	0
12788	24	7	2	16	4	2	1	4	0	0	0	99	41	0
27524	38	4	3	9	5	4	1	4	1	0	0	60	33	0
30497	39	6	0	6	3	13	1	4	0	0	1721	15	33	0
9118	23	4	0	6	4	11	2	3	1	0	0	40	33	0

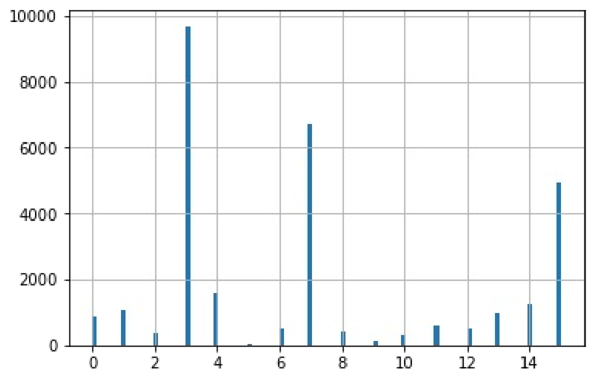
In [110…
```
data['workclass'].value_counts()
```

Out[110]:
```
4    20927
6     2336
2     1925
0     1694
7     1191
5     1020
1      888
8       13
3        6
Name: workclass, dtype: int64
```

You will notice that all the values are now numeric. Now, more computations and analysis can be performed on the dataset.

In [111…
```
data['education'].hist(bins=100)
```

Out[111]:
```
<AxesSubplot:>
```



In [112…
```
data.describe(include='all')
```

Out[112]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	ca
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	300
mean	38.589133	3.867367	7.078300	10.080400	2.611067	6.341533	1.445167	3.665333	0.668300	10
std	13.635182	1.455648	4.831344	2.572396	1.507229	4.263875	1.605411	0.848502	0.470832	74
min	17.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	28.000000	4.000000	3.000000	9.000000	2.000000	2.000000	0.000000	4.000000	0.000000	
50%	37.000000	4.000000	7.000000	10.000000	2.000000	6.000000	1.000000	4.000000	1.000000	
75%	48.000000	4.000000	12.000000	12.000000	4.000000	9.000000	3.000000	4.000000	1.000000	
max	90.000000	8.000000	15.000000	16.000000	6.000000	14.000000	5.000000	4.000000	1.000000	999

Report

Question no 6

Write a summary of the outcome of data.describe()

Answer When we describe the data, it will give the total number of values in each column. It also give the mean of each variable of the data. Like in **age** the mean is 38. It represents that avarage age is 38 in the data. The describe function also gives the maximum and minimum value and standard deviation of each variable. The minimum age is 17 and maximum is 90. it also gives 25% percentile, 50 percenttile and 75 percentile of the data.

Question no 7

What are the different data types (or attribut types) in data mining? Explain with the help of the examples from Adult dataset. HINT: Don't get confused with data types in Python or Pandas.

Answer There are mainly two attributes in the data mining

- Quantitative attribute such as discrete and conitnuous attribute
- Qualitative attribute such as oridnal, nominal and binary attributes

Question no 8

In [96]:
```
data1 = pd.read_csv('adult.csv')
```

Highest migrants belongs to which country?

In [97]:
```
data1['native-country'].value_counts()
```

Out[97]:
```
 United-States                 29170
 Mexico                          643
 ?                               583
 Philippines                     198
 Germany                         137
 Canada                          121
 Puerto-Rico                     114
 El-Salvador                     106
 India                           100
 Cuba                             95
 England                          90
 Jamaica                          81
 South                            80
 China                            75
 Italy                            73
 Dominican-Republic               70
 Vietnam                          67
 Guatemala                        64
 Japan                            62
 Poland                           60
 Columbia                         59
 Taiwan                           51
 Haiti                            44
 Iran                             43
 Portugal                         37
 Nicaragua                        34
 Peru                             31
 France                           29
 Greece                           29
 Ecuador                          28
 Ireland                          24
 Hong                             20
 Cambodia                         19
 Trinadad&Tobago                  19
 Laos                             18
 Thailand                         18
 Yugoslavia                       16
 Outlying-US(Guam-USVI-etc)       14
 Honduras                         13
 Hungary                          13
 Scotland                         12
 Holand-Netherlands                1
Name: native-country, dtype: int64
```

39 is assigned to United States. Most adults are from United States in the data.

Question no 9

Which occupation represents more males than females?

In [103…
```
data1['sex'].groupby(data1['occupation']).value_counts()
```

Out[103]:
```
occupation          sex   
 ?                   Male      1002
                     Female     841
 Adm-clerical        Female    2537
                     Male      1233
 Armed-Forces        Male         9
 Craft-repair        Male      3877
                     Female     222
 Exec-managerial     Male      2907
                     Female    1159
 Farming-fishing     Male       929
                     Female      65
 Handlers-cleaners   Male      1206
                     Female     164
 Machine-op-inspct   Male      1452
                     Female     550
 Other-service       Female    1800
                     Male      1495
 Priv-house-serv     Female     141
                     Male         8
 Prof-specialty      Male      2625
                     Female    1515
 Protective-serv     Male       573
                     Female      76
 Sales               Male      2387
                     Female    1263
 Tech-support        Male       580
                     Female     348
 Transport-moving    Male      1507
                     Female      90
Name: sex, dtype: int64
```

Almost all the occupation has more males than females, except adm-clerical, other service and pric-house-serv.

Question no 10

What is the difference between data.head() and data.tail()?

Answer Data.head shows the first 5 rows of the dataframe However, Data,tail() shows that last 5 rows of the data set

In [105…
```
data1.head()
```

Out[105]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	clas lab
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50

In [106…
```
data1.tail()
```

Out[106]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country
32556	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White	Female	0	0	38	United-States
32557	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0	0	40	United-States
32558	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Female	0	0	40	United-States
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Male	0	0	20	United-States
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Female	15024	0	40	United-States

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js