In [1]:

```
# Github: http://shaykhsiddique.me
import numpy as np
import matplotlib.pyplot as plt
import math
%matplotlib inline
```

In [2]:

```
# define a suitable object controller
class ContinuousDatasets:
    def __init__(self, ages, fnlwgt, education_num, capital_gain, capital_loss, hours_per_w
        #constructor for ContinuousDatasets
        #ensure that the list of data is always numeric using map
        self.ages = list(map(int, ages))
        self.fnlwgt = list(map(int, fnlwgt))
        self.education_num = list(map(int, education_num))
        self.capital_gain = list(map(int, capital_gain))
        self.capital_loss = list(map(int, capital_loss))
        self.hours_per_week = list(map(int, hours_per_week))
    def getAges(self):
        return self.ages
    def getFnlwgt(self):
        return self.fnlwgt
    def getEducation_num(self):
        return self.education_num
    def getCapital_gain(self):
        return self.capital_gain
    def getCapital_loss(self):
        return self.capital_loss
    def getHours_per_week(self):
        return self.hours_per_week
```

In [3]:

```
datafile = np.loadtxt('adult.data', dtype='str', delimiter=',')
# print a record to check file loading is sucessful or not
print(datafile[0])
```

```
['39' ' State-gov' ' 77516' ' Bachelors' ' 13' ' Never-married'
 ' Adm-clerical' ' Not-in-family' ' White' ' Male' ' 2174' ' 0' ' 40'
 ' United-States' ' <=50K']
```

In [4]:

```
# load data into object to make calculation easier
all_objs = ContinuousDatasets(datafile[:,0], datafile[:,2], datafile[:,4], datafile[:,10],
data_len = len(all_objs.getAges())
print("Number of Record: "+str(data_len))
```

```
Number of Record: 32561
```

In [5]:

```
# calculating Average
def avg_calculation(data):
    age_sum = 0;
    for val in data:
        age_sum+=val
    data=False
    return age_sum/(data_len)


# for all feature call the avg_calculation function
avg_age = avg_calculation(all_objs.getAges())
avg_fnlwgt = avg_calculation(all_objs.getFnlwgt())
avg_education_num = avg_calculation(all_objs.getEducation_num())
avg_capital_gain = avg_calculation(all_objs.getCapital_gain())
avg_capital_loss = avg_calculation(all_objs.getCapital_loss())
avg_hours_per_week = avg_calculation(all_objs.getHours_per_week())
# print all values
print("Average:- ")
print("\tAge: "+str(avg_age))
print("\tFnlwgt: "+str(avg_fnlwgt))
print("\tEducation Num: "+str(avg_education_num))
print("\tCapital Gain: "+str(avg_capital_gain))
print("\tCapital Loss: "+str(avg_capital_loss))
print("\tHours-per-week: "+str(avg_hours_per_week))
```

```
Average:-
        Age: 38.58164675532078
        Fnlwgt: 189778.36651208502
        Education Num: 10.0806793403151
        Capital Gain: 1077.6488437087312
        Capital Loss: 87.303829734959
        Hours-per-week: 40.437455852092995
```

In [6]:

```python
# calculating Max, Min, Median
def calculating_medianMaxMin(feature, data):
    data.sort()
    med_indx = int(data_len/2)+1
    print(feature+":-")
    print("\tMedian " + ": "+ str(data[med_indx]))
    print("\tMaximum " + ": "+ str(max(data)))
    print("\tMinimum " + ": "+ str(min(data))+"\n")

# Call functions- max, min, median
calculating_medianMaxMin("Age", all_objs.getAges())
calculating_medianMaxMin("Fnlwgt", all_objs.getFnlwgt())
calculating_medianMaxMin("Education Num", all_objs.getEducation_num())
calculating_medianMaxMin("Capital Gain", all_objs.getCapital_gain())
calculating_medianMaxMin("Capital Loss", all_objs.getCapital_loss())
calculating_medianMaxMin("Hours-per-week", all_objs.getHours_per_week())
```

```
Age:-
        Median : 37
        Maximum : 90
        Minimum : 17

Fnlwgt:-
        Median : 178370
        Maximum : 1484705
        Minimum : 12285

Education Num:-
        Median : 10
        Maximum : 16
        Minimum : 1

Capital Gain:-
        Median : 0
        Maximum : 99999
        Minimum : 0

Capital Loss:-
        Median : 0
        Maximum : 4356
        Minimum : 0

Hours-per-week:-
        Median : 40
        Maximum : 99
        Minimum : 1
```

In [7]:

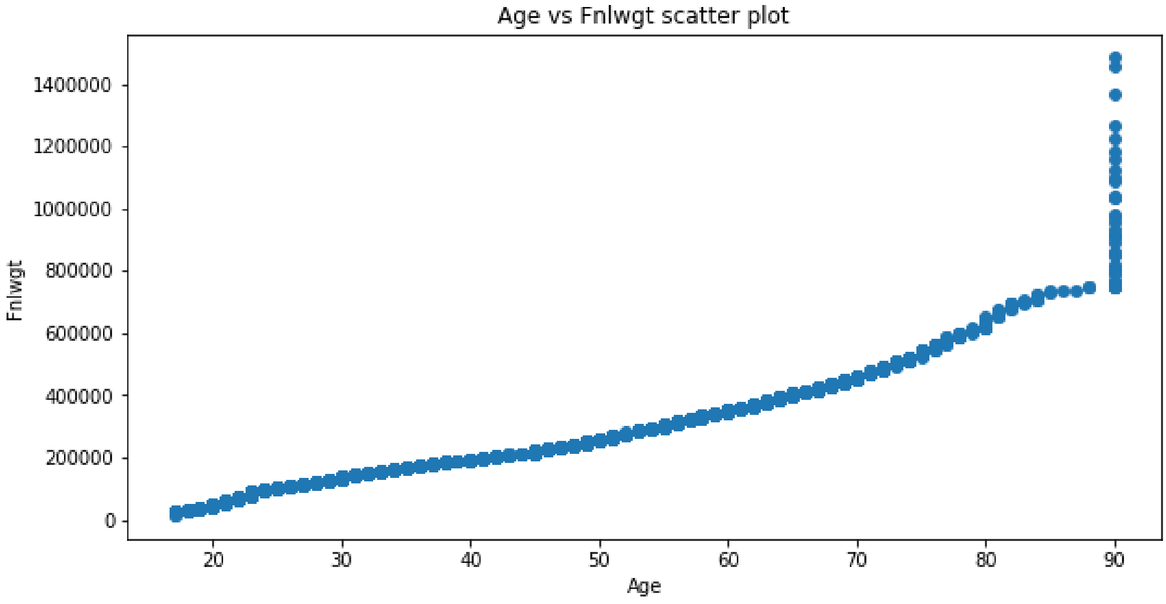
```
# calculating standard deviation sqrt(sum(v[i]*v[i])/n) where i=0 to n-1
def std_dev(data, avg):
    total_val = 0
    for val in data:
        temp = val-avg
        total_val += (temp*temp)
    return math.sqrt(total_val/data_len)


# calling the standard deviation function for each continuous features
print("standard deviation:-")
print("\tAge: "+str(std_dev(all_objs.getAges(), avg_age)))
print("\tFnlwgt: "+str(std_dev(all_objs.getFnlwgt(), avg_fnlwgt)))
print("\tEducation Num: "+str(std_dev(all_objs.getEducation_num(), avg_education_num)))
print("\tCapital Gain: "+str(std_dev(all_objs.getCapital_gain(), avg_capital_gain)))
print("\tCapital Loss: "+str(std_dev(all_objs.getCapital_loss(), avg_capital_loss)))
print("\tHours-per-week: "+str(std_dev(all_objs.getHours_per_week(), avg_hours_per_week)))
```

```
standard deviation:-
        Age: 13.640223092303957
        Fnlwgt: 105548.35688089058
        Education Num: 2.5726808256012275
        Capital Gain: 7385.178676947581
        Capital Loss: 402.9540308274799
        Hours-per-week: 12.347239075705962
```

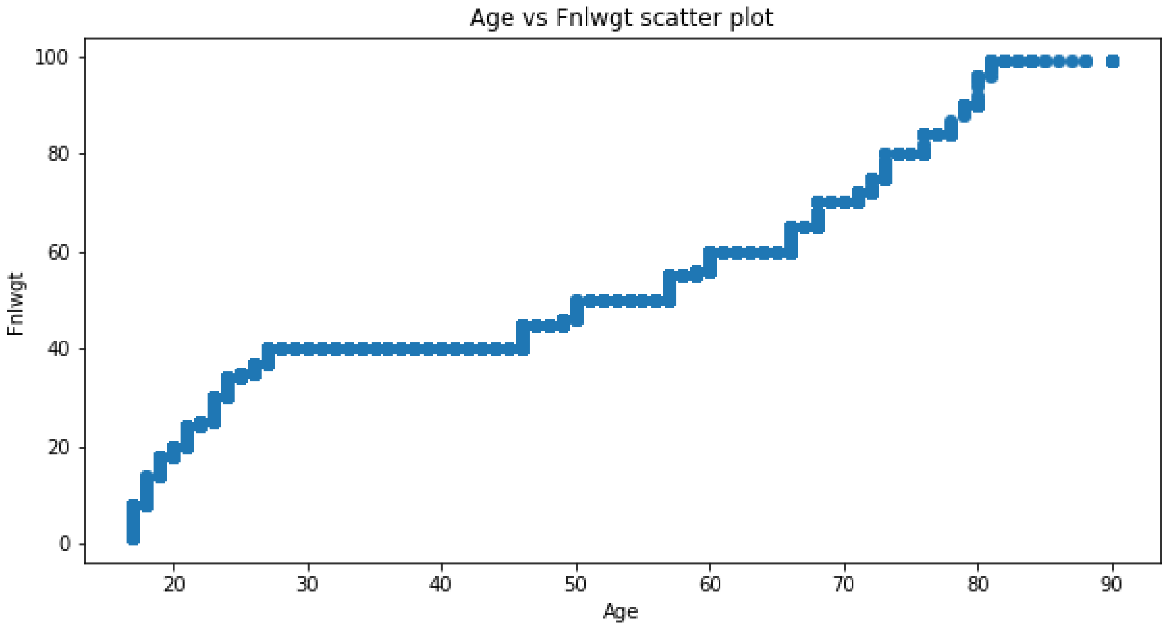
In [19]:

```
# scatter plot - 1 age vs Fnlwgt
plt.figure(figsize=(10,5))
plt.scatter(all_objs.getAges(), all_objs.getFnlwgt())
plt.title("Age vs Fnlwgt scatter plot")
plt.xlabel("Age")
plt.ylabel("Fnlwgt")
plt.show()
```



In [21]:

```
# scatter plot - 2 age vs Hours per week
plt.figure(figsize=(10,5))
plt.scatter(all_objs.getAges(), all_objs.getHours_per_week())
plt.title("Age vs Fnlwgt scatter plot")
plt.xlabel("Age")
plt.ylabel("Fnlwgt")
plt.show()
```



In [9]:

```
# function for numeric dissimilarity calculation
def numericDissimilarity(value1, value2, max_val, min_val):
    upper_val = abs(value1-value2)
    lower_val = max_val-min_val
    return(upper_val/lower_val)

#function for nominal dissimilarity calculation as well as it work's for binary
def nominalDissimilarity(data1, data2):
    if(data1==data2):
        return 0
    else:
        return 1
```

In [14]:

```
# calculating dissimilarity
def calculate_dissimilarity(record1, record2, numOfFeature):
    all_dist =0
#     call for ages and the index is the column number
    all_dist += numericDissimilarity(int(record1[0]), int(record2[0]), max(all_objs.getAges
#     call of workclass nominal data
    all_dist += int(nominalDissimilarity(str(record1[1]), str(record2[1])))
#     call of Fnlwg
    all_dist += numericDissimilarity(int(record1[2]), int(record2[2]), max(all_objs.getFnlw
#     call of education
    all_dist += int(nominalDissimilarity(str(record1[3]), str(record2[3])))
#     call of education-num
    all_dist += numericDissimilarity(int(record1[4]), int(record2[4]), max(all_objs.getEduc
#     marital-status
    all_dist += int(nominalDissimilarity(str(record1[5]), str(record2[5])))
#     occupation
    all_dist += int(nominalDissimilarity(str(record1[6]), str(record2[6])))
#     relationship
    all_dist += int(nominalDissimilarity(str(record1[7]), str(record2[7])))
#     race
    all_dist += int(nominalDissimilarity(str(record1[8]), str(record2[8])))
#     sex
    all_dist += int(nominalDissimilarity(str(record1[9]), str(record2[9])))
#     capital gain
    all_dist += numericDissimilarity(int(record1[10]), int(record2[10]), max(all_objs.getCa
#     capital loss
    all_dist += numericDissimilarity(int(record1[11]), int(record2[11]), max(all_objs.getCa
#     hours-per-week
    all_dist += numericDissimilarity(int(record1[12]), int(record2[12]), max(all_objs.getCa
#     native-country
    all_dist += int(nominalDissimilarity(str(record1[13]), str(record2[13])))
#     salary
    all_dist += int(nominalDissimilarity(str(record1[14]), str(record2[14])))
    return (all_dist/numOfFeature)


print("Record1: "+str(datafile[0]))
print("Record2: "+str(datafile[53]))
print("\n\nDissimilarity: "+str(calculate_dissimilarity(datafile[0], datafile[53], 15)))
```

```
Record1: ['39' ' State-gov' ' 77516' ' Bachelors' ' 13' ' Never-married'
 ' Adm-clerical' ' Not-in-family' ' White' ' Male' ' 2174' ' 0' ' 40'
 ' United-States' ' <=50K']
Record2: ['50' ' Federal-gov' ' 251585' ' Bachelors' ' 13' ' Divorced'
 ' Exec-managerial' ' Not-in-family' ' White' ' Male' ' 0' ' 0' ' 55'
 ' United-States' ' >50K']


Dissimilarity: 0.2862725560442913
```