Exploring data collections

Descriptive statistics

Genoveva Vargas-Solar
French Council of Scientific Research, LIG
genoveva.vargas@imag.fr

http://vargas-solar.com/data-centric-smart-everything/

Thanks to Prof. J.L. Zechinelli Martini, UDLAP-LAFMIA, Mexico
for our collaborative construction of these slides

* This presentation was created using the content of https://www.python.org/

DESCRIPTIVE STATISTICS

Helps to simplify large amounts of data in a sensible way

- A simple way to describe the data
- Presenting quantitative descriptions in a manageable form

Main steps

- **Data preparation**: generate statistically valid descriptions
- **Descriptive statistics**: Generate different statistics to
  - Describe and summarize the data concisely
  - Evaluate different ways to visualize them

PREPARING DATA

PREPARING DATA

Obtaining the data: Read from a file or obtained by scraping the web

Parsing the data: Format the data which can be in plain text, fixed columns, CSV, XML, HTML, etc.

Cleaning the data: A simple strategy is to remove or ignore incomplete records

Building data structures: A data structure that lends itself to the analysis we are interested in.

Databases provide a mapping from keys to values, so they serve as dictionaries

ANALYSING INCOME ACCORDING TO GENDER

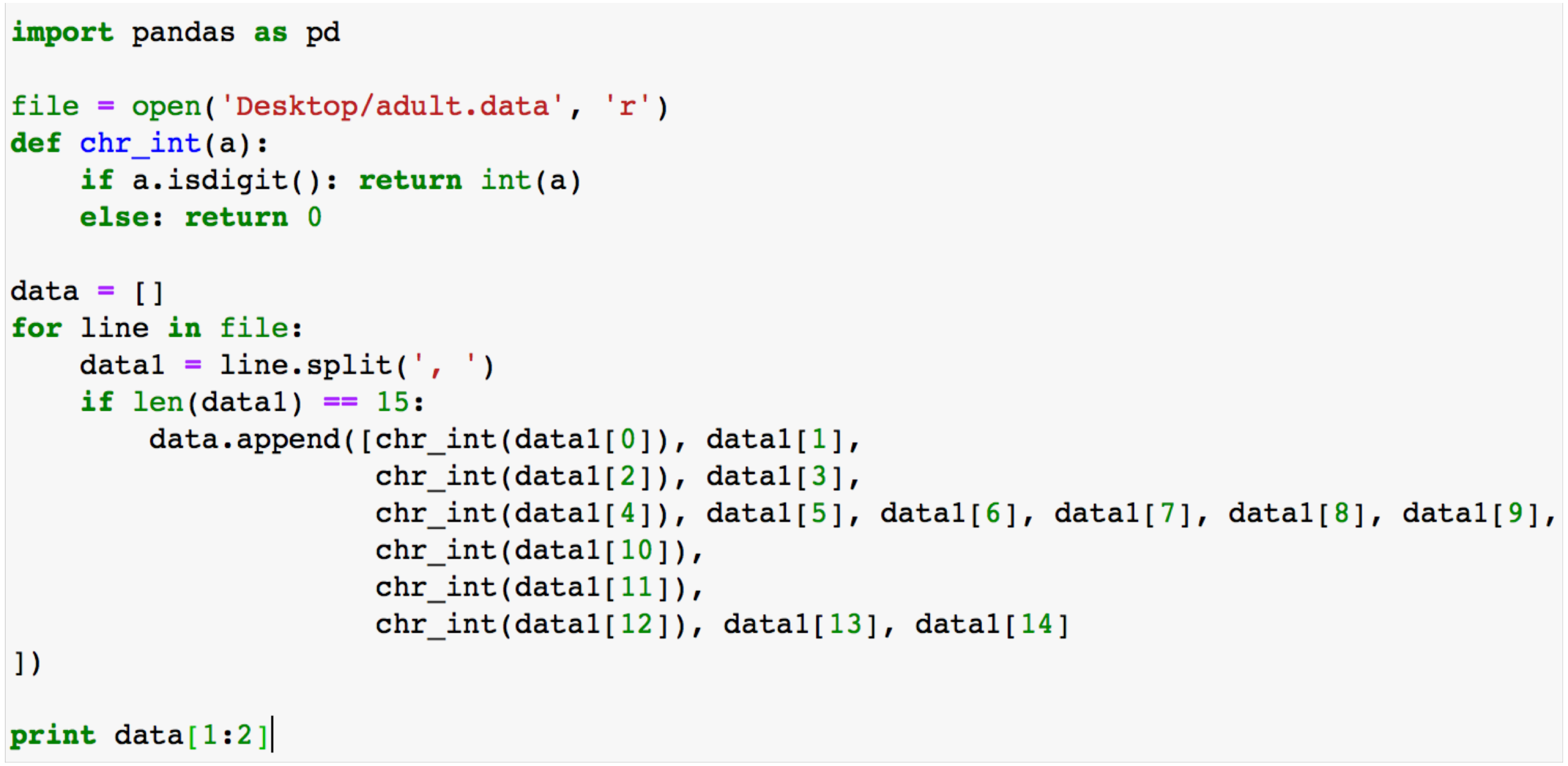
Financial parameters related to the US population*

- **Features**: Age, sex, marital, country, income, education, occupation, capital gain, etc.
- **Question**: Are men more likely to become high-income professionals than women, i.e., to receive an income of over $50,000 per year?

- Preparing data collections
  - Read and check the data
  - Represent the data, for instance using a tabular data structure with features (columns) and records (rows)
  - Group the data

* UCI's Machine Learning Repository : https://archive.ics.uci.edu/ml/datasets/Adult

READ & CHECK DATA



```
import pandas as pd

file = open('Desktop/adult.data', 'r')
def chr_int(a):
    if a.isdigit(): return int(a)
    else: return 0

data = []
for line in file:
    data1 = line.split(', ')
    if len(data1) == 15:
        data.append([chr_int(data1[0]), data1[1],
                     chr_int(data1[2]), data1[3],
                     chr_int(data1[4]), data1[5], data1[6], data1[7], data1[8], data1[9],
                     chr_int(data1[10]),
                     chr_int(data1[11]),
                     chr_int(data1[12]), data1[13], data1[14]
])

print data[1:2]
```

REPRESENT & GROUP DATA

```
df = pd.DataFrame(data)
df.columns = [ 'age', 'type_employer', 'fnlwgt', 'education', 'education_num',
               'marital', 'occupation', 'relationship', 'race', 'sex',
               'capital_gain', 'capital_loss', 'hr_per_week', 'country', 'income' ]
df.shape
```

```
counts = df.groupby('country').size()
print counts.head()

ml = df[(df.sex == 'Male')]
ml1 = df[(df.sex == 'Male') & (df.income=='>50K\n') ]

fm = df[(df.sex == 'Female')]
fm1 = df[(df.sex == 'Female') & (df.income=='>50K\n')]

ml1.describe()
```

EXPLORATORY DATA ANALYSIS

Measurements and categories represent a sample distribution of a variable:

- which approximately represents the population distribution of the variable
- to make tentative assumptions about the population distribution

Different *techniques*:

- **Summarizing the data**
- **Data distributions**
- **Outlier treatment**
- **Kernel density**

SUMMARIZING DATA

SUMMARIZING DATA

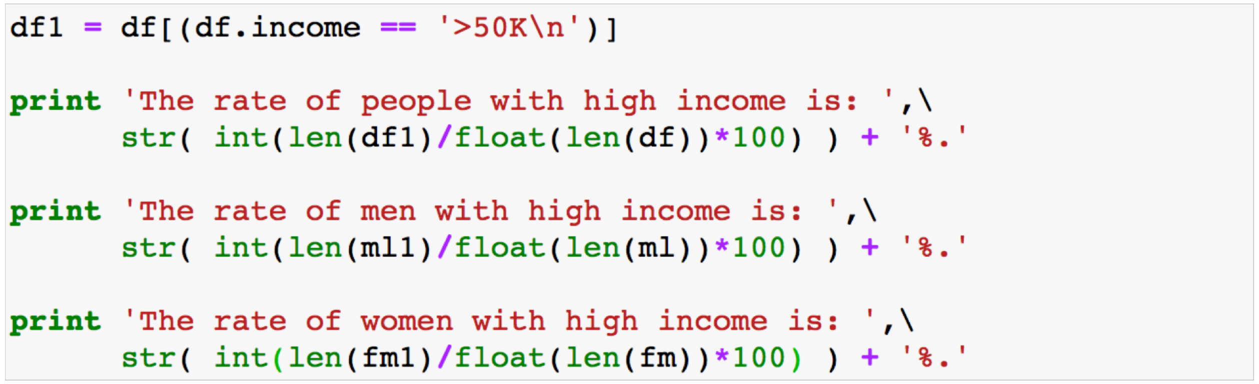
Categorical data:

- A simple tabulation of the frequency of each category is the best non-graphical exploration for data analysis
- For example, we can ask ourselves what is the proportion of high-income professionals in our database

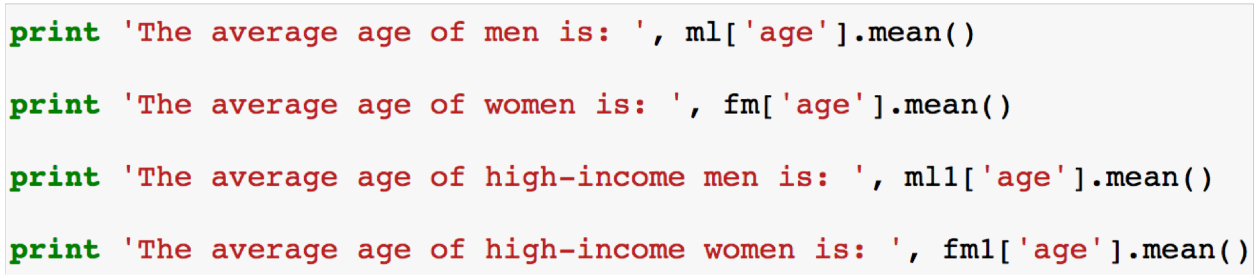
A quantitative variable:

- The characteristics of the population distribution of a quantitative variable are its mean, deviation, histograms, outliers, etc.
- Exploratory data analysis is a way to make preliminary assessments about the population distribution

SUMMARIZING DATA

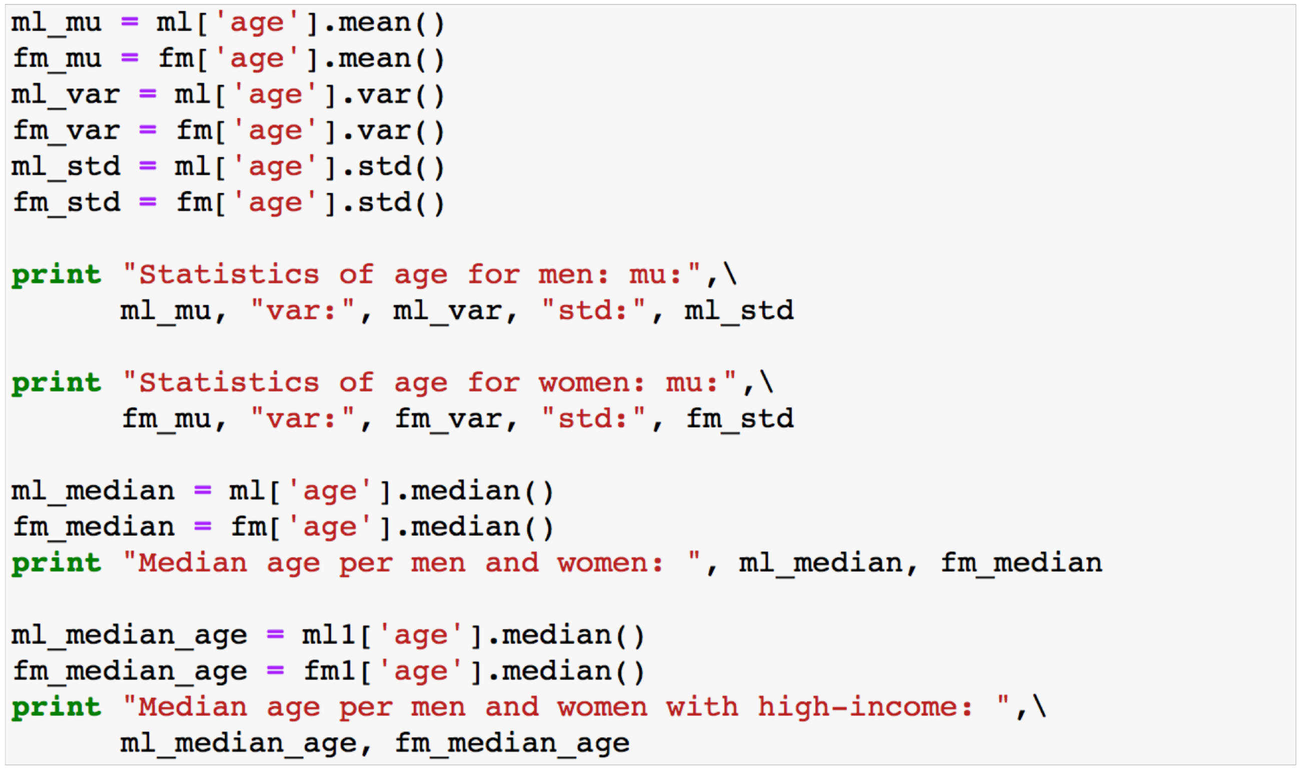


```
df1 = df[(df.income == '>50K\n')]

print 'The rate of people with high income is: ',\
      str( int(len(df1)/float(len(df))*100) ) + '%.'

print 'The rate of men with high income is: ',\
      str( int(len(ml1)/float(len(ml))*100) ) + '%.'

print 'The rate of women with high income is: ',\
      str( int(len(fm1)/float(len(fm))*100) ) + '%.'
```



```
print 'The average age of men is: ', ml['age'].mean()

print 'The average age of women is: ', fm['age'].mean()

print 'The average age of high-income men is: ', ml1['age'].mean()

print 'The average age of high-income women is: ', fm1['age'].mean()
```

SUMMARIZING DATA



```
ml_mu = ml['age'].mean()
fm_mu = fm['age'].mean()
ml_var = ml['age'].var()
fm_var = fm['age'].var()
ml_std = ml['age'].std()
fm_std = fm['age'].std()

print "Statistics of age for men: mu:",\
      ml_mu, "var:", ml_var, "std:", ml_std

print "Statistics of age for women: mu:",\
      fm_mu, "var:", fm_var, "std:", fm_std

ml_median = ml['age'].median()
fm_median = fm['age'].median()
print "Median age per men and women: ", ml_median, fm_median

ml_median_age = ml1['age'].median()
fm_median_age = fm1['age'].median()
print "Median age per men and women with high-income: ",\
      ml_median_age, fm_median_age
```

DATA DISTRIBUTION

DATA DISTRIBUTIONS

Summarizing data by just looking at their mean, median, and variance can be dangerous

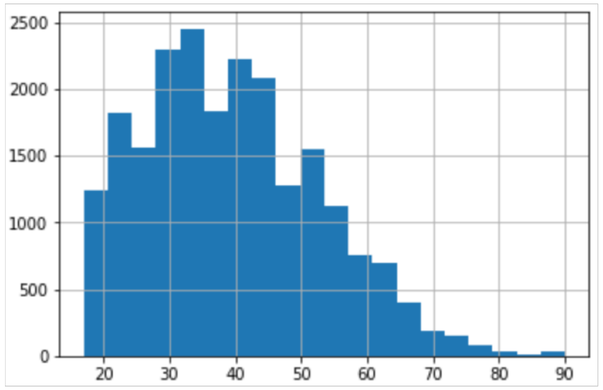
- Validate the data by inspecting them
- Very different data can be described by the same statistics

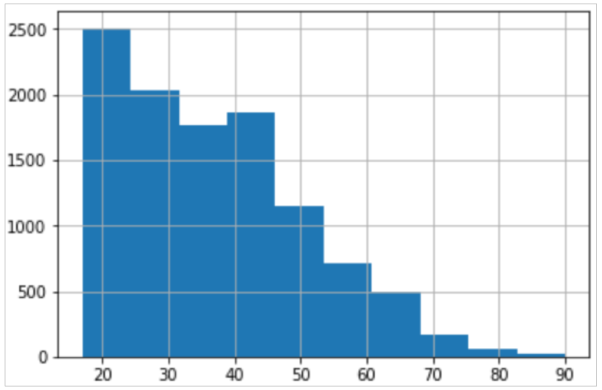
→ An histogram is a graph that shows the frequency of each value

DATA DISTRIBUTIONS

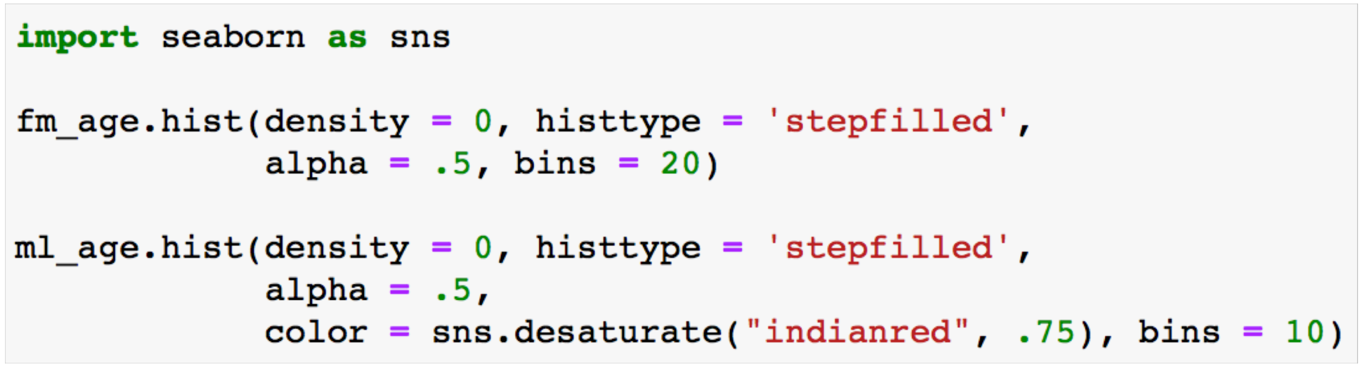
```
ml_age = ml['age']
ml_age.hist(density = 0, histtype = 'stepfilled', bins = 20)
```

```
fm_age = fm['age']
fm_age.hist(density = 0, histtype = 'stepfilled', bins = 10)
```

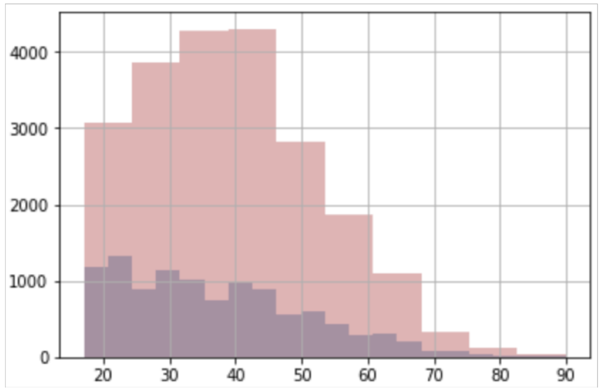




DATA DISTRIBUTIONS

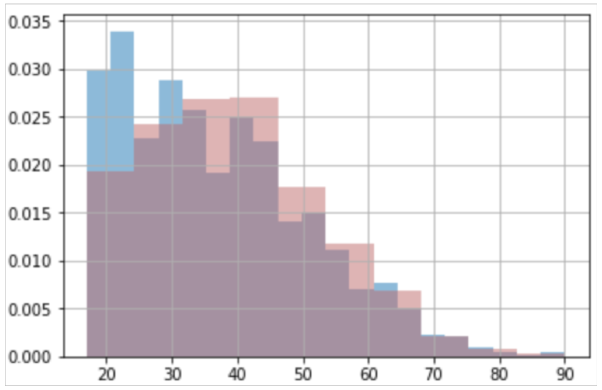


```
import seaborn as sns

fm_age.hist(density = 0, histtype = 'stepfilled',
            alpha = .5, bins = 20)

ml_age.hist(density = 0, histtype = 'stepfilled',
            alpha = .5,
            color = sns.desaturate("indianred", .75), bins = 10)
```

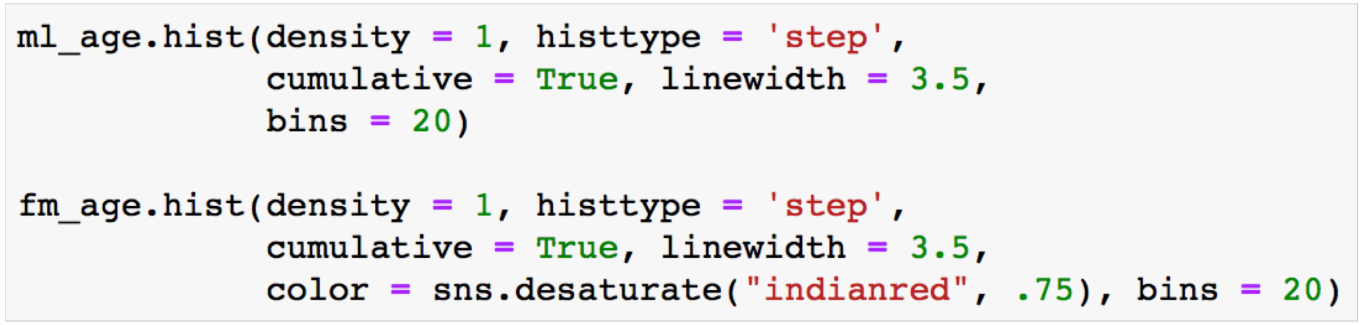


DATA DISTRIBUTIONS

```
fm_age.hist(density = 1, histtype = 'stepfilled',
            alpha = .5, bins = 20)

ml_age.hist(density = 1, histtype = 'stepfilled',
            alpha = .5,
            color = sns.desaturate("indianred", .75), bins = 10)
```

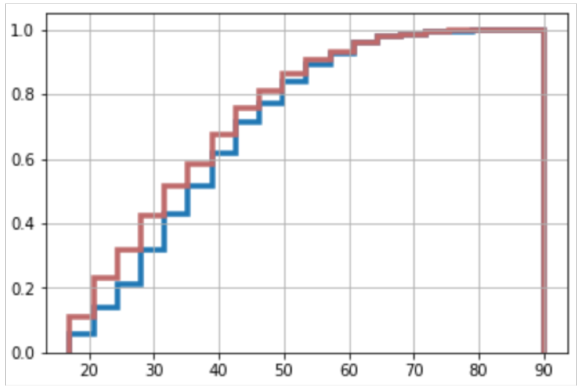


DATA DISTRIBUTIONS



```
ml_age.hist(density = 1, histtype = 'step',
            cumulative = True, linewidth = 3.5,
            bins = 20)

fm_age.hist(density = 1, histtype = 'step',
            cumulative = True, linewidth = 3.5,
            color = sns.desaturate("indianred", .75), bins = 20)
```

- The Cumulative Distribution Function (CDF) describes the probability that a real-valued random variable X with a given probability distribution will be found to have a value less than or equal to x



OUTLIERS

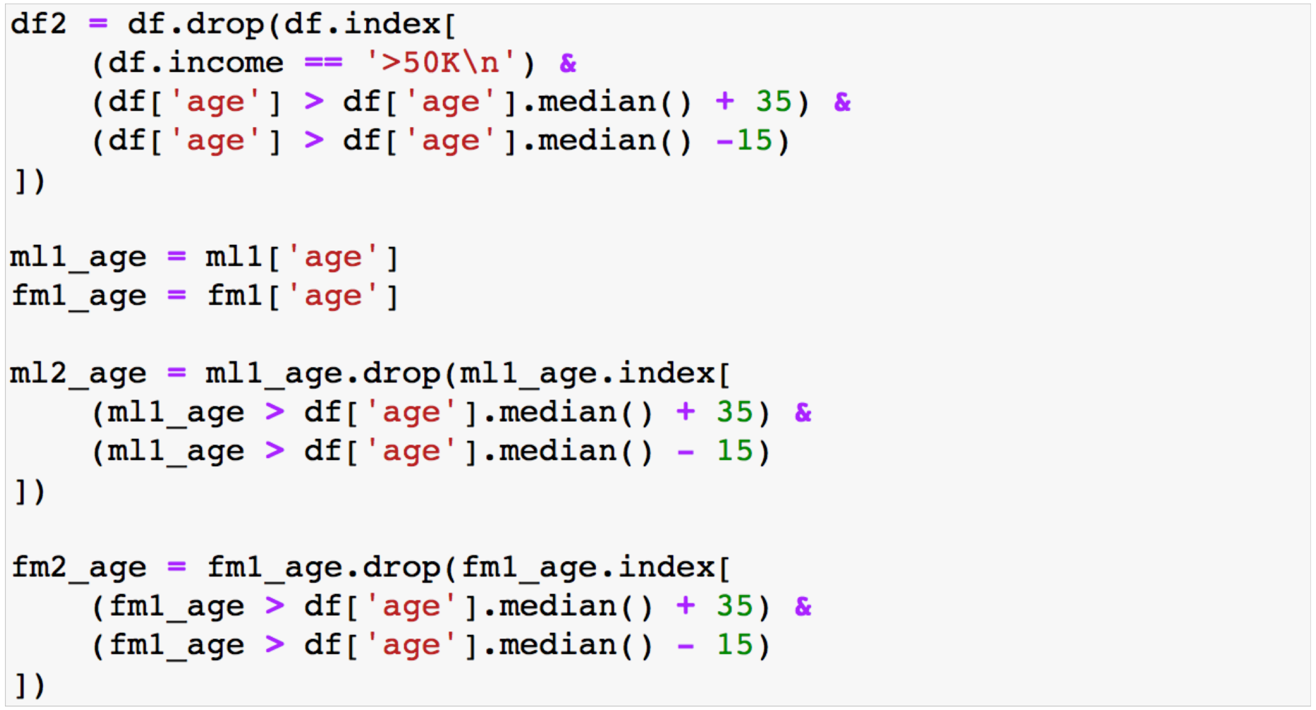
OUTLIER TREATMENT

Outliers are data samples with a value that is far from the central tendency

Different rules can be defined to detect outliers:

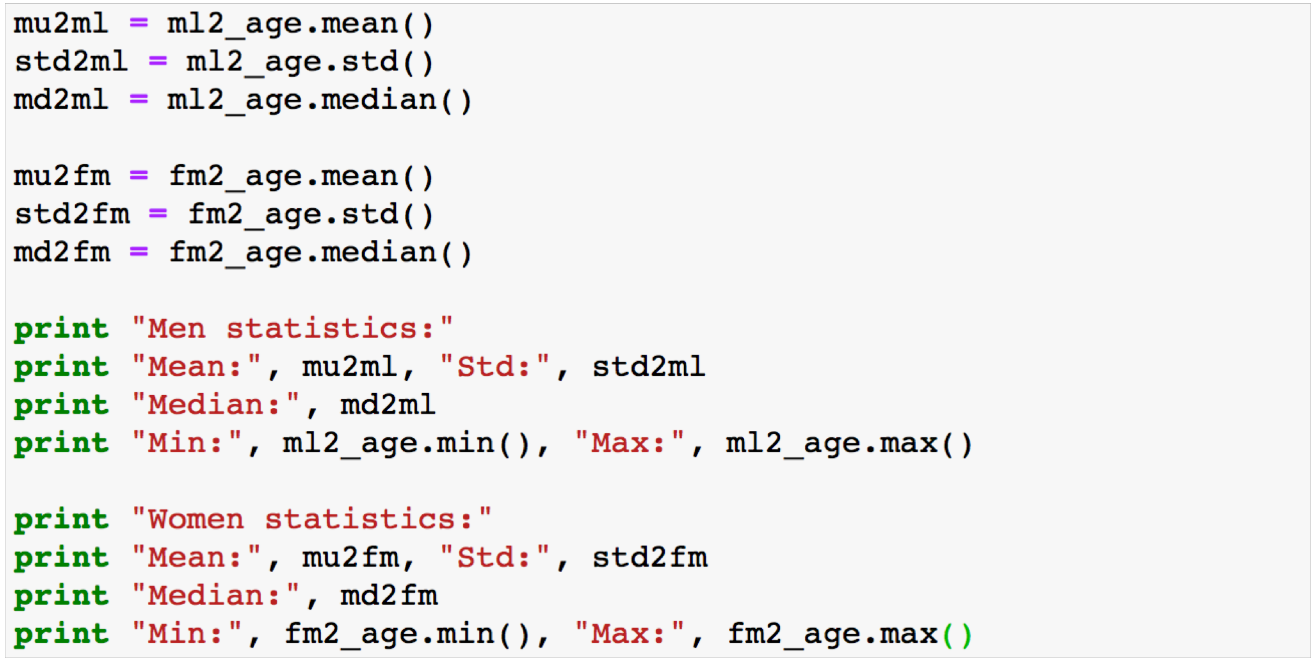
- Computing samples that are far from the median
- Computing samples whose values exceed the mean by 2 or 3 standard deviations

OUTLIER TREATMENT



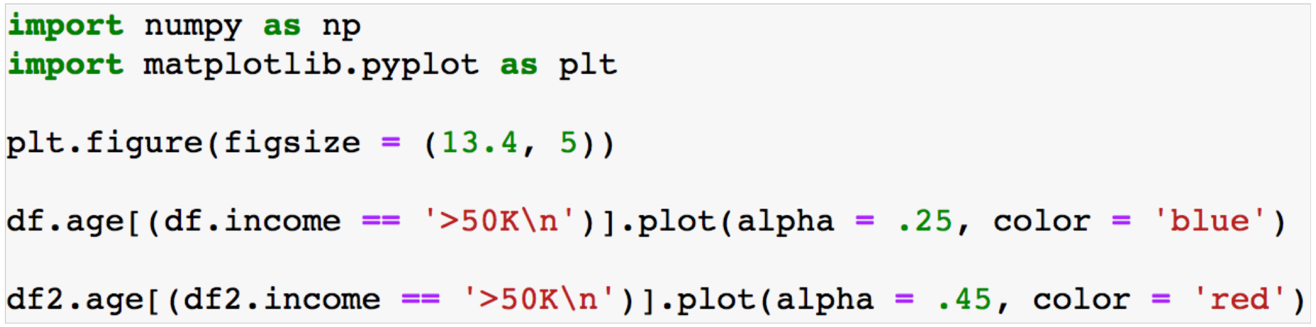
```
df2 = df.drop(df.index[
    (df.income == '>50K\n') &
    (df['age'] > df['age'].median() + 35) &
    (df['age'] > df['age'].median() -15)
])

ml1_age = ml1['age']
fm1_age = fm1['age']

ml2_age = ml1_age.drop(ml1_age.index[
    (ml1_age > df['age'].median() + 35) &
    (ml1_age > df['age'].median() - 15)
])

fm2_age = fm1_age.drop(fm1_age.index[
    (fm1_age > df['age'].median() + 35) &
    (fm1_age > df['age'].median() - 15)
])
```

OUTLIER TREATMENT

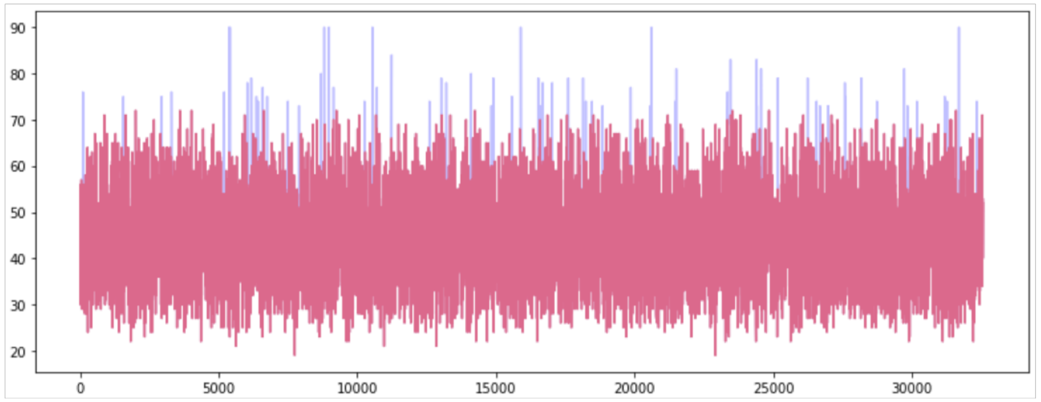


```
mu2ml = ml2_age.mean()
std2ml = ml2_age.std()
md2ml = ml2_age.median()

mu2fm = fm2_age.mean()
std2fm = fm2_age.std()
md2fm = fm2_age.median()

print "Men statistics:"
print "Mean:", mu2ml, "Std:", std2ml
print "Median:", md2ml
print "Min:", ml2_age.min(), "Max:", ml2_age.max()

print "Women statistics:"
print "Mean:", mu2fm, "Std:", std2fm
print "Median:", md2fm
print "Min:", fm2_age.min(), "Max:", fm2_age.max()
```

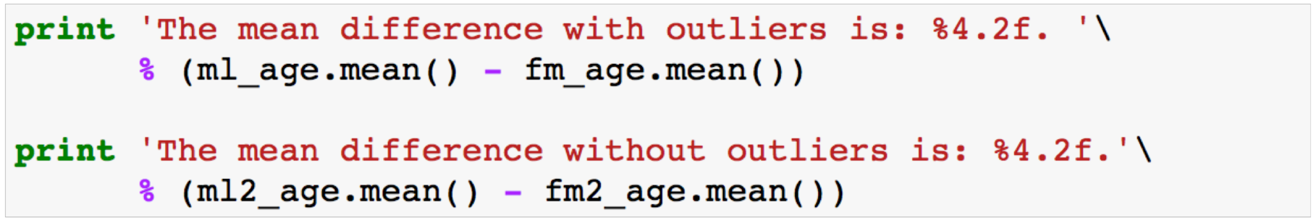
OUTLIER TREATMENT



```
import numpy as np
import matplotlib.pyplot as plt

plt.figure(figsize = (13.4, 5))

df.age[(df.income == '>50K\n')].plot(alpha = .25, color = 'blue')

df2.age[(df2.income == '>50K\n')].plot(alpha = .45, color = 'red')
```

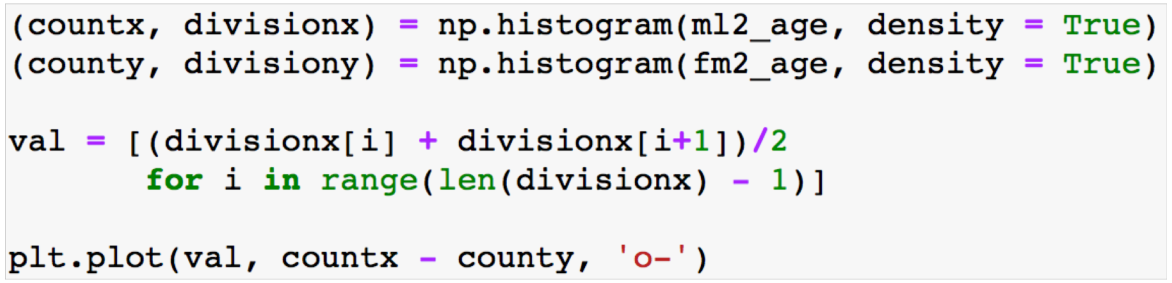


OUTLIER TREATMENT

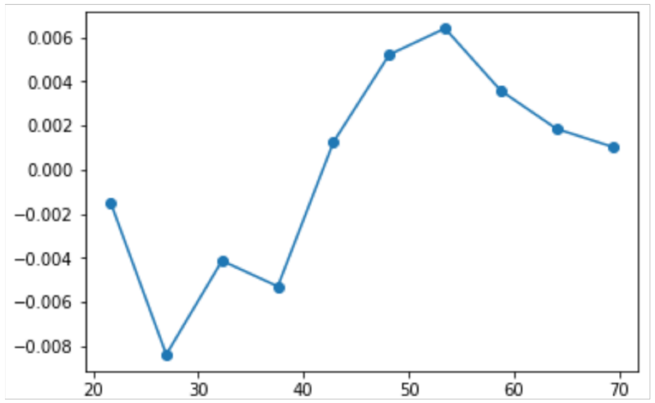


```
print 'The mean difference with outliers is: %4.2f. '\
      % (ml_age.mean() - fm_age.mean())

print 'The mean difference without outliers is: %4.2f.'\
      % (ml2_age.mean() - fm2_age.mean())
```

OUTLIER TREATMENT



```
(countx, divisionx) = np.histogram(ml2_age, density = True)
(county, divisiony) = np.histogram(fm2_age, density = True)

val = [(divisionx[i] + divisionx[i+1])/2
       for i in range(len(divisionx) - 1)]

plt.plot(val, countx - county, 'o-')
```



ESTIMATIONS

MEASURING ASYMMETRY

Skewness is a statistic that measures the asymmetry of the set of n data samples:

$$g(X) = \frac{1}{n}\frac{\sum_{i=1}^{n}(x_i - \mu)^3}{\sigma^3}$$

- Negative deviation indicates that the distribution “skews left” (it extends further to the left than to the right)
- One can easily see that the skewness for a normal distribution is zero, and any symmetric data must have a skewness of zero

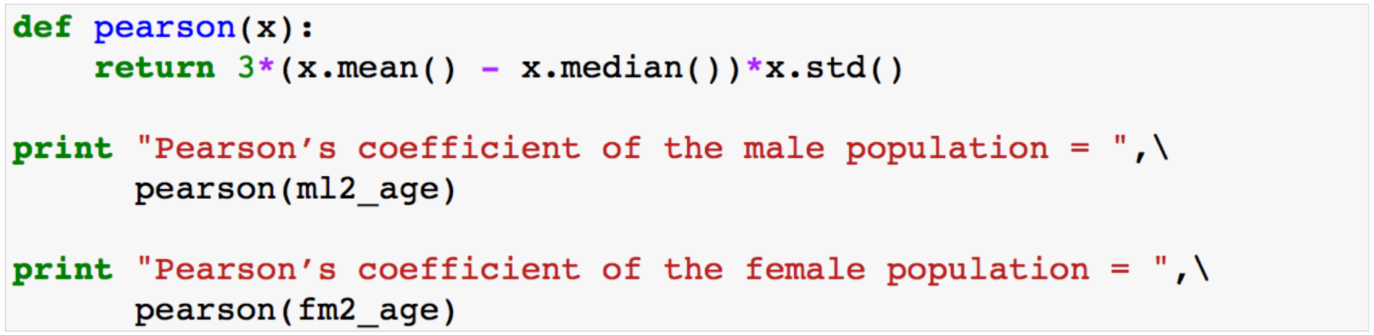
MEASURING ASYMMETRY

```
def skewness(x):
    res = 0
    m = x.mean()
    s = x.std()
    for i in x:
        res += (i-m) * (i-m) * (i-m)
    res /= (len(x) * s * s * s)
    return res

print "Skewness of the male population = ", skewness(ml2_age)
print "Skewness of the female population is = ", skewness(fm2_age)
```

MEASURING ASYMMETRY

The Pearson’s median skewness coefficient is a more robust alternative to the skewness coefficient:

$$g(X) = 3(\mu - \mu_{12})\sigma$$



```python
def pearson(x):
    return 3*(x.mean() - x.median())*x.std()

print "Pearson’s coefficient of the male population = ",\
      pearson(ml2_age)

print "Pearson’s coefficient of the female population = ",\
      pearson(fm2_age)
```

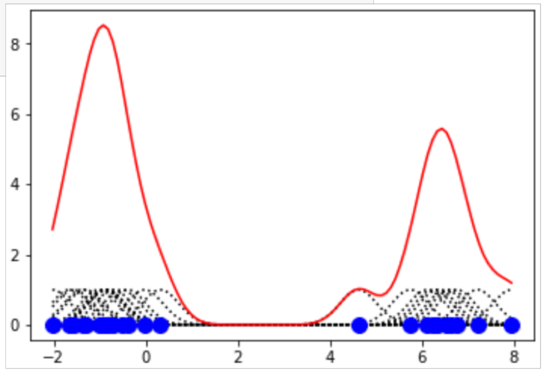
KERNEL DENSITY

Estimate the distribution non-parametrically (i.e., making no assumptions about the form of the underlying distribution) using kernel density estimation:

- We are interested in a continuous representation of the data
- For instance, a Gaussian kernel to generate the density around the data

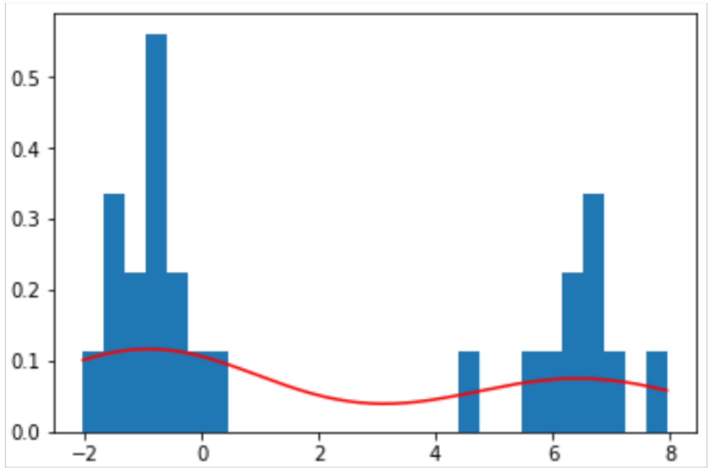
KERNEL DENSITY

```
x1 = np.random.normal(-1, 0.5, 15)
x2 = np.random.normal(6, 1, 10)
y = np.r_[x1, x2]  # r_ translates slice objects
                   # to concatenation along the first axis
x = np.linspace(min(y), max(y), 100)
s = 0.4 # Smoothing parameter

# Calculate the kernels
from scipy.stats import norm
kernels = np.transpose([norm.pdf(x, yi, s) for yi in y])
plt.plot(x, kernels, 'k:')
plt.plot(x, kernels.sum(1), 'r')
plt.plot(y, np.zeros(len(y)), 'bo', ms = 10)
```



KERNEL DENSITY

```
from scipy.stats import kde

density = kde.gaussian_kde(y)
xgrid = np.linspace(x.min(), x.max(), 200)

plt.hist(y, bins = 28, density = True)
plt.plot(xgrid, density(xgrid), 'r-')
```



ESTIMATION

Use estimates to approximate the values of unknown parameters of a dataset:

- **Estimated mean, variance, and standard score**: Point estimators that are single numerical estimates of parameters of a population

- **Covariance, Pearson's correlation and Spearman's rank correlation**: Variables of data can express relations

ESTIMATED MEAN

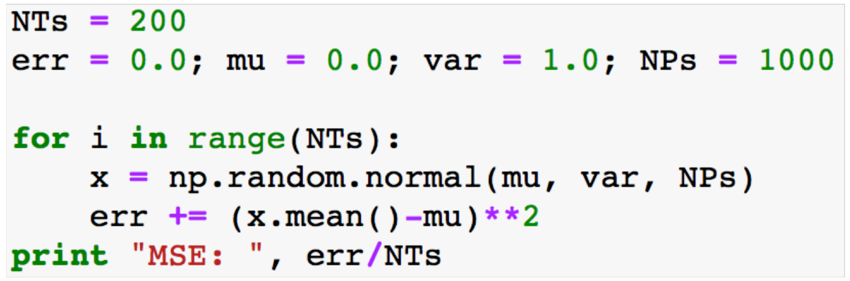
Given a dataset as a series of values, $\{ x_i \}$, the sample mean is an estimator

- For instance for the dataset $\{0.33, -1.76, 2.34, 0.56, 0.89\}$, the sample mean is 0.472
- Two ways:
  - Remove outliers and then calculate the sample mean
  - Use the sample median as an estimator of the mean of the distribution

ESTIMATED MEAN

If there are no outliers, the sample mean minimizes the following mean squared error:

$$MSE(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$$



```
NTs = 200
err = 0.0; mu = 0.0; var = 1.0; NPs = 1000

for i in range(NTs):
    x = np.random.normal(mu, var, NPs)
    err += (x.mean()-mu)**2
print "MSE: ", err/NTs
```

ESTIMATED VARIANCE

Given a dataset as a series of values, $\{ x_i \}$, we can use the sample variance as an estimator

- For **large** samples:

$$\bar{\sigma}^2(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

- For **small** samples:

$$\bar{\sigma}^2(X) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

STANDARD SCORE

Normalize data:

- Avoid data that come in different units
- Even data that come in the same units can still belong to different distributions

Given a dataset as a series of values, $\{ x_i \}$, we convert the data to standard scores by:

$$z_i = \frac{(x_i - \mu)}{\sigma}$$

Z is dimensionless and its distribution has a mean of 0 and variance of 1

COVARIANCE

When two datasets X, Y share the same tendency, we speak about covariance:

- Let us center the data with respect to their mean:

$$dx_i = x_i - \mu_X$$

$$dy_i = y_i - \mu_Y$$

- The covariance is defined as the mean of the following products:

$$Cov(X,Y) = \frac{1}{n}\sum_{i=1}^{n} dx_i \, dy_i$$

where n is the length of both sets

PEARSON'S CORRELATION

The Pearson's correlation is always between -1 and $+1$, where the magnitude depends on the degree of correlation:

$$\rho = \frac{1}{n}\sum_{i=1}^{n} \frac{x_i - \mu_X}{\sigma_X} \frac{y_i - \mu_Y}{\sigma_Y} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y}$$

- If the Pearson's correlation is 1 (or -1), it means that the variables are perfectly correlated; this means that one variable can predict the other very well
- Having $\rho = 0$, does not necessarily mean that the variables are not correlated! Pearson's correlation captures correlations of first order, but not nonlinear correlations

SPEARMAN’S RANK CORRELATION

Use the ranks of the sorted sample data, instead of the values themselves:

- It computes the correlation between the ranks of the data
- It comes as a solution to the robustness problem of Pearson’s correlation when the data contain outliers

For example:

- $X = [\ 10, 20, 30, 40, 1000\ ]$, $Y = [\ -70, -1000, -50, -10, -20\ ]$
- $R_X = [\ 1.0, 2.0, 3.0, 4.0, 5.0\ ]$, $R_Y = [\ 2.0, 1.0, 3.0, 5.0, 4.0\ ]$

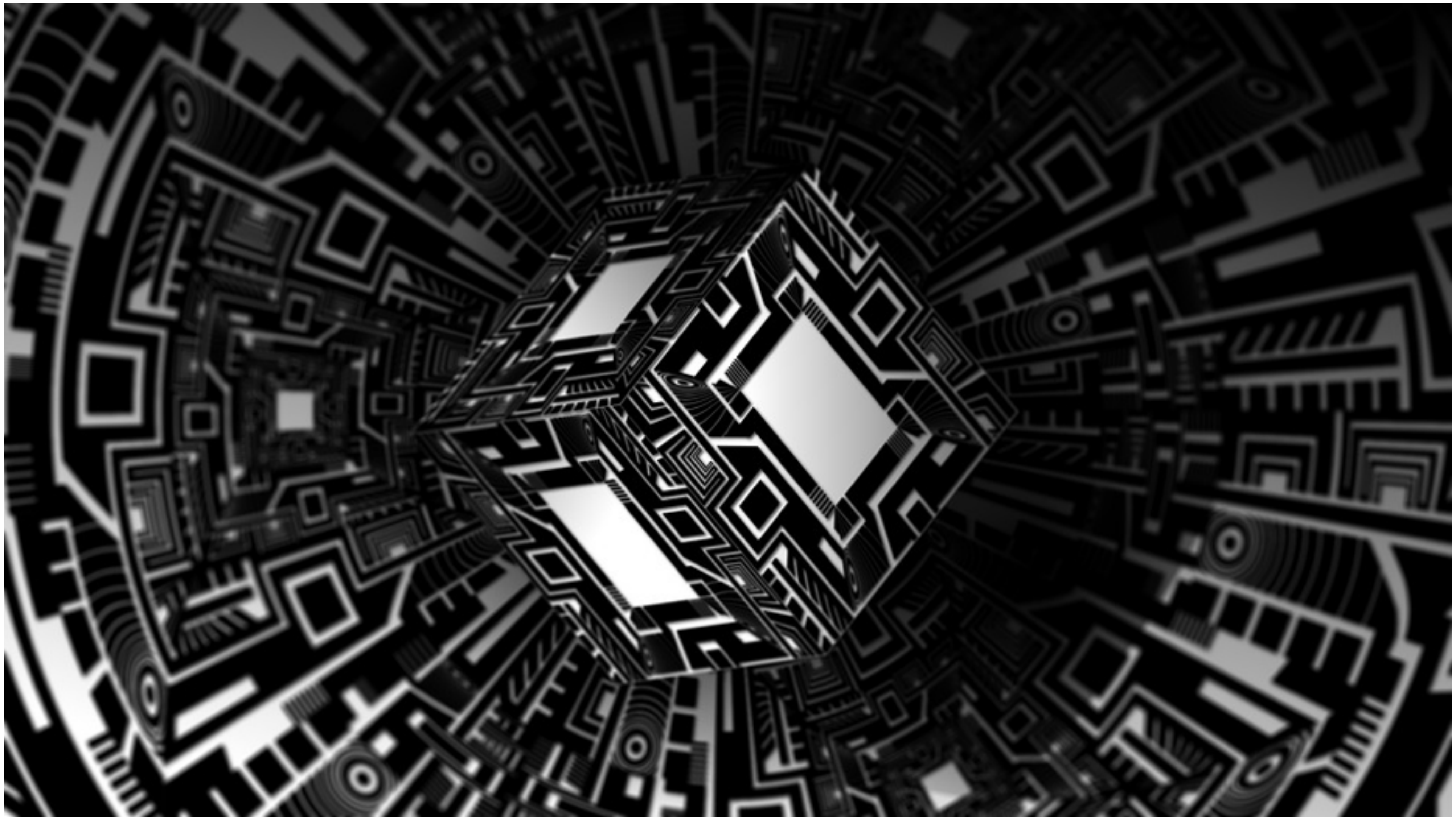
FINAL REMARKS

We have familiarized with the basic concepts and procedures of descriptive statistics to explore a dataset:

- Central measures of tendency such as the sample mean and median
- Variability measures such as the variance and standard deviation
- Measures can be affected by outliers

We obtain a continuous representation of the sample distribution using the kernel density

We estimate the correlation and the covariance of datasets using Pearson's and the Spearman's rank correlations



CONTINUOUS DISTRIBUTION

- Continuous distributions are distributions that are defined by a continuous function:
  - Exponential distributions describe the inter-arrival time between events

$$PDF(x) = \frac{1}{n}\frac{\sum_{i=0}^{n}(x_i - \mu^3)}{\sigma^3}$$

  - Normal distributions (Gaussian distributions) represent many real phenomena such as economic, natural, social, and others

$$g(x) = \frac{1}{n}\frac{\sum_{i=0}^{n}(x_i - \mu^3)}{\sigma^3}$$