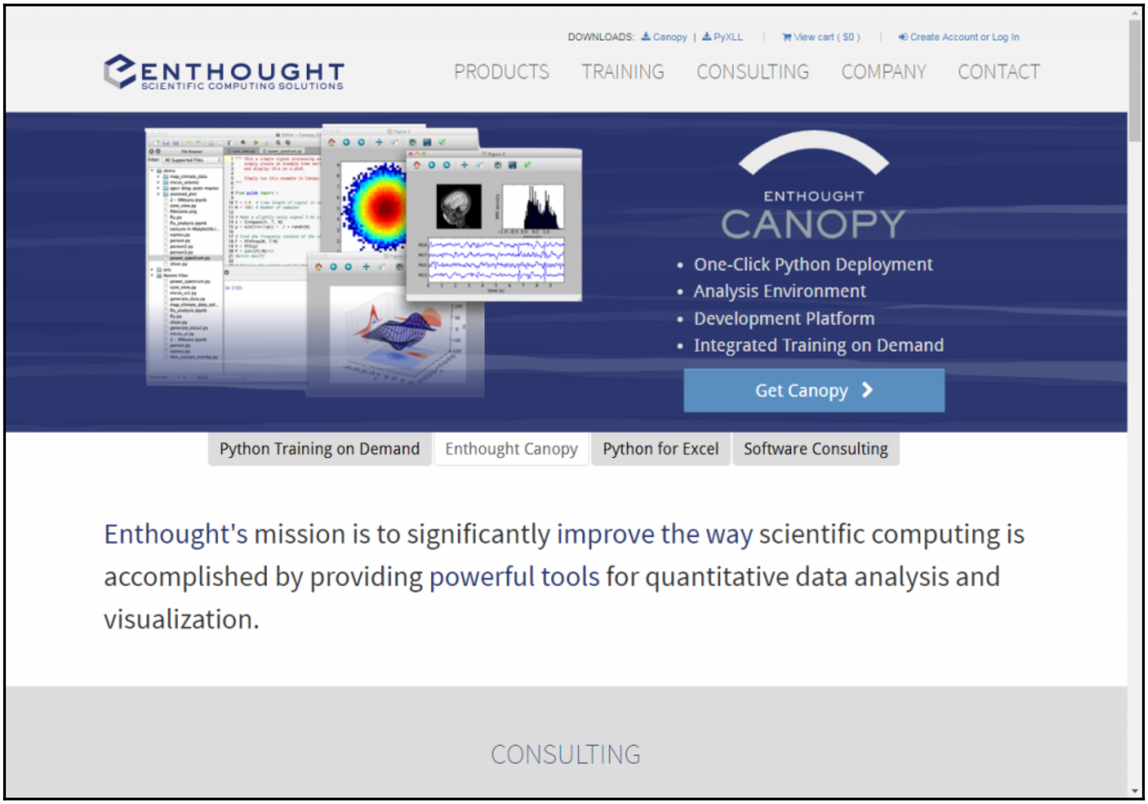
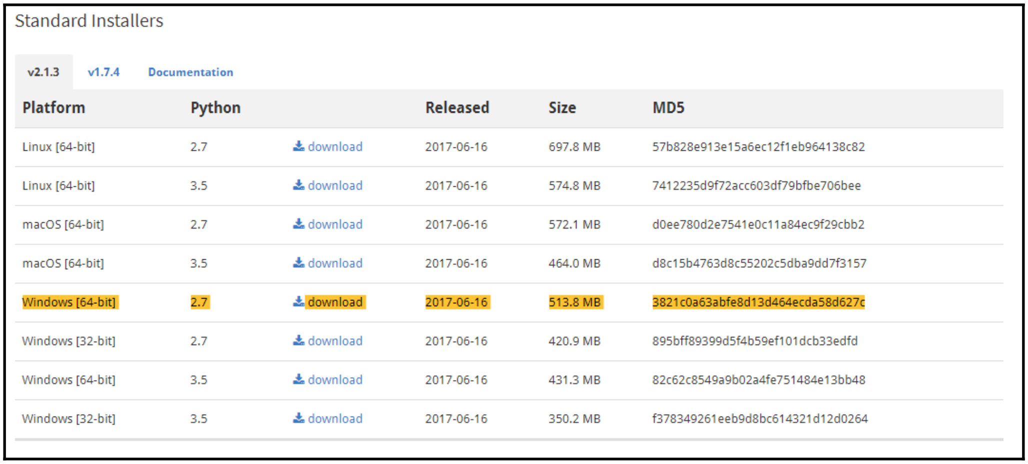
Graphic Bundle

Chapter 1 : Getting Started

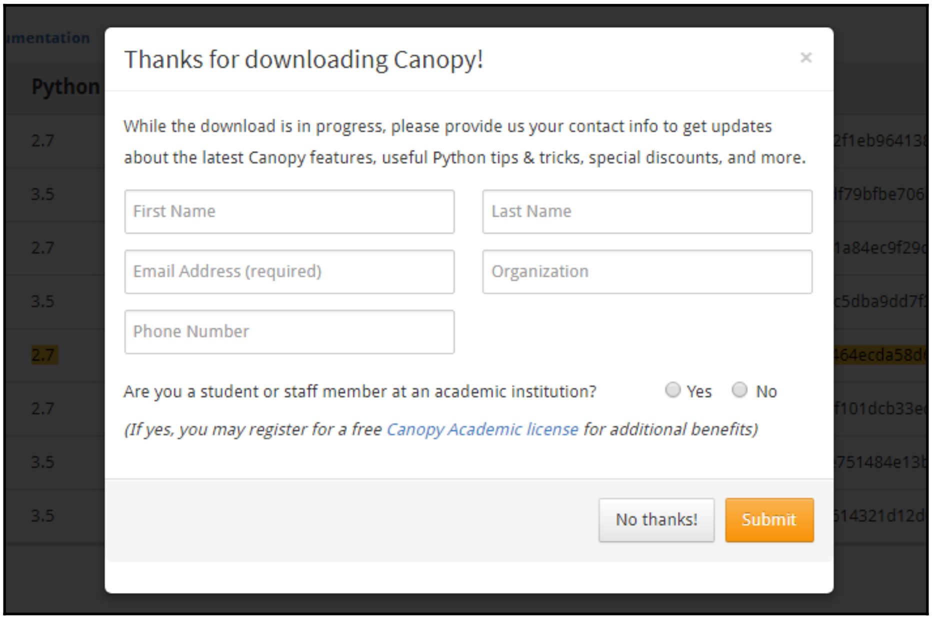


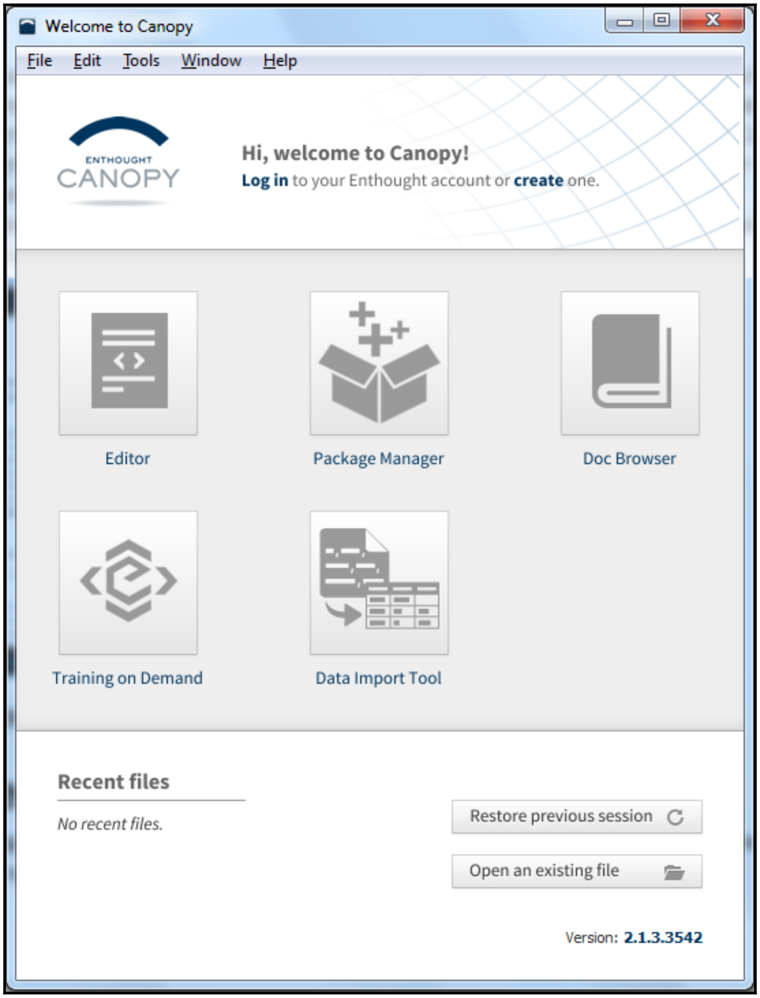


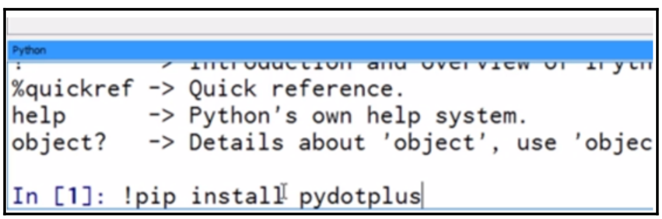
Standard Installers

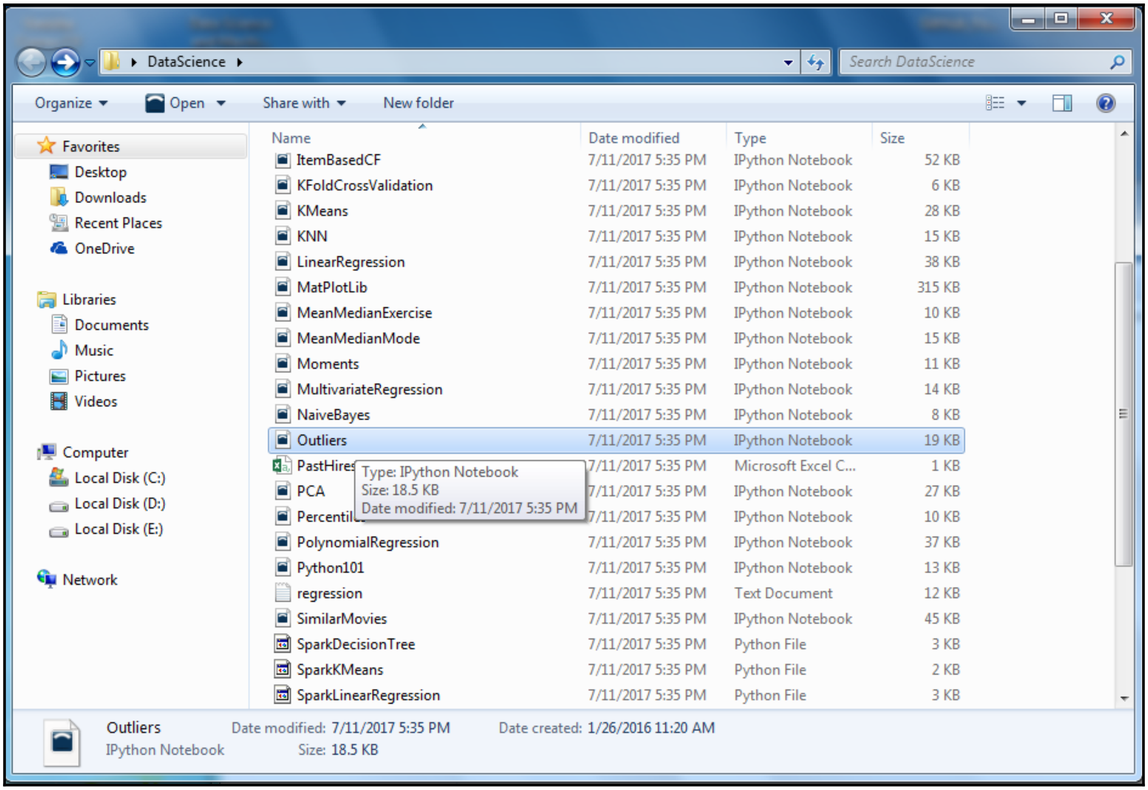
v2.1.3 v1.7.4 Documentation

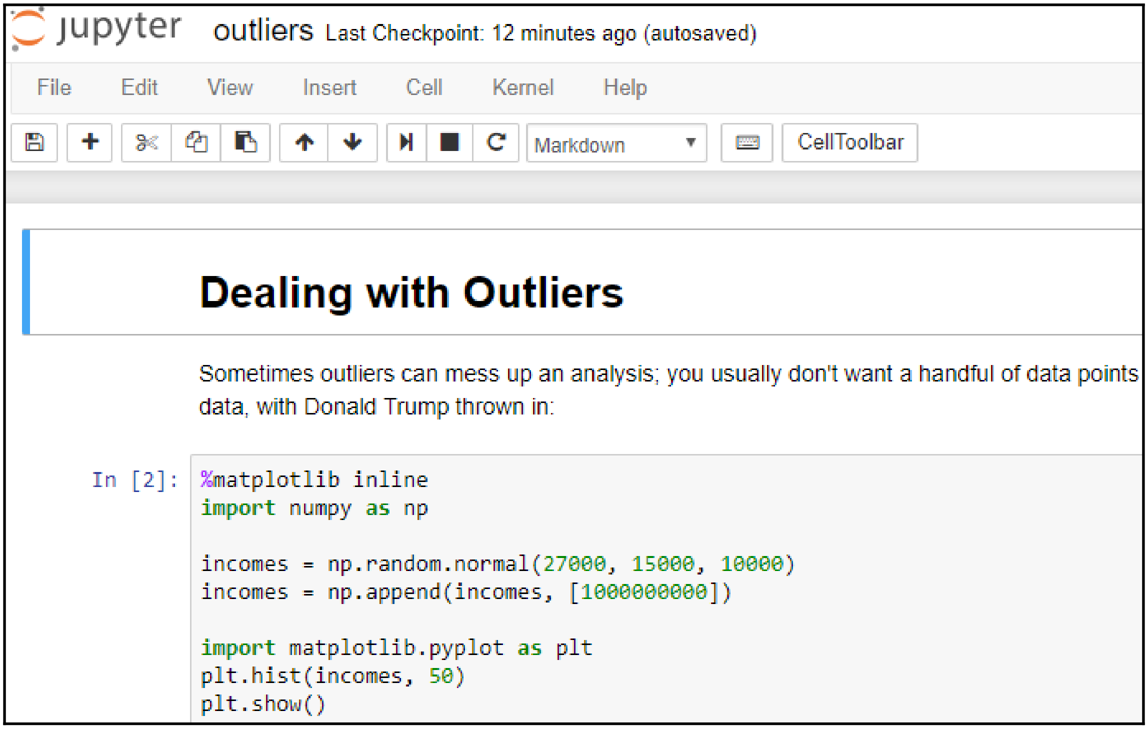
Platform	Python		Released	Size	MD5
Linux [64-bit]	2.7	download	2017-06-16	697.8 MB	57b828e913e15a6ec12f1eb964138c82
Linux [64-bit]	3.5	download	2017-06-16	574.8 MB	7412235d9f72acc603df79bfbe706bee
macOS [64-bit]	2.7	download	2017-06-16	572.1 MB	d0ee780d2e7541e0c11a84ec9f29cbb2
macOS [64-bit]	3.5	download	2017-06-16	464.0 MB	d8c15b4763d8c55202c5dba9dd7f3157
Windows [64-bit]	2.7	download	2017-06-16	513.8 MB	3821c0a63abfe8d13d464ecda58d627c
Windows [32-bit]	2.7	download	2017-06-16	420.9 MB	895bff89399d5f4b59ef101dcb33edfd
Windows [64-bit]	3.5	download	2017-06-16	431.3 MB	82c62c8549a9b02a4fe751484e13bb48
Windows [32-bit]	3.5	download	2017-06-16	350.2 MB	f378349261eeb9d8bc614321d12d0264




Welcome to Canopy
File
Edit
Tools
Window
Help
ENTHOUGHT
CANOPY
Hi, welcome to Canopy!
Log in to your Enthought account or create one.
Editor
Package Manager
Doc Browser
Training on Demand
Data Import Tool
Recent files
No recent files.
Restore previous session
Open an existing file
Version: 2.1.3.3542


Python
%quickref -> Quick reference.
help   -> Python's own help system.
object?   -> Details about 'object', use 'objec
In [1]: !pip install pydotplus


DataScience
Search DataScience
Organize
Open
Share with
New folder
Favorites
Desktop
Downloads
Recent Places
OneDrive
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
Local Disk (D:)
Local Disk (E:)
Network
Name
Date modified
Type
Size
ItemBasedCF 7/11/2017 5:35 PM IPython Notebook 52 KB
KFoldCrossValidation 7/11/2017 5:35 PM IPython Notebook 6 KB
KMeans 7/11/2017 5:35 PM IPython Notebook 28 KB
KNN 7/11/2017 5:35 PM IPython Notebook 15 KB
LinearRegression 7/11/2017 5:35 PM IPython Notebook 38 KB
MatPlotLib 7/11/2017 5:35 PM IPython Notebook 315 KB
MeanMedianExercise 7/11/2017 5:35 PM IPython Notebook 10 KB
MeanMedianMode 7/11/2017 5:35 PM IPython Notebook 15 KB
Moments 7/11/2017 5:35 PM IPython Notebook 11 KB
MultivariateRegression 7/11/2017 5:35 PM IPython Notebook 14 KB
NaiveBayes 7/11/2017 5:35 PM IPython Notebook 8 KB
Outliers 7/11/2017 5:35 PM IPython Notebook 19 KB
PastHires 7/11/2017 5:35 PM Microsoft Excel C... 1 KB
PCA 7/11/2017 5:35 PM IPython Notebook 27 KB
Percentil 7/11/2017 5:35 PM IPython Notebook 10 KB
Type: IPython Notebook
Size: 18.5 KB
Date modified: 7/11/2017 5:35 PM
PolynomialRegression 7/11/2017 5:35 PM IPython Notebook 37 KB
Python101 7/11/2017 5:35 PM IPython Notebook 13 KB
regression 7/11/2017 5:35 PM Text Document 12 KB
SimilarMovies 7/11/2017 5:35 PM IPython Notebook 45 KB
SparkDecisionTree 7/11/2017 5:35 PM Python File 3 KB
SparkKMeans 7/11/2017 5:35 PM Python File 2 KB
SparkLinearRegression 7/11/2017 5:35 PM Python File 3 KB
Outliers
IPython Notebook
Date modified: 7/11/2017 5:35 PM
Size: 18.5 KB
Date created: 1/26/2016 11:20 AM



Dealing with Outliers

Sometimes outliers can mess up an analysis; you usually don't want a handful of data points
data, with Donald Trump thrown in:

In [2]:
```
%matplotlib inline
import numpy as np

incomes = np.random.normal(27000, 15000, 10000)
incomes = np.append(incomes, [1000000000])

import matplotlib.pyplot as plt
plt.hist(incomes, 50)
plt.show()
```

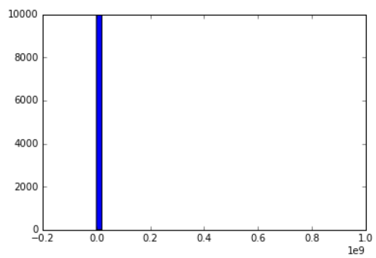
File Edit View Insert Cell Kernel Help | Python 2

CellToolbar

Dealing with Outliers

Sometimes outliers can mess up an analysis; you usually don't want a handful of data points to skew the overall results. Let's revisit our example of income data, with Donald Trump thrown in:

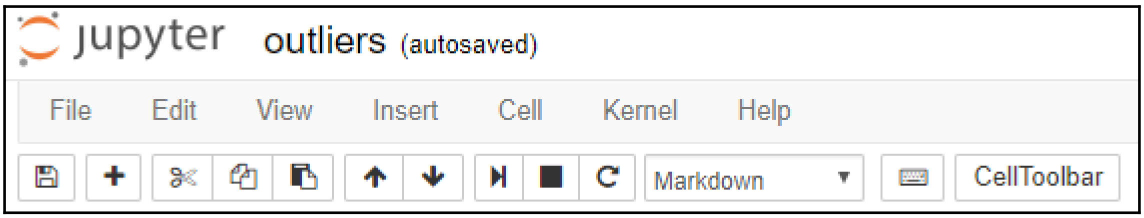
In [2]:
```
%matplotlib inline
import numpy as np

incomes = np.random.normal(27000, 15000, 10000)
incomes = np.append(incomes, [1000000000])

import matplotlib.pyplot as plt
plt.hist(incomes, 50)
plt.show()
```

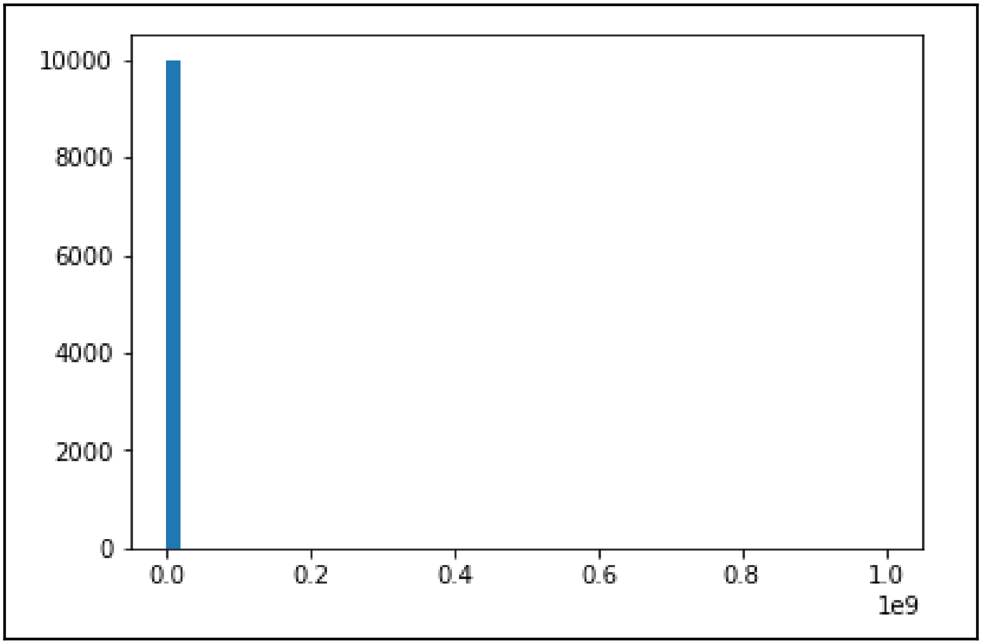


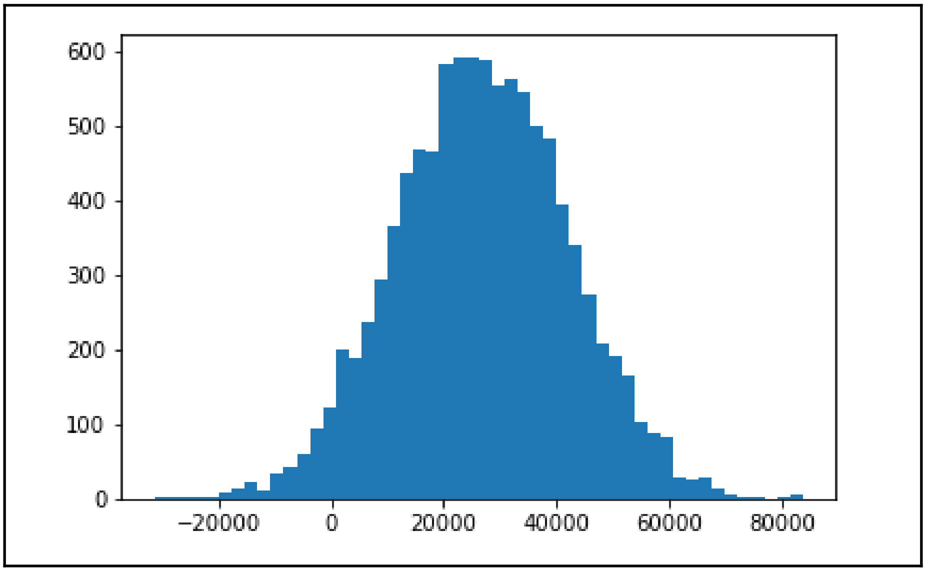
That's not very helpful to look at. One billionaire ended up squeezing everybody else into a single line in my histogram. Plus it skewed my mean income significantly:

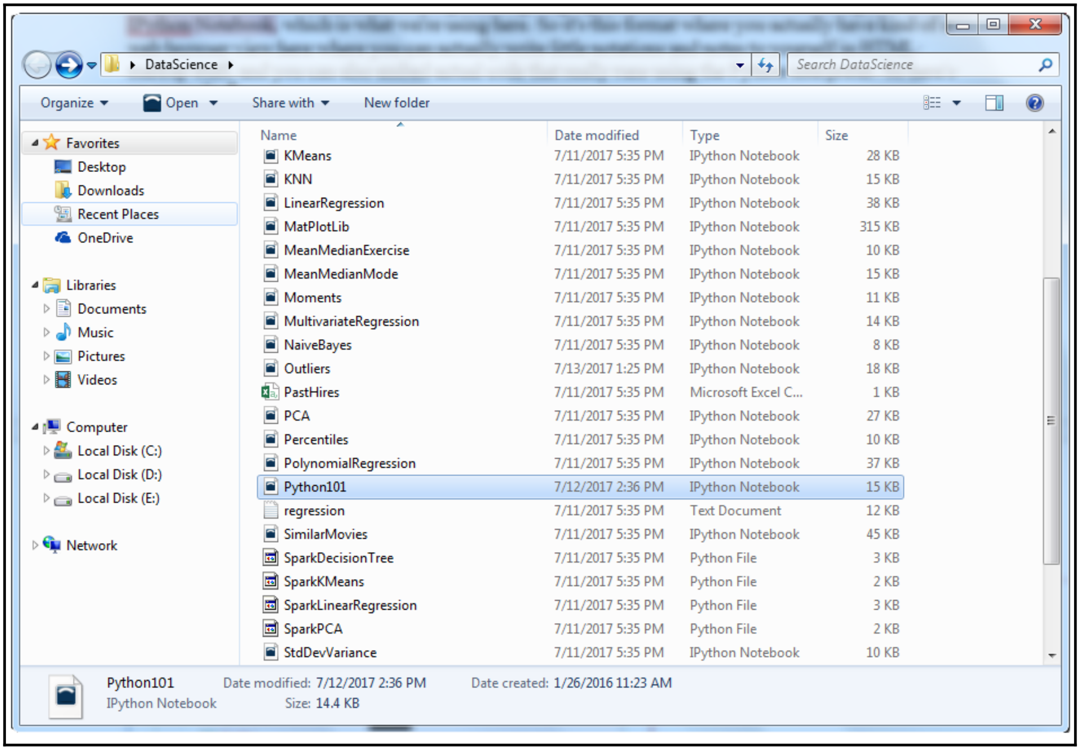
In [3]:
```
incomes.mean()
```

Out[3]:
```
127148.50796177129
```




10000
8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
1e9


600
500
400
300
200
100
0
−20000
0
20000
40000
60000
80000



Python Basics

Whitespace Is Important

In [1]:
```
listOfNumbers = [1, 2, 3, 4, 5, 6]

for number in listOfNumbers:
    print (number),
    if (number % 2 == 0):
        print ("is even")
    else:
        print ("is odd")

print ("All done.")
```

```
1
is odd
2
is even
3
is odd
4
is even
5
is odd
6
is even
All done.
```

Python Basics

Whitespace Is Important

In [1]:
```
listOfNumbers = [1, 2, 3, 4, 5, 6]

for number in listOfNumbers:
    print (number),
    if (number % 2 == 0):
        print ("is even")
    else:
        print ("is odd")

print ("All done.")
```

```
1
is odd
2
is even
3
is odd
4
is even
5
is odd
6
is even
All done.
```

Whitespace Is Important

In [4]:
```
listOfNumbers = [1, 2, 3, 4, 5, 6]
```

```
1 is odd
2 is even
3 is odd
4 is even
5 is odd
6 is even
All done.
```

Python Basics

Whitespace Is Important

In [9]:
```
listOfNumbers = [1, 2, 3, 4, 5, 6]

for number in listOfNumbers:
    print (number),
    if (number % 2 == 0):
        print ("is even")
    else:
        print ("is odd")

print ("Hooray! We're all done. Let's party!")
```

```
1
is odd
2
is even
3
is odd
4
is even
5
is odd
6
is even
Hooray! We're all done. Let's party!
```

```
[ 23.50119237  28.3470395   27.68512972  27.43957344  22.66626262
  25.98055199  27.87395644  25.99525487  20.36318406  22.77226693]
```

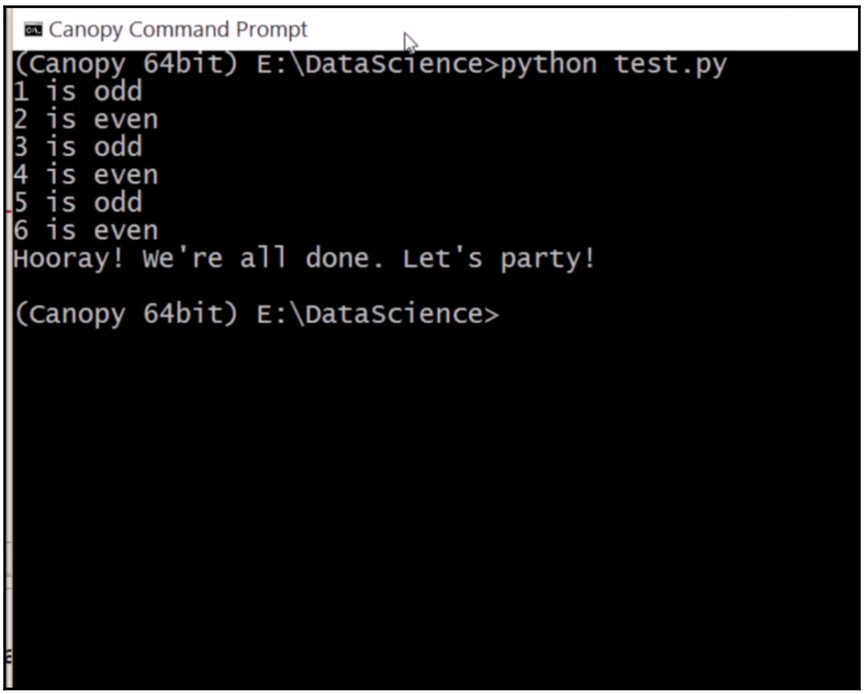
```
[ 48.79441876  63.77818473  61.24157056  47.38182128  52.5623337
  55.80574543  55.16594437  53.59688042  50.57639509  60.44058303]
```

```
In [8]: # Like a map or hash table in other languages
        captains = {}
        captains["Enterprise"] = "Kirk"
        captains["Enterprise D"] = "Picard"
        captains["Deep Space Nine"] = "Sisko"
        captains["Voyager"] = "Janeway"

        print (captains['Voyager'])

        Janeway

In [9]: print (captains.get("Enterprise"))

        Kirk

In [10]: print (captains.get("NX-01"))

        None

In [11]: for ship in captains:
             print (ship + ":" + captains[ship])

        Deep Space Nine:Sisko
        Enterprise:Kirk
        Voyager:Janeway
        Enterprise D:Picard
```

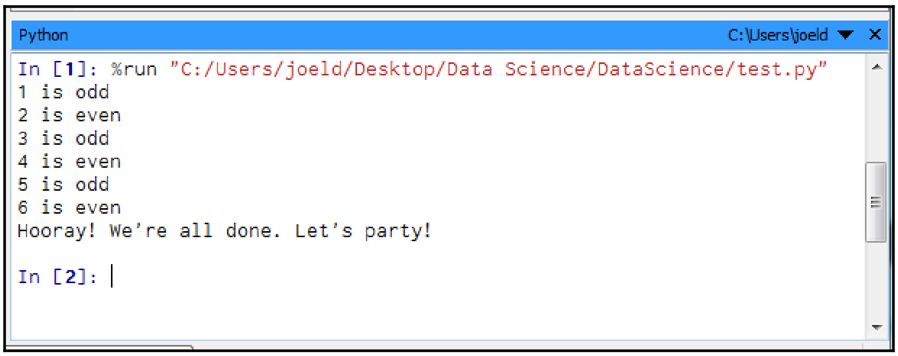
```
Enterprise D: Picard
Deep Space Nine: Sisko
Enterprise: Kirk
Voyager: Janeway
```

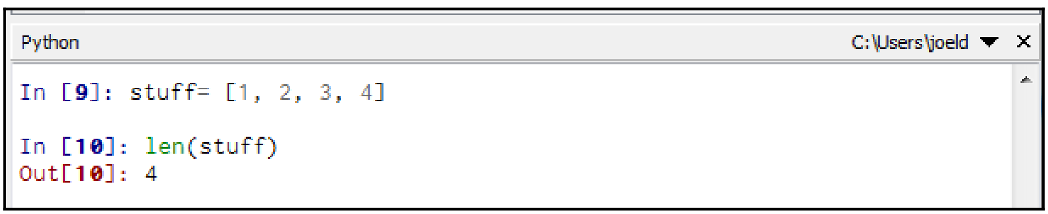
```
In [ ]:
```

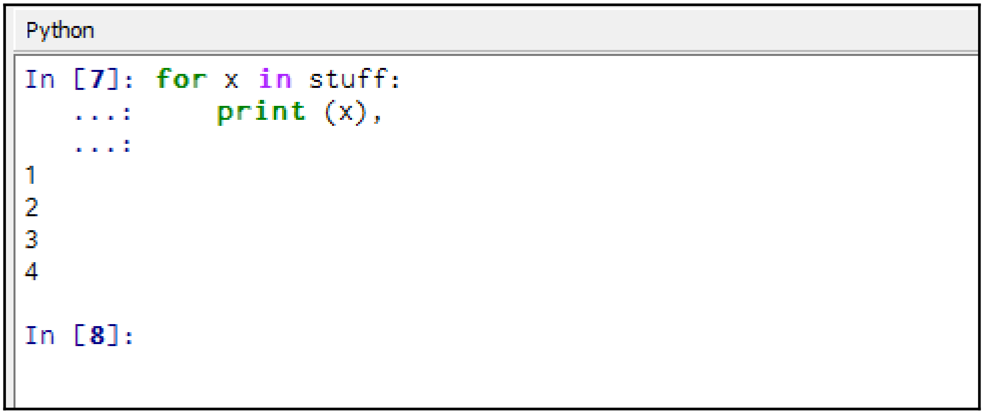

Editor - Canopy
File Edit View Search Run Tools Window Help
File Browser
Filter: All Files (*)
bhagyashree
Recent Files
test.py
test.py
1 listOfNumbers = [1, 2, 3, 4, 5, 6]
2
3 for number in listOfNumbers:
4   print (number),
5   if (number % 2 == 0):
6   print ("is even")
7   else:
8   print ("is odd")
9
10 print ("Hooray! We're all done. Let's party!")
Python
C:\Users\bhagyashree
Welcome to Canopy's interactive data-analysis environment!
Type '?' for more information.
Python 3.5.2 |Enthought, Inc. (x86_64)| (default, Mar 2 2017, 16:37:47) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 5.3.0 -- An enhanced Interactive Python.
?   -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help   -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
In [9]:
Cursor pos 10 : 47
Python 2
~\Desktop\DataScience\DataScience\test.py

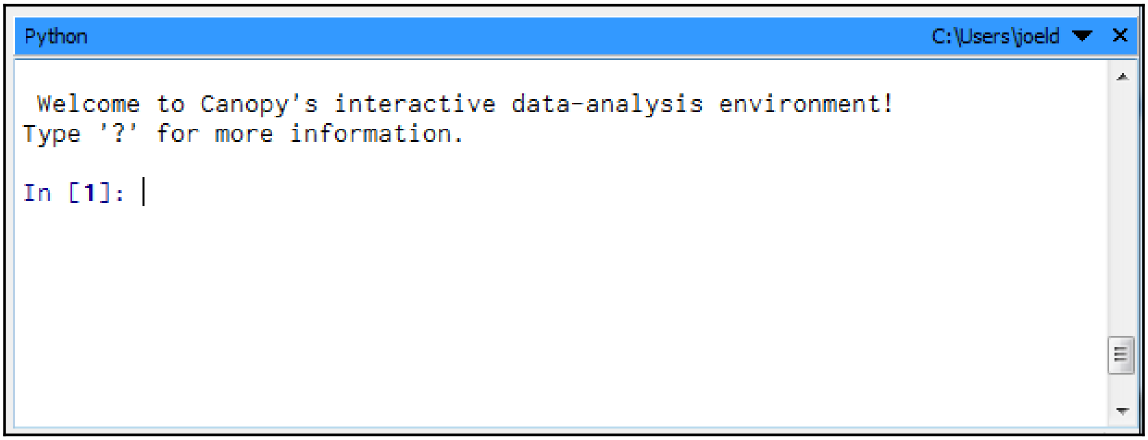

Canopy Command Prompt
(Canopy 64bit) E:\DataScience>python test.py
1 is odd
2 is even
3 is odd
4 is even
5 is odd
6 is even
Hooray! We're all done. Let's party!
(Canopy 64bit) E:\DataScience>

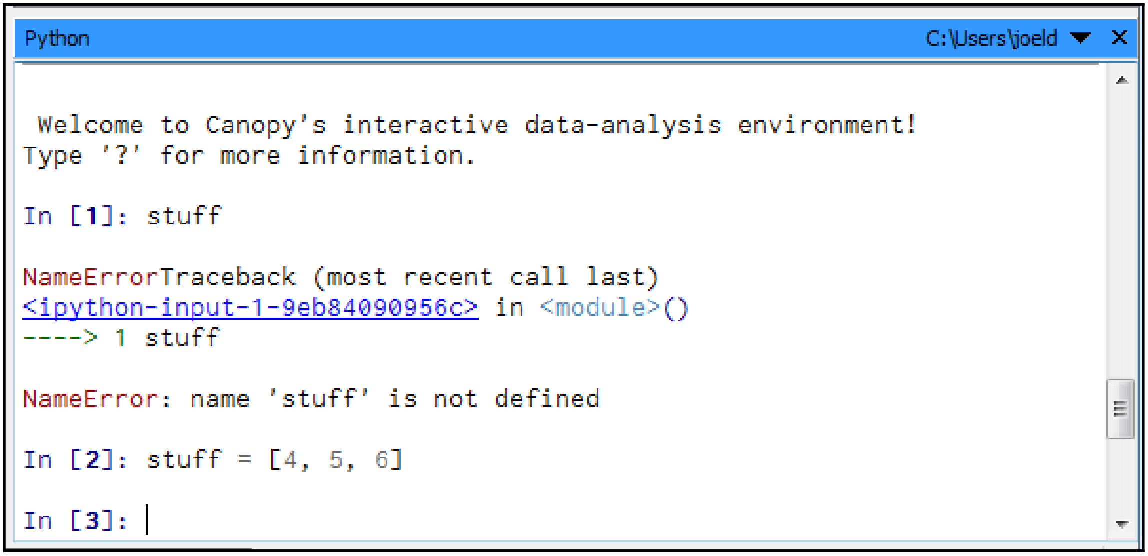

Editor - Canopy
File Edit View Search Run Tools Window Help
File Browser
Filter: All Files (*)
bhagyashree
Recent Files
test.py
test.py
listOfNumbers = [1, 2, 3, 4, 5, 6]
for number in listOfNumbers:
print (number),
if (number % 2 == 0):
print ("is even")
else:
print ("is odd")
print ("Hooray! We're all done. Let's party!")
Python
C:\Users\bhagyashree
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.
In [9]: %run "C:\Users\bhagyashree\Desktop\DataScience\DataScience\test.py"
1
is odd
2
is even
3
is odd
4
is even
5
is odd
6
is even
Hooray! We're all done. Let's party!
Cursor pos 10: 47
Python 2
~\Desktop\DataScience\DataScience\test.py


Python
C:\Users\joeld
In [1]: %run "C:/Users/joeld/Desktop/Data Science/DataScience/test.py"
1 is odd
2 is even
3 is odd
4 is even
5 is odd
6 is even
Hooray! We're all done. Let's party!
In [2]:

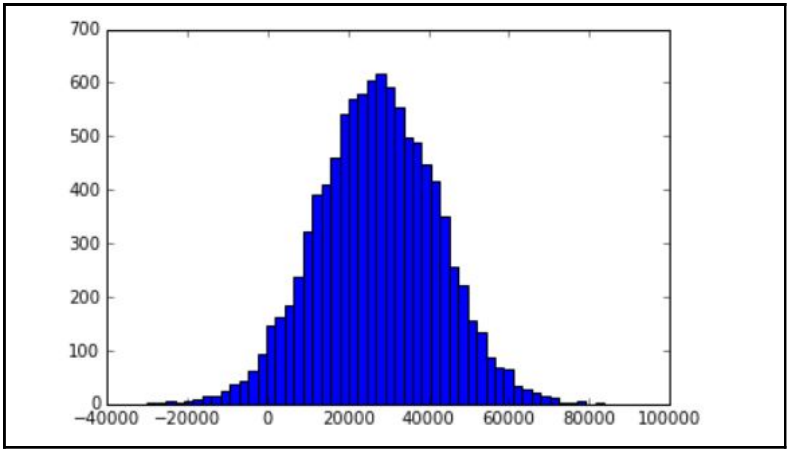

Python
C:\Users\joeld
In [9]: stuff= [1, 2, 3, 4]
In [10]: len(stuff)
Out[10]: 4


Python
In [7]: for x in stuff:
...:   print (x),
...:
1
2
3
4
In [8]:


Python
C:\Users\joeld
Welcome to Canopy's interactive data-analysis environment!
Type '?' for more information.
In [1]:

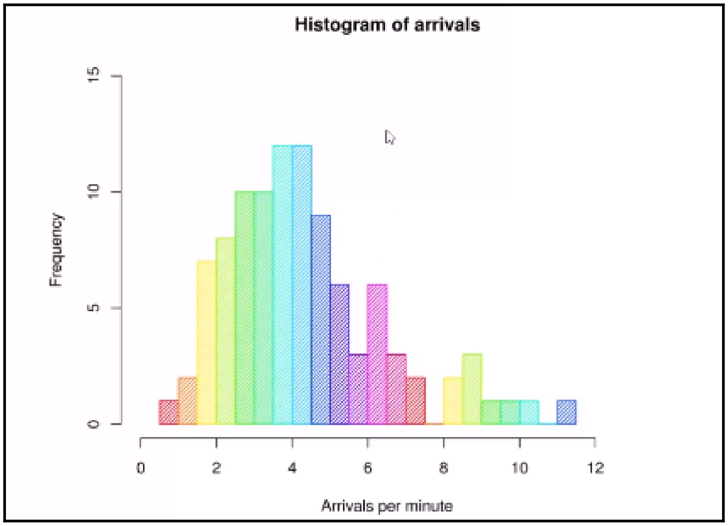


Chapter 2: Statistics and Probability Refresher, and Python Practice

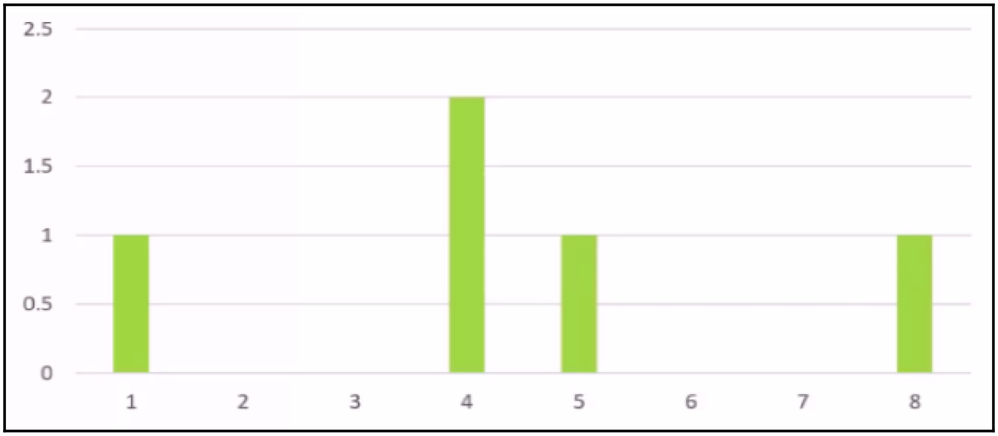



```
Out[7]: array([69, 87, 31, 22, 78, 37, 77, 32, 18, 59, 29, 43, 34, 33, 56, 83, 66,
               30, 77, 74, 31, 21, 85, 50, 47, 26, 72, 62, 33, 45, 86, 50, 86, 56,
               31, 84, 78, 27, 76, 42, 83, 64, 48, 54, 70, 56, 24, 50, 50, 71, 49,
               20, 85, 61, 33, 83, 55, 21, 60, 80, 56, 89, 61, 56, 52, 55, 20, 31,
               69, 50, 21, 52, 31, 83, 43, 77, 27, 67, 39, 39, 26, 38, 40, 73, 50,
               31, 87, 23, 50, 34, 69, 45, 83, 51, 88, 41, 64, 59, 40, 89, 57, 62,
               55, 75, 38, 51, 24, 21, 18, 75, 58, 62, 81, 65, 89, 64, 43, 33, 53,
               72, 20, 56, 19, 26, 81, 68, 70, 70, 41, 59, 50, 77, 62, 31, 87, 58,
               63, 83, 35, 55, 38, 85, 53, 66, 28, 74, 42, 28, 80, 69, 54, 25, 74,
               58, 27, 42, 87, 46, 43, 44, 33, 40, 21, 21, 73, 48, 87, 63, 84, 55,
               61, 66, 48, 73, 27, 60, 34, 77, 59, 58, 50, 70, 30, 76, 72, 33, 80,
               43, 63, 49, 60, 61, 53, 55, 79, 38, 46, 38, 81, 66, 29, 81, 46, 19,
               49, 57, 31, 18, 25, 47, 20, 88, 33, 88, 50, 22, 57, 39, 20, 59, 63,
               38, 35, 59, 28, 23, 56, 50, 46, 65, 46, 88, 87, 34, 73, 75, 32, 49,
               67, 77, 86, 38, 80, 36, 64, 79, 65, 51, 46, 54, 23, 82, 56, 41, 78,
               19, 45, 38, 70, 74, 56, 87, 49, 69, 30, 25, 22, 71, 39, 41, 46, 72,
               33, 72, 88, 37, 75, 39, 37, 21, 67, 86, 77, 20, 46, 53, 22, 85, 73,
               89, 67, 24, 24, 25, 62, 56, 58, 44, 63, 30, 36, 73, 49, 45, 26, 33,
               20, 62, 75, 34, 81, 59, 64, 27, 43, 23, 62, 75, 81, 40, 65, 29, 61,
               55, 81, 35, 68, 79, 86, 43, 35, 74, 59, 80, 75, 60, 82, 66, 54, 37,
               54, 71, 88, 46, 55, 63, 79, 89, 48, 61, 68, 78, 51, 32, 26, 48, 78,
               76, 62, 19, 19, 63, 20, 44, 28, 34, 58, 44, 36, 70, 34, 67, 50, 33,
               31, 18, 72, 55, 49, 63, 81, 65, 51, 46, 22, 55, 77, 76, 53, 79, 47,
               57, 46, 27, 29, 49, 71, 19, 85, 86, 77, 89, 59, 67, 26, 50, 79, 85,
               68, 51, 30, 18, 73, 52, 22, 53, 56, 26, 45, 60, 83, 50, 34, 68, 65,
               27, 72, 24, 34, 37, 52, 67, 79, 79, 24, 65, 71, 28, 29, 61, 34, 77,
               35, 59, 50, 83, 27, 32, 18, 81, 36, 46, 48, 39, 52, 23, 37, 62, 54,
               53, 50, 34, 36, 88, 83, 39, 89, 65, 83, 73, 66, 28, 36, 56, 86, 65,
               28, 46, 18, 61, 69, 80, 85, 29, 85, 44, 18, 61, 68, 83, 89, 53, 65,
               55, 66, 87, 55, 43, 32, 84])
```

```
Out[12]: array([41, 74, 26, 31, 31, 31, 20, 64, 59, 76, 80, 59, 53, 50, 29, 67, 55,
       82, 41, 40, 77, 41, 73, 52, 38, 87, 28, 87, 60, 47, 87, 66, 71, 77,
       85, 40, 22, 40, 74, 69, 44, 46, 72, 60, 69, 56, 19, 84, 80, 83, 22,
       63, 74, 31, 32, 20, 58, 71, 56, 43, 32, 67, 32, 51, 79, 54, 25, 81,
       50, 55, 86, 75, 30, 37, 37, 37, 56, 22, 85, 82, 58, 78, 32, 50, 52,
       70, 85, 37, 34, 83, 41, 52, 46, 55, 84, 64, 19, 86, 46, 65, 77, 80,
       82, 86, 65, 41, 35, 44, 45, 34, 46, 51, 83, 82, 53, 50, 84, 83, 29,
       47, 80, 75, 72, 81, 40, 75, 74, 57, 27, 71, 76, 65, 27, 75, 32, 26,
       34, 20, 58, 19, 18, 26, 73, 60, 31, 34, 46, 80, 76, 30, 70, 68, 71,
       45, 44, 47, 30, 39, 35, 60, 44, 45, 83, 64, 21, 35, 25, 70, 86, 53,
       65, 87, 66, 88, 48, 18, 29, 60, 50, 29, 67, 45, 83, 76, 62, 25, 41,
       56, 23, 60, 56, 59, 77, 64, 74, 39, 43, 24, 55, 88, 60, 19, 32, 49,
       59, 88, 69, 82, 56, 70, 34, 52, 85, 70, 79, 26, 37, 60, 40, 32, 20,
       81, 43, 47, 83, 67, 27, 30, 21, 24, 40, 43, 83, 79, 47, 36, 66, 37,
       76, 20, 48, 81, 58, 62, 27, 21, 88, 31, 62, 38, 83, 33, 41, 68, 38,
       43, 44, 49, 51, 82, 48, 53, 75, 56, 48, 38, 76, 37, 41, 62, 26, 32,
       53, 40, 89, 40, 19, 29, 73, 71, 81, 63, 36, 56, 30, 60, 67, 47, 20,
       62, 86, 84, 88, 37, 47, 35, 37, 26, 48, 36, 53, 19, 77, 46, 63, 87,
       60, 40, 72, 86, 41, 58, 29, 43, 36, 69, 75, 56, 55, 33, 66, 22, 46,
       73, 45, 30, 42, 51, 24, 18, 54, 45, 73, 37, 54, 84, 29, 73, 82, 47,
       55, 68, 42, 60, 25, 46, 89, 37, 20, 34, 24, 40, 61, 66, 72, 71, 30,
       50, 29, 24, 60, 30, 76, 67, 66, 19, 75, 33, 21, 21, 45, 38, 47, 69,
       71, 83, 50, 40, 24, 38, 47, 72, 25, 26, 77, 44, 39, 35, 36, 42, 73,
       78, 77, 62, 43, 84, 66, 41, 48, 69, 65, 52, 45, 85, 43, 77, 31, 50,
       61, 69, 71, 77, 89, 65, 41, 35, 88, 37, 87, 75, 21, 38, 73, 31, 66,
       25, 25, 69, 71, 46, 86, 66, 82, 24, 77, 44, 81, 72, 25, 50, 58, 22,
       85, 42, 44, 62, 71, 89, 77, 29, 65, 62, 62, 26, 65, 21, 49, 37, 82,
       26, 72, 26, 35, 45, 51, 63, 87, 25, 29, 72, 53, 33, 76, 65, 22, 22,
       87, 40, 46, 46, 89, 52, 55, 44, 66, 71, 78, 44, 70, 51, 73, 74, 44,
       71, 53, 84, 76, 61, 76, 33])
```

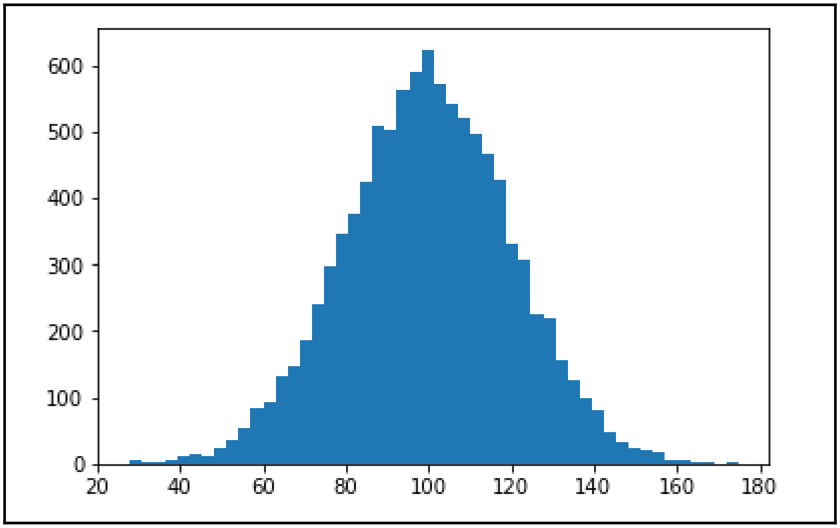


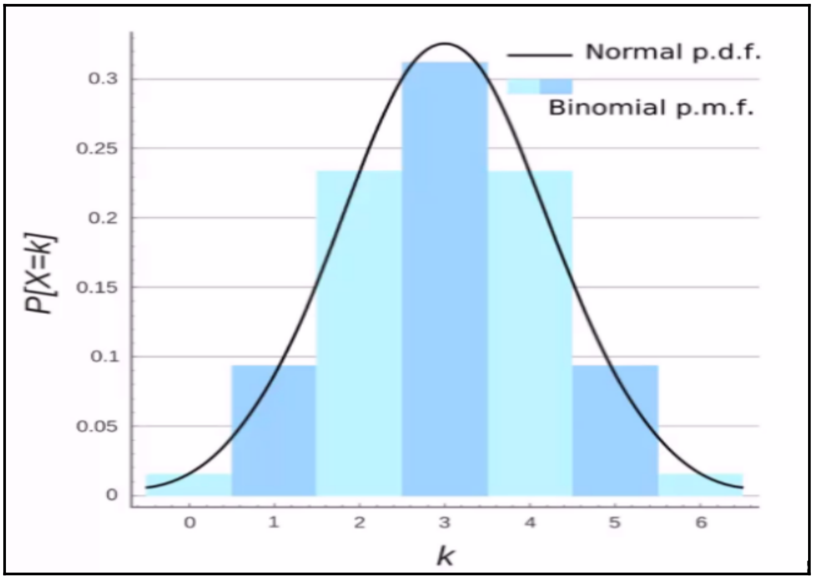
σ^2= (11.56 + 0.16 + 0.36 + 0.16 + 12.96) / 5 = 5.04

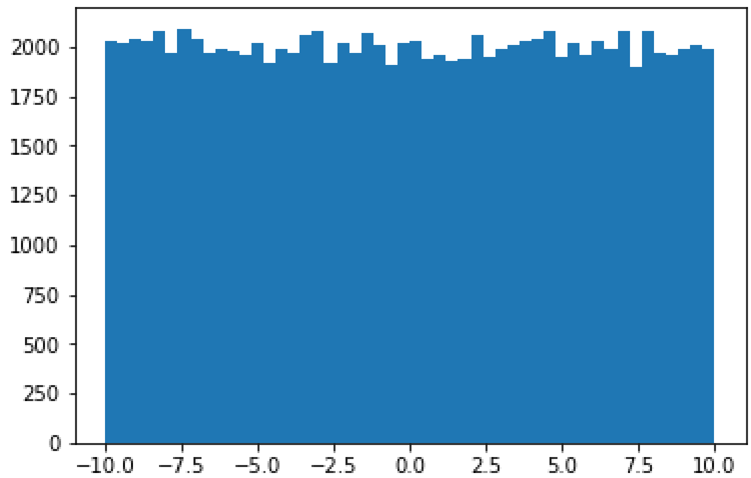


$$\sigma^2 = \frac{\sum(X-\mu)^2}{N}$$

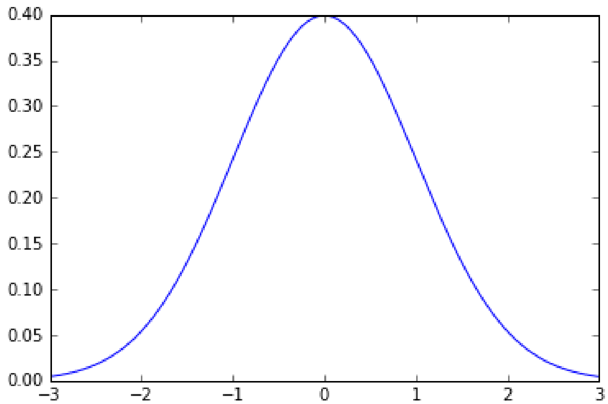
$$s^2 = \frac{\sum(X-M)^2}{N-1}$$

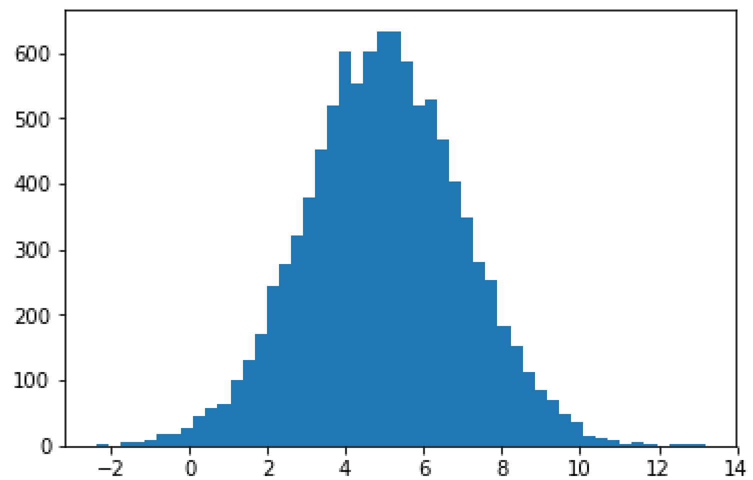

600
500
400
300
200
100
0
20
40
60
80
100
120
140
160
180


Normal p.d.f.
Binomial p.m.f.
P[X=k]
0.3
0.25
0.2
0.15
0.1
0.05
0
0
1
2
3
4
5
6
k

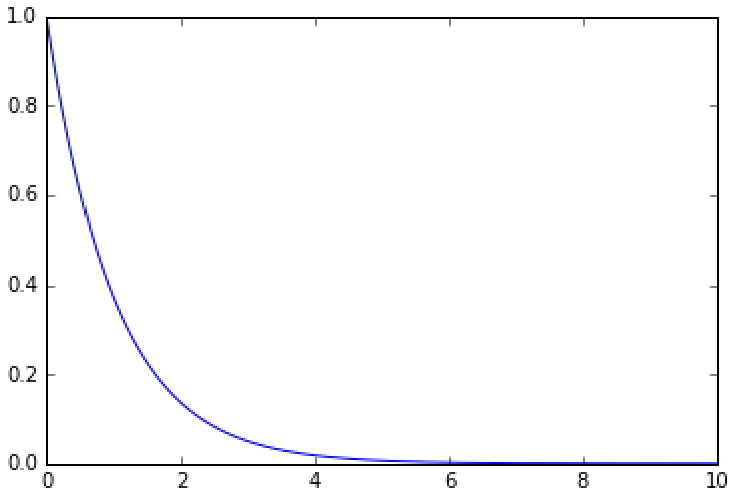


Out[3]: [<matplotlib.lines.Line2D at 0xde514e0>]

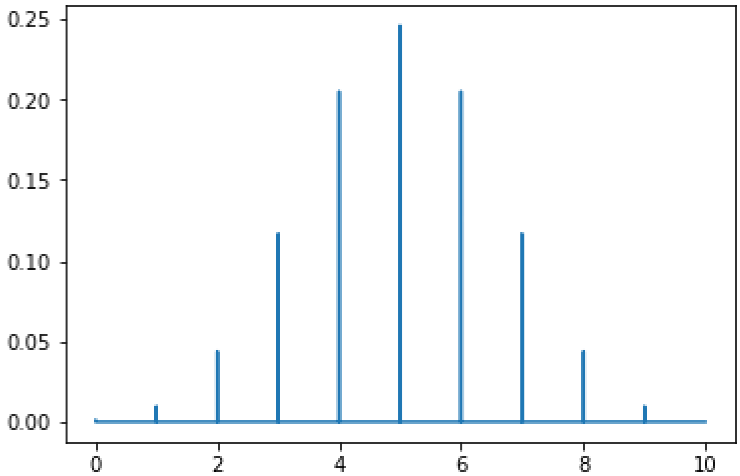




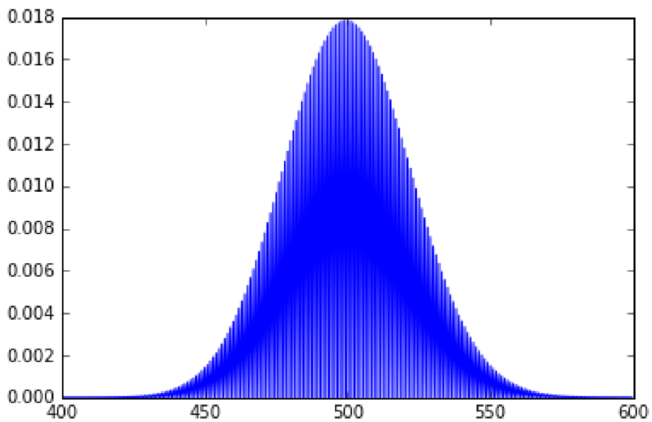
Out[5]: [<matplotlib.lines.Line2D at 0xe3304e0>]



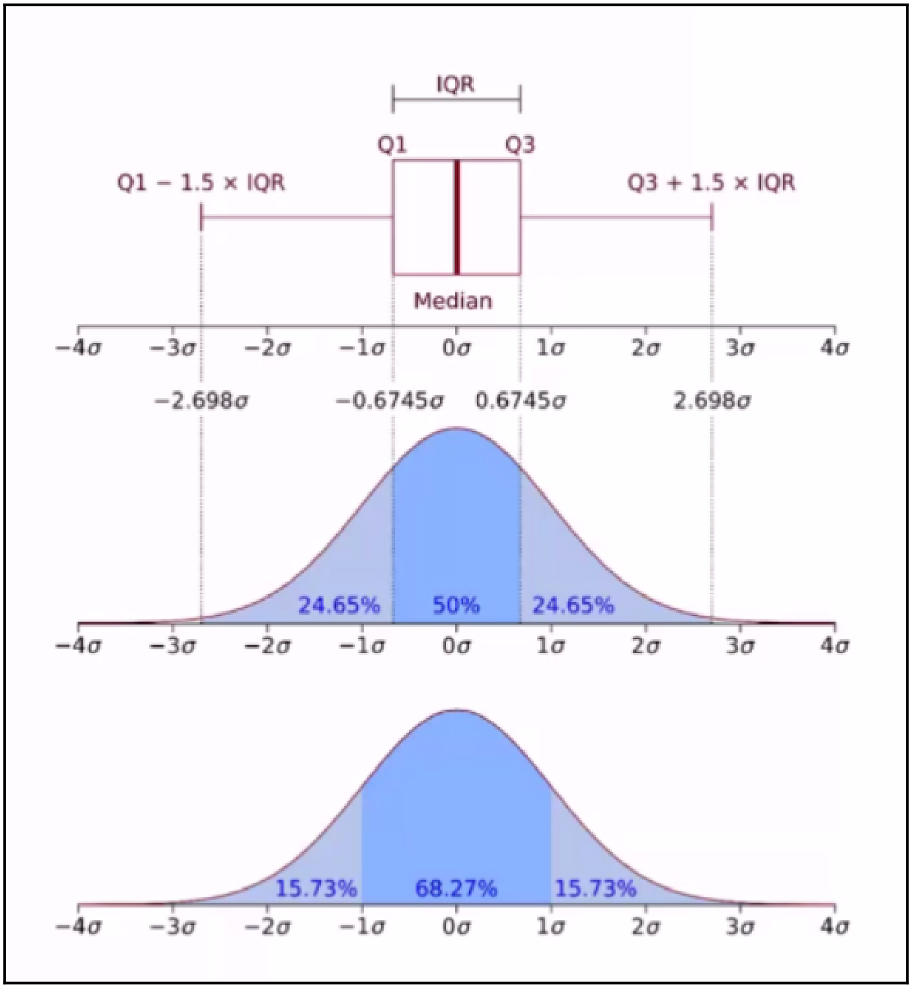
Out[5]: [<matplotlib.lines.Line2D at 0x966b128>]

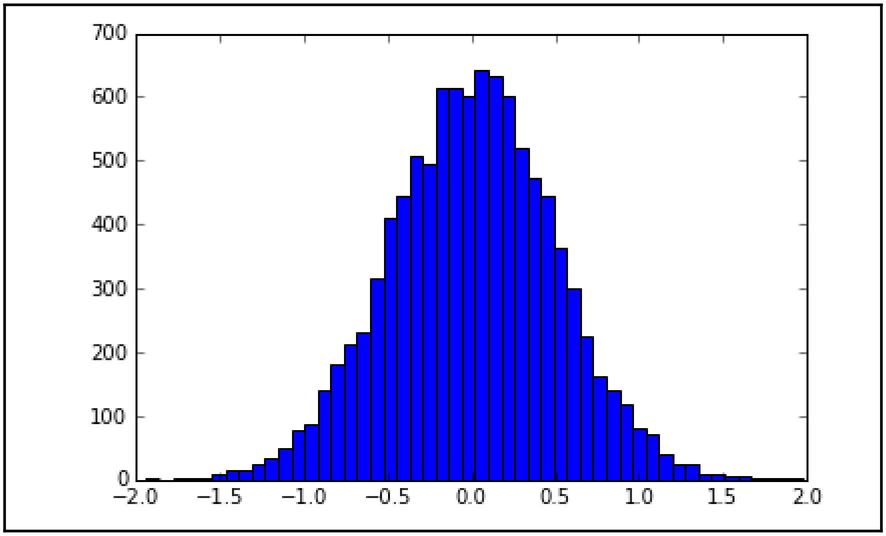


Out[7]: [<matplotlib.lines.Line2D at 0xe742e48>]

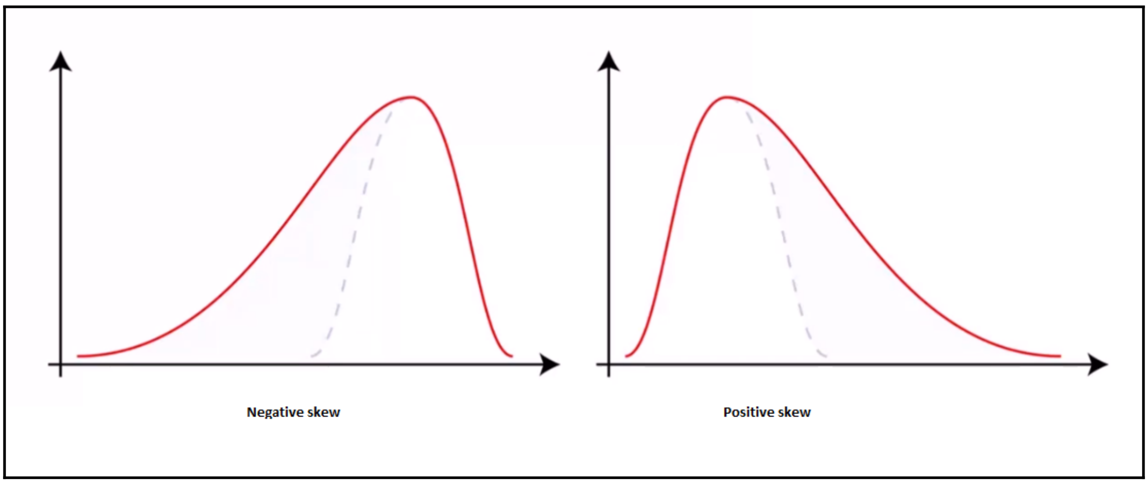


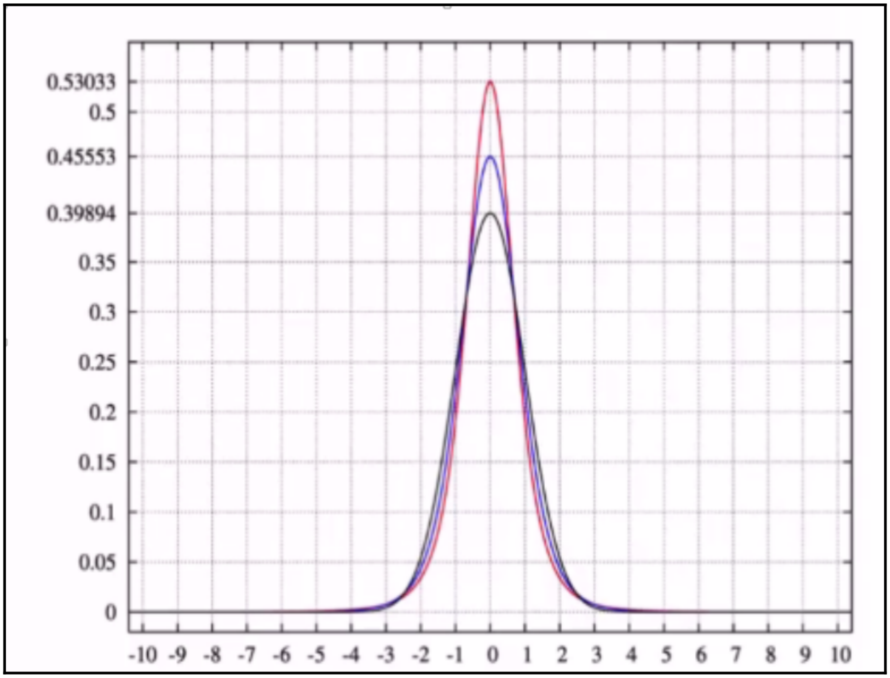

$550,000
500,000
450,000
400,000
350,000
300,000
250,000
200,000
150,000
100,000
50,000
0
99th
$506,553
75th
$85,811
50th
$42,327
25th
$19,375
1
10
20
30
40
50
60
70
80
90
99
percentile

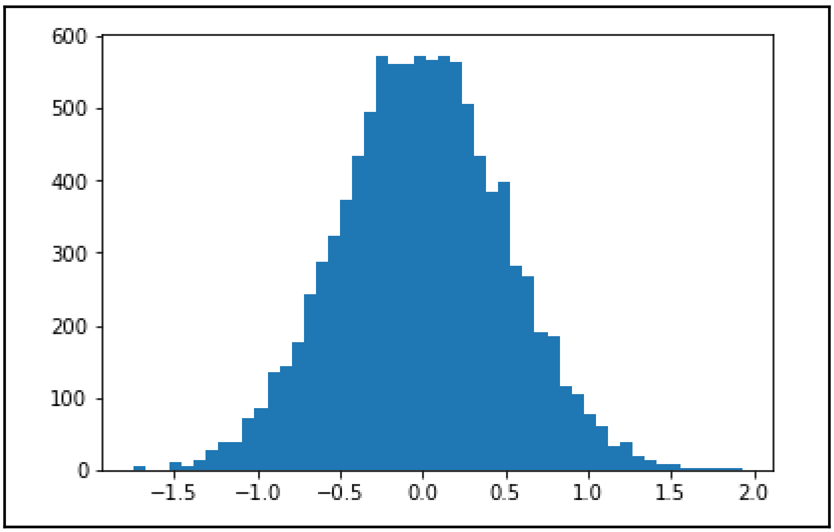

IQR
Q1
Q3
Q1 − 1.5 × IQR
Q3 + 1.5 × IQR
Median
−4σ −3σ −2σ −1σ 0σ 1σ 2σ 3σ 4σ
−2.698σ
−0.6745σ
0.6745σ
2.698σ
24.65%
50%
24.65%
−4σ −3σ −2σ −1σ 0σ 1σ 2σ 3σ 4σ
15.73%
68.27%
15.73%
−4σ −3σ −2σ −1σ 0σ 1σ 2σ 3σ 4σ



$\mu_n = \int_{-\infty}^{\infty} (x - c)^n f(x)dx$ (for moment n around value c)

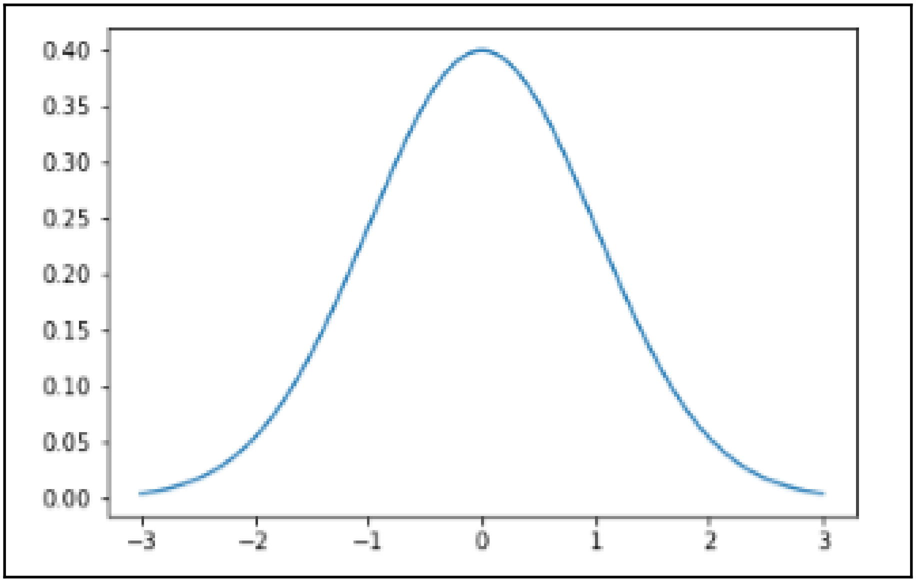


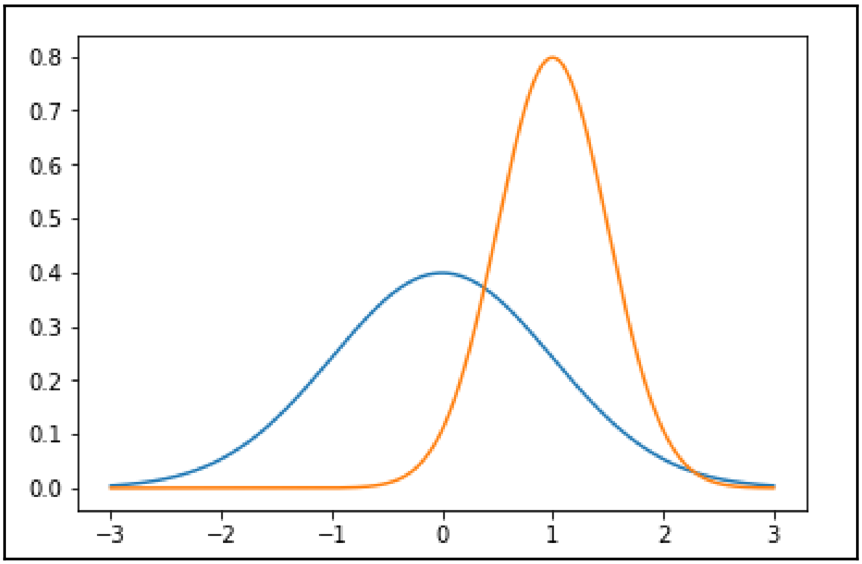


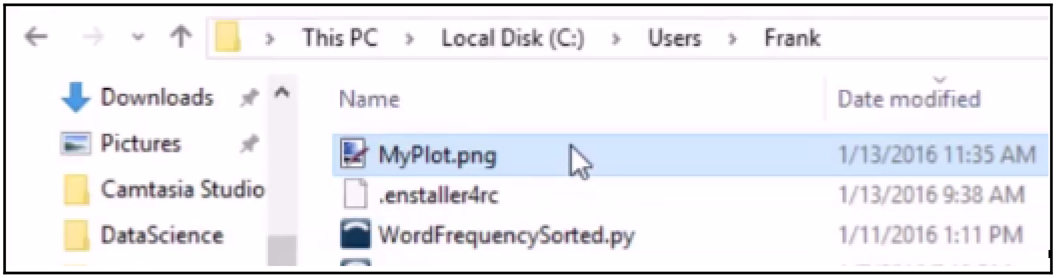


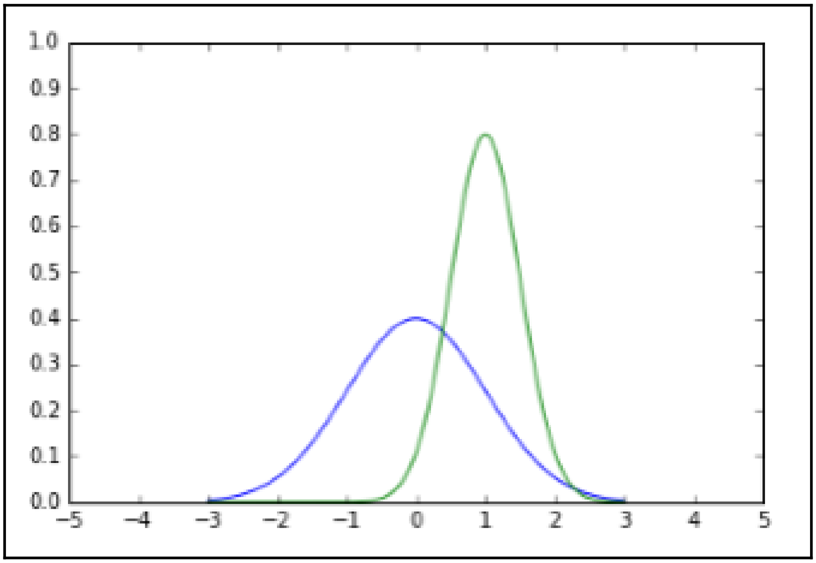
Chapter 3: Matplotlib and Advanced

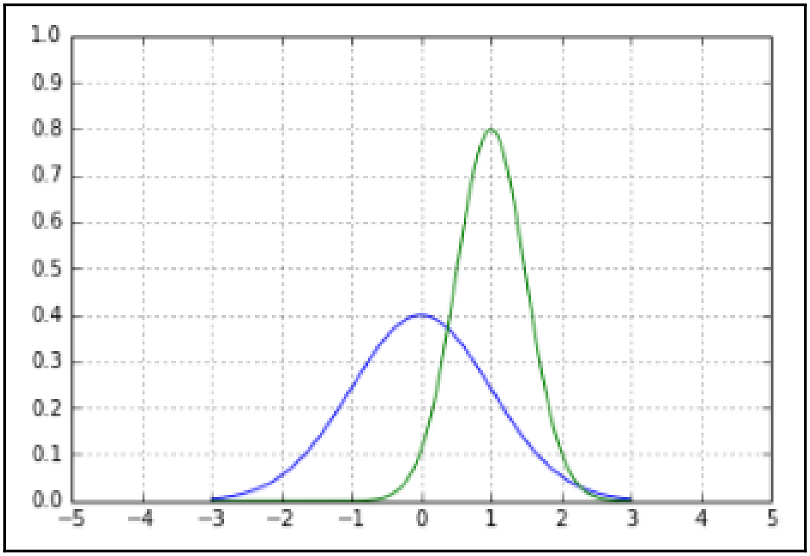
Probability Concepts

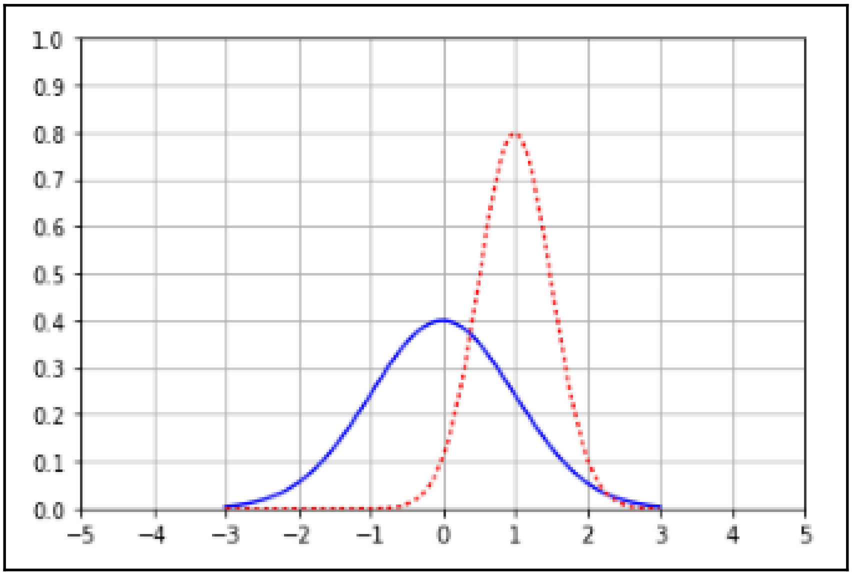


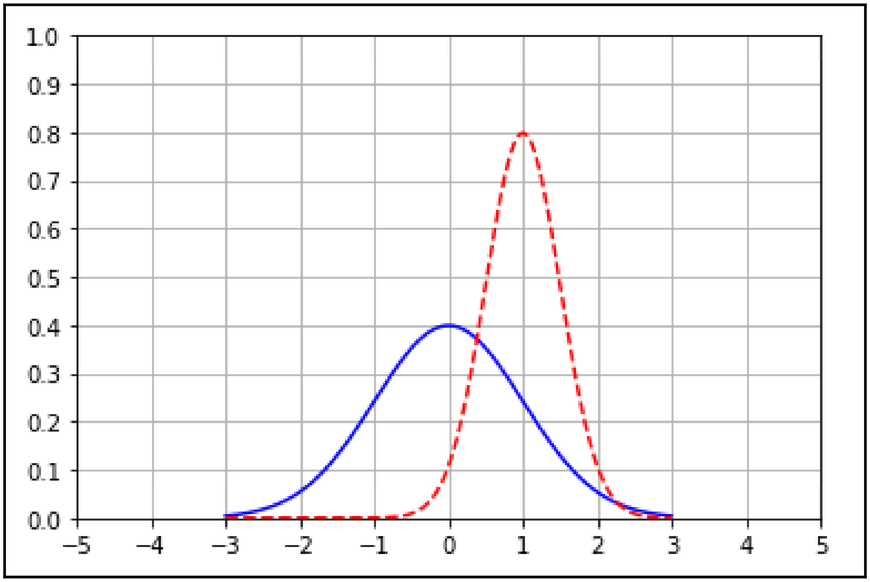



This PC
Local Disk (C:)
Users
Frank
Downloads
Pictures
Camtasia Studio
DataScience
Name
Date modified
MyPlot.png
1/13/2016 11:35 AM
.enstaller4rc
1/13/2016 9:38 AM
WordFrequencySorted.py
1/11/2016 1:11 PM

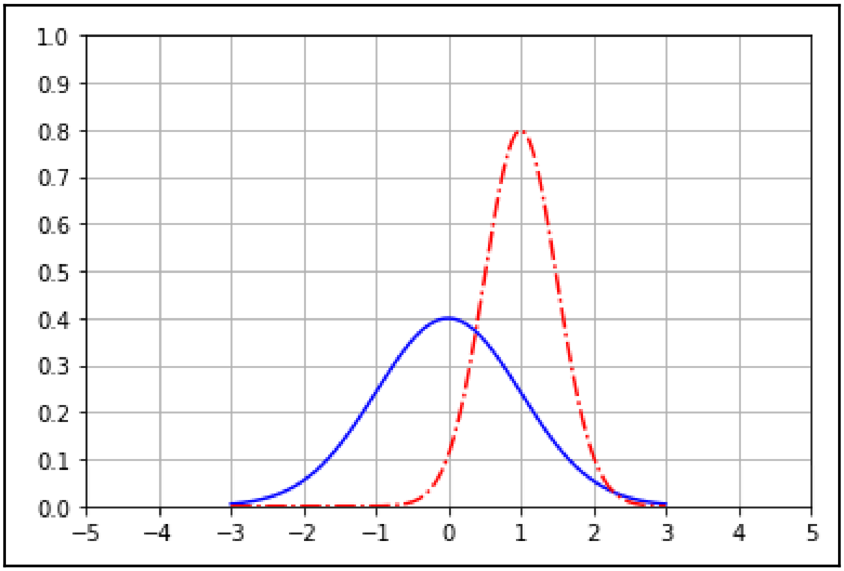

1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−5 −4 −3 −2 −1 0 1 2 3 4 5


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−5
−4
−3
−2
−1
0
1
2
3
4
5

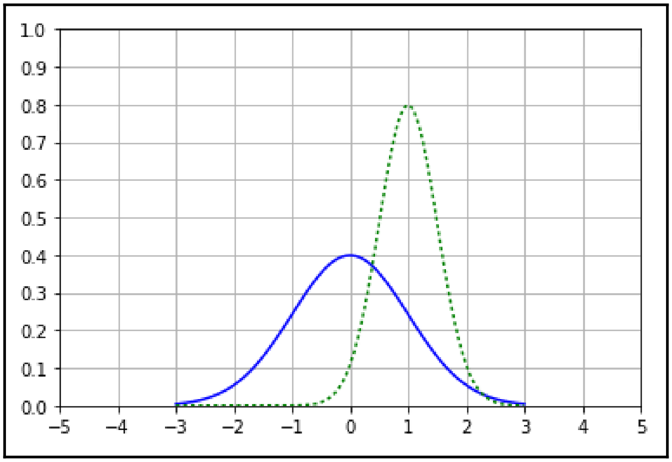

1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−5
−4
−3
−2
−1
0
1
2
3
4
5

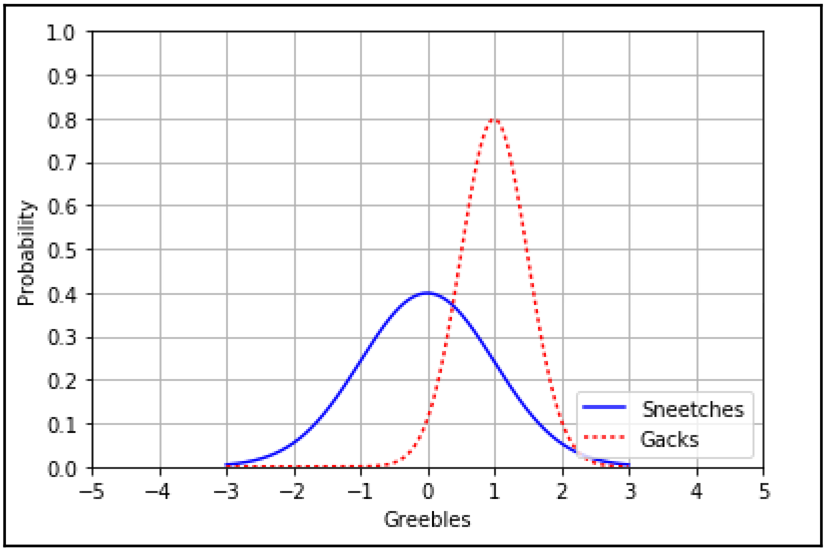


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−5 −4 −3 −2 −1 0 1 2 3 4 5

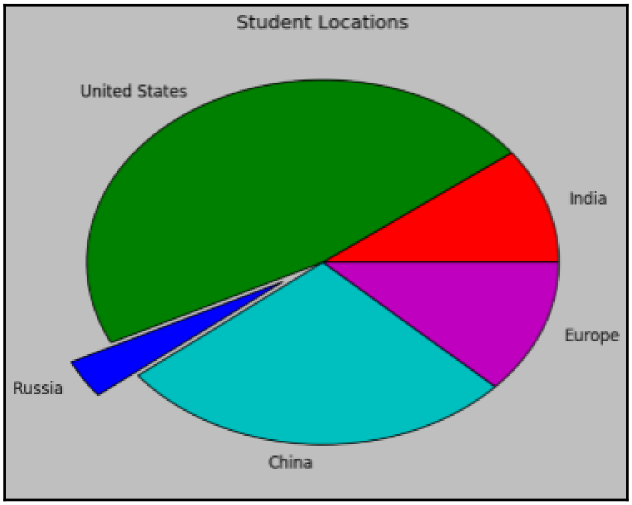


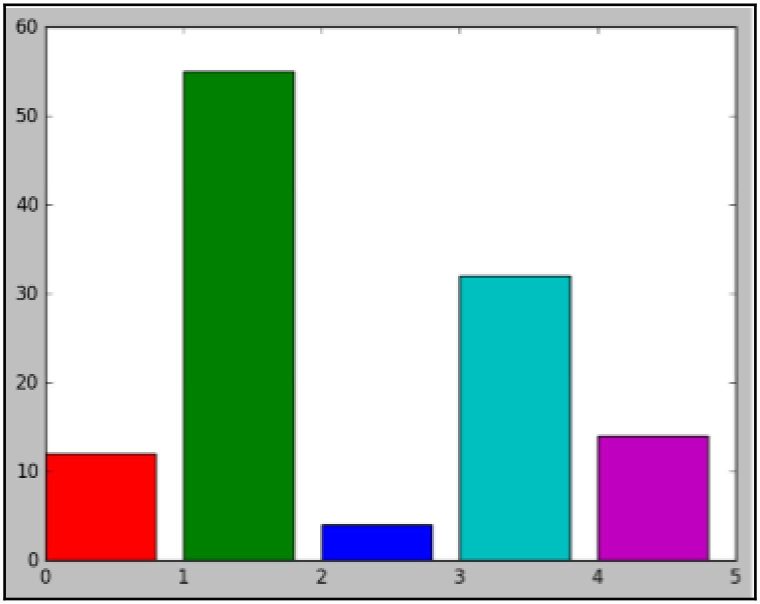
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−5 −4 −3 −2 −1 0 1 2 3 4 5

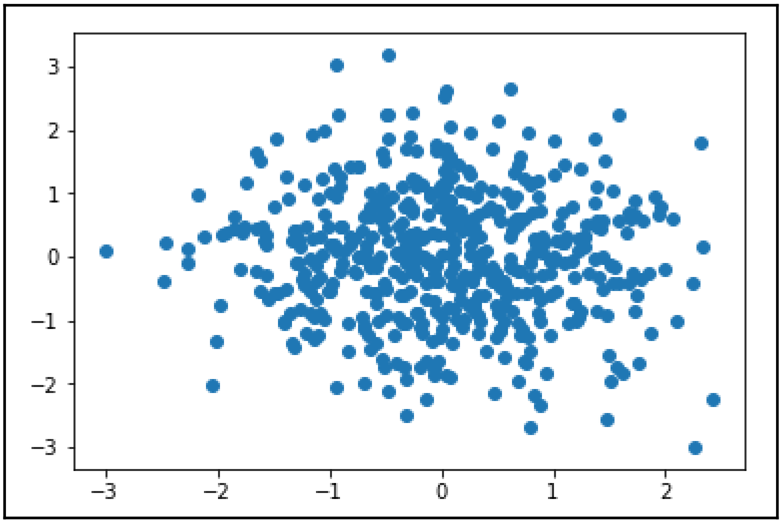

1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−5 −4 −3 −2 −1 0 1 2 3 4 5

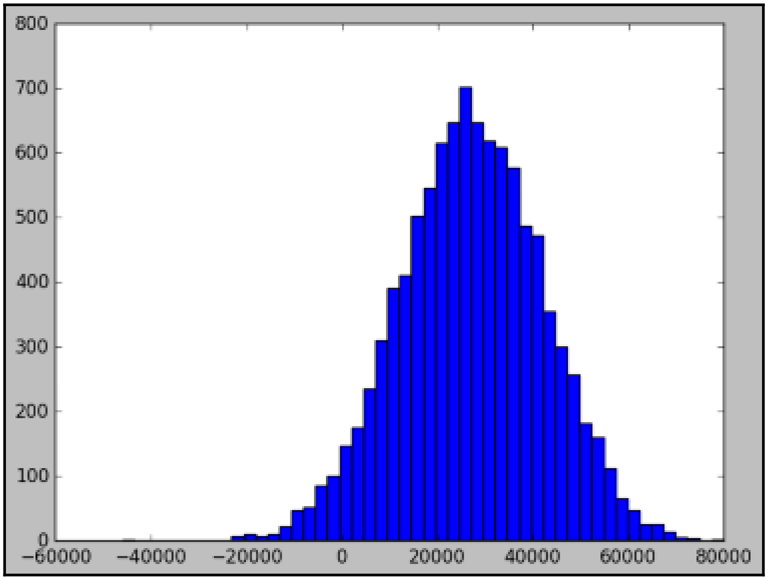

Probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−5 −4 −3 −2 −1 0 1 2 3 4 5
Greebles
Sneetches
Gacks

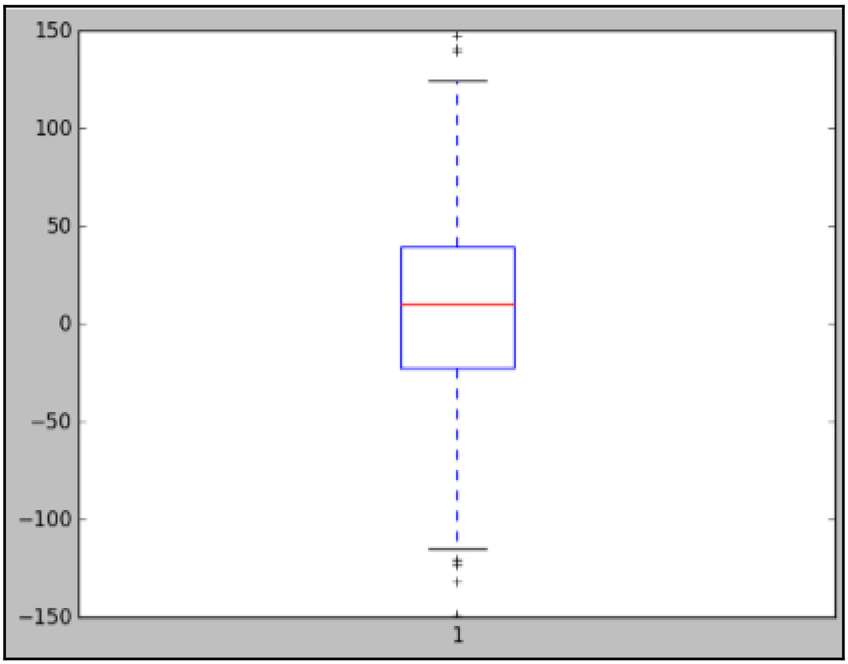

THE DAY I REALIZED
I COULD COOK BACON
WHENEVER I WANTED
my overall health
time

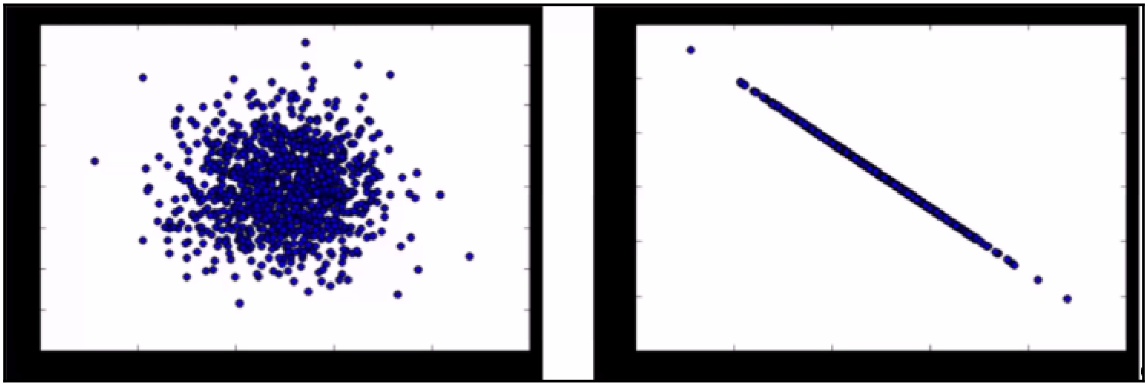

Student Locations
United States
India
Europe
China
Russia


60
50
40
30
20
10
0
0
1
2
3
4
5

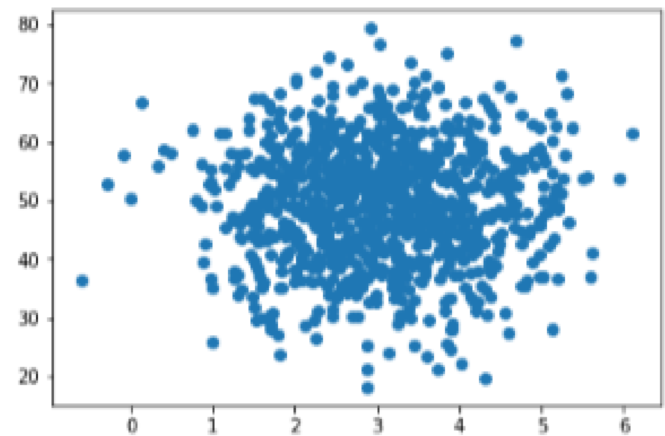

3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2


800
700
600
500
400
300
200
100
0
−60000
−40000
−20000
0
20000
40000
60000
80000

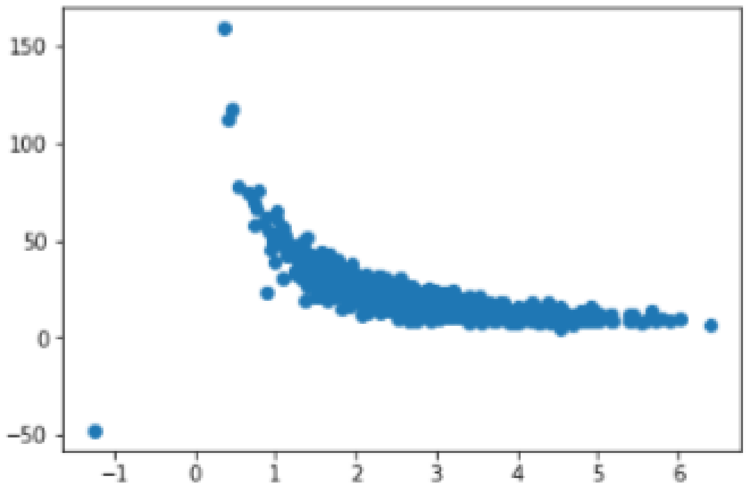

150
100
50
0
−50
−100
−150
1



Out[14]: 0.07555793784490449 9

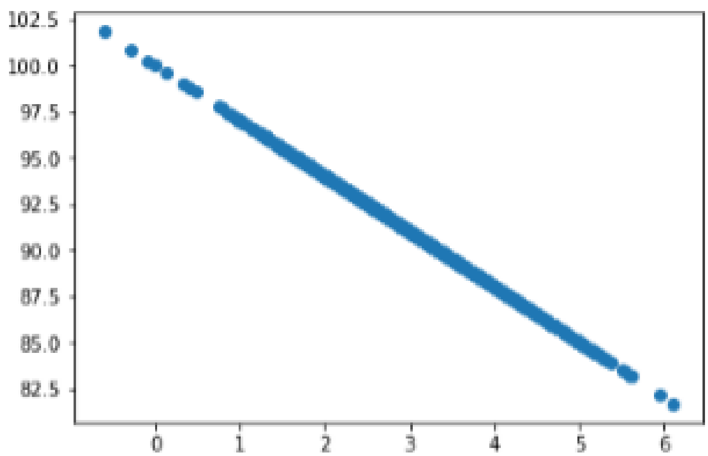


```
Out[3]: -8.1514752031255622
```




```
Out[3]: -0.46775563114087165
```

```
Out[28]: -1.0010010010010011
```



$$P(B|A) = \frac{P(A,B)}{P(A)}$$

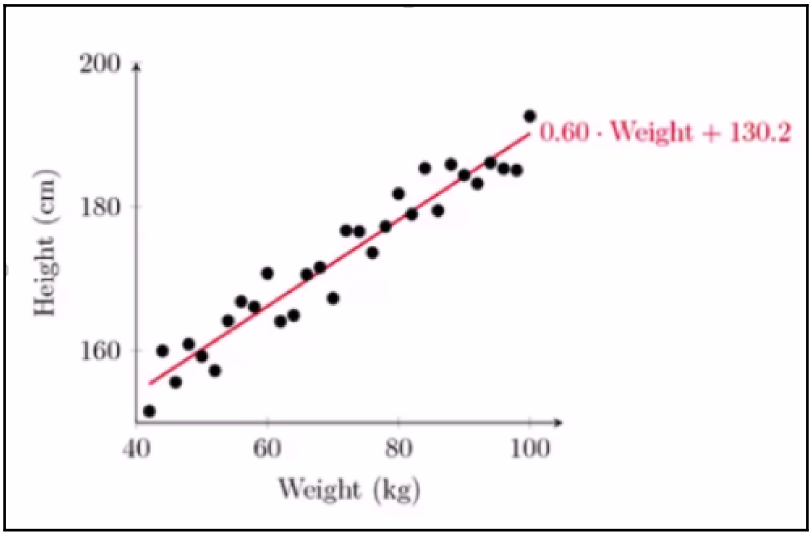
$$P(B|A) = \frac{P(A,B)}{P(A)} = \frac{0.6}{0.8} = 0.75$$

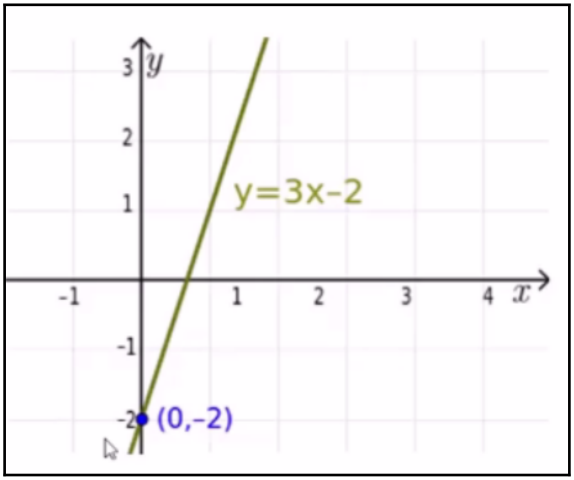
```
In [2]: totals
Out[2]: {20: 16576, 30: 16619, 40: 16632, 50: 16805, 60: 16664, 70: 16704}

In [3]: purchases
Out[3]: {20: 3392, 30: 4974, 40: 6670, 50: 8319, 60: 9944, 70: 11713}

In [4]: totalPurchases
Out[4]: 45012
```

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)}$$

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)} = \frac{0.003 * 0.99}{0.013} = 22.8\%$$

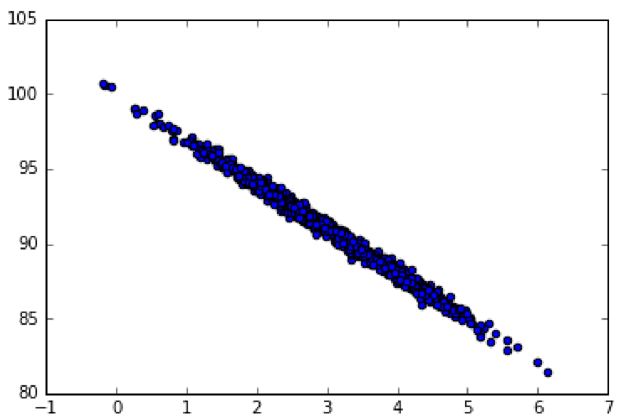
Chapter 4: Predictive Models



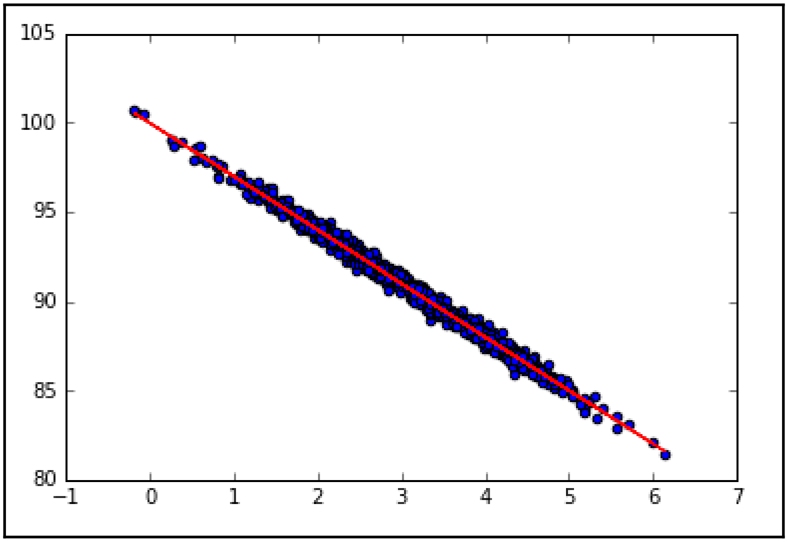


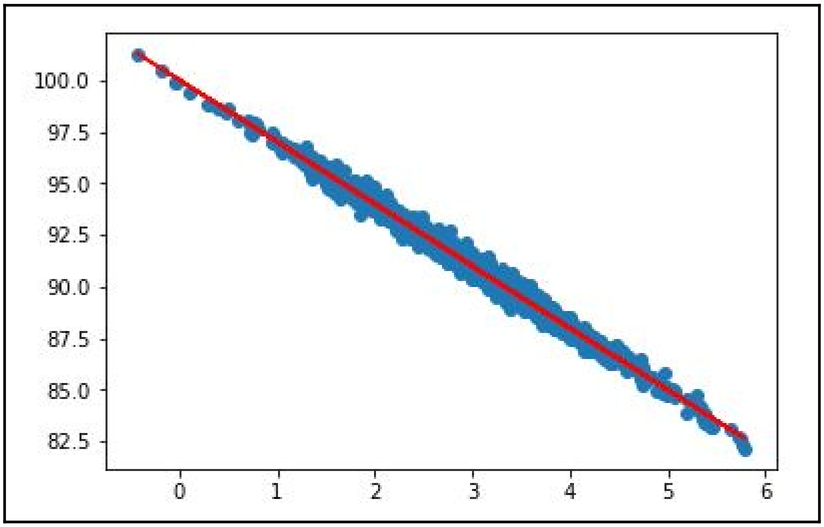
$$1.0 - \frac{sum\ of\ squared\ errors}{sum\ of\ squared\ variation\ from\ mean}$$

Out[5]: <matplotlib.collections.PathCollection at 0x8956c50>

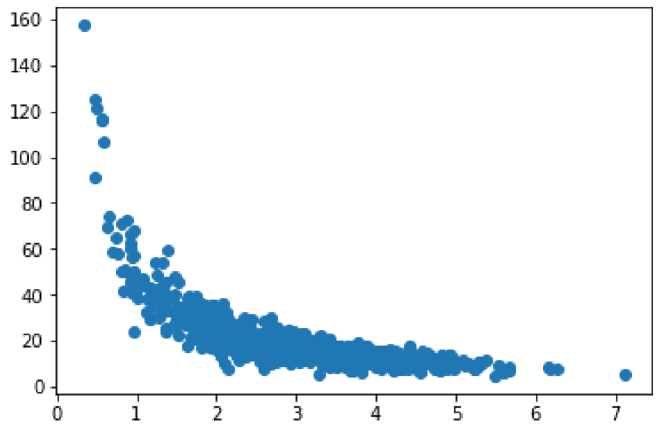


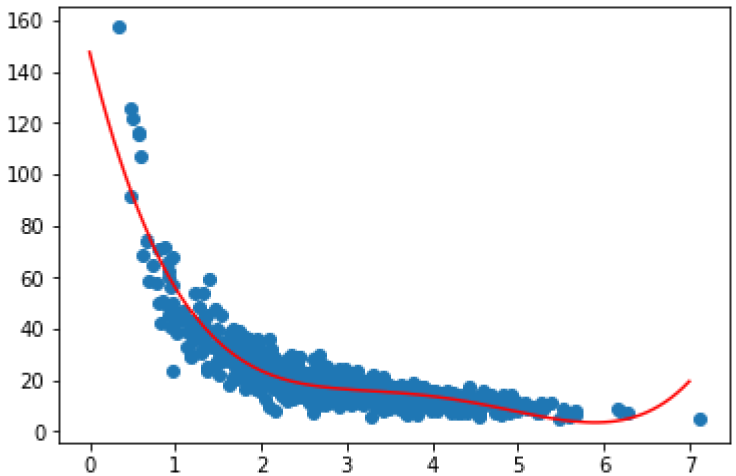
Out[4]: 0.98984146047689425


105
100
95
90
85
80
−1
0
1
2
3
4
5
6
7

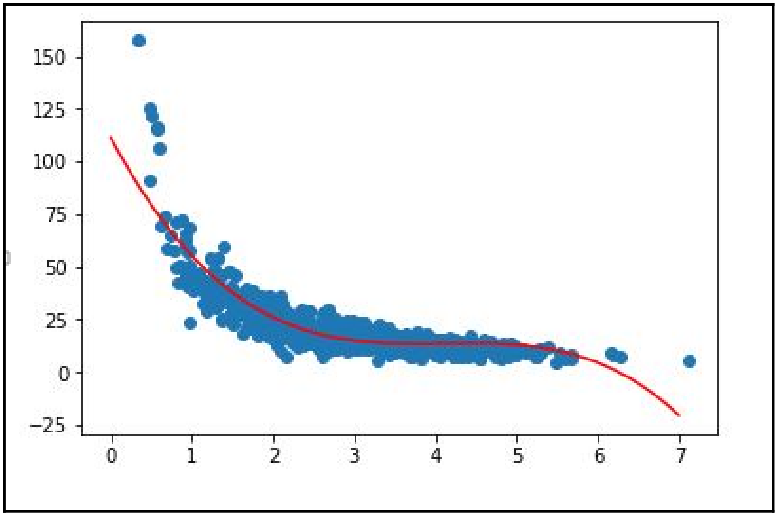

100.0
97.5
95.0
92.5
90.0
87.5
85.0
82.5
0
1
2
3
4
5
6

```
Out[1]: <matplotlib.collections.PathCollection at 0x7d8b0f0>
```






```
0.82937663963
```



For example, $price = \alpha + \beta_1$mileage + $\beta_2$age + $\beta_2$doors

Out[2]:

	Price	Mileage	Make	Model	Trim	Type	Cylinder	Liter	Doors	Cruise	Sound	Leather
0	17314.103129	8221	Buick	Century	Sedan 4D	Sedan	6	3.1	4	1	1	1
1	17542.036083	9135	Buick	Century	Sedan 4D	Sedan	6	3.1	4	1	1	0
2	16218.847862	13196	Buick	Century	Sedan 4D	Sedan	6	3.1	4	1	1	0
3	16336.913140	16342	Buick	Century	Sedan 4D	Sedan	6	3.1	4	1	0	0
4	16339.170324	19832	Buick	Century	Sedan 4D	Sedan	6	3.1	4	1	0	1

Out[3]:

OLS Regression Results

Dep. Variable:	Price	**R-squared:**	0.042
Model:	OLS	**Adj. R-squared:**	0.038
Method:	Least Squares	**F-statistic:**	11.57
Date:	Tue, 26 Jan 2016	**Prob (F-statistic):**	1.98e-07
Time:	12:18:05	**Log-Likelihood:**	-8519.1
No. Observations:	804	**AIC:**	1.705e+04
Df Residuals:	800	**BIC:**	1.706e+04
Df Model:	3		
Covariance Type:	nonrobust		

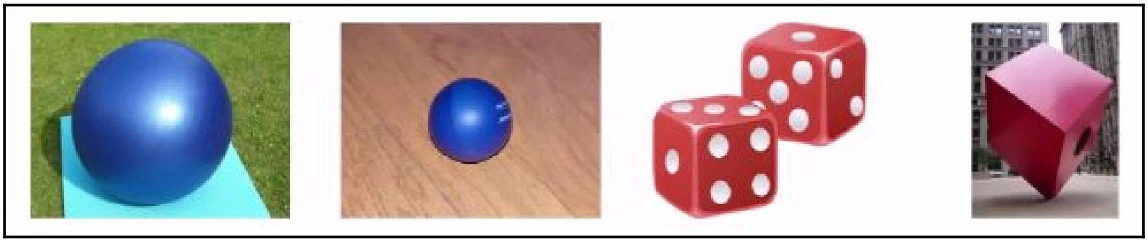
	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
const	3.125e+04	1809.549	17.272	0.000	2.77e+04 3.48e+04
Mileage	-0.1765	0.042	-4.227	0.000	-0.259 -0.095
Model_ord	-39.0387	39.326	-0.993	0.321	-116.234 38.157
Doors	-1652.9303	402.649	-4.105	0.000	-2443.303 -862.558

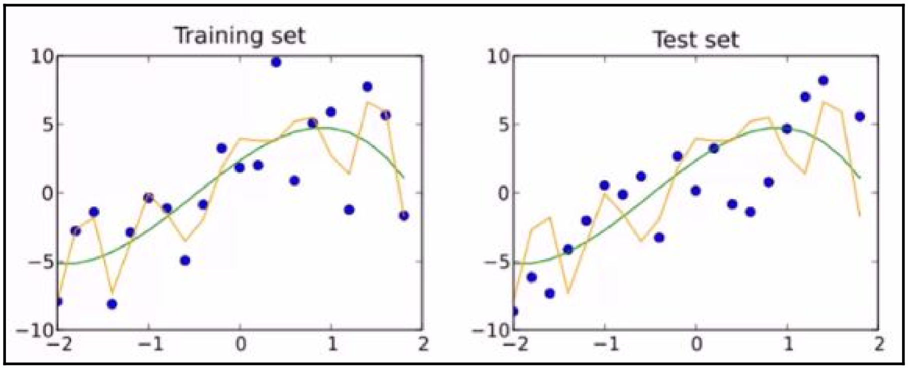
Omnibus:	206.410	**Durbin-Watson:**	0.080
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	470.872
Skew:	1.379	**Prob(JB):**	5.64e-103
Kurtosis:	5.541	**Cond. No.**	1.15e+05

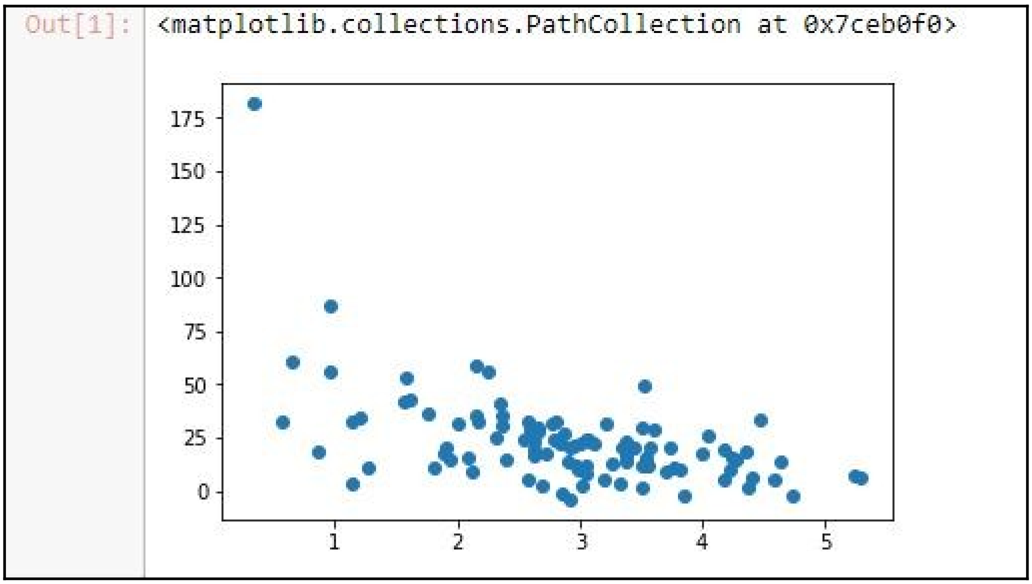
Out[5]:

	Price
Doors	
2	23807.135520
4	20580.670749

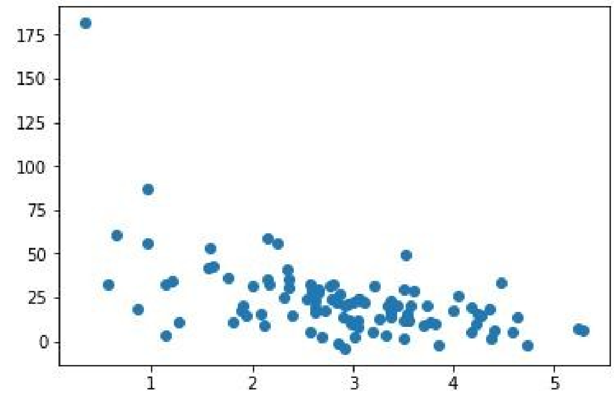
Chapter 5: Machine Learning with Python



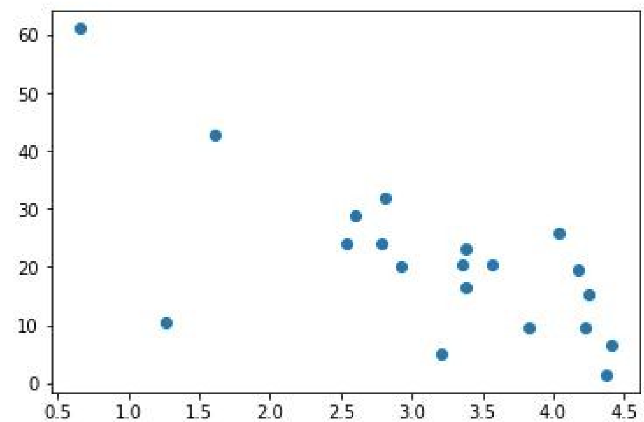


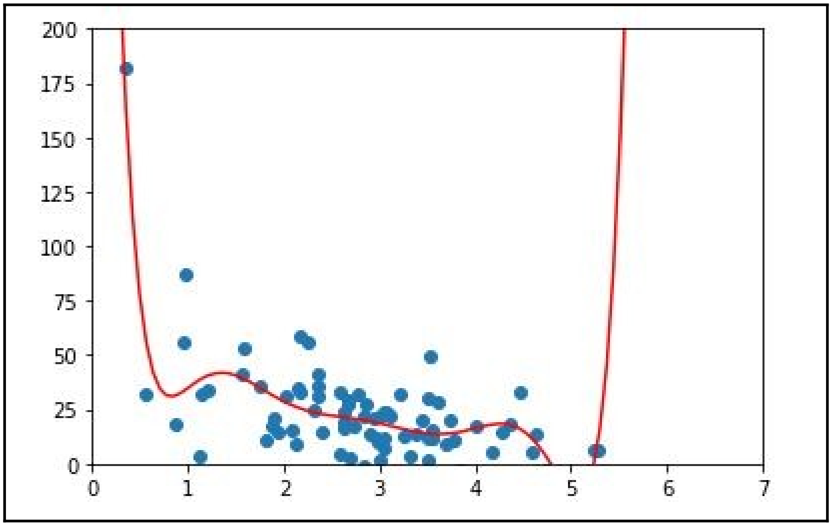


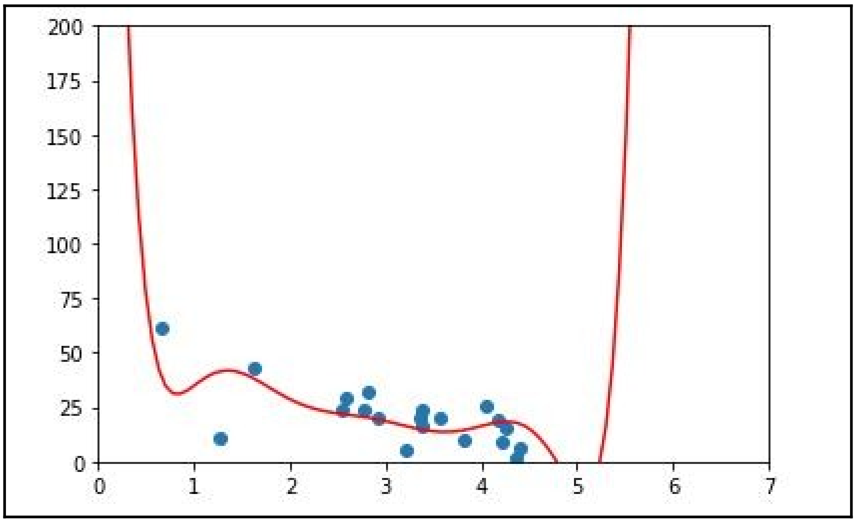
```
Out[1]: <matplotlib.collections.PathCollection at 0x7ceb0f0>
```




```
Out[4]: <matplotlib.collections.PathCollection at 0x8005710>
```







$$P(A|B) = \frac{P(A)P(B|A)}{P(B)}$$

$$P(Spam \mid Free) = \frac{P(Spam)P(Free \mid Spam)}{P(Free)}$$

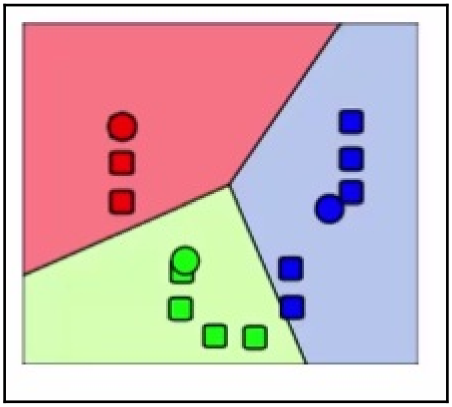
P(Free|Spam)P(Spam) + P(Free|Not Spam)P(Not Spam))

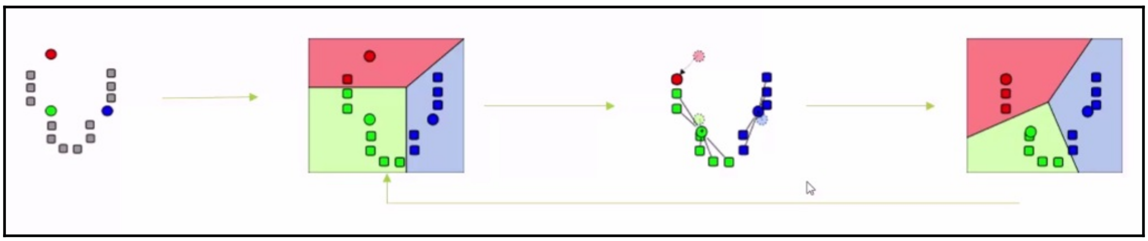
Out[12]:

	class	message
C:\Users\deveshc\Desktop\DataScience\emails\spam\00001.7848dde101aa985090474a91ec93fcf0	spam	<!DOCTYPE HTML PUBLIC "-//W3C//DTD H
C:\Users\deveshc\Desktop\DataScience\emails\spam\00002.d94f1b97e48ed3b553b3508d116e6a09	spam	1) Fight The Risk of Cancer!\n\nhttp://www.ad
C:\Users\deveshc\Desktop\DataScience\emails\spam\00003.2ee33bc6eacdb11f38d052c44819ba6c	spam	1) Fight The Risk of Cancer!\n\nhttp://www.ad
C:\Users\deveshc\Desktop\DataScience\emails\spam\00004.eac8de8d759b7e74154f142194282724	spam	##
C:\Users\deveshc\Desktop\DataScience\emails\spam\00005.57696a39d7d84318ce497886896bf90d	spam	I thought you might like these:\n\n1) Slim Dow

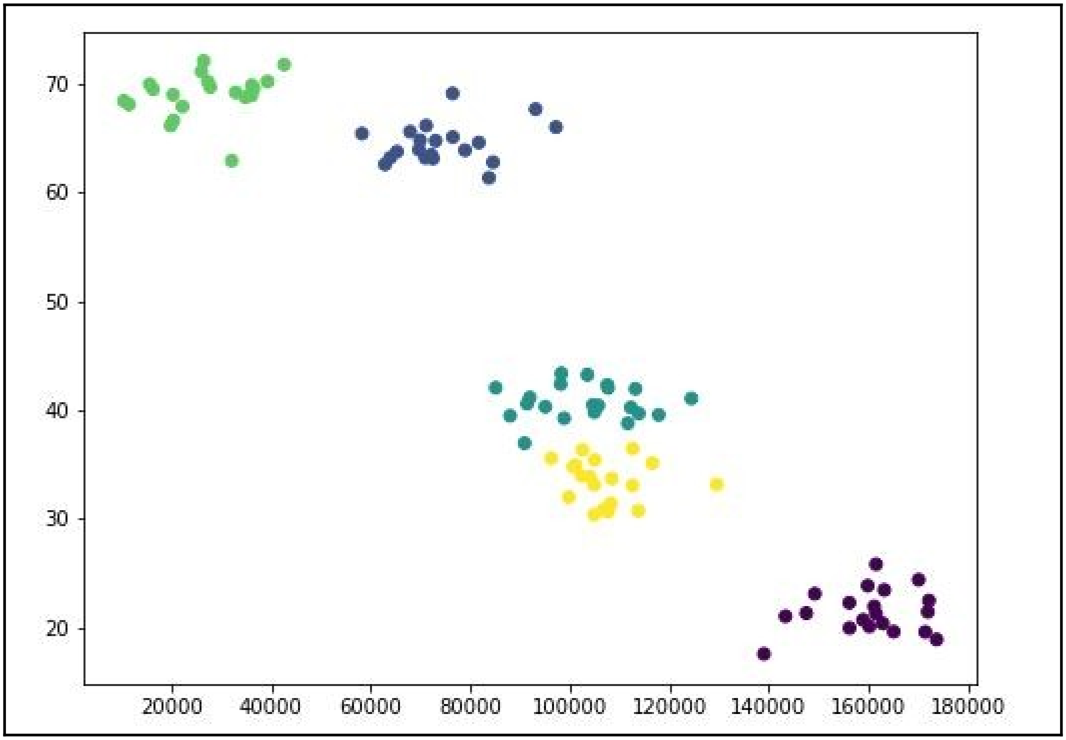
```
Out[13]: MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

```
Out[14]: array(['spam', 'ham'],
               dtype='|S4')
```



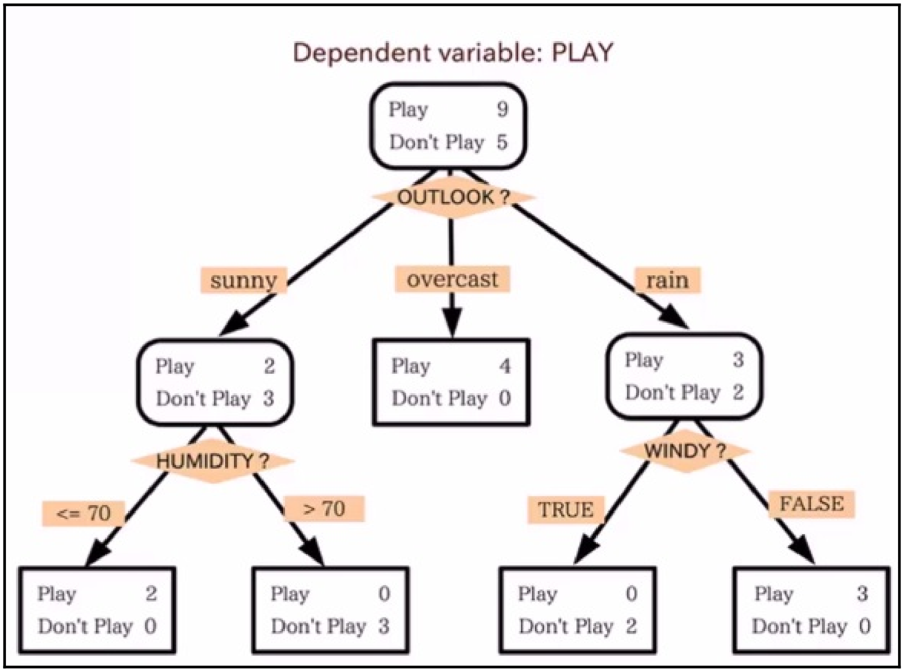


```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

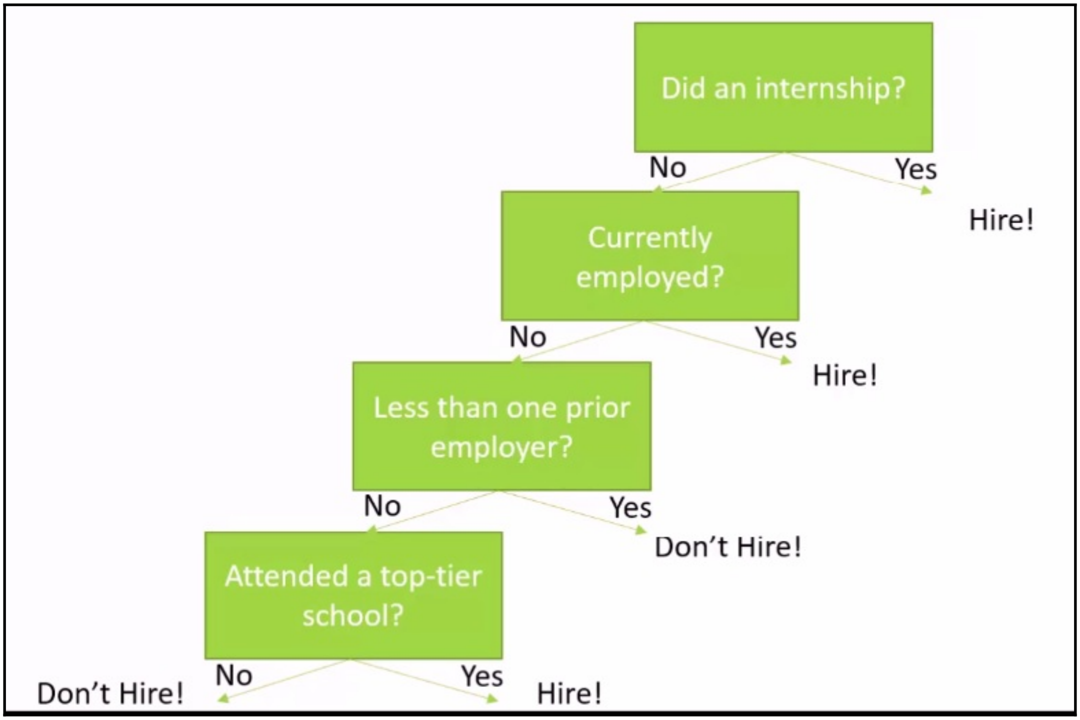


$H(S) = -p_1 \ln p_1 - \cdots - p_n \ln p_n$

p_i represents the proportion of the data labeled for each class



Candidate ID	Years Experience	Employed?	Previous employers	Level of Education	Top-tier school	Interned	Hired
0	10	1	4	0	0	0	1
1	0	0	0	0	1	1	1
2	7	0	6	0	0	0	0
3	2	1	1	1	1	0	1
4	20	0	2	2	1	0	0



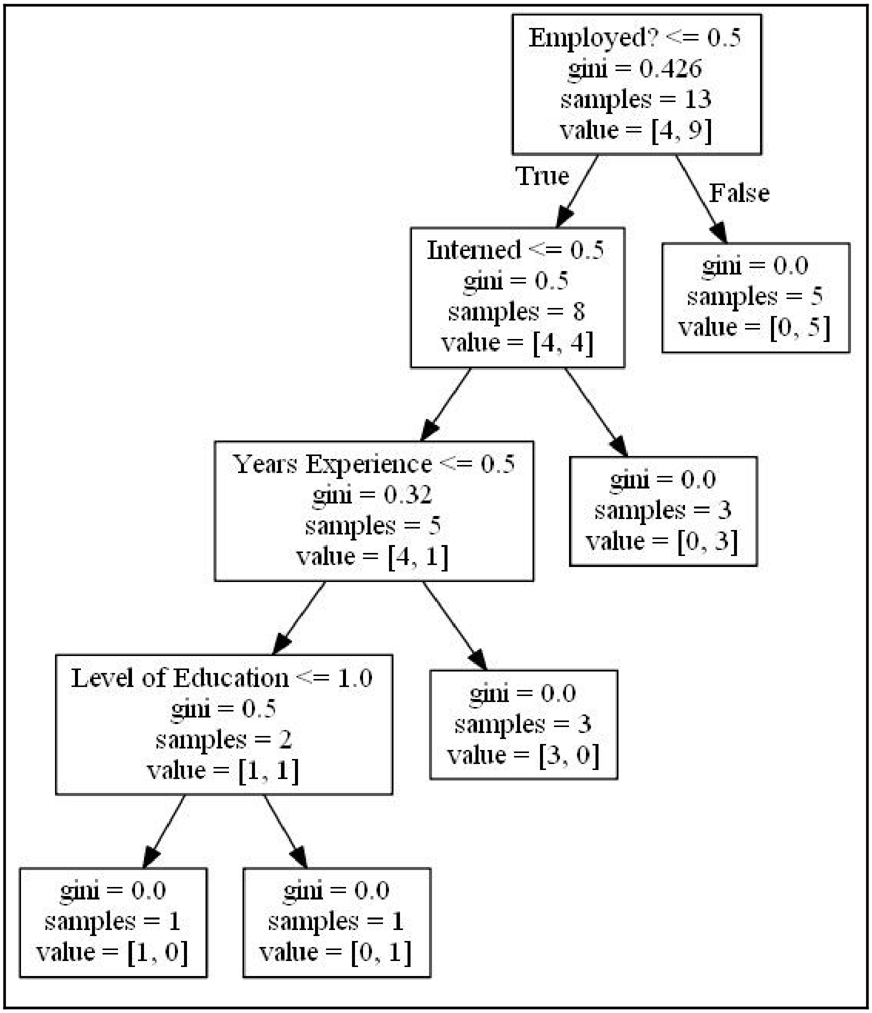
Out[2]:

	Years Experience	Employed?	Previous employers	Level of Education	Top-tier school	Interned	Hired
0	10	Y	4	BS	N	N	Y
1	0	N	0	BS	Y	Y	Y
2	7	N	6	BS	N	N	N
3	2	Y	1	MS	Y	N	Y
4	20	N	2	PhD	Y	N	N

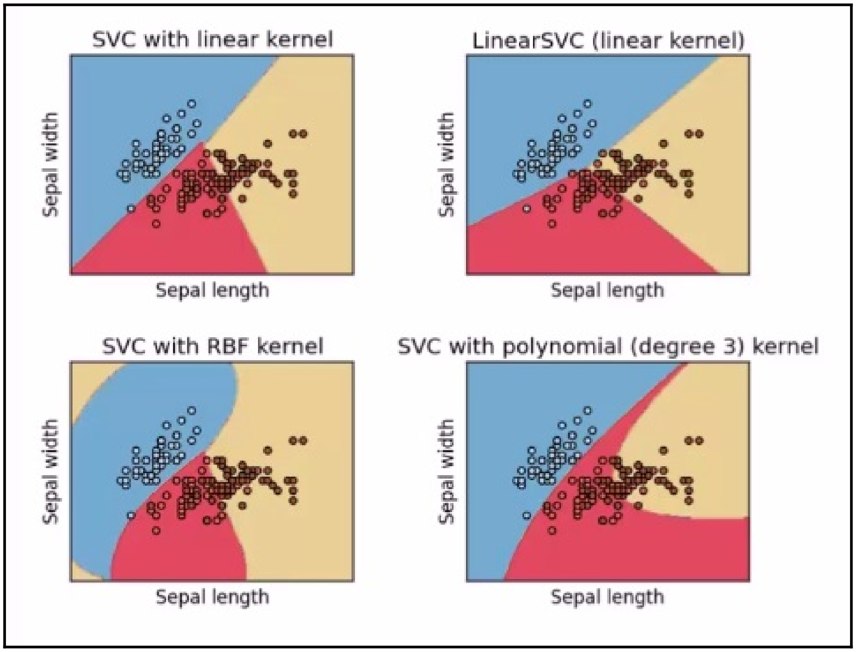
Out[3]:

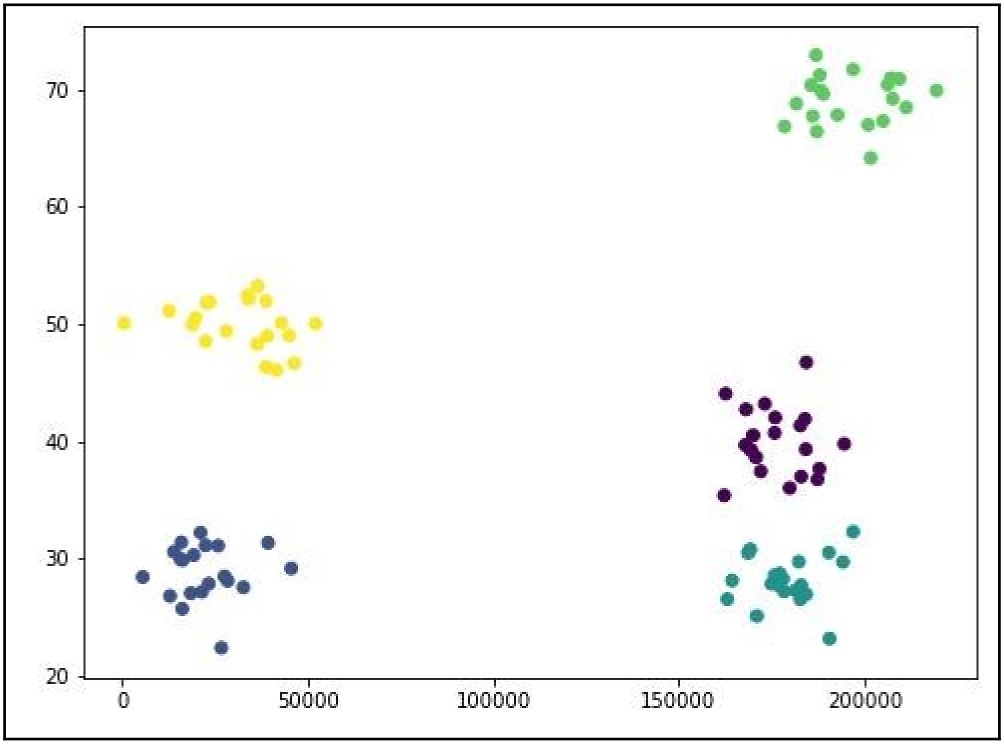
	Years Experience	Employed?	Previous employers	Level of Education	Top-tier school	Interned	Hired
0	10	1	4	0	0	0	1
1	0	0	0	0	1	1	1
2	7	0	6	0	0	0	0
3	2	1	1	1	1	0	1
4	20	0	2	2	1	0	0

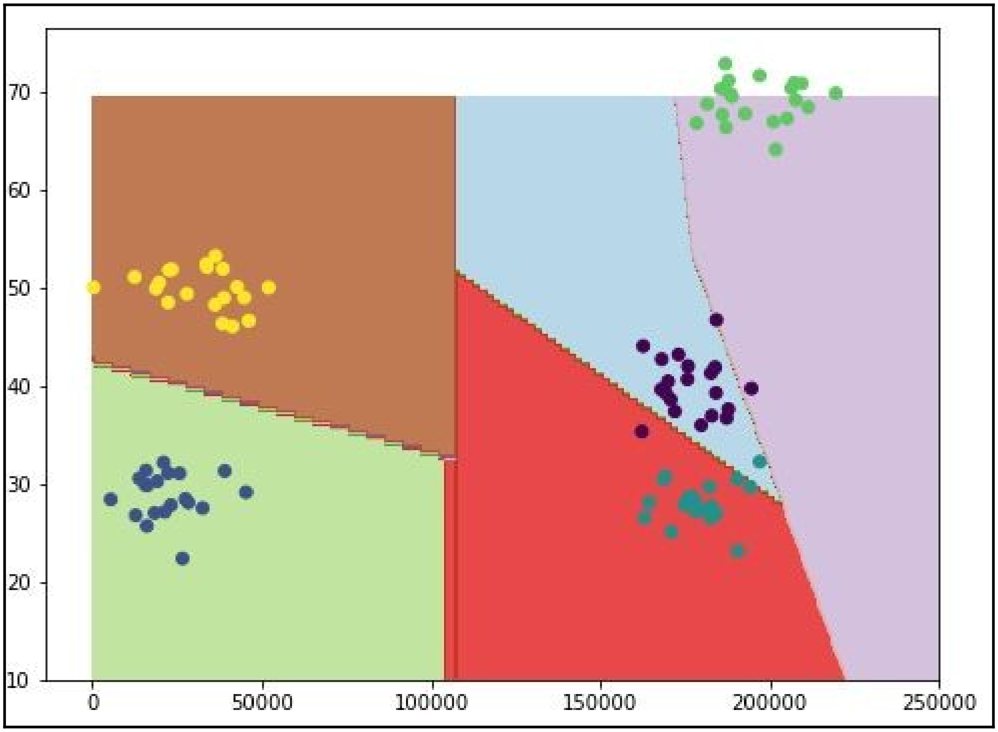
```
Out[4]: ['Years Experience',
         'Employed?',
         'Previous employers',
         'Level of Education',
         'Top-tier school',
         'Interned']
```



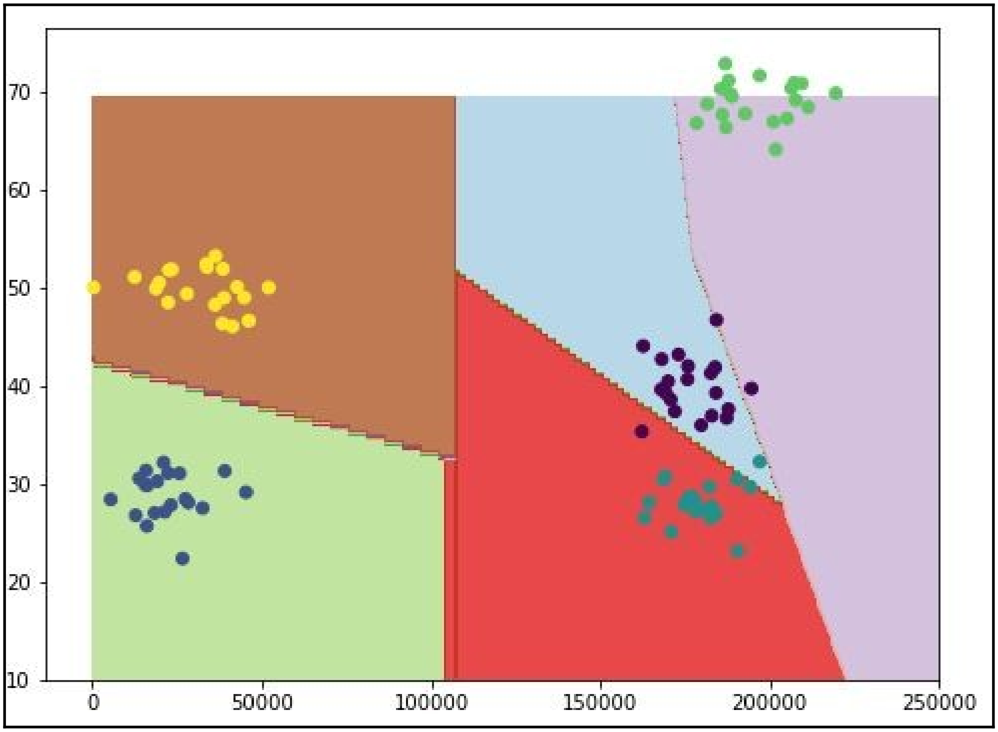

```
[1]
[1]
```


SVC with linear kernel
LinearSVC (linear kernel)
SVC with RBF kernel
SVC with polynomial (degree 3) kernel
Sepal width
Sepal length


70
60
50
40
30
20
0
50000
100000
150000
200000

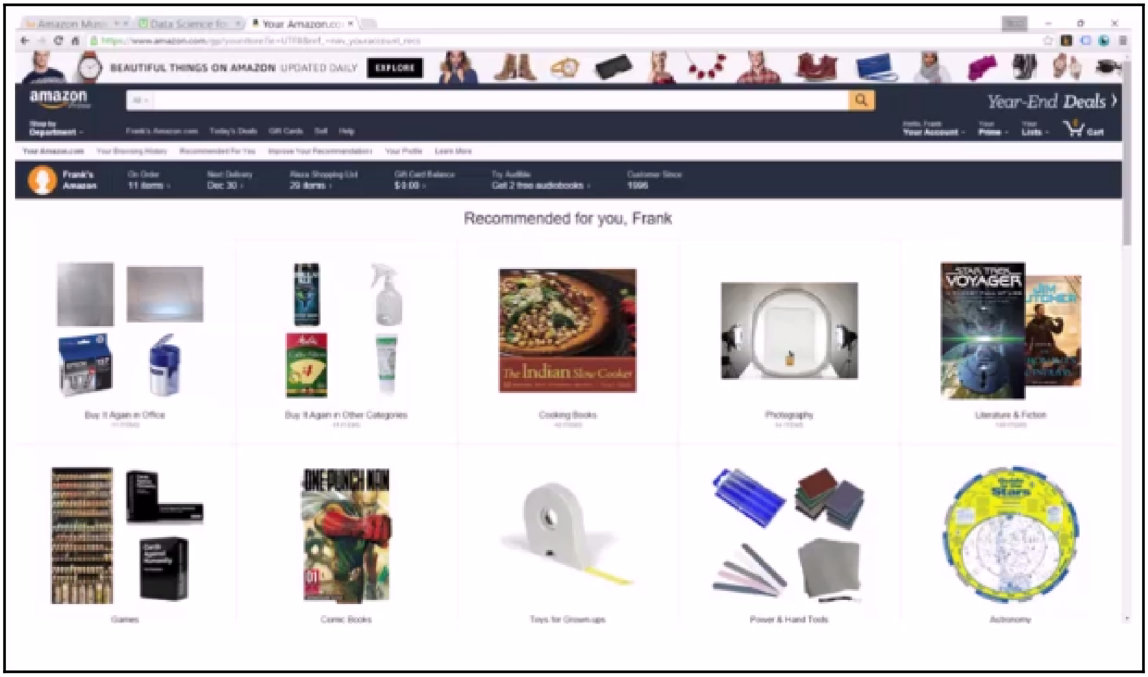


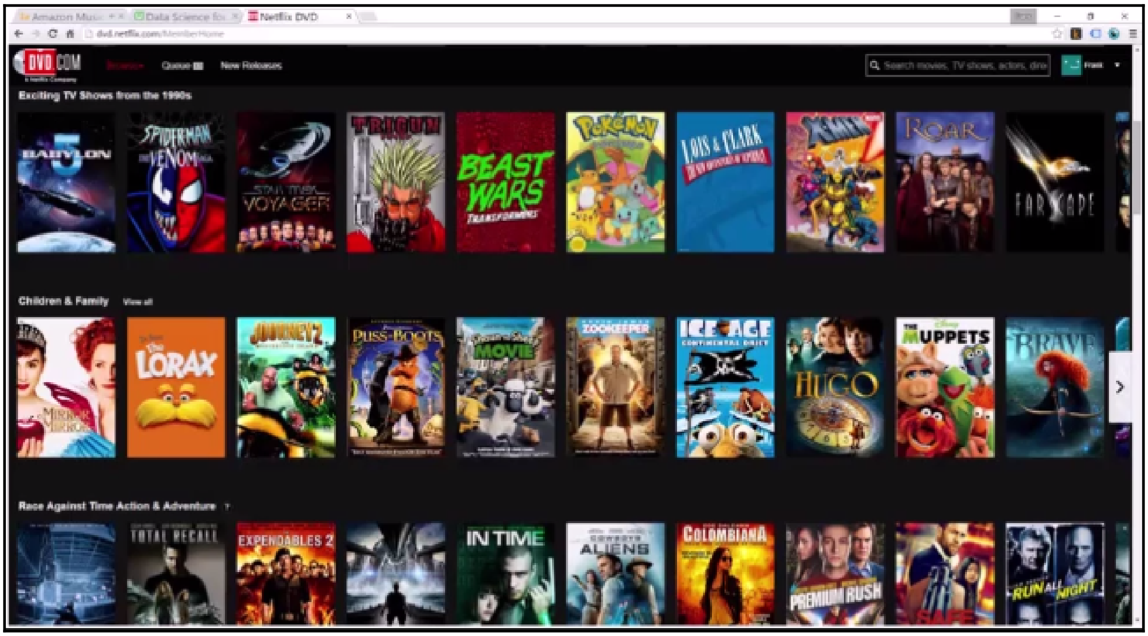
70
60
50
40
30
20
10
0
50000
100000
150000
200000
250000

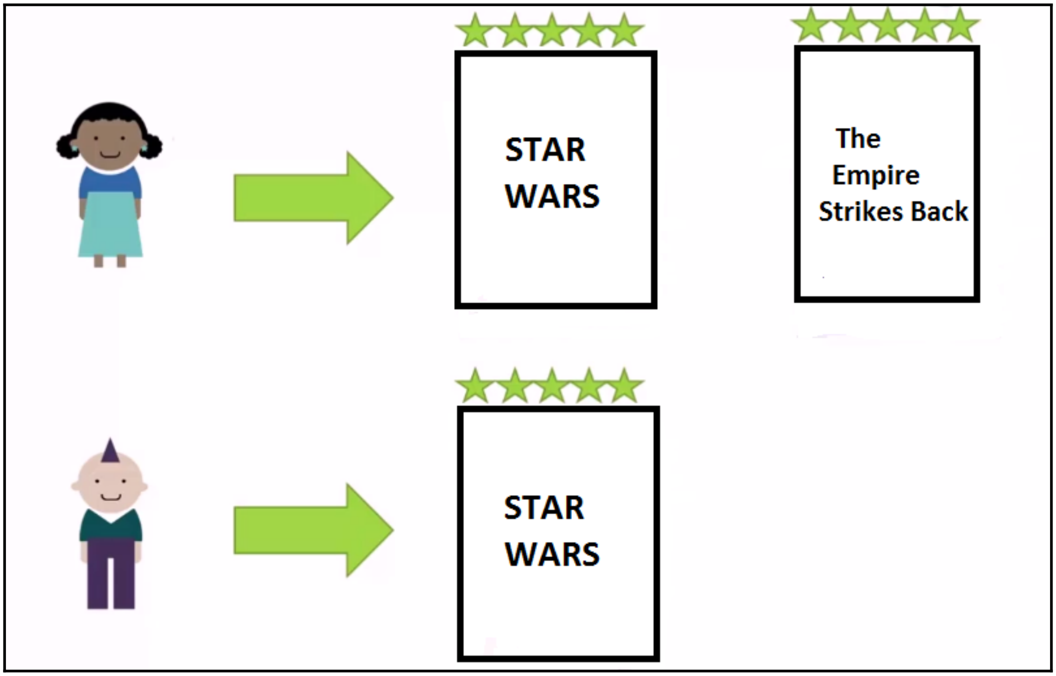


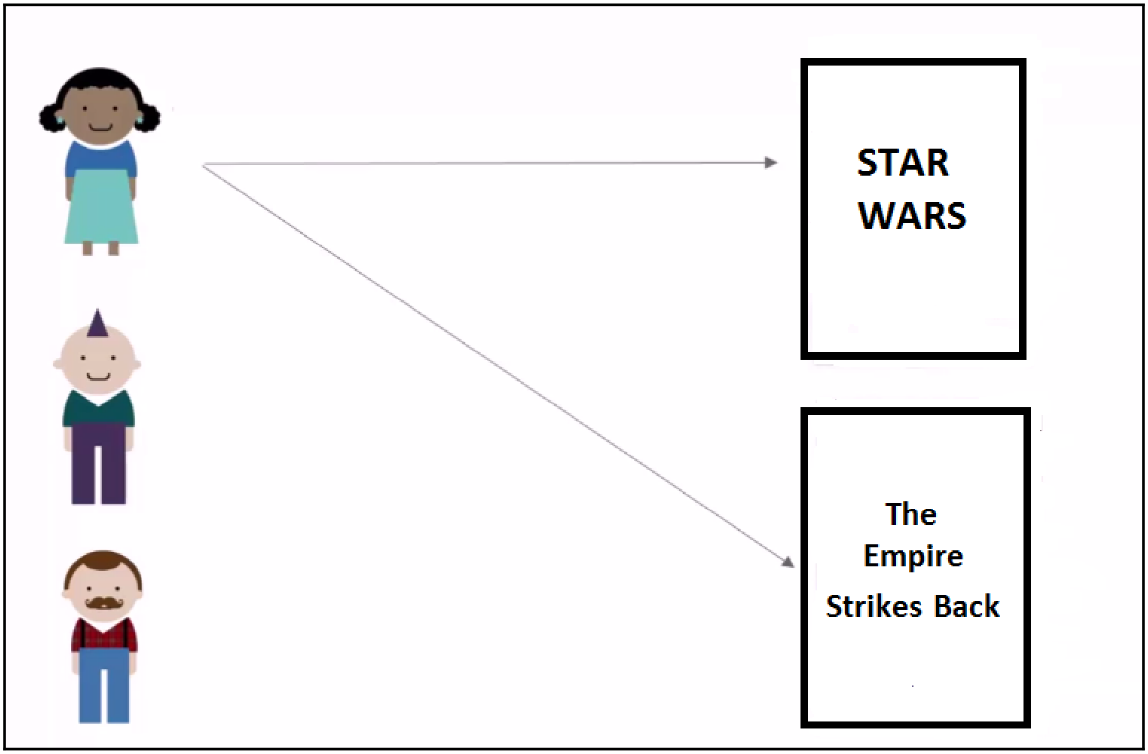

```
Out[6]:  array([4])
```

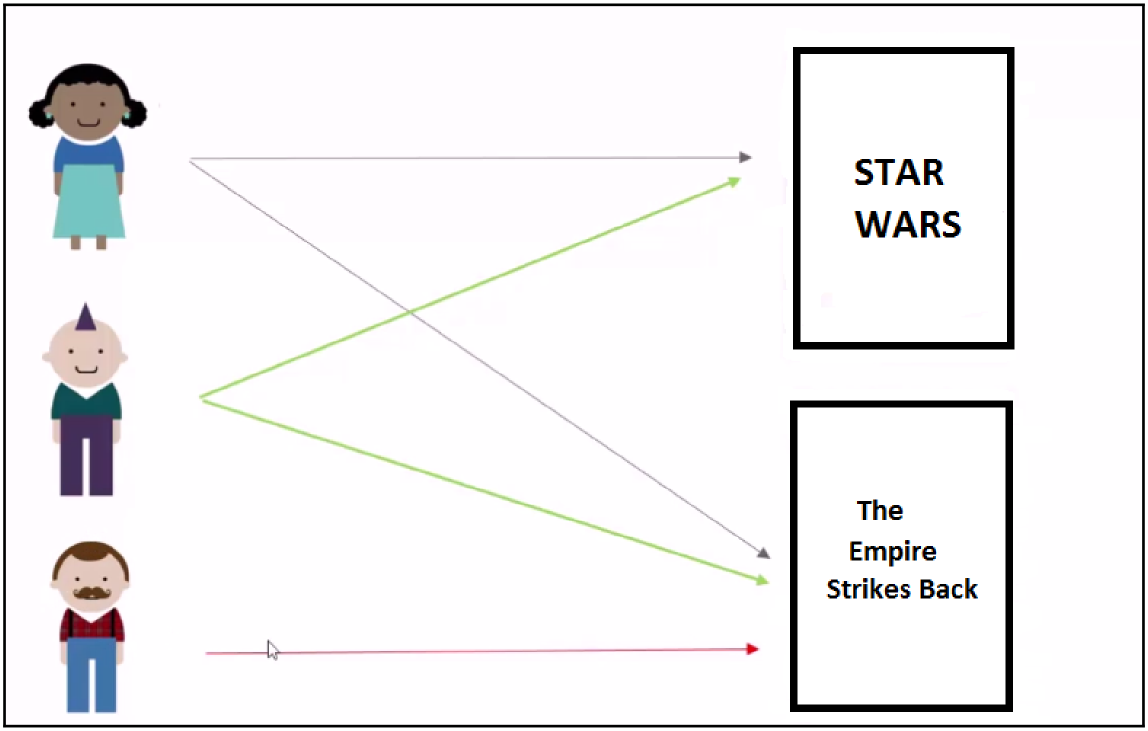
Chapter 6: Recommender Systems

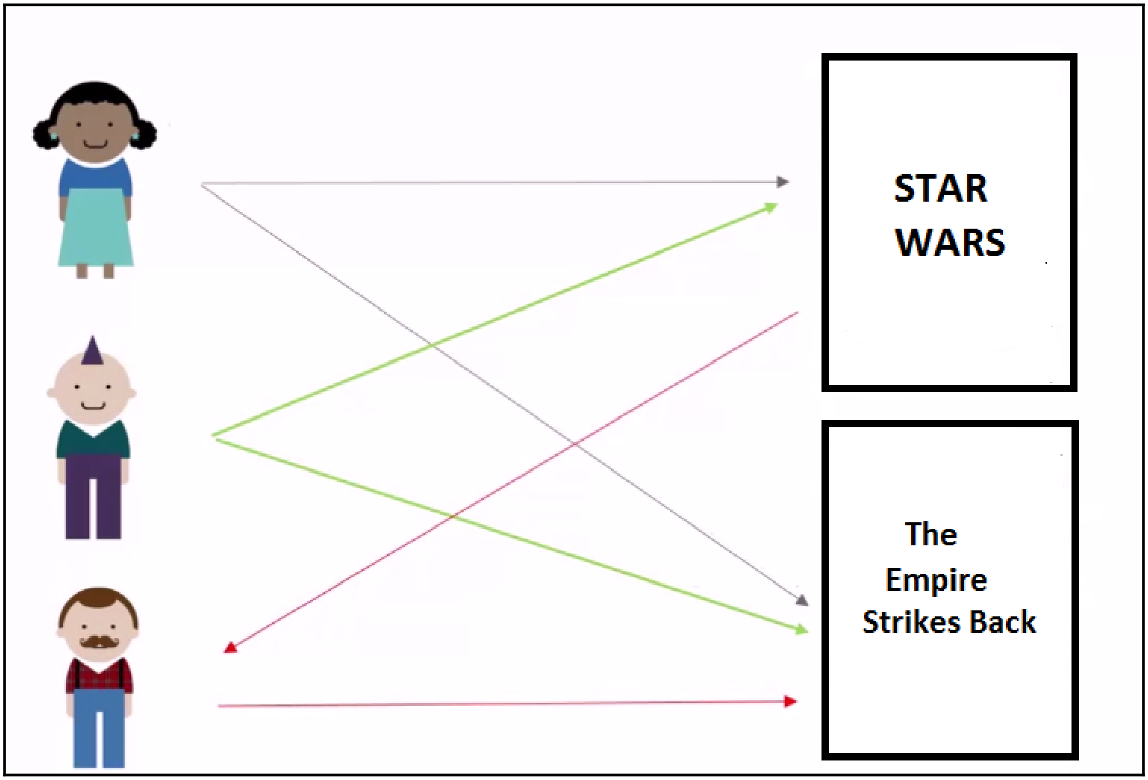

BEAUTIFUL THINGS ON AMAZON UPDATED DAILY
EXPLORE
amazon
Year-End Deals
Recommended for you, Frank
Buy It Again in Office
Buy It Again in Other Categories
Cooking Books
Photography
Literature & Fiction
Games
Comic Books
Toys for Grown-ups
Power & Hand Tools
Astronomy


Netflix DVD
Queue
New Releases
Exciting TV Shows from the 1990s
Children & Family
View all
Race Against Time Action & Adventure


STAR WARS
The Empire Strikes Back
STAR WARS


STAR WARS
The Empire Strikes Back


STAR WARS
The Empire Strikes Back



Finding Similar Movies

We'll start by loading up the MovieLens dataset. Using Pandas, we can very quickly load the rows of the u.data and u.item files that we care about, and merge them together so we can work with movie names instead of ID's. (In a real production job, you'd stick with ID's and worry about the names at the display layer to make things more efficient. But this lets us understand what's going on better for now.)

In [1]:
```
import pandas as pd

r_cols = ['user_id', 'movie_id', 'rating']
ratings = pd.read_csv('e:/sundog-consult/udemy/datascience/ml-100k/u.data', sep='\t', names=r_cols, usecols=range(3))

m_cols = ['movie_id', 'title']
movies = pd.read_csv('e:/sundog-consult/udemy/datascience/ml-100k/u.item', sep='|', names=m_cols, usecols=range(2))

ratings = pd.merge(movies, ratings)
```

In [2]:
```
ratings.head()
```

Out[2]:

	movie_id	title	user_id	rating
0	1	Toy Story (1995)	308	4
1	1	Toy Story (1995)	287	5
2	1	Toy Story (1995)	148	4
3	1	Toy Story (1995)	280	4
4	1	Toy Story (1995)	66	3

```
In [2]: ratings.head()
```

Out[2]:

	movie_id	title	user_id	rating
0	1	Toy Story (1995)	308	4
1	1	Toy Story (1995)	287	5
2	1	Toy Story (1995)	148	4
3	1	Toy Story (1995)	280	4
4	1	Toy Story (1995)	66	3

title	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: A Space Odyssey (1968)	3 Ninjas: High Noon At Mega Mountain (1998)	39 Steps, The (1935)	...	Yankee Zulu (1994)	Year of the Horse (1997)	You So Crazy (1994)	Young Frankenstein (1974)	Young Guns (1988)
user_id																
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	2	5	NaN	NaN	3	4	NaN	NaN	...	NaN	NaN	NaN	5	3
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1	NaN	...	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	2	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN

5 rows × 1664 columns

```
Out[4]: user_id
        0       5
        1       5
        2       5
        3     NaN
        4       5
        Name: Star Wars (1977), dtype: float64
```

Out[5]:

	0
title	
'Til There Was You (1997)	0.872872
1-900 (1994)	-0.645497
101 Dalmatians (1996)	0.211132
12 Angry Men (1957)	0.184289
187 (1997)	0.027398
2 Days in the Valley (1996)	0.066654
20,000 Leagues Under the Sea (1954)	0.289768
2001: A Space Odyssey (1968)	0.230884
39 Steps, The (1935)	0.106453
8 1/2 (1963)	-0.142977

Out[6]:

```
title
Full Speed (1996)                                                                    1.000000
Star Wars (1977)                                                                     1.000000
Mondo (1996)                                                                         1.000000
Man of the Year (1995)                                                               1.000000
Line King: Al Hirschfeld, The (1996)                                                 1.000000
Outlaw, The (1943)                                                                   1.000000
Hurricane Streets (1998)                                                             1.000000
Hollow Reed (1996)                                                                   1.000000
Scarlet Letter, The (1926)                                                           1.000000
Safe Passage (1994)                                                                  1.000000
Good Man in Africa, A (1994)                                                         1.000000
Golden Earrings (1947)                                                               1.000000
Old Lady Who Walked in the Sea, The (Vieille qui marchait dans la mer, La) (1991)    1.000000
No Escape (1994)                                                                     1.000000
Ed's Next Move (1996)                                                                1.000000
Stripes (1981)                                                                       1.000000
Cosi (1996)                                                                          1.000000
Commandments (1997)                                                                  1.000000
Twisted (1996)                                                                       1.000000
Beans of Egypt, Maine, The (1994)                                                    1.000000
Last Time I Saw Paris, The (1954)                                                    1.000000
Maya Lin: A Strong Clear Vision (1994)                                               1.000000
Designated Mourner, The (1997)                                                       0.970725
Albino Alligator (1996)                                                              0.968496
Angel Baby (1995)                                                                    0.962250
Prisoner of the Mountains (Kavkazsky Plennik) (1996)                                 0.927173
Love in the Afternoon (1957)                                                         0.923381
'Til There Was You (1997)                                                            0.872872
A Chef in Love (1996)                                                                0.868599
```

Out[8]:

	rating	
	size	**mean**
title		
'Til There Was You (1997)	9	2.333333
1-900 (1994)	5	2.600000
101 Dalmatians (1996)	109	2.908257
12 Angry Men (1957)	125	4.344000
187 (1997)	41	3.024390

Out[9]:

	rating	
	size	**mean**
title		
Close Shave, A (1995)	112	4.491071
Schindler's List (1993)	298	4.466443
Wrong Trousers, The (1993)	118	4.466102
Casablanca (1942)	243	4.456790
Shawshank Redemption, The (1994)	283	4.445230
Rear Window (1954)	209	4.387560
Usual Suspects, The (1995)	267	4.385768
Star Wars (1977)	584	4.359589
12 Angry Men (1957)	125	4.344000
Citizen Kane (1941)	198	4.292929
To Kill a Mockingbird (1962)	219	4.292237
One Flew Over the Cuckoo's Nest (1975)	264	4.291667
Silence of the Lambs, The (1991)	390	4.289744
North by Northwest (1959)	179	4.284916
Godfather, The (1972)	413	4.283293

Out[11]:

	(rating, size)	(rating, mean)	similarity
title			
101 Dalmatians (1996)	109	2.908257	0.211132
12 Angry Men (1957)	125	4.344000	0.184289
2001: A Space Odyssey (1968)	259	3.969112	0.230884
Absolute Power (1997)	127	3.370079	0.085440
Abyss, The (1989)	151	3.589404	0.203709

Out[12]:

	(rating, size)	(rating, mean)	similarity
title			
Star Wars (1977)	584	4.359589	1.000000
Empire Strikes Back, The (1980)	368	4.206522	0.748353
Return of the Jedi (1983)	507	4.007890	0.672556
Raiders of the Lost Ark (1981)	420	4.252381	0.536117
Austin Powers: International Man of Mystery (1997)	130	3.246154	0.377433
Sting, The (1973)	241	4.058091	0.367538
Indiana Jones and the Last Crusade (1989)	331	3.930514	0.350107
Pinocchio (1940)	101	3.673267	0.347868
Frighteners, The (1996)	115	3.234783	0.332729
L.A. Confidential (1997)	297	4.161616	0.319065
Wag the Dog (1997)	137	3.510949	0.318645
Dumbo (1941)	123	3.495935	0.317656
Bridge on the River Kwai, The (1957)	165	4.175758	0.316580
Philadelphia Story, The (1940)	104	4.115385	0.314272
Miracle on 34th Street (1994)	101	3.722772	0.310921

Out[1]:

	movie_id	title	user_id	rating
0	1	Toy Story (1995)	308	4
1	1	Toy Story (1995)	287	5
2	1	Toy Story (1995)	148	4
3	1	Toy Story (1995)	280	4
4	1	Toy Story (1995)	66	3

Out[2]:

title	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: A Space Odyssey (1968)	3 Ninjas: High Noon At Mega Mountain (1998)	39 Steps, The (1935)	...	Yankee Zulu (1994)	Year of the Horse (1997)	You So Crazy (1994)	Young Frankenstein (1974)	Young Guns (1988)
user_id																
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	2	5	NaN	NaN	3	4	NaN	NaN	...	NaN	NaN	NaN	5	3
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1	NaN	...	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	2	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN

5 rows × 1664 columns

Out[3]:

title	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: A Space Odyssey (1968)	3 Ninjas: High Noon At Mega Mountain (1998)	39 Steps, The (1935)	...	Yankee Zulu (1994)	Year of the Horse (1997)	You So Crazy (1994)
title														
'Til There Was You (1997)	1.0	NaN	-1.000000	-0.500000	-0.500000	0.522233	NaN	-0.426401	NaN	NaN	...	NaN	NaN	NaN
1-900 (1994)	NaN	1	NaN	NaN	NaN	NaN	NaN	-0.981981	NaN	NaN	...	NaN	NaN	NaN
101 Dalmatians (1996)	-1.0	NaN	1.000000	-0.049890	0.269191	0.048973	0.266928	-0.043407	NaN	0.111111	...	NaN	-1.000000	NaN
12 Angry Men (1957)	-0.5	NaN	-0.049890	1.000000	0.666667	0.256625	0.274772	0.178848	NaN	0.457176	...	NaN	NaN	NaN
187 (1997)	-0.5	NaN	0.269191	0.666667	1.000000	0.596644	NaN	-0.554700	NaN	1.000000	...	NaN	0.866025	NaN

5 rows × 1664 columns

Out[4]:

title	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: A Space Odyssey (1968)	3 Ninjas: High Noon At Mega Mountain (1998)	39 Steps, The (1935)	...	Yankee Zulu (1994)	Year of the Horse (1997)	You So Crazy (1994)	Young Frankenstein (1974)
title															
'Til There Was You (1997)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
1-900 (1994)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
101 Dalmatians (1996)	NaN	NaN	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
12 Angry Men (1957)	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
187 (1997)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN

5 rows × 1664 columns

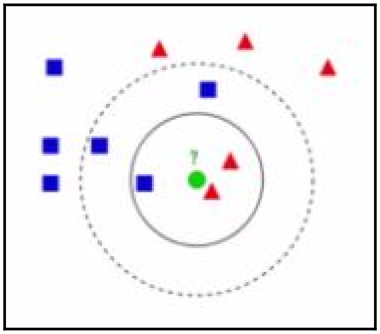
```
Out[5]: title
        Empire Strikes Back, The (1980)     5
        Gone with the Wind (1939)           1
        Star Wars (1977)                    5
        Name: 0, dtype: float64
```

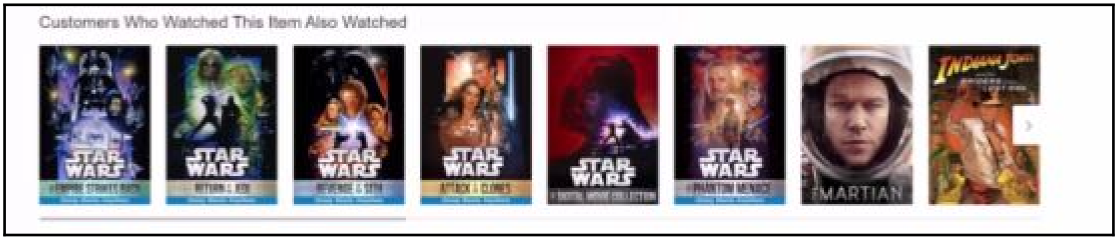
```
Adding sims for Empire Strikes Back, The (1980)...
Adding sims for Gone with the Wind (1939)...
Adding sims for Star Wars (1977)...
sorting...
title
Empire Strikes Back, The (1980)                        5.000000
Star Wars (1977)                                       5.000000
Empire Strikes Back, The (1980)                        3.741763
Star Wars (1977)                                       3.741763
Return of the Jedi (1983)                              3.606146
Return of the Jedi (1983)                              3.362779
Raiders of the Lost Ark (1981)                         2.693297
Raiders of the Lost Ark (1981)                         2.680586
Austin Powers: International Man of Mystery (1997)     1.887164
Sting, The (1973)                                      1.837692
dtype: float64
```

```
Out[8]: title
        Empire Strikes Back, The (1980)                8.877450
        Star Wars (1977)                               8.870971
        Return of the Jedi (1983)                      7.178172
        Raiders of the Lost Ark (1981)                 5.519700
        Indiana Jones and the Last Crusade (1989)      3.488028
        Bridge on the River Kwai, The (1957)           3.366616
        Back to the Future (1985)                      3.357941
        Sting, The (1973)                              3.329843
        Cinderella (1950)                              3.245412
        Field of Dreams (1989)                         3.222311
        dtype: float64
```

```
Out[9]: title
        Return of the Jedi (1983)                      7.178172
        Raiders of the Lost Ark (1981)                 5.519700
        Indiana Jones and the Last Crusade (1989)      3.488028
        Bridge on the River Kwai, The (1957)           3.366616
        Back to the Future (1985)                      3.357941
        Sting, The (1973)                              3.329843
        Cinderella (1950)                              3.245412
        Field of Dreams (1989)                         3.222311
        Wizard of Oz, The (1939)                       3.200268
        Dumbo (1941)                                   2.981645
        dtype: float64
```

Chapter 7 : More Data Mining and Machine Learning Techniques





Out[1]:

	user_id	movie_id	rating
0	0	50	5
1	0	172	5
2	0	133	1
3	196	242	3
4	186	302	3

Out[4]:

	size
movie_id	
1	0.773585
2	0.222985
3	0.152659
4	0.356775
5	0.145798

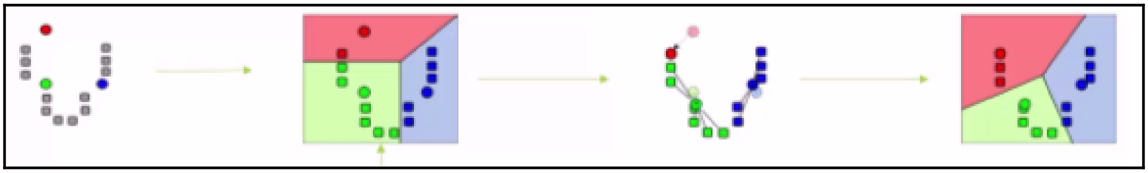
```
('Toy Story (1995)',
 [0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 0.77358490566037741,
 3.8783185840707963)
```

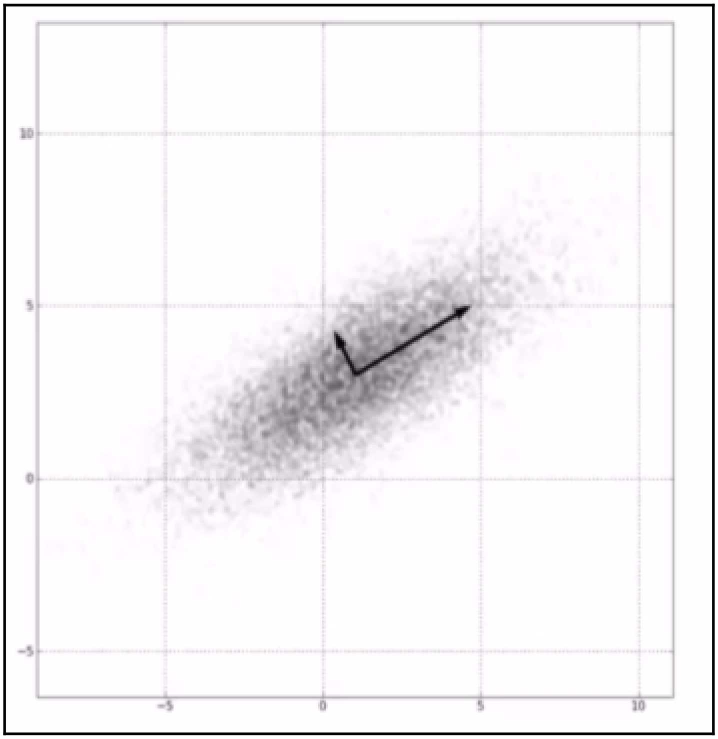
```
0.8004574042309891
```

```
('GoldenEye (1995)', [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0], 0.22298456260720412, 3.2061068702290076)
('Get Shorty (1995)', [0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], 0.35677530017152659, 3.5502392344497609)
```

```
Liar Liar (1997) 3.15670103093
Aladdin (1992) 3.81278538813
Willy Wonka and the Chocolate Factory (1971) 3.63190184049
Monty Python and the Holy Grail (1974) 4.0664556962
Full Monty, The (1997) 3.92698412698
George of the Jungle (1997) 2.68518518519
Beavis and Butt-head Do America (1996) 2.78846153846
Birdcage, The (1996) 3.44368600683
Home Alone (1990) 3.08759124088
Aladdin and the King of Thieves (1996) 2.84615384615
```

```
3.3445905900235564
```

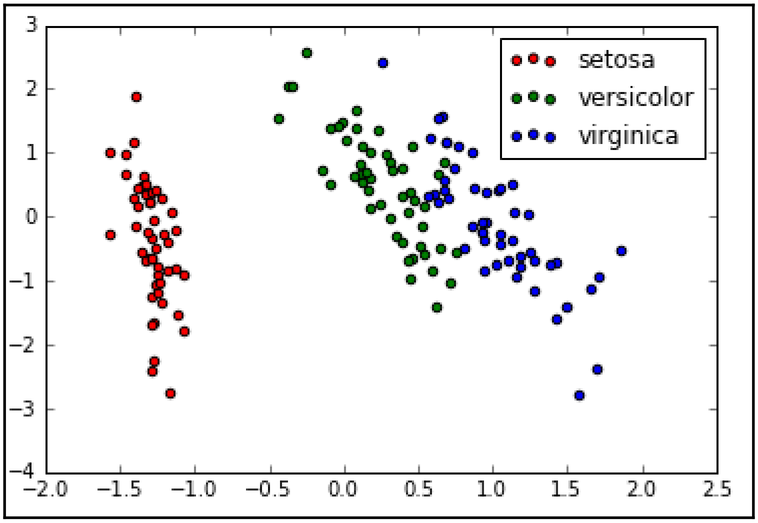


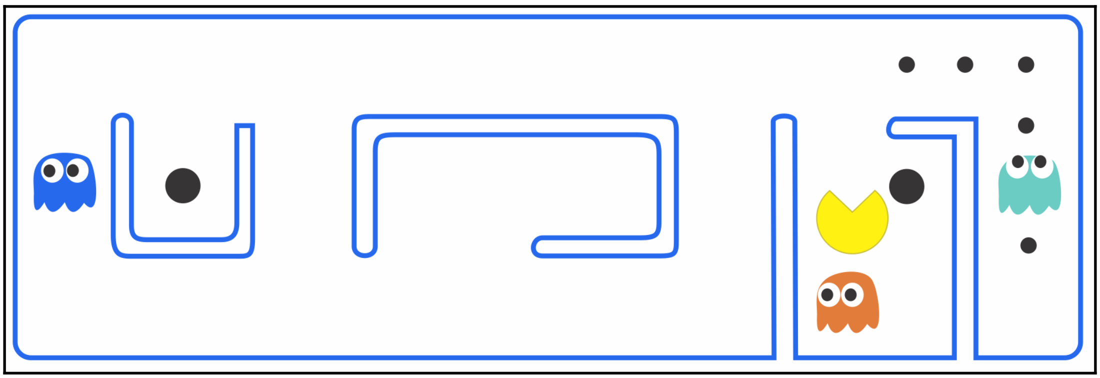


```
150
4
['setosa', 'versicolor', 'virginica']
```

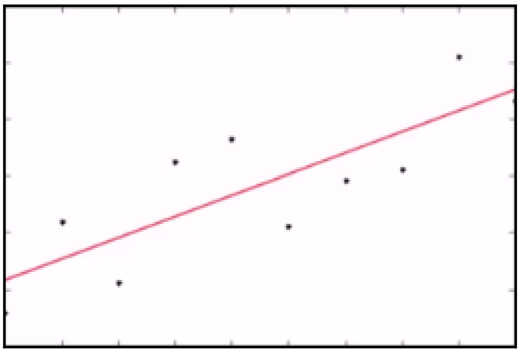
```
[[ 0.36158968 -0.08226889  0.85657211  0.35884393]
 [-0.65653988 -0.72971237  0.1757674   0.07470647]]
```

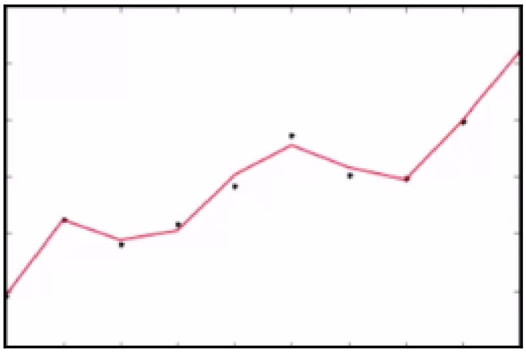
```
[ 0.92461621  0.05301557]
0.977631775025
```





Chapter 8: Dealing with Real-World Data





$$Error = Bias^2 + Variance$$

```
Out[2]:  0.96666666666666667
```

```
[ 1.          1.          0.9         0.93333333  1.        ]
0.966666666667
```

```
[ 1.          1.          0.9         0.93333333  1.        ]
0.966666666667
```

```
IOErrorTraceback (most recent call last)
<ipython-input-3-281d53278f3c> in <module>()
      1 URLCounts = {}
      2
----> 3 with open(logPath, "r") as f:
      4     for line in (l.rstrip() for l in f):
      5         match= format_pat.match(line)

IOError: [Errno 2] No such file or directory: 'E:\\sundog-consult\\Udemy\\DataScience\\access_log.txt'
```

```
['_\\xb0ZP\\x07tR\\xe5']
[]
[]
[]
[]
[]
[]
[]
[]
[]
[]
```

```
/xmlrpc.php: 68494
/wp-login.php: 1923
/: 440
/blog/: 138
/robots.txt: 123
/sitemap_index.xml: 118
/post-sitemap.xml: 118
/category-sitemap.xml: 117
/page-sitemap.xml: 117
/orlando-headlines/: 95
/san-jose-headlines/: 85
http://51.254.206.142/httptest.php: 81
/comics-2/: 76
/travel/: 74
/entertainment/: 72
/world/: 70
/business/: 70
/weather/: 70
/national/: 70
/national-headlines/: 70
```

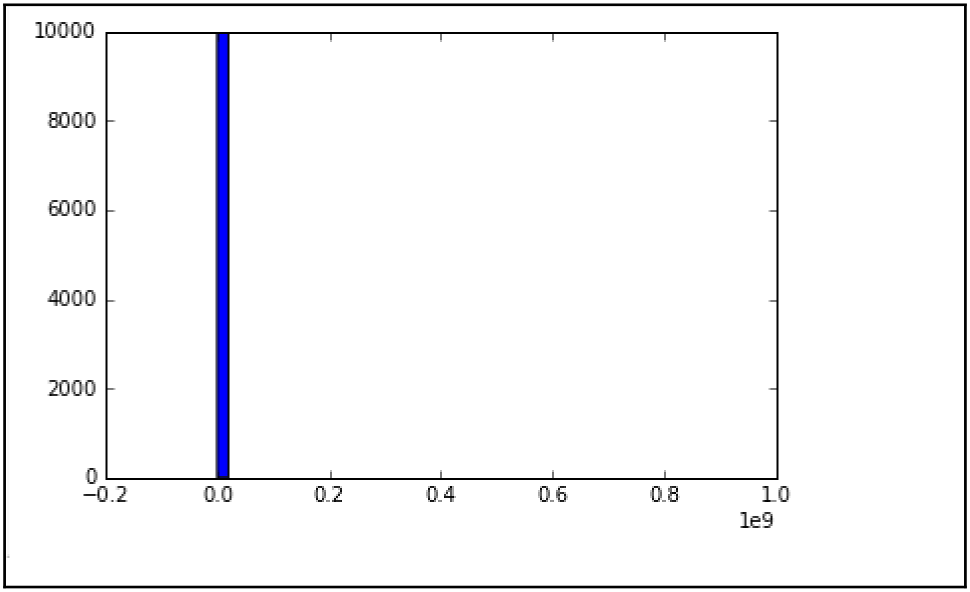
```
/: 434
/blog/: 138
/robots.txt: 123
/sitemap_index.xml: 118
/post-sitemap.xml: 118
/category-sitemap.xml: 117
/page-sitemap.xml: 117
/orlando-headlines/: 95
/san-jose-headlines/: 85
http://51.254.206.142/httptest.php: 81
/comics-2/: 76
/travel/: 74
/entertainment/: 72
/world/: 70
/business/: 70
/weather/: 70
/national/: 70
/national-headlines/: 70
/defense-sticking-head-sand/: 69
/about/: 69
```

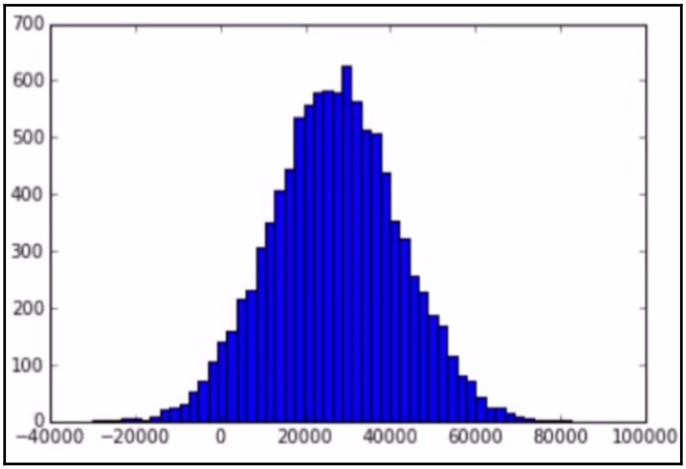
```
54.165.199.171 - - [05/Dec/2015:09:32:05 +0000] "GET /blog/ HTTP/1.0" 200 31670 "-" "-"
```

```
Mozilla/4.0 (compatible: MSIE 7.0; Windows NT 6.0): 68484
-: 4035
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0): 1724
W3 Total Cache/0.9.4.1: 468
Mozilla/5.0 (compatible; Baiduspider/2.0; +http://www.baidu.com/search/spider.html): 278
Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html): 248
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.249
0.86 Safari/537.36: 158
Mozilla/5.0 (Windows NT 6.1; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0: 144
Mozilla/5.0 (iPad; CPU OS 8_4 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) Versio
n/8.0 Mobile/12H143 Safari/600.1.4: 120
Mozilla/5.0 (Linux; Android 5.1.1; SM-G900T Build/LMY47X) AppleWebKit/537.36 (KHTML, like G
ecko) Chrome/46.0.2490.76 Mobile Safari/537.36: 47
Mozilla/5.0 (compatible; bingbot/2.0; +http://www.bing.com/bingbot.htm): 43
Mozilla/5.0 (compatible; MJ12bot/v1.4.5; http://www.majestic12.co.uk/bot.php?+): 41
Opera/9.80 (Windows NT 6.0) Presto/2.12.388 Version/12.14: 40
Mozilla/5.0 (compatible; YandexBot/3.0; +http://yandex.com/bots): 27
Ruby: 15
Mozilla/5.0 (Linux; Android 5.1.1; SM-G900T Build/LMY47X) AppleWebKit/537.36 (KHTML, like G
```

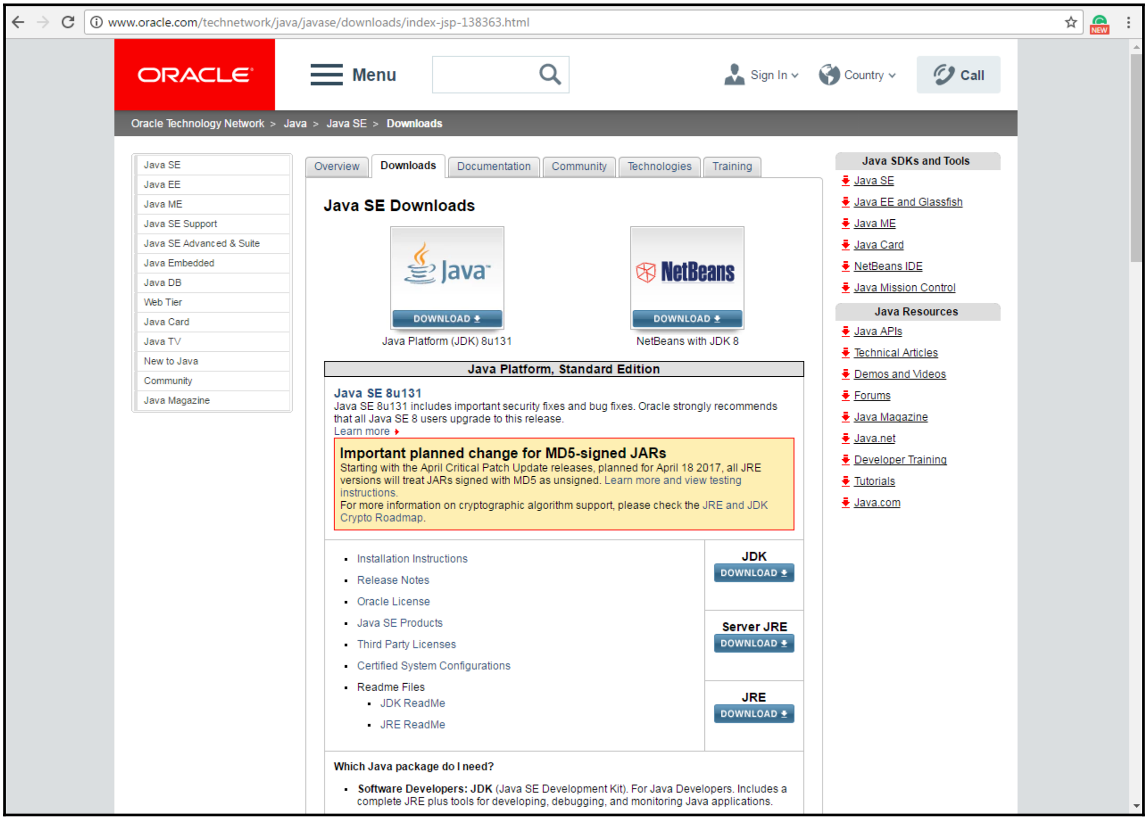
```
/: 77
/orlando-headlines/: 36
/?page_id=34248: 28
/wp-content/cache/minify/000000/M9bPKixNLarUy00szs8D0Zl5AA.js: 27
/wp-content/cache/minify/000000/lY7dDoIwDIVfiG0KxkfxfnbdKO4HuxICTy-it8Zw15PzfSftzPCckJem-x4qUWArqBPl5mygZLEgyhdOaoxTo
GyGaiALiOfUnIz0qDLOdSZGE-nOlpc3kopDzrSyavVVt_veb5qSDVhjsQ6dHh_B_eE_z2pYIGJ7iBWKeEio_eT9UQe4xHhDll27mGRryVu_pRc.js: 27
/wp-content/cache/minify/000000/M9AvyUjVzUstLy7PLErVz8lMKkosqtTPKtYvTi7KLCgpBgA.js: 27
/wp-content/cache/minify/000000/fY45DoAwDAQ_FMvkRQgFA5ZyWLajiN9zNHR0O83MRkyt-pIctqYFJPedKyYzfHg2PzOFiENAzaD07AxcpKmTo
lORvDjZt8KEfhBUGjZYCf8Fb0fvA1TXCw.css: 25
/?author=1: 21
/wp-content/cache/minify/000000/hcrRCYAwDAXAhXyEjiQ1YKAh4SVSx3cE7_uG7ASr4M9qg3kGWyk1adklK84LHtRj_My6Y0Pfqcz-AA.js: 20
/wp-content/uploads/2014/11/nhn1.png: 19
/wp-includes/js/wp-emoji-release.min.js?ver=4.3.1: 17
/wp-content/cache/minify/000000/BcGBCQAgCATAiUSaKYSERPk3avzuht4SkBJnt4tHJdqgnPBqKldesTcN1R8.js: 17
/wp-login.php: 16
/comics-2/: 12
/world/: 12
/favicon.ico: 10
/wp-content/uploads/2014/11/babyblues.jpg: 6
/wp-content/uploads/2014/11/garfield.jpg: 6
/wp-content/uploads/2014/11/violentcrime.jpg: 6
/robots.txt: 6
```

```
/: 77
/orlando-headlines/: 36
/world/: 12
/comics-2/: 12
/weather/: 4
/about/: 4
/australia/: 4
/national-headlines/: 3
/sample-page/feed/: 2
/feed/: 2
/technology/: 2
/science/: 2
/entertainment/: 1
/san-jose-headlines/: 1
/business/: 1
/travel/feed/: 1
```





Chapter 9: Apache Spark - Machine Learning on Big Data



www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html
ORACLE
Menu
Sign In
Country
Call
Oracle Technology Network > Java > Java SE > Downloads
Java SE
Java EE
Java ME
Java SE Support
Java SE Advanced & Suite
Java Embedded
Java DB
Web Tier
Java Card
Java TV
New to Java
Community
Java Magazine
Overview
Downloads
Documentation
Community
Technologies
Training
Java SE Downloads
Java
DOWNLOAD
Java Platform (JDK) 8u131
NetBeans
DOWNLOAD
NetBeans with JDK 8
Java Platform, Standard Edition
Java SE 8u131
Java SE 8u131 includes important security fixes and bug fixes. Oracle strongly recommends that all Java SE 8 users upgrade to this release.
Learn more
Important planned change for MD5-signed JARs
Starting with the April Critical Patch Update releases, planned for April 18 2017, all JRE versions will treat JARs signed with MD5 as unsigned. Learn more and view testing instructions.
For more information on cryptographic algorithm support, please check the JRE and JDK Crypto Roadmap.
Installation Instructions
Release Notes
Oracle License
Java SE Products
Third Party Licenses
Certified System Configurations
Readme Files
JDK ReadMe
JRE ReadMe
JDK
DOWNLOAD
Server JRE
DOWNLOAD
JRE
DOWNLOAD
Which Java package do I need?
Software Developers: JDK (Java SE Development Kit). For Java Developers. Includes a complete JRE plus tools for developing, debugging, and monitoring Java applications.
Java SDKs and Tools
Java SE
Java EE and Glassfish
Java ME
Java Card
NetBeans IDE
Java Mission Control
Java Resources
Java APIs
Technical Articles
Demos and Videos
Forums
Java Magazine
Java.net
Developer Training
Tutorials
Java.com

Java Platform, Standard Edition

Java SE 8u131

Java SE 8u131 includes important security fixes and bug fixes. Oracle strongly recommends that all Java SE 8 users upgrade to this release.
Learn more

Important planned change for MD5-signed JARs

Starting with the April Critical Patch Update releases, planned for April 18 2017, all JRE versions will treat JARs signed with MD5 as unsigned. Learn more and view testing instructions.
For more information on cryptographic algorithm support, please check the JRE and JDK Crypto Roadmap.

- Installation Instructions
- Release Notes
- Oracle License
- Java SE Products
- Third Party Licenses
- Certified System Configurations
- Readme Files
  - JDK ReadMe
  - JRE ReadMe

JDK

DOWNLOAD

Server JRE

DOWNLOAD

JRE

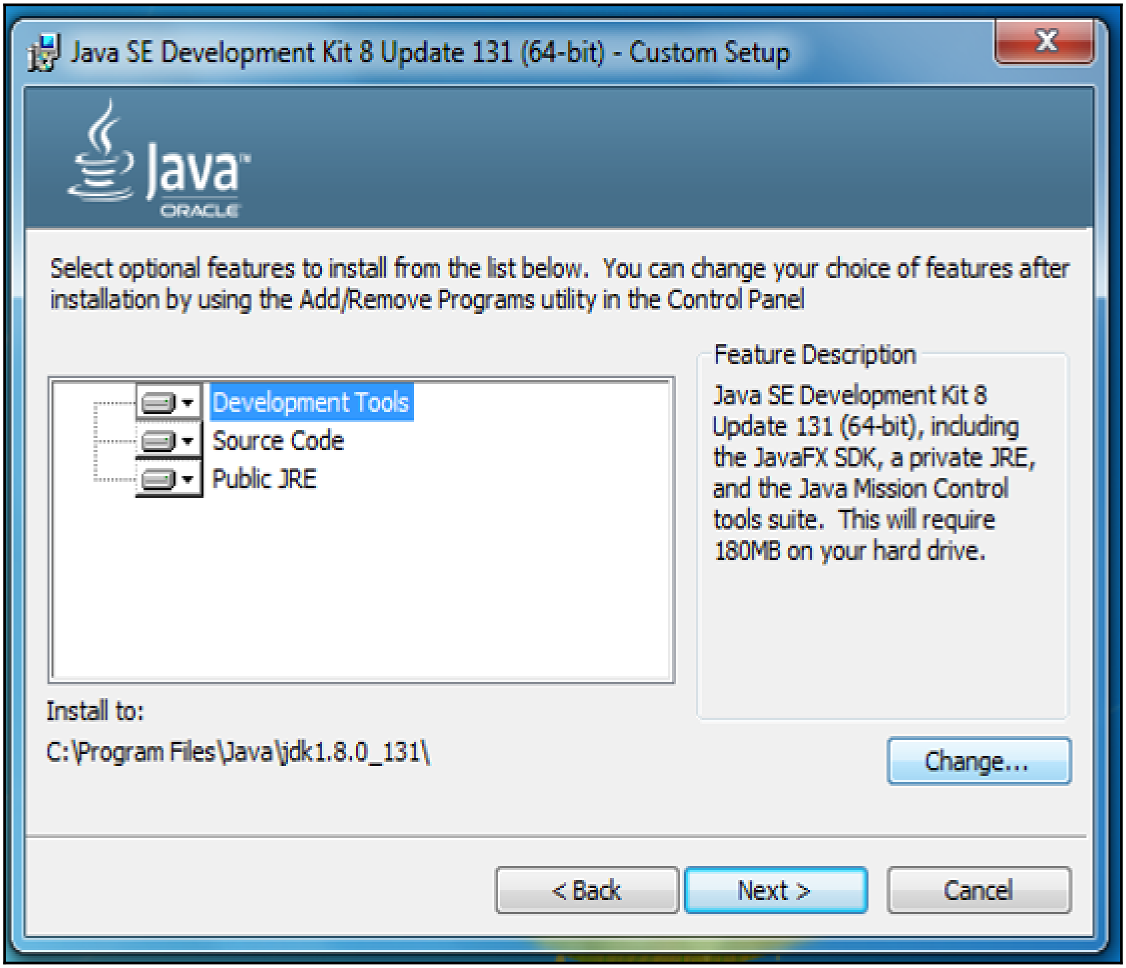
DOWNLOAD

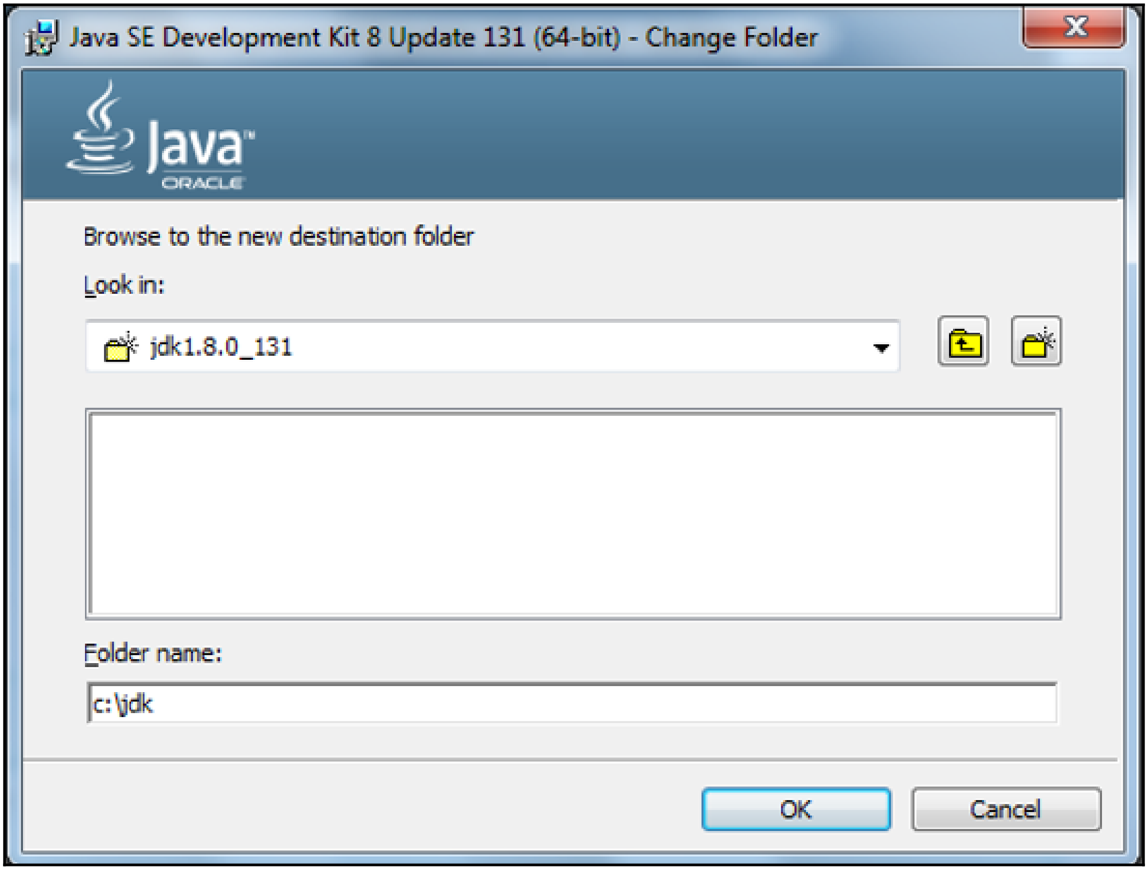
Java SE Development Kit 8u131

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

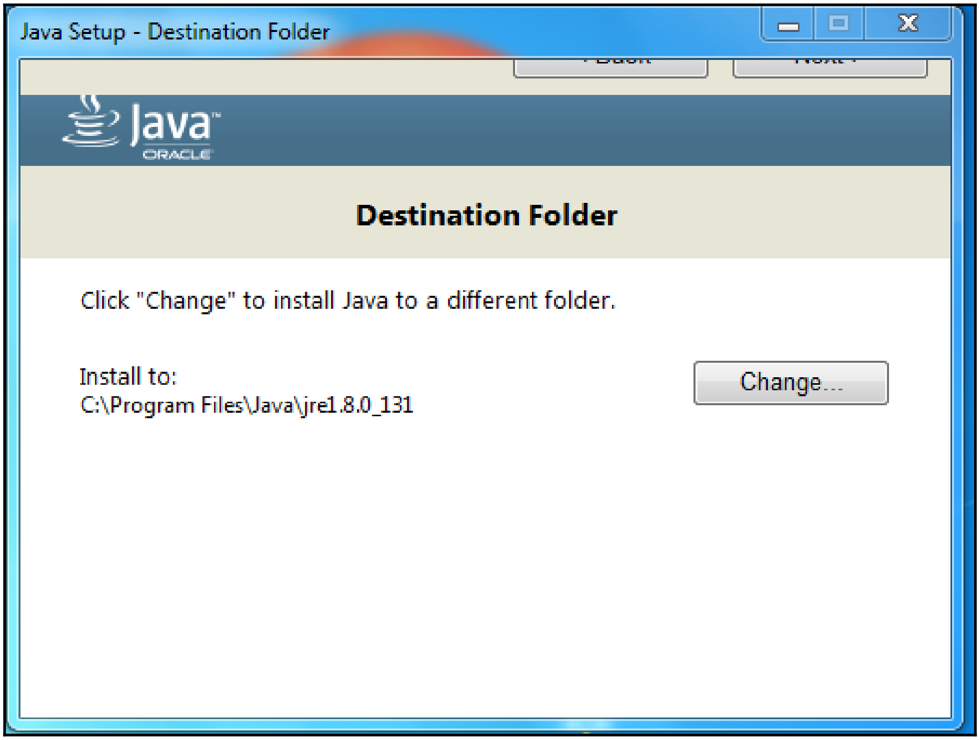
Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.

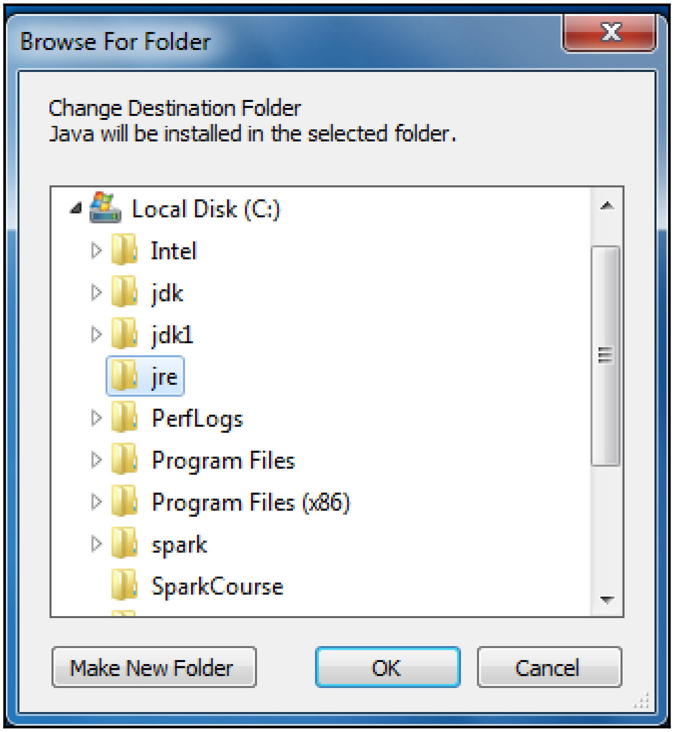
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.87 MB	jdk-8u131-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.81 MB	jdk-8u131-linux-arm64-vfp-hflt.tar.gz
Linux x86	164.66 MB	jdk-8u131-linux-i586.rpm
Linux x86	179.39 MB	jdk-8u131-linux-i586.tar.gz
Linux x64	162.11 MB	jdk-8u131-linux-x64.rpm
Linux x64	176.95 MB	jdk-8u131-linux-x64.tar.gz
Mac OS X	226.57 MB	jdk-8u131-macosx-x64.dmg
Solaris SPARC 64-bit	139.79 MB	jdk-8u131-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.13 MB	jdk-8u131-solaris-sparcv9.tar.gz
Solaris x64	140.51 MB	jdk-8u131-solaris-x64.tar.Z
Solaris x64	96.96 MB	jdk-8u131-solaris-x64.tar.gz
Windows x86	191.22 MB	jdk-8u131-windows-i586.exe
Windows x64	198.03 MB	jdk-8u131-windows-x64.exe

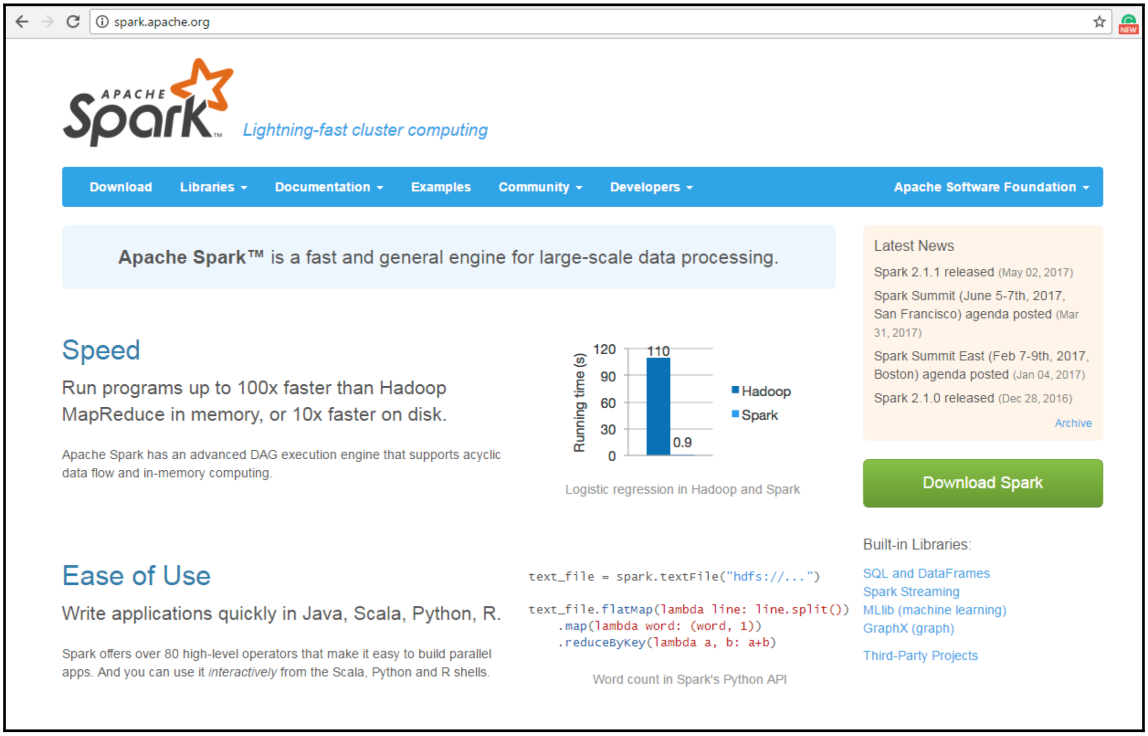

Java SE Development Kit 8 Update 131 (64-bit) - Custom Setup
Java
ORACLE
Select optional features to install from the list below. You can change your choice of features after installation by using the Add/Remove Programs utility in the Control Panel
Development Tools
Source Code
Public JRE
Feature Description
Java SE Development Kit 8 Update 131 (64-bit), including the JavaFX SDK, a private JRE, and the Java Mission Control tools suite. This will require 180MB on your hard drive.
Install to:
C:\Program Files\Java\jdk1.8.0_131\
Change...
< Back
Next >
Cancel

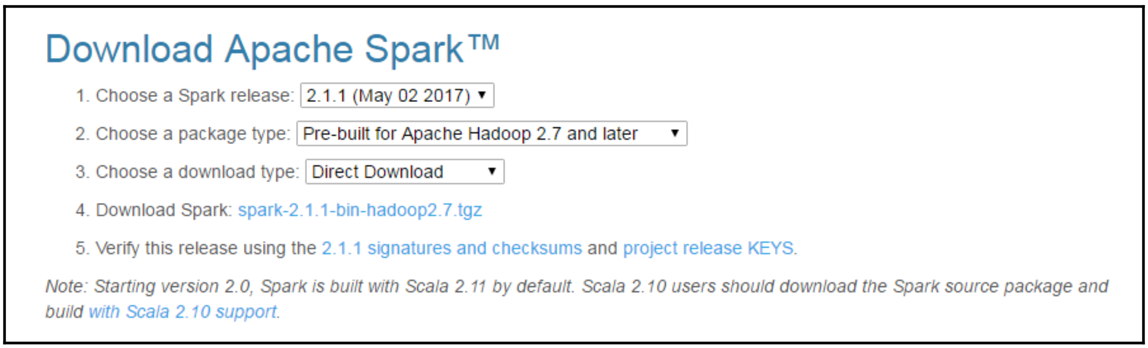


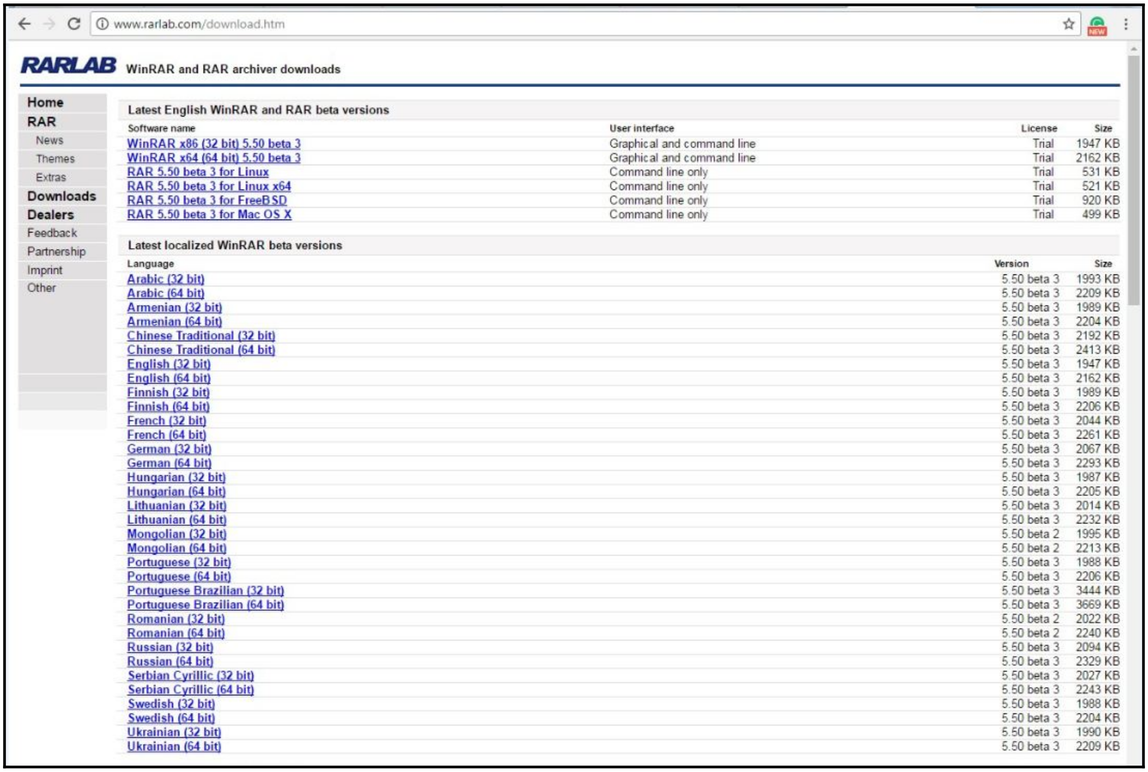
Java SE Development Kit 8 Update 131 (64-bit) - Change Folder
Java
ORACLE
Browse to the new destination folder
Look in:
jdk1.8.0_131
Folder name:
c:\jdk
OK
Cancel


Java Setup - Destination Folder
Java
ORACLE
Destination Folder
Click "Change" to install Java to a different folder.
Install to:
C:\Program Files\Java\jre1.8.0_131
Change...


Browse For Folder
Change Destination Folder
Java will be installed in the selected folder.
Local Disk (C:)
Intel
jdk
jdk1
jre
PerfLogs
Program Files
Program Files (x86)
spark
SparkCourse
Make New Folder
OK
Cancel


spark.apache.org
APACHE Spark
Lightning-fast cluster computing
Download
Libraries
Documentation
Examples
Community
Developers
Apache Software Foundation
Apache Spark™ is a fast and general engine for large-scale data processing.
Latest News
Spark 2.1.1 released (May 02, 2017)
Spark Summit (June 5-7th, 2017, San Francisco) agenda posted (Mar 31, 2017)
Spark Summit East (Feb 7-9th, 2017, Boston) agenda posted (Jan 04, 2017)
Spark 2.1.0 released (Dec 28, 2016)
Archive
Speed
Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.
Apache Spark has an advanced DAG execution engine that supports acyclic data flow and in-memory computing.
Running time (s)
120
90
60
30
0
110
0.9
Hadoop
Spark
Logistic regression in Hadoop and Spark
Download Spark
Built-in Libraries:
SQL and DataFrames
Spark Streaming
MLlib (machine learning)
GraphX (graph)
Third-Party Projects
Ease of Use
Write applications quickly in Java, Scala, Python, R.
Spark offers over 80 high-level operators that make it easy to build parallel apps. And you can use it *interactively* from the Scala, Python and R shells.
text_file = spark.textFile("hdfs://...")
text_file.flatMap(lambda line: line.split())
.map(lambda word: (word, 1))
.reduceByKey(lambda a, b: a+b)
Word count in Spark's Python API


Download Apache Spark™
1. Choose a Spark release: 2.1.1 (May 02 2017)
2. Choose a package type: Pre-built for Apache Hadoop 2.7 and later
3. Choose a download type: Direct Download
4. Download Spark: spark-2.1.1-bin-hadoop2.7.tgz
5. Verify this release using the 2.1.1 signatures and checksums and project release KEYS.
Note: Starting version 2.0, Spark is built with Scala 2.11 by default. Scala 2.10 users should download the Spark source package and build with Scala 2.10 support.


www.rarlab.com/download.htm

RARLAB WinRAR and RAR archiver downloads

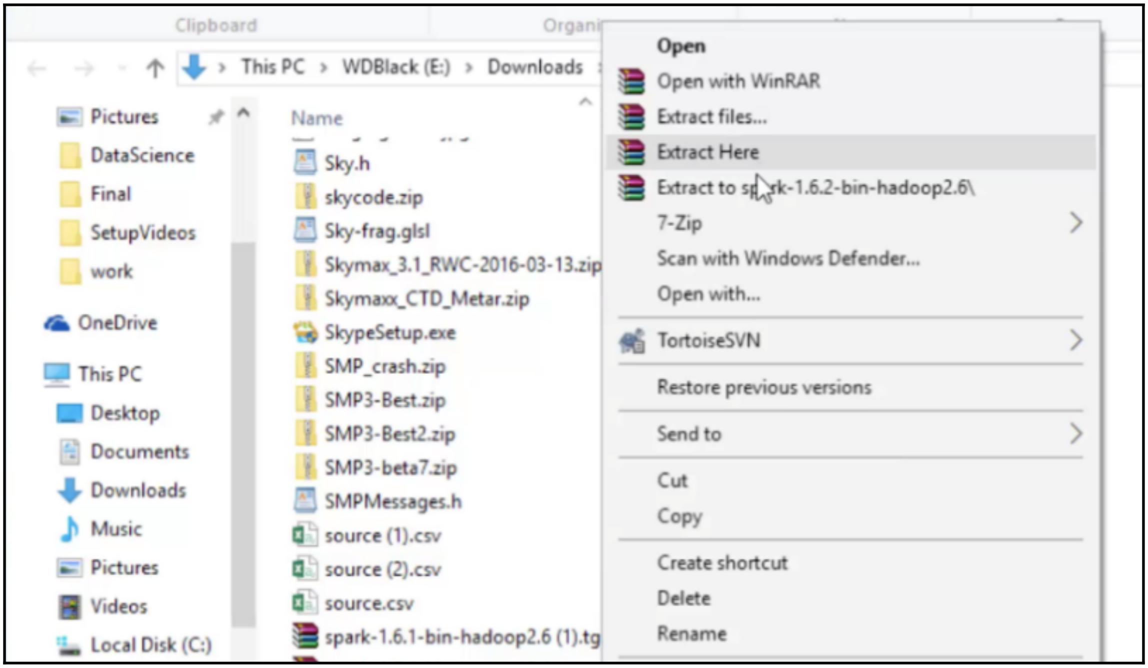
Home
RAR
News
Themes
Extras
Downloads
Dealers
Feedback
Partnership
Imprint
Other

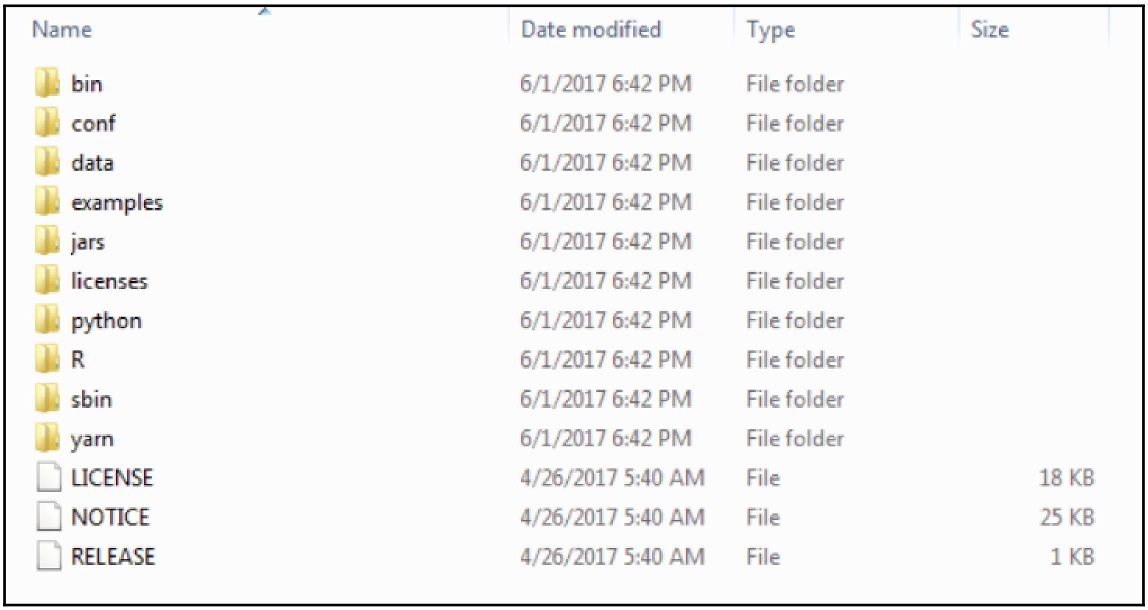
Latest English WinRAR and RAR beta versions

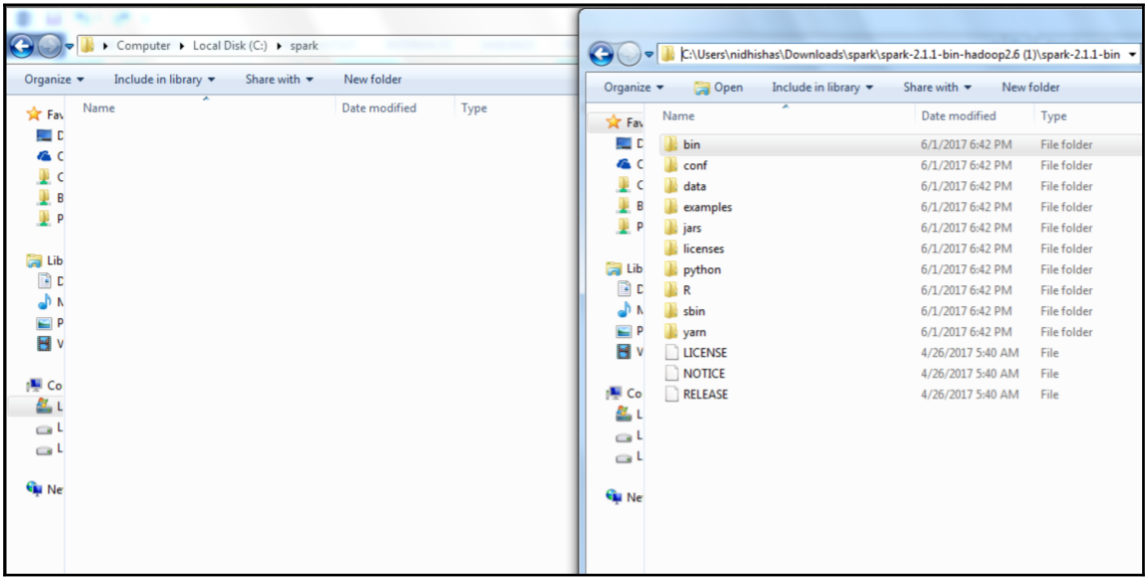
Software name	User interface	License	Size
WinRAR x86 (32 bit) 5.50 beta 3	Graphical and command line	Trial	1947 KB
WinRAR x64 (64 bit) 5.50 beta 3	Graphical and command line	Trial	2162 KB
RAR 5.50 beta 3 for Linux	Command line only	Trial	531 KB
RAR 5.50 beta 3 for Linux x64	Command line only	Trial	521 KB
RAR 5.50 beta 3 for FreeBSD	Command line only	Trial	920 KB
RAR 5.50 beta 3 for Mac OS X	Command line only	Trial	499 KB

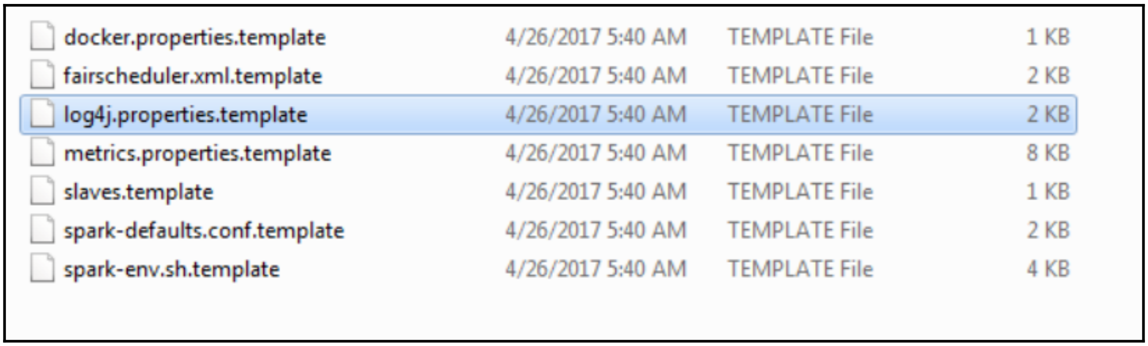
Latest localized WinRAR beta versions

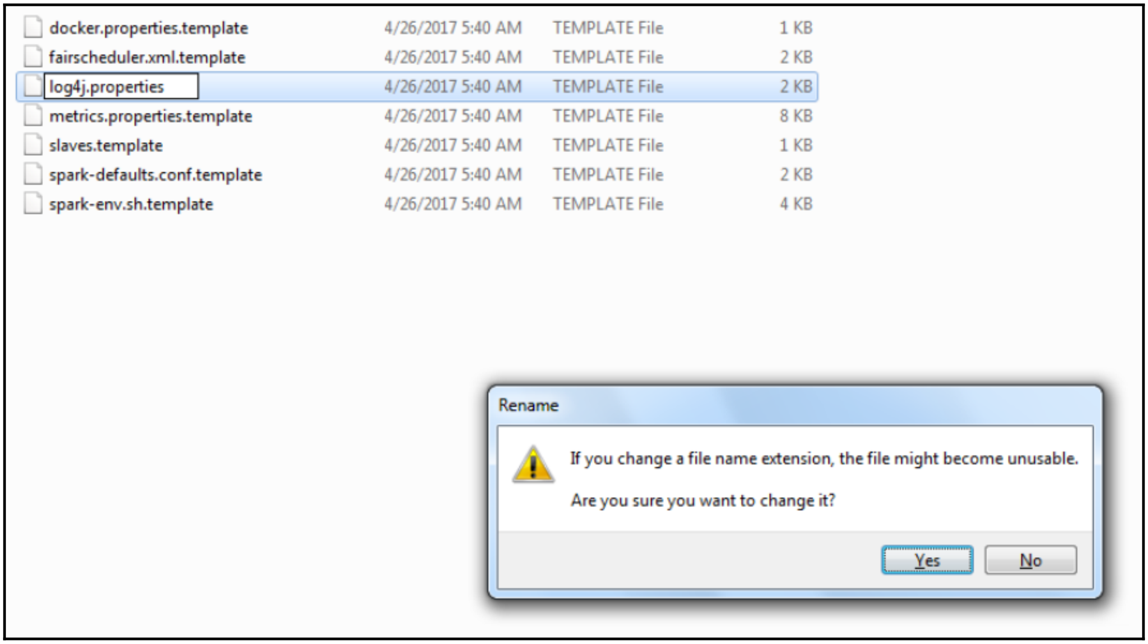
Language	Version	Size
Arabic (32 bit)	5.50 beta 3	1993 KB
Arabic (64 bit)	5.50 beta 3	2209 KB
Armenian (32 bit)	5.50 beta 3	1989 KB
Armenian (64 bit)	5.50 beta 3	2204 KB
Chinese Traditional (32 bit)	5.50 beta 3	2192 KB
Chinese Traditional (64 bit)	5.50 beta 3	2413 KB
English (32 bit)	5.50 beta 3	1947 KB
English (64 bit)	5.50 beta 3	2162 KB
Finnish (32 bit)	5.50 beta 3	1989 KB
Finnish (64 bit)	5.50 beta 3	2206 KB
French (32 bit)	5.50 beta 3	2044 KB
French (64 bit)	5.50 beta 3	2261 KB
German (32 bit)	5.50 beta 3	2067 KB
German (64 bit)	5.50 beta 3	2293 KB
Hungarian (32 bit)	5.50 beta 3	1987 KB
Hungarian (64 bit)	5.50 beta 3	2205 KB
Lithuanian (32 bit)	5.50 beta 3	2014 KB
Lithuanian (64 bit)	5.50 beta 3	2232 KB
Mongolian (32 bit)	5.50 beta 2	1995 KB
Mongolian (64 bit)	5.50 beta 2	2213 KB
Portuguese (32 bit)	5.50 beta 3	1988 KB
Portuguese (64 bit)	5.50 beta 3	2206 KB
Portuguese Brazilian (32 bit)	5.50 beta 3	3444 KB
Portuguese Brazilian (64 bit)	5.50 beta 3	3669 KB
Romanian (32 bit)	5.50 beta 2	2022 KB
Romanian (64 bit)	5.50 beta 2	2240 KB
Russian (32 bit)	5.50 beta 3	2094 KB
Russian (64 bit)	5.50 beta 3	2329 KB
Serbian Cyrillic (32 bit)	5.50 beta 3	2027 KB
Serbian Cyrillic (64 bit)	5.50 beta 3	2243 KB
Swedish (32 bit)	5.50 beta 3	1988 KB
Swedish (64 bit)	5.50 beta 3	2204 KB
Ukrainian (32 bit)	5.50 beta 3	1990 KB
Ukrainian (64 bit)	5.50 beta 3	2209 KB

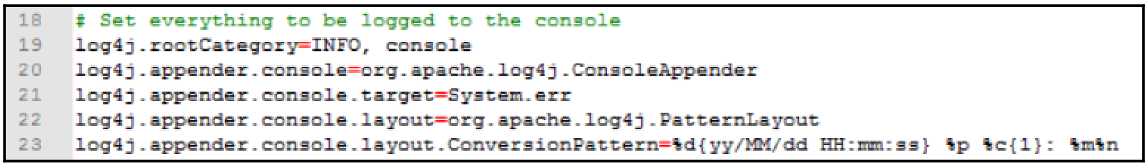

Clipboard
This PC › WDBlack (E:) › Downloads
Pictures
DataScience
Final
SetupVideos
work
OneDrive
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Name
Sky.h
skycode.zip
Sky-frag.glsl
Skymax_3.1_RWC-2016-03-13.zip
Skymaxx_CTD_Metar.zip
SkypeSetup.exe
SMP_crash.zip
SMP3-Best.zip
SMP3-Best2.zip
SMP3-beta7.zip
SMPMessages.h
source (1).csv
source (2).csv
source.csv
spark-1.6.1-bin-hadoop2.6 (1).tg
Open
Open with WinRAR
Extract files...
Extract Here
7-Zip
Scan with Windows Defender...
Open with...
TortoiseSVN
Restore previous versions
Send to
Cut
Copy
Create shortcut
Delete
Rename


Name
Date modified
Type
Size
bin 6/1/2017 6:42 PM File folder
conf 6/1/2017 6:42 PM File folder
data 6/1/2017 6:42 PM File folder
examples 6/1/2017 6:42 PM File folder
jars 6/1/2017 6:42 PM File folder
licenses 6/1/2017 6:42 PM File folder
python 6/1/2017 6:42 PM File folder
R 6/1/2017 6:42 PM File folder
sbin 6/1/2017 6:42 PM File folder
yarn 6/1/2017 6:42 PM File folder
LICENSE 4/26/2017 5:40 AM File 18 KB
NOTICE 4/26/2017 5:40 AM File 25 KB
RELEASE 4/26/2017 5:40 AM File 1 KB

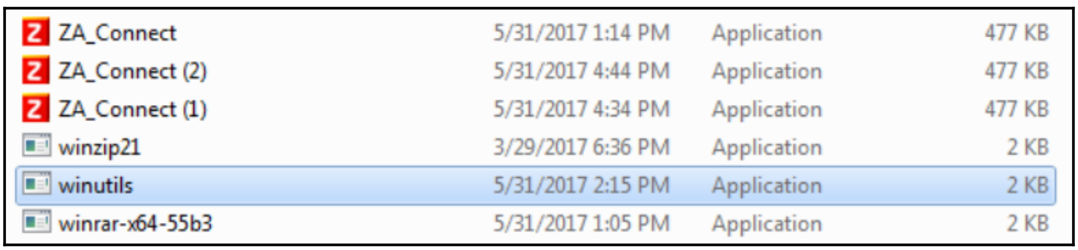

Computer ▸ Local Disk (C:) ▸ spark
Organize
Include in library
Share with
New folder
Name
Date modified
Type
C:\Users\nidhishas\Downloads\spark\spark-2.1.1-bin-hadoop2.6 (1)\spark-2.1.1-bin
Organize
Open
Include in library
Share with
New folder
Name
Date modified
Type
bin 6/1/2017 6:42 PM File folder
conf 6/1/2017 6:42 PM File folder
data 6/1/2017 6:42 PM File folder
examples 6/1/2017 6:42 PM File folder
jars 6/1/2017 6:42 PM File folder
licenses 6/1/2017 6:42 PM File folder
python 6/1/2017 6:42 PM File folder
R 6/1/2017 6:42 PM File folder
sbin 6/1/2017 6:42 PM File folder
yarn 6/1/2017 6:42 PM File folder
LICENSE 4/26/2017 5:40 AM File
NOTICE 4/26/2017 5:40 AM File
RELEASE 4/26/2017 5:40 AM File

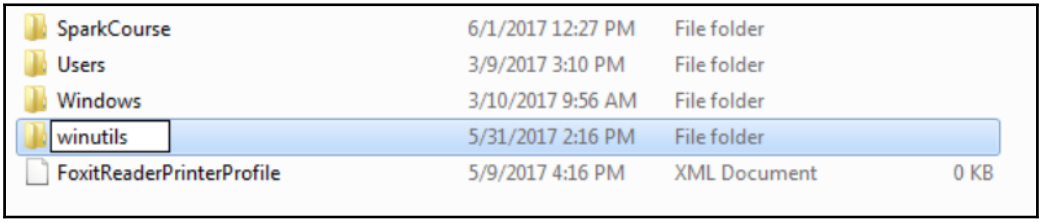

docker.properties.template 4/26/2017 5:40 AM TEMPLATE File 1 KB
fairscheduler.xml.template 4/26/2017 5:40 AM TEMPLATE File 2 KB
log4j.properties.template 4/26/2017 5:40 AM TEMPLATE File 2 KB
metrics.properties.template 4/26/2017 5:40 AM TEMPLATE File 8 KB
slaves.template 4/26/2017 5:40 AM TEMPLATE File 1 KB
spark-defaults.conf.template 4/26/2017 5:40 AM TEMPLATE File 2 KB
spark-env.sh.template 4/26/2017 5:40 AM TEMPLATE File 4 KB

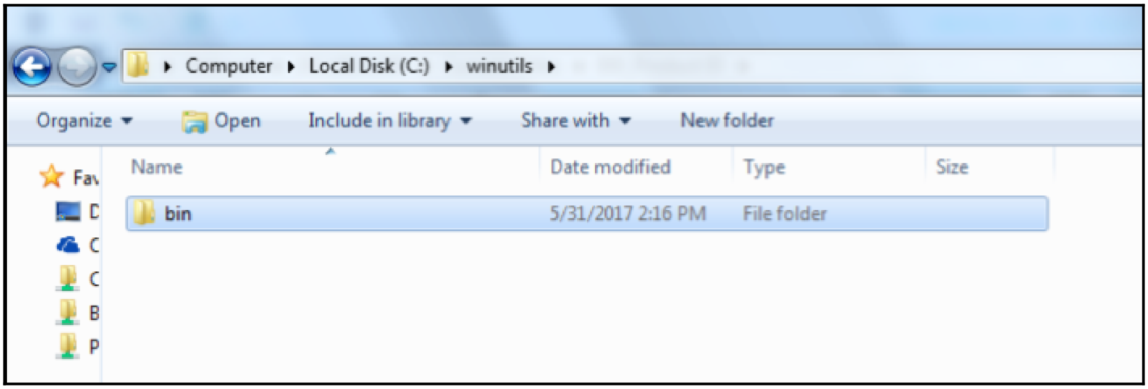

docker.properties.template 4/26/2017 5:40 AM TEMPLATE File 1 KB
fairscheduler.xml.template 4/26/2017 5:40 AM TEMPLATE File 2 KB
log4j.properties 4/26/2017 5:40 AM TEMPLATE File 2 KB
metrics.properties.template 4/26/2017 5:40 AM TEMPLATE File 8 KB
slaves.template 4/26/2017 5:40 AM TEMPLATE File 1 KB
spark-defaults.conf.template 4/26/2017 5:40 AM TEMPLATE File 2 KB
spark-env.sh.template 4/26/2017 5:40 AM TEMPLATE File 4 KB
Rename
If you change a file name extension, the file might become unusable.
Are you sure you want to change it?
Yes
No

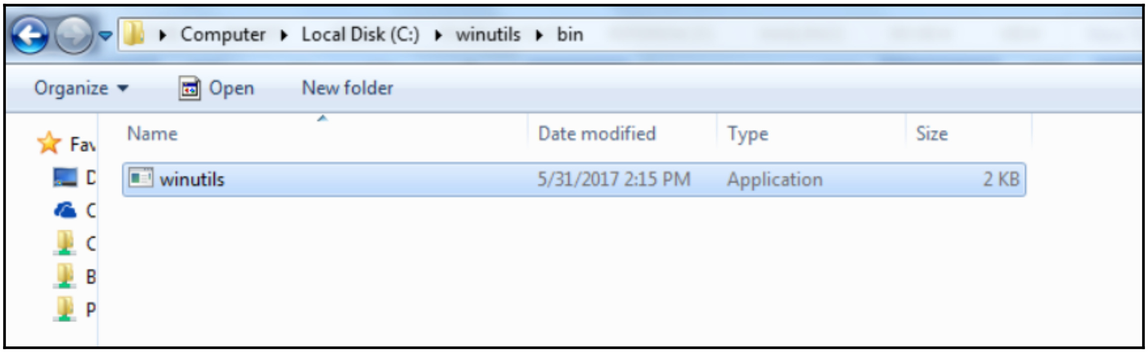

# Set everything to be logged to the console
log4j.rootCategory=INFO, console
log4j.appender.console=org.apache.log4j.ConsoleAppender
log4j.appender.console.target=System.err
log4j.appender.console.layout=org.apache.log4j.PatternLayout
log4j.appender.console.layout.ConversionPattern=%d{yy/MM/dd HH:mm:ss} %p %c{1}: %m%n

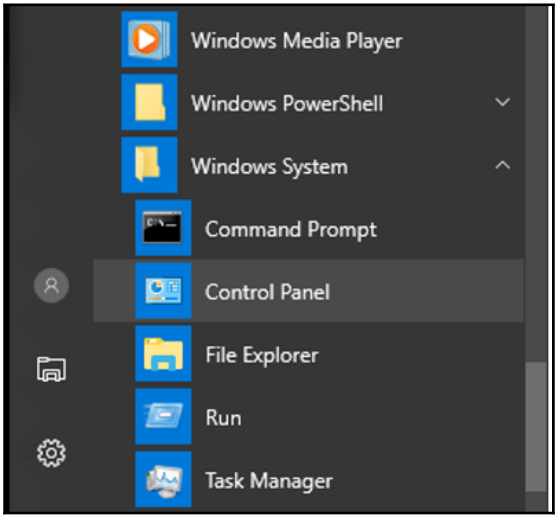

http://media.sundog-soft.com/winutils.exe
http://media.sundog-soft.com/winutils.exe
http://media.sundog-soft.com/winutils.exe - Google Search

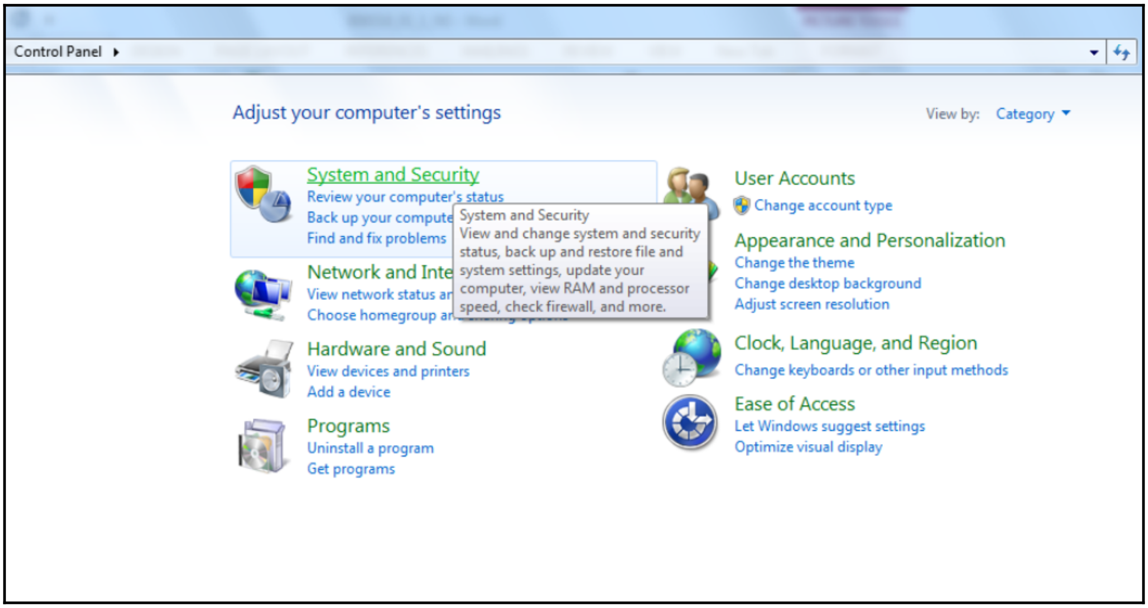

ZA_Connect 5/31/2017 1:14 PM Application 477 KB
ZA_Connect (2) 5/31/2017 4:44 PM Application 477 KB
ZA_Connect (1) 5/31/2017 4:34 PM Application 477 KB
winzip21 3/29/2017 6:36 PM Application 2 KB
winutils 5/31/2017 2:15 PM Application 2 KB
winrar-x64-55b3 5/31/2017 1:05 PM Application 2 KB

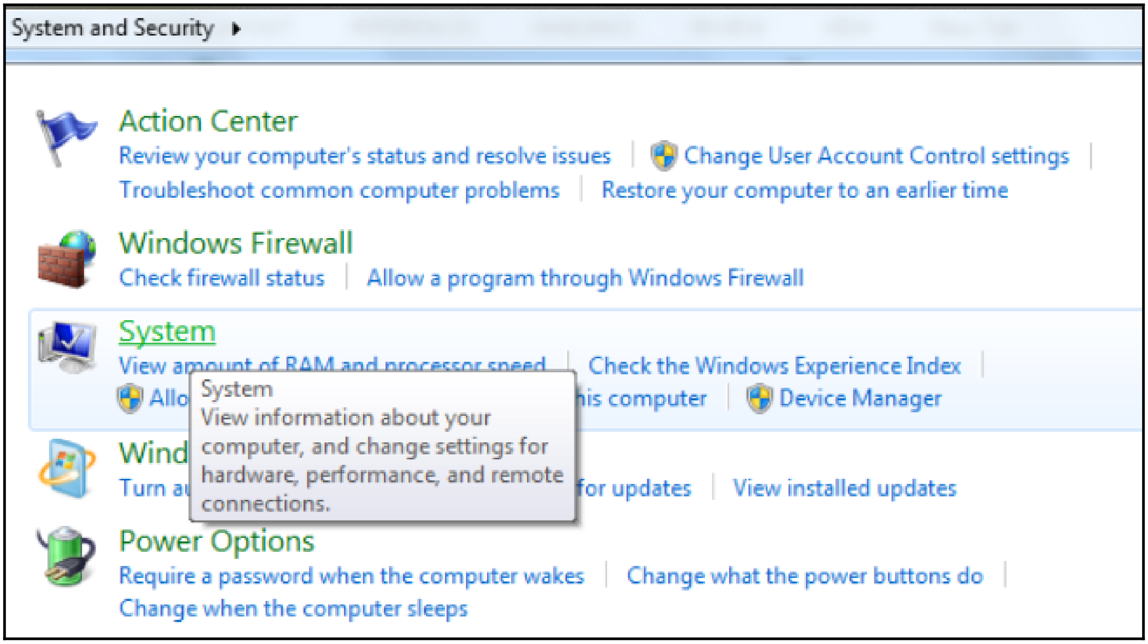

SparkCourse 6/1/2017 12:27 PM File folder
Users 3/9/2017 3:10 PM File folder
Windows 3/10/2017 9:56 AM File folder
winutils 5/31/2017 2:16 PM File folder
FoxitReaderPrinterProfile 5/9/2017 4:16 PM XML Document 0 KB

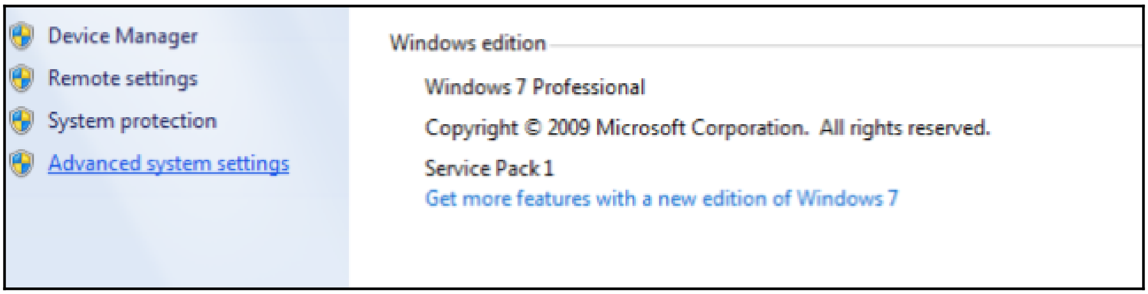

Computer ▸ Local Disk (C:) ▸ winutils ▸
Organize
Open
Include in library
Share with
New folder
Name
Date modified
Type
Size
bin 5/31/2017 2:16 PM File folder

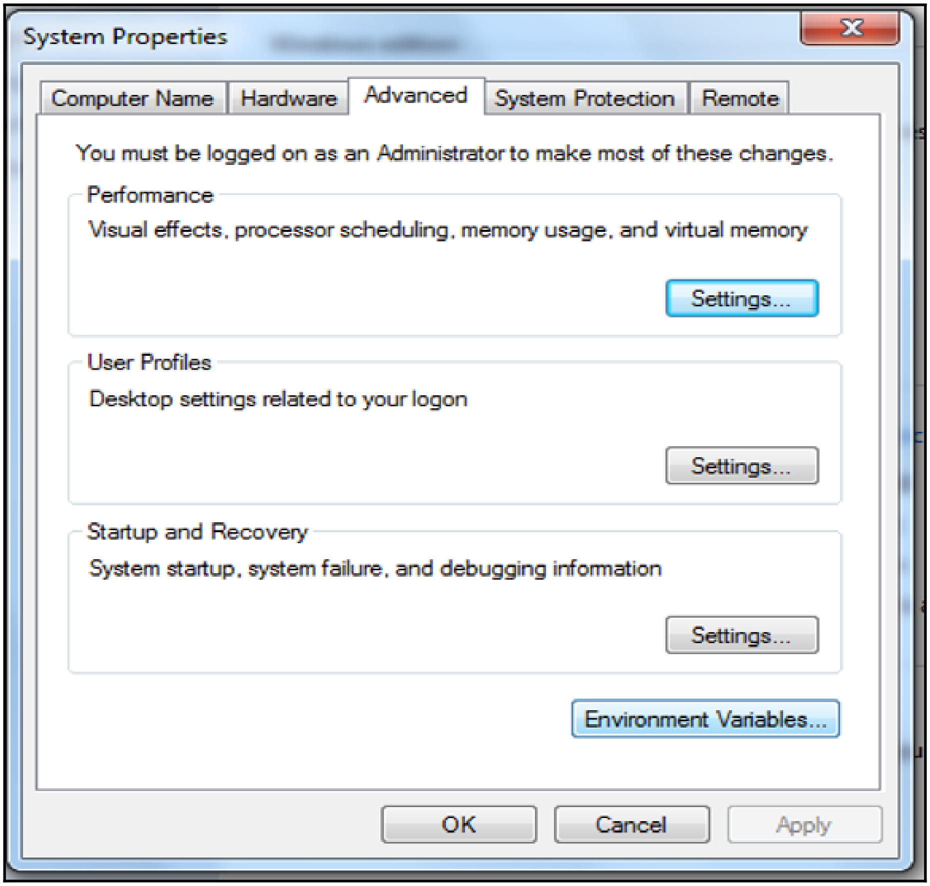

Computer ▸ Local Disk (C:) ▸ winutils ▸ bin
Organize
Open
New folder
Name
Date modified
Type
Size
winutils 5/31/2017 2:15 PM Application 2 KB

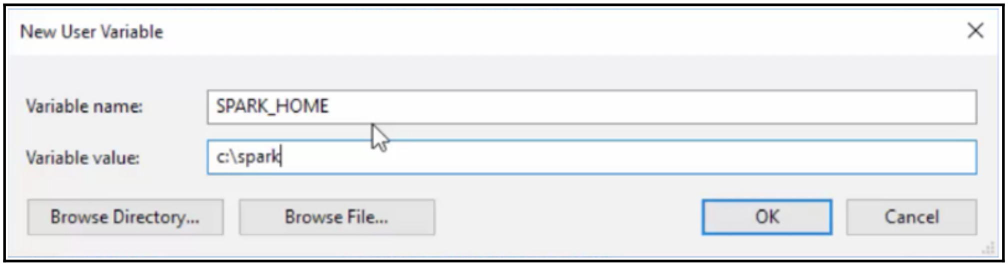

Windows Media Player
Windows PowerShell
Windows System
Command Prompt
Control Panel
File Explorer
Run
Task Manager

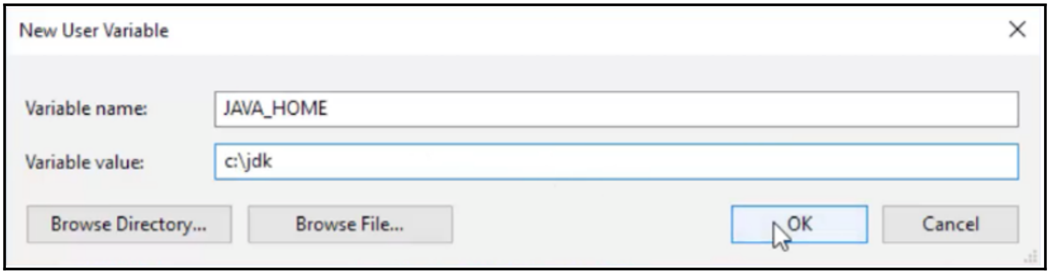

Control Panel
Adjust your computer's settings
View by: Category
System and Security
Review your computer's status
Find and fix problems
System and Security
View and change system and security status, back up and restore file and system settings, update your computer, view RAM and processor speed, check firewall, and more.
Hardware and Sound
View devices and printers
Add a device
Programs
Uninstall a program
Get programs
User Accounts
Change account type
Appearance and Personalization
Change the theme
Change desktop background
Adjust screen resolution
Clock, Language, and Region
Change keyboards or other input methods
Ease of Access
Let Windows suggest settings
Optimize visual display

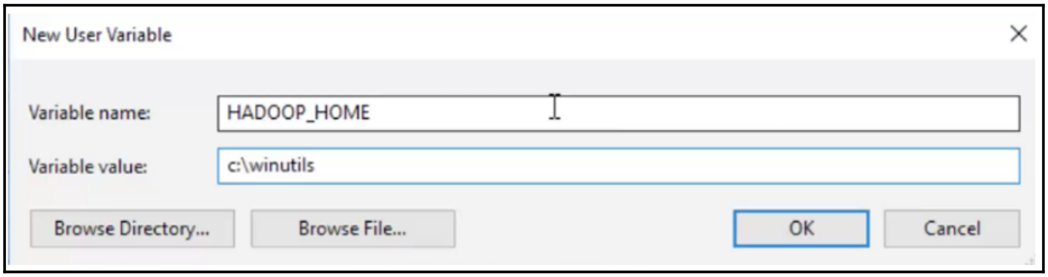

System and Security
Action Center
Review your computer's status and resolve issues
Change User Account Control settings
Troubleshoot common computer problems
Restore your computer to an earlier time
Windows Firewall
Check firewall status
Allow a program through Windows Firewall
System
View amount of RAM and processor speed
Check the Windows Experience Index
Allo
his computer
Device Manager
System
View information about your computer, and change settings for hardware, performance, and remote connections.
Wind
Turn a
for updates
View installed updates
Power Options
Require a password when the computer wakes
Change what the power buttons do
Change when the computer sleeps

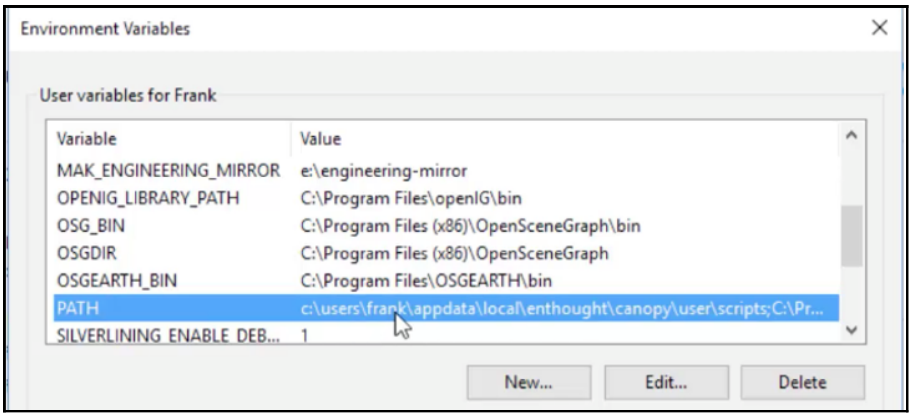

Device Manager
Remote settings
System protection
Advanced system settings
Windows edition
Windows 7 Professional
Copyright © 2009 Microsoft Corporation. All rights reserved.
Service Pack 1
Get more features with a new edition of Windows 7

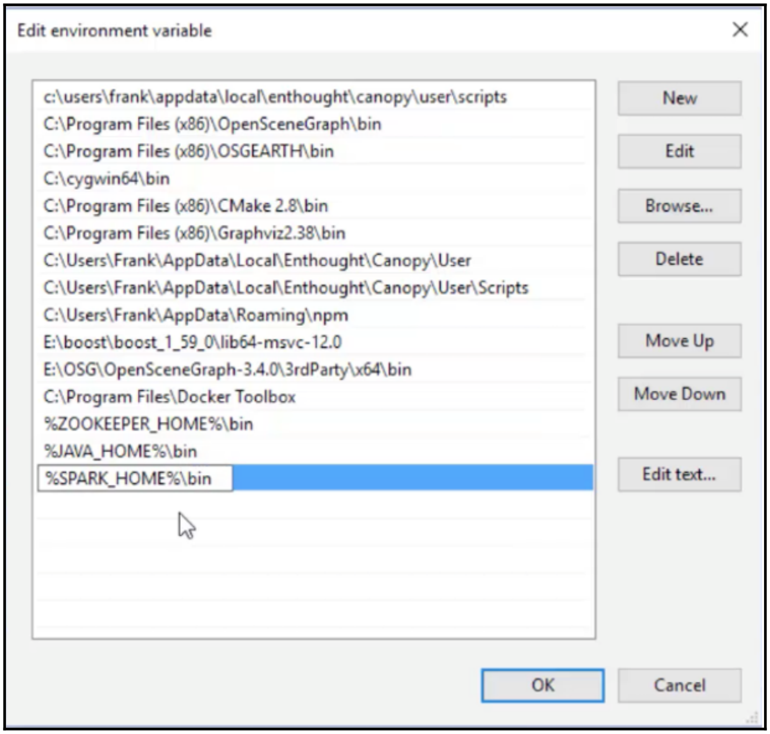

System Properties
Computer Name
Hardware
Advanced
System Protection
Remote
You must be logged on as an Administrator to make most of these changes.
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...
User Profiles
Desktop settings related to your logon
Settings...
Startup and Recovery
System startup, system failure, and debugging information
Settings...
Environment Variables...
OK
Cancel
Apply

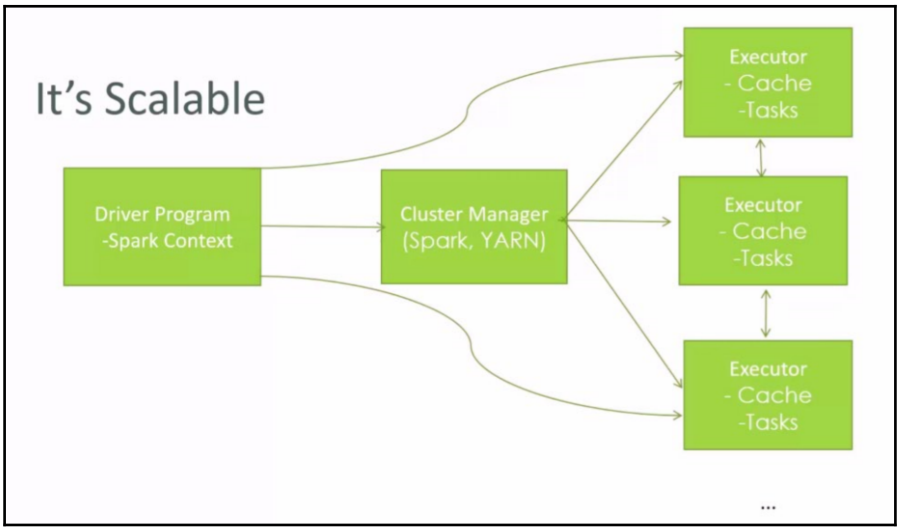

New User Variable
Variable name:
SPARK_HOME
Variable value:
c:\spark
Browse Directory...
Browse File...
OK
Cancel

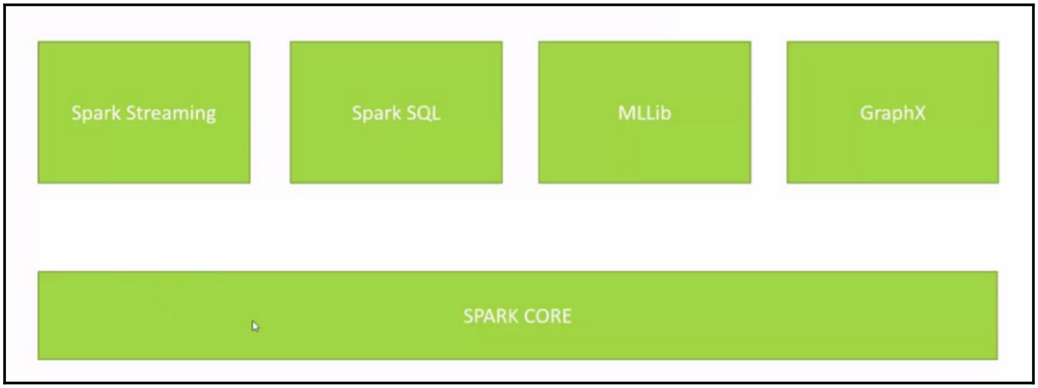

New User Variable
Variable name: JAVA_HOME
Variable value: c:\jdk
Browse Directory...
Browse File...
OK
Cancel


New User Variable
Variable name: HADOOP_HOME
Variable value: c:\winutils
Browse Directory...
Browse File...
OK
Cancel


Environment Variables
User variables for Frank
Variable
Value
MAK_ENGINEERING_MIRROR e:\engineering-mirror
OPENIG_LIBRARY_PATH C:\Program Files\openIG\bin
OSG_BIN C:\Program Files (x86)\OpenSceneGraph\bin
OSGDIR C:\Program Files (x86)\OpenSceneGraph
OSGEARTH_BIN C:\Program Files\OSGEARTH\bin
PATH c:\users\frank\appdata\local\enthought\canopy\user\scripts;C:\Pr...
SILVERLINING_ENABLE_DEB... 1
New...
Edit...
Delete


Edit environment variable
c:\users\frank\appdata\local\enthought\canopy\user\scripts
C:\Program Files (x86)\OpenSceneGraph\bin
C:\Program Files (x86)\OSGEARTH\bin
C:\cygwin64\bin
C:\Program Files (x86)\CMake 2.8\bin
C:\Program Files (x86)\Graphviz2.38\bin
C:\Users\Frank\AppData\Local\Enthought\Canopy\User
C:\Users\Frank\AppData\Local\Enthought\Canopy\User\Scripts
C:\Users\Frank\AppData\Roaming\npm
E:\boost\boost_1_59_0\lib64-msvc-12.0
E:\OSG\OpenSceneGraph-3.4.0\3rdParty\x64\bin
C:\Program Files\Docker Toolbox
%ZOOKEEPER_HOME%\bin
%JAVA_HOME%\bin
%SPARK_HOME%\bin
New
Edit
Browse...
Delete
Move Up
Move Down
Edit text...
OK
Cancel


It's Scalable
Driver Program
-Spark Context
Cluster Manager
(Spark, YARN)
Executor
- Cache
-Tasks
Executor
- Cache
-Tasks
Executor
- Cache
-Tasks
...



Python code to square numbers in a data set:

```
nums = sc.parallelize([1, 2, 3, 4])
squared = nums.map(lambda x: x * x).collect()
```

Scala code to square numbers in a data set:

```
val nums = sc.parallelize(List(1, 2, 3, 4))
val squared = nums.map(x => x * x).collect()
```

SparkDecisionTree.py

```python
from pyspark.mllib.regression import LabeledPoint
from pyspark.mllib.tree import DecisionTree
from pyspark import SparkConf, SparkContext
from numpy import array

# Boilerplate Spark stuff:
conf = SparkConf().setMaster("local").setAppName("SparkDecisionTree")
sc = SparkContext(conf = conf)

# Some functions that convert our CSV input data into numerical
# features for each job candidate
def binary(YN):
    if (YN == 'Y'):
        return 1
    else:
        return 0

def mapEducation(degree):
    if (degree == 'BS'):
        return 1
    elif (degree =='MS'):
        return 2
    elif (degree == 'PhD'):
        return 3
    else:
        return 0

```

SparkDecisionTree.py

```python
from pyspark.mllib.regression import LabeledPoint
from pyspark.mllib.tree import DecisionTree
from pyspark import SparkConf, SparkContext
from numpy import array

# Boilerplate Spark stuff:
conf = SparkConf().setMaster("local").setAppName("SparkDecisionTree")
sc = SparkContext(conf = conf)

# Some functions that convert our CSV input data into numerical
# features for each job candidate
def binary(YN):
    if (YN == 'Y'):
        return 1
    else:
        return 0

def mapEducation(degree):
    if (degree == 'BS'):
        return 1
    elif (degree =='MS'):
        return 2
    elif (degree == 'PhD'):
        return 3
    else:
        return 0

```

```python
rawData = sc.textFile("e:/sundog-consult/udemy/datascience/PastHires.csv")
header = rawData.first()
rawData = rawData.filter(lambda x:x != header)

# Split each line into a list based on the comma delimiters
csvData = rawData.map(lambda x: x.split(","))

# Convert these lists to LabeledPoints
trainingData = csvData.map(createLabeledPoints)

# Create a test candidate, with 10 years of experience, currently employed,
# 3 previous employers, a BS degree, but from a non-top-tier school where
# he or she did not do an internship. You could of course load up a whole
# huge RDD of test candidates from disk, too.
testCandidates = [ array([10, 1, 3, 1, 0, 0])]
testData = sc.parallelize(testCandidates)
```

	A	B	C	D	E	F	G
1	Years Exp	Employed	Previous	Level of E	Top-tier s	Interned	Hired
2	10	Y	4	BS	N	N	Y
3	0	N	0	BS	Y	Y	Y
4	7	N	6	BS	N	N	N
5	2	Y	1	MS	Y	N	Y
6	20	N	2	PhD	Y	N	N
7	0	N	0	PhD	Y	Y	Y
8	5	Y	2	MS	N	Y	Y
9	3	N	1	BS	N	Y	Y
10	15	Y	5	BS	N	N	Y
11	0	N	0	BS	N	N	N
12	1	N	1	PhD	Y	N	N
13	4	Y	1	BS	N	Y	Y
14	0	N	0	PhD	Y	N	Y

	A	B	C	D	E	F	G
1	Years Exp	Employed	Previous	Level of E	Top-tier s	Interned	Hired
2	10	Y	4	BS	N	N	Y
3	0	N	0	BS	Y	Y	Y

```
Hire prediction:
1.0
Learned classification tree model:
DecisionTreeModel classifier of depth 4 with 9 nodes
  If (feature 1 in {0.0})
   If (feature 5 in {0.0})
    If (feature 0 <= 0.0)
     If (feature 3 in {1.0})
      Predict: 0.0
     Else (feature 3 not in {1.0})
      Predict: 1.0
    Else (feature 0 > 0.0)
     Predict: 0.0
   Else (feature 5 not in {0.0})
    Predict: 1.0
  Else (feature 1 not in {0.0})
   Predict: 1.0
```

```
Hire prediction:
1.0
Learned classification tree model:
DecisionTreeModel classifier of depth 4 with 9 nodes
  If (feature 1 in {0.0})
   If (feature 5 in {0.0})
    If (feature 0 <= 0.0)
     If (feature 3 in {1.0})
      Predict: 0.0
     Else (feature 3 not in {1.0})
      Predict: 1.0
    Else (feature 0 > 0.0)
     Predict: 0.0
   Else (feature 5 not in {0.0})
    Predict: 1.0
  Else (feature 1 not in {0.0})
   Predict: 1.0
```

Years Exp	Employed	Previous e	Level of E	Top-tier s	Interned	Hired
10	Y	4	BS	N	N	Y
0	N	0	BS	Y	Y	Y
7	N	6	BS	N	N	N
2	Y	1	MS	Y	N	Y
20	N	2	PhD	Y	N	N
0	N	0	PhD	Y	Y	Y
5	Y	2	MS	N	Y	Y
3	N	1	BS	N	Y	Y
15	Y	5	BS	N	N	Y
0	N	0	BS	N	N	N
1	N	1	PhD	Y	N	N
4	Y	1	BS	N	Y	Y
0	N	0	PhD	Y	N	Y

```python
K = 5

# Boilerplate Spark stuff:
conf = SparkConf().setMaster("local").setAppName("SparkKMeans")
sc = SparkContext(conf = conf)

#Create fake income/age clusters for N people in k clusters
def createClusteredData(N, k):
    random.seed(10)
    pointsPerCluster = float(N)/k
    X = []
    for i in range (k):
        incomeCentroid = random.uniform(20000.0, 200000.0)
        ageCentroid = random.uniform(20.0, 70.0)
        for j in range(int(pointsPerCluster)):
            X.append([random.normal(incomeCentroid, 10000.0), random.normal(ageCentroid, 2.0)])
    X = array(X)
```

```
(Canopy 64bit) E:\sundog-consult\Udemy\DataScience>spark-submit SparkKMeans.py
Counts by value:
defaultdict(<type 'int'>, {0: 21, 1: 20, 2: 20, 3: 20, 4: 19})
Cluster assignments:
[3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 1, 1, 1, 1, 1, 1
, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
4, 4, 4, 4, 4, 4, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Within Set Sum of Squared Error = 19.9732765798

(Canopy 64bit) E:\sundog-consult\Udemy\DataScience>_
```

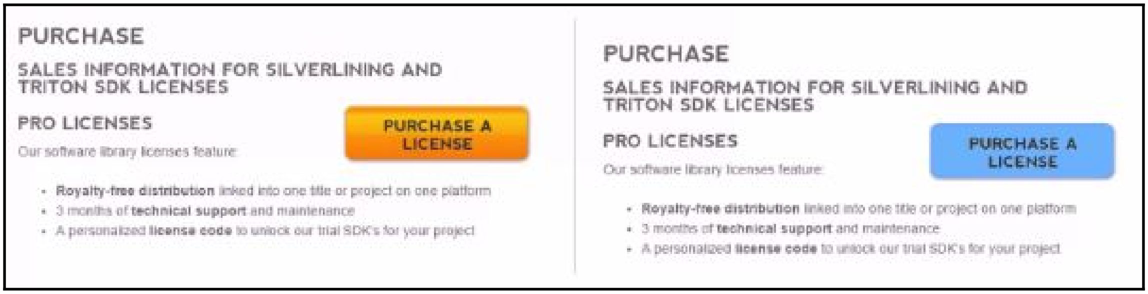
```
from pyspark import SparkConf, SparkContext
from pyspark.mllib.feature import HashingTF
from pyspark.mllib.feature import IDF

# Boilerplate Spark stuff:
conf = SparkConf().setMaster("local").setAppName("SparkTFIDF")
sc = SparkContext(conf = conf)

# Load documents (one per line).
rawData = sc.textFile("e:/sundog-consult/Udemy/DataScience/subset-small.tsv")
fields = rawData.map(lambda x: x.split("\t"))
documents = fields.map(lambda x: x[3].split(" "))

# Store the document names for later:
documentNames = fields.map(lambda x: x[1])

# Now hash the words in each document to their term frequencies:
hashingTF = HashingTF(100000)  #100K hash buckets just to save some memory
tf = hashingTF.transform(documents)

# At this point we have an RDD of sparse vectors representing each document,
# where each value maps to the term frequency of each unique hash value.

# Let's compute the TF*IDF of each term in each document:
tf.cache()
idf = IDF(minDocFreq=2).fit(tf)
tfidf = idf.transform(tf)
```

```
Best document for Gettysburg is:
(29.777067781559442, u'Abraham Lincoln')
```

```python
from __future__ import print_function

from pyspark.ml.regression import LinearRegression

from pyspark.sql import SparkSession
from pyspark.ml.linalg import Vectors

if __name__ == "__main__":

    # Create a SparkSession (Note, the config section is only for Windows!)
    spark = SparkSession.builder.config("spark.sql.warehouse.dir", "file:///C:/temp").appName("LinearRegression").getOrCreate()

    # Load up our data and convert it to the format MLLib expects.
    inputLines = spark.sparkContext.textFile("regression.txt")
    data = inputLines.map(lambda x: x.split(",")).map(lambda x: (float(x[0]), Vectors.dense(float(x[1]))))

    # Convert this RDD to a DataFrame
    colNames = ["label", "features"]
    df = data.toDF(colNames)

    # Note, there are lots of cases where you can avoid going from an RDD to a DataFrame.
    # Perhaps you're importing data from a real database. Or you are using structured streaming
```

```
(0.8643234408131227, 1.2)
(0.9571202568137612, 1.31)
(0.9428438235828938, 1.34)
(1.0499170728143994, 1.36)
(0.9571202568137612, 1.38)
(1.057055289429833, 1.41)
(0.9142909571211588, 1.44)
(0.9285673903520263, 1.47)
(1.1212992389687366, 1.5)
(0.8928763072748577, 1.5)
(1.2569253546619772, 1.53)
(1.135575672199604, 1.53)
(1.1070228057378693, 1.53)
(1.2355107048156762, 1.54)
(1.1498521054304716, 1.55)
(1.164128538661339, 1.56)
(1.135575672199604, 1.59)
(1.1784049718922063, 1.61)
(1.392551470355218, 1.78)
(1.214096054969375, 1.8)
(1.264063571277411, 1.82)
(1.342583954047182, 1.86)
(1.4924865029712902, 2.09)
```

Chapter 10: Testing and Experimental Design




```
Out[1]: Ttest_indResult(statistic=-14.075196812141339, pvalue=8.8277957363196977e-45)
```

```
Out[2]: Ttest_indResult(statistic=0.088886198511817435, pvalue=0.92917324220169051)
```

```
Out[6]: Ttest_indResult(statistic=0.20964627681745385, pvalue=0.83394397202032966)
```

```
Out[9]: Ttest_indResult(statistic=-0.075342911693641518, pvalue=0.93994188742749496)
```

```
Out[10]: Ttest_indResult(statistic=0.0, pvalue=1.0)
```