

# Python ile Enerji Analizi I

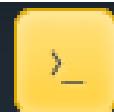
Başarış Sanlı  
*[barissanli.com/python](http://barissanli.com/python)*

23 Mart 2019

# Python

- 1994 'de 1.0
- 2008'de Python 3.0

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
```



# Python Zen'i

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Flat is better than nested.
- Sparse is better than dense.
- Readability counts.
- Special cases aren't special enough to break the rules.
- Although practicality beats purity.
- Errors should never pass silently.
- Unless explicitly silenced.
- In the face of ambiguity, refuse the temptation to guess.
- There should be one -- and preferably only one -- obvious way to do it.
- Although that way may not be obvious at first unless you're Dutch.
- Now is better than never.
- Although never is often better than \*right\* now.
- If the implementation is hard to explain, it's a bad idea.
- If the implementation is easy to explain, it may be a good idea.  
barissanli.com/python
- Namespaces are one honking great idea -- let's do more of those

# Tarihsel tecrübe

- 2011'de Doğalgaz şirketleri clustering için
- 2013'de ODET (On Günlük Doğal gaz ve Elektrik Talep Modeli)
  - <http://www.barissanli.com/calismalar/2013/ODET-v01.pdf>
- 2017'de Güneş panel sistem yönetimi
  - <http://www.barissanli.com/calismalar/2017/20170216-bsanli-iot.pdf>
- 2018'de barissanli.com/python

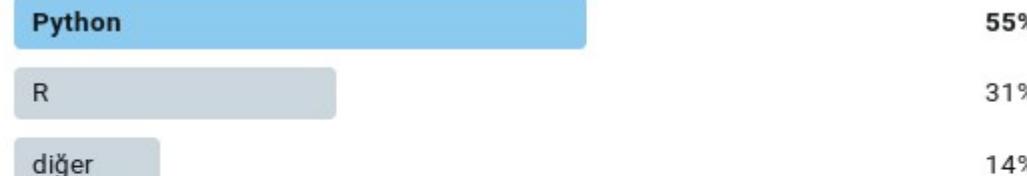
# Anket



**Baris Sanli**  
@barissanli

Enerji sektörüne gireceklerle Python u mu R mı  
öğretilmeli/Should we teach R or Python to newcomers  
in Energy business

[Translate Tweet](#)



249 votes · Final results

9:04 PM · Jan 7, 2019 · [Twitter Web App](#)



**Emin Batur Dizdar** @BaturDizdar · Jan 8

Replying to @barissanli  
R ile basic işlemler daha kolay öğrenilip uygulamaya geçebilir. Ayrıca "matrix oriented" olduğundan datayla doğrudan çalışmaya daha uygun. Python overall'da daha kapsamlı ancak öğrenmesi daha uzun. Python'la daha generic uygulamalar yazılabilir.



**Rüçhan Kaya** @ruchankaya · Jan 9

Replying to @barissanli  
R istatistikciacidan daha başarılı, çokça da paket yazılıyor bu sıralar. Python'a uygulama açısından daha iyi ve daha anlaşılır.



**B. NAZIM BAYRAKTAR** @bnbayraktar · Jan 7

Replying to @barissanli  
yeni gireceklerle R



**Gökhan Önal** @onal\_gokhan · Jan 7

Replying to @barissanli  
Python is very strong with its math libraries, hence it is used very often with power analysis tools. Moreover, a power system analysis tool can be developed in python, as we do:



**Orçun Başlak** @OrcunBaslak · Jan 7

Replying to @barissanli  
Pek karşılaşmamak gereklidir. R daha istatistik tarafa hitap ettiğinde Python genel kullanımına uygun. R fonksiyonel programlama, Python nesne tabanlı. R'a kısa bir girişten sonra esas Python'a geçilebilir.



[barissanli.com/python](#)

<https://twitter.com/barissanli/status/1082337159825248261>

# Günlük petrol fiyatları

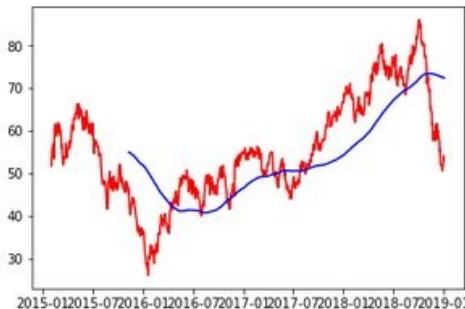
Python ile günlük petrol fiyatlarını indirerek, son 200 günlük ortalama ile birlikte grafiklemek

Barış Sanlı

```
[4]: ### %matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

xls = pd.ExcelFile("https://www.eia.gov/dnav/pet/hist_xls/RBRTEd.xls") #internetten günlük petrol fiyatlarını indir
data=xls.parse('Data 1',skiprows=2) # exceldeki 2 satırı at
df=pd.DataFrame(data) # veriyi daha rahat işlemek için veri çerçevesine çevir
verisayisi=1000 # son 1000 veri
son1000=df.iloc[-verisayisi:,1] # son 1000 veri -> excel deki 2.sütun
son1000d=df.iloc[-verisayisi:,0] # son 1000 verinin tarihler -> exceldeki 1.sütun
plt.plot(son1000d, son1000, "r", son1000d,son1000.rolling(window=200).mean(),"b") #grafikleyelim kırmızı ile veri, mavi ile son 200 gün ortalama
```

[4]: [<matplotlib.lines.Line2D at 0x7f31ae6a2ef0>, <matplotlib.lines.Line2D at 0x7f31ae6be390>]



# Anaconda

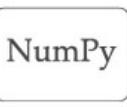
## Anaconda Distribution

The World's Most Popular Python/R Data Science Platform

[Download](#)

The open-source [Anaconda Distribution](#) is the easiest way to perform Python/R data science and machine learning on Linux, Windows, and Mac OS X. With over 11 million users worldwide, it is the industry standard for developing, testing, and training on a single machine, enabling *individual data scientists* to:

- Quickly download 1,500+ Python/R data science libraries
- Manage libraries, dependencies, and environments
- Develop and train machine learning and deep learning models with [TensorFlow](#) and [Theano](#)
- Analyze data with scalability and performance with [Apache Spark](#)
- Visualize results with [Matplotlib](#), [Bokeh](#), [DataViz](#), and [HoloViews](#)



### Anaconda 2018.12 for macOS Installer

#### Python 3.7 version

[Download](#)

64-Bit Graphical Installer (652.7 MB)  
64-Bit Command Line Installer (557 MB)

#### Python 2.7 version

[Download](#)

64-Bit Graphical Installer (640.7 MB)  
64-Bit Command Line Installer (547 MB )

[TensorFlow](#)

[CONDA](#)

# Miniconda

- Daha küçük
- Fakat tüm paketleri kendin indirirsin
- Yalın

<a href="#">Miniconda2-latest-Linux-x86_64.sh</a>	42.8M
<a href="#">Miniconda2-latest-MacOSX-x86_64.pkg</a>	38.4M
<a href="#">Miniconda2-latest-MacOSX-x86_64.sh</a>	33.1M
<a href="#">Miniconda2-latest-Windows-x86.exe</a>	55.0M
<a href="#">Miniconda2-latest-Windows-x86_64.exe</a>	59.2M
<a href="#">Miniconda3-4.5.12-Linux-x86.sh</a>	62.7M
<a href="#">Miniconda3-4.5.12-Linux-x86_64.sh</a>	66.6M
...	

# Jupyter : Arayüz



jupyter spectrogram Last Checkpoint: an hour ago (autosaved) | Python 2

In [1]: `from scipy.io import wavfile  
rate, x = wavfile.read('test_mono.wav')`

In [2]: `import matplotlib.pyplot as plt  
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))  
ax1.plot(x); ax1.set_title('Raw audio signal')  
ax2.specgram(x); ax2.set_title('Spectrogram')  
plt.show()`

The image shows two plots side-by-side. The left plot, titled "Raw audio signal", is a line graph of a mono audio signal over time. The x-axis ranges from 0 to 250,000 samples, and the y-axis ranges from -40,000 to 40,000 amplitude. The signal is highly noisy with several sharp peaks. The right plot, titled "Spectrogram", is a heatmap showing frequency over time. The x-axis ranges from 0 to 120,000 samples, and the y-axis ranges from 0.0 to 1.0 frequency. The spectrogram displays vertical bands of energy, with a prominent diagonal trend indicating the fundamental frequency of the sound.

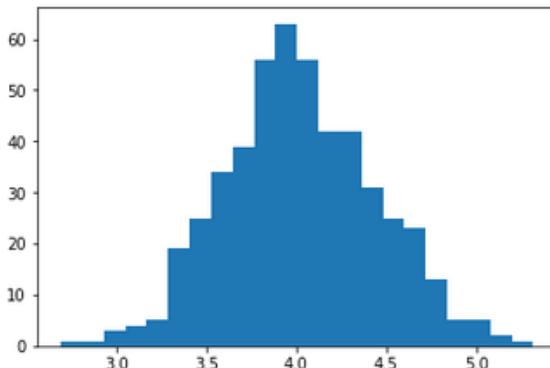
# Etkileşimli

```
[1]: import warnings
warnings.filterwarnings('ignore')
%pylab inline
from ipywidgets import interact, interactive, fixed, interact_manual
import ipywidgets as widgets
```

Populating the interactive namespace from numpy and matplotlib

```
[2]: def func_gen(mean, std, Size):
    hist(np.random.normal(mean, std, size=Size), bins=int(np.sqrt(Size)))
    return 0
```

```
[3]: p=interactive(func_gen,mean=(0,5,1),std=(0.1,50,0.1),Size=(5,1000,10))
display(p)
```



# Programlamaya giriş

- Değişken ve fonksiyon



# Petrol fiyatı

```
petrol_fiyati=70
print("Petrol fiyatı ", petrol_fiyati)
```

```
Petrol fiyatı 70
```

```
def petrol_talebi ( petrol_fiyati):
    if petrol_fiyati>100:
        return 0.5
    elif petrol_fiyati<50:
        return 1.6
    else :
        return 1.2
```

```
petrol_talebi(80)
```

```
1.2
```

```
brent_fiyat=60
petrol_talebi(brent_fiyat)
```

```
1.2
```

```
[4]: brent_fiyat=60
petrol_talebi(brent_fiyat)
```

```
[4]: 1.2
```

```
[7]: print ("Brent petrol fiyatı ", brent_fiyat)
print (brent_fiyat," $/v de petrol talep artışı", petrol_talebi(brent_fiyat))
```

```
Brent petrol fiyatı 60
60 $/v de petrol talep artışı 1.2
```

```
[8]: for brent_fiyat in range(80,120,10):
    print (brent_fiyat," $/v de petrol talep artışı", petrol_talebi(brent_fiyat))
```

```
80 $/v de petrol talep artışı 1.2
90 $/v de petrol talep artışı 1.2
100 $/v de petrol talep artışı 1.2
110 $/v de petrol talep artışı 0.5
```

# Jupyter (arayüz komutları)

- Esc ile command mode
  - Y ve M tuşları
  - A (üste – above) , B(asağı – below)
  - Shift (tamamlama)
  - Shift + Tab (komut açıklama)

# Jupyter Markdown

- # Başlık
- ## Alt başlık
- \*\*komut\*\*
- \*kelime\*
- \*\*\* düz çizgi

The screenshot shows a Jupyter Notebook interface. At the top, there's a toolbar with icons for file operations and a dropdown menu set to 'Markdown'. Below the toolbar, a horizontal line separates the header from the content area. In the content area, there's a section of text labeled 'ilk seviye' (first level) in bold. Below this, there's a code cell containing Python code related to a石油模型 (oil model). The code includes variable definitions for oil price, OPEC output, and non-OPEC output.

```
# Oyuncu tabanlı - agent based Basit bir Dünya Petrol Modeli
**petrol fiyatları** çalışması

petrol_fiyati=60
opec_uretim=32
nopec_uretim=68
```

# Oyuncu tabanlı model

- Üretim
  - OPEC
  - NOPEC (non OPEC): OPEC dışı
- Tüketim
  - OECD
  - NOECD (non OECD): OECD dışı
- Petrol fiyatı
- Fiyat hareketi -> net üretim ve tüketim sonucu

# Kodlama – Dünya Petrol Modeli

```
petrol_fiyati=60
opec_uretim=32
nopec_uretim=68

oecd_talep=30
noecd_talep=70

petrol_fiyati

60

print("petrol fiyatı : ",petrol_fiyati)

petrol fiyatı : 60

opec_uretim_artisi=1 # yüzde yani 1+ yüzde artış oranı
nopec_uretim_artisi= 1 # yüzde

oecd_talep_artisi=1.005 # yüzde
noecd_talep_artisi=1.013 # yüzde

rezerv_kapasite=2.5 # mv/g
```

```
if petrol_fiyati<60:
    #talep
    oecd_talep_artisi=oecd_talep_artisi+0.003
    noecd_talep_artisi=noecd_talep_artisi+0.006
    #üretim
    opec_uretim_artisi=opec_uretim_artisi-0.01
    nopec_uretim_artisi=nopec_uretim_artisi-0.02
```

```
if petrol_fiyati>60:
    #talep
    oecd_talep_artisi=oecd_talep_artisi-0.003
    noecd_talep_artisi=noecd_talep_artisi-0.006
    #üretim
    opec_uretim_artisi=opec_uretim_artisi+0.01
    nopec_uretim_artisi=nopec_uretim_artisi+0.02
```

```
#talepleri hesaplayalım
oecd_talep=oecd_talep*oecd_talep_artisi
noecd_talep=noecd_talep*noecd_talep_artisi
```

```
#simdi net talep farkı
net_talep=oecd_talep+noecd_talep
```

```
#üretimleri hesaplayalım
opec_uretim=opec_uretim*opec_uretim_artisi
nopec_uretim=nopec_uretim*nopec_uretim_artisi
```

```
#net üretim
net_uretim=opec_uretim+nopec_uretim
```

# Fiyat hareketi ve sonuç

```
#net_uretim  
net_uretim=opec_uretim+nopec_uretim  
  
rezerv_kapasite=net_uretim-net_talep+rezerv_kapasite  
  
fiyat_hareketi=((rezerv_kapasite-2.5)**4)*(10)  
  
fiyat_hareketi  
12.624769600000107  
  
rezerv_kapasite  
1.439999999999977  
  
net_uretim  
100
```

```
print("Petrol fiyatı:",petrol_fiyati)  
print("OPEC üretimi:",opec_uretim," OPEC üretim artışı:", opec_uretim_artisi)  
print("Non-OPEC üretimi:", nopec_uretim, " Non-OPEC üretim artışı:", nopec_uretim_artisi)  
print("")  
print("OECD talebi: ", oecd_talep , " OECD talep artışı:", oecd_talep_artisi)  
print("Non-OECD talep:", noecd_talep, " Non-OECD talep artışı:", noecd_talep_artisi)  
print("")  
print("Dünya talebi:",net_talep, " Dünya üretimi", net_uretim)
```

```
Petrol fiyatı: 60  
OPEC üretimi: 32 OPEC üretim artışı: 1  
Non-OPEC üretimi: 68 Non-OPEC üretim artışı: 1  
  
OECD talebi: 30.15 OECD talep artışı: 1.005  
Non-OECD talep: 70.91 Non-OECD talep artışı: 1.013  
  
Dünya talebi: 101.06 Dünya üretimi 100
```

# Simulasyon için

- Her yıl tekrar tüm değerler hesaplanmalı
- For bunun için

```
for yil in range(1,10):
    print(yil, end=" ")
```

1 2 3 4 5 6 7 8 9

```
for yil in range(2019,2030):
    print(yil,end=" ")
```

2019 2020 2021 2022 2023 2024 2025 2026 2027 2028 2029

```
for yil in range(2019,2030, 5):
    print(yil, end= " ")
```

2019 2024 2029

# Tüm kod

```
petrol_fiyati=60
opec_uretim=32
nopec_uretim=68

oecd_talep=30
noecd_talep=70

opec_uretim_artisi=1 # yüzde yani 1+ yüzde artış oranı
nopec_uretim_artisi= 1 # yüzde

oecd_talep_artisi=1.005 # yüzde
noecd_talep_artisi=1.013 # yüzde

rezerv_kapasite=2.5 # mv/g

for yil in range(2019,2030):

    # petrol fiyatları 60$'ın altında ise

    if petrol_fiyati<60:
        #talep
        oecd_talep_artisi=oecd_talep_artisi+0.003
        noecd_talep_artisi=noecd_talep_artisi+0.006
        #üretim
        opec_uretim_artisi=opec_uretim_artisi-0.01
        nopec_uretim_artisi=nopec_uretim_artisi+0.02

    # petrol fiyatları 60$'ın üstünde ise
    if petrol_fiyati>60:
        #talep
        oecd_talep_artisi=oecd_talep_artisi-0.003
        noecd_talep_artisi=noecd_talep_artisi-0.006
        #üretim
        opec_uretim_artisi=opec_uretim_artisi+0.01
        nopec_uretim_artisi=nopec_uretim_artisi+0.02
```

```
# şimdi dengeyi hesaplayalım

#talepleri hesaplayalım
oecd_talep=oecd_talep*oecd_talep_artisi
noecd_talep=noecd_talep*noecd_talep_artisi

#simdi net talep farkı
net_talep=oecd_talep+noecd_talep

#üretimleri hesaplayalım
opec_uretim=opec_uretim*opec_uretim_artisi
nopec_uretim=nopec_uretim*nopec_uretim_artisi

#net üretim
net_uretim=opec_uretim+nopec_uretim
print("-----")
print("Petrol fiyatı:",petrol_fiyati)
print("OPEC Üretimi:",opec_uretim, " OPEC Üretim artışı:", opec_uretim_artisi)
print("Non-OPEC Üretimi:", nopec_uretim, " Non-OPEC Üretim artışı:", nopec_uretim_artisi)
print("")
print("OECD talebi: ", oecd_talep, " OECD talep artışı:", oecd_talep_artisi)
print("Non-OECD talep:", noecd_talep, " Non-OECD talep artışı:", noecd_talep_artisi)
print("")
print("Dünya talebi:",net_talep, " Dünya üretimi", net_uretim)

rezerv_kapasite=net_uretim-net_talep+rezerv_kapasite

#fiyat hareketi=((rezerv_kapasite-2.5)**2)*(10)
fiyat_hareketi=(rezerv_kapasite-2.5)*-0.2
print("Fiyat hareketi",fiyat_hareketi)

petrol_fiyati=petrol_fiyati+fiyat_hareketi
```

# Çıktı

---

Petrol fiyatı: 60

OPEC üretimi: 32 OPEC üretim artışı: 1

Non-OPEC üretimi: 68 Non-OPEC üretim artışı: 1

OECD talebi: 30.15 OECD talep artışı: 1.005

Non-OECD talep: 70.91 Non-OECD talep artışı: 1.013

Dünya talebi: 101.06 Dünya üretimi 100

Fiyat hareketi 0.2120000000000047

---

Petrol fiyatı: 60.212

OPEC üretimi: 32.32 OPEC üretim artışı: 1.01

Non-OPEC üretimi: 69.36 Non-OPEC üretim artışı: 1.02

OECD talebi: 30.2103 OECD talep artışı: 1.002

Non-OECD talep: 71.40637 Non-OECD talep artışı: 1.007

Dünya talebi: 101.61667 Dünya üretimi 101.68

Fiyat hareketi 0.1993339999999893

---

Petrol fiyatı: 60.41133400000004

OPEC üretimi: 32.9664 OPEC üretim artışı: 1.02

Non-OPEC üretimi: 72.1344 Non-OPEC üretim artışı: 1.04

OECD talebi: 30.1800897 OECD talep artışı: 0.999

Non-OECD talep: 71.47777636999999 Non-OECD talep artışı: 1.001

Dünya talebi: 101.65786606999998 Dünya üretimi 105.10079999999999

Fiyat hareketi -0.48925278600003

---

Petrol fiyatı: 59.922081214

OPEC üretimi: 33.296064 OPEC üretim artışı: 1.01

Non-OPEC üretimi: 73.577088 Non-OPEC üretim artışı: 1.02

OECD talebi: 30.2404498794 OECD talep artışı: 1.002

# Bize bir kayıt lazım

```
#version 3
import pandas as pd

kayit_yil=[]
kayit_fiyat=pd.Series()
kayit_uretim=[]
kayit_tuketim=[]

kayit=pd.DataFrame(index=kayit_yil)

: kayit['uretim']=kayit_uretim
kayit['tuketim']=kayit_tuketim
kayit['fiyat']=kayit_fiyat

: kayit
```

kayit			
	uretim	tuketim	fiyat
2019	100.000000	101.060000	62.650000
2020	102.044000	98.967698	54.959246
2021	98.586702	102.353586	64.376456
2022	101.551571	99.229430	58.571103
2023	100.168741	100.594636	59.635839
2024	99.403478	101.372705	64.558905
2025	102.496236	98.175803	53.757822
2026	98.331918	102.199914	63.427813
2027	100.766601	99.631383	60.589767
2028	101.664077	98.734021	53.264628
2029	97.255320	103.058149	67.771702
2030	102.021300	97.924280	57.529152
2031	100.035843	99.847222	57.057599
2032	97.827892	102.077450	67.681493
2033	102.574384	97.048628	53.867102
2034	98.470624	100.962181	60.095995

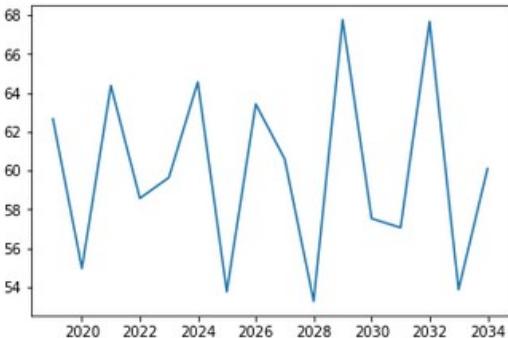
# Grafik?

```
%pylab inline
```

Populating the interactive namespace from numpy and matplotlib

```
plot(kayit.fiyat)
```

```
[<matplotlib.lines.Line2D at 0x7f44010392b0>]
```



```
plot(kayit.tuketim)
```

```
[<matplotlib.lines.Line2D at 0x7f4401011eb8>]
```



```
import numpy
import matplotlib
from matplotlib import pylab, mlab, pyplot
np = numpy
plt = pyplot

from IPython.display import display
from IPython.core.pylabtools import figsize, getfigs

from pylab import *
from numpy import *
```

# Petrol fiyat analizi

```
# %pylab inline komutu ile numpy, matplotlib dahil edilir  
%pylab inline
```

```
# veri düzenlemeleri için de Pandas'ı pd olarak alalım  
import pandas as pd
```

Populating the interactive namespace from numpy and matplotlib

## Veriye ilk bakış

Tüm örnek boyunca tek bir veri kullanacağım dan, ana veriyi indirerek **prices** adlı bir değişkene yükleyeceğim. Bunun üzerinde değişiklikleri yaparsam, değişen halini başka değişkenlere yükleyeceğim. Internet bağlantısı var ise, dosyayı **read\_excel** komutu ile indirebiliriz. Eğer kütüphaneyi bulamazsa **Terminal** açarak **xld** yani Excel dosyası okuma kütüphanesini kurabilirsiniz. Bilgisayarınıza otomatik kurar. Bunun için

- Terminal/Konsol'a çıkip xld kurmak için "**pip install xld**" yazıyoruz
- Veri kaynak noktası da **Brent\_data** olarak tanımlıyorum

```
Brent_data="https://www.eia.gov/dnav/pet/hist_xls/RBRTEd.xls"
```

**read\_excel** internetten dosyayı alarak Pandas'ın meşhur veri çerçevelerine çevirir. Fakat bir Excel dosyasının başında boş satırlar, sütünlar olabilir. Ayrıca veri birden çok alt sayfada da olabilir. Bu yüzden çalışma sayfası "**Data 1**"de 2 satır atlayarak veri çekmeye başlıyoruz

```
prices=pd.read_excel("https://www.eia.gov/dnav/pet/hist_xls/RBRTEd.xls",sheet_name="Data 1", skiprows=2)
```

# Veriye ilk bakış

- `prices.shape`
- `prices.head(10)`
- `prices.tail(5)`
- `list(prices)`

Verimizin kaç satır/sütun olduğunu `.shape` alt özelliği görebiliriz.

```
: prices.shape  
:(8048, 2)
```

Verimizin 8048 satır ve 2 sütunu var. Python da (satır,sütün) şeklinde veri tanımları vardır. Şimdi verimizin baş ve son tarafına bakalım Sirasıyla `head` ve `tail` komutları ile verinin başını ve sonunu görebiliriz. İstersek baştaki ya da sondakı kaç satır görmek istediğimizi de parantez içinde ekleyebiliriz.

```
: #ilk 4 satır  
prices.head(4)
```

	Date	Europe Brent Spot Price FOB (Dollars per Barrel)
0	1987-05-20	18.63
1	1987-05-21	18.45
2	1987-05-22	18.55
3	1987-05-25	18.60

```
: # son 3 satır  
prices.tail(3)
```

	Date	Europe Brent Spot Price FOB (Dollars per Barrel)
8045	2019-01-31	62.46
8046	2019-02-01	61.86
8047	2019-02-04	62.26

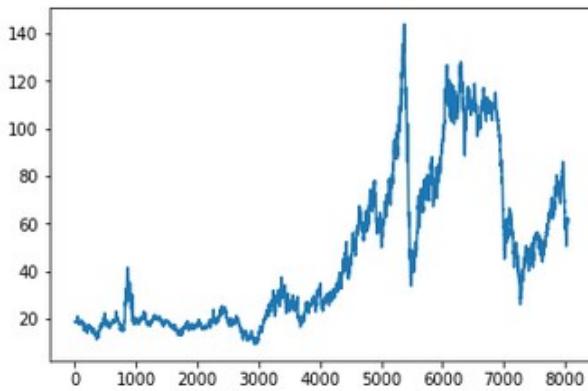
And let's see the column titles with list command

```
: list(prices)
```

# Grafikleme

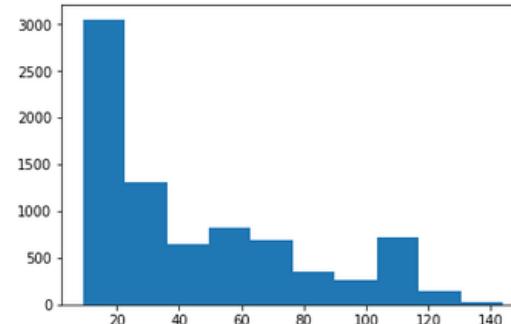
```
plot(prices['Europe Brent Spot Price FOB (Dollars per Barrel)'])
```

```
[<matplotlib.lines.Line2D at 0x7f37846d86d8>]
```

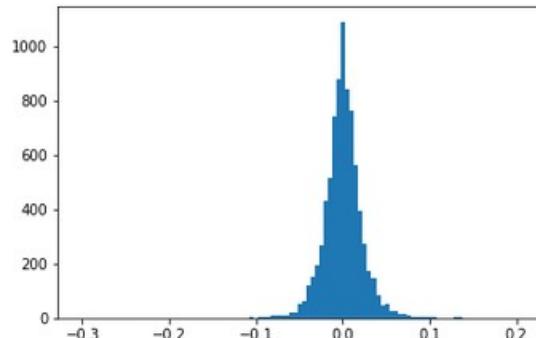


```
hist(prices['Europe Brent Spot Price FOB (Dollars per Barrel)'])
```

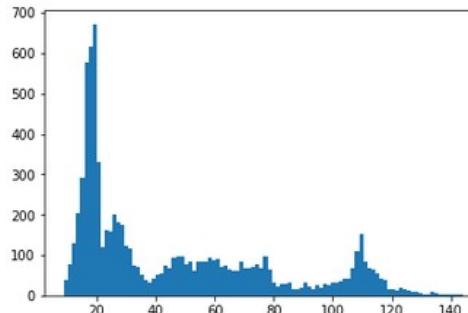
```
(array([3052., 1314., 644., 818., 696., 353., 268., 726., 149.
       28.]),
 array([ 9.1 , 22.585, 36.07 , 49.555, 63.04 , 76.525, 90.01
        103.495, 116.98 , 130.465, 143.95 ]),
 <a list of 10 Patch objects>)
```



```
hist(prices['Europe Brent Spot Price FOB (Dollars per Barrel)'].pct_change(), bins=100);
```

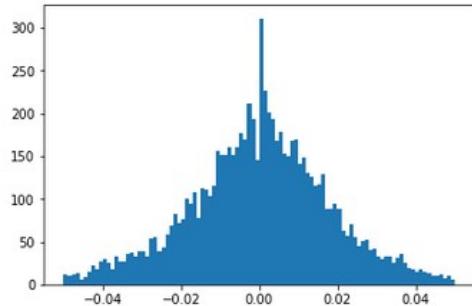


```
hist(prices['Europe Brent Spot Price FOB (Dollars per Barrel)'], bins=100);
```



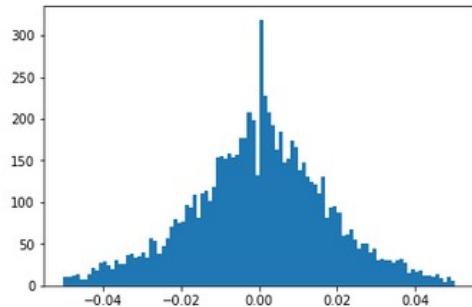
# Filtreleme

```
hist(pc[(pc<0.05) & (pc>-0.05)], bins=100);
```



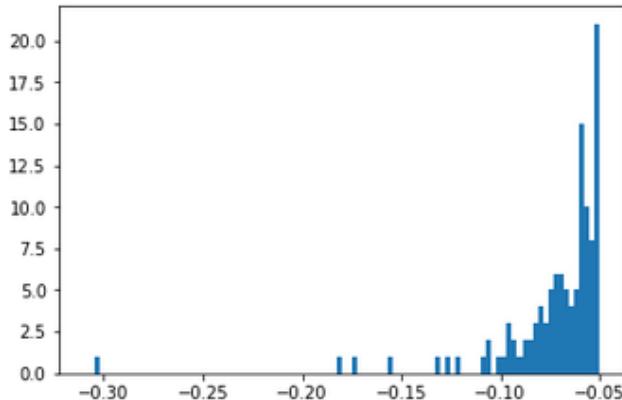
Tüm bunun daha kolay bir yöntemi var, tüm veri setini alıp, **hist** komutuna **range=(-0.05, 0.05)** parametresi girerek histogramın %5 ve %-5 arasındaki grafiğini görebiliriz

```
hist(pc, bins=100, range=(-0.05,0.05));
```



# Uç değer görmek

```
hist(pc, bins=100, range=(-0.31,-0.05));
```



Ama en düşük değer tam olarak kaç? Bunun için **min** komutunu kullanıyoruz

```
pc.min()
```

```
-0.30317040951122853
```

Bir günde en büyük fiyat düşüşü **-30%**. Peki bu hangi gün oldu? Önce kaçinci satırda minimum değerin gerçekleştiğini görelim, komutumuz **idxmax()**

```
pc.idxmin()
```

```
936
```

# .iloc (integer location)

```
prices.iloc[[935,936,937]]
```

	Date	Europe Brent Spot Price FOB (Dollars per Barrel)
935	1991-01-16	30.28
936	1991-01-17	21.10
937	1991-01-18	19.10

```
prices.iloc[range(935,938)]
```

	Date	Europe Brent Spot Price FOB (Dollars per Barrel)
935	1991-01-16	30.28
936	1991-01-17	21.10
937	1991-01-18	19.10

Peki neden **pc** den istemedik? Görelim, şimdi pc ile o satırlara bakalım.

```
pc.iloc[[935,936,937]]
```

```
935    0.035214
936   -0.303170
937   -0.094787
Name: Europe Brent Spot Price FOB (Dollars per Barrel), dtype: float64
```

# Haftalık hareketler

```
# Hafta isimli yeni bir sütun oluştur ve veriseti endeksinde tarihin hafta değerini al  
prices["Hafta"] = prices["Date"].dt.week
```

Veri setimizin son 3 satırını görelim

```
prices.tail(3)
```

	Date	Europe Brent Spot Price FOB (Dollars per Barrel)	Hafta
8045	2019-01-31	62.46	5
8046	2019-02-01	61.86	5
8047	2019-02-04	62.26	6

Eğer problem yaşamadıysak ay, yıl ve haftanın günü(HG) değişkenlerini de ekleyebilir. Değişken isimlerinde Türkçe karakterler sorun olduğundan ben ingilizce alfabe ile değişkenleri tanımlamayı tercih ettim.

```
prices["Ay"] = prices["Date"].dt.month  
prices["Yıl"] = prices["Date"].dt.year  
prices["HG"] = prices["Date"].dt.dayofweek
```

Veri setimizi genişlettik. Haftanın günü (HG) verisi Pazartesi için 0, Cuma için 4 değerini verir. Petrol piyasaları haftasonu açık olmadığından sadece haftaiçi değişkenleri var.

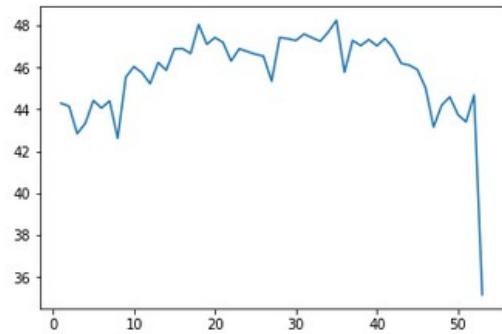
```
#let's see last 4 lines  
prices.tail(4)
```

	Date	Europe Brent Spot Price FOB (Dollars per Barrel)	Hafta	Ay	Yıl	HG
8044	2019-01-30	61.89	5	1	2019	2
8045	2019-01-31	62.46	5	1	2019	3
8046	2019-02-01	61.86	5	2	2019	4
8047	2019-02-04	62.26	6	2	2019	0

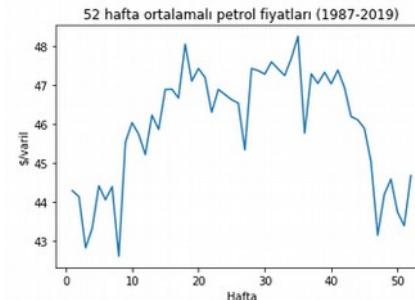
# Gruplama

```
plot(prices.groupby("Hafta")['Europe Brent Spot Price FOB (Dollars per Barrel)'].mean())
```

```
[<matplotlib.lines.Line2D at 0x7f37842bd748>]
```



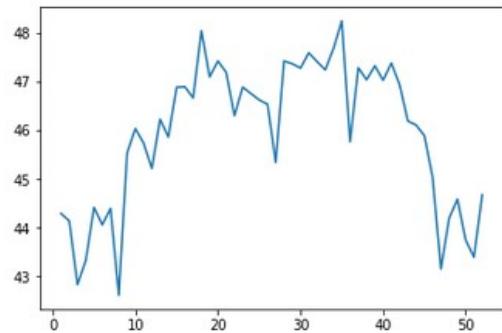
```
plot(prices.groupby("Hafta")['Europe Brent Spot Price FOB (Dollars per Barrel)'].mean().head(52))  
title("52 hafta ortalamalı petrol fiyatları (1987-2019)")  
xlabel("Hafta")  
ylabel("$/varil");
```



Maalesef bazı yıllarda 53 hafta olabiliyor, bu yüzden ilk 52 haftayı almak için **head(52)** komutunu kullanabiliriz.

```
plot(prices.groupby("Hafta")['Europe Brent Spot Price FOB (Dollars per Barrel)'].mean().head(52))
```

```
[<matplotlib.lines.Line2D at 0x7f37843f5898>]
```



# Pivot

```
pt=prices.pivot_table( values='Europe Brent Spot Price FOB (Dollars per Barrel)', index=['Hafta'],
                      columns=['Yil'], aggfunc=np.mean)
```

Pivot tablomuzu oluşturarak yüklediğimi pt değişkenine bakalım(pt pivot tablo kısaltması)

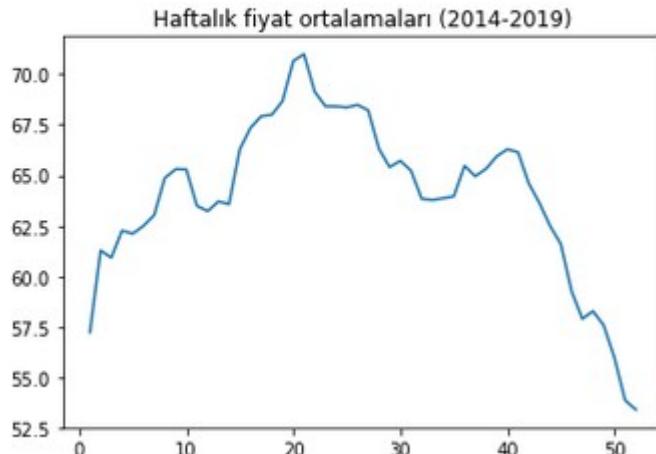
```
pt.head(4)
```

Yil	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	...	2010	2011	2012	2013	2014	2015	2016	2017
Hafta																			
1	NaN	17.4525	16.590	23.566	22.332	18.475	17.510	14.1775	16.0725	20.763333	...	79.8180	94.722	112.168	111.8020	76.648	55.3800	34.194	55.1275
2	NaN	16.4360	16.980	21.674	25.328	17.772	17.052	14.0000	16.2320	18.528000	...	78.3100	97.086	111.776	112.3720	107.014	49.4660	29.100	54.0160
3	NaN	16.9080	17.896	21.052	25.856	18.370	16.978	14.2160	16.8840	17.614000	...	74.2825	97.340	109.600	111.3460	107.828	46.5780	27.756	54.1940
4	NaN	16.2400	17.328	20.494	20.766	18.112	18.004	14.6420	16.8480	17.188000	...	71.8820	96.620	109.180	113.7175	109.140	46.4425	31.748	54.9060

4 rows × 33 columns

# Filtreleme

```
plot(pt.loc[:,range(2014,2020)].mean(axis=1).head(52))
title ("Haftalık fiyat ortalamaları (2014-2019)");
```



# Elektrik verileri

```
%pylab inline  
import pandas as pd
```

Populating the interactive namespace from numpy and matplotlib

```
prices=pd.read_excel("2018-elektrik.xlsx",sheet_name="2018-saatlik")
```

```
list(prices)
```

```
['gun',  
'ay',  
'yil',  
'saat',  
'haftanin_gunu',  
'lisansli_toplam',  
'dogalgaz',  
'barajli',  
..]
```

```
x=prices.columns.values  
syc=0  
for ind in x:  
    print(ind,syc)  
    syc+=1
```

```
gun 0  
ay 1  
yil 2  
saat 3  
haftanin_gunu 4  
lisansli_toplam 5  
dogalgaz 6  
barajli 7  
linyit 8  
akarsu 9  
ithalkomur 10  
ruzgar 11  
lisansli_gunes 12  
fueloil 13  
jeotermal 14  
ASFALTIT 15  
taskomuru 16  
biyokutle 17  
nafta 18  
lng 19  
uluslararası 20  
lisanssiz_toplam 21  
l_ruzgar 22  
l_biyogaz 23  
l_kanaltipi 24  
l_biyokutle 25  
l_gunes 26  
l_diger 27  
genel_toplam 28  
ptf 29  
smf 30
```

# Korelasyon ve regresyon

```
a=prices.corr()
```

```
gunes=prices[['lisansli_gunes','l_gunes']]
prices[['lisansli_gunes','l_gunes']].corr()
```

	lisansli_gunes	l_gunes
lisansli_gunes	1.000000	0.765178
l_gunes	0.765178	1.000000

```
from statsmodels.formula.api import ols
ols("l_gunes ~ lisansli_gunes-1", data=gunes).fit().summary()
```

OLS Regression Results

Dep. Variable:	l_gunes	R-squared:	0.696
Model:	OLS	Adj. R-squared:	0.696
Method:	Least Squares	F-statistic:	2.008e+04
Date:	Fri, 22 Mar 2019	Prob (F-statistic):	0.00
Time:	22:16:30	Log-Likelihood:	-71316.
No. Observations:	8760	AIC:	1.426e+05
Df Residuals:	8759	BIC:	1.426e+05
Df Model:	1		
Covariance Type:	nonrobust		

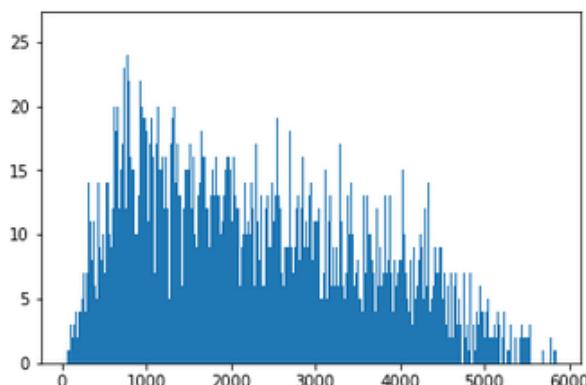
# Pivot table

```
: pt=prices.pivot_table( values='ruzgar', index=['saat'], columns=['haftanin_gunu'], aggfunc=np.mean)
```

```
: pt.head(3)
```

	haftanin_gunu	1	2	3	4	5	6	7
	saat							
0	2122.200189	2259.935192	2252.335000	2312.174231	2251.690962	2300.505385	2430.512500	
1	2025.921509	2182.630385	2211.606538	2238.489231	2202.946923	2249.016538	2331.129423	
2	1957.157736	2160.858077	2213.280385	2224.492885	2166.859038	2222.343654	2290.703077	

```
: hist(prices.ruzgar,bins=1000);
```



Teşekkürler

[barissanli.com](http://barissanli.com)