,

2019-09-16

Contents

Chapter 1

1.1

R Python,
Stata.

R > R - ,
. R RStudio.

RStudio Mac OS Windows.

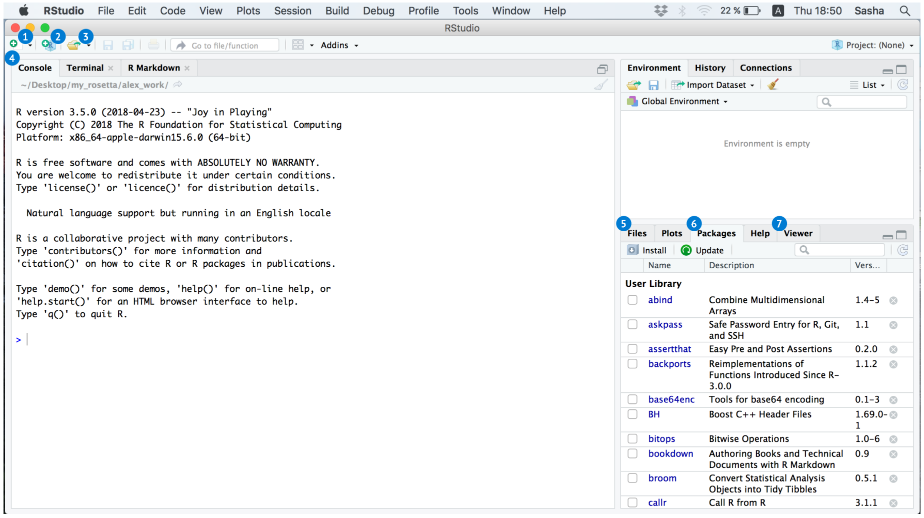
RStudio Windows / Mac OS:

1. R .

- Windows: “Download R for Windows” “base” “Download R 3.x.x for Windows”.

- Mac OS: “Download R for (Mac) OS X” “Latest Release” “R 3.x.x”.

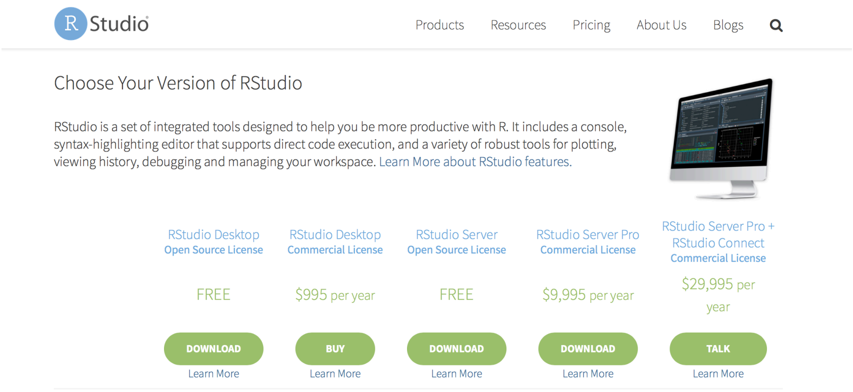
2. RStudio (
). .



```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

	Name	Description	Vers...
User Library			
□	abind	Combine Multidimensional Arrays	1.4–5
□	askpass	Safe Password Entry for R, Git, and SSH	1.1
□	assertthat	Easy Pre and Post Assertions	0.2.0
□	backports	Reimplementations of Functions Introduced Since R–3.0.0	1.1.2
□	base64enc	Tools for base64 encoding	0.1–3
□	BH	Boost C++ Header Files	1.69.0–1
□	bitops	Bitwise Operations	1.0–6
□	bookdown	Authoring Books and Technical Documents with R Markdown	0.9
□	broom	Convert Statistical Analysis Objects into Tidy Tibbles	0.5.1
□	callr	Call R from R	3.1.1

Figure 1.1:



Choose Your Version of RStudio

RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your workspace. Learn More about RStudio features.

RStudio Desktop Open Source License	RStudio Desktop Commercial License	RStudio Server Open Source License	RStudio Server Pro Commercial License	RStudio Server Pro + RStudio Connect Commercial License
FREE	$995 per year	FREE	$9,995 per year	$29,995 per year
Learn More	Learn More	Learn More	Learn More	Learn More

, RStudio .

#####

1. **New file -** .
2. **New project -** .
3. **Open file -** .
4. **Console -** , .
5. **Files -** , .
6. **Packages -** , . . . , *installed.packages()*.

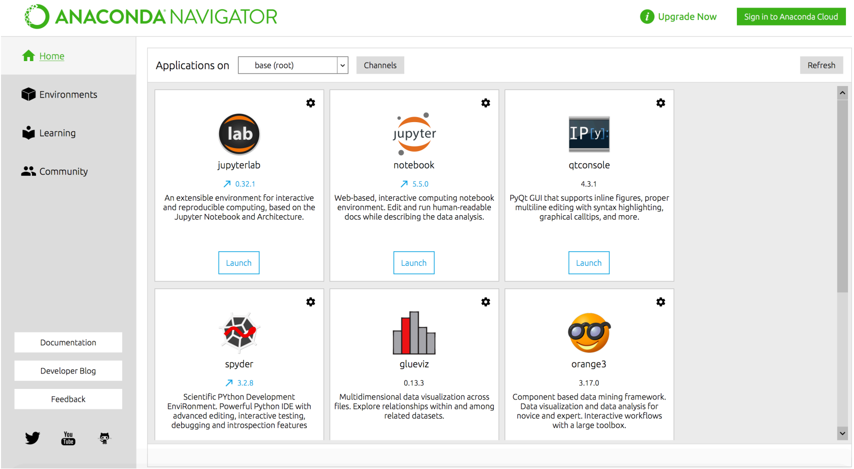
7. **Viewer** - .

Python > Python - , . Python Jupyter Notebook.

#####

1. Anaconda .

2. Anaconda Navigator,



An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

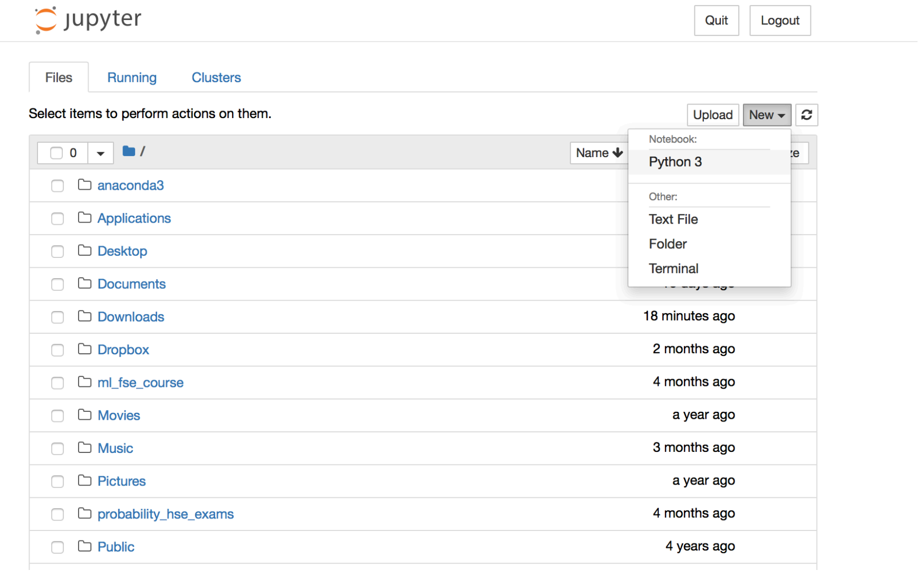
Multidimensional data visualization across files. Explore relationships within and among related datasets.

Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

Jupyter Notebook. Navigator.bb

#####

Jupyter Notebook, , . , “New” “Notebook: Python 3”. File in Jupyter.bb

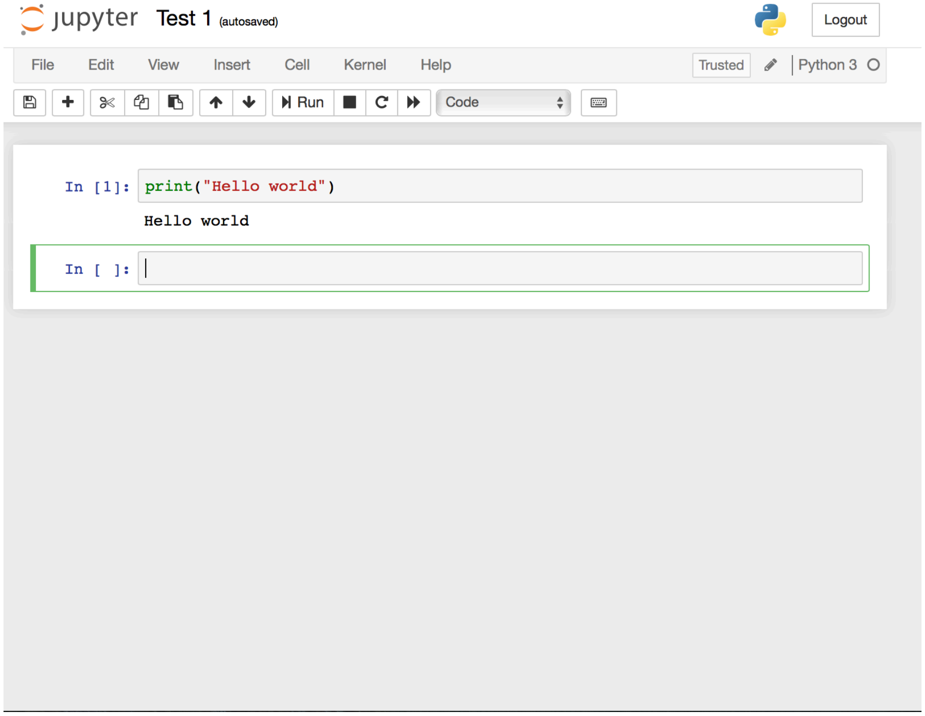


Select items to perform actions on them.

```
anaconda3
Applications
Desktop
Documents
Downloads                18 minutes ago
Dropbox                  2 months ago
ml_fse_course            4 months ago
Movies                   a year ago
Music                    3 months ago
Pictures                 a year ago
probability_hse_exams    4 months ago
Public                   4 years ago
```

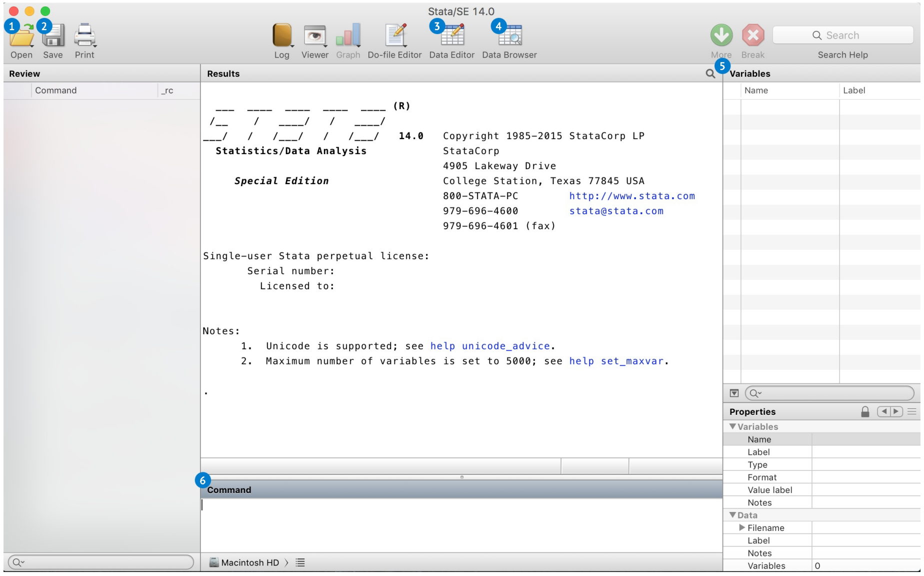
, , . .

, “Shift” + “Enter”, . in Jupyter.bb



```
In [1]: print("Hello world")

        Hello world

In [ ]:
```

Interface.bb



```
  ___  ____  ____  ____  ____ (R)
 /__    /   ____/   /   ____/
___/   /   /___/   /   /___/   14.0   Copyright 1985-2015 StataCorp LP
  Statistics/Data Analysis            StataCorp
                                      4905 Lakeway Drive
     Special Edition                  College Station, Texas 77845 USA
                                      800-STATA-PC        http://www.stata.com
                                      979-696-4600        stata@stata.com
                                      979-696-4601 (fax)

Single-user Stata perpetual license:
       Serial number:
         Licensed to:


Notes:
      1.  Unicode is supported; see help unicode_advice.
      2.  Maximum number of variables is set to 5000; see help set_maxvar.

.
```

Figure 1.2: *Stata*

STATA > Stata, R Python, ,
. .

:

Stata - . Stata SE,
Stata MP.

:

1. **Open File -** .
2. **Save -** .
3. **Data Editor -** .
4. **Data Browser -** .
5. **Variables -** .
6. **Command -** , .

Chapter 2

R .

.

```r
library(tidyverse) #
library(skimr) #         summary
library(rio) #       .dta
library(car) #
library(tseries) #
library(sjPlot) #
```

.

```r
df = import("us-return.dta")
```

.

```r
# skim_with(numeric = list(hist = NULL, p25 = NULL, p75 = NULL)) #
skim(df) #
```

```
Skim summary statistics
 n obs: 2664
 n variables: 22

-- Variable type:character ---------------------------------------------------------------------
 variable missing complete    n min max empty n_unique
        B       0     2664 2664   0   6  2544       31

-- Variable type:numeric -----------------------------------------------------------------------
 variable missing complete    n    mean      sd      p0     p25     p50
        A    2544      120 2664 60.5    34.79    1      30.75   60.5
    BOISE    2544      120 2664  0.017   0.097  -0.27   -0.045   0.015
```

```
   CITCRP     2544       120 2664  0.012    0.081  -0.28    -0.037    0.011
    CONED     2544       120 2664  0.019    0.05   -0.14    -0.012    0.019
   CONTIL     2544       120 2664 -0.0011   0.15   -0.6     -0.051    0
   DATGEN     2544       120 2664  0.0075   0.13   -0.34    -0.072    0.017
      DEC     2544       120 2664  0.02     0.099  -0.36    -0.051    0.024
    DELTA     2544       120 2664  0.012    0.096  -0.26    -0.053    0.013
   GENMIL     2544       120 2664  0.017    0.065  -0.15    -0.026    0.011
   GERBER     2544       120 2664  0.016    0.088  -0.29    -0.036    0.015
      IBM     2544       120 2664  0.0096   0.059  -0.19    -0.029    0.002
   MARKET     2544       120 2664  0.014    0.068  -0.26    -0.013    0.012
    MOBIL     2544       120 2664  0.016    0.08   -0.18    -0.032    0.013
    MOTOR     2544       120 2664  0.018    0.097  -0.33    -0.053    0.017
    PANAM     2544       120 2664  0.0035   0.13   -0.31    -0.065    0
     PSNH     2544       120 2664 -0.0042   0.11   -0.48    -0.049    0
   rkfree     2544       120 2664  0.0068   0.0022  0.0021   0.0052   0.0066
   RKFREE     2544       120 2664  0.0068   0.0022  0.0021   0.0052   0.0066
    TANDY     2544       120 2664  0.025    0.13   -0.25    -0.058    0.022
   TEXACO     2544       120 2664  0.012    0.08   -0.19    -0.037    0.01
    WEYER     2544       120 2664  0.0096   0.085  -0.27    -0.049   -0.002
     p75    p100     hist
 90.25   120
  0.07     0.38
  0.064    0.32
  0.045    0.15
  0.058    0.97
  0.078    0.53
  0.075    0.39
  0.063    0.29
  0.06     0.19
  0.065    0.23
  0.05     0.15
  0.062    0.15
  0.057    0.37
  0.084    0.27
  0.074    0.41
  0.043    0.32
  0.0078   0.013
  0.0078   0.013
  0.094    0.45
  0.048    0.4
  0.06     0.27
```

```
df = rename(df, n = A, date = B) #
```

```
df = na.omit(df) #
```

CAPM :) MOTOR. , -

MOTOR , - .

```r
#
df = mutate(df, y = MOTOR - RKFREE, x = MARKET - RKFREE)
```

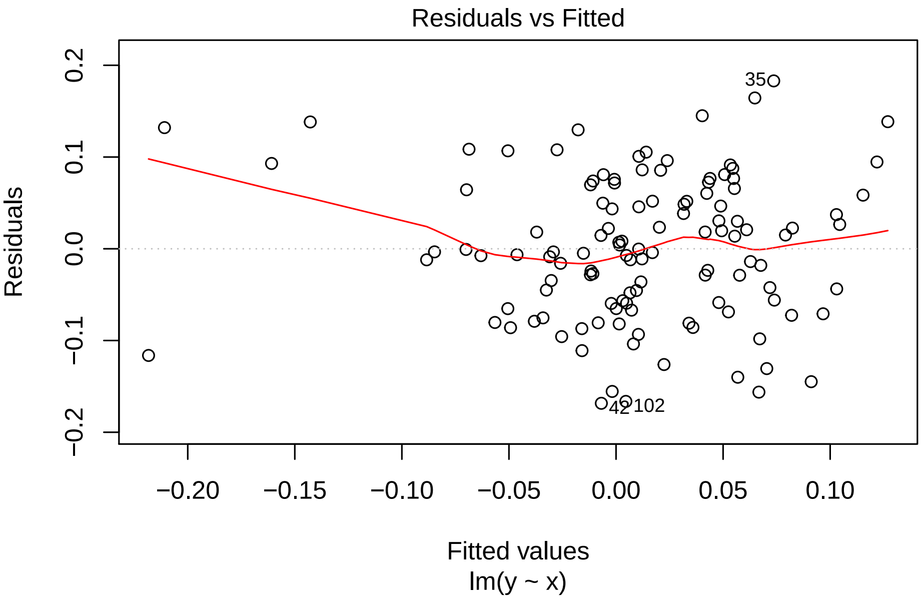
.

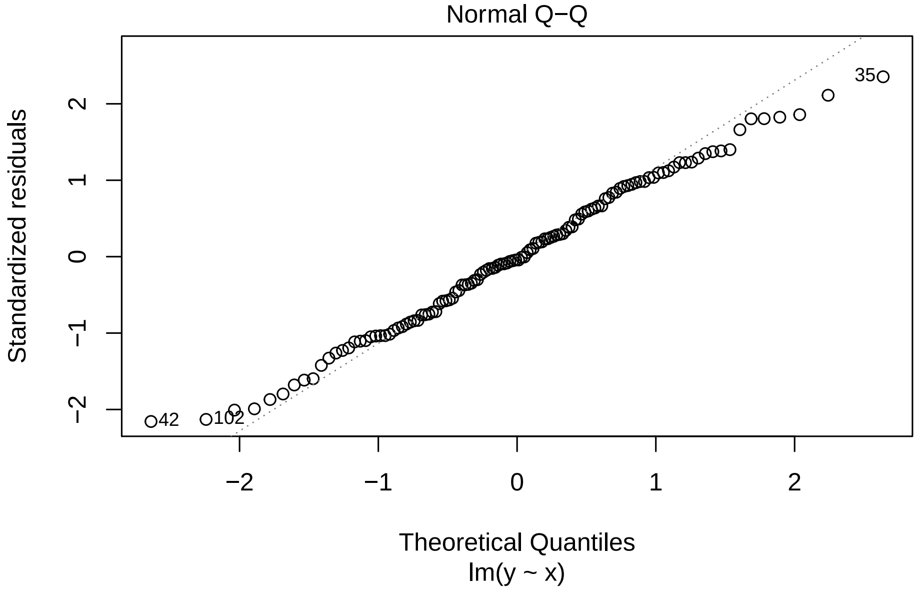
```r
ols = lm(y ~ x, data = df)
summary(ols)
```

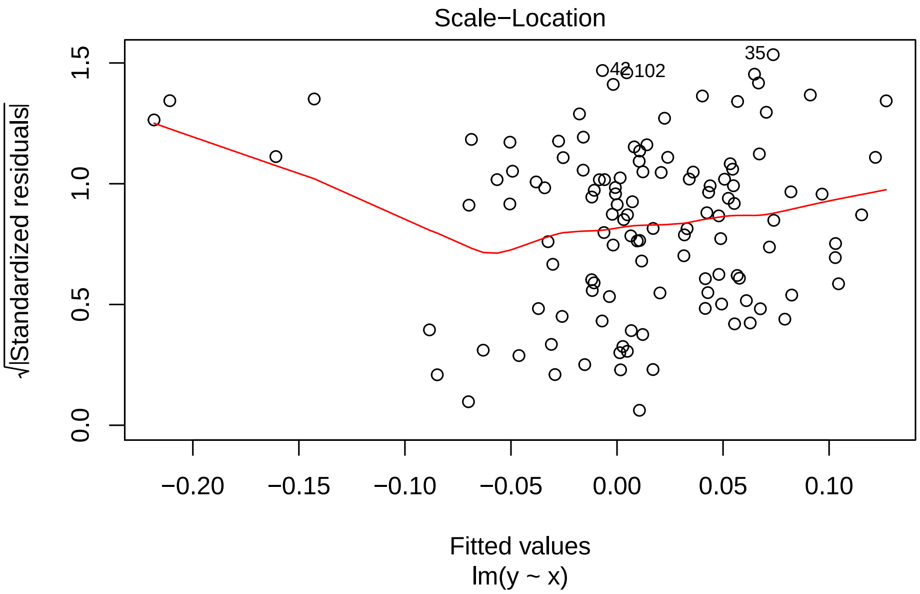
```
Call:
lm(formula = y ~ x, data = df)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.168421 -0.059381 -0.003399  0.061373  0.182991 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 0.005253   0.007200   0.730    0.467    
x           0.848150   0.104814   8.092 5.91e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.07844 on 118 degrees of freedom
Multiple R-squared:  0.3569,	Adjusted R-squared:  0.3514 
F-statistic: 65.48 on 1 and 118 DF,  p-value: 5.913e-13
```

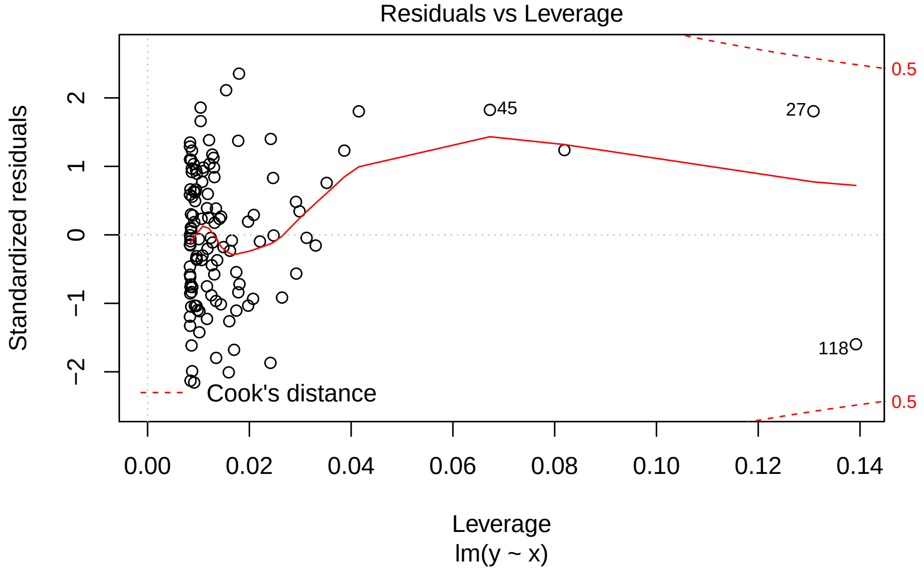
. ,
, .

```r
plot(ols)
```


Residuals vs Fitted
Residuals
0.2
0.1
0.0
−0.1
−0.2
35
42
102
−0.20
−0.15
−0.10
−0.05
0.00
0.05
0.10
Fitted values
lm(y ~ x)


Normal Q−Q
Standardized residuals
2
1
0
−1
−2
35
42
102
−2
−1
0
1
2
Theoretical Quantiles
lm(y ~ x)





.

```
est = cbind(Estimate = coef(ols), confint(ols))
```

.

```
linearHypothesis(ols, c("x = 1"))
```

```
Linear hypothesis test
```

```
Hypothesis:
x = 1

Model 1: restricted model
Model 2: y ~ x

  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1    119 0.73900
2    118 0.72608  1  0.012915 2.0989 0.1501
```

:) - .

$$H_0 : S = 0, K = 3,$$

S — (Skewness), K — (Kurtosis)

```
jarque.bera.test(resid(ols))
```

```

    Jarque Bera Test

data:  resid(ols)
X-squared = 1.7803, df = 2, p-value = 0.4106
```

- .

$H_0 : \epsilon_i \sim N(\mu, \sigma^2)$

```
shapiro.test(resid(ols))
```

```

    Shapiro-Wilk normality test

data:  resid(ols)
W = 0.99021, p-value = 0.5531
```

.

.

```
set.seed(7)
newData = df
newData = mutate(newData, x = x + rnorm(n = n())) #
yhat = predict(ols, newdata = newData, se = TRUE)
```

2.0.0.1

.

```
use us-return.dta
```

```
end of do-file
```

.

```
summarize
ren A n
ren B date
```

```
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
           A |        120        60.5    34.78505          1        120
           B |          0
       MOBIL |        120    .0161917    .0803075      -.178       .366
      TEXACO |        120    .0119417    .0797036      -.194       .399
         IBM |        120    .0096167     .059024      -.187        .15
-------------+---------------------------------------------------------
         DEC |        120      .01975    .0991438      -.364       .385
      DATGEN |        120    .0074833    .1275399      -.342       .528
       CONED |        120    .0185083    .0502719      -.139       .151
        PSNH |        120   -.0042167    .1094712      -.485       .318
       WEYER |        120    .0096333    .0850664      -.271        .27
-------------+---------------------------------------------------------
       BOISE |        120     .016675    .0974882      -.274       .379
       MOTOR |        120    .0181583    .0972656      -.331        .27
       TANDY |        120    .0250083     .127566      -.246       .454
       PANAM |        120    .0035167    .1318054      -.313       .406
       DELTA |        120    .0116917    .0959317       -.26       .289
-------------+---------------------------------------------------------
      CONTIL |        120      -.0011    .1506992        -.6       .974
      CITCRP |        120    .0118583    .0809719      -.282       .318
      GERBER |        120       .0164    .0877379      -.288       .234
      GENMIL |        120    .0165833    .0650403      -.148        .19
      MARKET |        120    .0139917    .0683532       -.26       .148
-------------+---------------------------------------------------------
      RKFREE |        120    .0068386    .0021869     .00207     .01255
      rkfree |        120    .0068386    .0021869     .00207     .01255
```

.

```
drop if n == .
gen y = MOTOR - RKFREE
gen x = MARKET - RKFREE
```

```
(2,544 observations deleted)
```

.   .

```
reg y x
```

```
      Source |       SS           df       MS      Number of obs   =       120
```

```
-------------+----------------------------------   F(1, 118)       =     65.48
       Model |  .402913404         1  .402913404   Prob > F        =    0.0000
    Residual |  .726081541       118  .006153233   R-squared       =    0.3569
-------------+----------------------------------   Adj R-squared   =    0.3514
       Total |  1.12899494       119  .009487352   Root MSE        =    .07844

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   .8481496   .1048138     8.09   0.000     .6405898    1.055709
       _cons |   .0052529   .0071999     0.73   0.467     -.009005    .0195107
------------------------------------------------------------------------------
```

.

```
test x = 1

 ( 1)  x = 1

       F(  1,   118) =    2.10
            Prob > F =    0.1501
```

.

```
predict u_hat, resid
predict y_hat

(option xb assumed; fitted values)
```

- . , -
. . Stata .
https://www.stata.com/manuals13/rsktest.pdf

```
sktest u_hat

                    Skewness/Kurtosis tests for Normality
                                                          ------ joint ------
    Variable |        Obs  Pr(Skewness)  Pr(Kurtosis) adj chi2(2)   Prob>chi2
-------------+---------------------------------------------------------------
       u_hat |        120     0.8841        0.1027        2.74         0.2539
```

- . R.

```
swilk u_hat

                   Shapiro-Wilk W test for normal data

    Variable |        Obs       W           V         z       Prob>z
-------------+------------------------------------------------------
       u_hat |        120    0.99021      0.942    -0.133    0.55310
```

.

QQ -

```
qnorm u_hat
```

.

~~~
```stata
rvfplot, yline(0)
```
~~~

- (R).

```
lvr2plot
```

“““

.

.

```
predict D, cooksd
predict standard, rstandard

graph twoway scatter standard y_hat [aweight=D], msymbol(oh) yline(0)
```

```
set seed 7

set obs 120
gen x_new = x+ 0.5 *rnormal()
gen y_hat_new =  .8481496 * x_new+ .0052529
```

```
number of observations (_N) was 120, now 120
```

2.0.0.2 python

Statsmodels.

```
import pandas as pd #
import numpy as np #      ,
import matplotlib.pyplot as plt #
import statsmodels.api as sm
import statsmodels.formula.api as smf
import statsmodels.graphics.gofplots as gf
from statsmodels.stats.outliers_influence import summary_table
import seaborn as sns #
from scipy.stats import shapiro #
import statsmodels.discrete.discrete_model
```

, :)

```
plt.style.use('seaborn')
plt.rc('font', size=14)
plt.rc('figure', titlesize=15)
plt.rc('axes', labelsize=15)
plt.rc('axes', titlesize=15)
```

.

```
df = pd.read_stata('us-return.dta')
```

.

```
df.dropna(inplace=True) ##      (                              )
df.reset_index(drop=True, inplace=True)
```

.

```
df = df.rename(columns={'A':'n', 'B': 'date'})
```

```
df['y'] = df['MOTOR'] - df['RKFREE']
df['x'] = df['MARKET'] - df['RKFREE']
```

:)

```
regr = smf.ols('y~x', data = df).fit()
regr.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.357
Model:                            OLS   Adj. R-squared:                  0.351
Method:                 Least Squares   F-statistic:                     65.48
Date:                   , 16    2019   Prob (F-statistic):           5.91e-13
Time:                        10:46:25   Log-Likelihood:                 136.18
No. Observations:                 120   AIC:                            -268.4
Df Residuals:                     118   BIC:                            -262.8
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0053      0.007      0.730      0.467      -0.009       0.020
x              0.8481      0.105      8.092      0.000       0.641       1.056
==============================================================================
Omnibus:                        2.684   Durbin-Watson:                   2.030
Prob(Omnibus):                  0.261   Jarque-Bera (JB):                1.780
```

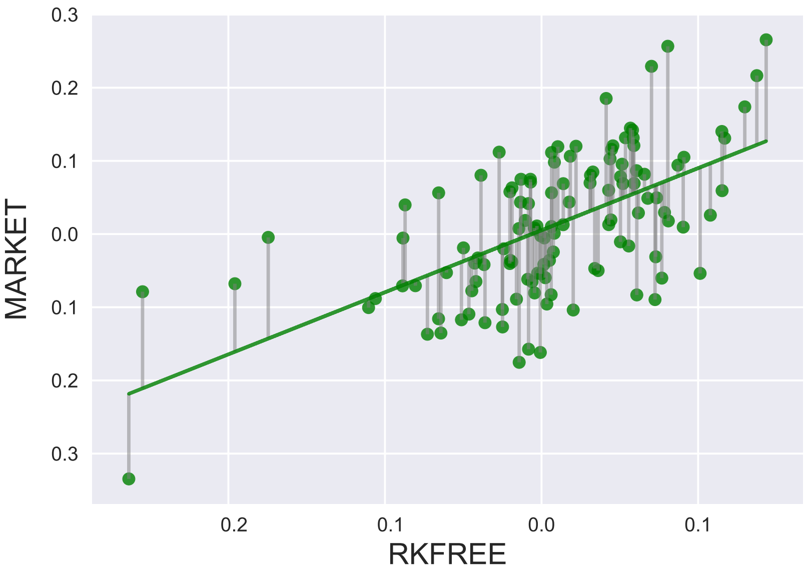
```
Skew:                          -0.031   Prob(JB):                        0.411
Kurtosis:                       2.406   Cond. No.                         14.6
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""
```

.

```
df['yhat'] = regr.fittedvalues
```

, , R, .
:)

```
fig, ax = plt.subplots()
ax.plot(df['x'],regr.fittedvalues, color='g', alpha =0.8)
ax.scatter(df['x'],regr.fittedvalues+regr.resid, color = 'g', alpha = 0.8, s = 40)
ax.vlines(df['x'],regr.fittedvalues,regr.fittedvalues+regr.resid, color = 'gray', alpha = 0.5)
plt.title('                    ')
plt.xlabel('RKFREE')
plt.ylabel('MARKET')
plt.show()
```



.

```
regr.conf_int()
```

```
                  0         1
Intercept -0.009005  0.019511
x          0.640590  1.055709
```

F-test.

```
hypotheses = '(x = 1)'
regr.f_test(r_matrix = hypotheses)
```

```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[2.09891771]]), p=0.1500556415866233, df_denom=118, df_num=1>
```

. , R. .

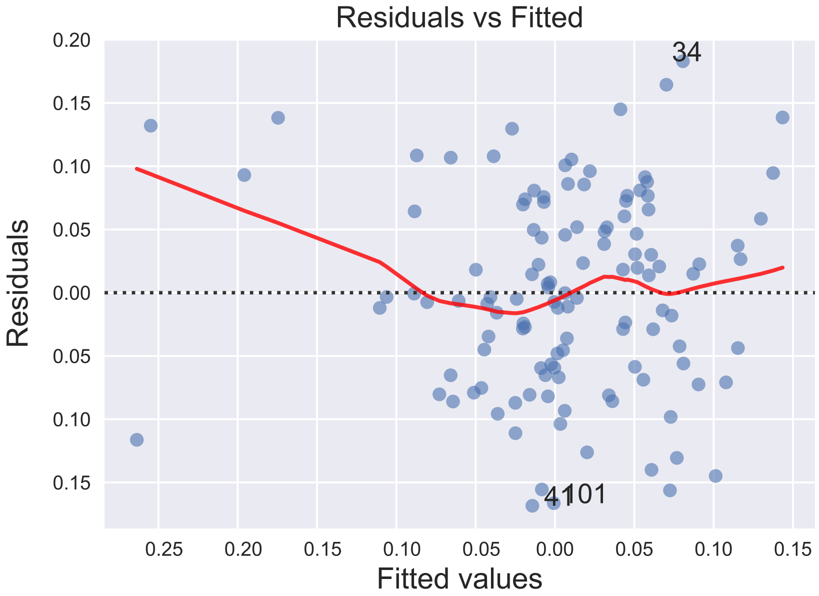
```
W, p_value = shapiro(regr.resid)
#pd.DataFrame(data = {'W': [round(W,3)], 'p_value': [round(p_value,3)]})
```

.

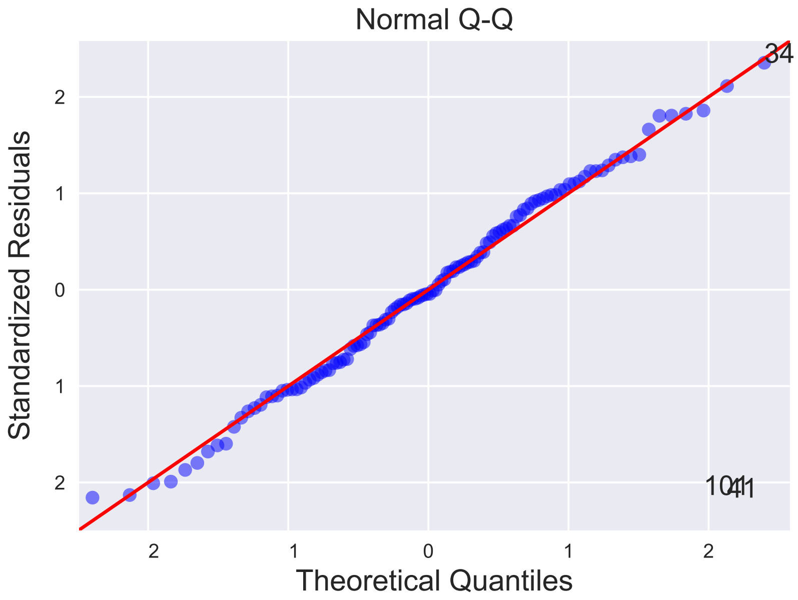
```
import random
random.seed(7)

newData = df['x'] + 0.5*np.random.normal(len(df))
prediction = regr.predict(newData)
```

! , autoplot R.

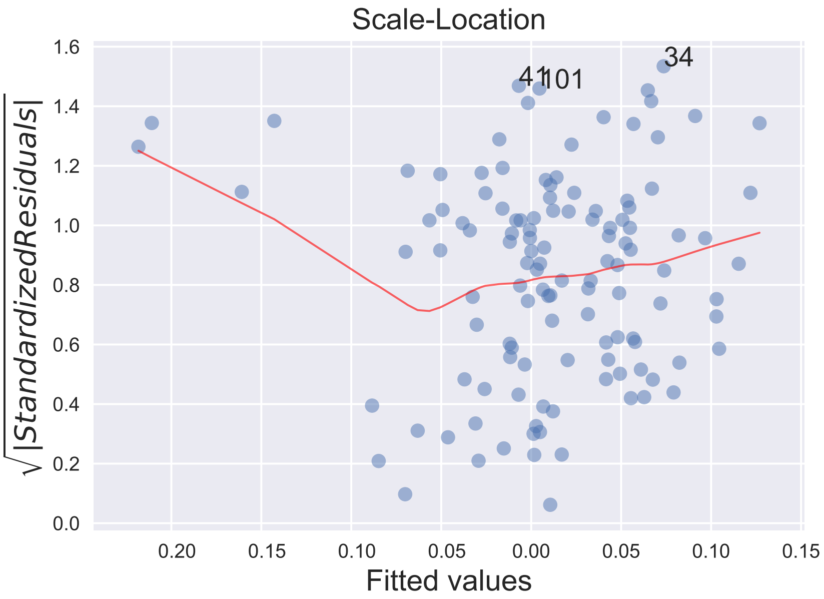
```
fig_1 = plt.figure(1)

fig_1.axes[0] = sns.residplot(df['x'], df['y'],
                                  lowess=True,
                                  scatter_kws={'alpha': 0.6},
                                  line_kws={'color': 'red', 'lw': 2, 'alpha': 0.8})

fig_1.axes[0].set_title('Residuals vs Fitted')
fig_1.axes[0].set_xlabel('Fitted values')
fig_1.axes[0].set_ylabel('Residuals')


#
abs_resid = abs(regr.resid).sort_values(ascending=False)
abs_resid_top3 = abs_resid[:3]

for i in abs_resid_top3.index:
    fig_1.axes[0].annotate(i,
                               xy=(regr.fittedvalues[i],
                                   regr.resid[i]))
```



```
norm_residuals = regr.get_influence().resid_studentized_internal #


QQ = gf.ProbPlot(norm_residuals)
fig_2 = QQ.qqplot(line='45', alpha=0.5, color='b', lw=1)


fig_2.axes[0].set_title('Normal Q-Q')
fig_2.axes[0].set_xlabel('Theoretical Quantiles')
fig_2.axes[0].set_ylabel('Standardized Residuals');

#
abs_norm_resid = np.flip(np.argsort(abs(norm_residuals)), 0)
abs_norm_resid_top3 = abs_norm_resid[:3]

for r, i in enumerate(abs_norm_resid_top3):
    fig_2.axes[0].annotate(i,
                               xy=(np.flip(QQ.theoretical_quantiles, 0)[r],
                                   norm_residuals[i]))
```



```
fig_3 = plt.figure(3)

plt.scatter(regr.fittedvalues, np.sqrt(abs(norm_residuals)), alpha=0.5)
sns.regplot(regr.fittedvalues, np.sqrt(abs(norm_residuals)),
            scatter=False,
            ci=False,
            lowess=True,
            line_kws={'color': 'red', 'lw': 1, 'alpha': 0.6})

fig_3.axes[0].set_title('Scale-Location')
fig_3.axes[0].set_xlabel('Fitted values')
fig_3.axes[0].set_ylabel('$\sqrt{|Standardized Residuals|}$')

#       !)
abs_sq_norm_resid = np.flip(np.argsort(np.sqrt(abs(norm_residuals)), 0))
abs_sq_norm_resid_top3 = abs_sq_norm_resid[:3]

for i in abs_sq_norm_resid_top3:
    fig_3.axes[0].annotate(i, xy=(regr.fittedvalues[i],
                                   np.sqrt(abs(norm_residuals)[i])))
```



```
leverage = regr.get_influence().hat_matrix_diag #                        -
cook_dist = regr.get_influence().cooks_distance[0]  #

fig_4 = plt.figure(4)

plt.scatter(leverage, norm_residuals, alpha=0.5)
sns.regplot(leverage, norm_residuals,
            scatter=False,
            ci=False,
            lowess=True,
            line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})

fig_4.axes[0].set_xlim(0, 0.20)
```

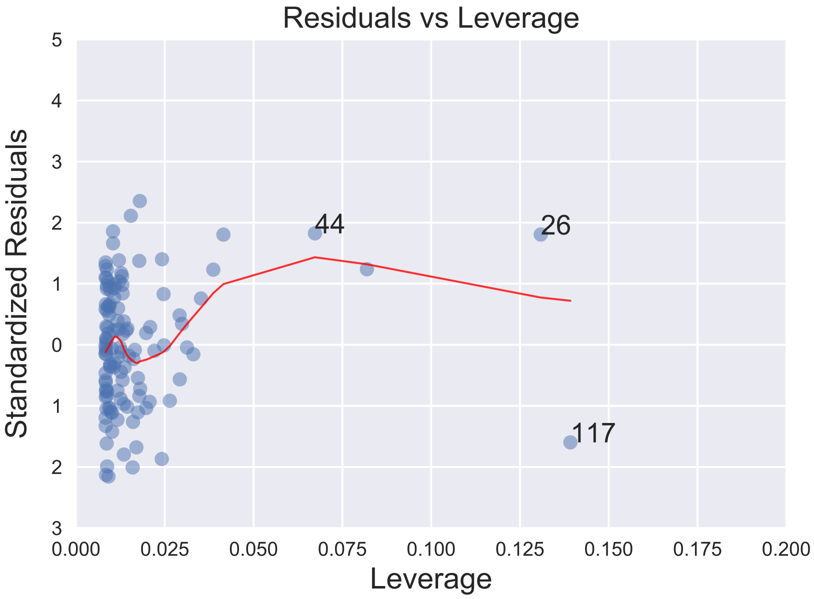
```
(0, 0.2)
```

```
fig_4.axes[0].set_ylim(-3, 5)
```

```
(-3, 5)
```

```
fig_4.axes[0].set_title('Residuals vs Leverage')
fig_4.axes[0].set_xlabel('Leverage')
fig_4.axes[0].set_ylabel('Standardized Residuals')


leverage_top3 = np.flip(np.argsort(cook_dist), 0)[:3]
```

```
for i in leverage_top3:
    fig_4.axes[0].annotate(i,
                                       xy=(leverage[i],
                                           norm_residuals[i]))
plt.show()
```


Residuals vs Leverage
Standardized Residuals
Leverage