Quick List of Tests and Commands for Python

Contents

1. The Symbolic Maths Using the "turnonsympy" module

For any of the commands below to work you need first have the following lines in the first line of a jupyter.ipynb file. Note that the first line needs to import the code from the file "turnonsympy.py". This file (turnonsympy.py) needs to be sitting within the directory you are using within jupyter.

```
from turnonsympy import *
```

Note that the graphics will appear inline within the juypter file. However, if you wish to generate the graphics outside then you can run the command

```
%matplotlib qt5
```

When you run a line a * should appear on the left hand side of the box. When it turns to a number the following should be functional

`y=2*x` will assign y to be identical to 2*x

`y=x**2+2` will assign y to be identical to x**2+2 (x**2 is x-squared, or x to the power of 2)

`eq(y,x)` will create an equation of the form y=x

`def f(x): return x**2` will create a function f(x) which squares
plot(expressions or equations) will plot the expression or equation. You need not use f or x and any function can be used e.g. you could specify
`def g(z): return z**3` where g is the function that cubes z

1.1 If you want a basic and quick help type helpme()

It will then prompt you for further details about what you wish to have help with. This will work only after you have loaded either the turnonsympy or shortcuts module.

1.2 Plotting functions

`plot(x**2)` or `plot(eq(y,x**2)` will give identical plots for x-squared note that more than one expression or equation can be entered e.g. `plot(x,x**2)` will plot both x and x**2

You can change set the range of x values from -10 to 10 by including

```
plot(x**2,(x,-10,10))
```

1.3 Solving Functions

solv1(expression or equation) solves the equation or expression in the bracket but making it equal zero e.g. solv1(x+1,x) will solve the for the value of x which makes (x+1)=0 i.e. (x=-1)

solv2(eq(y,x),x) would rearrange y=x to be solved in terms of x=ysolv2(expressions or equations)

solv2(x+2,2*x+4) would solve the simultaneous equations (y=x+2,y=2*x+4)

solv2(eq(y,x+2),eq(y,2*x+4) will also solve the simultaneous equations as above

equate(expressions)will set to equations equal and solve
e.g equate(x+1,2*x+2) would solve x+1=2*x+2

1.4 Calculus

diff(expression, equation or function,x) will differentiate a function

The following will all give the derivative of x-squared (2*x)

```
diff(x**2,x)
diff(eq(y,x**2),x)
diff(f(x),x)                    where def f(x): return x**2
```

diff(x**2,x,2) would take the second order derivative (and could be used with equations and functions also)

crits(expression or function)
will solve for the critical points for the expression, along with the second order derivatives

integrate(expression or function) will integrate the expression or function indefinitely
integrate(expression or function, (x,lower,upper)) will integrate the expression or function definitely between lower<x<upper

Note that you can plot derivatives directly e.g. plot(diff(x**2))

1.5 Matrices

A=M(1,2) will create the row vector (1,2)
A=M([1,2],[3,4]) will create the matrix with rows [1,2] and [3,4]

A.T would be the transpose of A
inv(A) will give the inverse of A
det(A) will give the determinant of A

Matrices can be added subtracted and multiplied.

e.g. A*B in matrix multiplication, A-B, A+B are B added and subtracted from A

1.6 Fractions

If you want to specify a fraction such as ½ then you can do so using frac(1,2). Adding subtracting and multiplying fractions will produce answers in the form of fractions

2. To use the "shortcuts" module

For any of the commands below to work you need first have the following lines in the first line of a jupyter.ipynb file. Note that the first line imports the code from the file "shortcuts.py" and needs to be sitting within the directory you are using within jupyter.

```
from shortcuts import *
```

Note that the graphics will appear inline within the juypter file. However, if you wish to generate the graphics outside then you can run the command

```
%matplotlib qt5
```

When you run a line a * should appear on the left hand side of the box. When it turns to a number the following should be functional.

***Warning:** do not run shortcuts or turnonsympy in the same file as there may be conflicts.

2.1 Loading Data

All of the commands below operate under the assumption that data is the Data Frame, when you load in data from Excel using

`data=load("full name of file including suffix")` it will be a data frame.

Note that if you use

`data=load(datadirec+ "filename")` this will direct you to download the file from a web address that I have set up for the files

One column of a data frame is a "series", and all the commands relating to data frames and series can be found below. However, this rather large array can be quite baffling, so we will use only a few. For a more extensive explanation you can look at the pandas module using the help menu provided in jupyter.

Note that for the instructions below, if a label e.g. y has commas ('') or ("") around it, this means that this argument must be a string (note a number) that identifies a variable name within a DataFrame.

If you wish to operate on any row subset of the data frame for the commands below, you need to precede it by selecting the subgroup of the rows
e.g. if the dataframe is called data (as below), and has a variable group has a subgroup denoted by 1, then obtaining s=data['group']==1 then for any of the commands below, replacing data with data[s] will operate on that subgroup e.g. describe(data) will describe the full data set, whereas describe(data[s]) will describe data subgroup defined by s

2.2 Some quick and important methods for manipulating with data frames

data=frame([1,2,3]) will create a data frame (it is just a shortened command for creating pandas.dataFrames) Thus data will have all the properties and functions of a pandas data frame object

data=frame([1,2,3],columns=['y'],index=['a','b',c']) will create a data frame with columns and index

data=findex(data) will reindex the data frame to have and index starting at one (Python likes to start from 0)

data.sort_values(['x,'y'],ascending=True) will sort a data set by the columns, x then y in ascending order

res=datatypes(data) will list all the variables and the types of data they contain

data=recode(data,'y',[oldvalues],[newvalues]) will recode the variable y from the old values to the new ones

data=recode(data,'y',integers(),alphabet()) will recode integers to the alphabet and vice versa if swapped

If you wish to select a subset of the dataframe according to the rule that variable y takes a particular value then

s=data['y']==value then data[s] is the dataframe containing rows for which y is equal to x

variableview(data) will give you a summary of the data with a drop down menu to look at particular variables

data_=nospace(data1) will remove all spaces from the variable names
data=lowercase(data1) will make all the variable names lower case
data=shorten(data1) will make shorter variable names without spaces

(you can specify using the first or first two words)

2.3 Statistics

2.3.1 Menu Driven Demands

The following command initiate menus that then employ the tests listed under direct commands

t_tests(data) will start a menu do to t-tests about means (with the data frame named data)
np_tests(data) will start a menu to do non-parametric tests about medians (with the data frame named data)
p_tests(data) will start a menu to do tests of proportions
u_tests(data) will start a menu to do uniformity tests
v_tests(data) will start a menu to test for variances
contables(data) will start a menu to do contingence tables
norm_tests(data) will start a menu to do test for normality
mean_CIs(data) will start a menu to construct confidence intervals about the mean
corrs(data) will start a menu to look at bivariate correlations
regression(data) will start a menu to conduct regressions and Anovas
probit(data) will start a menu to conduct binomial probit regressions
ordprobit(data) will start a menu to conduct ordinal probit regressions

A number of the options above have plot options however, there are some pure interactive plot options as below
scatterplot(data) will start a menu to do scatter plots
boxplot(data) will start a menu to do box plots
histogram(data) will start a menu to do histograms
counts(data) will start a menu plot counts of data (or proportions)

The following interactive command for pivot-tables is also included
pivot(data)

2.3.2 Direct Commands

The menu driven commands above use options that can be specified directly. A list of these direct commands can be seen below.

https://en.wikipedia.org/wiki/Descriptive_statistics
describe(data) Will give the summary statistics for all variables in data frame.

describe(data,y=['y','x]) Will give a set of summary statistics for x and y within data (they must be column labels)
https://en.wikipedia.org/wiki/Cronbach%27s_alpha

cronbachalpha(data,y='') Will give the Cronbach Alpha for the Variables in Columns of the data dataframe selected by y
https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c

a=princomp(data,y=['y','x'...],typ='corr',n=3,varimax=False)

typ=corr can be cov, varimax can be set to True
if corr, this is equivalent to normalising the data first then doing principle components.

n can be set to any number less than or equal to the number of variables used

```
a[0] #Eigenvalues etc
a[1] #Raw Factor Loadings
a[2] #Normalised Factor Loadings
a[3] #Unormalised PC
a[4] #Normalised  PCs
a[5] #Communalities, if n>0
a[6] #Normalised Varimax Rotated Factors
a[7] #Rotation
a[8] #Rotated Factor Loadings
a[9] #Communalities Rotated if n>0
```

https://en.wikipedia.org/wiki/K-means_clustering
out=kclust(data,y='',k=2) K means clustering, k is the number of groups
out[0] is the cluster centers
out[1] is the data set augmented with cluster membership

Tests about Distributions

https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test
norm_ks_test(data,y='y') Will test whether the variable y is normally distribution (using a Kolmogrov-Smirnov Test)

norm_ks_test(data,y='y',groupby='x',group=1) Will test the variable y for normality for the subgroup group1 of group
https://en.wikipedia.org/wiki/Pearson%27s_chi-squared_test

u_chi_test(data,y='y') This will test for a uniform distribution for the data y that is a categorical variable using a
chi-square test

Tests about Probabilities or Proportions

https://en.wikipedia.org/wiki/Binomial_test

http://www.surveyanalysis.org/wiki/Testing_Differences_Between_Proportions

p_bnm_test(data,y='y',p=0.5,groupy=1) This is a binomial test for y taking the value 1 with hypothesised probability 0.5

p_z_test(data,y='y',x=0.5,groupy=1) In this case it will test whether the probability of group1 within y is x=0.5

p_z_test(data,y='y',x='x',groupy=1,groupx=2)
z-Test for no difference in proportions for groupy within y compared to groupx within the grouping variable x

Means or medians

https://en.wikipedia.org/wiki/Confidence_interval
mean_ci(data, y=['y','x',...]) Will construct confidence intervals for the variables in y

mean_ci(data, y='y',groupby='x',confidence=0.95) Will construct confidence intervals on y by group x (just on y if group='')

https://en.wikipedia.org/wiki/Student%27s_t-test
m_t_test(data,y='y',x='x') independent t-Test the difference between the means of y and x (variable names)

m_t_test(data,y='y',x='x',paired=True) paired t-Test the difference between the means of y and x (variable names)

m_t_test(data,'y',mu=8) t-Test whether the mean of y1 is equal to the value set by x (e.g. 8)

m_t_test(data,y='y',groupby='x',group1=1,group2=2)
independent t-test for the difference in the Mean of of y for groups 1 and 2, grouped by the variable x, with the two groups being indicated by group1 and group2
https://en.wikipedia.org/wiki/Wilcoxon_signed-rank_test

m_w_test(data,y='y',mu=8) Wilcoxen sign rank test for distribution of y being symmetric around x (=8)

m_w_test(data,y='y',x='x') Wilcoxen sign rank test for the distribution of y-x being symmetric around 0. Note that for second sign rank test is the non-parametric equivalent of a paired t test
https://en.wikipedia.org/wiki/Mann%E2%80%93Whitney_U_test

m_mww_test(data,y='y',x='x') Mann-Whitney-Wilcoxon-Test the difference between the medians of y and x (also known as Mann Whitney U test)

m_www_test(data,y='y',groupby='x',group1=1,group2=2) #Mann-Whitney-Wilcoxon Test for the difference in the Mean of of y for groups 1 and 2, grouped by the variable x,, with the two groups being indicated group1 and group2

Variances

https://www.itl.nist.gov/div898/handbook/eda/section3/eda358.htm

v_chi_test(data,y='y',mu=1) Chi-Square test test for the variance of y being equal to some value, (in this case 1)
https://www.itl.nist.gov/div898/handbook/eda/section3/eda359.htm

https://en.wikipedia.org/wiki/F-test_of_equality_of_variances
v_f_test(data,y='y',x='x') F-Test of difference between variances between y and x

v_f_v_test(data,y='y',groupby='x',group1=1,group1=2)
F-Test for the difference in the variances of of y for groups 1 and 2, grouped by the variable x

https://en.wikipedia.org/wiki/Levene%27s_test
v_l_test(data,y=['y1,'y2',...])

v_l_test(data,y='y',groupby='x')
These are both Levene's test for common variances either for rows of variables or one variable by groups. Can be up to 10 variables

Dependency

See the section under testing for independence in
https://en.wikipedia.org/wiki/Pearson%27s_chi-squared_test
contable(data,y='y',x=x')
#Contingency Table with Chi Square Test for independence of variables y and x (categorical variables, it uses the chi-square test statistic)
https://en.wikipedia.org/wiki/Correlation_and_dependence

corr(data,y='y',x='x') Pearson Correlation between y and x

corr(data,y='y',x='x',spearman=True) The spearman correlation coefficient (Pearson on ranks)

corr(data,y='y',x='x',kendtau=True) The kendal tau correlation,(note there is also a ranking option if they are not numeric)

https://en.wikipedia.org/wiki/Regression_analysis
regols(formula,data) Conducts regressions
e.g.

regols('y=x',data) An OLS regression of y on x where x is treated continuously

regols('y=x+z',data) A multiple OLS regression on x and z (treated continuously)regols(data,'y~C(x)') #An OLS regression of y on x where x is treated as a categorical variable, also known as an "Anova"

regols(data,'y=C(x)+z') A multiple OLS regression on x, x treated continously and z (treated continuously) also known as an ANCOVA

Note: that for the myols command you can also specify a rank command
regols(data,formula,rank=True) In which case it will run a regression on the ranks of y rather than y. This is equivalent to
the performing Kruskall-Wallis test in the case where an ANOVA is specified

regdraw(data,'y=x',regressor=1) This runs a regression but gives a scatter plot with a line through it according to the numbered regressor (in this case 1 (the x regressor))
https://en.wikipedia.org/wiki/Analysis_of_variance
If you have run res=myols() or res=regdraw(), then there is also a subsequent command

anova(res) that will test for the significance of the categorical variables collectively

https://en.wikipedia.org/wiki/Probit_model
regprobit(data,'y=x') A Probit Regression of y on x where x is treated continuously

Statistics Commands not using Data Frames.

Some direct tests that do not require specification of dataframes (you enter variables or values directly (though in some cases these can be columns from a dataframe)

v_chi_test_(y,x=1) Chi-square test for variance of y equal to 1

v_l_test_(y) Levene's test for common variances on the columns of y

m_w_test_(y,x) Signed rank test

u_ks_test_(y) Test for normality of y

u_chi_test_([count1,count2,...]) Tests for unin
formative of the counts

p_bnm_test_(successes,trials, hyp value) Exact Binomial test for number of successe

p_z_test_(value(s),sample size(s), hypothesised value) proportions test

m_t_test_(y,x) t-test for no difference between mean two variables or ttest(y,value)

m_mww_test_(y,x) mann-whitney-wilcoxon test (also known as Mann Whitney U test)

v_f_test_(y,x) f-test for no difference in variances

contable_(table of counts) contingency table (test for independence)

ktau_(y,x) kendall tau between y and x

corr_(y,x) Standard correlation (Pearson, or Spearman if specified as True)
or

corr_(y,x,spearman=True) corr between y and x, must be numerical

Data Plot Commands

In what follows below data refers to a dataframe called data

https://en.wikipedia.org/wiki/Scatter_plot
plotscatter(data,'y','x') A scatter plot between y and x

https://en.wikipedia.org/wiki/Line_chart
plotsequence(data,y=['y','x']) A plot of the sequence of data, in this case for variables 'y' and 'x

https://en.wikipedia.org/wiki/Histogram
plothist(data,y='y',normed=True)

plothist(data,y='y',normal=True) In this case it will automatically norm the histogram and put a normal curve through it
This is a plot of the frequency distribution described in https://en.wikipedia.org/wiki/Frequency_distribution

plotcounts(data,y='y',normed=True) The default will be a bar graph, if norm is False then, will be counts not proportions

plotcounts(data,y='y',pie=True) A pie graph with proportions
This is a bar chart of the means https://en.wikipedia.org/wiki/Bar_chart

plotmeanbar(data,y=['y','x']) Bar Graph of Means of variables 'y' and 'x' with confidence intervals

plotmeanbar(data,y='y',groupby='x')Bar Graph of Means of variables 'y' grouped by 'x' (a categorical variable) with CIs

Note that the default confidence=0.95 but this can be changed within the plotmeanbar command.

https://en.wikipedia.org/wiki/Box_plot
plotbox(data,y=['y','x']) Box Plot of 'y' and 'x'

plotbox(data,y='y', groupby='x') Box Plot of 'y' grouped by the variable 'x' (a categorical vairable)plotbar(data) #Bar Graph If df is simply a data frame or list with counts

plotpie(data) Pie Graph If df is simply a data frame or list with counts, if counts it will give proportions

2.4 Maths Plot Commands

plotfunction(0,c1=1,p1=1,c2=1,p2=2,c3=1,p3=1) Plot a polynomial coefficients c and powers p, and its derivative belowplotline(a,b) plot a line y=a+b*x

plotlinefrompoints([x0,x1],[x1,y1])Plot a point through the coordinates

https://en.wikipedia.org/wiki/Break-even_(economics)

plotbreakeven(fc=10,mc=0.8,mr=0.9) Find and plot the break-even point with fixed cost fc, marginal cost mc and marginal rev

mrplotbreakeven_exam(point0,point1,mr,addfc=5,addmc=0,addmr=0) This will solve a break even problem as past exams: see examples

https://gcseguide.co.uk/maths/algebra/simultaneous-equations/
plotsimult(a0,b0,a1,b1) Solve and plot the simultaneous Equations y=a0 + b0*x and y=a1+ b1*x

https://en.wikipedia.org/wiki/Supply_and_demand
plotSD(s0,s1,d0,d1,loglinear=True) Solve and plot the supply and demand equations Supply =s0+s1*p, Demand=d0+d1*p
If loglinear is true then the equations are in logged quantities and prices

2.5 Discounting (Present Value)

https://en.wikipedia.org/wiki/Net_present_value
y=presentval(flow,r,plot=True) Calculate the net present value of the flow series and interest rate r
https://en.wikipedia.org/wiki/Annuity

y=annuity(x,r,t) The value of an annuity for amount x with interest rate r for time T

2.6 Distributions and Plots of Distributions

2.61 Discrete Distributions

https://en.wikipedia.org/wiki/Binomial_distribution
dist=binom(n,p) Binomial Distribution for n Trials with probability of success p

prb=dist.pmf(x) prb is the probability of x successes

cprb=dist.cdf(x) cprb is the probabilit of less than or equal to x successes
https://en.wikipedia.org/wiki/Negative_binomial_distribution

dist=nbinom(x,p) Negative Binomial Distribution for x=Number of Successes probability of success=p

prb=dist.pmf(z) Probability of z failures to get x successes

cprb=dist.cdf(z)Probability of less than or equal to z failures to get x failures

Note that because z=n-x
nbinom(x,p).pmf(n-x) The probability of needing n trials to obtain x successes

https://en.wikipedia.org/wiki/Poisson_distribution
dist=poisson(mu) Poisson distribution with mean number of events mu

prb=dist.pmf(x) Probability of x events

cprb=dist.cdf(x) Probability of less than or equal to x

eventsf=distmap(dist,x or z) will give the full range or probabilities (cumulative or pointwise) for any of the distributions above
***Note that the distmap is essential to complete a QM1 assignment.

You can plot these distributions using
plotbinom(n,p) for the binomial

plotnbinom(x,p)for the negative binomial

plotpossion(mu) for the poisson

and plotuniform(n)for the uniform

2.62 Continuous Distributions

https://en.wikipedia.org/wiki/Normal_distribution
https://en.wikipedia.org/wiki/Student%27s_t-distribution

https://en.wikipedia.org/wiki/Chi-squared_distribution
https://en.wikipedia.org/wiki/F-distribution

norm(mean,stdv) the normal

tdist(df) the t-distribution

chidist(df) the chi-square distribution

fdist(df1,df2) For any of the distributions above, you can specify the distribution and find the probabilities of more than and less than up to two points e.g.

distcmap(norm(0,1),2) you can call up distributions (in this case normal with mean 0 and variance 1) then find the probability above and below or between values. For the case above it finds the probability above and below 2. For the case were distcmap(norm(0,1),-1,2) it will give the same information but also the probability between -1 and 2

plotdist(val=.6,tail='u',distr=norm,mean=1,stdv=2,value=False) Normal mean=1, variance 2, with upper probability 0.6

plotdist(val=1,tail='u', distr=norm,mean=1,stdv=2, value=True) Normal mean=1, variance 2, with probability for value above 1

Note if you change tail to 's' or 'l' it will split or put the lower probabilities rather than upper probabilities
This particular function can also plot the same for t-distributions, chi-square, and F-distributions, examples being

plotdist(val=.6,tail='u',distr=chidist,df=5,value=False)

plotdist(val=.6,tail='s',distr=tdist, df=5, value=False)

plotdist(val=.6,tail='s',distr=fdist, df=5,df2=10,value=False)

3 Seven things to check if your code does not appear to be working.

1. **%matplotlib** currently needs to read **%matplotlib qt5** whereas a previous version had this as a default and did not need to be stated. Also, in some previous versions a space was allowed i.e. **% matplotlib** but is now no longer an option for python 3.7. Therefore, in any example files containing **%matplotlib or % matplotlib** should be changed to **%matplotlib qt5** or alternatively the command should be removed altogether.
2. When running shortcuts or turnonsympy, you need to have the files in your working directory (QM). They also need to be called turnonsympy.py and shortcuts.py. Any variations will cause an error. BB tends to add version numbers to the file, so you need to remove these.
3. Running a line without having first called the shortcuts or turnonsympy command will cause an error. **It is not sufficient to have these lines in the jupyter notebook. They have to be run first. The most common error is when students say there file is not working is because they have not run the first line importing the shortcuts or turnonsympy files.**
4. Once you have imported shortcuts or turnonsympy they will remain loaded until it is the kernel is restarted. Therefore, if you run both at the same time (as in the same jupyter notebook) a conflict may occur and you may get errors in some commands.
5. Python is case sensitive. If you have a capital letter anywhere where it should be lower case or vice versa, it be viewed as an error. Likewise if you have an empty space before after or within an instruction this may trigger and error.
6. If you are importing data frames from excel the first line of the sheet needs to the names of the variables. There are also certain characters that can cause problems, and it is good practice to avoid blank spaces in these names as variables.
7. The original shortcuts.py file on BB this year had a deprecated command for the load command for the very latest version of pandas. If the load statement does not work then replace the shortcuts.py with the current one on BB and the load command will then work.