Last login: Wed Feb 5 20:25:49 on ttys000

**➜ ~** ls

**Desktop** **Library** **Pictures** file\_new.txt text\_file.txt

**Documents** **Movies** **Public** pwd

**Downloads** **Music** file\_copied.txt **temp**

**➜ ~** cd desktop

**➜ desktop** ls

010114 seanmok.docx

011313 roncv.doc

050213checklist HK.pdf

060313 roncv.docx

060813 BALANCES.xlsx

072313 roncv.docx

091513 Response to final OA\_XGLK-1201-USPT (00172972-2).doc

12130265g.txt

13098775g.txt

3D prinbting

401-multreg.pdf

9c Hoi To Court

**ACFE**

Additional functions available to General Counsel and designated.docx

**Animals and the future**

Begin forwarded message---.textClipping

**CC&A**

Cloud Forensics Keynote.pptx

Copy of 072613 Help to translatex.xlsx

**DS\_HK\_1**

**DS\_HK\_1 lessons lesson02 matrix.py at 2678b50506569ed2d4305110e40f5300806d138f ?? ga-students DS\_HK\_1\_files**

**E Y 2014**

Foreword.docx

**General Assembly**

**Gilkron**

HK Poly Ver 6 7,8

**HKU**

HSF content for gilkron.xlsx

**ICRA 2014**

**IP Strategy Book**

Launcher.app

**Misc**

Music

NatlSummChart\_Classof2011.pdf

Patent application

**Photographs**

Presentation1.pptx

Presentation3.pptx

**Profectional**

Response to final OA\_XGLK-1201-USPT (00172972-2).doc

Ron Yu Signature copy.jpg

**Ron yu**

**Split&Concat.app**

**Survival**

**The Fraud Hunter**

**The Hive and General Assembly**

Untitled.rtf

Whats-next-for-China-Jan-22-v2.pdf

cs229-linalg.pdf

ipresearch-bankingip.pdf

j.1467-9639.2010.00437.x.pdf

preface.docx

shield.docx

testing.txt

**➜ desktop** cd ds\_hk\_1

**➜ ds\_hk\_1 git:(gh-pages) ✗** ls

DS\_HK\_1.sublime-project **finals**

Project1.sublime-workspace **lessons**

cs229-linalg.pdf readme.md

**data** ronald

**➜ ds\_hk\_1 git:(gh-pages) ✗** cd lessons

**➜ lessons git:(gh-pages) ✗** ls

**class** **lesson01** **lesson02** readme.md **ronald**

**➜ lessons git:(gh-pages) ✗** git pull upstream gh-pages

From github.com:ga-students/DS\_HK\_1

\* branch gh-pages -> FETCH\_HEAD

Already up-to-date.

**➜ lessons git:(gh-pages) ✗** git push

Counting objects: 23, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (13/13), done.

Writing objects: 100% (13/13), 1.47 KiB | 0 bytes/s, done.

Total 13 (delta 7), reused 0 (delta 0)

To git@github.com:hermania/DS\_HK\_1.git

9a38032..5b12edc gh-pages -> gh-pages

**➜ lessons git:(gh-pages) ✗**

**➜ lessons git:(gh-pages) ✗** y = array([[1], [2], [3], [4]])

zsh: bad base syntax

zsh: bad base syntax

**➜ lessons git:(gh-pages) ✗** n = inv(dot(X.T, X))

zsh: no matches found: inv(dot(X.T, X))

**➜ lessons git:(gh-pages) ✗** k = dot(X.T, y)

zsh: unknown file attribute

**➜ lessons git:(gh-pages) ✗** coef\_ = dot(n, k)ipython

zsh: no matches found: dot(n, k)ipython

**➜ lessons git:(gh-pages) ✗** cd..

**➜ ds\_hk\_1 git:(gh-pages) ✗** cd ..

**➜ desktop** ipython

Python 2.7.5 |Anaconda 1.8.0 (x86\_64)| (default, Oct 24 2013, 07:02:20)

Type "copyright", "credits" or "license" for more information.

IPython 1.1.0 -- 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 [**1**]:

In [**2**]: from numpy import array, dot

In [**3**]: from numpy.linalg import inv

In [**4**]:

In [**4**]: X = array([[1, 1], [1, 2], [1, 3], [1, 4]])

In [**5**]: y = array([[1], [2], [3], [4]])

In [**6**]: n = inv(dot(X.T, X))

In [**7**]: k = dot(X.T, y)

In [**8**]: coef\_ = dot(n, k)

In [**9**]: from numpy import \*

In [**10**]: y

Out[**10**]:

array([[1],

[2],

[3],

[4]])

In [**11**]: coef

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-11-ee27e174ffff> in <module>()

----> 1 coef

NameError: name 'coef' is not defined

In [**12**]: coef\_

Out[**12**]:

array([[ 0.],

[ 1.]])

In [**13**]: arrayOne

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-13-d63952d23479> in <module>()

----> 1 arrayOne

NameError: name 'arrayOne' is not defined

In [**14**]: from numpy import \*

In [**15**]: arrayOne = arange(15).reshape(3, 5)

In [**16**]: arrayone

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-16-3957af624024> in <module>()

----> 1 arrayone

NameError: name 'arrayone' is not defined

In [**17**]: arrayOne

Out[**17**]:

array([[ 0, 1, 2, 3, 4],

[ 5, 6, 7, 8, 9],

[10, 11, 12, 13, 14]])

In [**18**]: arrayTwo = arange(15).reshape(5, 3)

In [**19**]: arrayTwo

Out[**19**]:

array([[ 0, 1, 2],

[ 3, 4, 5],

[ 6, 7, 8],

[ 9, 10, 11],

[12, 13, 14]])

In [**20**]: vector = array([10, 15, 20])

In [**21**]: vector

Out[**21**]: array([10, 15, 20])

In [**22**]: print vector

[10 15 20]

In [**23**]: matrixOne = matrix('1 2 3; 4 5 6')

In [**24**]: matrixOne

Out[**24**]:

matrix([[1, 2, 3],

[4, 5, 6]])

In [**25**]: matrixTwo = matrix('1 2; 3 4; 5 6')

In [**26**]: matrixTwo

Out[**26**]:

matrix([[1, 2],

[3, 4],

[5, 6]])

In [**27**]: print(vector)

[10 15 20]

In [**28**]: print(arrayOne)

[[ 0 1 2 3 4]

[ 5 6 7 8 9]

[10 11 12 13 14]]

In [**29**]: print(arrayTwo)

[[ 0 1 2]

[ 3 4 5]

[ 6 7 8]

[ 9 10 11]

[12 13 14]]

In [**30**]: a1 = array([ [1, 2], [3, 4] ])

In [**31**]: a2 = array([ [1, 3], [2, 4] ])

In [**32**]: m1 = matrix('1 2; 3 4')

In [**33**]: m2 = matrix('1 3; 2 4')

In [**34**]: a1 \* a2

Out[**34**]:

array([[ 1, 6],

[ 6, 16]])

In [**35**]: m1 \* m2

Out[**35**]:

matrix([[ 5, 11],

[11, 25]])

In [**36**]: a1.T

Out[**36**]:

array([[1, 3],

[2, 4]])

In [**37**]: m1.T

Out[**37**]:

matrix([[1, 3],

[2, 4]])

In [**38**]: iFive = eye(10)

In [**39**]: eye(10)

Out[**39**]:

array([[ 1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],

[ 0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],

[ 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],

[ 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],

[ 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],

[ 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],

[ 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],

[ 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],

[ 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],

[ 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]])

In [**40**]: dot(a1, a2)

Out[**40**]:

array([[ 5, 11],

[11, 25]])

In [**41**]: dot(m1,m2)

Out[**41**]:

matrix([[ 5, 11],

[11, 25]])

In [**42**]:

In [**42**]: t.timeit('dot(a1, a2)', setup="from \_\_main\_\_ import dot, a1, a2", number = 100000)

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-42-c6e8c0b470c8> in <module>()

----> 1 t.timeit('dot(a1, a2)', setup="from \_\_main\_\_ import dot, a1, a2", number = 100000)

NameError: name 't' is not defined

In [**43**]: import timeit as t

In [**44**]: t.timeit('dot(a1, a2)', setup="from \_\_main\_\_ import dot, a1, a2", number = 100000)

Out[**44**]: 0.13025498390197754

In [**45**]: exp(100)

Out[**45**]: 2.6881171418161356e+43

In [**46**]: log(3)

Out[**46**]: 1.0986122886681098

In [**47**]: sqrt(144)

Out[**47**]: 12.0

In [**48**]: sqrt(576)

Out[**48**]: 24.0

In [**49**]: sqrt(1253241546542788)

Out[**49**]: 35401151.768590637

In [**50**]: import pandas as pd

In [**51**]: df = pd.read\_csv('nytimes1.csv')

---------------------------------------------------------------------------

IOError Traceback (most recent call last)

<ipython-input-51-69e7456ad300> in <module>()

----> 1 df = pd.read\_csv('nytimes1.csv')

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in parser\_f(filepath\_or\_buffer, sep, dialect, compression, doublequote, escapechar, quotechar, quoting, skipinitialspace, lineterminator, header, index\_col, names, prefix, skiprows, skipfooter, skip\_footer, na\_values, na\_fvalues, true\_values, false\_values, delimiter, converters, dtype, usecols, engine, delim\_whitespace, as\_recarray, na\_filter, compact\_ints, use\_unsigned, low\_memory, buffer\_lines, warn\_bad\_lines, error\_bad\_lines, keep\_default\_na, thousands, comment, decimal, parse\_dates, keep\_date\_col, dayfirst, date\_parser, memory\_map, nrows, iterator, chunksize, verbose, encoding, squeeze, mangle\_dupe\_cols, tupleize\_cols)

**398** )

**399**

--> 400 return \_read(filepath\_or\_buffer, kwds)

**401**

**402** parser\_f.\_\_name\_\_ = name

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_read(filepath\_or\_buffer, kwds)

**196**

**197** # Create the parser.

--> 198 parser = TextFileReader(filepath\_or\_buffer, \*\*kwds)

**199**

**200** if nrows is not None:

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, f, engine, \*\*kwds)

**477** self.options['has\_index\_names'] = kwds['has\_index\_names']

**478**

--> 479 self.\_make\_engine(self.engine)

**480**

**481** def \_get\_options\_with\_defaults(self, engine):

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_make\_engine(self, engine)

**584** def \_make\_engine(self, engine='c'):

**585** if engine == 'c':

--> 586 self.\_engine = CParserWrapper(self.f, \*\*self.options)

**587** else:

**588** if engine == 'python':

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, src, \*\*kwds)

**955** kwds['allow\_leading\_cols'] = self.index\_col is not False

**956**

--> 957 self.\_reader = \_parser.TextReader(src, \*\*kwds)

**958**

**959** # XXX

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_\_cinit\_\_ (pandas/parser.c:2987)()

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_setup\_parser\_source (pandas/parser.c:5345)()

IOError: File nytimes1.csv does not exist

In [**52**]: df = pd.read\_csv('nytime1.csv')

---------------------------------------------------------------------------

IOError Traceback (most recent call last)

<ipython-input-52-8d21465e4da8> in <module>()

----> 1 df = pd.read\_csv('nytime1.csv')

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in parser\_f(filepath\_or\_buffer, sep, dialect, compression, doublequote, escapechar, quotechar, quoting, skipinitialspace, lineterminator, header, index\_col, names, prefix, skiprows, skipfooter, skip\_footer, na\_values, na\_fvalues, true\_values, false\_values, delimiter, converters, dtype, usecols, engine, delim\_whitespace, as\_recarray, na\_filter, compact\_ints, use\_unsigned, low\_memory, buffer\_lines, warn\_bad\_lines, error\_bad\_lines, keep\_default\_na, thousands, comment, decimal, parse\_dates, keep\_date\_col, dayfirst, date\_parser, memory\_map, nrows, iterator, chunksize, verbose, encoding, squeeze, mangle\_dupe\_cols, tupleize\_cols)

**398** )

**399**

--> 400 return \_read(filepath\_or\_buffer, kwds)

**401**

**402** parser\_f.\_\_name\_\_ = name

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_read(filepath\_or\_buffer, kwds)

**196**

**197** # Create the parser.

--> 198 parser = TextFileReader(filepath\_or\_buffer, \*\*kwds)

**199**

**200** if nrows is not None:

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, f, engine, \*\*kwds)

**477** self.options['has\_index\_names'] = kwds['has\_index\_names']

**478**

--> 479 self.\_make\_engine(self.engine)

**480**

**481** def \_get\_options\_with\_defaults(self, engine):

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_make\_engine(self, engine)

**584** def \_make\_engine(self, engine='c'):

**585** if engine == 'c':

--> 586 self.\_engine = CParserWrapper(self.f, \*\*self.options)

**587** else:

**588** if engine == 'python':

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, src, \*\*kwds)

**955** kwds['allow\_leading\_cols'] = self.index\_col is not False

**956**

--> 957 self.\_reader = \_parser.TextReader(src, \*\*kwds)

**958**

**959** # XXX

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_\_cinit\_\_ (pandas/parser.c:2987)()

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_setup\_parser\_source (pandas/parser.c:5345)()

IOError: File nytime1.csv does not exist

In [**53**]: df = pd.read\_csv('nytimes.csv')

---------------------------------------------------------------------------

IOError Traceback (most recent call last)

<ipython-input-53-d5b6abaeabc0> in <module>()

----> 1 df = pd.read\_csv('nytimes.csv')

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in parser\_f(filepath\_or\_buffer, sep, dialect, compression, doublequote, escapechar, quotechar, quoting, skipinitialspace, lineterminator, header, index\_col, names, prefix, skiprows, skipfooter, skip\_footer, na\_values, na\_fvalues, true\_values, false\_values, delimiter, converters, dtype, usecols, engine, delim\_whitespace, as\_recarray, na\_filter, compact\_ints, use\_unsigned, low\_memory, buffer\_lines, warn\_bad\_lines, error\_bad\_lines, keep\_default\_na, thousands, comment, decimal, parse\_dates, keep\_date\_col, dayfirst, date\_parser, memory\_map, nrows, iterator, chunksize, verbose, encoding, squeeze, mangle\_dupe\_cols, tupleize\_cols)

**398** )

**399**

--> 400 return \_read(filepath\_or\_buffer, kwds)

**401**

**402** parser\_f.\_\_name\_\_ = name

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_read(filepath\_or\_buffer, kwds)

**196**

**197** # Create the parser.

--> 198 parser = TextFileReader(filepath\_or\_buffer, \*\*kwds)

**199**

**200** if nrows is not None:

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, f, engine, \*\*kwds)

**477** self.options['has\_index\_names'] = kwds['has\_index\_names']

**478**

--> 479 self.\_make\_engine(self.engine)

**480**

**481** def \_get\_options\_with\_defaults(self, engine):

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_make\_engine(self, engine)

**584** def \_make\_engine(self, engine='c'):

**585** if engine == 'c':

--> 586 self.\_engine = CParserWrapper(self.f, \*\*self.options)

**587** else:

**588** if engine == 'python':

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, src, \*\*kwds)

**955** kwds['allow\_leading\_cols'] = self.index\_col is not False

**956**

--> 957 self.\_reader = \_parser.TextReader(src, \*\*kwds)

**958**

**959** # XXX

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_\_cinit\_\_ (pandas/parser.c:2987)()

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_setup\_parser\_source (pandas/parser.c:5345)()

IOError: File nytimes.csv does not exist

In [**54**]: df = pd.read\_csv('nytimes.csv')

---------------------------------------------------------------------------

IOError Traceback (most recent call last)

<ipython-input-54-d5b6abaeabc0> in <module>()

----> 1 df = pd.read\_csv('nytimes.csv')

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in parser\_f(filepath\_or\_buffer, sep, dialect, compression, doublequote, escapechar, quotechar, quoting, skipinitialspace, lineterminator, header, index\_col, names, prefix, skiprows, skipfooter, skip\_footer, na\_values, na\_fvalues, true\_values, false\_values, delimiter, converters, dtype, usecols, engine, delim\_whitespace, as\_recarray, na\_filter, compact\_ints, use\_unsigned, low\_memory, buffer\_lines, warn\_bad\_lines, error\_bad\_lines, keep\_default\_na, thousands, comment, decimal, parse\_dates, keep\_date\_col, dayfirst, date\_parser, memory\_map, nrows, iterator, chunksize, verbose, encoding, squeeze, mangle\_dupe\_cols, tupleize\_cols)

**398** )

**399**

--> 400 return \_read(filepath\_or\_buffer, kwds)

**401**

**402** parser\_f.\_\_name\_\_ = name

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_read(filepath\_or\_buffer, kwds)

**196**

**197** # Create the parser.

--> 198 parser = TextFileReader(filepath\_or\_buffer, \*\*kwds)

**199**

**200** if nrows is not None:

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, f, engine, \*\*kwds)

**477** self.options['has\_index\_names'] = kwds['has\_index\_names']

**478**

--> 479 self.\_make\_engine(self.engine)

**480**

**481** def \_get\_options\_with\_defaults(self, engine):

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_make\_engine(self, engine)

**584** def \_make\_engine(self, engine='c'):

**585** if engine == 'c':

--> 586 self.\_engine = CParserWrapper(self.f, \*\*self.options)

**587** else:

**588** if engine == 'python':

//anaconda/lib/python2.7/site-packages/pandas/io/parsers.pyc in \_\_init\_\_(self, src, \*\*kwds)

**955** kwds['allow\_leading\_cols'] = self.index\_col is not False

**956**

--> 957 self.\_reader = \_parser.TextReader(src, \*\*kwds)

**958**

**959** # XXX

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_\_cinit\_\_ (pandas/parser.c:2987)()

//anaconda/lib/python2.7/site-packages/pandas/parser.so in pandas.parser.TextReader.\_setup\_parser\_source (pandas/parser.c:5345)()

IOError: File nytimes.csv does not exist

In [**55**]: ls

010114 seanmok.docx

011313 roncv.doc

050213checklist HK.pdf

060313 roncv.docx

060813 BALANCES.xlsx

072313 roncv.docx

091513 Response to final OA\_XGLK-1201-USPT (00172972-2).doc

12130265g.txt

13098775g.txt

3D prinbting/

401-multreg.pdf

9c Hoi To Court/

**ACFE**/

Additional functions available to General Counsel and designated.docx\*

**Animals and the future**/

Begin forwarded message---.textClipping

**CC&A**/

Cloud Forensics Keynote.pptx\*

Copy of 072613 Help to translatex.xlsx\*

**DS\_HK\_1**/

**DS\_HK\_1 lessons lesson02 matrix.py at 2678b50506569ed2d4305110e40f5300806d138f ?? ga-students DS\_HK\_1\_files**/

**E Y 2014**/

Foreword.docx

**General Assembly**/

**Gilkron**/

HK Poly Ver 6 7,8/

**HKU**/

HSF content for gilkron.xlsx

**ICRA 2014**/

**IP Strategy Book**/

Launcher.app@

**Misc**/

Music/

NatlSummChart\_Classof2011.pdf

Patent application/

**Photographs**/

Presentation1.pptx

Presentation3.pptx

**Profectional**/

Response to final OA\_XGLK-1201-USPT (00172972-2).doc\*

Ron Yu Signature copy.jpg\*

**Ron yu**/

**Split&Concat.app**/

**Survival**/

**The Fraud Hunter**/

**The Hive and General Assembly**/

Untitled.rtf

Whats-next-for-China-Jan-22-v2.pdf

cs229-linalg.pdf

ipresearch-bankingip.pdf

j.1467-9639.2010.00437.x.pdf

preface.docx

shield.docx

testing.txt

In [**56**]: cd DS\_hK\_1

/Users/ronaldyu/Desktop/DS\_HK\_1

In [**57**]: ls

DS\_HK\_1.sublime-project **finals**/

Project1.sublime-workspace **lessons**/

cs229-linalg.pdf readme.md

**data**/ ronald

In [**58**]: cd lessons

/Users/ronaldyu/Desktop/DS\_HK\_1/lessons

In [**59**]: ls

**class**/ **lesson01**/ **lesson02**/ readme.md **ronald**/

In [**60**]: cd ronald

/Users/ronaldyu/Desktop/DS\_HK\_1/lessons/ronald

In [**61**]: ls

In [**62**]: cd lesson03

[Errno 2] No such file or directory: 'lesson03'

/Users/ronaldyu/Desktop/DS\_HK\_1/lessons/ronald

In [**63**]: cd ..

/Users/ronaldyu/Desktop/DS\_HK\_1/lessons

In [**64**]: ls

**class**/ **lesson01**/ **lesson02**/ readme.md **ronald**/

In [**65**]: cd ronald

/Users/ronaldyu/Desktop/DS\_HK\_1/lessons/ronald

In [**66**]: ls

nytimes.csv\*

In [**67**]: /Users/ronaldyu/Desktop/DS\_HK\_1/lessons/class/lesson03/nytimes.csv

File "<ipython-input-67-45ee8ff7a505>", line 1

Users(/ronaldyu/Desktop/DS\_HK\_1/lessons/class/lesson03/nytimes.csv)

^

SyntaxError: invalid syntax

In [**68**]: df = pd.read\_csv('nytimes.csv')

In [**69**]: df

Out[**69**]:

<class 'pandas.core.frame.DataFrame'>

Int64Index: 458441 entries, 0 to 458440

Data columns (total 5 columns):

Age 458441 non-null values

Gender 458441 non-null values

Impressions 458441 non-null values

Clicks 458441 non-null values

Signed\_In 458441 non-null values

dtypes: int64(5)

In [**70**]: df.describe()

Out[**70**]:

Age Gender Impressions Clicks Signed\_In

count 458441.000000 458441.000000 458441.000000 458441.000000 458441.000000

mean 29.482551 0.367037 5.007316 0.092594 0.700930

std 23.607034 0.481997 2.239349 0.309973 0.457851

min 0.000000 0.000000 0.000000 0.000000 0.000000

25% 0.000000 0.000000 3.000000 0.000000 0.000000

50% 31.000000 0.000000 5.000000 0.000000 1.000000

75% 48.000000 1.000000 6.000000 0.000000 1.000000

max 108.000000 1.000000 20.000000 4.000000 1.000000

In [**71**]: df[:10]

Out[**71**]:

Age Gender Impressions Clicks Signed\_In

0 36 0 3 0 1

1 73 1 3 0 1

2 30 0 3 0 1

3 49 1 3 0 1

4 47 1 11 0 1

5 47 0 11 1 1

6 0 0 7 1 0

7 46 0 5 0 1

8 16 0 3 0 1

9 52 0 4 0 1

In [**72**]: dfg = df[ ['Age', 'Impressions', 'Clicks'] ].groupby(['Age']).agg([numpy.mean])

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-72-ea92db767786> in <module>()

----> 1 dfg = df[ ['Age', 'Impressions', 'Clicks'] ].groupby(['Age']).agg([numpy.mean])

NameError: name 'numpy' is not defined

In [**73**]: dfg[:10]dfg = df[ ['Age', 'Impressions', 'Clicks'] ].groupby(['Age']).agg([numpy.mean])

File "<ipython-input-73-60a9af8997ac>", line 1

dfg[:10]dfg = df[ ['Age', 'Impressions', 'Clicks'] ].groupby(['Age']).agg([numpy.mean])

^

SyntaxError: invalid syntax

In [**74**]: dfg[:10]

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-74-b04f35c294e7> in <module>()

----> 1 dfg[:10]

NameError: name 'dfg' is not defined

In [**75**]: dfg = df[ ['Age', 'Impressions', 'Clicks'] ].groupby(['Age']).agg([numpy.mean])

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-75-ea92db767786> in <module>()

----> 1 dfg = df[ ['Age', 'Impressions', 'Clicks'] ].groupby(['Age']).agg([numpy.mean])

NameError: name 'numpy' is not defined

In [**76**]: import numpy

In [**77**]: dfg = df[ ['Age', 'Impressions', 'Clicks'] ].groupby(['Age']).agg([numpy.mean])

In [**78**]: dfg[:10]

Out[**78**]:

Impressions Clicks

mean mean

Age

0 4.999657 0.142080

7 5.200000 0.600000

8 6.266667 0.266667

9 5.520833 0.104167

10 4.814159 0.168142

11 5.095406 0.127208

12 4.885666 0.156997

13 4.995279 0.149197

14 5.035506 0.133877

15 4.944338 0.150480

In [**79**]: df['log\_impressions'] = df['Impressions'].apply(numpy.log)

In [**80**]: def map\_age\_category(x):

....: if x < 18:

....: return '1'

....: elif x < 25:

....: return '2'

....: elif x < 32:

....: return '3'

....: elif x < 45:

....: return '4'

....: else:

....: return '5'

....:

In [**81**]: df['age\_categories'] = df['Age'].apply(map\_age\_category)

In [**82**]: df[5]

---------------------------------------------------------------------------

KeyError Traceback (most recent call last)

<ipython-input-82-0c7cf4ee5b30> in <module>()

----> 1 df[5]

//anaconda/lib/python2.7/site-packages/pandas/core/frame.pyc in \_\_getitem\_\_(self, key)

**2001** # get column

**2002** if self.columns.is\_unique:

-> 2003 return self.\_get\_item\_cache(key)

**2004**

**2005** # duplicate columns

//anaconda/lib/python2.7/site-packages/pandas/core/generic.pyc in \_get\_item\_cache(self, item)

**665** return cache[item]

**666** except Exception:

--> 667 values = self.\_data.get(item)

**668** res = self.\_box\_item\_values(item, values)

**669** cache[item] = res

//anaconda/lib/python2.7/site-packages/pandas/core/internals.pyc in get(self, item)

**1653** def get(self, item):

**1654** if self.items.is\_unique:

-> 1655 \_, block = self.\_find\_block(item)

**1656** return block.get(item)

**1657** else:

//anaconda/lib/python2.7/site-packages/pandas/core/internals.pyc in \_find\_block(self, item)

**1933**

**1934** def \_find\_block(self, item):

-> 1935 self.\_check\_have(item)

**1936** for i, block in enumerate(self.blocks):

**1937** if item in block:

//anaconda/lib/python2.7/site-packages/pandas/core/internals.pyc in \_check\_have(self, item)

**1940** def \_check\_have(self, item):

**1941** if item not in self.items:

-> 1942 raise KeyError('no item named %s' % com.pprint\_thing(item))

**1943**

**1944** def reindex\_axis(self, new\_axis, method=None, axis=0, copy=True):

KeyError: u'no item named 5'

In [**83**]: df[age]

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-83-508c108685d6> in <module>()

----> 1 df[age]

NameError: name 'age' is not defined

In [**84**]: df[Age]

---------------------------------------------------------------------------

NameError Traceback (most recent call last)

<ipython-input-84-dab6d3b3a71c> in <module>()

----> 1 df[Age]

NameError: name 'Age' is not defined

In [**85**]: cd['Age']

[Errno 2] No such file or directory: '[Age]'

/Users/ronaldyu/Desktop/DS\_HK\_1/lessons/ronald

In [**86**]: df['Age']

Out[**86**]:

0 36

1 73

2 30

3 49

4 47

5 47

6 0

7 46

8 16

9 52

10 0

11 21

12 0

13 57

14 31

...

458426 49

458427 43

458428 40

458429 49

458430 0

458431 21

458432 30

458433 21

458434 61

458435 51

458436 0

458437 0

458438 72

458439 0

458440 0

Name: Age, Length: 458441, dtype: int64

In [**87**]: df['Impressions']

Out[**87**]:

0 3

1 3

2 3

3 3

4 11

5 11

6 7

7 5

8 3

9 4

10 8

11 3

12 4

13 6

14 5

...

458426 9

458427 9

458428 4

458429 11

458430 6

458431 5

458432 4

458433 2

458434 6

458435 7

458436 2

458437 4

458438 5

458439 5

458440 3

Name: Impressions, Length: 458441, dtype: int64

In [**88**]: df['Clicks']

Out[**88**]:

0 0

1 0

2 0

3 0

4 0

5 1

6 1

7 0

8 0

9 0

10 1

11 0

12 0

13 0

14 0

...

458426 0

458427 0

458428 0

458429 0

458430 0

458431 0

458432 0

458433 0

458434 0

458435 0

458436 0

458437 0

458438 0

458439 0

458440 0

Name: Clicks, Length: 458441, dtype: int64

In [**89**]:

% magic

IPython's 'magic' functions

===========================

The magic function system provides a series of functions which allow you to

control the behavior of IPython itself, plus a lot of system-type

features. There are two kinds of magics, line-oriented and cell-oriented.

Line magics are prefixed with the % character and work much like OS

command-line calls: they get as an argument the rest of the line, where

arguments are passed without parentheses or quotes. For example, this will

time the given statement::

%timeit range(1000)

Cell magics are prefixed with a double %%, and they are functions that get as

an argument not only the rest of the line, but also the lines below it in a

separate argument. These magics are called with two arguments: the rest of the

call line and the body of the cell, consisting of the lines below the first.

For example::

%%timeit x = numpy.random.randn((100, 100))

numpy.linalg.svd(x)

:

pandas.io.parsers.read\_csv

pandas.io.parsers.**read\_csv**(*filepath\_or\_buffer*, *sep='*, *'*, *dialect=None*, *compression=None*,*doublequote=True*, *escapechar=None*, *quotechar='"'*, *quoting=0*, *skipinitialspace=False*,*lineterminator=None*, *header='infer'*, *index\_col=None*, *names=None*, *prefix=None*, *skiprows=None*,*skipfooter=None*, *skip\_footer=0*, *na\_values=None*, *na\_fvalues=None*, *true\_values=None*,*false\_values=None*, *delimiter=None*, *converters=None*, *dtype=None*, *usecols=None*, *engine='c'*,*delim\_whitespace=False*, *as\_recarray=False*, *na\_filter=True*, *compact\_ints=False*, *use\_unsigned=False*,*low\_memory=True*, *buffer\_lines=None*, *warn\_bad\_lines=True*, *error\_bad\_lines=True*,*keep\_default\_na=True*, *thousands=None*, *comment=None*, *decimal='.'*, *parse\_dates=False*,*keep\_date\_col=False*, *dayfirst=False*, *date\_parser=None*, *memory\_map=False*, *nrows=None*,*iterator=False*, *chunksize=None*, *verbose=False*, *encoding=None*, *squeeze=False*,*mangle\_dupe\_cols=True*, *tupleize\_cols=False*, *infer\_datetime\_format=False*)

Read CSV (comma-separated) file into DataFrame

Also supports optionally iterating or breaking of the file into chunks.

|  |  |
| --- | --- |
| **Parameters :** | **filepath\_or\_buffer***: string or file handle / StringIO. The string could be*  a URL. Valid URL schemes include http, ftp, s3, and file. For file URLs, a host is expected. For instance, a local file could be file ://localhost/path/to/table.csv  **sep***: string, default ‘,’*  Delimiter to use. If sep is None, will try to automatically determine this. Regular expressions are accepted.  **lineterminator***: string (length 1), default None*  Character to break file into lines. Only valid with C parser  **quotechar***: string (length 1)*  The character used to denote the start and end of a quoted item. Quoted items can include the delimiter and it will be ignored.  **quoting***: int or csv.QUOTE\_\* instance, default None*  Control field quoting behavior per csv.QUOTE\_\* constants. Use one of QUOTE\_MINIMAL (0), QUOTE\_ALL (1), QUOTE\_NONNUMERIC (2) or QUOTE\_NONE (3). Default (None) results in QUOTE\_MINIMAL behavior.  **skipinitialspace***: boolean, default False*  Skip spaces after delimiter  **escapechar***: string*  **dtype***: Type name or dict of column -> type*  Data type for data or columns. E.g. {‘a’: np.float64, ‘b’: np.int32}  **compression***: {‘gzip’, ‘bz2’, None}, default None*  For on-the-fly decompression of on-disk data  **dialect***: string or csv.Dialect instance, default None*  If None defaults to Excel dialect. Ignored if sep longer than 1 char See csv.Dialect documentation for more details  **header***: int row number(s) to use as the column names, and the start of the*  data. Defaults to 0 if no names passed, otherwise None. Explicitly passheader=0 to be able to replace existing names. The header can be a list of integers that specify row locations for a multi-index on the columns E.g. [0,1,3]. Intervening rows that are not specified will be skipped. (E.g. 2 in this example are skipped)  **skiprows***: list-like or integer*  Row numbers to skip (0-indexed) or number of rows to skip (int) at the start of the file  **index\_col***: int or sequence or False, default None*  Column to use as the row labels of the DataFrame. If a sequence is given, a MultiIndex is used. If you have a malformed file with delimiters at the end of each line, you might consider index\_col=False to force pandas to \_not\_ use the first column as the index (row names)  **names***: array-like*  List of column names to use. If file contains no header row, then you should explicitly pass header=None  **prefix***: string or None (default)*  Prefix to add to column numbers when no header, e.g ‘X’ for X0, X1, ...  **na\_values***: list-like or dict, default None*  Additional strings to recognize as NA/NaN. If dict passed, specific per-column NA values  **true\_values***: list*  Values to consider as True  **false\_values***: list*  Values to consider as False  **keep\_default\_na***: bool, default True*  If na\_values are specified and keep\_default\_na is False the default NaN values are overridden, otherwise they’re appended to  **parse\_dates***: boolean, list of ints or names, list of lists, or dict*  If True -> try parsing the index. If [1, 2, 3] -> try parsing columns 1, 2, 3 each as a separate date column. If [[1, 3]] -> combine columns 1 and 3 and parse as a single date column. {‘foo’ : [1, 3]} -> parse columns 1, 3 as date and call result ‘foo’ A fast-path exists for iso8601-formatted dates.  **keep\_date\_col***: boolean, default False*  If True and parse\_dates specifies combining multiple columns then keep the original columns.  **date\_parser***: function*  Function to use for converting a sequence of string columns to an array of datetime instances. The default uses dateutil.parser.parser to do the conversion.  **dayfirst***: boolean, default False*  DD/MM format dates, international and European format  **thousands***: str, default None*  Thousands separator  **comment***: str, default None*  Indicates remainder of line should not be parsed Does not support line commenting (will return empty line)  **decimal***: str, default ‘.’*  Character to recognize as decimal point. E.g. use ‘,’ for European data  **nrows***: int, default None*  Number of rows of file to read. Useful for reading pieces of large files  **iterator***: boolean, default False*  Return TextFileReader object  **chunksize***: int, default None*  Return TextFileReader object for iteration  **skipfooter***: int, default 0*  Number of line at bottom of file to skip  **converters***: dict. optional*  Dict of functions for converting values in certain columns. Keys can either be integers or column labels  **verbose***: boolean, default False*  Indicate number of NA values placed in non-numeric columns  **delimiter***: string, default None*  Alternative argument name for sep. Regular expressions are accepted.  **encoding***: string, default None*  Encoding to use for UTF when reading/writing (ex. ‘utf-8’)  **squeeze***: boolean, default False*  If the parsed data only contains one column then return a Series  **na\_filter: boolean, default True**  Detect missing value markers (empty strings and the value of na\_values). In data without any NAs, passing na\_filter=False can improve the performance of reading a large file  **usecols***: array-like*  Return a subset of the columns. Results in much faster parsing time and lower memory usage.  **mangle\_dupe\_cols: boolean, default True**  Duplicate columns will be specified as ‘X.0’...’X.N’, rather than ‘X’...’X’  **tupleize\_cols: boolean, default False**  Leave a list of tuples on columns as is (default is to convert to a Multi Index on the columns)  **error\_bad\_lines: boolean, default True**  Lines with too many fields (e.g. a csv line with too many commas) will by default cause an exception to be raised, and no DataFrame will be returned. If False, then these “bad lines” will dropped from the DataFrame that is returned. (Only valid with C parser).  **warn\_bad\_lines: boolean, default True**  If error\_bad\_lines is False, and warn\_bad\_lines is True, a warning for each “bad line” will be output. (Only valid with C parser).  **infer\_datetime\_format***: boolean, default False*  If True and parse\_dates is enabled for a column, attempt to infer the datetime format to speed up the processing |
| **Returns :** | **result***: DataFrame or TextParser* |

pandas.io.parsers.**read\_csv**(http://stat.columbia.edu/~rachel/datasets/nyt1.csv

, *sep='*, *'*, *dialect=None*, *compression=None*,*doublequote=True*, *escapechar=None*, *quotechar='"'*, *quoting=0*, *skipinitialspace=False*,*lineterminator=None*, *header='infer'*, *index\_col=None*, *names=None*, *prefix=None*, *skiprows=None*,*skipfooter=None*, *skip\_footer=0*, *na\_values=None*, *na\_fvalues=None*, *true\_values=None*,*false\_values=None*, *delimiter=None*, *converters=None*, *dtype=None*, *usecols=None*, *engine='c'*,*delim\_whitespace=False*, *as\_recarray=False*, *na\_filter=True*, *compact\_ints=False*, *use\_unsigned=False*,*low\_memory=True*, *buffer\_lines=None*, *warn\_bad\_lines=True*, *error\_bad\_lines=True*,*keep\_default\_na=True*, *thousands=None*, *comment=None*, *decimal='.'*, *parse\_dates=False*,*keep\_date\_col=False*, *dayfirst=False*, *date\_parser=None*, *memory\_map=False*, *nrows=None*,*iterator=False*, *chunksize=None*, *verbose=False*, *encoding=None*, *squeeze=False*,*mangle\_dupe\_cols=True*, *tupleize\_cols=False*, *infer\_datetime\_format=False*)