BI PROJECT

data\_set.iloc[0,1].split('|||')

<https://www.shanelynn.ie/select-pandas-dataframe-rows-and-columns-using-iloc-loc-and-ix/>

## Selecting pandas data using “iloc”

The [iloc](http://pandas.pydata.org/pandas-docs/version/0.17.0/generated/pandas.DataFrame.iloc.html) indexer for Pandas Dataframe is used for [integer-location based indexing / selection](http://pandas.pydata.org/pandas-docs/stable/indexing.html#selection-by-position) by position.

The iloc indexer syntax is data.iloc[<row selection>, <column selection>], which is sure to be a source of confusion for R users. “iloc” in pandas is used to **select rows and columns by number**, in the order that they appear in the data frame. You can imagine that each row has a row number from 0 to the total rows (data.shape[0])  and iloc[] allows selections based on these numbers. The same applies for columns (ranging from 0 to data.shape[1] )

There are two “arguments” to iloc – a row selector, and a column selector.

<https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.str.split.html>

Splitting based on separator |||

types=np.unique(np.array(ds['type']))

<https://docs.scipy.org/doc/numpy/reference/generated/numpy.unique.html>

<https://www.javatpoint.com/numpy-unique>

<https://towardsdatascience.com/the-ultimate-beginners-guide-to-numpy-f5a2f99aef54>

allPost=pd.DataFrame()

<https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html>

temp3=pd.Series(temp2)

<https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.html>

allPost.shape

<https://towardsdatascience.com/the-ultimate-beginners-guide-to-numpy-f5a2f99aef54>

# How to inspect the size and shape of a NumPy array

np.shape()  
np.size()

You can get the dimensions of a NumPy array any time using **ndarray.shape**. NumPy will return the dimensions of the array as a tuple.

For example, if you create this array:

**Input:**

np\_arr = np.array([[1 , 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])

You can use **np.shape** to find the shape of your array.

**Input:**

np\_arr.shape

**Output:**

(3, 4)

This output tells you that your array has three rows and four columns.

You can find just the number of rows by specifying [0]:

**Input:**

num\_of\_rows = np\_arr.shape[0]print('Number of Rows : ', num\_of\_rows)

**Output:**

Number of Rows : 3

Or just the number of columns by specifying [1]:

**Input:**

num\_of\_columns = np\_arr.shape[1]print('Number of Columns : ', num\_of\_columns)

**Output:**

Number of Columns : 4

It’s also easy to find the total number of elements in your array:

**Input:**

print(np\_arr.shape[0] \* np\_arr.shape[1])