1. 获取数组的某列：

df = df[‘列名’]

1. 建空值的dataframe

pd.DataFrame(columns=[],index=[])

1. 读存文件

1.读取csv文件：

datafile = “\*.csv”

df = pd.read\_csv(datafile, header=0/1， names=’ \* ’, index\_col=0)

names为列名列表

header = 0, 替换原来存在的列名

index\_col 行索引

2.存为csv

df.to\_csv(‘\*.csv’, index\_col = 0)

1. 数组转列表/列表转dataframe/数组转DataFrame/ dataframe转数组

df.tolist() / pd.DataFrame((\*) / pd.DataFrame(\*) / df.values

1. 获取所有列名/行名

df.columns / df.index

1. 对某列计数

df.value\_counts()

1. 取dataframe的前几列/后几列

df.head(\*) / df.tail(\*)

1. 以dataframe生成tuple

tuple(df.iloc[i])

以dataframe生成list of tuple

Alist = []

for i in range(len(df\_\_6)):

Alist.append(tuple(df.iloc[i]))

1. dataframe的值得替换

df.loc[‘ ‘, ‘ ‘] = value

1. 选取某列值df.ix[:,’ ‘]
2. 重新生成一列(new)并赋值（list）

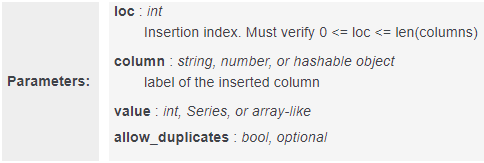
df[new] = list

1. dataframe排序

df.sort\_values(by=[“ “], inplace=True, ascending=True)

1. 插入行和列

\*.insert(self, loc, column, value, allow\_duplicates=False)



1. 给数据重新命名行列名



1. 数据映射：

<https://blog.csdn.net/weixin_37226516/article/details/64137043>

pd.merge(df1,df2, on=’ \* ‘, left\_index = True, right\_index = True)

1. 批量替换：

df.replace(to\_replace, value)

1. 重设行名(将某列设为index)

df.set\_index([‘ ‘], inplace=True)

1. 重设列名

df.columns = [ ]

df.rename(columns=[ ])

1. 去除重复项：

df. drop\_duplicates (subset=None, keep='first', inplace=False)

1. 对df的某列相同数值进行计数：

df[ \* ].value\_counts()

1. 删除含有空值的行

1. 直接删除：

df.dropna(axis=0, how='any', inplace=True)

axis：0-行操作（默认），1-列操作

how：any-只要有空值就删除（默认），all-全部为空值才删除

inplace：False-返回新的数据集（默认），True-在愿数据集上操作

2. 先索引后填充最后删除：

找出空值索引：

df[‘列名’].isnull().value\_counts()

将空值填充为999：

df[‘列名’] = df[‘列名’].fillna(‘999’)

空值所在行索引：

df[(df.target==’999’)].index.tolist()

删除空值所在行：

df = df.drop([索引])

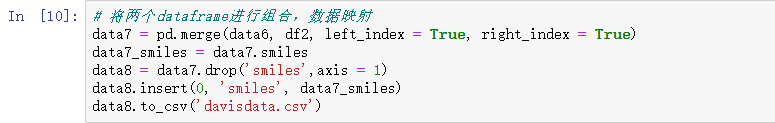
1. 将dataframe中的某列放置其他位置：

1. 先提出某列

df1 = df.drop(‘列名’, axis=1)

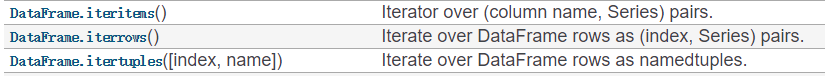
2. 后插入某位置

df1.insert(位置,’ 列名‘,列的值)



1. pandas按行遍历DataFrame

<https://blog.csdn.net/sinat_29675423/article/details/87972498>



简单对上面三种方法进行说明：

iterrows(): 按行遍历，将DataFrame的每一行迭代为(index, Series)对，可以通过row[name]对元素进行访问。

itertuples(): 按行遍历，将DataFrame的每一行迭代为元祖，可以通过row[name]对元素进行访问，比iterrows()效率高。

iteritems():按列遍历，将DataFrame的每一列迭代为(列名, Series)对，可以通过row[index]对元素进行访问。

