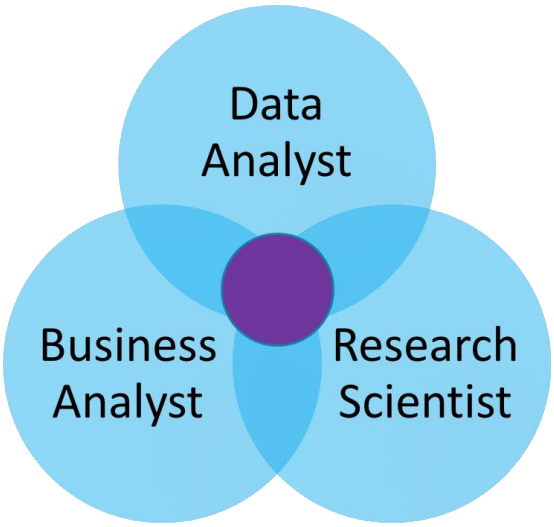
Python Data Analytics Eco-system

1 Demystify "Data Scientist"

1.1 What is a Data Scientist



Analytical skills

Design algorithm

Business insight

在前几年前，是没有 data scientist 这个职业的，当时的市场内有 data analyst，business analyst，和 research scientist 这三个职业。随时代的发展，企业发现他们需要这三方面都全能的人才，于是数据科学家就诞生了。

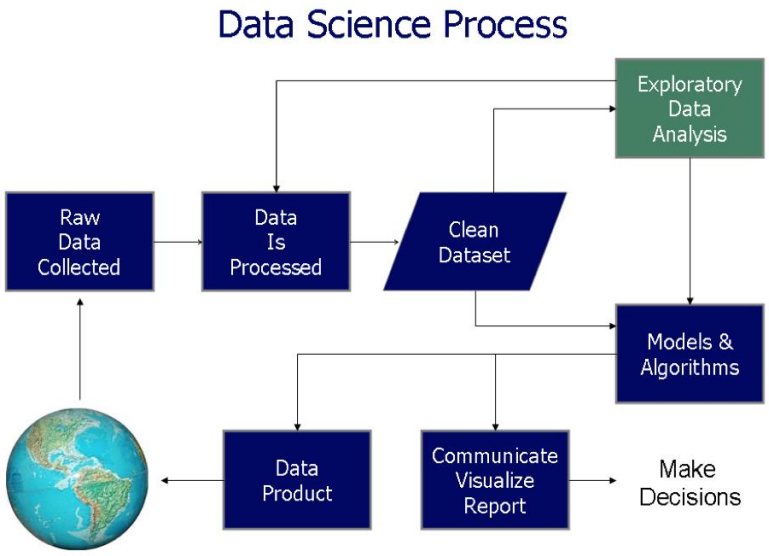
- **As a data analyst**：得到一定数据，对其进行数据挖掘（data mining），或者寻找数据模式（data pattern）；用基本的技能和基本的结构化查询语言（basic query language）去获取数据并做相关的数据处理与数据清理（data process/cleaning），并做试探性分析（exploratory analysis）。
- **As a business analyst**：从商业问题中提取数学模型，确定所需做的分析，并结合一定的商业洞察（business insights）。
    - **e.g.** 影响销售额水平的可能因素有用户数量、季节性影响以及人均购买产品数目的变化等等。
- **As a research scientist**：算法设计，如何应用高级机器学习算法解决高阶问题。考虑如何使用，以及使用哪些机器学习算法，对结果的可解释性（interpretability）进行分析。
    - e.g.: 当一个销售额下降，需要设置一个标准（baseline），根据历史数据进行预测（forecast），对真实数据与预测数据进行量化分析，并寻找差值。进而设置一个回归模型，把不同产品相对不同的变量进行回归分析，对每一种产品的销售额在各种变量的情况下进行预测。

- Data Scientist 和 Data Engineer 区别

  - DE：是软件工程师的一种，建立数据管道（data pipeline）；为 DS 建立大数据基础架构；ETL 处理（Extract Transform and Load）；为 DS 搭建报告分析的数据库。
  - DS：不涉及数据管道的建立。DS 主要应用统计学、机器学习和分析方法解决关键的商业问题。
- Data scientist 和 statistician 的区别
  - Statistician：传统统计学。通常不涉及算法的设计。
  - DS：偏向工程方面，对算法进行原型（prototype）设计，甚至进一步进行产品化。二者有很多交叉，具体要按照公司需求和职位描述。
- 数据挖掘和机器学习的区别
  - 数据挖掘：从数据中找到模式，为机器学习所使用。
  - 机器学习：比较广的范围，可以让机器学习模式（pattern），也可以是趋势（trend）或者相关性（correlation）。

1.2 Life as a Data Scientist

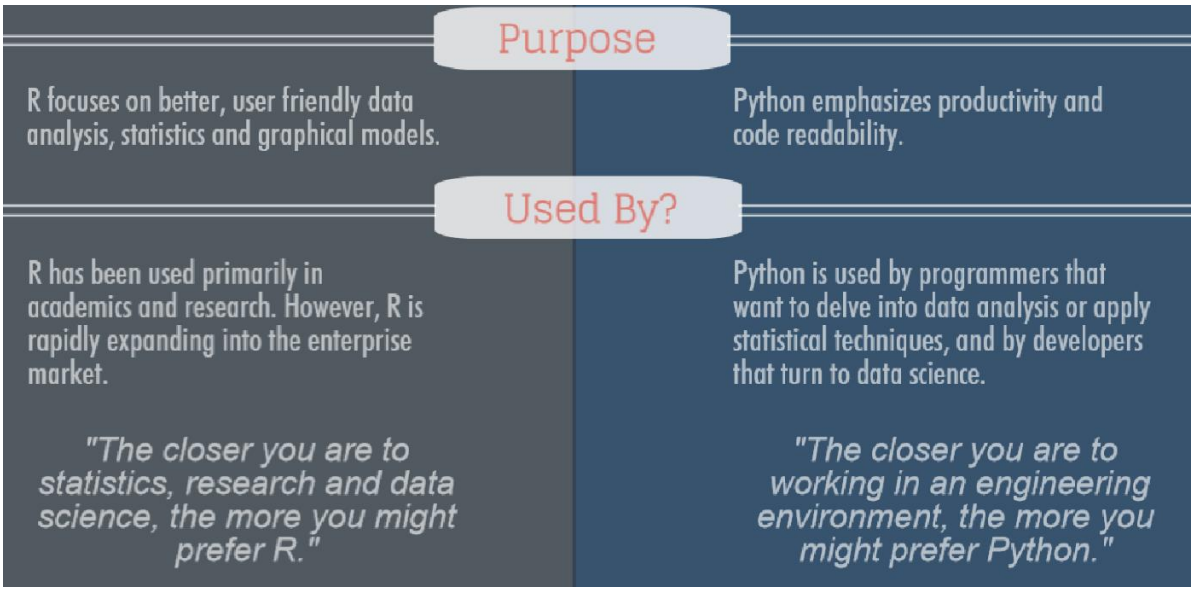
Data scientist 的工作内容比较多样，由 data engineer 对数据进行采集以及基本处理后(Raw data collected)，data scientist 会对数据进行数据清理(data process & clean)，反馈到试探性分析（相关性分析、数据分布等 Exploration），再进行建模以及算法设计(Models & Algorithms)。整体是一个较大的反馈循环过程。最终得到两种输出结果（output）：数据产品（data product，便于经营影响分析（business impact analysis），例如推荐系统、匹配系统等）以及可视性报告（visualization report，有助于执行人员作出公司决策）。



1.3 Why Python

Python 作为通用编程语言，语法和其他编程语言差别不大，便于将其他语言的模块联结在一起。Python 对于 DS 初学者来讲比较容易理解。许多公司的后台都是用 Python 写的，Python 也比较容易转化成类似 Java 等后台流行的语言。R 语言对于内存的处理不强大，当数据量较大时程序容易崩溃。R 在一些公司（比如 Google）的 quantitive analysts 中应用较多，因为其内容主要专注在统计分析以及内部咨询。

- 除了 Python 和 R，其他 DS 会使用到的软件：SAS（主要被应用在在金融领域和制药行业，且需要收费）。
- DS 对 Scala 和 Java 的要求：根据具体公司的具体要求（更多偏重在工程的方面）。
- 对 data scientist 两类 output 的要求：没有太大区别，基本两种 output 都要涉及。



2 Python Basics

2.1 Jupyter Notebook

注意：Python2 将在 2020 年停止维护，推荐学 Python3。3.6 版本有一些 package 不兼容（例如 spark2.1），可以安装 3.5.2。

基本概念

- 如何使用开发环境

- 什么是 cell（运行方便，是 code 的运行场所）
- 基本 notebook 操作

```
In [11]: a = 1
         b = 2

In [12]: a + b
Out[12]: 3

In [13]: d = 5

In [14]: a - d
Out[14]: -4
```

常用快捷键 short-cuts

- Insert below (B)
- Insert above (A)
- Delete current cell (dd)
- Shortcut help (h)
- Run current cell (Ctr + Enter)
- Run current cell, then move to the next cell (Shift + Enter)

参考资料

- Jupyter Notebook 安装使用官方教程 https://jupyter.readthedocs.io/en/latest/reference/content-reference.html
- https://www.toptal.com/python/python-3-is-it-worth-the-switch
- https://blog.appdynamics.com/engineering/the-key-differences-between-python-2-and-python-3/

2.2 Python Basics Operation

Lambda function

```
In [7]: a = 1
        b = 1.0
        print(type(a), type(b))

        <class 'int'> <class 'float'>
```

```
In [8]: def a(x):
            return x** 2
```

```
In [10]: f = lambda x: x**2
```

```
In [11]: f(10)
Out[11]: 100
```

```
In [13]: x = 20
         if x > 10:
             print(10)
         elif x < -10:
             print(-10)
         else:
             print(x)

         10
```

For Loop

```
In [14]: for i in range(10):
             print(i)

         0
         1
         2
         3
         4
         5
         6
         7
         8
         9
```

```
In [15]: # list, set, tuple, dict
         # list -- basic
         []
         # tuple -- strict (immutable)
         weekday = ('Mon', 'Tues', 'Wed')
         # dict , OrderedDict
         weekday = {'Mon': 1, 'Tues': 2}
         # set
         setweek = set(['Mon', 'Tues', 'Tues'])
```

```
In [16]: aset = list(set(alist))

         # a in aset, size does not matter much
         # a in alist, linear computation change
Out[16]: {'Mon', 'Tues'}
```

- **Tuple**：是预定义（pre-define），无法重新赋值（re-assign），主要作为 reference 使用
- **List**：可以重新赋值

3 Numpy: analytics foundation

- Numpy
- Scipy
- Datetime

```
In [17]: import numpy as np

In [18]: a1 = np.array([1, 2, 3])

In [21]: a1 * a1
Out[21]: array([1, 4, 9])

In [25]: a2 = np.random.rand(3, 3)

In [27]: a1.shape, a2.shape
Out[27]: ((3,), (3, 3))

In [28]: a2.reshape(9, 1)
Out[28]: array([[ 0.17878074],
                [ 0.10043778],
                [ 0.64510805],
                [ 0.26200209],
                [ 0.55658494],
                [ 0.6684782 ],
                [ 0.7877888 ],
                [ 0.964742  ],
                [ 0.24131666]])

In [33]: (t2 - t1).total_seconds()

Out[33]: -2610000.0

In [29]: # create a datetime
         t1 = datetime.datetime(2016,12,1, 10, 23, 2)
         t2 = datetime.datetime(2016,11,1, 5, 23, 2)
         print(type(t1), type(t2))
         <class 'datetime.datetime'> <class 'datetime.datetime'>
```

4 Pandas: data analytics

- np.array
- pd.Series
- pd.DataFrame

4.1 Series

- **Attributes**

```
In [49]: s1 = pd.Series([1, 4, 3, 5, 6], index=['A', 'B', 'C', 2, 'E'], name='Test')
         s1

Out[49]: A    1
         B    4
         C    3
         2    5
         E    6
         Name: Test, dtype: int64

In [40]: s1.values

Out[40]: array([1, 4, 3, 5, 6])

In [41]: s1.index

Out[41]: Index(['A', 'B', 'C', 'D', 'E'], dtype='object')

In [42]: s1.name

Out[42]: 'Test'

In [51]: s1
         # loc
         # iloc

Out[51]: A    1
         B    4
         C    3
         2    5
         E    6
         Name: Test, dtype: int64

In [45]: s1['C']

Out[45]: 3

In [50]: s1[2]

Out[50]: 5

In [46]: s1.loc['C']

Out[46]: 3

In [47]: s1.iloc[2]

Out[47]: 3

In [ ]: .ix # not recommended

In [52]: s1.shape, s1.size

Out[52]: ((5,), 5)
```

- **Methods & Fast Visualization**

```
In [54]: s1.loc['E'] = 16
         s1
```

```
Out[54]: A     1
         B     4
         C     3
         2     5
         E    16
         Name: Test, dtype: int64
```

```
In [55]: s1 = pd.Series(['1', '4', '3', '5', '6'], index=['A', 'B', 'C', 2, 'E'], name='Test')
```

```
In [61]: s1.astype('int') + 1
```

```
Out[61]: A    2
         B    5
         C    4
         2    6
         E    7
         Name: Test, dtype: int64
```

```
In [60]: pd.to_datetime()
```

```
Out[60]: A    1
         B    4
         C    3
         2    5
         E    6
         Name: Test, dtype: object
```

```
In [62]: # exploratory
         s1.head(2)
```

```
Out[62]: A    1
         B    4
         Name: Test, dtype: object
```

```
In [63]: s1.tail(1)
```

```
Out[63]: E    6
         Name: Test, dtype: object
```

```
In [64]: s1.describe()
```

```
Out[64]: count     5
         unique    5
         top       4
         freq      1
         Name: Test, dtype: object
```

```
In [68]: s1.head(2)
```

```
Out[68]: A    1
         B    4
         Name: Test, dtype: object
```

```
In [70]: s2 = s1.astype('int')
```

In [70]:
```
s2 = s1.astype('int')
```

In [74]:
```
s2.apply(lambda x: x ** 2 + 1)
```

Out[74]:
```
A     2
B    17
C    10
2    26
E    37
Name: Test, dtype: int64
```

In [77]:
```
s1.drop_duplicates()
```

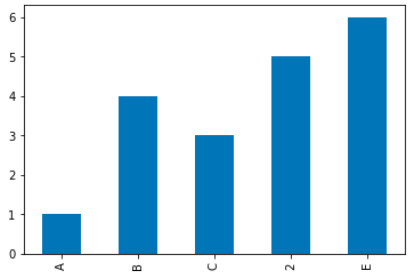
Out[77]:
```
A    1
B    4
C    3
2    5
E    6
Name: Test, dtype: object
```

In [80]:
```
s2.sort_values()
```

Out[80]:
```
A    1
C    3
B    4
2    5
E    6
Name: Test, dtype: int64
```

In [83]:
```
#想要显示图片:
%matplotlib inline
s2.plot.bar()
```

Out[83]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x118b1cd68>
```

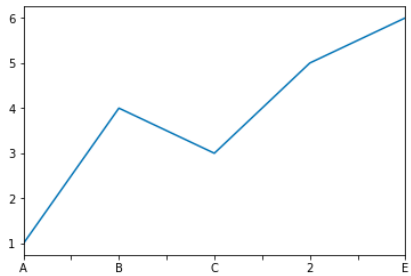


In [85]:
```
# import Pandas as pd, 报错说没有Pandas包, 可以这样下载:
! pip install pandas
```

```
Requirement already satisfied: pandas in /Users/peter/anaconda/lib/python3.6/site-packages
Requirement already satisfied: python-dateutil>=2 in /Users/peter/anaconda/lib/python3.6/site-packages (from pandas)
Requirement already satisfied: pytz>=2011k in /Users/peter/anaconda/lib/python3.6/site-packages (from pandas)
Requirement already satisfied: numpy>=1.7.0 in /Users/peter/anaconda/lib/python3.6/site-packages (from pandas)
Requirement already satisfied: six>=1.5 in /Users/peter/anaconda/lib/python3.6/site-packages (from python-dateutil>=2
->pandas)
```

In [84]:
```
s2.plot()
```

Out[84]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x118bf7fd0>
```



4.2 DataFrame

- **Attribute**

```
In [89]: df1 = pd.DataFrame([[1, 2, 3], [4, 5, 6]], index=['A', 'B'], columns=['C1', 'C2', 'C3'])
```

```
In [90]: df1
```

Out[90]:

	C1	C2	C3
A	1	2	3
B	4	5	6

```
In [91]: df1.values
```

```
Out[91]: array([[1, 2, 3],
                [4, 5, 6]])
```

```
In [92]: df1.index
```

```
Out[92]: Index(['A', 'B'], dtype='object')
```

```
In [93]: df1.columns
```

```
Out[93]: Index(['C1', 'C2', 'C3'], dtype='object')
```

```
In [94]: df1.T
```

Out[94]:

	A	B
C1	1	4
C2	2	5
C3	3	6

```
In [95]: df1.shape
```

```
Out[95]: (2, 3)
```

```
In [96]: df1.size
```

```
Out[96]: 6
```

- **Method**

```
In [99]: #查看数据最末端
         df1.tail(1)
```

Out[99]:

	C1	C2	C3
B	4	5	6

```
In [100]: df1.describe()
```

Out[100]:

	C1	C2	C3
count	2.00000	2.00000	2.00000
mean	2.50000	3.50000	4.50000
std	2.12132	2.12132	2.12132
min	1.00000	2.00000	3.00000
25%	1.75000	2.75000	3.75000
50%	2.50000	3.50000	4.50000
75%	3.25000	4.25000	5.25000
max	4.00000	5.00000	6.00000

```
In [102]: df1.loc['B']
```

```
Out[102]: C1    4
          C2    5
          C3    6
          Name: B, dtype: int64
```

In [103]:
```
df1
```

Out[103]:

	C1	C2	C3
A	1	2	3
B	4	5	6

In [104]:
```
df1.loc['B'].loc['C2']
```

Out[104]: 5

In [106]:
```
df1['C2'].loc['B']
```

Out[106]: 5

In [107]:
```
df1.loc['B', 'C2']
```

Out[107]: 5

In [109]:
```
df1.iloc[1, 1]
```

Out[109]: 5

In [112]:
```
df1 + 10 * 15
```

Out[112]:

	C1	C2	C3
A	151	152	153
B	154	155	156

In [124]:
```
#把s1转成int格式
s1.astype('int').std()
```

Out[124]: 1.9235384061671346

In [136]:
```
#我们可以把这些程序写在一起
df1.assign(C2 = lambda x: x['C2'] ** 2 + 10,
           C3 = lambda x: x['C3'] * 2 - 10) \
    .loc['A'] \
    .max()
```

Out[136]: 206

- 如果有多个 table，可以进行 df1.merge()或者 join 的处理。
- Query 里面的 command 的限制：参考 Pandas 的 API 说明。
- 仔细参阅 pandas 和 numpy 的官方教程 tutorial。

4.3 Titanic example

In [137]:
```
#load data
df = pd.read_csv('train.csv')
```

In [138]:
```
df.shape
```

Out[138]:
```
(891, 12)
```

In [141]:
```
#查看数据
! head train.csv
```

```
PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked
1,0,3,"Braund, Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S
2,1,1,"Cumings, Mrs. John Bradley (Florence Briggs Thayer)",female,38,1,0,PC 17599,71.2833,C85,C
3,1,3,"Heikkinen, Miss. Laina",female,26,0,0,STON/O2. 3101282,7.925,,S
4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",female,35,1,0,113803,53.1,C123,S
5,0,3,"Allen, Mr. William Henry",male,35,0,0,373450,8.05,,S
6,0,3,"Moran, Mr. James",male,,0,0,330877,8.4583,,Q
7,0,1,"McCarthy, Mr. Timothy J",male,54,0,0,17463,51.8625,E46,S
8,0,3,"Palsson, Master. Gosta Leonard",male,2,3,1,349909,21.075,,S
9,1,3,"Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)",female,27,0,2,347742,11.1333,,S
```

In [139]:
```
df.head(2)
```

Out[139]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C

In [140]:
```
df.tail(2)
```

Out[140]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

In [142]:
```
df.shape
```

Out[142]:
```
(891, 12)
```

In [146]:
```
#处理NuLL的数据
df['Age'] = df['Age'].round(0)
```

In [147]:
```
df.dtypes
```

Out[147]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

In [164]:
```
df2.isnull().sum()
```

Out[164]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [ ]:
```
# 处理missing value
```

In [152]:
```
df1 = df.drop('Cabin', axis=1)
```

In [160]:
```
df1['Age'] = df1['Age'].fillna(20)
```

In [163]:
```
df2 = df1[df1['Embarked'].notnull()]
```

In [167]:
```
# missing value removal
df1 = df.drop('Cabin', axis=1) \
    .assign(Age = lambda x: x['Age'].fillna(20)) \
    .loc[lambda x: x['Embarked'].notnull()]
```

In [168]:
```
df1.isnull().sum()
```

Out[168]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [ ]:
```
# Exploration (basic statistics)
```

In [170]:
```
df1.loc[10:14, ['Name', 'Sex', 'Survived']]
```

Out[170]:

	Name	Sex	Survived
10	Sandstrom, Miss. Marguerite Rut	female	1
11	Bonnell, Miss. Elizabeth	female	1
12	Saundercock, Mr. William Henry	male	0
13	Andersson, Mr. Anders Johan	male	0
14	Vestrom, Miss. Hulda Amanda Adolfina	female	0

In [172]:
```
df1.columns
```

Out[172]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Embarked'],
      dtype='object')
```

In [173]:
```
df1.pivot_table(values='PassengerId', index='Survived', columns='Sex', aggfunc='count')
```

Out[173]:

Sex	female	male
Survived		
0	81	468
1	231	109

In [175]:
```
df2 = df1.loc[lambda x: x['Survived'] == 1]
df2.shape
```

Out[175]:
```
(340, 11)
```

In [176]:
```
df3 = df1.loc[lambda x: x['Age'] > 30]
df3.shape
```

Out[176]:
```
(301, 11)
```

In [182]:
```
df4 = df2[['PassengerId', 'Name']].merge(df3[['PassengerId', 'Age']], on='PassengerId', how='outer')
df4.shape
# SQL join type
```

Out[182]:
```
(519, 3)
```

In [183]:
```
df
```

Out[183]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S