Pandas

Friday, November 6, 2020 7:16 PM

1. Complete Python Pandas Data Science Tutorial! (Reading CSV/Excel files, Sorting, Filterin
   (Reading CSV/Excel files, Sorting, Filtering, Groupby)

   - load data into Pandas
     - `df = pd.read_csv('pokemon_data.csv' , delimiter='\t')`
   - read a few rows:
     - `df.head(5)`
       读进5行
   - read a few column:
     - `df[['Name', 'Type 1', 'HP']]`
   - read each row:
     - `for index, row in df.iterrows():`
       `print(index, row)`

ng, Groupby) Complete Python Pandas Data Science Tutorial!

```
print(index, row)
```

- read a specific location
  - `df.iloc[2, 1]`

- sort data:
  - `df.sort_values(['Name', 'HP'], ascending=False)`

- drop column:
  - `df = df.drop(columns=['Total'])`

- add column:
  1. `df['Total'] = df['HP'] + df['Attack']`
     增加一个column 名字 Total 数值是响应row 的两个column HP 和 at
  2. `df['Total'] = df.iloc[:, 4:10].sum(axis=1)`
     - iloc[row, column] 在这里是所有的row, 4-9 columns
     - axis 表示 1：horizontally 0: vertically

- adjust the position of columns 把原来最后一列放到index 4的位置的方法
  - ```
    cols = list(df.columns)
    df = df[cols[0:4] + [cols[-1]] + cols[4:12]]
    ```
    注意： cols[-1]会是一个str, 而左右两边是list, 所以需要加 [ ]

- save the data/file
  - `df.t_csv('modified.csv', index=Fasle)`
    去掉自动生成的第一列index column

    - save as tsv
      `df.to_csv('modified.txt', index=False, sep='\t')`

- filtering data
  - 找到想要的符合条件的data
    `df.loc[(df['Type 1'] == 'Grass') & (df['Type 2'] == 'Poi`
    找出type 1 为grass and type 2为poison的data
    - df.loc[*conditions*]
    ```

ttack对应的值相加

去

son')]

- save the new data frame:
  ```
  new_df = df.loc[(df['Type 1'] == 'Grass') & (df['Type 2'
  new_df.to_csv('filtered.csv')
  ```

- reset index
  ```
  new_df.reset_index(drop=True, inplace=True)
  ```
  drop: 去掉原来的index, inplace: 在原来的基础上更改，不用新的varia

- 找到包含某些字符的data
  ```
  df.loc[df['Name'].str.contains('Mega')]
  ```
  ```
  df.loc[~df['Name'].str.contains('Mega')]
  ```
  不包含字符的

- 用 regular expression
  import re
  - `df.loc[df['Type 1'].str.contains('Fire|Grass', regex`
    - 注意 大小写
  - `df.loc[df['Type 1'].str.contains('fire|grass', flags`
    - flags=re.I 表示 ignore case
  - `df.loc[df['Name'].str.contains('^pi[a-z]*', flags=re`
    - ^pi[a-z]* z 表示 start with "pi" following zero or more

- conditional changes
  - `df.loc[df['Type 1']] == 'Fire', 'Legendary'] = True`
  - 把所有type 1 为 Fire的 Legendary列的data 设置为true

- change multiple columns
  ```
  df.loc[df['Total'] > 500, ['Generation', 'Legendary']] =
  ```
  把total>500的行的 generation legendary列改为 test 1, test 2

```
df.loc[df['Total'] > 500, ['Generation','Legendary']] = ['Test 1', 'Test 2']

df
```

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legend
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	F
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	F
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	Test 1	Te
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	Test 1	Te

```
] == 'Poison')]
```

able接着

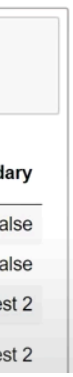
```
=True)]

=re.I, regex=True)]

.I, regex=True)]
```

letters from a to z

```
['Test 1', 'Test 2']
```



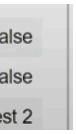
4	4	Charmander	Fire	NaN	309	39	52	43	60	50	65	1	F
5	5	Charmeleon	Fire	NaN	405	58	64	58	80	65	80	1	F
6	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	Test 1	Te

- Aggregate using groupby
  - `df.groupby(['Type 1']).mean().sort_values('Defense', asc`
    根据type 1来group, 看该组各个column的mean值并且把mean 按照
    mean 可以换成 sum , count

    `df.groupby(['Type 1']).count()['count']`
    `['Type 1']` 可以加多个groupby的参数
    最后的`['count']` 表示只显示 count列的数据

- working with large amounts of data
  - read in chunks

```
for df in pd.read_csv('modified.csv', chunksize=5):
    (operations)
```




```
ending=False)
```

Defense降序排列