C28 - pandas

author & date

- author: Akshar Patel
- date: 4/25/2022

Loading the pandas package

```
import pandas as pd
```

Series

- 1D labeled array
- A single column of a DataFrame
- Can hold any data type
- The axis labels are referred to as the index
- pd.Series( )

From a list

```
list_a = [1, 3, 5, 7, 9]
```

```
pd.Series(list_a)
```

```
0    1
1    3
2    5
3    7
4    9
dtype: int64
```

```
s1 = pd.Series([1, 3, 5, 7, 9])
```

```
s1.index #index check
```

```
RangeIndex(start=0, stop=5, step=1)
```

```
s1
```

```
s2 = pd.Series([1, 3, 5, 7, 9], index=["A", "b", "C", "d", "E"]) #if index is passed
```

```
s2
```

```
A    1
b    3
C    5
d    7
E    9
dtype: int64
```

```
s2_error = pd.Series([1, 3, 5, 7, 9], index=["A", "b",
```

Akshar Patel
6:22 PM Today
Resolve

Shows Error due to unequal index length

```
---------------------------------------------------------------------------
ValueError                                Traceb
<ipython-input-45-aee5486b3454> in <module>()
----> 1 s2_error = pd.Series([1, 3, 5, 7, 9], in
length as data

                              1 frames
/usr/local/lib/python3.7/dist-packages/pandas/co
    530     if len(data) != len(index):
    531         raise ValueError(
--> 532             "Length of values "
    533             f"({len(data)}) "
    534             "does not match length of in

ValueError: Length of values (5) does not match
```

SEARCH STACK OVERFLOW

```
# error message: Length of values (5) does not match length of index (4)
```

From a dict

```
dict_a = {"today": 1, "yesterday": 0, "tomorrow": 2}
```

```
dict_a.keys()
```

```
dict_keys(['today', 'yesterday', 'tomorrow'])
```

```
dict_a.values()
```

```
dict_values([1, 0, 2])
```

```
series_a = pd.Series(dict_a)
```

```
type(series_a)
```

```
pandas.core.series.Series
```

```
series_a
```

```
today        1
yesterday    0
tomorrow     2
dtype: int64
```

```
series_a.index
```

```
Index(['today', 'yesterday', 'tomorrow'], dtype=
```

```
series_a[0] #indexing by position value #Series is ndarray-like
```

```
1
```

```
series_a["tomorrow"] #indexing by name #Series is dict-like
```

```
2
```

Dataframe

- 2D labeled data structure with rows and cols
- Excel speadsheet
- A dict of Series objects
- pd.DataFrame( )

From dict of Series

```
dict_of_series = {
    "one": pd.Series([1.0, 2.0, "three", 4.0], index=["a", "b", "c", "d"]),
```

```
    "two": pd.Series([5, 6, "seven", 8], index=["a", "b", "c", "d"]),
}
df = pd.DataFrame(dict_of_series)
```

```
df
```

	one	two
a	1.0	5
b	2.0	6
c	three	seven
d	4.0	8

Dataframe indexing

```
df['one'] # to select a column
```

```
a      1.0
b      2.0
c    three
d      4.0
Name: one, dtype: object
```

```
df.one # to select a column
```

```
a      1.0
b      2.0
c    three
d      4.0
Name: one, dtype: object
```

```
df[0:2] #to slice rows
```

	one	two
a	1.0	5
b	2.0	6

```
df[0] # to select a single row #error
```

```
---------------------------------------------------
KeyError                                  Traceb
/usr/local/lib/python3.7/dist-packages/pandas/co
tolerance)
   3360             try:
-> 3361                 return self._engine.get_
   3362             except KeyError as err:

  ———————————————————————————— 4 frames ——
pandas/_libs/hashtable_class_helper.pxi in panda

pandas/_libs/hashtable_class_helper.pxi in panda

KeyError: 0

The above exception was the direct cause of the

KeyError                                  Traceb
/usr/local/lib/python3.7/dist-packages/pandas/co
tolerance)
   3361                 return self._engine.get_
   3362             except KeyError as err:
-> 3363                 raise KeyError(key) from
   3364
   3365         if is_scalar(key) and isna(key)
```

```
df.loc['a'] # to select a single row
```

```
one    1.0
two      5
Name: a, dtype: object
```

```
df.loc['a', "one"] # to select a value by a row and a col
```

```
1.0
```

```
df.iloc[0] # to select a single row
```

```
one    1.0
two      5
Name: a, dtype: object
```

```
df.iloc[2:] #to slice rows
```

	one	two
c	three	seven
d	4.0	8

```
df.iloc[2,1] # to select a value by a row and a col
```

```
'seven'
```

```
df.iloc[2,:] # to select values by a row and cols
```

```
one    three
two    seven
Name: c, dtype: object
```

```
df.head(2)
```

	one	two
a	1.0	5
b	2.0	6

```
df.tail(1)
```

	one	two
d	4.0	8

```
df.shape # 4 rows 2 cols # to identify a shape of `df`
```

```
(4, 2)
```

```
df.info() # to identify a basic info of `df`
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4 entries, a to d
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   one     4 non-null      object
 1   two     4 non-null      object
dtypes: object(2)
memory usage: 268.0+ bytes
```

```
df.describe() # to identify a basic stat of `df`
```

File import

.

```
payment.head()
```

freq	1.0	1

```
payment.shape #the number of rows and cols
```

```
payment.age.head()
```

```
payment.race.value_counts() #the number of `race`
```

```
payment.age.mean() #the mean of `age`
```

```
payment[["age", "score"]].median()
```

```
payment.groupby('gender').mean()
```

```
payment.groupby('gender')[["age", "score"]].mean()
```

```
payment.sort_values(by = 'score', ascending=False).head(10) #sort
```

```
payment.plot.scatter(x="age", y="payment") #scatter plot
```

```
payment['age'].plot.box() #box plot
```

```
payment[['height', 'age', 'score']].plot.box()
```