9 - Pandas-features and groupby

March 22, 2023

Table of Contents

[4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
import seaborn as sns
from numpy.random import randn
pd.set_option("display.precision", 1)
sns.set_style('white')
```

[5]:
```
df_original = pd.read_csv('https://raw.githubusercontent.com/jimcody2014/
↪python-data/main/diabetes_inspect.csv')
```

[6]:
```
df_original.head()
```

[6]:

	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	…	glipizide	glyburide	tolbutamide	miglitol	insulin	glyburide-metformin	glipizide-metformin	glimepiride-pioglitazone	diabetesMed	readmitted
0	2278392	8222157	Caucasian	Female	xyz	?	6	25	1	1	…	No	No	No	No	No	No	No	No	No	NO
1	149190	55629189	Caucasian	Female	NaN	?	1	1	7	3	…	No	No	No	No	Up	No	No	No	Yes	>30
2	64410	86047875	AfricanAmerican	female	[20-30)	?	1	1	7	2	…	Steady	No	No	No	No	No	No	No	Yes	NO
3	500364	82442376	Caucasian	Mle	[30-40)	?	1	1	7	2	…	No	No	No	No	Up	No	No	No	Yes	NO
4	16680	42519267	Caucasian	M	[40-50)	?	1	1	7	1	…	Steady	No	No	No	Steady	No	No	No	No	NO

```
4         Yes         NO

[5 rows x 33 columns]
```

1 Data Preparation

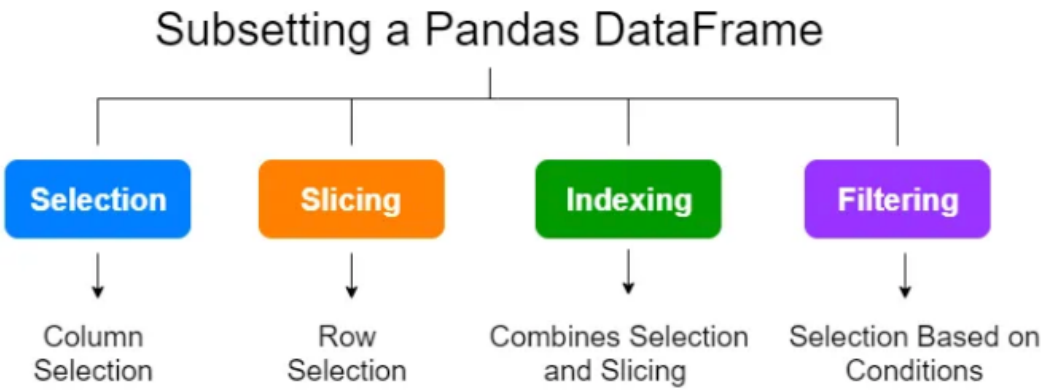
[7]:
```
df = df_original
```

[8]:
```
df.drop_duplicates(keep = 'first', inplace = True)

df['encounter_id'] = df['encounter_id'].astype(str)
df['patient_nbr'] = df['patient_nbr'].astype(str)
df['admission_type_id'] = df['admission_type_id'].astype(str)
df['discharge_disposition_id'] = df['discharge_disposition_id'].astype(str)
df['admission_source_id'] = df['admission_source_id'].astype(str)

short_names = {'admission_type_id':'admin_type', # creating a dict of the names␣
 ↪to be changed
               'discharge_disposition_id':'discharge_dispo',
               'admission_source_id':'admin_source',
               'num_lab_procedures':'lab_procedures',
               'num_procedures':'procedures'}

df.rename(columns=short_names, inplace=True)
no_age = df[df['age'].isnull()].index
drop_columns = {'medical_specialty','glyburide-metformin','glipizide-metformin',
                'glimepiride-pioglitazone', 'payer_code', 'weight'}
df = df.drop(columns = drop_columns)

df.num_medications.fillna( df.num_medications.mean(),inplace=True )
df['gender'] = df['gender'].replace({'M':'Male', 'Mle':'Male', 'F':'Female'})
df['gender'] = df['gender'].apply(lambda x:x.lower())
df['gender'] = df['gender'].replace({'?':'male', 'unknown/invalid':'male'})
df = df.loc[df['age']!= 'xyz']
df = df.loc[df.gender != '?']
df = df.drop(no_age, axis = 0)
df2 = df                          # This is used for new dataframe examples
```

[9]:
```
data = {
    'Weather': ['Sunny','Sunny','Sunny','Cloudy','Shower','Shower','Sunny'],
    'Temperature': [78,76,78,68,70,71,82],
    'Wind': [13,28,16,11,26,27,20],
    'Humidity': [30,96,20,22,79,62,10],
}
weather = pd.DataFrame(data, index = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat',␣
 ↪'Sun'])
```

2 Feature Engineering

- New dataframes
  - a subset of columns
  - a subset of rows
- New columns



2.1 New dataframes

[10]:
```
df.columns
```

[10]:
```
Index(['encounter_id', 'patient_nbr', 'race', 'gender', 'age', 'admin_type',
       'discharge_dispo', 'admin_source', 'time_in_hospital', 'lab_procedures',
       'procedures', 'num_medications', 'number_outpatient',
       'number_emergency', 'number_inpatient', 'diag_1', 'max_glu_serum',
       'A1Cresult', 'metformin', 'glimepiride', 'glipizide', 'glyburide',
       'tolbutamide', 'miglitol', 'insulin', 'diabetesMed', 'readmitted'],
      dtype='object')
```

Side note - Series One row of a dataframe or one columns of a dataframe is referred to as a Series. A Series is like a list or an array.

[11]:
```
n1 = df.iloc[2]
n2 = df.insulin
print(type(n1))
print(type(n2))
```

```
<class 'pandas.core.series.Series'>
<class 'pandas.core.series.Series'>
```

2.1.1 New dataframe based on columns (selecting)

Select using column names

[12]:
```
medications = df[['miglitol', 'insulin', 'glipizide']] # Notice that the␣
  ↪columns are passed in as a list
print(type(medications))
print()
print(medications.info())
```

```
<class 'pandas.core.frame.DataFrame'>

<class 'pandas.core.frame.DataFrame'>
Int64Index: 101764 entries, 2 to 101765
Data columns (total 3 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   miglitol   101764 non-null  object
 1   insulin    101764 non-null  object
 2   glipizide  101764 non-null  object
dtypes: object(3)
memory usage: 3.1+ MB
None
```

Select using .loc[ ]

[13]:
```
medications2 = df.loc[:,['insulin', 'miglitol']] # Notice that the columns are␣
  ↪passed in as a list
print(type(medications2))
print()
print(medications2.info())
```

```
<class 'pandas.core.frame.DataFrame'>

<class 'pandas.core.frame.DataFrame'>
Int64Index: 101764 entries, 2 to 101765
Data columns (total 2 columns):
 #   Column    Non-Null Count   Dtype
---  ------    --------------   -----
 0   insulin   101764 non-null  object
 1   miglitol  101764 non-null  object
dtypes: object(2)
memory usage: 2.3+ MB
None
```

Select using .iloc[ ]

[14]:
```
medications3 = df.iloc[:,[23,24]] # Notice that the columns are passed in as a␣
  ↪list
print(type(medications3))
print()
print(medications3.info())
```

```
<class 'pandas.core.frame.DataFrame'>

<class 'pandas.core.frame.DataFrame'>
Int64Index: 101764 entries, 2 to 101765
Data columns (total 2 columns):
 #   Column    Non-Null Count   Dtype
---  ------    --------------   -----
 0   miglitol  101764 non-null  object
 1   insulin   101764 non-null  object
dtypes: object(2)
memory usage: 2.3+ MB
None
```

[15]:
```
medications4 = df.iloc[:,18:25] # Notice that the columns are NOT passed in as a list (because of slicing (the colon))
print(type(medications4))
print()
print(medications4.info())
```

```
<class 'pandas.core.frame.DataFrame'>

<class 'pandas.core.frame.DataFrame'>
Int64Index: 101764 entries, 2 to 101765
Data columns (total 7 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   metformin    101764 non-null  object
 1   glimepiride  101764 non-null  object
 2   glipizide    101764 non-null  object
 3   glyburide    101764 non-null  object
 4   tolbutamide  101764 non-null  object
 5   miglitol     101764 non-null  object
 6   insulin      101764 non-null  object
dtypes: object(7)
memory usage: 6.2+ MB
None
```

[16]:
```
del medications
del medications2
del medications3
del medications4
```

2.1.2 New dataframe based on rows (slicing)

[17]:
```
weather.head(10)
```

[17]:

	Weather	Temperature	Wind	Humidity
Mon	Sunny	78	13	30
Tue	Sunny	76	28	96
Wed	Sunny	78	16	20
Thu	Cloudy	68	11	22
Fri	Shower	70	26	79
Sat	Shower	71	27	62
Sun	Sunny	82	20	10

When a column is used as the row index

[18]:
```
rows1 = weather.loc[['Tue', 'Thu', 'Sun'],]
rows1
```

[18]:

	Weather	Temperature	Wind	Humidity
Tue	Sunny	76	28	96
Thu	Cloudy	68	11	22
Sun	Sunny	82	20	10

Slice using iloc[ ]

[19]:
```
rows2 = df.iloc[[2,4,6],]  # Notice, you do not need the colon
print(type(rows2))
print()
print(rows2.shape)
```

```
<class 'pandas.core.frame.DataFrame'>

(3, 27)
```

[20]:
```
rows3 = df.iloc[3:20,]  # Notice, you do not need the colon
print(type(rows3))
print()
print(rows3.shape)
```

```
<class 'pandas.core.frame.DataFrame'>

(17, 27)
```

2.1.3 New dataframe based on rows and columns (indexing)

[21]:
```
rows4 = df.iloc[3:20,5:10]
print(type(rows4))
print()
print(rows4.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
(17, 5)
```

[22]:
```
rows5 = df.loc[3:20,['insulin','miglitol']]
print(type(rows5))
print()
print(rows5.shape)
```

```
<class 'pandas.core.frame.DataFrame'>

(18, 2)
```

2.1.4 New dataframe based on row filtering

[23]:
```
df['A1Cresult'] == 'None'
```

[23]:
```
2          True
3          True
4          True
5          True
6          True
          …
101761    False
101762     True
101763     True
101764     True
101765     True
Name: A1Cresult, Length: 101764, dtype: bool
```

[24]:
```
NoA1C = df[df['A1Cresult'] == 'None'] # If True, put into new dataframe
print(type(NoA1C))
print()
print(NoA1C.shape)
```

```
<class 'pandas.core.frame.DataFrame'>

(84746, 27)
```

[25]:
```
NoA1C.head()
```

[25]:

	encounter_id	patient_nbr	race	gender	age	admin_type	…
2	64410	86047875	AfricanAmerican	female	[20-30)	1	
3	500364	82442376	Caucasian	male	[30-40)	1	
4	16680	42519267	Caucasian	male	[40-50)	1	
5	35754	82637451	Caucasian	male	[50-60)	2	
6	55842	84259809	Caucasian	male	[60-70)	3	

	discharge_dispo	admin_source	time_in_hospital	lab_procedures	…
2	1	7	2	11	…

```
3                1            7                 2              44  …
4                1            7                 1              51  …
5                1            2                 3              31  …
6                1            2                 4              70  …

   A1Cresult  metformin  glimepiride  glipizide  glyburide tolbutamide  \
2       None         No           No     Steady         No          No
3       None         No           No         No         No          No
4       None         No           No     Steady         No          No
5       None         No           No         No         No          No
6       None     Steady       Steady         No         No          No

  miglitol insulin diabetesMed readmitted
2       No      No         Yes         NO
3       No      Up         Yes         NO
4       No  Steady         Yes         NO
5       No  Steady         Yes        >30
6       No  Steady         Yes         NO

[5 rows x 27 columns]
```

[26]:
```
NoA1C2 = df[(df['A1Cresult'] == 'None') & (df['time_in_hospital'] > 4)] #␣
 ↪Notice parens around test
print(type(NoA1C2))
print()
print(NoA1C2.shape)
```

```
<class 'pandas.core.frame.DataFrame'>

(31145, 27)
```

[27]:
```
NoA1C3 = df[(df['A1Cresult'] == 'None') | (df['time_in_hospital'] > 4)] #␣
 ↪Notice parens around test
print(type(NoA1C3))
print()
print(NoA1C3.shape)
```

```
<class 'pandas.core.frame.DataFrame'>

(92255, 27)
```

[28]:
```
NoA1C4 = df[df['lab_procedures'].between(11,25)]
print(type(NoA1C4))
print()
print(NoA1C4.shape)
print()
print(NoA1C4.head())
```

```
<class 'pandas.core.frame.DataFrame'>

(10151, 27)

   encounter_id patient_nbr             race  gender      age admin_type  \
2         64410    86047875  AfricanAmerican  female  [20-30)          1
24       216156    62718876  AfricanAmerican  female  [70-80)          3
27       248916   115196778        Caucasian  female  [50-60)          1
46       486156    86240259        Caucasian  female  [70-80)          3
67       792402    83775519        Caucasian  female  [80-90)          2

   discharge_dispo admin_source  time_in_hospital  lab_procedures  …  \
2                1            7                 2              11  …
24               1            2                 3              19  …
27               1            1                 2              25  …
46               5            4                 9              25  …
67               1            4                 2              25  …

    A1Cresult  metformin  glimepiride  glipizide  glyburide tolbutamide  \
2        None         No           No     Steady         No          No
24       None         No           No     Steady         No          No
27       None         No           No         No         No          No
46       None         No           No         No     Steady          No
67       None         No           No         No         No          No

   miglitol insulin diabetesMed readmitted
2        No      No         Yes         NO
24       No  Steady         Yes         NO
27       No  Steady         Yes        >30
46       No    Down         Yes        <30
67       No  Steady         Yes         NO

[5 rows x 27 columns]
```

3 New Columns

[29]:
```
df['new1'] = df.procedures+1
```

[30]:
```
df.head()
```

[30]:
```
   encounter_id patient_nbr             race  gender      age admin_type  \
2         64410    86047875  AfricanAmerican  female  [20-30)          1
3        500364    82442376        Caucasian    male  [30-40)          1
4         16680    42519267        Caucasian    male  [40-50)          1
5         35754    82637451        Caucasian    male  [50-60)          2
6         55842    84259809        Caucasian    male  [60-70)          3
```

```
  discharge_dispo admin_source  time_in_hospital  lab_procedures  …  \
2               1            7                 2              11  …
3               1            7                 2              44  …
4               1            7                 1              51  …
5               1            2                 3              31  …
6               1            2                 4              70  …

   metformin  glimepiride  glipizide  glyburide  tolbutamide miglitol insulin  \
2         No           No     Steady         No           No       No      No
3         No           No         No         No           No       No      Up
4         No           No     Steady         No           No       No  Steady
5         No           No         No         No           No       No  Steady
6     Steady       Steady         No         No           No       No  Steady

  diabetesMed readmitted new1
2         Yes         NO    6
3         Yes         NO    2
4         Yes         NO    1
5         Yes        >30    7
6         Yes         NO    2

[5 rows x 28 columns]
```

apply() is a dataframe method that replaces loops. It takes a function as input and applies it to all rows of the dataframe.

[31]:
```
df['new2'] = df['gender'].apply(lambda x:x.upper())
```

[32]:
```
df['new3'] = df['procedures'].apply(lambda x:x*2)
```

[33]:
```
# The Row object is a read-only dictionary-like structure which contains the␣
 ↪cell values for a particular row.
def do_math(row):
    return row['procedures'] + row['lab_procedures']
```

[34]:
```
df['new4'] = df.apply(do_math, axis=1)
```

[35]:
```
df['new5'] = df.procedures + df.lab_procedures
```

[36]:
```
df.head()
```

[36]:
```
  encounter_id patient_nbr             race  gender      age admin_type  \
2        64410    86047875  AfricanAmerican  female  [20-30)          1
3       500364    82442376        Caucasian    male  [30-40)          1
4        16680    42519267        Caucasian    male  [40-50)          1
5        35754    82637451        Caucasian    male  [50-60)          2
6        55842    84259809        Caucasian    male  [60-70)          3
```

```
  discharge_dispo admin_source  time_in_hospital  lab_procedures  …  \
2               1            7                 2              11  …
3               1            7                 2              44  …
4               1            7                 1              51  …
5               1            2                 3              31  …
6               1            2                 4              70  …

   tolbutamide  miglitol  insulin  diabetesMed  readmitted new1    new2 new3  \
2           No        No       No          Yes          NO    6  FEMALE   10
3           No        No       Up          Yes          NO    2    MALE    2
4           No        No   Steady          Yes          NO    1    MALE    0
5           No        No   Steady          Yes         >30    7    MALE   12
6           No        No   Steady          Yes          NO    2    MALE    2

  new4 new5
2   16   16
3   45   45
4   51   51
5   37   37
6   71   71

[5 rows x 32 columns]
```

[37]:
```
# Sample of a user defined function being used

# def clean_text_round1(text):
#     text = text.lower()
#     text = re.sub('\[.*?\]', '', text)
#     text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
#     text = re.sub('\w*\d\w*', '', text)
#     text = re.sub('[‘’""…]', '', text)
#     text = re.sub('\n', '', text)
#     return text

# round1 = lambda x: clean_text_round1(x)

# latuda.review = pd.DataFrame(latuda.review.apply(round1))
```

[ ]:
```

```

4 Group By

4.1 Basics

Information we might be interested in: - For each gender, what is the mean for every numeric column? What are they by age group? - For each gender, what is the mean value of procedures? -

How many rows are there in each group when the group is gender? - By gender, what are the min, max and median values for number of medications(individualy)?

[38]:
```
df.columns
```

[38]:
```
Index(['encounter_id', 'patient_nbr', 'race', 'gender', 'age', 'admin_type',
       'discharge_dispo', 'admin_source', 'time_in_hospital', 'lab_procedures',
       'procedures', 'num_medications', 'number_outpatient',
       'number_emergency', 'number_inpatient', 'diag_1', 'max_glu_serum',
       'A1Cresult', 'metformin', 'glimepiride', 'glipizide', 'glyburide',
       'tolbutamide', 'miglitol', 'insulin', 'diabetesMed', 'readmitted',
       'new1', 'new2', 'new3', 'new4', 'new5'],
      dtype='object')
```

[39]:
```
df.groupby('gender').mean()
```

[39]:

gender	time_in_hospital	lab_procedures	procedures	num_medications	number_outpatient	number_emergency	number_inpatient	new1	new3	new4	new5
female	4.5	43.2	1.2	16.2	0.4	0.2	0.7	2.2	2.5	44.4	44.4
male	4.3	43.0	1.4	15.8	0.4	0.2	0.6	2.4	2.9	44.5	44.5

[40]:
```
df.groupby('gender').procedures.mean() # Requesting the mean of one column
```

[40]:
```
gender
female    1.2
male      1.4
Name: procedures, dtype: float64
```

[41]:
```
df.groupby('gender').size()  # .size() provides # of rows in each group
```

[41]:
```
gender
female    54706
male      47058
dtype: int64
```

[42]:
```
df.groupby('gender').num_medications.min()
```

[42]:
```
gender
female    1.0
male      1.0
Name: num_medications, dtype: float64
```

[43]:
```
df.groupby('gender').num_medications.max()
```

[43]:
```
gender
female    75.0
male      81.0
Name: num_medications, dtype: float64
```

[44]:
```
df.groupby('gender').num_medications.median()
```

[44]:
```
gender
female    15.0
male      14.0
Name: num_medications, dtype: float64
```

[45]:
```
df.groupby('gender').num_medications.count()
```

[45]:
```
gender
female    54706
male      47058
Name: num_medications, dtype: int64
```

[46]:
```
# 1 - df.groupby('gender').mean()                          mean for all columns
# 2 - df.groupby('gender').num_medications.min()           mean for a specific column
# 3 -            NEXT                                      multiple agregations for␣
 ↪a column(s)

# there is a function called .agg() and it allows specifiying multiple␣
 ↪aggregation functions at once

x = df.groupby('gender').procedures.agg(['max', 'min', 'count', 'median',␣
 ↪'mean'])
x
```

[46]:

gender	max	min	count	median	mean
female	6	0	54706	1.0	1.2
male	6	0	47058	1.0	1.4

[47]:
```
# with custom column name
df.groupby('gender').procedures.agg(
    most=('max'),
    least=('min'),
```

```
)
```

[47]:
```
        most  least
gender
female     6      0
male       6      0
```

[48]:
```
df.columns
```

[48]:
```
Index(['encounter_id', 'patient_nbr', 'race', 'gender', 'age', 'admin_type',
       'discharge_dispo', 'admin_source', 'time_in_hospital', 'lab_procedures',
       'procedures', 'num_medications', 'number_outpatient',
       'number_emergency', 'number_inpatient', 'diag_1', 'max_glu_serum',
       'A1Cresult', 'metformin', 'glimepiride', 'glipizide', 'glyburide',
       'tolbutamide', 'miglitol', 'insulin', 'diabetesMed', 'readmitted',
       'new1', 'new2', 'new3', 'new4', 'new5'],
      dtype='object')
```

4.2 Create a new dataframe for grouped data

[49]:
```
x = df.groupby('A1Cresult')  #### Notice no method on the end
x.groups

GT7 = x.get_group('>7')
GT7.head()
```

[49]:
```
     encounter_id patient_nbr       race  gender      age admin_type  \
26         236316    40523301  Caucasian    male  [80-90)          1
74         955884    93196251  Caucasian  female  [70-80)          1
117       1968528      720936  Caucasian  female  [70-80)          6
148       2371176      966042  Caucasian  female  [50-60)          6
203       2664138     8432703  Caucasian  female  [60-70)          6

     discharge_dispo admin_source  time_in_hospital  lab_procedures  …  \
26                 3            7                 6              64  …
74                 3            7                 5              34  …
117               25            1                10              56  …
148               25            7                 3              18  …
203               25            1                 4              45  …

     tolbutamide  miglitol  insulin  diabetesMed  readmitted new1    new2  \
26            No        No       No          Yes          NO    4    MALE
74            No        No       Up          Yes         >30    1  FEMALE
117           No        No     Down          Yes         >30    3  FEMALE
148           No        No   Steady          Yes          NO    4  FEMALE
203           No        No       No          Yes         >30    1  FEMALE
```

```
    new3 new4 new5
26     6   67   67
74     0   34   34
117    4   58   58
148    6   21   21
203    0   45   45

[5 rows x 32 columns]
```

[50]:
```
GT7 = x.get_group('>7')
GT7.head()
```

[50]:
```
    encounter_id patient_nbr       race  gender      age admin_type  \
26        236316    40523301  Caucasian    male  [80-90)          1
74        955884    93196251  Caucasian  female  [70-80)          1
117      1968528      720936  Caucasian  female  [70-80)          6
148      2371176      966042  Caucasian  female  [50-60)          6
203      2664138     8432703  Caucasian  female  [60-70)          6

    discharge_dispo admin_source  time_in_hospital  lab_procedures  …  \
26                3            7                 6              64  …
74                3            7                 5              34  …
117              25            1                10              56  …
148              25            7                 3              18  …
203              25            1                 4              45  …

     tolbutamide  miglitol  insulin  diabetesMed  readmitted new1    new2  \
26            No        No       No          Yes          NO    4    MALE
74            No        No       Up          Yes         >30    1  FEMALE
117           No        No     Down          Yes         >30    3  FEMALE
148           No        No   Steady          Yes          NO    4  FEMALE
203           No        No       No          Yes         >30    1  FEMALE

    new3 new4 new5
26     6   67   67
74     0   34   34
117    4   58   58
148    6   21   21
203    0   45   45

[5 rows x 32 columns]
```

[51]:
```
x.first()
```

[51]:
```
          encounter_id patient_nbr             race  gender      age  \
A1Cresult
>7              236316    40523301        Caucasian    male  [80-90)
```

```
>8             1257282    84488562            Other  female  [50-60)
None             64410    86047875  AfricanAmerican  female  [20-30)
Norm           1270524    67897251        Caucasian    male  [60-70)

           admin_type discharge_dispo admin_source  time_in_hospital  \
A1Cresult
>7                  1              3            7                 6
>8                  1              1            7                 2
None                1              1            7                 2
Norm                1              2            7                 1

            lab_procedures  …  tolbutamide  miglitol  insulin  diabetesMed  \
A1Cresult                   …
>7                      64  …           No        No       No          Yes
>8                      53  …           No        No       Up          Yes
None                    11  …           No        No       No          Yes
Norm                    59  …           No        No   Steady          Yes

            readmitted new1    new2 new3 new4 new5
A1Cresult
>7                  NO    4    MALE    6   67   67
>8                  NO    1  FEMALE    0   53   53
None                NO    6  FEMALE   10   16   16
Norm                NO    1    MALE    0   59   59

[4 rows x 31 columns]
```

[52]:
```
x.last()
```

[52]:
```
           encounter_id patient_nbr             race  gender      age  \
A1Cresult
>7            443842016   183087545        Caucasian  female  [70-80)
>8            443847548   100162476  AfricanAmerican    male  [70-80)
None          443867222   175429310        Caucasian    male  [70-80)
Norm          443835140   175326800        Caucasian    male  [70-80)

           admin_type discharge_dispo admin_source  time_in_hospital  \
A1Cresult
>7                  1              1            7                 9
>8                  1              3            7                 3
None                1              1            7                 6
Norm                3              6            1                13

            lab_procedures  …  tolbutamide  miglitol  insulin  diabetesMed  \
A1Cresult                   …
>7                      50  …           No        No   Steady          Yes
>8                      51  …           No        No     Down          Yes
```

```
None                   13  …           No        No       No           No
Norm                   77  …           No        No       Up          Yes

           readmitted new1    new2 new3 new4 new5
A1Cresult
>7               >30    3  FEMALE    4   52   52
>8               >30    1    MALE    0   51   51
None              NO    4    MALE    6   16   16
Norm              NO    7    MALE   12   83   83

[4 rows x 31 columns]
```

[53]:
```
print(type(x))
print(type(GT7))
```

```
<class 'pandas.core.groupby.generic.DataFrameGroupBy'>
<class 'pandas.core.frame.DataFrame'>
```

[54]:
```
x.mean()
```

[54]:
```
           time_in_hospital  lab_procedures  procedures  num_medications  \
A1Cresult
>7                      4.9            53.4         1.3             16.8
>8                      4.7            54.9         1.3             16.1
None                    4.3            40.8         1.3             15.9
Norm                    4.9            54.2         1.3             16.5

           number_outpatient  number_emergency  number_inpatient  new1  new3  \
A1Cresult
>7                       0.3               0.1               0.4   2.3   2.6
>8                       0.3               0.2               0.5   2.3   2.6
None                     0.4               0.2               0.7   2.3   2.7
Norm                     0.3               0.2               0.4   2.3   2.5

           new4  new5
A1Cresult
>7         54.7  54.7
>8         56.2  56.2
None       42.2  42.2
Norm       55.4  55.4
```

5 Using the penguin dataset

It has a wider range of values

[55]:
```
df = sns.load_dataset('penguins')
```

[56]:
```
# with custom column name
df.groupby('sex').body_mass_g.agg(
    sex_max=('max'),
    sex_min=('min'),
)
```

[56]:
```
        sex_max  sex_min
sex
Female   5200.0   2700.0
Male     6300.0   3250.0
```

[57]:
```
# Custom aggregation function
def categorize(x):
    m = x.mean()
    return True if m > 4000 else False

df.groupby('sex').body_mass_g.agg(['max', 'mean', categorize])
```

[57]:
```
           max    mean  categorize
sex
Female  5200.0  3862.3       False
Male    6300.0  4545.7        True
```

[58]:
```
# Use lambda
df.groupby('sex').body_mass_g.agg(
    ['max', 'mean', lambda x: True if x.mean() > 4000 else False]
)
```

[58]:
```
           max    mean  <lambda_0>
sex
Female  5200.0  3862.3       False
Male    6300.0  4545.7        True
```

[59]:
```
# REMINDER
# With a groupby, a specific column for the aggregation does not have to be␣
 ↪specified.
# Without a column, it will perform the aggregation across all of the numeric␣
 ↪columns

df.groupby('sex').mean()
```

[59]:
```
        bill_length_mm  bill_depth_mm  flipper_length_mm  body_mass_g
sex
Female            42.1           16.4              197.4       3862.3
Male              45.9           17.9              204.5       4545.7
```

[ ]:
```
df.groupby('sex').agg(['mean', 'median'])
```

6 Transforming Data

[61]:
```
# A lambda expression for Standardization.
standardization = lambda x: (x - x.mean()) / x.std()
```

[62]:
```
df.groupby('sex').body_mass_g.transform(standardization)
```

[62]:
```
0     -1.0e+00
1     -9.3e-02
2     -9.2e-01
4     -6.2e-01
5     -1.1e+00
        …
338    1.6e+00
340    1.5e+00
341    1.5e+00
342    2.0e+00
343    1.1e+00
Name: body_mass_g, Length: 333, dtype: float64
```

[63]:
```
df.groupby('sex').body_mass_g.apply(standardization)
```

[63]:
```
0     -1.0e+00
1     -9.3e-02
2     -9.2e-01
4     -6.2e-01
5     -1.1e+00
        …
338    1.6e+00
340    1.5e+00
341    1.5e+00
342    2.0e+00
343    1.1e+00
Name: body_mass_g, Length: 333, dtype: float64
```

7 Filtering data

[64]:
```
# How many rows fall into each island group?
df.groupby('island').size()
```

[64]:
```
island
Biscoe       168
Dream        124
Torgersen     52
dtype: int64
```

[65]:
```
# filter data to return all islands that have at least 100 observations.
df.groupby('island').filter(lambda x: len(x) >= 100)
```

[65]:
```
     species  island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
20    Adelie  Biscoe            37.8           18.3              174.0
21    Adelie  Biscoe            37.7           18.7              180.0
22    Adelie  Biscoe            35.9           19.2              189.0
23    Adelie  Biscoe            38.2           18.1              185.0
24    Adelie  Biscoe            38.8           17.2              180.0
..       …       …               …              …                  …
339   Gentoo  Biscoe             NaN            NaN                NaN
340   Gentoo  Biscoe            46.8           14.3              215.0
341   Gentoo  Biscoe            50.4           15.7              222.0
342   Gentoo  Biscoe            45.2           14.8              212.0
343   Gentoo  Biscoe            49.9           16.1              213.0

     body_mass_g     sex
20        3400.0  Female
21        3600.0    Male
22        3800.0  Female
23        3950.0    Male
24        3800.0    Male
..           …     …
339          NaN     NaN
340       4850.0  Female
341       5750.0    Male
342       5200.0  Female
343       5400.0    Male

[292 rows x 7 columns]
```

8 Group by multiple categories

[66]:
```
# Creating a df that is a subset of penguins

small = df.loc[:, ['species', 'island', 'bill_depth_mm', 'bill_length_mm']]
small
```

[66]:
```
     species     island  bill_depth_mm  bill_length_mm
0     Adelie  Torgersen           18.7            39.1
1     Adelie  Torgersen           17.4            39.5
2     Adelie  Torgersen           18.0            40.3
3     Adelie  Torgersen            NaN             NaN
4     Adelie  Torgersen           19.3            36.7
..       …        …              …               …
339   Gentoo     Biscoe            NaN             NaN
```

```
340  Gentoo     Biscoe           14.3            46.8
341  Gentoo     Biscoe           15.7            50.4
342  Gentoo     Biscoe           14.8            45.2
343  Gentoo     Biscoe           16.1            49.9

[344 rows x 4 columns]
```

[67]:
```
# Grouping by multiple categories

small.groupby(['species', 'island']).mean()
```

[67]:
```
                    bill_depth_mm  bill_length_mm
species   island
Adelie    Biscoe             18.4            39.0
          Dream              18.3            38.5
          Torgersen          18.4            39.0
Chinstrap Dream              18.4            48.8
Gentoo    Biscoe             15.0            47.5
```

[68]:
```
df.groupby(['species', 'island']).mean()
```

[68]:
```
                    bill_length_mm  bill_depth_mm  flipper_length_mm  \
species   island
Adelie    Biscoe              39.0           18.4              188.8
          Dream               38.5           18.3              189.7
          Torgersen           39.0           18.4              191.2
Chinstrap Dream               48.8           18.4              195.8
Gentoo    Biscoe              47.5           15.0              217.2

                    body_mass_g
species   island
Adelie    Biscoe         3709.7
          Dream          3688.4
          Torgersen      3706.4
Chinstrap Dream          3733.1
Gentoo    Biscoe         5076.0
```

[69]:
```
# Group by multi column
df_groupby_multi = small.groupby(['species', 'island']).mean()
df_groupby_multi
```

[69]:
```
                    bill_depth_mm  bill_length_mm
species   island
Adelie    Biscoe             18.4            39.0
          Dream              18.3            38.5
          Torgersen          18.4            39.0
Chinstrap Dream              18.4            48.8
```

```
Gentoo    Biscoe              15.0              47.5
```

[70]:
```
df_groupby_multi.reset_index()
```

[70]:

	species	island	bill_depth_mm	bill_length_mm
0	Adelie	Biscoe	18.4	39.0
1	Adelie	Dream	18.3	38.5
2	Adelie	Torgersen	18.4	39.0
3	Chinstrap	Dream	18.4	48.8
4	Gentoo	Biscoe	15.0	47.5

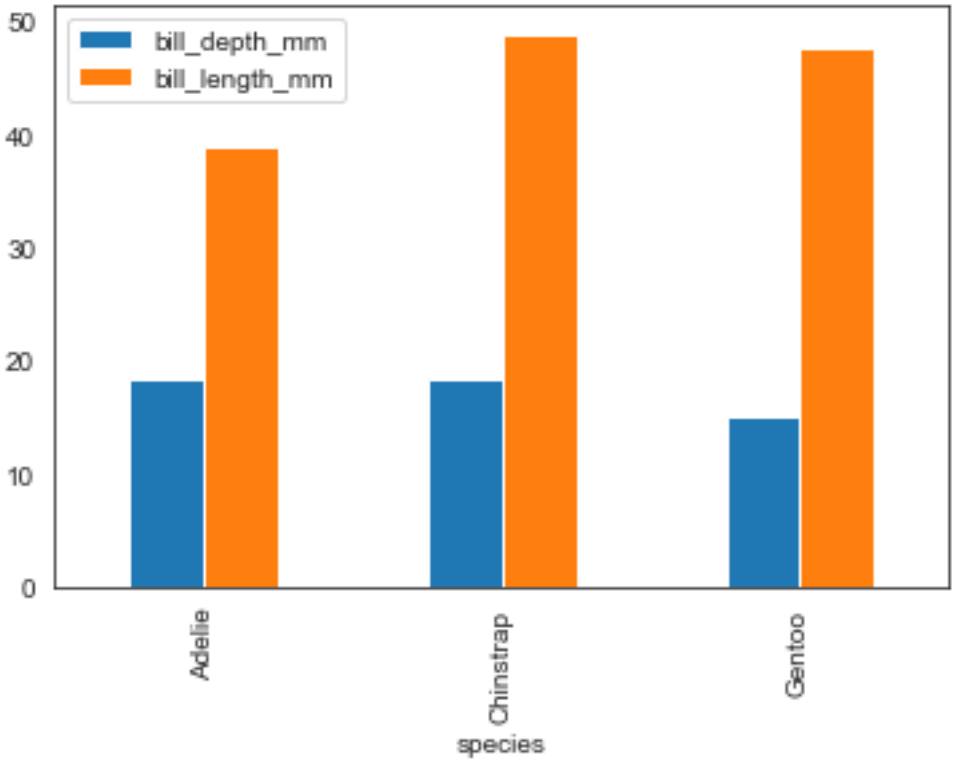
[71]:
```
# A better way is to set as_index=False
small.groupby(['species', 'island'], as_index=False).mean()
```

[71]:

	species	island	bill_depth_mm	bill_length_mm
0	Adelie	Biscoe	18.4	39.0
1	Adelie	Dream	18.3	38.5
2	Adelie	Torgersen	18.4	39.0
3	Chinstrap	Dream	18.4	48.8
4	Gentoo	Biscoe	15.0	47.5

[72]:
```
small.groupby('species').mean().plot(kind='bar')  # This is actually a pandas␣
↪plot
```

[72]:
```
<AxesSubplot:xlabel='species'>
```



9 Group by numerical data using .cut() and .qcut()

[73]:
```
df['mass_group'] = pd.cut(df['body_mass_g'],
                          bins=[0, 3000, 4000, 5000, 10000],
                          labels=('small', 'medium','large', 'wow'))
df.head()
```

[73]:
```
  species     island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
0  Adelie  Torgersen            39.1           18.7              181.0
1  Adelie  Torgersen            39.5           17.4              186.0
2  Adelie  Torgersen            40.3           18.0              195.0
3  Adelie  Torgersen             NaN            NaN                NaN
4  Adelie  Torgersen            36.7           19.3              193.0

   body_mass_g     sex mass_group
0       3750.0    Male     medium
1       3800.0  Female     medium
2       3250.0  Female     medium
3          NaN     NaN        NaN
4       3450.0  Female     medium
```

[ ]:
```
df.groupby('mass_group').agg(["mean", "median"])
```

[75]:
```
df.groupby(pd.cut(df['body_mass_g'],
                          bins=[0, 3000, 4000, 5000, 10000],
                          labels=('small', 'medium','large', 'wow'))).mean()
```

[75]:
```
             bill_length_mm  bill_depth_mm  flipper_length_mm  body_mass_g
body_mass_g
small                  38.1           17.2              186.0       2900.0
medium                 41.5           18.1              190.6       3576.1
large                  45.0           16.6              206.1       4512.6
wow                    49.3           15.6              221.1       5501.6
```

[76]:
```
df.groupby(pd.qcut(df["body_mass_g"],4, duplicates="drop")).mean()
```

[76]:
```
                    bill_length_mm  bill_depth_mm  flipper_length_mm  \
body_mass_g
(2699.999, 3550.0]            39.9           17.7              188.6
(3550.0, 4050.0]              43.2           18.5              192.7
(4050.0, 4750.0]              44.4           16.8              203.9
(4750.0, 6300.0]              48.5           15.5              219.3

                    body_mass_g
```

```
body_mass_g
(2699.999, 3550.0]       3297.8
(3550.0, 4050.0]         3808.0
(4050.0, 4750.0]         4430.6
(4750.0, 6300.0]         5333.2
```

[77]:
```
# Just a note....
```

[78]:
```
df.groupby(['species', 'island']).bill_length_mm.sum().reset_index()
```

[78]:
```
     species     island  bill_length_mm
0     Adelie     Biscoe          1714.9
1     Adelie      Dream          2156.1
2     Adelie  Torgersen          1986.5
3  Chinstrap      Dream          3320.7
4     Gentoo     Biscoe          5843.1
```

[79]:
```
# A different way to write he same code
df.groupby(['species', 'island'])['bill_length_mm'].sum().reset_index()
```

[79]:
```
     species     island  bill_length_mm
0     Adelie     Biscoe          1714.9
1     Adelie      Dream          2156.1
2     Adelie  Torgersen          1986.5
3  Chinstrap      Dream          3320.7
4     Gentoo     Biscoe          5843.1
```

10 Return to the Beer notebook and complete part 2