

lab3

November 29, 2019

```
[2]: import pandas as pd
import matplotlib.pyplot as plt
from IPython.display import display
import seaborn as sns
```

```
[3]: pd_city = pd.read_csv("./city_data.csv")
pd_city.head(5)
```

```
[3]:
```

	city	driver_count	type
0	Richardfort	38	Urban
1	Williamsstad	59	Urban
2	Port Angela	67	Urban
3	Rodneyfort	34	Urban
4	West Robert	39	Urban

```
[4]: pd_city.dtypes
```

```
[4]: city          object
driver_count    int64
type            object
dtype: object
```

```
[5]: pd_ride = pd.read_csv("./ride_data.csv")
pd_ride.head(5)
```

```
[5]:
```

	city	date	fare	ride_id
0	Lake Jonathanshire	2018-01-14 10:14:22	13.83	5739410935873
1	South Michelleport	2018-03-04 18:24:09	30.24	2343912425577
2	Port Samanthamouth	2018-02-24 04:29:00	33.44	2005065760003
3	Rodneyfort	2018-02-10 23:22:03	23.44	5149245426178
4	South Jack	2018-03-06 04:28:35	34.58	3908451377344

```
[6]: pd_ride.dtypes
```

```
[6]: city          object
date            object
fare           float64
ride_id         int64
dtype: object
```

```
[7]: pd_join = pd_city.join(pd_ride.set_index("city"), on="city")
pd_join['type'].unique()
```

```
[7]: array(['Urban', 'Suburban', 'Rural'], dtype=object)
```

```
[8]: pd_join[pd_join['city']=='Amandaburgh']
```

```
[8]:
```

	city	driver_count	type	date	fare	\
61	Amandaburgh	12	Urban	2018-03-05 02:15:38	26.28	
61	Amandaburgh	12	Urban	2018-02-24 23:10:49	43.66	
61	Amandaburgh	12	Urban	2018-02-10 20:42:46	36.17	
61	Amandaburgh	12	Urban	2018-01-11 02:22:07	29.24	
61	Amandaburgh	12	Urban	2018-01-21 04:12:54	9.26	
61	Amandaburgh	12	Urban	2018-04-19 16:30:12	6.27	
61	Amandaburgh	12	Urban	2018-03-20 07:40:33	27.45	
61	Amandaburgh	12	Urban	2018-04-01 09:24:21	24.29	
61	Amandaburgh	12	Urban	2018-04-20 02:16:07	16.27	
61	Amandaburgh	12	Urban	2018-03-13 12:52:31	13.88	
61	Amandaburgh	12	Urban	2018-04-22 21:34:17	42.52	
61	Amandaburgh	12	Urban	2018-02-06 10:02:30	11.93	
61	Amandaburgh	12	Urban	2018-04-24 08:02:27	14.55	
61	Amandaburgh	12	Urban	2018-04-05 10:22:33	25.55	
61	Amandaburgh	12	Urban	2018-03-07 02:26:33	18.76	
61	Amandaburgh	12	Urban	2018-01-02 09:57:04	33.06	
61	Amandaburgh	12	Urban	2018-01-13 16:04:10	23.35	
61	Amandaburgh	12	Urban	2018-01-29 23:28:12	41.06	

```
ride_id
```

61	906850928986
61	6573820412437
61	6455620849753
61	7279902884763
61	5528427024492
61	4400632718421
61	3701008274871
61	1995462170530
61	3513123734716
61	6222134922674
61	1901157522591
61	7550325158038
61	7836117055007
61	8581415267582
61	3419454549176
61	6330658179518
61	9975084532253
61	4296858665195

```
[9]: pd_group = pd.DataFrame(data=list(pd_join.groupby(by='city').  
    ↪mean()['fare']),columns=['Average Fare'])  
  
type(pd_join.groupby(by='city').mean()['fare'])
```

```

pd_group['city'] = pd_join.groupby(by='city').indices
a = pd_join.groupby(by='city').count()['driver_count']
pd_group['Number of Drivers'] = list(pd_join.groupby(by='city').
    →count()['driver_count'])
pd_group['Number of Rides'] =pd_group['Number of Drivers']
pd_group.set_index('city')
#pd_group.set_index('city')
#pd_group() why re set_index in other cell

```

```

[9]:

```

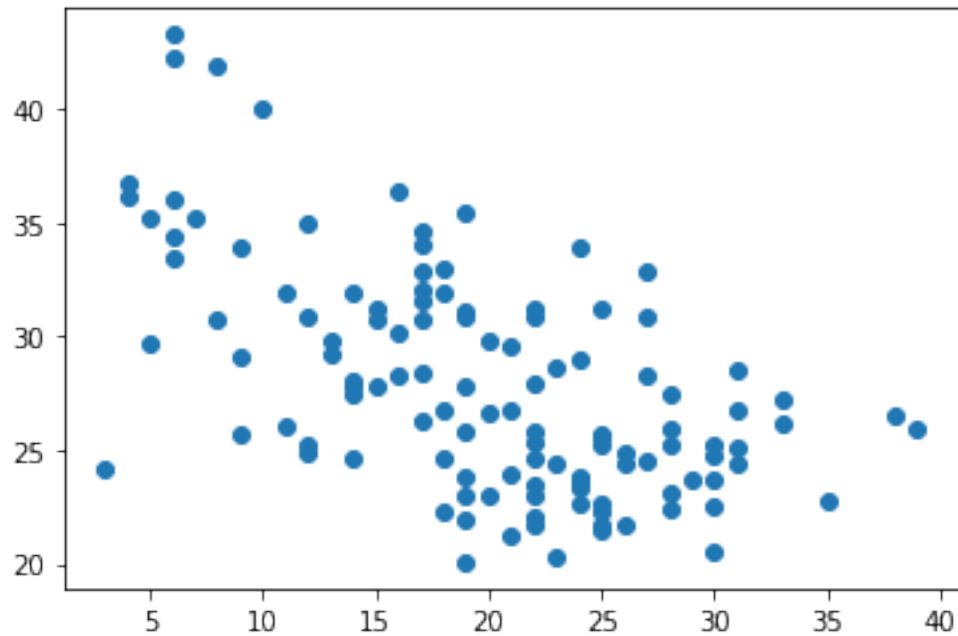
	Average Fare	Number of Drivers	Number of Rides
city			
Amandaburgh	24.641667	18	18
Barajasview	25.332273	22	22
Barronchester	36.422500	16	16
Bethanyland	32.956111	18	18
Bradshawfurt	40.064000	10	10
Brandonfort	35.437368	19	19
Carriemouth	28.314444	27	27
Christopherfurt	24.501852	27	27
Colemanland	30.894545	22	22
Davidfurt	31.995882	17	17
Deanville	25.842632	19	19
East Aaronbury	25.661111	9	9
East Danielview	31.560588	17	17
East Kaylahaven	23.757931	29	29
East Kentstad	29.823077	13	13
East Marymouth	30.835185	27	27
Erikaland	24.906667	12	12
Garzaport	24.123333	3	3
Grahamburgh	25.221200	25	25
Grayville	27.763333	15	15
Harringtonfort	33.470000	6	6
Huntermouth	28.993750	24	24
Hurleymouth	25.891429	28	28
Jerryton	25.649200	25	25
Jessicaport	36.013333	6	6
Johnton	26.785714	21	21
Joneschester	22.289600	25	25
Josephside	32.858148	27	27
Justinberg	23.694333	30	30
Karenberg	26.340000	17	17
...
South Evanton	26.726129	31	31
South Jack	22.965263	19	19
South Jennifer	35.264286	7	7
South Karenland	26.535526	38	38
South Latoya	20.093158	19	19

South Marychester	41.870000	8	8
South Michelleport	24.451613	31	31
South Phillip	28.571290	31	31
South Saramouth	36.160000	4	4
South Teresa	31.220455	22	22
Taylorhaven	42.263333	6	6
Valentineton	24.636364	22	22
Veronicaberg	32.828235	17	17
Victoriaport	27.780000	14	14
West Angela	25.990000	39	39
West Anthony	24.736667	30	30
West Christopherberg	24.421154	26	26
West Ericstad	22.347222	18	18
West Gabriel	20.346087	23	23
West Hannah	29.547619	21	21
West Heather	33.890000	9	9
West Heidi	23.133929	28	28
West Josephberg	21.720385	26	26
West Kimmouth	29.871500	20	20
West Patrickchester	28.233125	16	16
West Robert	25.123871	31	31
West Samuelburgh	21.767600	25	25
Williamsonville	31.875000	14	14
Williamsstad	24.362174	23	23
Williamsview	26.599000	20	20

[120 rows x 3 columns]

```
[10]: plt.scatter(list(pd_group['Number of Drivers']),list(pd_group['Average Fare']))
```

```
[10]: <matplotlib.collections.PathCollection at 0x268db365710>
```

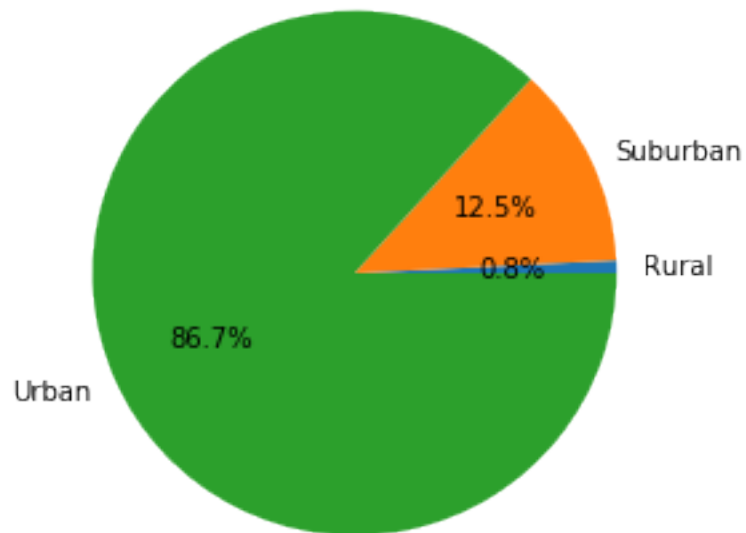
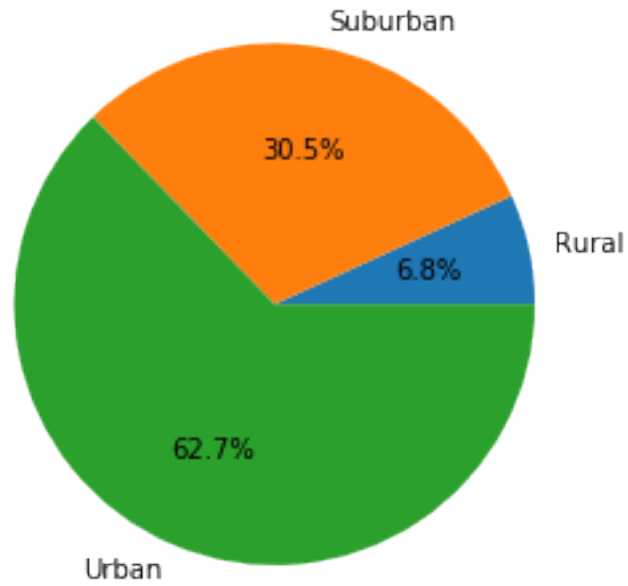


```
[11]: pd_join.groupby(by='type').sum()
```

```
[11]:
```

	driver_count	fare	ride_id
type			
Rural	537	4327.93	580968240341287
Suburban	8570	19356.33	3106884522576766
Urban	59602	39854.38	7919412664056093

```
[12]: fig,axe1 = plt.subplots()
total_fare_index = pd_join.groupby(by='type').sum()['fare'].index
total_fare_value = list(pd_join.groupby(by='type').sum()['fare'])
axe1.pie(total_fare_value,labels=total_fare_index,autopct='%1.1f%%')
axe1.axis('equal')
total_ride_index = pd_join.groupby(by='type').sum()['driver_count'].index
total_ride_value = list(pd_join.groupby(by='type').sum()['driver_count'])
fig,axe2 = plt.subplots()
axe2.pie(total_ride_value,labels=total_ride_index,autopct='%1.1f%%')
axe2.axis('equal')
plt.show()
```



```
[30]: #t = sns.load_dataset("titanic") why "" not working
t = sns.load_dataset('titanic')
#len(t.index)
t.head(10)
```

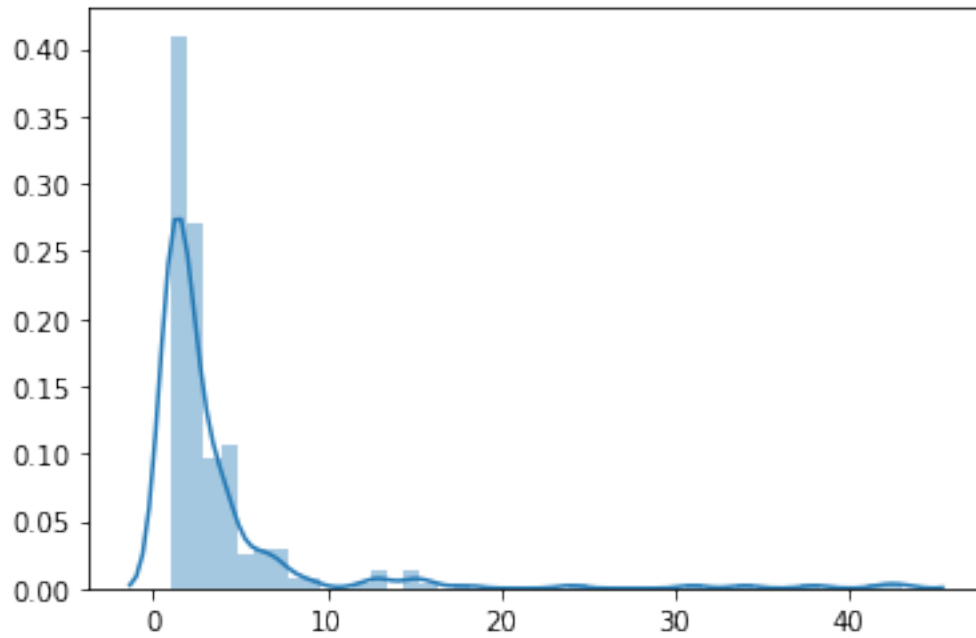
```
[30]:   survived  pclass    sex  age  sibsp  parch   fare embarked  class \
0         0      3  male  22.0     1     0   7.2500         S   Third
1         1      1 female  38.0     1     0  71.2833         C   First
```

2	1	3	female	26.0	0	0	7.9250	S	Third
3	1	1	female	35.0	1	0	53.1000	S	First
4	0	3	male	35.0	0	0	8.0500	S	Third
5	0	3	male	NaN	0	0	8.4583	Q	Third
6	0	1	male	54.0	0	0	51.8625	S	First
7	0	3	male	2.0	3	1	21.0750	S	Third
8	1	3	female	27.0	0	2	11.1333	S	Third
9	1	2	female	14.0	1	0	30.0708	C	Second

	who	adult_male	deck	embark_town	alive	alone
0	man	True	NaN	Southampton	no	False
1	woman	False	C	Cherbourg	yes	False
2	woman	False	NaN	Southampton	yes	True
3	woman	False	C	Southampton	yes	False
4	man	True	NaN	Southampton	no	True
5	man	True	NaN	Queenstown	no	True
6	man	True	E	Southampton	no	True
7	child	False	NaN	Southampton	no	False
8	woman	False	NaN	Southampton	yes	False
9	child	False	NaN	Cherbourg	yes	False

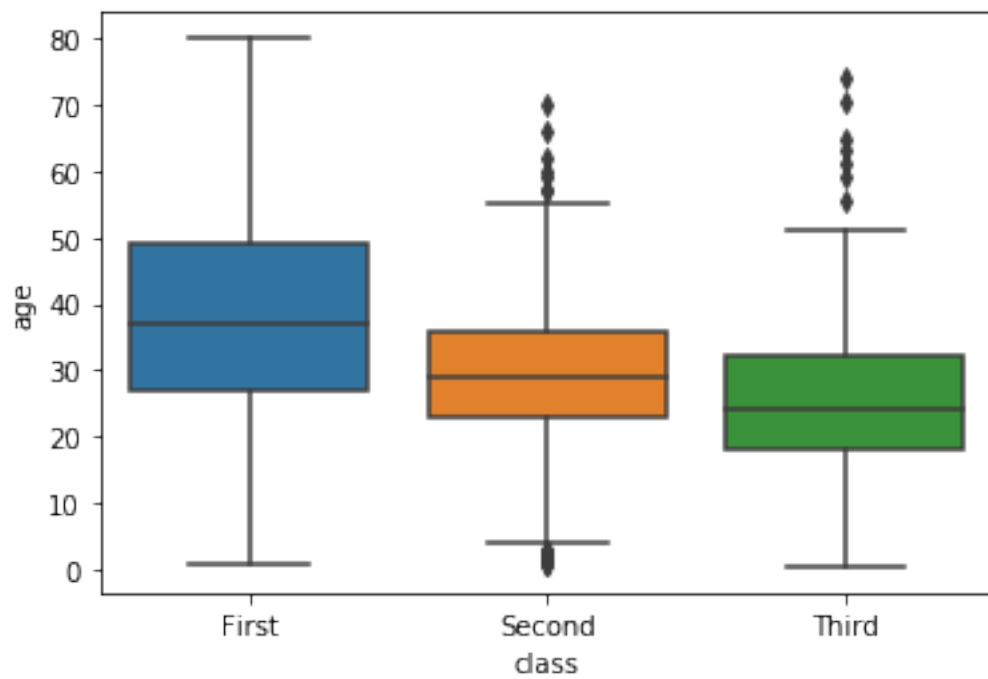
```
[49]: fare_pd = pd.DataFrame()
fare_pd['fare'] = t.groupby('fare').indices
fare_pd['count'] = t.groupby('fare').count()['sex']
fare_pd.set_index('fare', inplace=True, drop=True)
#type(fare_pd)
#fare_pd
#t.groupby('fare').count().head(3)
sns.distplot(fare_pd)
```

```
[49]: <matplotlib.axes._subplots.AxesSubplot at 0x268e653a978>
```



```
[51]: sns.boxplot(x='class',y='age',data=t)
```

```
[51]: <matplotlib.axes._subplots.AxesSubplot at 0x268e6679c18>
```




```
[76]: male_pd = t[t['sex']=='male']
male_pd.head(5)
male_pd.dropna(inplace=True)
male_pd['age']
#male_pd.groupby('age').count()
#male_pd['age']<=80
sns.distplot(male_pd['age'])
```

E:\software\anaconda\lib\site-packages\ipykernel_launcher.py:3:

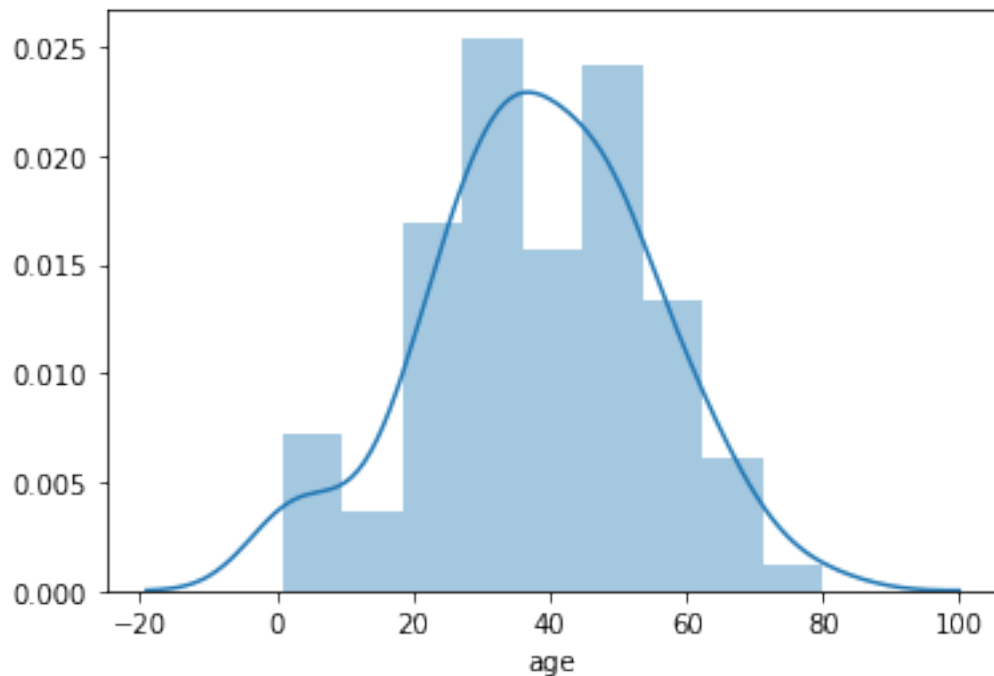
SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: <http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy>

This is separate from the ipykernel package so we can avoid doing imports until

```
[76]: <matplotlib.axes._subplots.AxesSubplot at 0x268e6981c50>
```

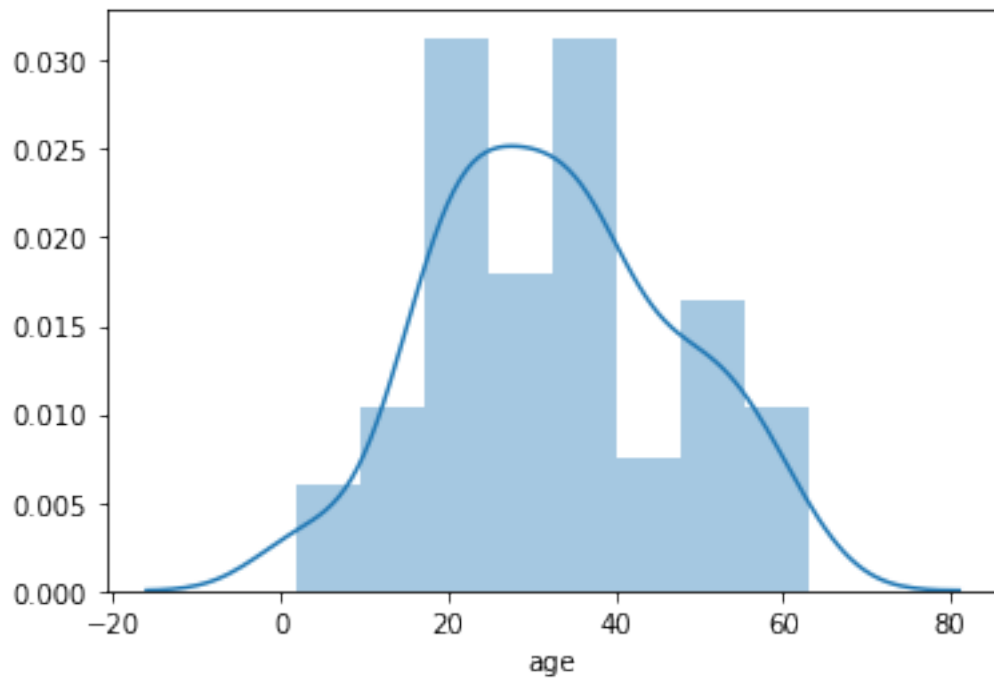


```
[79]: female_pd = t[t['sex']=='female']
female_pd.dropna(inplace=True)
#female_pd['age']
#male_pd.groupby('age').count()
#male_pd['age']<=80
sns.distplot(female_pd['age'])
```

```
E:\software\anaconda\lib\site-packages\ipykernel_launcher.py:2:  
SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame
```

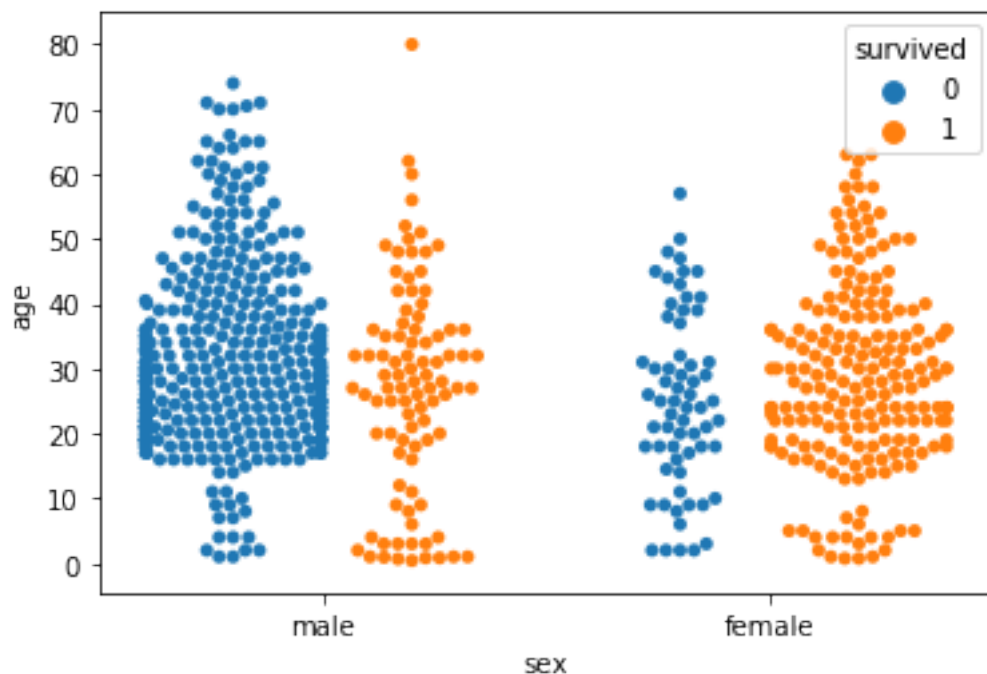
See the caveats in the documentation: <http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy>

[79]: <matplotlib.axes._subplots.AxesSubplot at 0x268e6a06438>



[82]: `sns.swarmplot(x='sex',y='age',hue='survived',dodge=True,data=t)`

[82]: <matplotlib.axes._subplots.AxesSubplot at 0x268e6beb400>



```
[78]: wine_pd = pd.read_csv("./wine_data.csv")
      #wine_pd.head()
      wine_pd['Label'].unique()
      labels = wine_pd['Label']
      labels
      wine_pd = wine_pd.drop(axis=1,columns='Label')
      wine_pd.head(5)
```

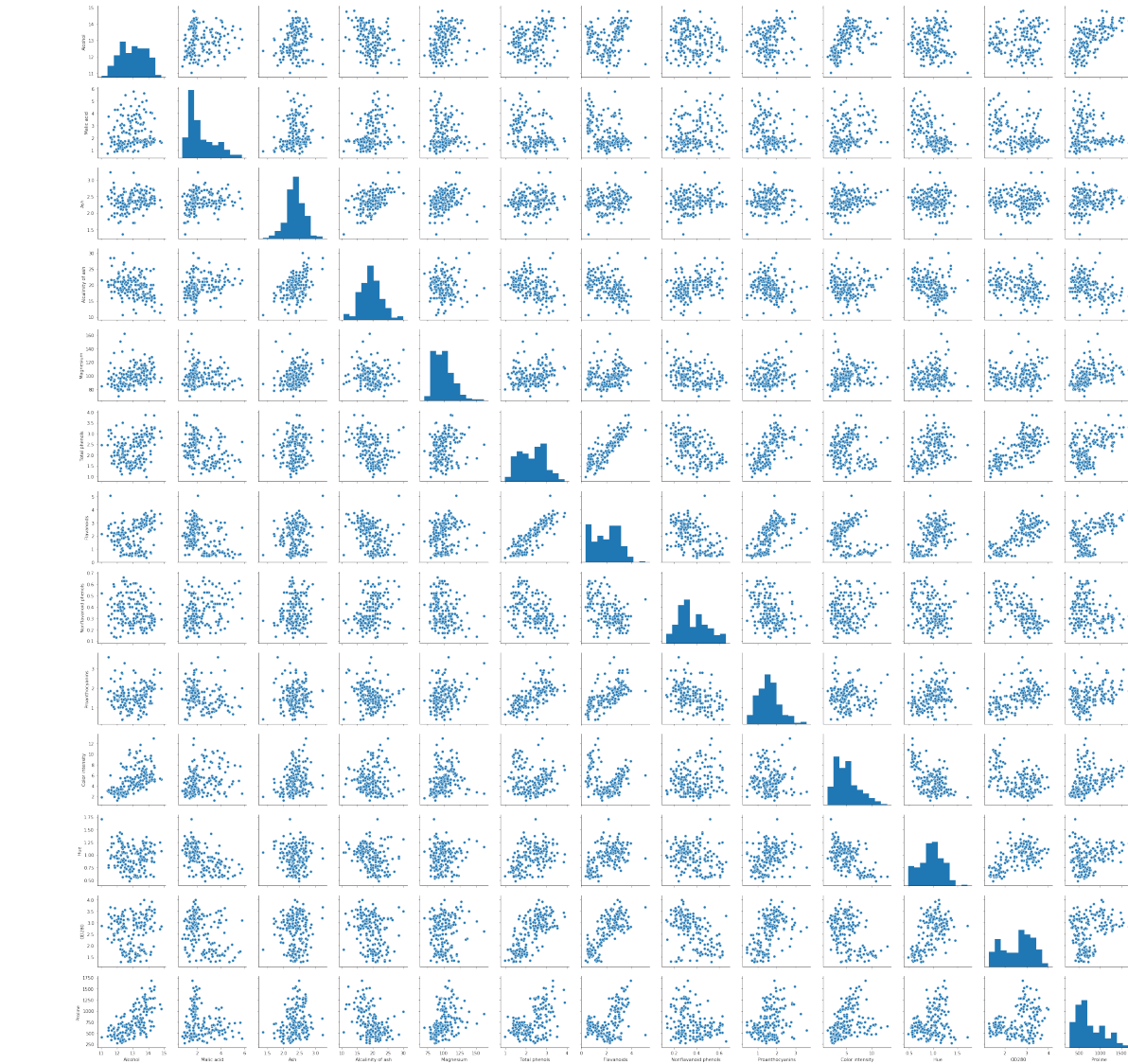
```
[78]:   Alcohol  Malic acid  Ash  Alcalinity of ash  Magnesium  Total phenols  \
0    14.23      1.71  2.43             15.6         127         2.80
1    13.20      1.78  2.14             11.2         100         2.65
2    13.16      2.36  2.67             18.6         101         2.80
3    14.37      1.95  2.50             16.8         113         3.85
4    13.24      2.59  2.87             21.0         118         2.80

      Flavanoids  Nonflavanoid phenols  Proanthocyanins  Color intensity  Hue  \
0          3.06              0.28          2.29          5.64  1.04
1          2.76              0.26          1.28          4.38  1.05
2          3.24              0.30          2.81          5.68  1.03
3          3.49              0.24          2.18          7.80  0.86
4          2.69              0.39          1.82          4.32  1.04

      OD280  Proline
0    3.92    1065
1    3.40    1050
2    3.17    1185
```

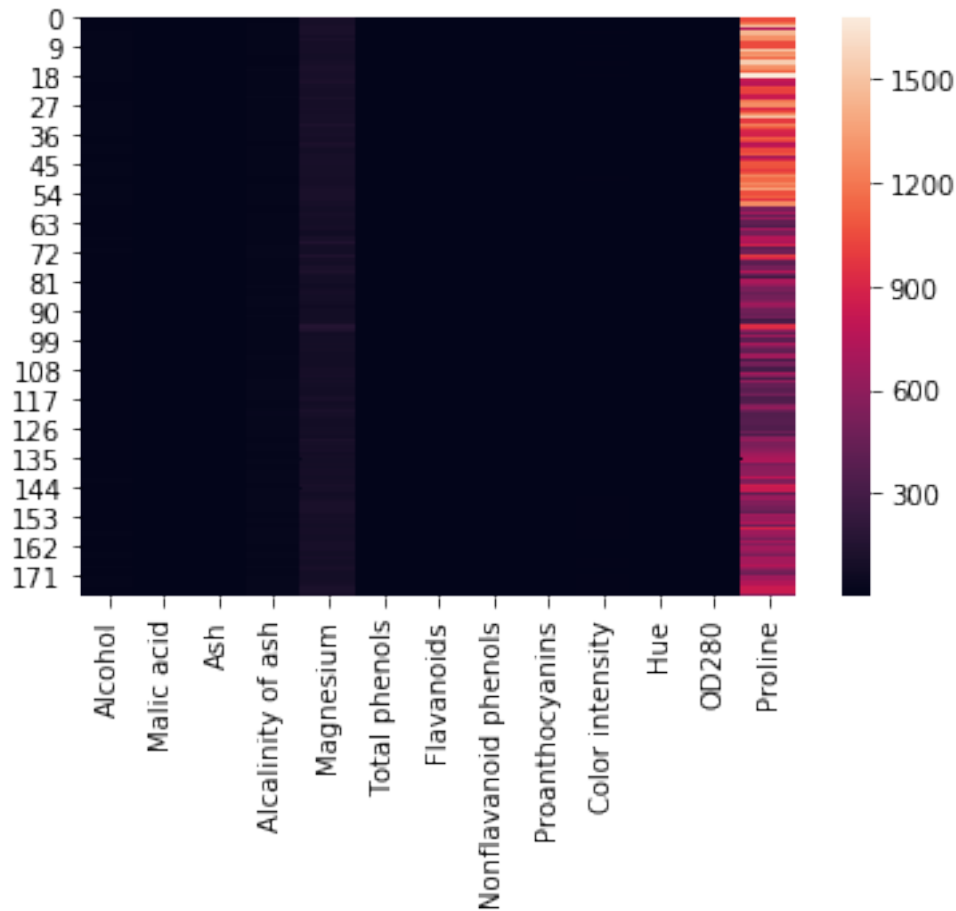
```
3    3.45    1480
4    2.93    735
```

```
[14]: sns_plot = sns.pairplot(wine_pd,diag_kind="hist")
```



```
[15]: #plt.subplots(figsize=(100,100))
sns.heatmap(wine_pd)
```

```
[15]: <matplotlib.axes._subplots.AxesSubplot at 0x268e1e7a080>
```



```
[16]: from sklearn import preprocessing
      from sklearn.cluster import KMeans
```

```
[17]: standardScaler = preprocessing.StandardScaler()
      standardScaler.fit(wine_pd)
      X_scaled_array = standardScaler.transform(wine_pd)
      X_scaled_array
      #normalizedData = pd.DataFrame(X_scaled_array, columns = wine_pd.columns)
      #normalizedData.head(5)
      #len(normalizedData.index)
```

```
[17]: array([[ 1.51861254, -0.5622498 ,  0.23205254, ...,  0.36217728,
                1.84791957,  1.01300893],
              [ 0.24628963, -0.49941338, -0.82799632, ...,  0.40605066,
                1.1134493 ,  0.96524152],
              [ 0.19687903,  0.02123125,  1.10933436, ...,  0.31830389,
                0.78858745,  1.39514818],
              ...,
              [ 0.33275817,  1.74474449, -0.38935541, ..., -1.61212515,
                -1.48544548,  0.28057537],
```

```
[ 0.20923168,  0.22769377,  0.01273209, ..., -1.56825176,
 -1.40069891,  0.29649784],
 [ 1.39508604,  1.58316512,  1.36520822, ..., -1.52437837,
 -1.42894777, -0.59516041]])
```

```
[18]: kMeansClustering = KMeans(n_clusters = 3)
      res = kMeansClustering.fit_predict(normalizedData)
      res
```

```

      □
↳ -----

      NameError                                Traceback (most recent call↳
↳ last)

      <ipython-input-18-6005a674d9fd> in <module>
          1 kMeansClustering = KMeans(n_clusters = 3)
      ----> 2 res = kMeansClustering.fit_predict(normalizedData)
          3 res

      NameError: name 'normalizedData' is not defined
```

```
[ ]: normalizedData['cluster'] = res
      normalizedData.head(3)
```

```
[ ]: sns_plot = sns.pairplot(normalizedData, hue = "cluster",diag_kind="hist")
```