Pandas Part II and matplotlib Notes

March 7, 2022

1 More pandas

Last time, we learned about pandas two main data structures, Series and DataFrame, and how to create these objects. In this final class, we will learn about two of the most commonly encountered tasks in data management, namely data cleaning and data visualisation.

1.1 Data cleaning

[31]:
```
import numpy as np

import pandas as pd
```

[32]:
```
wine = pd.read_csv('./wineData.csv', sep = ',')
```

[33]:
```
wine.shape
```

[33]:
```
(150930, 11)
```

We can check if *any* row has NULL values, using following: `df.isnull().any()`. First, the `.isnull()` function runs on the whole data frame, returning a Boolean data frame. Then `.any()` checks if any values are true. This returns a Series telling us what, if anything, satisfies the condition. In our case, we find that there are several values missing!

[34]:
```
wine.isnull().any()
```

[34]:
```
Unnamed: 0     False
country         True
description    False
designation     True
points         False
price           True
province        True
region_1        True
region_2        True
variety        False
winery         False
dtype: bool
```

Missing data is commonly encountered in data exploration. Although one has to proceed with caution, we can "clean" the data in a number of ways. Some examples include:

- replace the value with a known value using `df.replace(value, replacement_value)` (for example, if new data becomes available)
- `df.fillna()`. Some options for inside the brackets are (i) replace with a known value or (ii) use `method = ffill`/`method = backfill` to replace the rows containing `NaN` with the row before/after.
- delete rows or columns using `df.dropna(axis = x)` where `x = 0` for rows and `x = 1` for columns. (Should be used with care, we don't want to delete any information that might be important!)
- interpolate between values using `df.interpolate()`

Let's try `.dropna()` on our wine DataFrame.

[42]:
```
wineCut = wine.dropna()

wineCut.shape
```

[42]:
```
(39241, 11)
```

By applying this operation, we have lost 111689 rows! (Hence my advise to proceed with caution: this is a dramatic reduction.)

2 Data visualisation using matplotlib

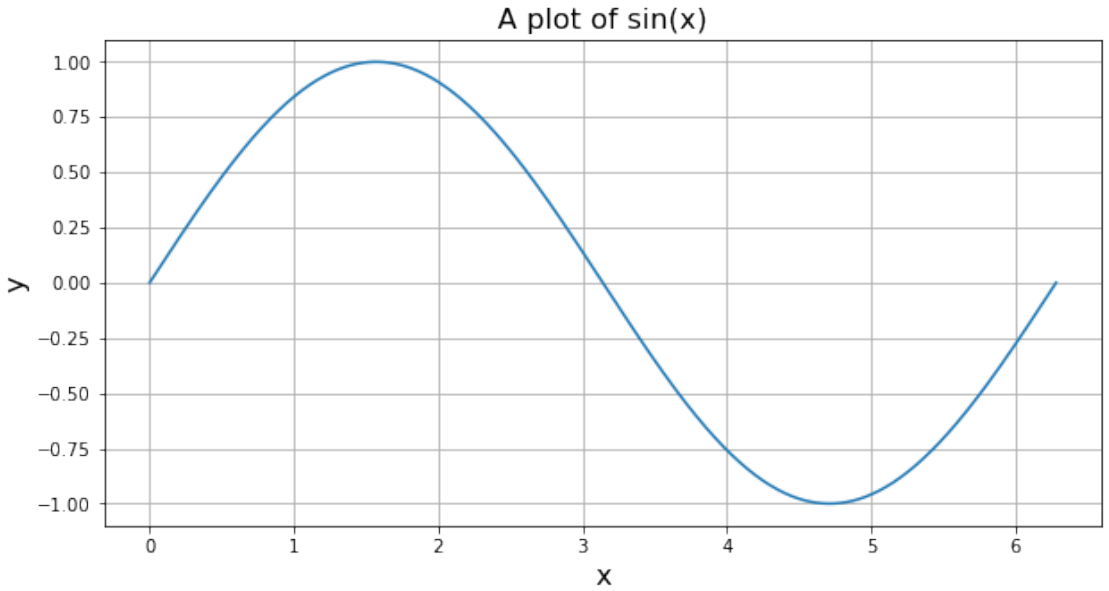
We've imported, inspected, and cleaned our data. We now want to visualise the data. This can be done in Python using **matplotlib**, an *extensive* library of plotting functionalities with similarities to MATLAB. Some options are:

- `df.plot.bar()`, bar plot
- `df.plot.box()`, box plot
- `df.plot.hist()`, histogram
- `df.plot()`, line plot

2.1 Line plot

[131]:
```
x = np.linspace(0, 2*np.pi, 1000)

y = np.sin(x)

fig1, ax1 = plt.subplots(figsize = [10, 5])

ax1.plot(x, y)
```

```
plt.title('A plot of sin(x)', fontsize = 16)
plt.xlabel('x', fontsize = 16)
plt.ylabel('y', fontsize = 16)
plt.grid(True)
```



2.2 Histogram

[132]:
```
wine.head()
```

[132]:

	Unnamed: 0	country	description	designation	points	price	province	region_1	region_2	variety
0	0	US	This tremendous 100% varietal wine hails from …	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon
1	1	Spain	Ripe aromas of fig, blackberry and cassis are …	Carodorum Selección Especial Reserva	96	110.0	Northern Spain			
2	2	US	Mac Watson honors the memory of a wine once ma…	Special Selected Late Harvest	96	90.0	California			
3	3	US	This spent 20 months in 30% new French oak, an…	Reserve	96	65.0	Oregon			
4	4	France	This is the top wine from La Bégude, named aft…	La Brûlade	95	66.0	Provence			

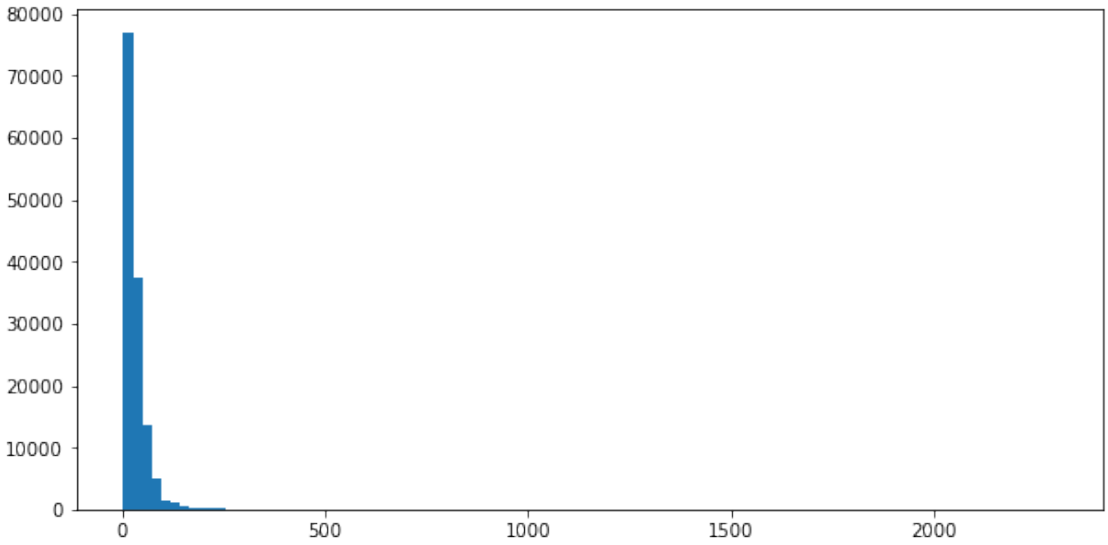
```
1              Toro                NaN       Tinta de Toro
2     Knights Valley             Sonoma     Sauvignon Blanc
3  Willamette Valley  Willamette Valley          Pinot Noir
4             Bandol                NaN  Provence red blend

                   winery
0                  Heitz
1  Bodega Carmen Rodríguez
2                Macauley
3                  Ponzi
4    Domaine de la Bégude
```

[133]:
```
len(wine['price'].unique())
```

[133]: 358

[134]:
```
# Needed for proper output in Jupyter

%matplotlib inline

import matplotlib.pyplot as plt
```

[135]:
```
fig2, ax2 = plt.subplots(figsize=[10,5])

ax2.hist(wine['price'], bins = 100); # We specify the number of bins to be 358,␣
 ↪ the number of unique prices in the database

#ax2.set_xlim([0, 500]) #uncomment to zoom in on 0-500 range
```



We can see that the majority of wines are priced less than 500 dollars in this database. To zoom in on the 0-500 dollar range, we uncomment the `ax2.set_xlim()` option above.

2.3 Saving and exporting figures

We'll want to export figures from Python and save them to a directory. A common mistake people make is to export a figure, only to find that, when the figure is included in LaTeX for example, the figure is too small. It is therefore a good idea to do all your sizing and formatting *in Python* before you export it (this could take a couple of tries before you're happy, but it's worth it.)

[138]:
```
# Export the sin curve above as a high quality .png file. Need to specify the␣
↪facecolor - one of matplotlib many annoying quirks.

fig1.savefig('sinPlot.png', format='png',dpi=800, facecolor = 'w')
```