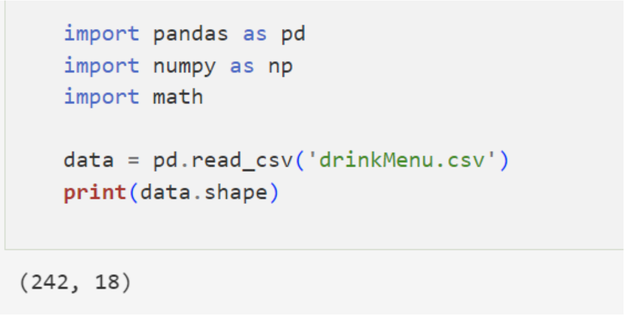
Data Science Task

Import dataset

- Dataset loaded from excel file using pandas library API pandas.read_csv(filename).



```
import pandas as pd
import numpy as np
import math

data = pd.read_csv('drinkMenu.csv')
print(data.shape)
```

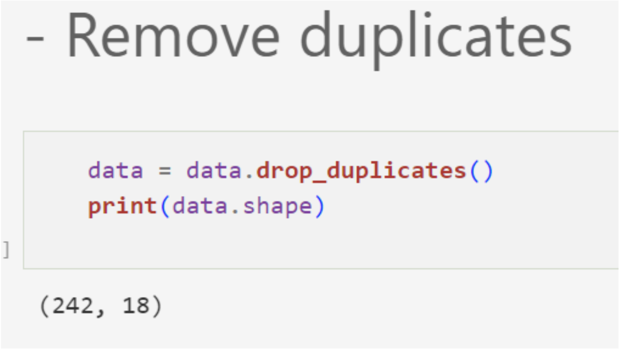
```
(242, 18)
```

- This code output the shape of the data loaded that has 242 row * 18 columns.

Data Preparation

Before applying any analysis on our data, we need to prepare it by some techniques:

- **Remove duplicates**
  - Using dataframe API dataframe.drop_duplicates(), this return distinct rows from our data and remove repeated rows.
  - But in our case the output same as input because there is no duplicates.



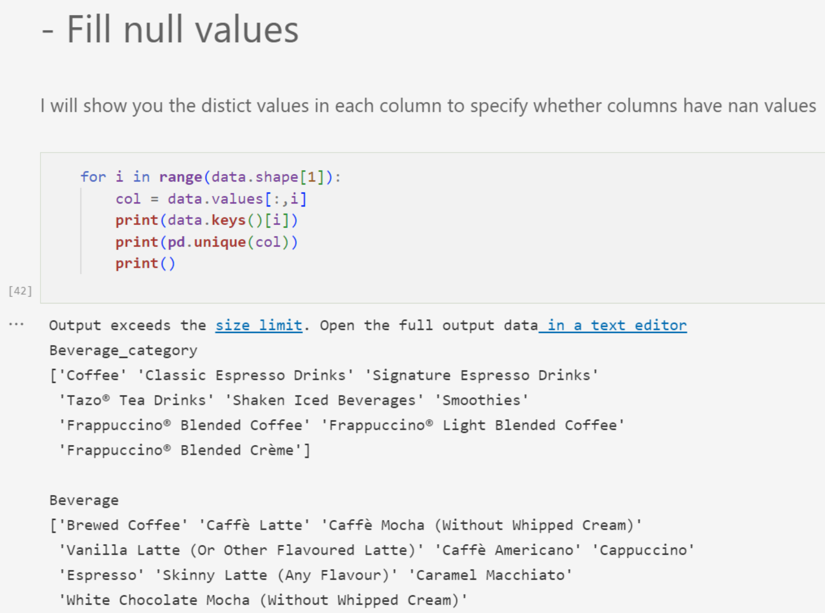
- Remove duplicates

```
data = data.drop_duplicates()
print(data.shape)
```

```
(242, 18)
```

  - Notice the out shape same as the input shame above.

- **Fill null values**
  - First, I have to check which columns have nan values, so I will print all columns distinct values using pandas.unique().



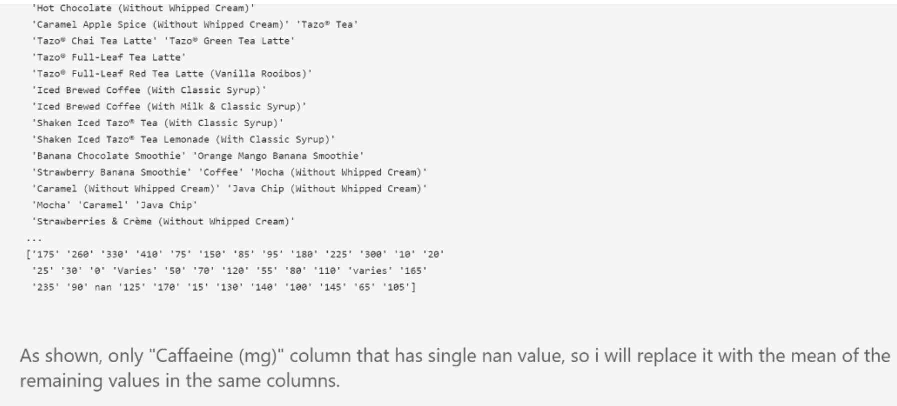
- Fill null values

I will show you the distict values in each column to specify whether columns have nan values

```
for i in range(data.shape[1]):
    col = data.values[:,i]
    print(data.keys()[i])
    print(pd.unique(col))
    print()
```

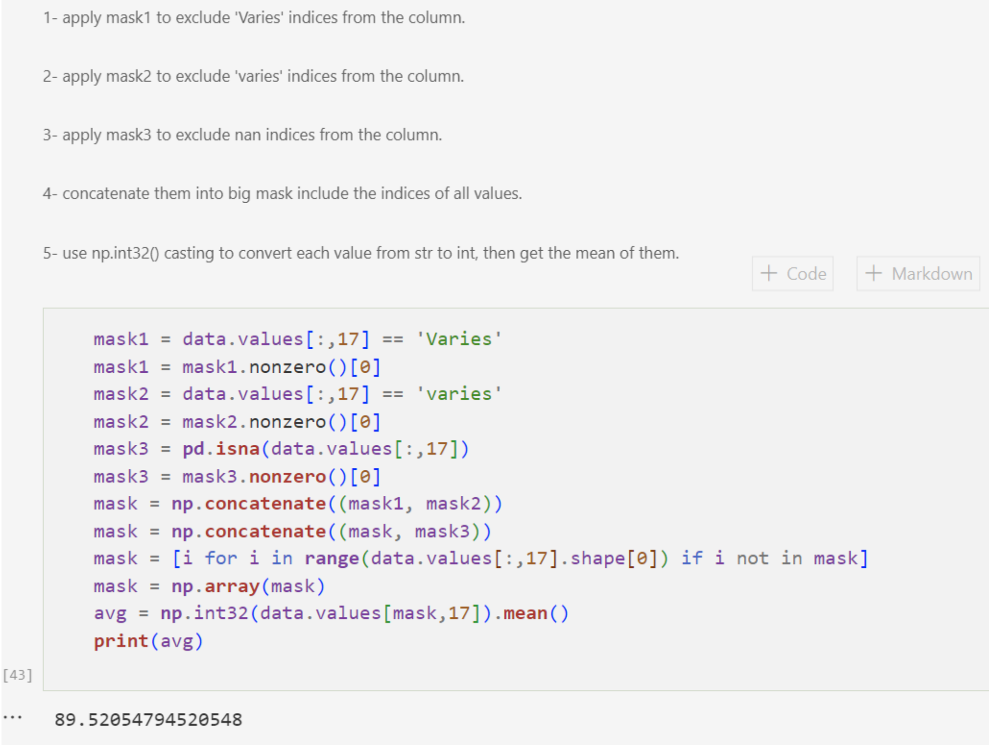
```
Output exceeds the size limit. Open the full output data in a text editor
Beverage_category
['Coffee' 'Classic Espresso Drinks' 'Signature Espresso Drinks'
 'Tazo® Tea Drinks' 'Shaken Iced Beverages' 'Smoothies'
 'Frappuccino® Blended Coffee' 'Frappuccino® Light Blended Coffee'
 'Frappuccino® Blended Crème']

Beverage
['Brewed Coffee' 'Caffè Latte' 'Caffè Mocha (Without Whipped Cream)'
 'Vanilla Latte (Or Other Flavoured Latte)' 'Caffè Americano' 'Cappuccino'
 'Espresso' 'Skinny Latte (Any Flavour)' 'Caramel Macchiato'
 'White Chocolate Mocha (Without Whipped Cream)'
```



```
 'Hot Chocolate (Without Whipped Cream)'
 'Caramel Apple Spice (Without Whipped Cream)' 'Tazo® Tea'
 'Tazo® Chai Tea Latte' 'Tazo® Green Tea Latte'
 'Tazo® Full-Leaf Tea Latte'
 'Tazo® Full-Leaf Red Tea Latte (Vanilla Rooibos)'
 'Iced Brewed Coffee (With Classic Syrup)'
 'Iced Brewed Coffee (With Milk & Classic Syrup)'
 'Shaken Iced Tazo® Tea (With Classic Syrup)'
 'Shaken Iced Tazo® Tea Lemonade (With Classic Syrup)'
 'Banana Chocolate Smoothie' 'Orange Mango Banana Smoothie'
 'Strawberry Banana Smoothie' 'Coffee' 'Mocha (Without Whipped Cream)'
 'Caramel (Without Whipped Cream)' 'Java Chip (Without Whipped Cream)'
 'Mocha' 'Caramel' 'Java Chip'
 'Strawberries & Crème (Without Whipped Cream)'
...
['175' '260' '330' '410' '75' '150' '85' '95' '180' '225' '300' '10' '20'
 '25' '30' '0' 'Varies' '50' '70' '120' '55' '80' '110' 'varies' '165'
 '235' '90' nan '125' '170' '15' '130' '140' '100' '145' '65' '105']
```

As shown, only "Caffaeine (mg)" column that has single nan value, so i will replace it with the mean of the remaining values in the same columns.



1- apply mask1 to exclude 'Varies' indices from the column.

2- apply mask2 to exclude 'varies' indices from the column.

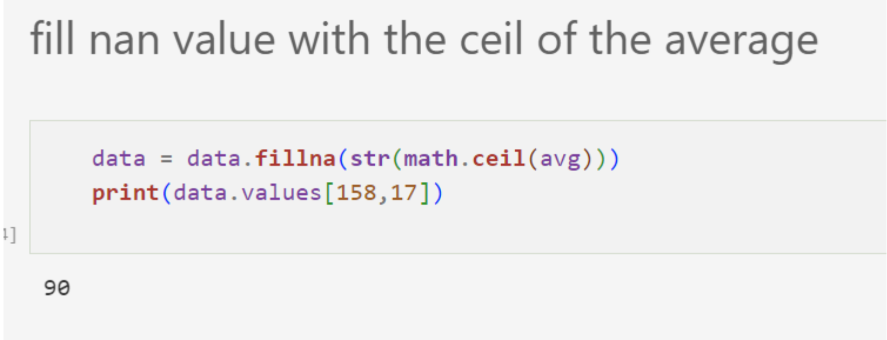
3- apply mask3 to exclude nan indices from the column.

4- concatenate them into big mask include the indices of all values.

5- use np.int32() casting to convert each value from str to int, then get the mean of them.

```
mask1 = data.values[:,17] == 'Varies'
mask1 = mask1.nonzero()[0]
mask2 = data.values[:,17] == 'varies'
mask2 = mask2.nonzero()[0]
mask3 = pd.isna(data.values[:,17])
mask3 = mask3.nonzero()[0]
mask = np.concatenate((mask1, mask2))
mask = np.concatenate((mask, mask3))
mask = [i for i in range(data.values[:,17].shape[0]) if i not in mask]
mask = np.array(mask)
avg = np.int32(data.values[mask,17]).mean()
print(avg)
```

```
89.52054794520548
```



fill nan value with the ceil of the average

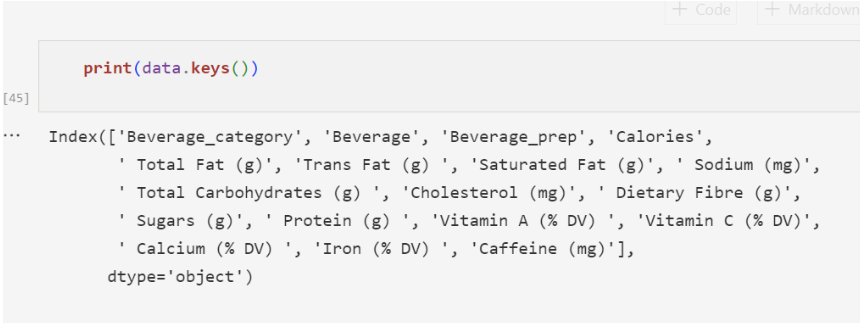
```
data = data.fillna(str(math.ceil(avg)))
print(data.values[158,17])
```

```
90
```

- Notice here the value of nan value exchanged to 90.

- Drop unnecessary columns

These are all columns’ labels:

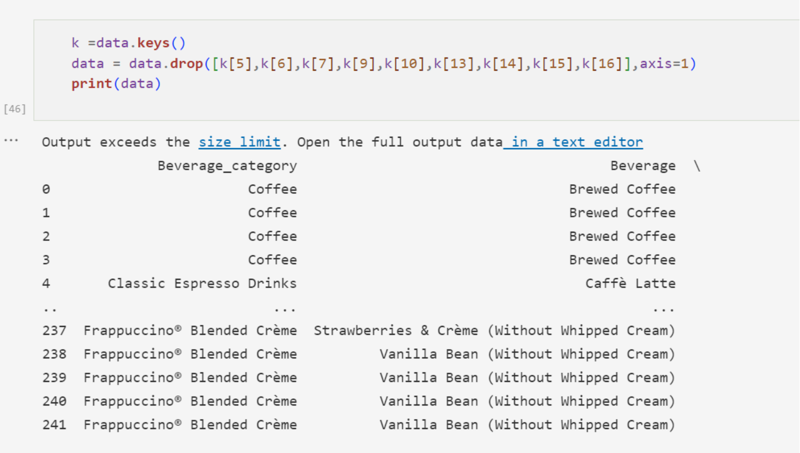


```
print(data.keys())
```

```
Index(['Beverage_category', 'Beverage', 'Beverage_prep', 'Calories',
       ' Total Fat (g)', 'Trans Fat (g) ', 'Saturated Fat (g)', ' Sodium (mg)',
       ' Total Carbohydrates (g) ', 'Cholesterol (mg)', ' Dietary Fibre (g)',
       ' Sugars (g)', ' Protein (g) ', 'Vitamin A (% DV) ', 'Vitamin C (% DV)',
       ' Calcium (% DV) ', 'Iron (% DV) ', 'Caffeine (mg)'],
      dtype='object')
```

- in this problem i suggest that the most important columns (i mean DrinkMenu) are

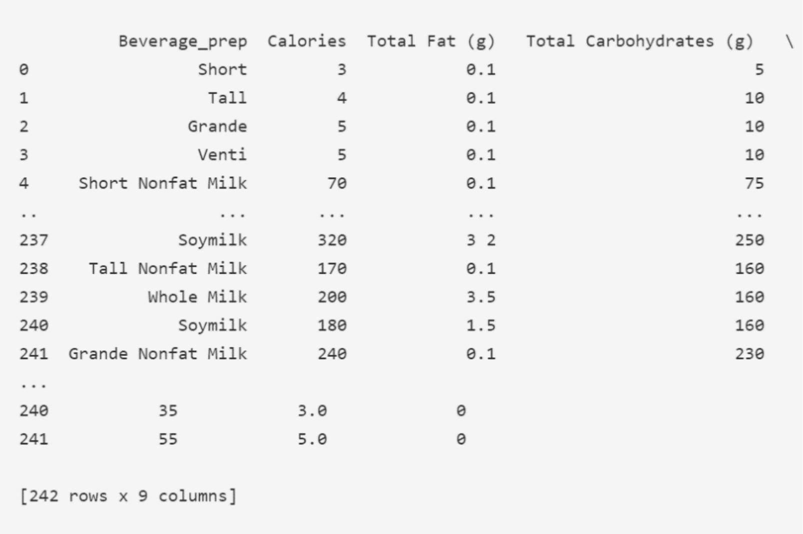
["Beverage_category", "Beverage", "Beverage_prep", "Calories", "Total Fat (g)", "Total Carbohydrates (g)", "Sugars (g)", "Protein (g)", "Caffeine (mg)"], so i will drop the remaining ones.



```
k =data.keys()
data = data.drop([k[5],k[6],k[7],k[9],k[10],k[13],k[14],k[15],k[16]],axis=1)
print(data)
```

[46]

```
Output exceeds the size limit. Open the full output data in a text editor
            Beverage_category                                          Beverage  \
0                      Coffee                                     Brewed Coffee
1                      Coffee                                     Brewed Coffee
2                      Coffee                                     Brewed Coffee
3                      Coffee                                     Brewed Coffee
4     Classic Espresso Drinks                                       Caffè Latte
..                        ...                                               ...
237  Frappuccino® Blended Crème  Strawberries & Crème (Without Whipped Cream)
238  Frappuccino® Blended Crème          Vanilla Bean (Without Whipped Cream)
239  Frappuccino® Blended Crème          Vanilla Bean (Without Whipped Cream)
240  Frappuccino® Blended Crème          Vanilla Bean (Without Whipped Cream)
241  Frappuccino® Blended Crème          Vanilla Bean (Without Whipped Cream)
```

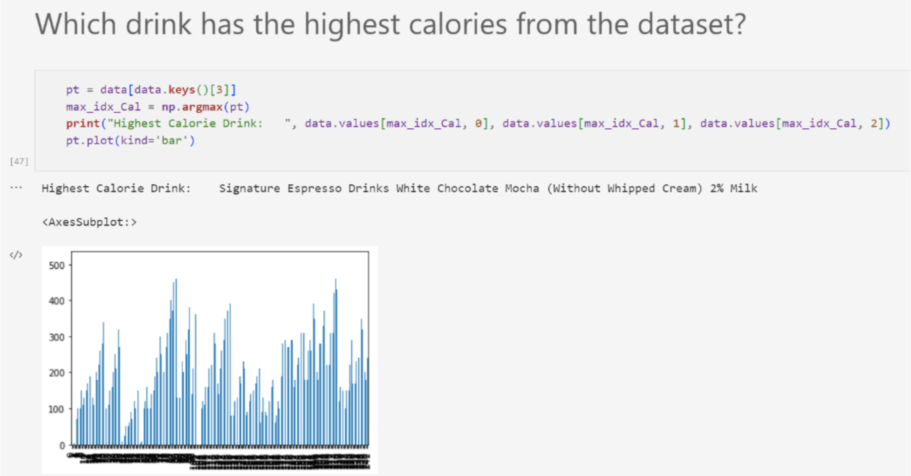


```
          Beverage_prep  Calories  Total Fat (g)   Total Carbohydrates (g)   \
0                 Short         3           0.1                         5
1                  Tall         4           0.1                        10
2                Grande         5           0.1                        10
3                 Venti         5           0.1                        10
4     Short Nonfat Milk        70           0.1                        75
..                  ...       ...           ...                       ...
237             Soymilk       320           3 2                       250
238    Tall Nonfat Milk       170           0.1                       160
239          Whole Milk       200           3.5                       160
240             Soymilk       180           1.5                       160
241  Grande Nonfat Milk       240           0.1                       230
...
240            35            3.0            0
241            55            5.0            0

[242 rows x 9 columns]
```

- notice that the columns dropped from 17 to 9.

- **Data visualization**



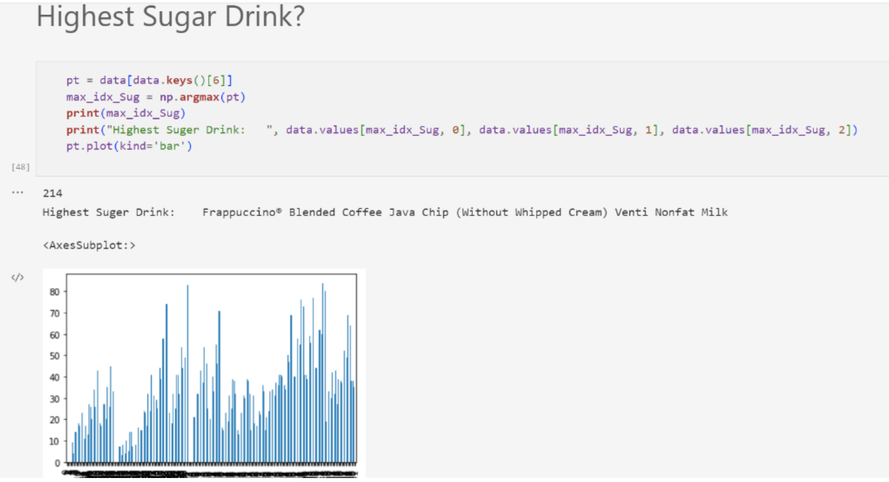
Which drink has the highest calories from the dataset?

```
pt = data[data.keys()[3]]
max_idx_Cal = np.argmax(pt)
print("Highest Calorie Drink:    ", data.values[max_idx_Cal, 0], data.values[max_idx_Cal, 1], data.values[max_idx_Cal, 2])
pt.plot(kind='bar')
```

[47]

```
Highest Calorie Drink:    Signature Espresso Drinks White Chocolate Mocha (Without Whipped Cream) 2% Milk

<AxesSubplot:>
```



Highest Sugar Drink?

```
[48]
    pt = data[data.keys()[6]]
    max_idx_Sug = np.argmax(pt)
    print(max_idx_Sug)
    print("Highest Suger Drink:   ", data.values[max_idx_Sug, 0], data.values[max_idx_Sug, 1], data.values[max_idx_Sug, 2])
    pt.plot(kind='bar')
```

```
214
Highest Suger Drink:    Frappuccino® Blended Coffee Java Chip (Without Whipped Cream) Venti Nonfat Milk

<AxesSubplot:>
```