Tutorial 2

ST2137-2420

Material

This tutorial covers basic Python syntax and data manipulations with `pandas`. It provides practice on chapter 2 from the course textbook.

The following numpy functions may be useful: `np.where`, `np.exp`. The following pandas functions too: `pd.Series.value_counts`.

In the final section, we use the concept of MLE, which you would be familiar from ST2132. It is a method of deriving optimal estimators for distribution parameters.

Dataset: Liverpool

The dataset `liverpool_2223_season.csv` contains information on games that Liverpool Football Club[1] played. The data was obtained from footballref. The team played 19 other teams, and played each of them Home and Away.

1. Read the dataset into Python:

```
import pandas as pd
import numpy as np

liv = pd.read_csv("data/liverpool_2223_season.csv")
liv.head()
```

```
         Date  Day Venue  GF  GA        Opponent
0  2022-08-06  Sat  Away   2   2          Fulham
1  2022-08-15  Mon  Home   1   1  Crystal Palace
2  2022-08-22  Mon  Away   1   2  Manchester Utd
3  2022-08-27  Sat  Home   9   0     Bournemouth
4  2022-08-31  Wed  Home   2   1   Newcastle Utd
```

2. Tabulate the number of times Liverpool played games on the different days.

```
liv.Day.value_counts()
```

```
Day
Sat    17
Mon     7
Sun     7
Wed     5
Fri     1
Tue     1
Name: count, dtype: int64
```

3. Add two columns to the dataset:
   - One string column named `result`, that contains W (for Win), D (for Draw) or L (for Loss).

[1] Liverpool play in the English Premier League.

- One numeric column `pts` that contains the value of 3, 1 or 0, corresponding to a win, draw or loss respectively.

```
result = []
pts = []

for _,y in liv.iterrows():
    if y.GF > y.GA:
        result.append('W')
        pts.append(3)
    elif y.GF == y.GA:
        result.append('D')
        pts.append(1)
    else:
        result.append('L')
        pts.append(0)

liv['result'] = result
liv['pts'] = pts
```

4. Write a `for` loop to produce the following output, which computes the total number of points Liverpool won off each team:

```
i = 0
for x,y in liv.groupby('Opponent'):
    print(f"Points against {y.Opponent.iloc[0]}: {y.pts.sum()}")
    if i == 5:
       break
    i += 1
```

```
Points against Arsenal: 1
Points against Aston Villa: 4
Points against Bournemouth: 3
Points against Brentford: 3
Points against Brighton: 1
Points against Chelsea: 2
```

```
    #print(y.Opponent.iloc[0])
    #print(y.pts.sum())
```

The output above has been truncated. Your output should include all 19 opponents.

5. It is often said that a team needs to accumulate 40 points to be safe from relegation (forced down to a lower division). Add a column that computes the cumulative number of points Liverpool obtains, and use it to identify when Liverpool first accumulated 40 points.

```
liv['cumul_pts'] = liv.pts.cumsum()
id = np.where(liv.cumul_pts >= 40)
liv.iloc[id[0], ].head(1)
```

```
          Date  Day Venue  GF  GA        Opponent result  pts  cumul_pts
24  2023-03-05  Sun  Home   7   0  Manchester Utd      W    3         42
```

6. Use the internet to figure out what this function does. Then use it to compute the length of the longest winning streak that Liverpool had.

```
import re
def fn1(str1):
    out_string = re.split('[LD]+', str1)
    st1 = [len(x) for x in out_string]
    return st1
```

```
rle_out = fn1(liv.result.str.cat())
np.max(rle_out)
```

```
np.int64(7)
```

7. The following questions are on the use of the slice operator in Python. Use .iloc along with the slice operator to retrieve:
    1. The first 10 rows
    2. Alternate rows from the first 10 rows, starting with the first.
    3. Every alternate row, and columns Date, Venue, GF, GA and result.
    4. The last 5 rows.
    5. All rows in reverse order.

```
# First ten rows:
liv.iloc[0:10, ]

# First ten rows, alternate rows:
liv.iloc[0:10:2, ]

# Alternate rows, columns Date, Venue, GF, GA, result
liv.iloc[0::2, [0, 2, 3, 4]]

# Last 5 rows
liv.iloc[-5:,:]

# reverse order
liv.iloc[::-1, ]
```

Truncated Poisson

Suppose that $X \sim Pois(\lambda)$, and the distribution of Y is given by

$$P(Y = y) = P(X = y \,|\, X > 0), \quad y = 1, 2, ...$$

This is known as the truncated Poisson distribution. Given observations $y_1, ..., y_n$, the Maximum Likelihood Estimate $\hat{\lambda}$ is given by the solution to

$$\bar{y} = \frac{\hat{\lambda}}{1 - \exp(-\hat{\lambda})}$$

Suppose we observe the following readings from 30 observations of Y:

$Y = 1$	$Y = 2$	$Y = 3$	$Y = 5$
12	14	3	1

8. Perform a grid search over (0.5, 2) with spacing 0.01 to identify the MLE.

```
## Example taken from MASS textbook
y_bar = (12*1 + 14*2 + 3*3 + 5)/30
# y_bar
lam_range = np.arange(0.5, 2, 0.01)
ll_vals = y_bar - lam_range / (1.0 - np.exp(-lam_range))
id = np.argmin(np.abs(ll_vals))
lam_range[id]
```

```
np.float64(1.3200000000000007)
```