資料分析與學習基石

(Fundamental of Data Analytics and Learning)

Homework 1 - First visit in Kaggle data

指導教授：高宏宇 教授

系級：電機系 111

日期：2021/3/24

- # Dataset description

Title :

League of Legends Ranked Games

(Details from over 50,000 ranked games of LoL)

Brief introduction of LoL：

League of Legends，簡稱 LoL，是一款 5v5 MOBA 遊戲，遊戲中玩家可藉由殺死敵方隊伍的英雄(champion)或防禦塔(Tower)來取得金幣，並運用金錢來購買裝備，勝利目標是要摧毀對方的主要基地「主堡」

Data :

1. champion_info.json :

   basic information for all LoL Champions, key of dict is champion id


```
▼ "root" : { 3 items
  "type" : string "champion"
  "version" : string "7.17.2"
  ▼ "data" : { 138 items
    ▼ "1" : { 4 items
      "title" : string "the Dark Child"
      "id" : int 1
      "key" : string "Annie"
      "name" : string "Annie"
    }
    ▶ "2" : {...} 4 items
    ▶ "3" : {...} 4 items
    ▶ "4" : {...} 4 items
    ▶ "5" : {...} 4 items
```


2. champion_info_2.json :

   basic information for all LoL champs, key of dict is champion name


```
▼ "root" : { 3 items
  "type" : string "champion"
  "version" : string "7.18.1"
  ▼ "data" : { 139 items
    ▶ "None" : {...} 5 items
    ▶ "MonkeyKing" : {...} 5 items
    ▼ "Jax" : { 5 items
      ▼ "tags" : [ 2 items
        0 : string "Fighter"
        1 : string "Assassin"
      ]
      "title" : string "Grandmaster at Arms"
      "id" : int 24
      "key" : string "Jax"
      "name" : string "Jax"
    }
    ▶ "Fiddlesticks" : {...} 5 items
    ▶ "Shaco" : {...} 5 items
```

3. games.csv :

   in-game events of over 50,000 games

gameId	# gameDura...	# winner	# firstBlood	# firstTower	# firstBaron	# firstDragon
3326086514	1949	1	2	1	1	1
3327363504	1493	1	2	1	1	2
3326856598	1758	1	1	1	1	1
3330080762	2094	1	2	1	1	1
3287435705	2059	1	2	2	1	2
3314215542	1993	1	1	2	1	1
3329224025	1334	1	1	1	0	2
3318040883	1387	2	2	2	0	2

4. summoner_spell_info.json :

   basic information for all summoner spells


```
"root" : { 3 items
  "type" : string "summoner"
  "version" : string "7.17.2"
  "data" : { 17 items
    "1" : {...} 5 items
    "3" : {...} 5 items
    "4" : { 5 items
      "id" : int 4
      "summonerLevel" : int 8
      "name" : string "Flash"
      "key" : string "SummonerFlash"
      "description" : string "Teleports your champion a short distance toward your cursor's location."
    }
    "6" : {...} 5 items
```


Data General Info :

- **Game ID**：該局遊戲的編號
- **Creation Time** (in Epoch format)：該局遊戲開始時間
- **Game Duration** (in seconds)：該局遊戲時長
- **Season ID**：第幾賽季，蒐集到的資料皆為第 9 賽季
- **Winner** (1 = team1, 2 = team2)：勝利隊伍
- **First Baron, dragon, tower, blood, inhibitor and Rift Herald** (1 = team1, 2 = team2, 0 = none)：先擊殺 baron 的隊伍、先擊殺 dragon 的隊伍、…　(firstblood：先擊殺敵方英雄的隊伍)
- **Champions and summoner spells for each team** (Stored as Riot's champion and summoner spell IDs)：各英雄的召喚師技能
- **The number of tower, inhibitor, Baron, dragon and Rift Herald kills each team has**：隊伍摧毀的防禦塔總數、隊伍摧毀的 inhibitor 總數、…
- **The 5 bans of each team** (Again, champion IDs are used)：選擇英雄前各隊可以禁止 5 隻英雄被選擇

Possible Uses :

There is a vast amount of data in just a single LoL game. This dataset takes the most relevant information and makes it available easily for use in things such as attempting to predict the outcome of a LoL game, analysing which in-game events are most likely to lead to victory, understanding how big of an effect bans of a specific champion have, and more.

- # Data analysis

Notebook1

Code Title:

Let's Predict League of Legends Match Score!

Goal :

In this kernel, League of Legends ranked matches were analyzed and a decision tree classification algorithm was developed to predict match scores. To develop this algorithm:

- Winner (1 = team1, 2 = team2)
- First Baron, dragon, tower, blood, inhibitor and Rift Herald (1 = team1, 2 = team2, 0 = none)
- The number of tower, inhibitor, Baron and dragon kills each team has

(挑出對勝負影響較大的 feature，建立決策樹預測遊戲結果)

Step :

1. **DATA ANALYSIS**

   直接存取需要的資料，因為資料形式簡單，沒有另外處理

In [2]:
```
data=lol[["winner","firstBlood","firstTower","firstInhibitor","firstBaron","firstDrago
n","firstRiftHerald","t1_towerKills","t1_inhibitorKills",'t1_baronKills',
         't1_dragonKills','t2_towerKills','t2_inhibitorKills', 't2_baronKills', 't2_dragonKi
lls'
         ]]
```

shows first 5 entries of dataset.

In [4]:
```
data.head()
```

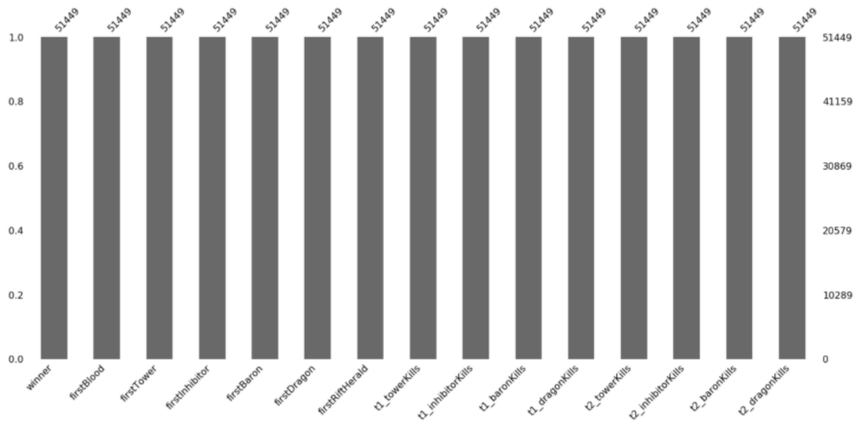
Out[4]:

	winner	firstBlood	firstTower	firstInhibitor	firstBaron	firstDragon	firstRiftHerald	t1_towerKills	t1_inhibitorKills
0	1	2	1	1	1	1	2	11	1
1	1	1	1	1	0	1	1	10	4
2	1	2	1	1	1	2	0	8	1
3	1	1	1	1	1	1	0	9	2
4	1	2	1	1	1	1	0	9	2

用 missingno()確認有沒有 missing data

Hide code

In [5]:
```
import missingno as msno
msno.bar(data)
plt.show()
```



Following subplots shows probabilities of Different Features when a team wins.
For example

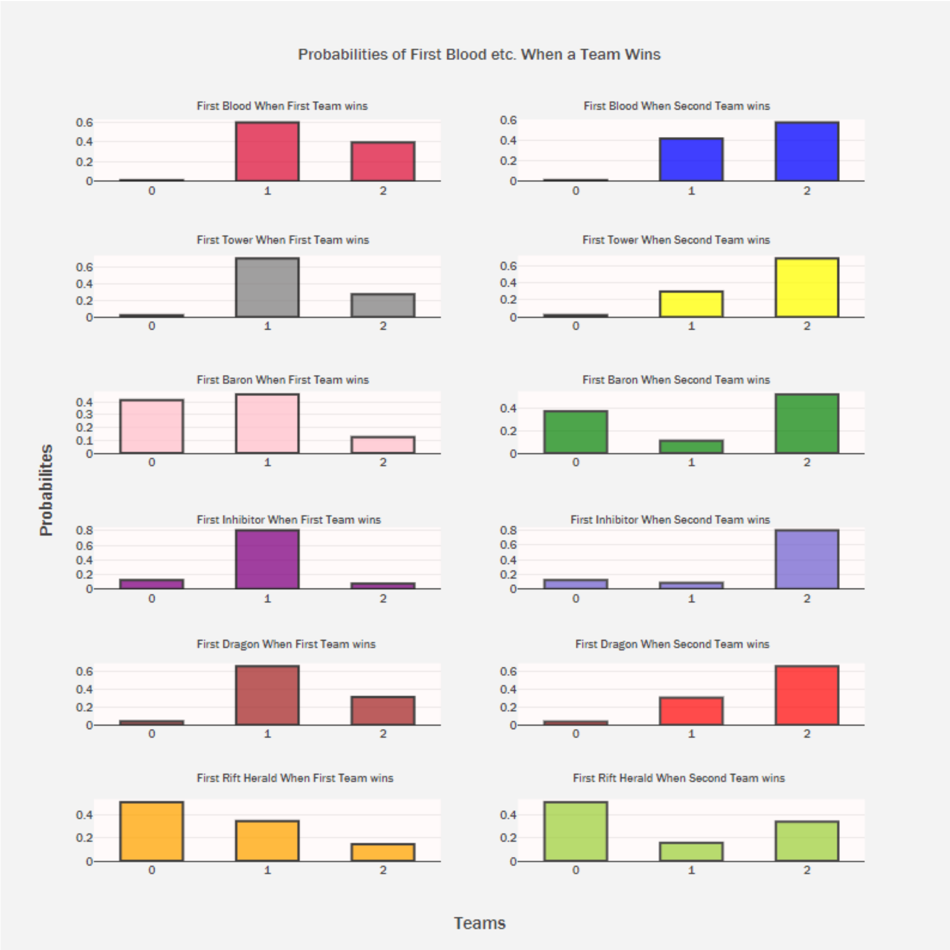
In matches with first team win:

The probability of taking first tower of first team is about 70%.
The probability of taking first tower of second team is about 27%.
The probability of taking first tower of any team is about 2% (surrender)

In matches with second team win:

The probability of taking first blood of first team is about 30%.
The probability of taking first blood of second team is about 68%.
The probability of taking first blood of any team is about 2% (surrender)

可以看出獲勝的一方有較高機率拿到首個物件或擊殺



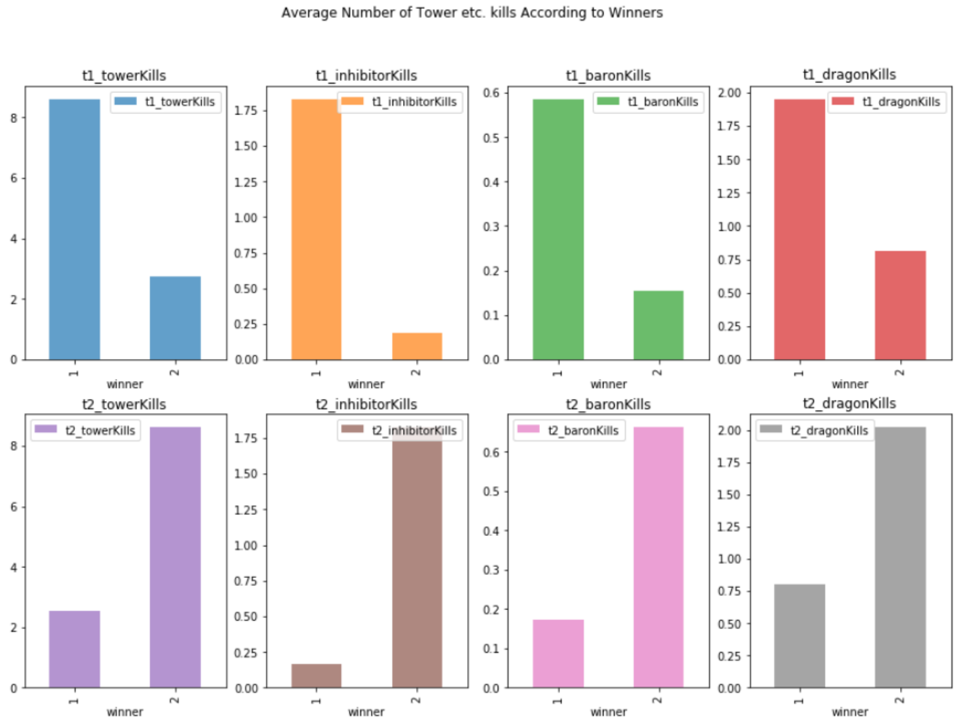
底下的圖顯示當其中一方勝利時，雙方的 average number of tower, inhibitor, Baron and dragon kills

可以清楚看出贏的一方所有物件擊殺數都會高於輸的一方

In [8]:
```
data_new=data[["winner","t1_towerKills","t1_inhibitorKills","t1_baronKills","t1_dragonKills","t2_towerKills","t2_inhibitorKills","t2_baronKills","t2_dragonKills"]]
data_new.groupby("winner").mean()
```

Out[8]:

	t1_towerKills	t1_inhibitorKills	t1_baronKills	t1_dragonKills	t2_towerKills	t2_inhibitorKills	t2_baronKills	t2_dra
winner								
1	8.607006	1.830696	0.586188	1.953142	2.558381	0.166270	0.172914	0.804
2	2.729627	0.186086	0.153740	0.809586	8.622831	1.825562	0.662771	2.021



2. **Decision Tree with Grid Search Method**

To develop best model we searched best parameters. To find that we used Grid Search Method.

In [11]:

```
criterion=["gini","entropy"]
max_depth=range(1,20,2)
splitter=["best","random"]
dt=DecisionTreeClassifier()
grid_decision_tree=GridSearchCV(estimator=dt,cv=15,param_grid=dict(criterion=criterion,max
_depth=max_depth,splitter=splitter))
```

In [12]:

```
grid_decision_tree.fit(x_train,y_train)
print("best score: ", grid_decision_tree.best_score_)
print("best param: ", grid_decision_tree.best_params_)
```

```
best score:  0.9693174876436941
best param:  {'criterion': 'entropy', 'max_depth': 7, 'splitter': 'best'}
```

And we found best parameters : criterion parameter as entropy , max depth is 7 and splitter is best. And then we test our model.

將數據給入決策樹中，得出可以利用 in-game data 預測勝負的模型並且有 96.6% accuracy

In [13]:

```
dt2=DecisionTreeClassifier(criterion="entropy",max_depth=7, splitter="best")
dt2.fit(x_train,y_train)
print("score:", dt2.score(x_test,y_test))
```

```
score: 0.9663103336572725
```

利用 classification]_report_驗證 model 準確性

In [15]:

```
from sklearn.metrics import confusion_matrix,classification_report
predicted_values = dt2.predict(x_test)
cm=confusion_matrix(y_test,predicted_values)
cr=classification_report(y_test,predicted_values)
print('Classification report : \n',cr)
```

```
Classification report :
               precision    recall  f1-score   support

           1       0.96      0.97      0.97      7917
           2       0.97      0.96      0.97      7518

   micro avg       0.97      0.97      0.97     15435
   macro avg       0.97      0.97      0.97     15435
weighted avg       0.97      0.97      0.97     15435
```

嘗試代入數據預測勝利隊伍

Out[17]:

	0	1	2	3	4	5	6	7	8
feature	first_blood	first_tower	first_inhibitor	first_Baron	first_Dragon	first_RiftHerald	t1_tower	t1_inhibitor	t1_barc
value	1	1	2	1	1	1	10	2	1

Hide code

In [18]:

```
x1=[[1,1,2,1,1,1,10,2,1,4,7,2,1,1]]
c=dt2.predict_proba(x1).reshape(-1,1)
print("winner is :" , dt2.predict(x1) )
print("first team win probability is % ", list(c[0]*100),"\nsecond team win probability is
%:",list(c[1]*100)  )
```

```
winner is : [1]
first team win probability is %  [85.39325842696628]
second team win probability is %: [14.606741573033707]
```

Our model says The winner will be First Team with 85% probability.

Notebook2

Code Title:

League of Legends data analysis

Goal :

資料整理成較易閱讀的形式，提供問題讓其他人有研究方向

1. make a countplot for total champion picks and bans over the entire dataset
2. make a countplot of types of champions and summoner spells used

Step :

整理出所有被選擇、被 ban 的英雄和召喚師技能

In [9]:

```
data[champCols].head()
```

Out[9]:

	t1_champ1id	t1_champ2id	t1_champ3id	t1_champ4id	t1_champ5id	t2_champ1id	t2_champ2id	t2_champ3id
0	Vladimir	Bard	Kog'Maw	Master Yi	Viktor	Graves	Xayah	Darius
1	Draven	Irelia	Nidalee	Kayle	Shaco	Malphite	Morgana	Hecarim
2	Tristana	Kayn	Nami	Rumble	Kassadin	Cassiopeia	Thresh	Jayce
3	Maokai	Brand	Twitch	Orianna	Dr. Mundo	Malzahar	Warwick	Thresh
4	Warwick	Twitch	Janna	Draven	Syndra	Sona	Jarvan IV	Kayn

In [10]:

```
data[banCols].head()
```

Out[10]:

	t1_ban1	t1_ban2	t1_ban3	t1_ban4	t1_ban5	t2_ban1	t2_ban2	t2_ban3	t2_ban4	t2_ban5
0	Riven	Janna	Cassiopeia	Draven	Kayn	Fiora	Vayne	Karma	Soraka	Caitlyn
1	Caitlyn	Darius	Teemo	Xayah	Warwick	Master Yi	Vayne	Zed	Caitlyn	Illaoi
2	Lulu	Janna	Twitch	Soraka	Blitzcrank	Yasuo	Zed	Kha'Zix	Maokai	Evelynn
3	Zed	Vayne	Ornn	Fiora	Cho'Gath	Camille	Tristana	Kayn	Janna	Caitlyn
4	Malzahar	Lee Sin	Thresh	Morgana	Cho'Gath	Garen	Master Yi	Braum	Darius	Tristana

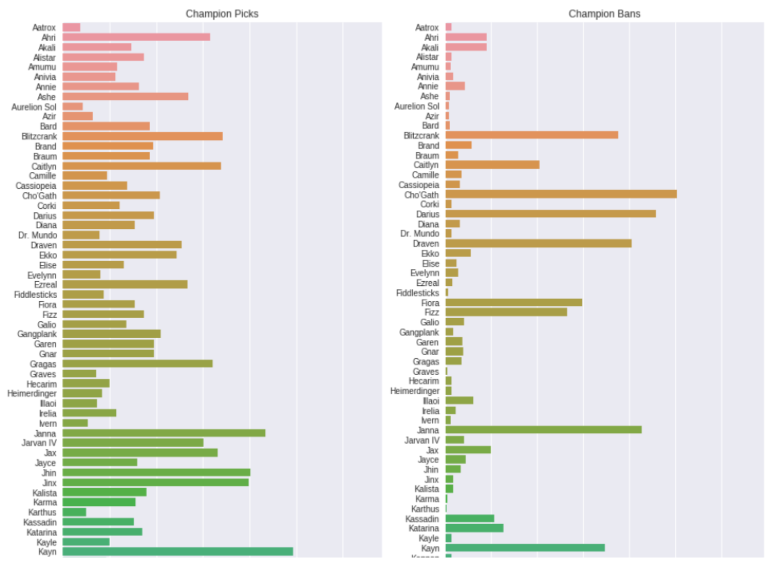
In [11]:

```
data[sumSpellsCols].head()
```

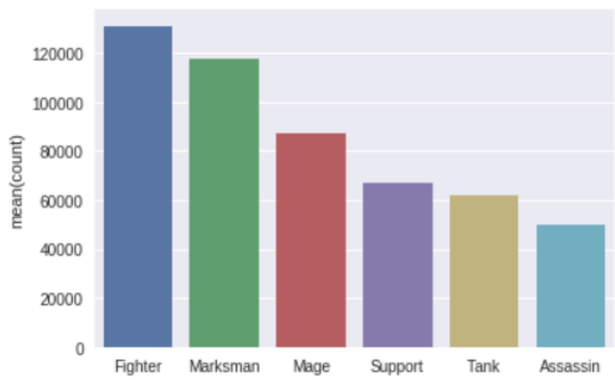
Out[11]:

	t1_champ1_sum1	t1_champ1_sum2	t1_champ2_sum1	t1_champ2_sum2	t1_champ3_sum1	t1_champ3_sum2	t1_c
0	Teleport	Flash	Exhaust	Flash	Flash	Heal	Smit
1	Heal	Flash	Teleport	Flash	Flash	Exhaust	Flas
2	Flash	Heal	Smite	Flash	Exhaust	Flash	Flas
3	Flash	Teleport	Flash	Ignite	Flash	Heal	Flas
4	Flash	Teleport	Smite	Flash	Flash	Exhaust	Flas

用圖表呈現



統計各種類型的英雄出場次數



- **notebook comparison**

2 者對於這個數據的分析目標完全不同，Notebook2 著重在將資料轉換成更易於分析的模樣，供後人使用，Notebook1 則著重在利用遊戲內的資訊預測遊戲結果。

Notebook2 就像是開發工具，Notebook1 就像是開發者，Notebook2 可以讓 Notebook1 的開發更輕鬆、更有效率。

- **資料分析價值與可能產出**

1. 對於 LoL 遊戲開發者，能夠從各種角色選取率與勝率判斷哪些角色較受歡迎或是較強勢，可以將其削弱或是推出該角色的造型賺錢
2. 對於電競隊伍可以分析各種地圖物件對於勝利的影響力，進而決定戰術重點，讓隊伍競爭力提升

- **我的觀點**

原先想到資料分析都會像是 Notebook1 一樣，想要將資料轉換成一個預測模型，用原有的資訊預測未來，看完這個資料集才發現，原來像 Notebook2 一樣將資料整理好也是一件不容易的事情，甚至比起預測我更想要擁有這個技能，先整理資料更能有條理地分析數據，也能更清楚向別人描述自己的想法。

對於 Kaggle 上居然有這種資料集，我感到滿興奮的，居然連這種看似沒什麼價值的資料都有人蒐集和分析，但是這就表示無論什麼東西都是有價值的，端看自己看待它的角度

目前的資料集通常都很龐大，對於使用者來說，這些資料集有很多種詮釋方式，有不同的需求就有不同的使用方法，完全可以依照自己的需求決定要如何利用這些資源。而我們目前最需要的就是利用這些資料的能力，培養自己獨特的對於資料的見解，適當、精準的運用手上的資源，將它轉換成有價值的資訊。