sample:ドミノピザの当選数の推移の集計

2019 年 11 月 30 日

1　はじめにこのパッケージの簡単な説明

思いつくままに書いているので、まとまりがないですがお許しください。主に markdown の対応について書いています。

最初のセルの部分にコメントをつけてみる。例えば今回の場合、

```
# 27;figureoption:keepaspectratio,width=0.9\hsize
# 32;figureoption:keepaspectratio,width=0.75\hsize
```

と書いています。「#セル番号;figureoption:オプション」とすることで、一番はじめのセルを 1 番として、指定番目のセルで表示されている tex の figure の

```
\includegraphics[ここ！]{なんか.png}
```

ここ！の部分の設定を指定できます。また、このようにコードを markdown の方に貼り付けも使えます。

また、引用もできます。

> tex の quote に対応させています。

また、この pdf のもととなった google colaboratory のリンクはこれなのですが、このようにリンクにも対応しています。また、

- itemize
- enumerate

については、このような一段階までなら対応していますが、二段以上の入れ子は対応していません。また、**強調**、*italic*（ただし英語のみ）にも対応しています。

線も一応引けます。

シャープ は 1 つで section

1.1　2 つで subsection

1.1.1　3 つで subsubsection

■4 つで pragraph

5 つで subparagraph に対応します。 また、このように

分布名	ポアソン分布	ガウス分布
確率密度関数	$\frac{\mu^x}{x!}e^{-\mu}$	$\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$
平均値	μ	μ
分散	μ	σ^2
—	—	—

テーブルにも対応しています。python コード実行によって作成されたテーブルも、うまく出力されるはずです（iminuit のフィッティングによるものはしっかりと表示できました。）さらに、数式に対応していることもわかると思います。こういうタイプのものもいけるということで、とりあえずオイラーの等式をだしてみます。

$$e^{i\pi} + 1 = 0$$

ただし、equation を使おうとすると、エラーを吐く場合があるので、うまくあとで調整してください。

以下は、実際のレポートのようなものです。雰囲気をなんとなくでも掴めるのではないでしょうか。

※このレポートはデモのために適当に作成したものです。変数名の雑さなどには目を瞑ってください。ただし、取ったデータなどは実際のものです。

2 概要

以前のドミノピザの RT キャンペーンで、当たりの数や出る時間に明らかな偏りがあるように思えたため、軽く統計をとって調べてることにした。

3 データの取得方法

別途用意した python プログラム（後述）で適当に twitter の api 叩いてドミノピザ公式が当たり、はずれ、という画像とともにリプライを飛ばしてるツイートを取得して集計。当たりなら 1、はずれなら 0 を txt ファイルにタイムスタンプと一緒に記録。cron による定期実行、2 分毎に直近 2 分間最大 3200 件のツイートの結果が各々の txt ファイルに記録されるようにした。

3.1 txt ファイル作成に使用したプログラム

twitter からのテータの収集には tweepy を使用しています。

```
1  import tweepy
2  import datetime
3  import re
4
5
6  CK="Consumer Key"
7  CS="Consumer Secret"
8  AT="Access Token"
9  AS="Access Token Secret"
10
11
```

```
auth = tweepy.OAuthHandler(CK, CS)
auth.set_access_token(AT, AS)
api = tweepy.API(auth)


dt_now = datetime.datetime.now()
dt_L = datetime.datetime(*dt_now.timetuple()[:5]) - datetime.timedelta(minutes=2)
dt_R = datetime.datetime(*dt_now.timetuple()[:5])
filename = dt_now.strftime('\%y\%m\%d\%H\%M\%S') + '.txt'


with open(filename,'w') as f:
    for i in range(1,17):
        res = api.user_timeline(screen_name="dominos_JP", count=200, page=i)


        for tweet in res:
            timestamp = datetime.datetime.fromtimestamp(((tweet.id >>
                22)+1288834974657)/1000.0)
            if dt_L <= timestamp:
                if timestamp < dt_R:
                    if re.search(r'残念',tweet.text):
                        f.write(str(timestamp).split()[1] + ' 0' + '\n')
                    elif re.search(r'当選',tweet.text):
                        f.write(str(timestamp).split()[1] + ' 1' + '\n')
            else:
                break
        else:
            continue
        break
```

4　2019 年 10 月 30 日の集計結果

この日は このキャンペーン の最終日でした。

4.1　下準備

分析にあたり、モジュールの読み込みを行っています。

In [1]:
```
import os
import pandas as pd
import numpy as np
from scipy import interpolate
import matplotlib.pyplot as plt
from matplotlib import dates as mdates
from datetime import datetime as dt
```

google colaboratory 上のディレクトリにあらかじめ全 txt ファイルが存在しています。

In [2]:
```
files=os.listdir('/content')
```

In [3]:
```
files.remove('sample_data')
files.remove('.config')
```

適当にデータを作成します。

In [4]:
```
data=[]
for file in files:
  with open('/content/'+file, encoding="utf-8") as f:
    for line in f:
      line.strip('\n')
      data.append(line.split())
```

In [5]:
```
data=[[dt.strptime('2019-10-29 '+[datum[0][:-3],datum[0]+'.000'][len(datum[0])<9], '%Y-%m-%d %H:%M:%S.%f'),int(datum[1])] for datum in data]
data=sorted(data)
```

In [6]:
```
x=[a[0] for a in data]
```

In [7]:
```
y=[a[1] for a in data]
```

In [8]:
```
lll=0
rrr=0
for i in range(len(data)):
  if data[i][0]>dt.strptime('2019-10-29 17:00','%Y-%m-%d %H:%M'):
    lll=i
    break
```

取得した全データ数は以下のようになっています。

In [9]:
```
len(data)
```

Out [9]: 48259

4.2 時間 vs 当たりはずれのプロット

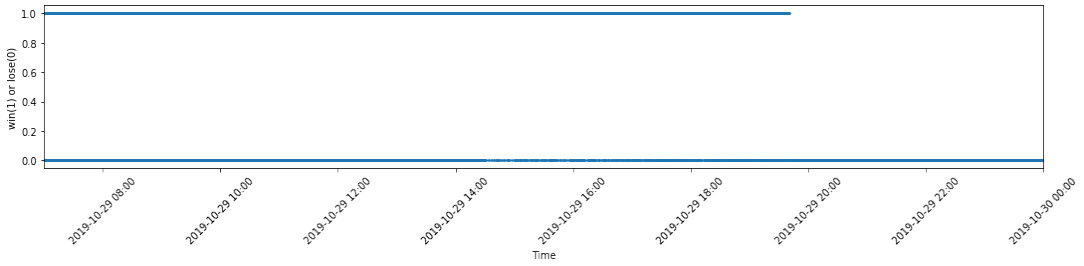
当たりを 1、はずれを 0 に、横を時間軸としてプロットしたものです。明らかに後半は当たりが出ていません。

In [10]:
```
countwin=y_win_sum[-1]-y_win_sum[lll]
countlose=y_lose_sum[-1]-y_lose_sum[lll]
```

In [11]:
```
fig, ax = plt.subplots(figsize=(18,3))

ax.scatter(x,y,s=1)

ax.xaxis.set_major_locator(mdates.HourLocator(interval=2))
ax.xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d %H:%M"))
ax.set_xlim([dt.strptime('2019-10-29 7:00','%Y-%m-%d %H:%M'), dt.strptime('2019-10-30 00:00','%Y-%m-%d %H:%M')])

ax.set_ylabel("win(1) or lose(0)")
ax.set_xlabel("Time")

#ax.annotate("Win:%d" % countwin, xy = (dt.strptime('2019-10-29 17:30','%Y-%m-%d %H:%M'), 1), size = 10, color='r')
#ax.annotate("Lose:%d" % countlose, xy = (dt.strptime('2019-10-29 17:30','%Y-%m-%d %H:%M'), 0), size = 10, color='r')

labels = ax.get_xticklabels()
```

```
plt.setp(labels, rotation=45, fontsize=10)

plt.show()
```

Out [11]:



4.3 時間 vs 累計で推移をみる

In [12]:
```
y_win=[a[1] for a in data]
y_win_sum=[int(y_win.pop(0))]
for yi in y_win:
  y_win_sum.append(y_win_sum[-1] + yi)
```

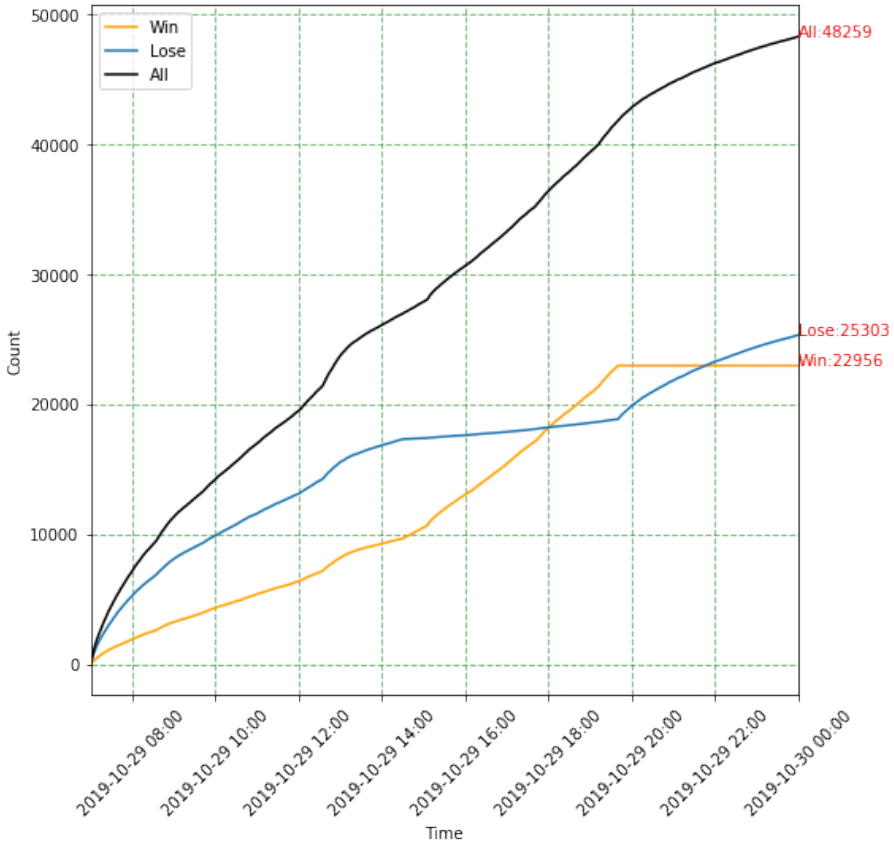
In [13]:
```
y_lose=[1^a[1] for a in data]
y_lose_sum=[int(y_lose.pop(0))]
for yi in y_lose:
  y_lose_sum.append(y_lose_sum[-1] + yi)
```

In [14]:
```
y_sum=[i+1 for i in range(len(data))]
```

In [15]:
```
fig, ax = plt.subplots(figsize=(8, 8))

ax.plot(x,y_win_sum,"-",label="Win",color="orange")
ax.plot(x,y_lose_sum,"-",label="Lose")
ax.plot(x,y_sum,"-",label="All",color="black")

ax.xaxis.set_major_locator(mdates.HourLocator(interval=2))
ax.xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d %H:%M"))
ax.set_xlim([dt.strptime('2019-10-29 07:00','%Y-%m-%d %H:%M'), dt.strptime(
    '2019-10-30 00:00','%Y-%m-%d %H:%M')])

ax.grid(color = "green", alpha = 0.5, linestyle = "-", linewidth = 1)

ax.set_ylabel("Count")
ax.set_xlabel("Time")

labels = ax.get_xticklabels()
plt.setp(labels, rotation=45, fontsize=10)

ax.annotate("Win:%d" % y_win_sum[-1], xy = (dt.strptime('2019-10-30 00:00','%Y-%m-%d %
    H:%M'), y_win_sum[-1]), size = 10, color='r')
ax.annotate("Lose:%d" % y_lose_sum[-1], xy = (dt.strptime('2019-10-30 00:00','%Y-%m-%d
    %H:%M'), y_lose_sum[-1]), size = 10, color='r')
ax.annotate("All:%d" % y_sum[-1], xy = (dt.strptime('2019-10-30 00:00','%Y-%m-%d %H:%
```

```
        M'), y_sum[-1]), size = 10, color='r')

ax.legend()
plt.show()
```

Out [15]:



グラフを見てみると

- はじめのうちは当たり：外れ＝2：1くらいで出ている
- 中盤から当たりが外れに比べてだいぶ出やすくなった
- **終盤は出せる当たりが尽きたためか？はずれしかでなくなっている**

ことがわかります。**なんだか露骨でおもしろい！**