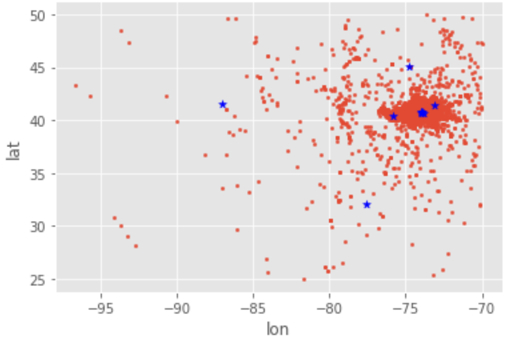
巨量資料hw1

309552063吳冠潔

the scale of data : 資料總數有41859906行以及18列
analytical tools : python pandas

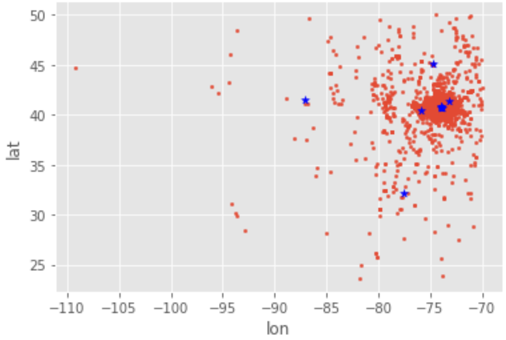
- Q1: What regions have the most pickups? What are the top-5 regions with the most pickups and drop-offs (pickups and drop-offs should be counted separately)?

  1. pickup

     由圖上可知，最多人上車的地方為經緯度約(-73.9 40.7)，最多人上車的前五個地方在經緯度(-73.96 40.78), (-73.13 41.36), (-73.996 40.73), (-73.78 40.72), (-73.98 40.76)的附近

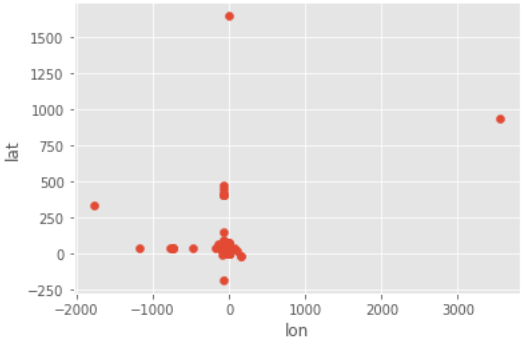
  2. dropoff

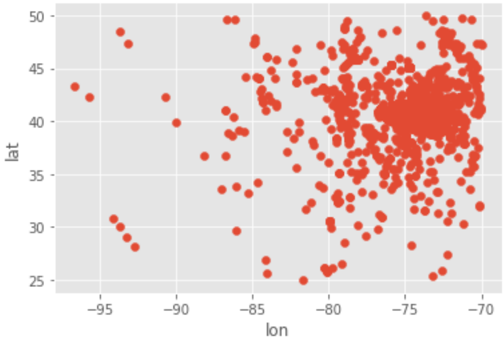
     由圖上可知，最多人下車的地方為經緯度約(-73.9 40.7)，最多人下車的前五個地方在經緯度(-73.99 40.73), (-73.96 40.78), (-73.972 40.679), (-73.98 40.757), (-73.88 40.76)的附近

  - how I solve this question :
    一開始先將1-3月的資料匯入到pandas中，並合併再一起，再來將pickup、dropoff的經緯度做成array，並將經緯度錯誤的資料刪除，有許多的資料經緯度不在範圍內

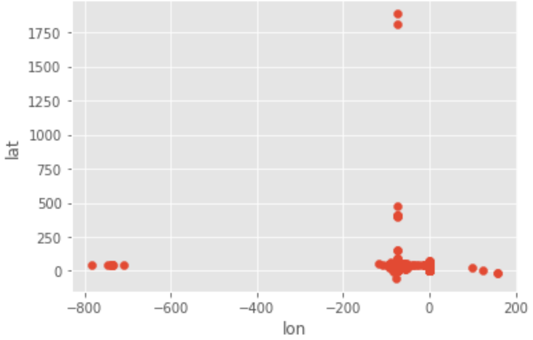
- pickup原始未刪除錯誤經緯度的圖



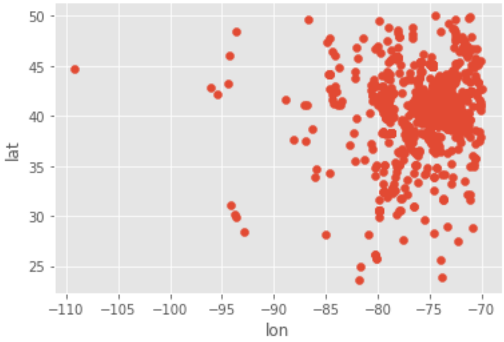
- pickup 刪除後的圖



- dropoff原始未刪除錯誤經緯度的圖



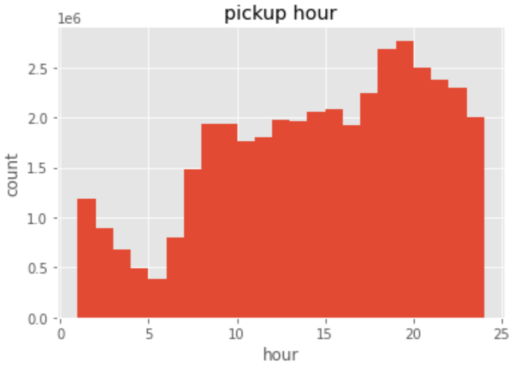
- dropoff 刪除後的圖



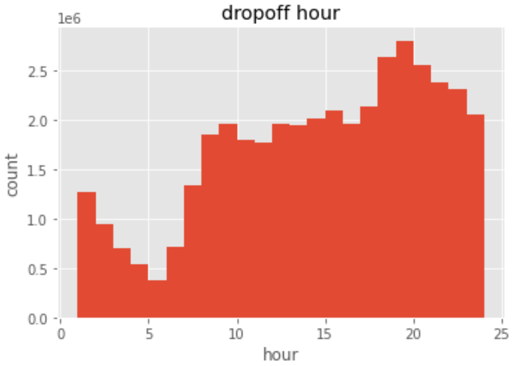
接著用kmeans取出10個cluster，並觀察其中心點，取得5個最多人上下車的區域

- Q2: When are the peak hours and off-peak hours for taking a taxi?
  - peak hours : 17:00~22:00
  - off-peak hours : 1:00~7:00

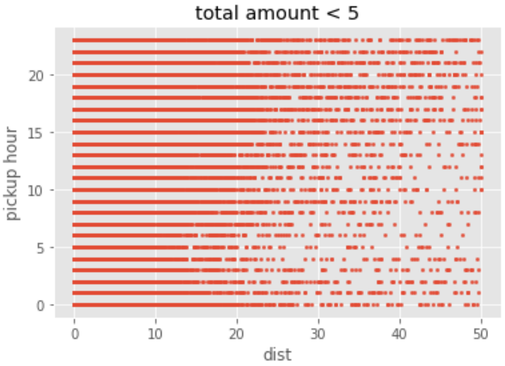
- how I solve this question : 取出pickup, dropoff時間中的小時資料，並繪製成直方圖，看哪些時段多人哪些時段少人
    - pickup



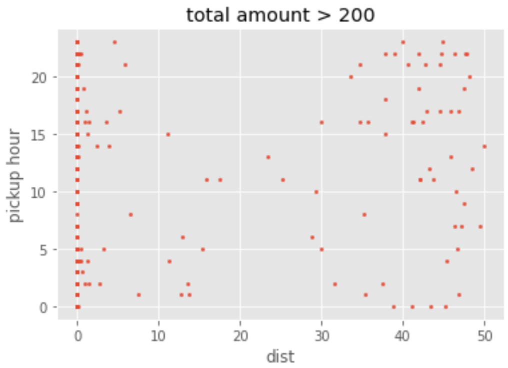
    - dropoff



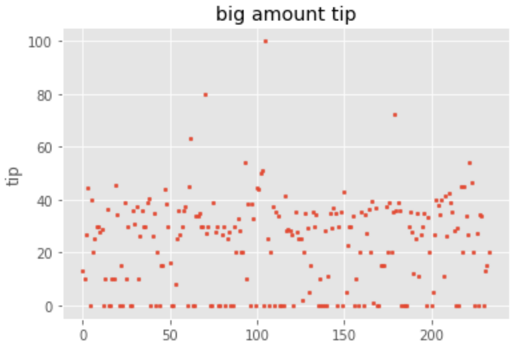
- Q3: What are the differences between big and small total amounts when taking a taxi?
  - 一開始先將small total amounts定義為5元以下，big total amounts定義為200元以上，並分別紀錄距離及pickup hour
    - total amount < 5



        - total amount > 200



    - 初始我把big/small amoun分別定為100/50，但畫出來的圖很接近，可以看出基本大家的乘車費都在那之間，所以我進一步將其範圍縮小，我拿距離及上車時間作為判斷標準，因為搭計程車中最會影響費用的為距離，另一個我用上車時間做為標準，想看出是否在不同時間收費的標準會有不同，在定義small amount時我從50，縮減到10，再進一步縮減到5，但是數量都還是很大，在兩張圖中也看不出顯著差異；於是我進一步將total amount > 200所拿到的tip畫出來
        - tip (total amount > 200)



    - 車費超過200元的，在小費上都會給很多，光是小費就高於定義的small amount，可以看出，雖然距離相差不多，但車費相差多也有出自於客人給的小費比較多

- Difficulties Encountered
  一開始我使用pandas，並用geopy來獲取正確的地理位置，但會有runtime error的問題，後來改用pyspark，但因為不熟悉pyspark，導致出現很多error，需要查一些資料才能解決問題，但在使用kmeans時會有記憶體問題出現，所以最後還是用pandas，並用kmeans來取得最多上下車地點，雖然無法得出確切地址，但還是能獲得其中心經緯度

- source code

```python
import pandas as pd
import numpy as np
from glob import glob
from sklearn.cluster import KMeans
import time
import matplotlib.pyplot as plt
from matplotlib import style
import time
style.use("ggplot")

files = glob('yellow_tripdata_2009*.csv')
print(files)
df = pd.concat((pd.read_csv(file) for file in files))

print(df.columns)
df.head(1)

# Q1
# pickup
sLon = df['Start_Lon']
sLat = df['Start_Lat']

plt.scatter(sLon, sLat)
plt.xlabel('lon')
plt.ylabel('lat')
plt.show()

# lon(-70~-130) lat(23.5~50)
# delete data that out of range
delIndex1 = np.append(np.where(sLon < -130), np.where(sLon > -70))
delIndex2 = np.append(np.where(sLat < 23.5), np.where(sLat > 50))
delIndex = np.append(delIndex1, delIndex2)

tpickupLoc = np.column_stack((sLon, sLat))
pickupLoc = np.delete(tpickupLoc, delIndex, axis=0)

plt.scatter(pickupLoc[:,0], pickupLoc[:,1])
plt.xlabel('lon')
plt.ylabel('lat')
plt.show()

# use kmeans
km_fit = KMeans(n_clusters=10).fit(pickupLoc)
cluster_centers = km_fit.cluster_centers_
print(cluster_centers)

plt.scatter(pickupLoc[:,0], pickupLoc[:,1], s=5)
plt.scatter(cluster_centers[:,0], cluster_centers[:,1], marker='*', c='b')
plt.xlabel('lon')
plt.ylabel('lat')
plt.show()

# dropoff
eLon = df['End_Lon']
eLat = df['End_Lat']

plt.scatter(eLon, eLat)
plt.xlabel('lon')
plt.ylabel('lat')
plt.show()

# lon(-70~-130) lat(23.5~50)
# delete data that out of range
delIndex1 = np.append(np.where(eLon < -130), np.where(eLon > -70))
delIndex2 = np.append(np.where(eLat < 23.5), np.where(eLat > 50))
delIndex = np.append(delIndex1, delIndex2)

tdropoffLoc = np.column_stack((eLon, eLat))
dropoffLoc = np.delete(tdropoffLoc, delIndex, axis=0)

plt.scatter(dropoffLoc[:,0], dropoffLoc[:,1])
plt.xlabel('lon')
plt.ylabel('lat')
plt.show()

# use kmeans
km_fit = KMeans(n_clusters=10).fit(dropoffLoc)
cluster_centers = km_fit.cluster_centers_
print(cluster_centers)

plt.scatter(dropoffLoc[:,0], dropoffLoc[:,1], s=5)
plt.scatter(cluster_centers[:,0], cluster_centers[:,1], marker='*', c='b')
plt.xlabel('lon')
plt.ylabel('lat')
plt.show()

# Q2
time_format = '%Y-%m-%d %H:%M:%S'

time_hour = []
for i in range(1, 25):
    time_hour.append(i)
print(time_hour)

def get_hour(t):
    h = time.strptime(t, time_format).tm_hour
    return h

pickupTime = list(map(get_hour, df['Trip_Pickup_DateTime']))
dropoffTime = list(map(get_hour, df['Trip_Dropoff_DateTime']))

plt.hist(pickupTime, bins=time_hour)
plt.title('pickup hour')
plt.xlabel('hour')
plt.ylabel('count')
plt.show()

plt.hist(dropoffTime, bins=time_hour)
plt.title('dropoff hour')
plt.xlabel('hour')
plt.ylabel('count')
plt.show()

# Q3
t_totalAmt = df['Total_Amt']
t_dist = df['Trip_Distance']
totalAmt = t_totalAmt.to_numpy()
dist = t_dist.to_numpy()

small_x = []
small_y = []
for i in range(len(totalAmt)):
    if totalAmt[i] < 5:
        small_x.append(dist[i])
        small_y.append(pickupTime[i])

plt.scatter(small_x, small_y, s=5)
plt.title('total amount < 5')
plt.xlabel('dist')
plt.ylabel('pickup hour')
plt.show()

big_x = []
big_y = []
for i in range(len(totalAmt)):
    if totalAmt[i] > 200:
        big_x.append(dist[i])
        big_y.append(pickupTime[i])

plt.scatter(big_x, big_y, s=5)
plt.title('total amount > 200')
plt.xlabel('dist')
plt.ylabel('pickup hour')
plt.show()

t_tip = df['Tip_Amt']
tip = t_tip.to_numpy()
big_tip = []
tx = []

j = 0
for i in range(len(totalAmt)):
    if totalAmt[i] > 200:
        tx.append(j)
        j += 1
        big_tip.append(tip[i])

plt.scatter(tx, big_tip, s=7)
plt.title('big amount tip')
plt.ylabel('tip')
plt.show()
```