

MLFN Lab2 Report

309552012 洪立宇

Data Preprocessing

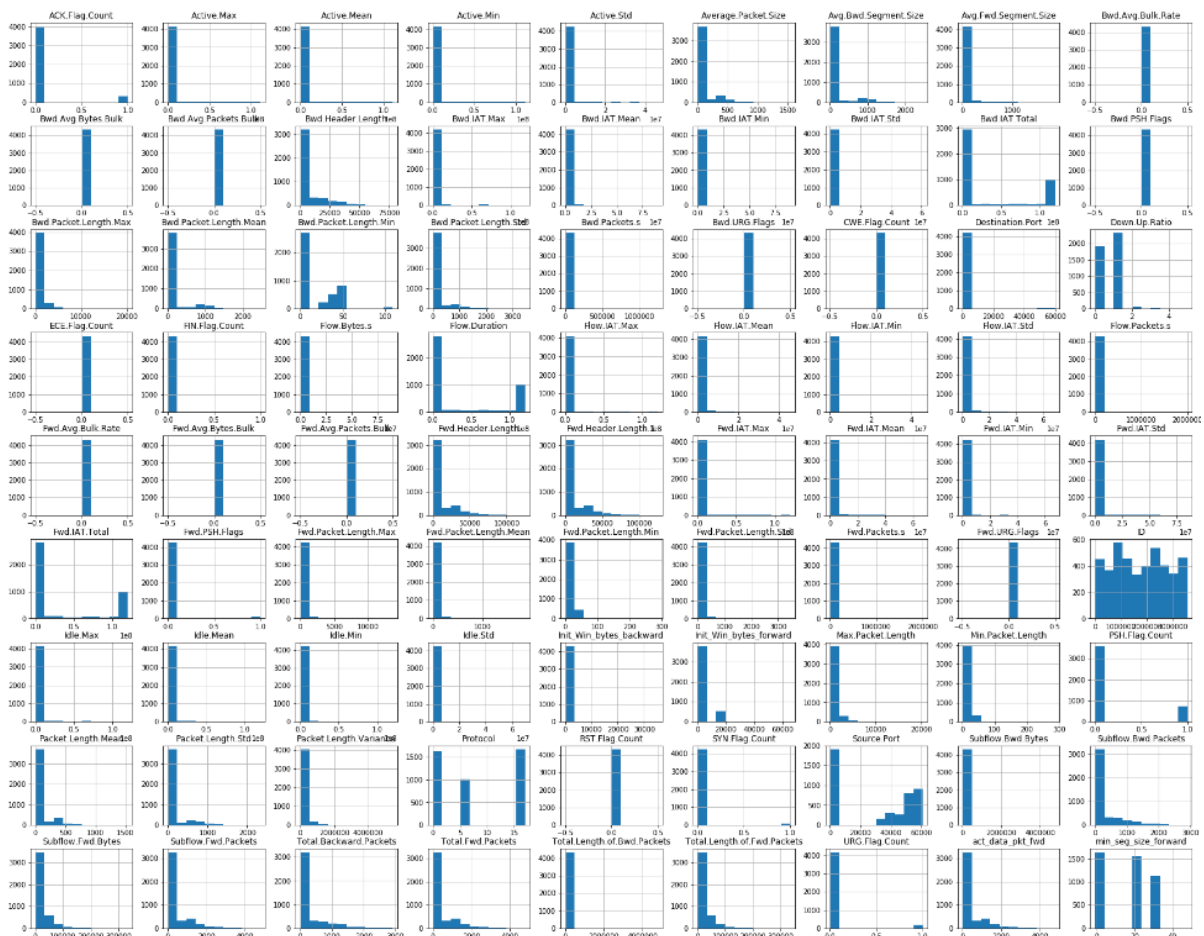
- Dataset: provided by TAs

本次的lab資料是由作業提供, dataset已經相對於lab1的資料算整理得比較好的, 也沒有重複或是有缺值的数据, 此外在clustering也不用先shuffle資料, 以及分出training或validation set

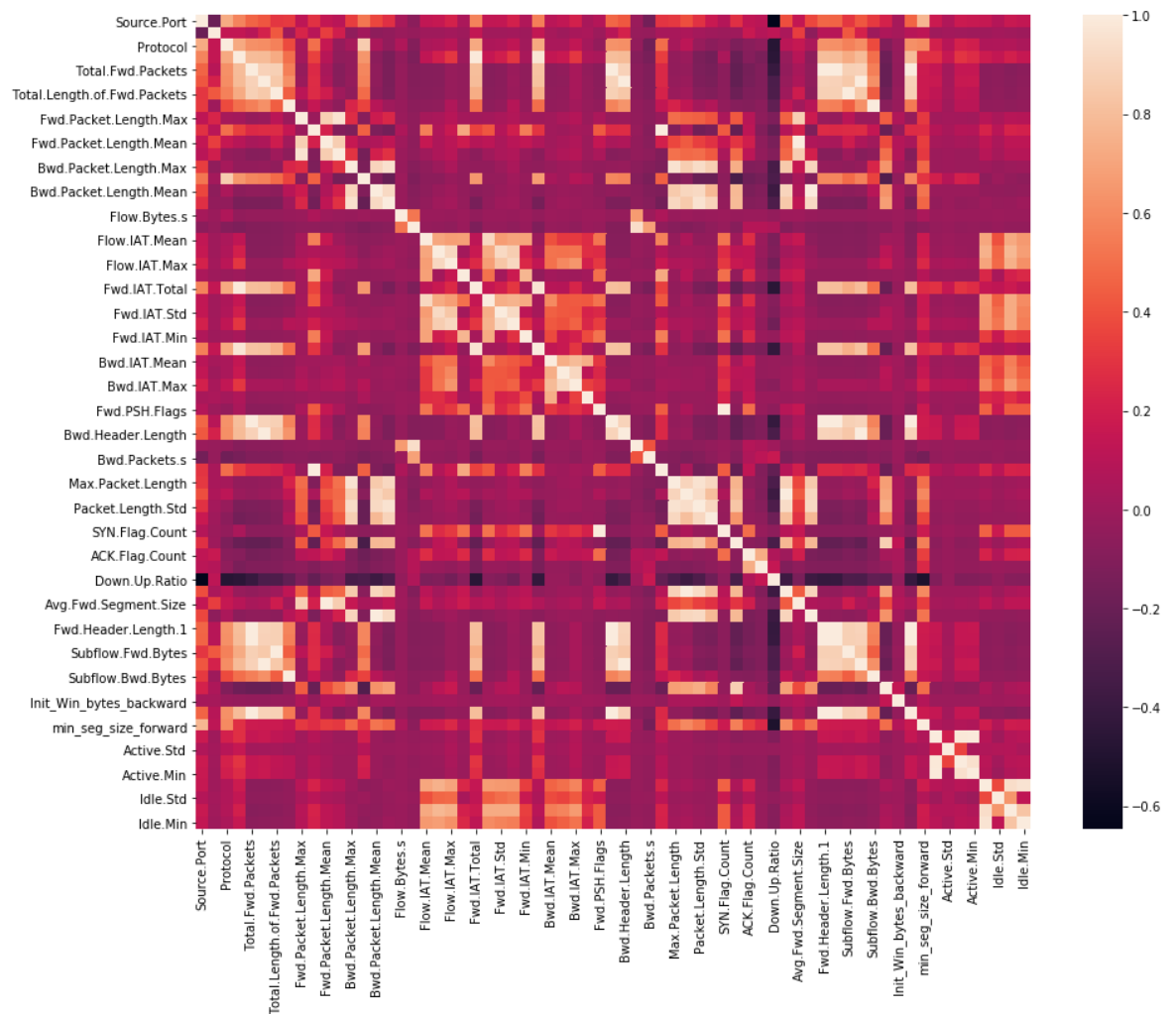
在資料處理上, 首先, 將'ID', 'Flow.ID', 'Source.IP', 'Destination.IP', 'Timestamp' 這些較不相關的欄位去掉, 因為ID是本身資料的身份, IP則可能對著每次網路flow不同而有所不同, Timestamp則是因為每個網路flow的時間跟這些傳輸的類別沒有關係。

Data Visualization

- 首先可以先列出所有的feature的分佈來看哪些分佈比較零散或是集中, 並可以觀察哪些類別的資料可能比較可以拿來做分群效果會比較好



- 用corr_matrix來看資料的相關係數



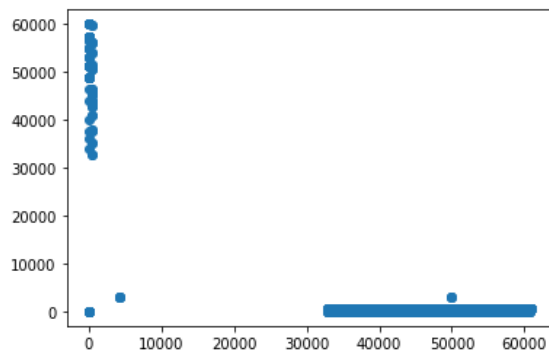
- 並將幾個可能比較重要的指標項目拿出來看分佈，像是destination port可以觀察到主要有0,53,443,123的比例很高，而這些port跟一些TCP, UDP 的傳輸有關(53:DNS, 443:HTTP, 123:NTP)

2	df['Source.Port'].value_counts()	2	df['Destination.Port'].value_counts()
0	1631	0	1631
57429	254	53	1593
59979	237	443	812
51242	225	123	135
48859	181	51242	40
...		57429	20
35059	1	59979	18
49382	1	48859	8
43215	1	3128	7
45260	1	52931	4
43007	1	54944	4
Name: Source.Port, Length: 882, dtype: int64			

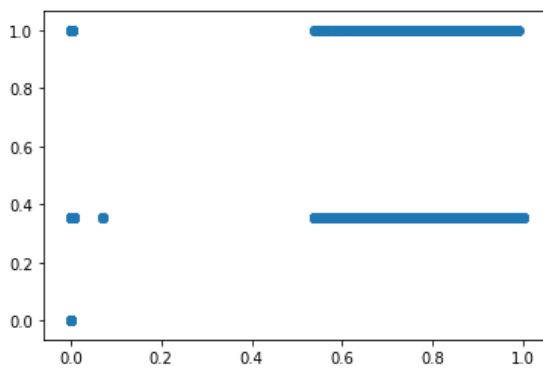
```
1 df['Protocol'].value_counts()
17    1675
0     1631
6     1011
Name: Protocol, dtype: int64
```

- 我們將幾個推測出來可能比較有關的資料拿去畫圖來看他們的資料分佈，可以發現這些資料都偏集中，感覺應該蠻好做分群的

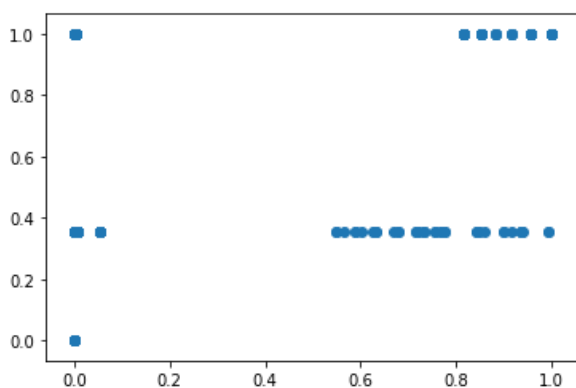
```
: 1 plt.scatter(df['Source.Port'],df['Destination.Port'])
: <matplotlib.collections.PathCollection at 0x7f5d2033ca90>
```



```
1 plt.scatter(df['Source.Port'],df['Protocol'])
<matplotlib.collections.PathCollection at 0x7f5d22532048>
```



```
1 plt.scatter(df['Destination.Port'],df['Protocol'])
<matplotlib.collections.PathCollection at 0x7f5d2268acf8>
```



Feature engineering

先對類別的feature做用使用minmax scalar 來 rescale data, 將 ['Source.Port', 'Destination.Port'] 這些類別的資料做onehot encoding再用pca降為成兩個維度的feature, 然後將 ['Label', 'Protocol'] 這裡個類別直接做onehot encoding, 總共會得到有88個feature的資料

	Source.Port	Destination.Port	Protocol	Flow.Duration	Total.Fwd.Packets	Total.Backward.Packets	Total.Length.of.Fwd.Packets	Total.Length.of.Bwd.Packet
0	0.647741	0.007386	0.352941	0.016844	0.001668	0.001715	0.002376	0.00013
1	0.705797	0.007386	0.352941	0.000546	0.002711	0.002744	0.001115	0.00110
2	0.705912	0.007386	0.352941	0.000892	0.002711	0.004117	0.001115	0.00225
3	0.706454	0.007386	0.352941	0.000628	0.002711	0.003774	0.001115	0.00225
4	0.706650	0.007386	0.352941	0.000549	0.002920	0.004460	0.001115	0.00238
...
4312	0.983940	0.000884	1.000000	0.992006	0.447341	0.733448	0.239550	0.04917
4313	0.983940	0.000884	1.000000	0.261736	0.134724	0.220240	0.072987	0.01375
4314	0.801519	0.000884	1.000000	0.636258	0.000626	0.000000	0.000538	0.00000
4315	0.801519	0.000884	1.000000	0.113510	0.000626	0.000000	0.000538	0.00000
4316	0.801519	0.000884	1.000000	0.869335	0.000626	0.000000	0.000538	0.00000

4317 rows × 88 columns

Clustering Models & Visualize clusters & Measure performance

使用sklearn的模組來做clustering, 並且將features 用pca降成二維, 最後在將cluster的圖形視覺化出來

1. K-means

- n_cluster: 為4
- init: default='k-means++' 選擇初始的放上4個中心位置
- n_init: 設定做幾次kmeans的初始點, 找出最好的起始點
- max_iter: 做多少次找重心的動作

在k-means 作法中因為每個參數表現差異不大, 所以使用default參數

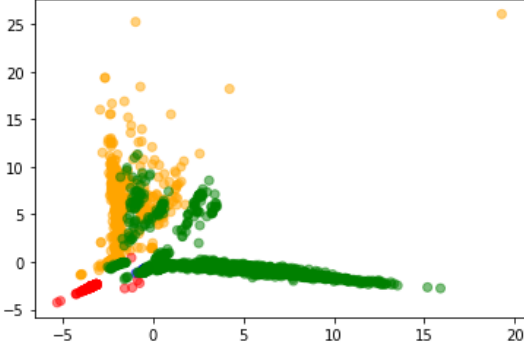
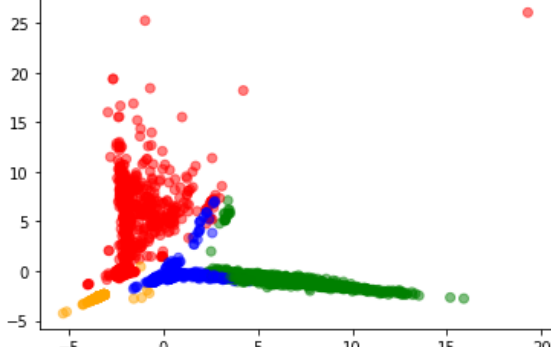
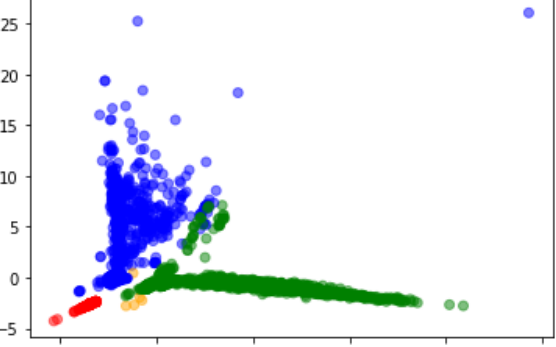
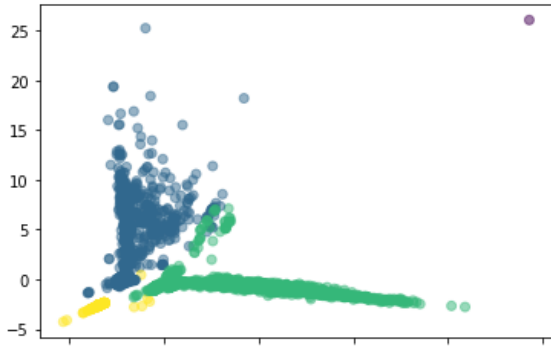
2. AgglomerativeClustering

- n_cluster: 為4
- affinity: 算距離的方式 有{"euclidean", "l1", "l2", "manhattan", "cosine", or "precomputed"} 等方式
- Linkage: 是指每個cluster set 之間計算距離的方式, 有{'ward', 'complete', 'average', 'single'} 等方式

- 在參數選擇方面是嘗試 affinity 和 linkage的組合的結果下去做，發現 (affinity='cosine',linkage='single') 效果表現最好

3. DBSCAN

- 不能設定cluster數量參數
- eps:是兩個same間最大的距離
- min_samples: 在一群sample中找幾個點來算core point
- 也是在不同的eps 和min_sample的參數去組合做實驗，發現 (eps=1.6 and min_samples=5)時表現最好

助教提供的 cluster	K-means
	score: 0.7974237548937673
	
Agglomerative clustering (affinity='cosine',linkage='single')	DBSCAN (eps=1.6 and min_samples=5)
score: 0.8498698090530058	score: 0.8522030136492977
	

- 找任兩個feature來做clustering的 並看看哪些feature 可以做比較好的分群

```

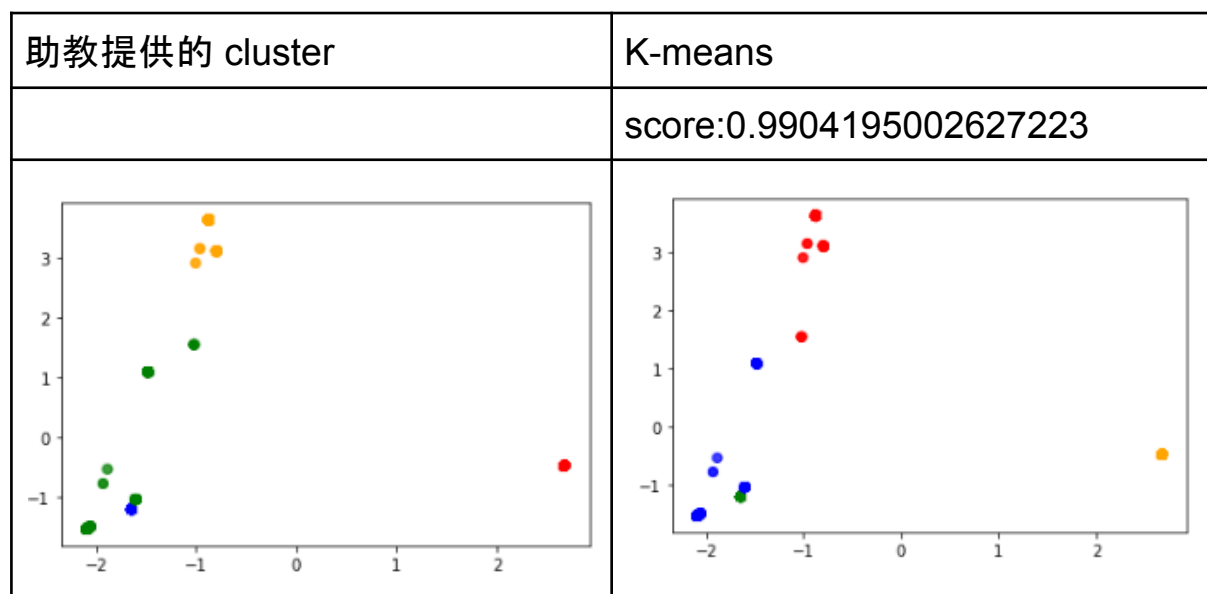
K-means
-----
Current best accuracy score:    0.6393548225029877
Current best feature:    ['Source.Port', 'Destination.Port']
K-means
-----
Current best accuracy score:    0.8176580776308202
Current best feature:    ['Source.Port', 'Protocol']
K-means
-----
Current best accuracy score:    0.9196742097137706
Current best feature:    ['Source.Port', 'Destination.Port_0']
----- Result: -----
K-means| accuracy: 0.9256311364608241 feature: ['Fwd.Packet.Length.Min', 'Destination.Port_0']

```

可以發現 單只用source port 和 destination port 的表現, score 大概只有64%, 而用Protocol 和 source port 準確度達到82%, 而source port 加上經過onehot encoding 還有pca的 destination port_0 feature 表現達到了92%, Fwd.Packet.Length.Min 加上 同一個 destination port_0 feature 也差不多達到93%。

Using Domain knowledge to cluster

就對於network flow的了解, 以及對資料視覺化的觀察, 想說可以用 {'source port', 'detination port', 'portocol'} 這三個feature來做分類即可, 所以我先將所有這些feature 去做onehot encoding, 這個資料拿去做分群, 發現結果意外的好有達到**99%**。



Port 53:DNS, Port 443:HTTP, Port 123:NTP, Port 0 為非固定port(過多可能是攻擊行為)

Discussions and Conclusion

- 由最後的使用domain knowledge 來分類可以發現feature的選擇對於分群上也是相當的關鍵的，如果能對feature有更深的了解，或許可以在分群上有很好的幫助
- DBSCAN 雖然有時候不見的會分成完整的4群，但是在表現上也是能維持在一定的水準，感覺比較適合當各個群都有比較平均數量的時候來用，表現會更好，由下圖助教提供的cluster發現 cluster 1的比例相對小很多可能用DBSCAN上分類就沒辦法完整的分出想要的類數

3	1695
2	1631
0	856
1	135

- 可能是因為這次的traffic flow 的資料剛好可以透過protocol, destination port, source port 去做效果還不錯，如果遇到一些比較特別 flow，不知道有有沒有辦法也將它也好的分類？