**Report**

1. **What environments the members are using?**

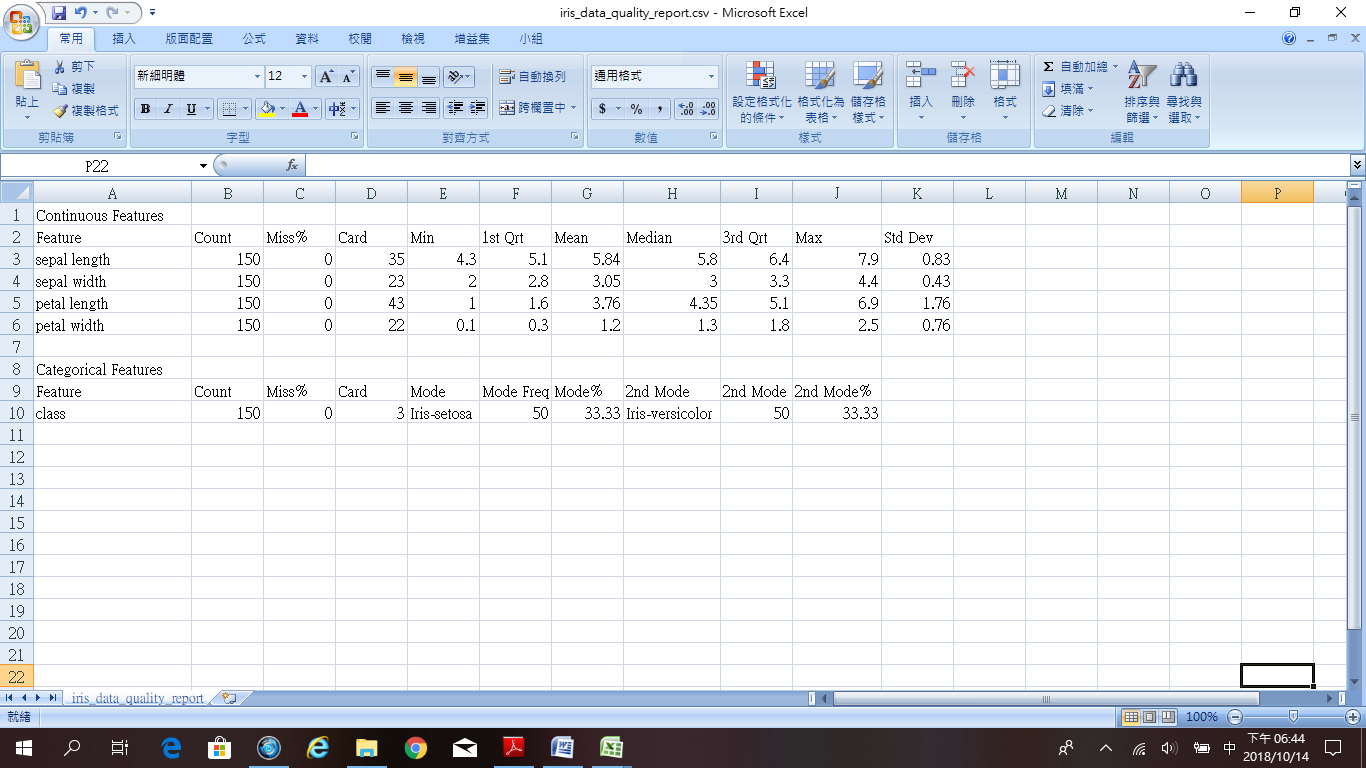
* 使用Python 3

1. **Basic statistic visualization of the data?**

(1) Iris

* 原始資料：iris.csv
* 程式：iris.py（執行方式：python3 iris.py）
* 執行程式所產生的兩個檔案：iris\_data\_quality\_report.csv，preprocessed\_iris.csv
* Basic statistic visualization：

iris\_data\_quality\_report.csv資料如下：



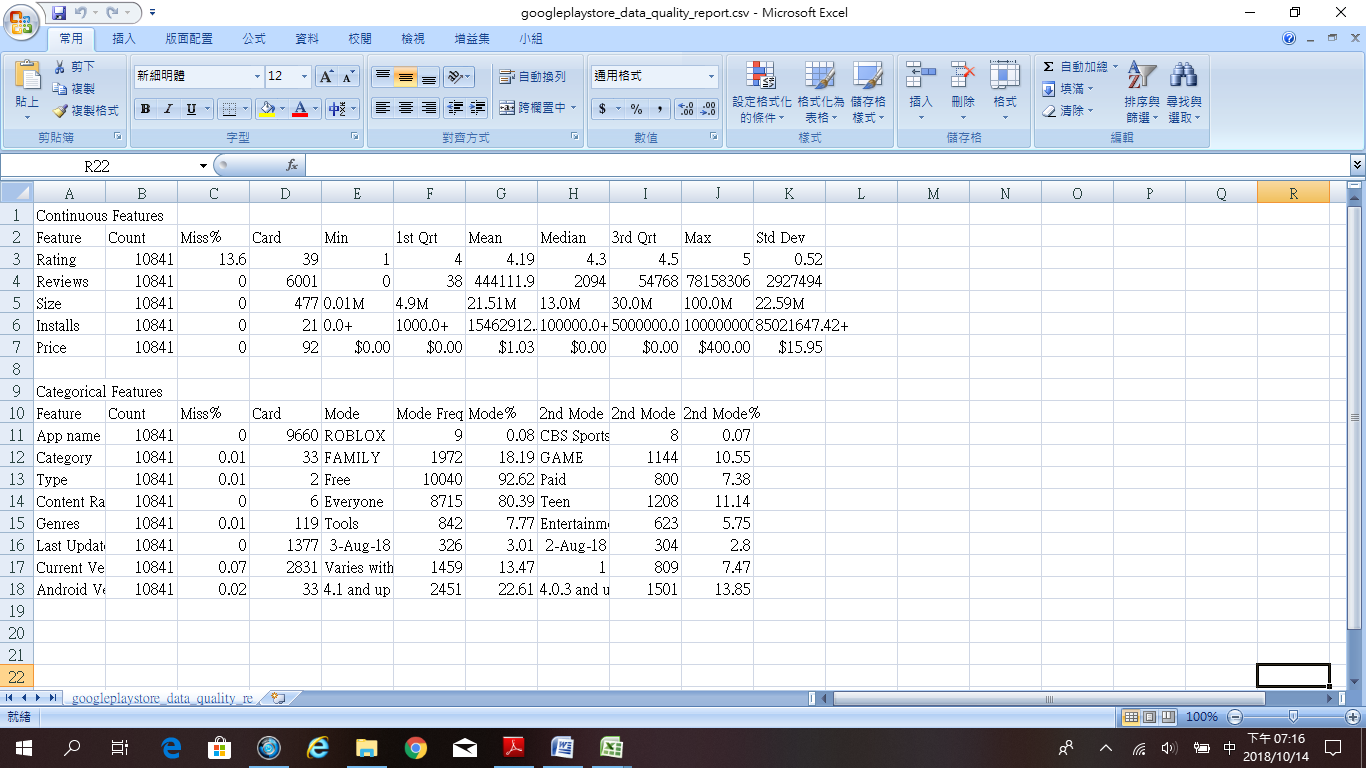
說明：計算Continuous Features之個數、資料遺失率、相異值個數、最小值、第一四分位數、算術平均數、中位數、第三四分位數、最大值及標準差。計算Categorical Features之個數、資料遺失率、相異值個數、最常出現的值、最常出現的值之個數、最常出現的值之百分比、第二常出現的值、第二常出現的值之個數、第二常出現的值之百分比。

* Training data：preprocessed\_iris.csv為iris.csv經前處理過的training data

(2) Google play store

* 原始資料：googleplaystore.csv
* 程式：googleplaystore.py（執行方式：python3 googleplaystore.py）
* 執行程式所產生的兩個檔案：googleplaystore\_data\_quality\_report.csv，preprocessed\_googleplaystore.csv
* Basic statistic visualization：

googleplaystore\_data\_quality\_report.csv資料如下：

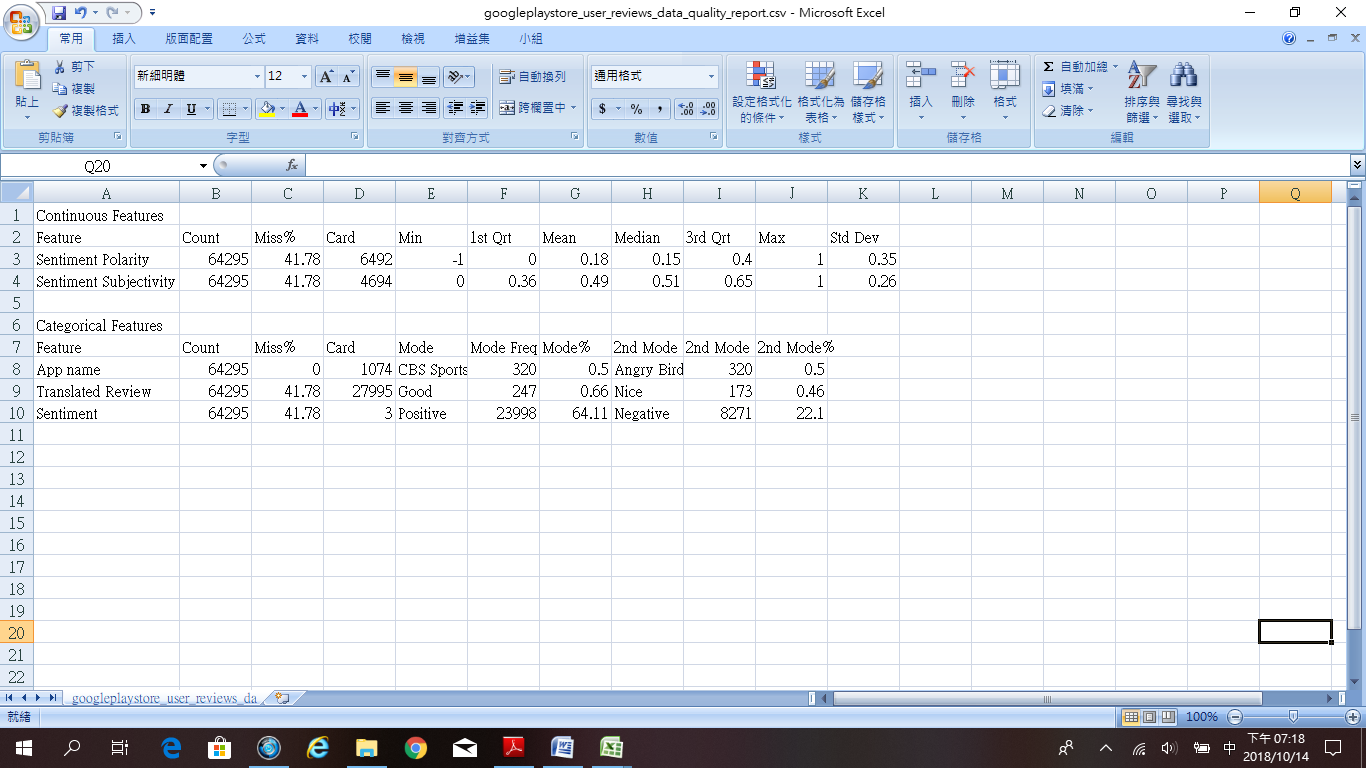


* Training data：preprocessed\_googleplaystore.csv為googleplaystore.csv經前處理過的training data

(3) Google play store user reviews

* 原始資料：googleplaystore\_user\_reviews.csv
* 程式：googleplaystore\_user\_reviews.py，Preprocess\_csvs.py（執行方式：先python3 googleplaystore\_user\_reviews.py再執行 python3 Preprocess\_csvs.py）
* 執行程式所產生的兩個檔案：googleplaystore\_user\_reviews\_data\_quality\_report.csv，preprocessed\_googleplaystore\_user\_reviews.csv
* Basic statistic visualization：

googleplaystore\_user\_reviews\_data\_quality\_report.csv資料如下：



* Training data：preprocessed\_googleplaystore\_user\_reviews.csv為googleplaystore\_user\_reviews.csv經前處理過的training data

1. **Data preprocessing methods?**

觀察所有的data quality reports，因為每個feature的missing rate都沒有大於60%，且每個feature的Cardinality都比1大，因此沒有必要把任何的feature刪除，且每個feature的missing rate都小於50％，故採取Imputation的方法，也就是對於每個instance，若碰到有任何feature是missing的，continuous feature用算術平均數（Mean），categorical feature用最常出現的值（Mode）填補上去。

1. **How you generate decision tree and random forest models?**

(1) Iris dataset

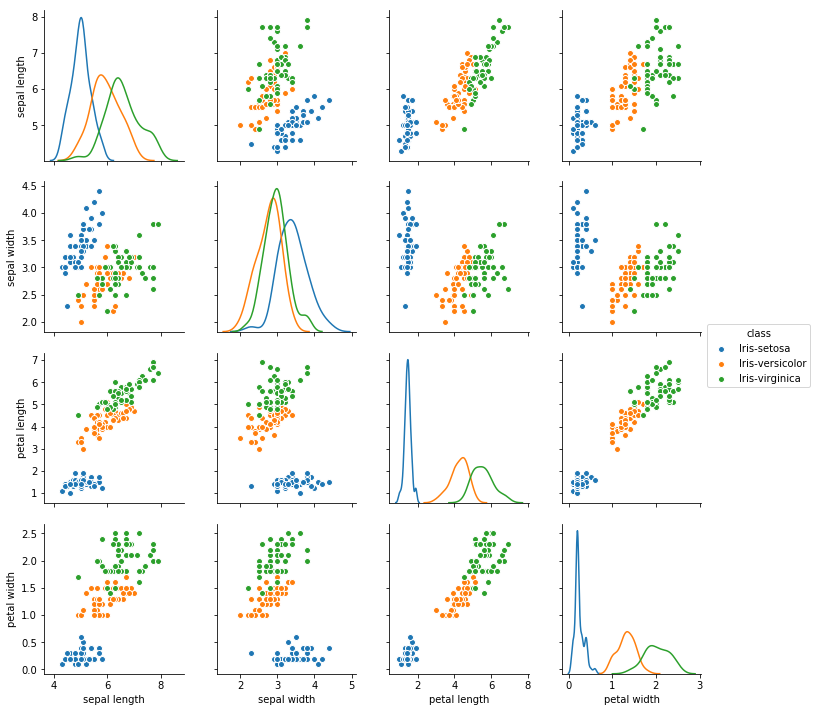


圖1. 利用Python Seaborn產生三種不同class在各feature下的散布圖

* 利用圖像化分析Iris.data，可以發現三筆不同的class (Iris的種類) 在各種feature下的分布算是相當分散，應能順利分辨三種class，因此能用所有feature來產生decision tree。
* Decision tree在這組data中使用到的是Classifier，因為只有三種class；criterion設定為entropy，max\_depth設為3避免overfitting。
* 切割Training data和Testing data的比例為7:3，利用sklearn中的train\_test\_split做分割。
* 最後以sklearn.metrics中的accuracy\_score簡單評估本model的好壞。

(2) Google play store dataset

首先和Iris dataset一樣做圖像化分析處理完畢的googleplaystore.csv

在這裡做兩種預測。

1. **Installs**
2. **Rating**
3. **Installs 分析**

首先對googleplaystore.csv用Seaborn做圖像化分析，

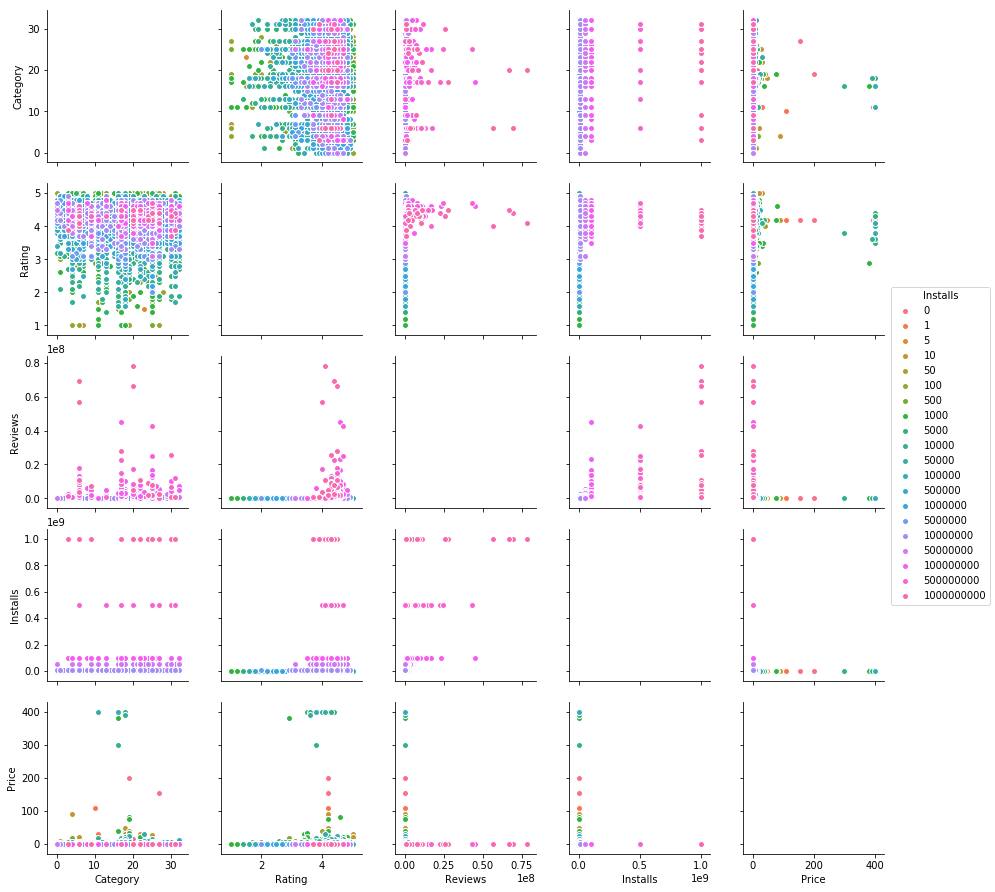


圖2. googleplaystore.csv中，各feature和Install數的關聯散布圖

註: 這裡feature只取可數值化的部分，Category中共有33筆不同資料，以0~32作編號

接著分析googleplaystore\_user\_review.csv。觀察其中App column，可以發現此data可能為googleplaystore.csv中的部分App review。故將googleplaystore.csv中Installs數以App為index加在googleplaystore\_user\_review.csv各row的後面作分析。

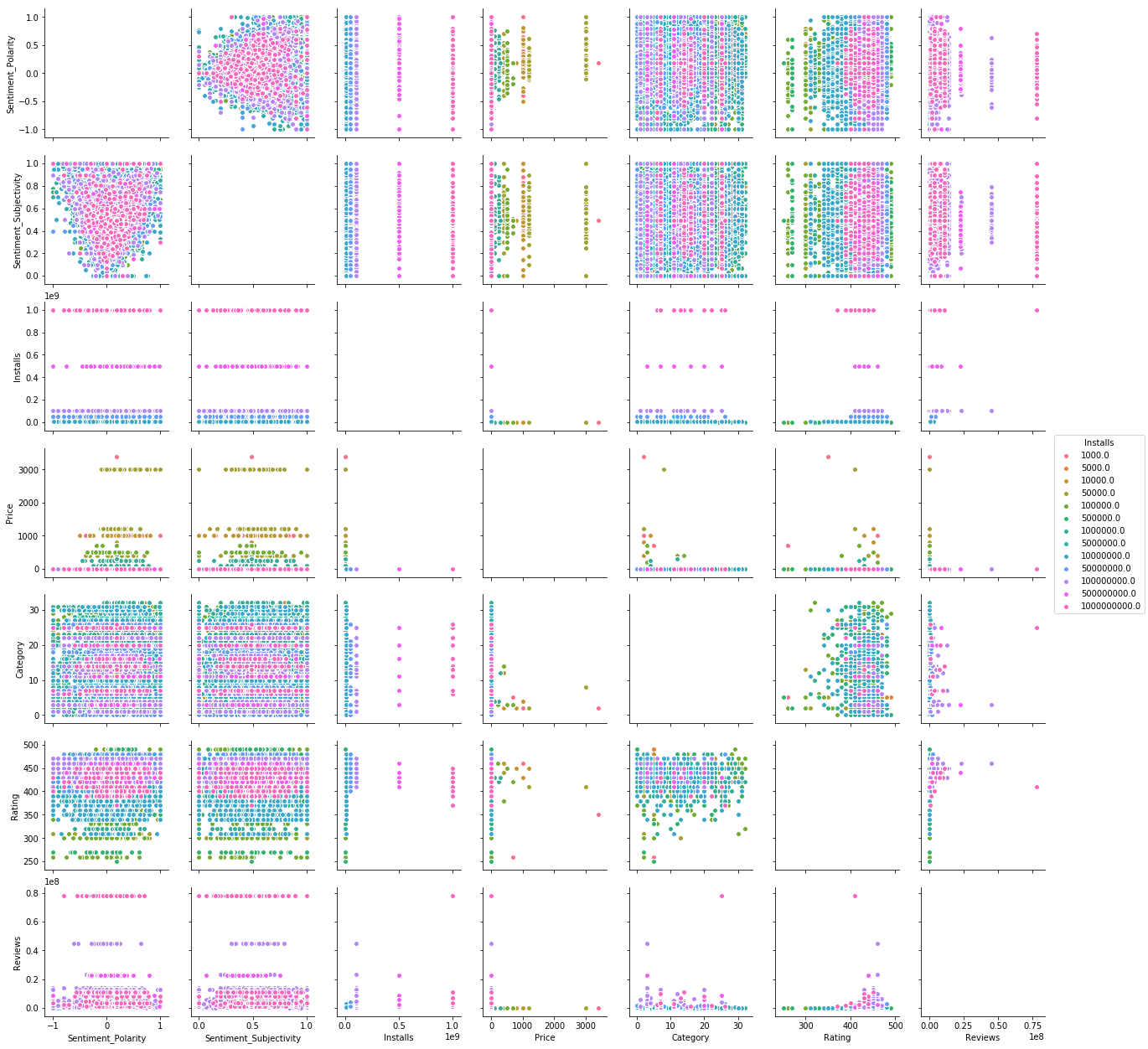


圖3. googleplaystore\_user\_review.csv中，各feature和Install數的關聯散布圖

註: Rating 經過處理 (\*100)

* Installs數為一離散值 (因原本資料無詳細下載數)，以Classifier做Decision Tree
* 可以看到，高Installs數的data (圖2.中偏向紫色的data)，較可能為免費下載 (price為0)、Rating較高 (4~5)、Reviews較多的部分data，和Category、Sentiment (Review語意的正向性)並無明顯關聯。
* 圖3. 的結果和圖2. 的預估相同，因資料的重複性可能造成有不太一樣的結果，需兩方比較。
* 因此取Price、Rating、Reviews做Decision Tree model

1. **Rating分析**

對兩個csv file做和Installs相同的分析步驟

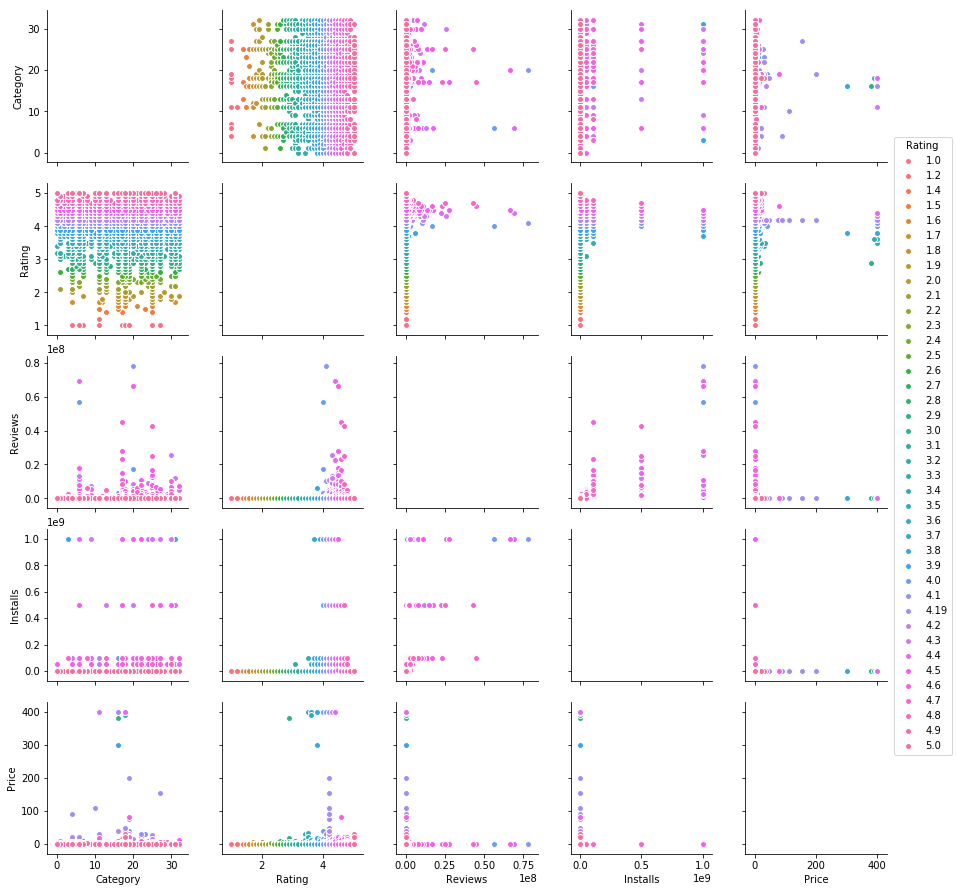


圖3. googleplaystore.csv中，各feature和Rating的關聯散布圖

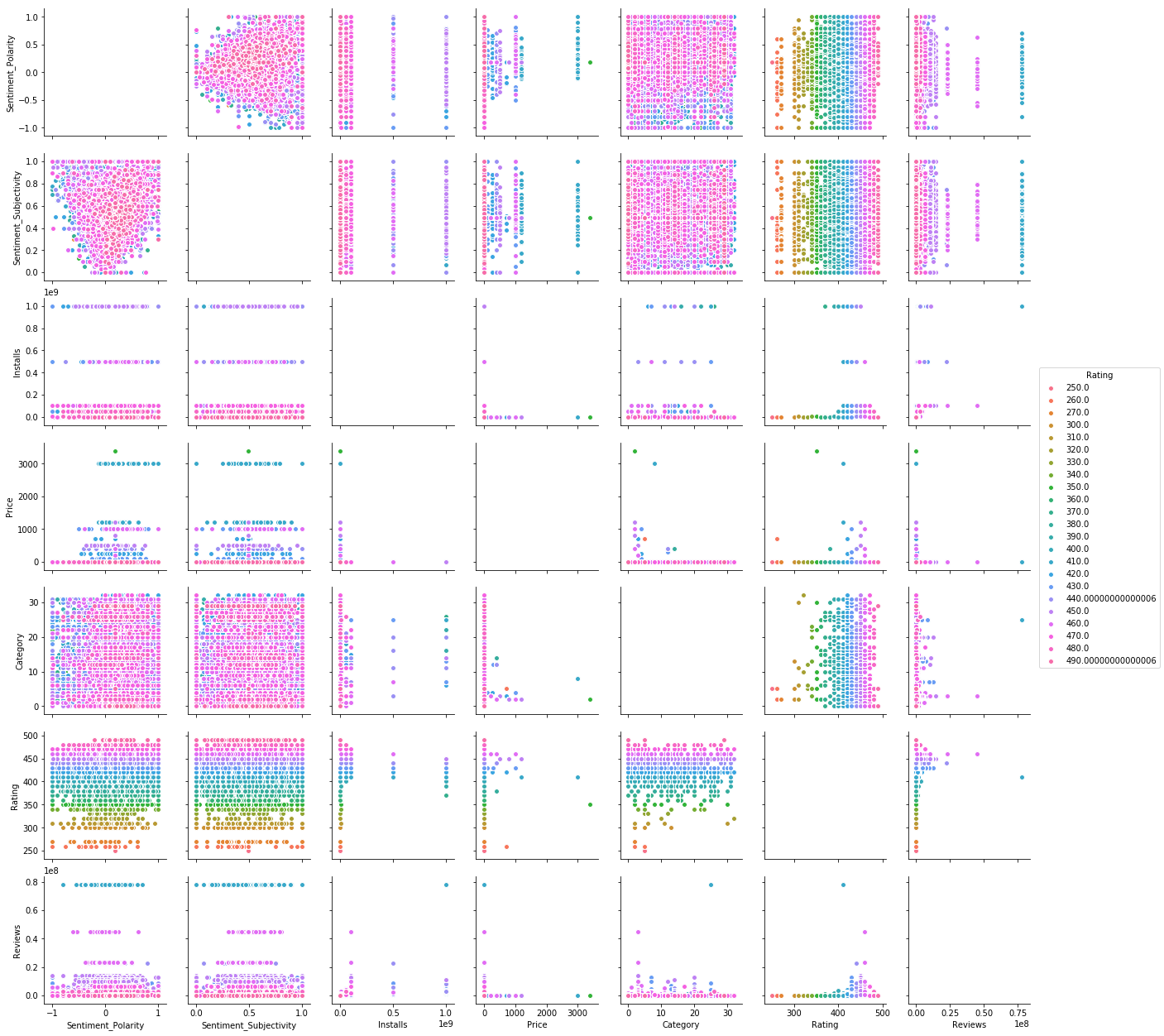


圖4. googleplaystore.csv中，各feature和Rating的關聯散布圖

* 從圖3.、圖4.中，一樣可以看到和Installs類似的結果
* 即Rating和Installs、Price、Reviews的相關性較高
* 故一樣以此三Features做Decision Tree model
* 因Rating為一連續的資料，故以Regressor建Decision Tree model

因為Installs和Rating都與Sentiment無明顯關聯，故以googleplaystore.csv的資料建model。

1. **Generate Random Forest**

兩個Dataset皆以以下方法建Random Forest model：

在這裡我應用的是Bootstrapping做resample產生許多subsample。

此subsample中的data row，因取樣後重新放回後再次取樣，因此可能會有重複data row的情況發生。

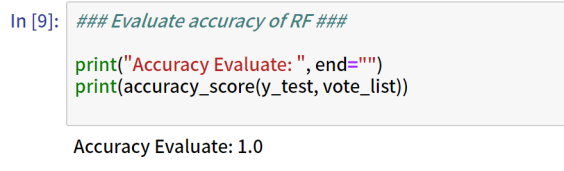
用這些subsample，並從Features中隨機取幾個Feature，搭配產生出許多Decision Tree，這裡我產生出100個隨機樹。

對每個Decision Tree餵入這100個Testing data，得到各data row的預測結果，因此每個data row會有100個預測結果。

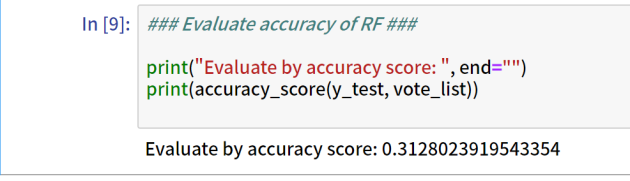
接著利用眾數的方式取100個結果中出現最多的數字，以此做為該data row的最終預測值。即以100張票投票的方式產生預測結果。

1. **The performance?**

* Decision Tree:
* Random Forest:
  + Iris, evaluated by accuracy score



* + Google play store (Installs), evaluated by accuracy score



* + Google play store (Rating), evaluated by r2-score



1. **Conclusion?**