**評論分類器**  
Comment Filter  
<專題實驗報告>  


成員 : 10227215 祝銘  
 10227221 黃礪鞍

指導教授: 柯士文

**摘要**

現在是一個網路的世代，任何議題放上網路後，都會有許多人關注，並且留下”評論(Human Comment)”。然而更會有人利用程式在這些評論下面留出一些”廣告評論(Machine Spam)”。而我們發現Youtube上的留言被檢舉之後會多一個”Spam標籤”但是卻無法判斷此留言是所謂的廣告留言還是使用者所發的垃圾話。所以此專題目的致力於**精確分析出Human Spam以及Machine Spam**，並且**利用使用者所提供的回饋重新訓練出更準確的模型**。

**Abstraction**

It is a generation of a cyber . Any issue , blog ,video or tweet be post on internet can be anyone discussed . Sometimes people post a comment on it . However Some Advertisers make some Commercial Message by program That we call Machine Spam . We found the comments were reported would create a “spam label”, but we cannot recognize which comment is machine spam or human spam .   
Therefore, our project is devoting to distinguish between human comment and machine spam precisely. We also use feedback that provided by user to retrain our model .

目錄

[壹、 緒論 1](#_Toc466993181)

[一、 研究動機 1](#_Toc466993182)

[二、 開發平台(Ubuntu) 1](#_Toc466993183)

[三、 開發簡介 1](#_Toc466993184)

[貳、 流程圖 1](#_Toc466993185)

[一、 概念 1](#_Toc466993186)

[二、 前置作業 1](#_Toc466993187)

[三、 實作 1](#_Toc466993188)

[參、 系統實做 1](#_Toc466993189)

[一、 前置處理 1](#_Toc466993190)

[二、 網頁介面 1](#_Toc466993191)

[肆、 分析報告 1](#_Toc466993192)

[一、 數據呈現 1](#_Toc466993193)

[二、 結論 1](#_Toc466993195)

[伍、 研究結果 1](#_Toc466993196)

[一、 介面 1](#_Toc466993197)

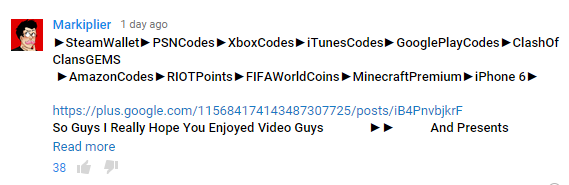
[陸、 文獻參考 1](#_Toc466993198)

1. 緒論
   1. 研究動機

現代人越來越喜歡使用社群了，無論是Facebook、Tweet、Youtube、論壇…等等，都有許多使用者。而這些平台的開發者都會提供發文與**回復**的基本功能。  
 當一個使用者發佈了一個訊息後，許多人都可以至此做一個回復的動作，可想而知的，許多廣告業者也不會錯過這樣的散佈機會。所以往往我們在預覽留言時，常常會看到許多與主題無關的留言，像是廣告、保險、色情…等等字眼。  
 如果今天沒有這些不好的評論，發文者以及關注此文章的留言者，都可以更清楚的了解這些評論是正向或者負向。我們也發現Youtube有提供一個檢舉的功能，這些被檢舉的留言稱為Spam[[1]](#footnote-1)，然而有些被檢舉的留言卻不是被機器產生的廣告留言，只是讓檢舉人不愉悅的留言罷了(Human Spam[[2]](#footnote-2))。  
 因此我們的專題致力於能夠從這些Spam中，找出真正屬於Machine Spam[[3]](#footnote-3)的留言，將他與一般的留言做區分。

*[本專題目前以Youtube的評論為分析對象]*

**(以下為垃圾留言範例)**



* 1. 開發平台(Ubuntu)
     1. 系統管理

Ubuntu所有系統相關的任務均需使用Sudo指令是它的一大特色，這種方式比傳統的以系統管理員帳號進行管理工作的方式更為安全，此為Linux、Unix系統的基本思維之一。

* + 1. Ubuntu的主要應用
       - **網路伺服器**

Web Server、FTP、Samba、DNS、DHCP、Firewall、Proxy、Mail Server、 ...

* + - * **軟體開發**

C/C++、Java、Python、Shell Script、Perl、Ruby、…

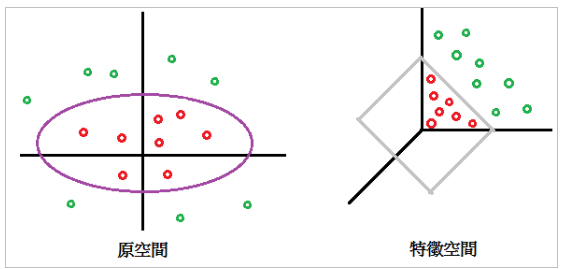
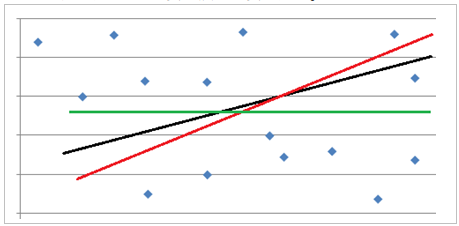
* + - * **資料庫**

MySQL、PostgreSQL、…

* + 1. 選用原因
       - **Server運行較快速**
       - **Python運行較Window穩定**
  1. 開發簡介
     1. 程式語言
        + **Python**
        + **JavaScript**
        + **JQuery**
        + **Css**
     2. 開發工具
        + **YoutubeAPI + BeautifulSoup** Google在官方提供了YoutubeAPI，提供使用者能夠與Youtube進行互動。它提供了Activity、Channel、playlist、video……等多種的運算。當我們鎖定某個影片後，並將對應的參數設定好，則能夠得到結果  
           (*以下為使用API搜尋影片中的留言得到之結果*)  
            
            
            
            
            
            
            
            
            
            
            
            
            
            
          如此一來，我們還需要工具來幫我幫把這些有用的資料抓出來。因此我 們採用Python所提供的套件json，它可以簡易的將JSON檔一層一層化簡，讓我們得到特定欄位的訊息。

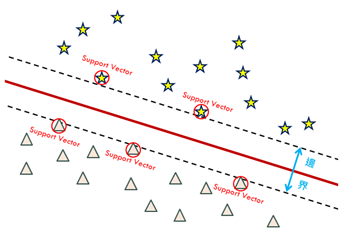


此為一筆留言的紀錄，接著將該物件中的值取出。

* + - * **SVM**
        1. 介紹  
            在[機器學習](https://zh.wikipedia.org/wiki/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0)中，支援向量機（Support Vector Machine，常簡稱為SVM，又名支援向量網路）是在[分類](https://zh.wikipedia.org/wiki/%E5%88%86%E7%B1%BB%E9%97%AE%E9%A2%98" \o "分類問題)與[迴歸分析](https://zh.wikipedia.org/wiki/%E8%BF%B4%E6%AD%B8%E5%88%86%E6%9E%90)中分析資料的[監督式學習](https://zh.wikipedia.org/wiki/%E7%9B%A3%E7%9D%A3%E5%BC%8F%E5%AD%B8%E7%BF%92)模型與相關的學習[演算法](https://zh.wikipedia.org/wiki/%E7%AE%97%E6%B3%95)。給定一組訓練例項，每個訓練例項被標記為屬於兩個類別中的一個或另一個，SVM訓練演算法建立一個將新的例項分配給兩個類別之一的模型，使其成為非機率[二元](https://zh.wikipedia.org/w/index.php?title=%E4%BA%8C%E5%85%83%E5%88%86%E7%B1%BB%E5%99%A8&action=edit&redlink=1)[線性分類器](https://zh.wikipedia.org/wiki/%E7%BA%BF%E6%80%A7%E5%88%86%E7%B1%BB%E5%99%A8)。SVM模型是將例項表示為空間中的點，這樣對映就使得單獨類別的例項被儘可能寬的明顯的間隔分開。然後，將新的例項對映到同一空間，並基於它們落在間隔的哪一側來預測所屬類別。
        2. 概念  
            SVM最主要的概念就是想要讓訓練用資料在特徵空間中，找到一個超平面將這些資料分開來。  
           這邊就必須來解釋一下什麼是特徵空間呢?我們可以看到下圖，如果想要在原空間將這些零散的資料，清楚劃分成兩類，就需要用到橢圓方程式，但是如果我們透過一些方式將原空間的資料轉換到特徵空間，這時候我們就可以找到一個平面將資料成功劃分
        3. [](http://3.bp.blogspot.com/-_8ei_Ogm7qA/VE9gf6JBSQI/AAAAAAAAHwU/9zxXNXDWqBE/s1600/Screenshot+from+2014-10-28+17:20:33.png)資料切割  
            我們可以看到下面這張圖，上面有很多的點，我們也可以找到很多條線可以將這些資料分割成兩類，但是到底要如何畫出那一條線，我們又該如何知道正確的那一條線在哪?這就是SVM所要解決的主要問題了。

至於要選哪一條線，SVM會選擇邊界越大越好，原因是邊界如果選擇太小，那麼只要資料有任何細微的變動，都很有可能導致分類的錯誤。  
座落於邊界上的點就是所謂的支撐向量(Support Vector)。

SVM產生出一條直線，將資料分為兩個區域。並將邊界附近的點視為Support Vector

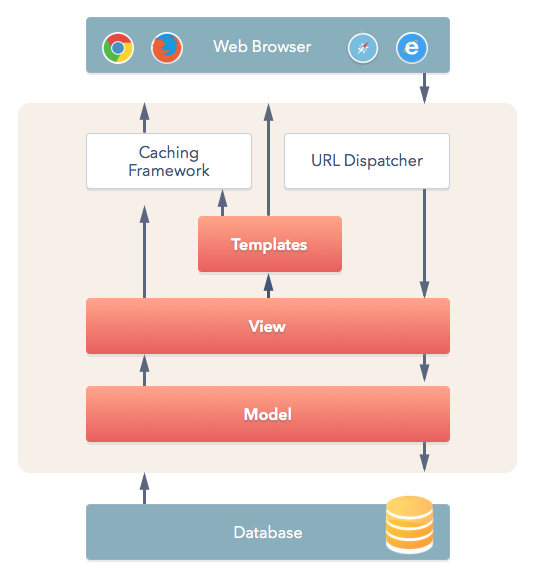
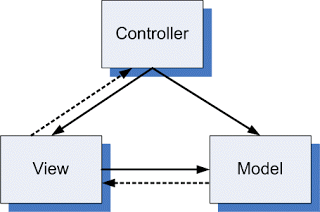


* + - * **Django**
        1. 介紹  
           [Django](https://www.djangoproject.com/)可以說是Python最名的 Web Framework它有以下的特色:

免費開放原始碼。

著重快速開發、高效能。

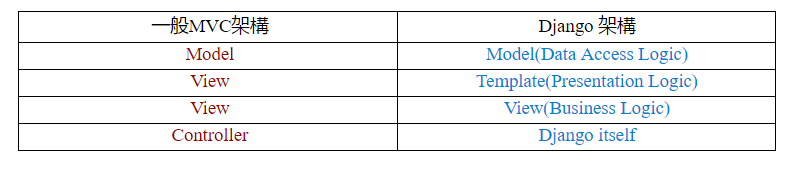
遵從 [DRY ( Don't Repeat Yourself )](http://c2.com/cgi/wiki?DontRepeatYourself) 守則，致力於淺顯易懂的程式碼 使用類似 Model–view–controller (MVC) pattern 的架構。

* + - * 1. Django的架構  
            如同一些比較著名的 Web framework，Django 同樣也使用了類似 MVC 的架構，只是在定義和解釋上略為不同，稱為 MTV ( Model–Template–View )，我們可以透過下面這張圖來了解其運作方式：
        2. MVC架構模式  
            MVC模式就是架構模式的一種，MVC模式的目的是實作一種動態的程式設計，使後續對程式的修改和擴充功能簡化，並且使程式某一部分的重複利用成為可能。除此之外，此模式通過對複雜度的簡化，使程式結構更加直覺。[](http://2.bp.blogspot.com/-PsIlp7IzG70/Vdgg051LuZI/AAAAAAAA6BE/DLn4eEFnMe0/s1600/mvc.png)

控制器 Controller - 負責轉發請求，對請求進行處理。

視圖View - 介面設計人員進行圖形介面設計。

模型Model - 程式設計師編寫程式應有的功能（實作演算法等等）、資料庫專家進行資料管理和資料庫設計（可以實作具體的功能）

* + - * 1. MVC與MTV比較  
           View所描述的是你看到什麼，而不是你看到的是什麼樣子，這兩者是很微妙的差異。在這樣的理解之下，在Django的框架的View是特定網址(URL)的回呼函式(callback function)，回呼函式會決定哪一個資料被獲取，所以View描述的是"你看到什麼"。  
           更進一步來說，將內容(content)從展現的視圖中分離開來是明智的選擇，因此Django採用樣板(template)的方法。Django的View決定呈現哪一個資料，而View所指定的Template則會描述"你如何看到、長什麼樣子"。換句話說，Django的View和Template都是對應到MVC架構的View，都是關於資料顯示的部分。
        2. **URL概念的改變** 以往我們的概念總是一個 URL 對應到一個檔案，如 <http://localhost/test.htm> ，那麼我們讀取的網頁毫無疑問的就是 test.htm。然而若是動態網頁，我們會發現其實這樣子的分類並不恰當，因為網頁其實可以用功能性來做區分。就因為網址的型態從檔案改為功能區分，因此我們會需要一個有效的方式來處理。在 Django 中，它是讀取一個名為 urls.py 的檔案，其中網址的部份使用「正規表示式 (Regular Expression)」 來做核對，若是某一部分 match 了，則使用某一支程式來處理這個 request。
        3. **Template(樣板)** 有別於原始的 PHP 寫法，Django 使用樣板來呈現網頁的內容。樣板大部分都是 HTML 的程式碼，只有資料的部份會用特殊的 Django template 格式來取代，當然還有一些控制的機制例如 for 迴圈、if else 判斷等。這樣一來，網頁設計的部份就和資料的處理分離，不論是資料處理或是網頁的部份，程式碼看起來都乾淨許多。
      * **WSGI（Python Web Server Gateway Interface）介紹**
        1. 介紹 是為[Python](https://zh.wikipedia.org/wiki/Python)語言定義的[Web伺服器](https://zh.wikipedia.org/wiki/%E7%B6%B2%E9%A0%81%E4%BC%BA%E6%9C%8D%E5%99%A8" \o "網頁伺服器)和[Web應用程式](https://zh.wikipedia.org/wiki/%E7%BD%91%E7%BB%9C%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F)或[框架](https://zh.wikipedia.org/wiki/Web%E5%BA%94%E7%94%A8%E6%A1%86%E6%9E%B6)之間的一種簡單而通用的[介面](https://zh.wikipedia.org/wiki/%E4%BB%8B%E9%9D%A2_(%E7%A8%8B%E5%BC%8F%E8%A8%AD%E8%A8%88))。
        2. 規範概觀  
            WSGI區分為兩個部份：一為「[伺服器](https://zh.wikipedia.org/wiki/%E4%BC%BA%E6%9C%8D%E5%99%A8" \o "伺服器)」或「閘道」，另一為「應用程式」或「應用框架」。在處理一個WSGI請求時，伺服器會為應用程式提供環境資訊及一個回呼函式（Callback Function）。當應用程式完成處理請求後，透過前述的回呼函式，將結果回傳給伺服器。  
           所謂的 WSGI [中介軟體](https://zh.wikipedia.org/wiki/%E4%B8%AD%E4%BB%8B%E8%BB%9F%E9%AB%94)同時實作了API的兩方，因此可以在WSGI伺服器和WSGI應用之間起調解作用：從WSGI伺服器的角度來說，中介軟體扮演應用程式，而從應用程式的角度來說，中介軟體扮演伺服器。「中介軟體」元件可以執行以下功能：

重寫[環境變數](https://zh.wikipedia.org/wiki/%E7%8E%AF%E5%A2%83%E5%8F%98%E9%87%8F" \o "環境變數)後，根據目標[URL](https://zh.wikipedia.org/wiki/%E7%BB%9F%E4%B8%80%E8%B5%84%E6%BA%90%E5%AE%9A%E4%BD%8D%E7%AC%A6)，將請求訊息路由到不同的應用物件。

允許在一個[行程](https://zh.wikipedia.org/wiki/%E8%A1%8C%E7%A8%8B" \o "行程)中同時執行多個應用程式或應用框架。

[負載均衡](https://zh.wikipedia.org/wiki/%E8%B4%9F%E8%BD%BD%E5%9D%87%E8%A1%A1)和遠端處理，通過在[網路](https://zh.wikipedia.org/wiki/%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%BD%91%E7%BB%9C)上轉發請求和回應訊息。

進行內容後處理，例如應用[XSLT](https://zh.wikipedia.org/wiki/XSLT" \o "XSLT)樣式表。

* + - * 1. 實際應用  
            當使用者送出一個Request來到伺服器，接著網頁程式產生內容，然後送出Response回去給Browser，最簡單的網頁程式做的事情大概就像是這樣，但是真實世界的網頁程式要做的事情很多，他們得記錄Session資訊、取得Cookie、認證、錯誤處理等等雜事，而傳統的架構你可以想像一下這些東西都沒有標準可言，都只限於單一的網頁程式，他們可能都跟特定模組糾結在一起，而WSGI的Middleware將這些該做的事都限定於Middleware中，它們跟上下游溝通的方式都是按照標準來實作的，這表示這些Middleware都能重覆被利用，這就是WSGI帶來的好處之一。

1. 流程圖
   1. 概念

因為Youtube上時常會出現與主題無關的留言，像是廣告、保險、色情…等等字眼。而這些並不是使用者所樂於看見的。  
 因此我們欲做出媲美Youtube的網頁，提供使用者查詢影片，我們利用SVM所建立出一個分類的過濾器，對該影片下面的留言逐一放入模型產生出分類，看哪些留言屬於Machine Spam以及那些屬於一般留言。  
 並且提供使用者回饋的功能，讓使用者可以一起訓練Model，讓分析能夠越來越精準。

* 1. 前置作業
  2. 實作

輸入欲查詢的影片名稱

將此筆紀錄儲存下來，並設定Model每隔一段時間(24HR)，把這些紀錄與Support Vector重新訓練

將相關影片返回前端

點選影片

呈現影片，並分析該影片之評論，並傳回前端

回饋?

Yes

No

結束

1. 系統實做
   1. 前置處理
      1. 資料蒐集  
         原始的Youtube Comment 是從 Lab的資料庫取得。
      2. 資料庫連線設定  
          CreateDictionary.py and the LoadDictionary.py 的資料庫連線 設定都是透過python 的 pymysql 套件取得與資料庫的連線。
      3. 建立model需要依序執行下列三個檔案
         * **CreateDictionary.py** 此py檔產生一份文字檔(Output-stemmed.txt) 內容為所有獨立字和出現頻率，判斷特殊字的規則定義在Comment.py 的 tokenize():以下為此method的流程。
           1. Step1將所有字元轉成小寫。
           2. Step2  
              處理URL 後將前面加上’#’ 例:  
              <http://abc.efg.com/2010/09> -> #abc.efg.com，如果某字的第一個字元是@ 把此自從句子裡移除 例: @user0129283->“”
           3. Step3  
              接受所有字母
           4. Step4  
              把’\n’ 換成 ‘ ‘(空白)
           5. Step5  
              刪除”’”(單引號)
           6. Step6  
              同樣字母不能連續出現三次 例: . Helllllloo -> Hellloo
           7. Step7  
              其他所有字園一律變成” “(White Space)

https://en.wikipedia.org/wiki/Django\_(web\_framework)

Django (web framework)

From Wikipedia, the free encyclopedia

1 History

2 Features

2.1 Components

2.2 Bundled applications

2.3 Extensibilityabsolut 48

absurd 5

https 1

wikipedia 2

org 1

wiki 1

django 2

web 2

framework 2

free 1

encyclopedia 1

histori 1

featur 1

compon 1

bundl 1

applic 1

extens 1

* + - * **LoadDictionary.py** 此檔輸出為資料庫裡評論的Vector Space Model，格式為 :  
        [flag] [Word**1**Index]:[Word**1**Frequency] [Word**2**Index]:[Word**2**Frequency] …[Word**N**Index]:[Word**N**Frequency]

0 978:1 3331:1 3124:1 2942:1 1825:1 1343:1 1300:1 3334:1 2514:1

0 1704:1 1685:1 890:1 1699:1 3905:1

0 3745:1 1825:1 1699:2 2149:3 2003:1 2388:1 1230:1 3501:1 1158:1 921:1 3869:1 2356:1 3688:1 979:1 890:1 2292:1 1897:1

0 25:1 1923:1 1699:1 721:1

0 2373:1 451:1 3600:1 3907:1

* + - * **CreateModel.py**此py檔使用SVM輸出，檔名為”libsvm.model”的model。
  1. 網頁介面
     1. Sever系統  
        我們的整個網頁是利用Apache2 server + mod\_wsgi驅動的。
     2. 網頁架構  
         當使用者輸入IP address 之後 會被直接導向到home.html這時候views.py 是直接回傳home.py沒有做過任何處理，而首頁基本是提供使用者一個搜尋的表單而已。  
         接著使用者輸入表單時(搜尋關鍵字)，系統會重新導向到views.py 這時候 ，我們透過 Youtube API 得到一份 JSON檔，而這份JSON檔裡面包含所有搜尋影片的資料，views.py利用BeautufulSoup套件取得資料之後把需要的資料傳到Templates 裡的index.html 再把結果回傳給使用者。  
         當使用者點選需要分析的影片之後，系統會重新導向到views.py 這時候 我們透過 Youtube API 得到一份 JSON檔而這份JSON裡面包含所有該影片的評論資料的資料經過資料處理之後會呼叫MachineOrNot( comments ) ，此function 會將 評論逐筆的拿去分析最後將評論加上label回傳最後再將資料一起送到youtube.html。

1. 分析報告
   1. 數據呈現

M% = NM% =

*※註: 影片從”熱門”挑出前50部、M(Machine)、NM(Not Machine)*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 影片數量 | 總評論數量 | M數量 | NM數量 | M Mistake[[4]](#footnote-4) | NM Mistake[[5]](#footnote-5) | 總錯誤量[[6]](#footnote-6) | M% | NM% |
| 正確答案 | 50 | 9453 | 58 | 9395 | 0 | 0 | 0 |  |  |
| 原始結果 | 50 | 9453 | 122 | 9331 | 86 | 22 | 98 | 29.5% | 99.70% |
| 第一次重新訓練 | 50 | 9453 | 106 | 9347 | 67 | 19 | 86 | 36.7% | 99.79% |
| 第二次重新訓練 | 50 | 9453 | 97 | 9356 | 55 | 16 | 71 | 43.2% | 99.82% |

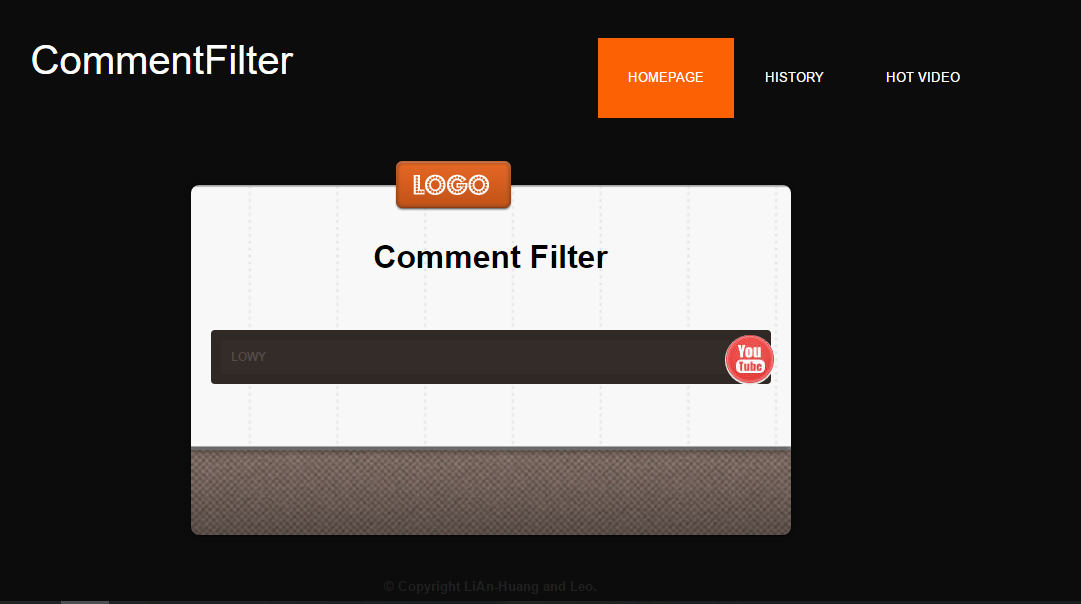
* 1. 結論

由於Youtube最近也開始有處理垃圾留言的情況，所以大部分的留言都是正常的，只有少數幾筆資料是Machine產生的或者是廣告留言，導致我們的數據沒有那麼的完美。

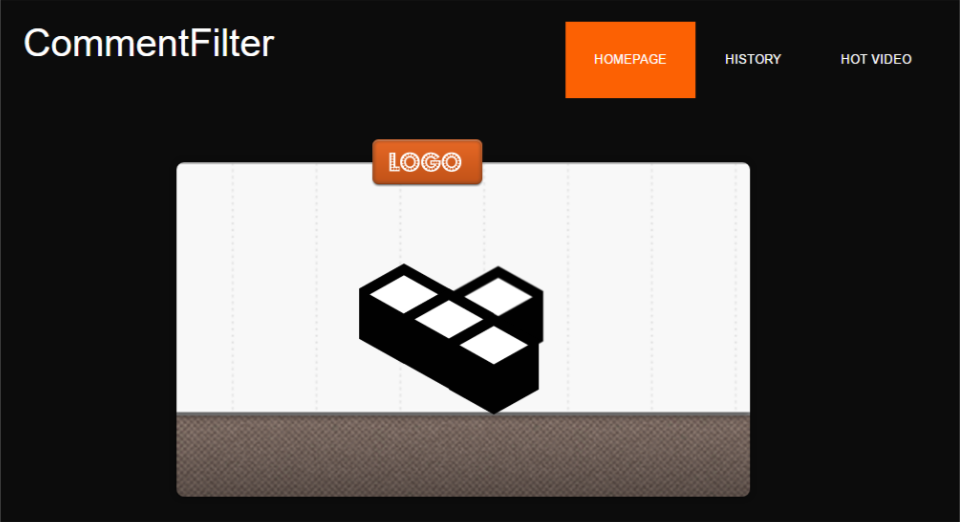
雖然一開始SVM分析的結果沒有很精確，但經過使用者的回饋後，我們重新訓練了Model，再次測試後，確實有變比較準確。

因此，我們認為當越多人使用我們的系統，並且幫助我們一起訓練Model的話，效果一定會更顯著，並讓使用者更滿意。

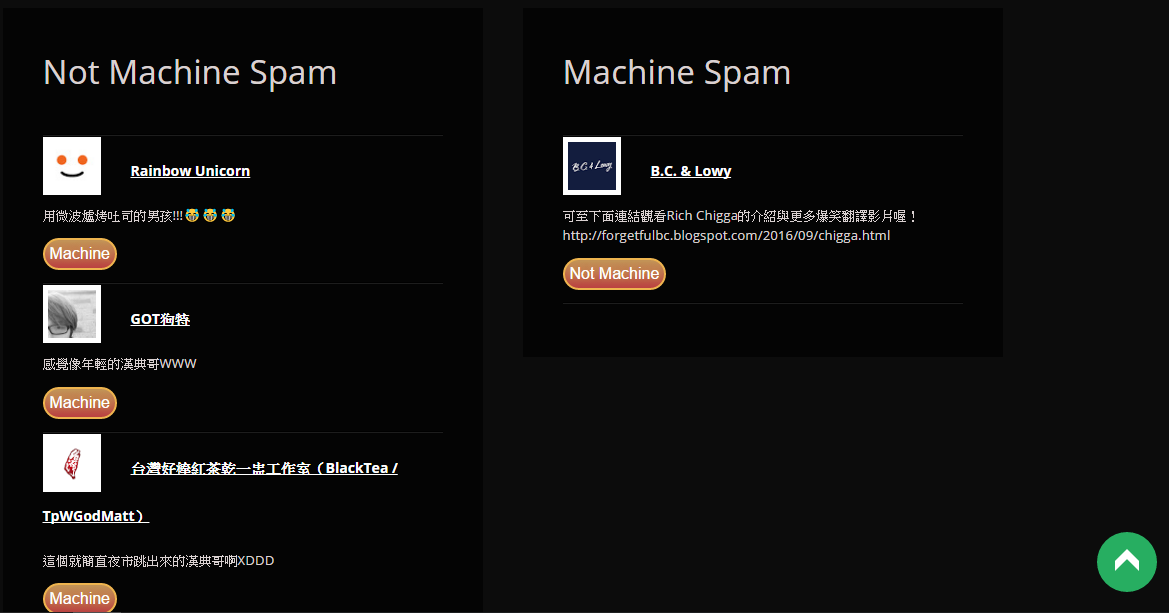
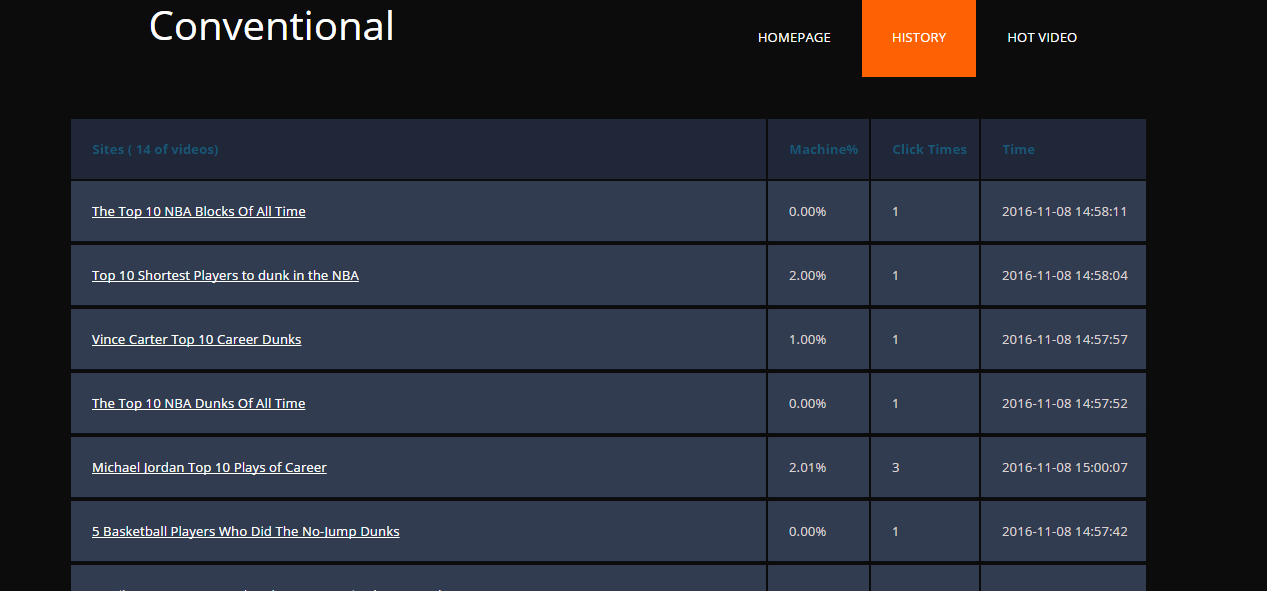
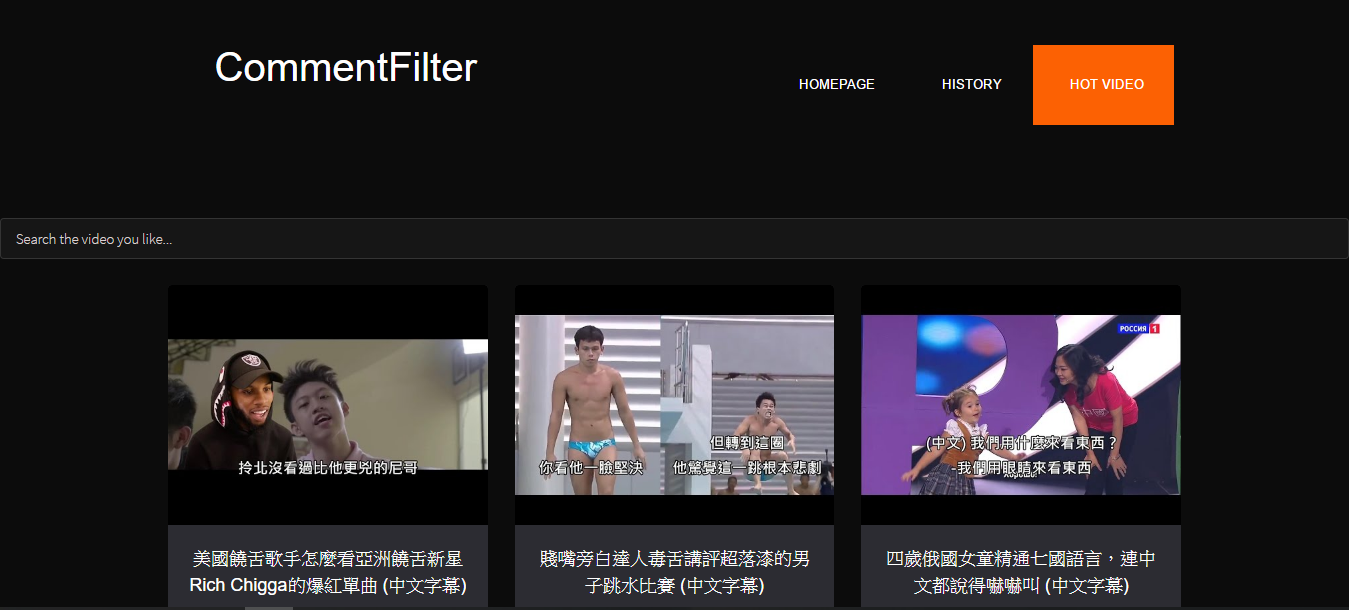
1. 研究結果



輸入愈查詢影片



* 1. 介面



提供使用者回饋，助系統能更準確分析出結果。

[每天將回饋重新訓練Model]

**(顯示Machine%、點擊次數、日期於History)**

1. 文獻參考

<https://djangogirlstaipei.gitbooks.io/django-girls-taipei-tutorial/content/django/introduction.html>  
  
<http://www.openfoundry.org/tw/news/1330>  
  
<http://mropengate.blogspot.tw/2015/08/mvcdjangomtv.html>  
  
https://zh.wikipedia.org/wiki/%E6%94%AF%E6%8C%81%E5%90%91%E9%87%8F%E6%9C%BA  
  
https://cg2010studio.com/2012/05/20/%E6%94%AF%E6%8C%81%E5%90%91%E9%87%8F%E6%A9%9F%E5%99%A8-support-vector-machine/  
  
<http://www.bituzi.com/2014/10/helper-of-classification.html>  
  
<https://zh.m.wikipedia.org/zh-tw/Web%E6%9C%8D%E5%8A%A1%E5%99%A8%E7%BD%91%E5%85%B3%E6%8E%A5%E5%8F%A3>  
  
<https://www.youtube.com/yt/dev/zh-TW/api-resources.html>  
  
https://www.crummy.com/software/BeautifulSoup/bs4/doc/

1. Spam(垃圾留言) : 使用者檢舉的言論，包括廣告言論跟另檢舉人不舒服的文字。 [↑](#footnote-ref-1)
2. Human Spam(人為垃圾留言) : 使用者的留言，可能內容有不雅字眼或歧視字眼讓其他使用者不 舒服。 [↑](#footnote-ref-2)
3. Machine Spam(機器產生的垃圾留言) : 由程式所產生的垃圾留言，通常多為廣告主要特性包括：(1)未經消費者的同意

   (2)與消費者需求不相關

   (3)以詐欺的方式騙取郵件位址

   (4)攻擊性的廣告：例如誇張不實，包括情色、釣魚網站

   (5)散布的數量龐大。 [↑](#footnote-ref-3)
4. Machine mistake : 使用者回報錯誤的Machine Spam量 [↑](#footnote-ref-4)
5. Not Machine mistake : 使用者回報錯誤Not的Machine Spam量 [↑](#footnote-ref-5)
6. 總錯誤量:使用者回報錯誤的總數量 [↑](#footnote-ref-6)