Quora Question Pairs

104062703 曾若淳

Problem Description

這是 Kaggle 最近的比賽題目，問題為：給定一組問題 pair (Q1, Q2)，預測 Q1, Q2 是不是重複的問題，舉例來說：

Q1: How do I read and find my YouTube comments?	duplicate
Q2: How I can see all my Youtube comments?	

Q1: What are the best thing to buy on Amazon?	non-duplicate
Q2: What is the best thing I can buy for 2€ on Amazon?	

Dataset

資料內容只有問題的描述文字，以及要預測的是否為重複的label。

Training set 有 404,290 筆，格式為（qid1, qid2, question1, question2, is_duplicate）

Testing set 有 2,345,796 筆，格式為（question1, question2）

Training set 比起 Testing set 多了問題的編號 qid 是為了方便參賽者做資料分析。

Methods

由於我只有文字的資料，所以我的方法會著重在 feature extraction，我首先做了觀察是什麼原因造成不同但是重複的問題。

Feature Extraction

一、character-level similarity

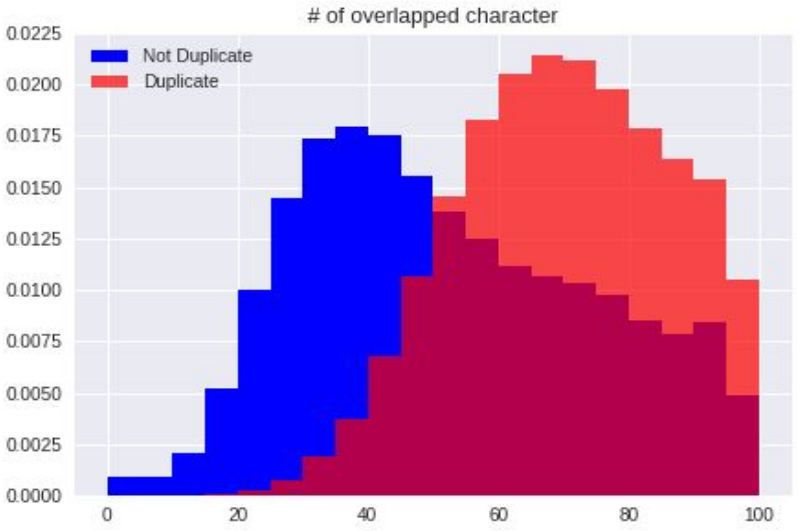
由於人打字很容易會出現 typo 或在不重要的細節描述上會有些微差異，比如：

Q1: Should people over **98** not be allowed to vote?	duplicate
Q2: Should people over **90** not allowed to vote?	

這時簡單計算字元重複的個數「重複越多字元是 duplate 的可能越高」但當差異的字元是決定字義的關鍵時，比如：

Q1: What are some **solved** problems in math?	non-duplicate
Q2: What are some **unsolved** problems in math?	

便是這種想法的反例。而關於這個 feature 對分辨是否 duplicate 的效果如何？我透過統計資料中，問題 pair 重複的字元個數對label 的相關性：



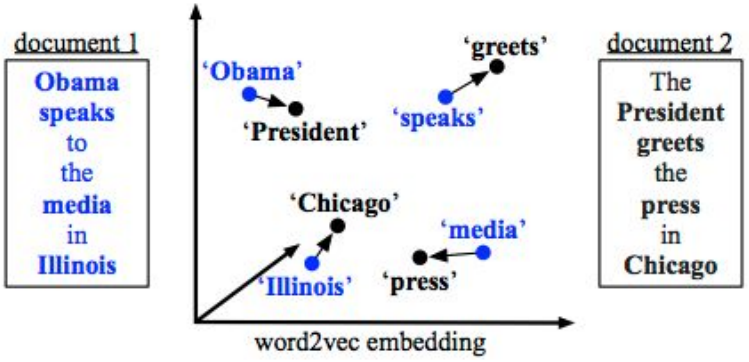
可以觀察到當問題 pair 重複字元(x軸)少的時候，pair 幾乎只會是 non-duplicate，而 duplicate pair 的分佈幾乎全跟 non-duplicate pair 重疊，可以猜想這個 feature 較適合分辨 non-duplicate pair 而對 duplicate 的分辨幫助不大。

二、word-level similarity

同義字的使用是另一個造成問題重複的原因，比如：

Q1: How to **determine** the boiling point of a substance?	duplicate
Q2: How to **calculate** the boiling point of a substance?	

這項feature 需要知道字的相似度才能計算，因此我使用 pre-trained 的word embedding - GloVe 來得到字的向量，相似的字會被 embed 到靠近的位置。但我的目標是要算出兩個問題句子在用字上的相似度，那該如何定義兩個句子的相似度呢？直覺作法是取字的向量的平均得到句子中心，再計算兩個句子中心的差距。除了算句子中心差距，我額外也抽取了有paper 用實驗證明表現更好的 Word Mover's Distance [1]

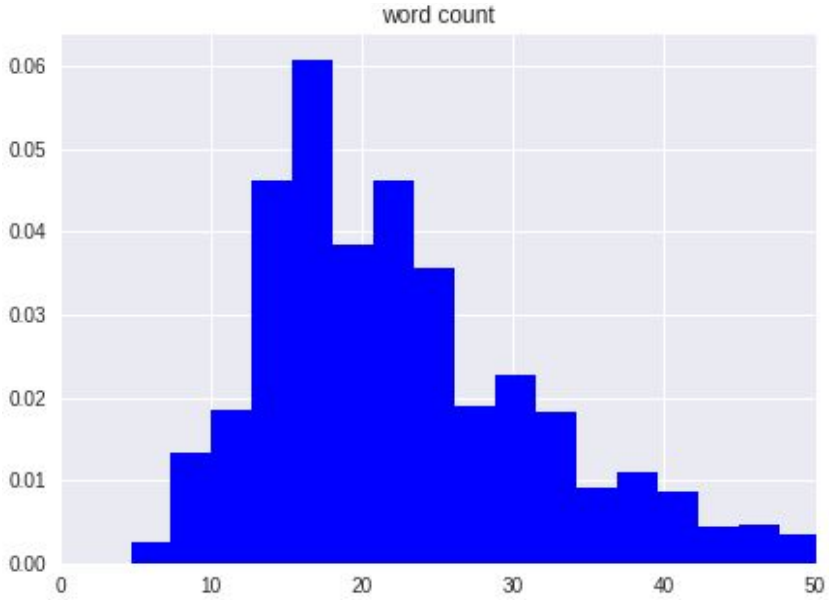


WMD的定義為將其中一個句子完全變成另一個句子所需的最少的搬動量。

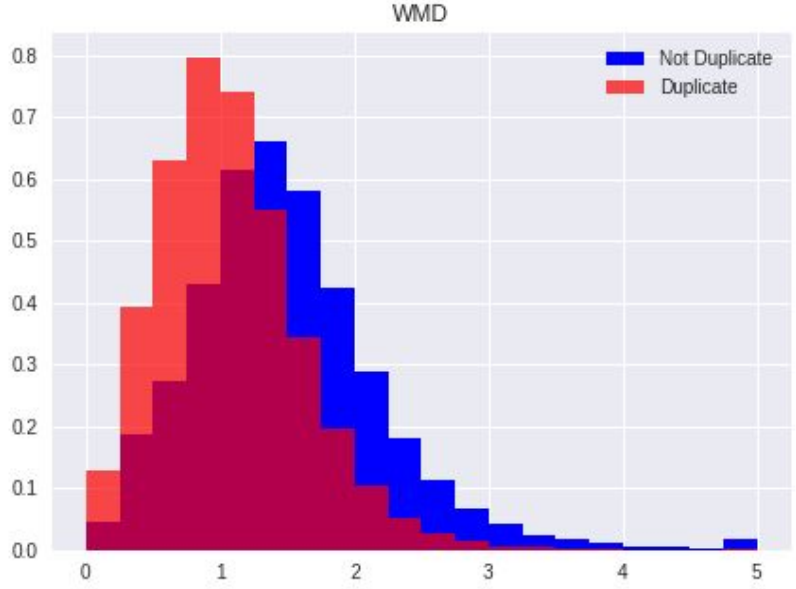
$$\min_{T\geq 0} \sum_{i,j=1}^{n} T_{ij}c(i,j) \text{ subject to } \sum_{j=1}^{n} T_{ij} = d_i, \ \forall i \text{ and } \sum_{i=1}^{n} T_{ij} = d'_j, \ \forall j$$

將兩句子先表示成 BoW 向量 d, d'，但假如兩句子長度不同會造成搬動量無法對應，所以 d, d' 是normalized 的BoW。而 T_{ij} 表示第一個句子中第 i 個單字搬到第二個句子第 j 個單字的搬動量，$c(i,j)$ 則是第 i 個單字和第 j 個單字的距離。至於兩個constraint 則非常直覺，一個在規定第一個句子的字必須全部搬出

，另一個則是搬完的結果必須等於第二個句子。問題pair 用字的相似度用WMD計算的很準確，但是計算WMD的複雜度相當高，目前最好的solver 需要 $O(l^3 \ln l)$, l 為句子長度，我關心的是能否在這次比賽中使用，所以統計dataset中問題句子的長度：



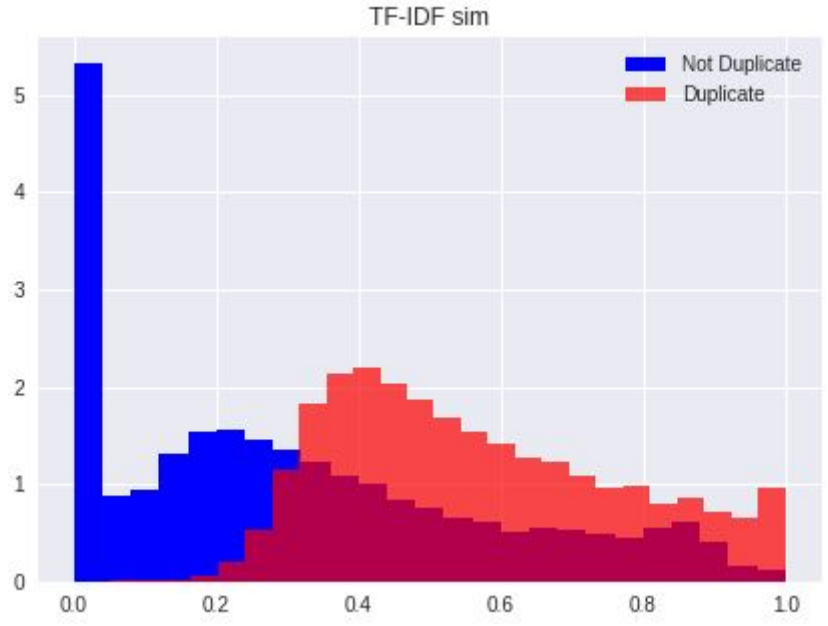
平均是10個字，雖然training 加上 testing set 共有200 多萬筆，但簡單做平行計算同時開48個process 能夠在半小時內算完。至於，WMD預期分辨效果如何？



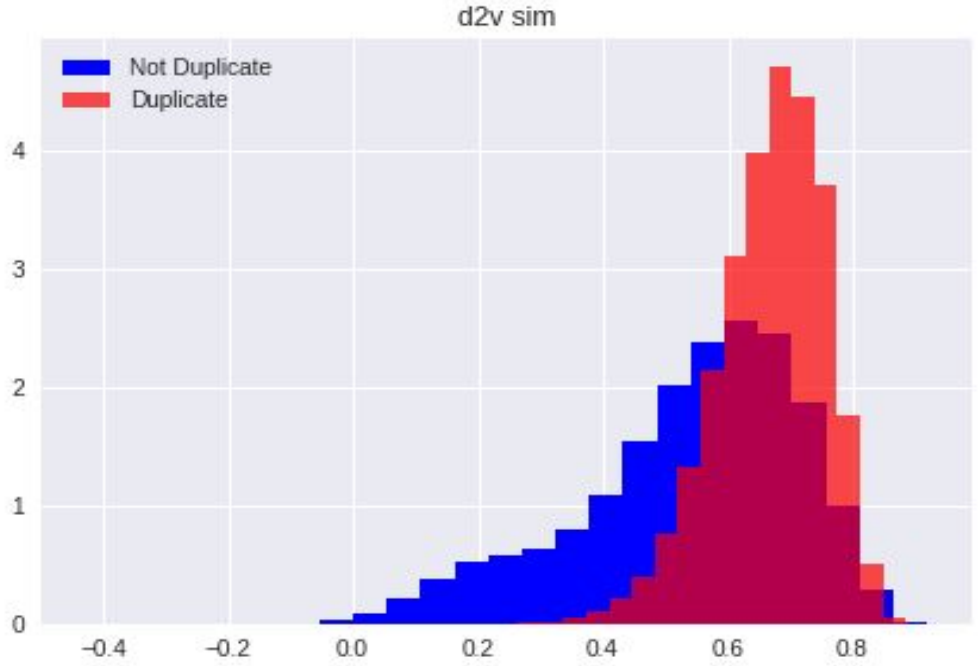
看起來 WMD 大小和是否duplicate 的相關程度沒有上一個高，我想原因可能是問題的句型類似，比如都是 What are … the best …? 或 How …? 所以有決定性差異的字被稀釋了。

三、document embedding

WMD 只能直接算出document 的距離，但可想而知隱藏在document 中的資訊肯定更多，所以我另外使用了兩種 document embedding：TF-IDF 和 doc2vec。其中TF-IDF 相似度的效果為：



勉強對於non-duplicate 的分辨有幫助。至於 doc2vec 我使用gensim 來抽取，相比於WMD 和 TF-IDF，doc2vec 向量的維度較低，還可以直接給 neural network 做訓練。而doc2vec 相似度的效果為：



和 WMD 對 label 相關度差不多。

四、common word match

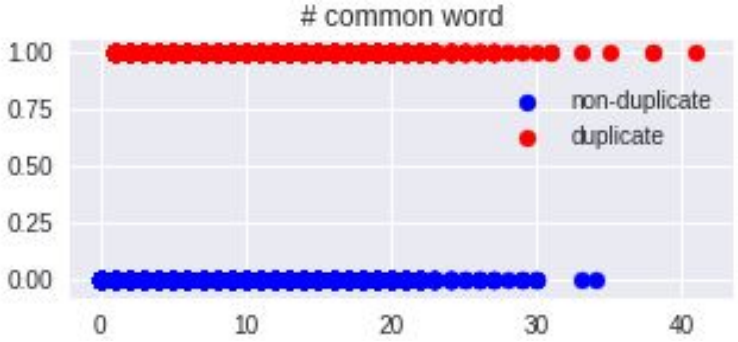
造成問題重複的原因，還可能是描述的詳細程度不同，比如：

Q1: Can a person lose weight without going to the gym?	duplicate
Q2: Can I lose weight and grow muscle without going to the gym or use weights?	

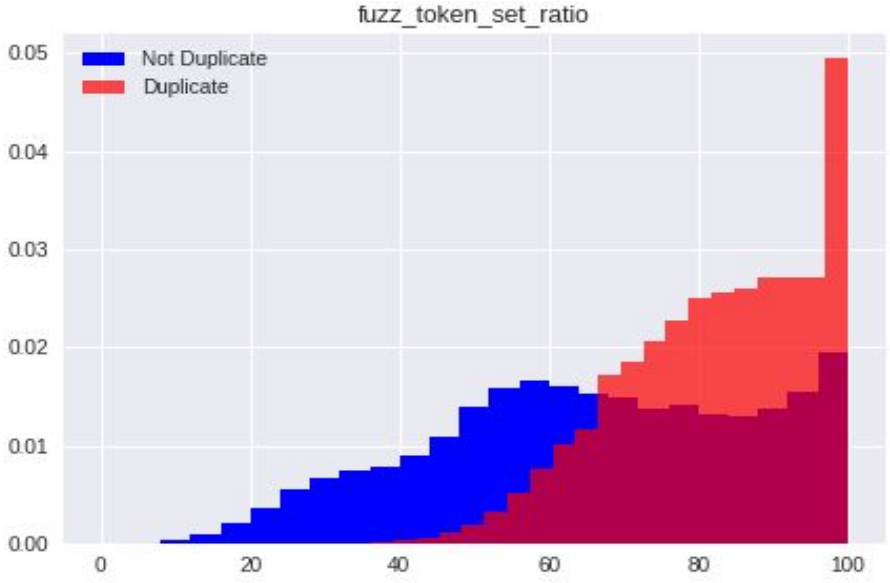
這時適合做common word match，數總共有幾個相同的字，當然也存在反例，比如：

Q1: What is the step by step guide to invest in share market?	non-duplicate
Q2: What is the step by step guide to invest in share market **in india**?	

不過這項feature 的分辨效果很適合抓出duplicate pair：



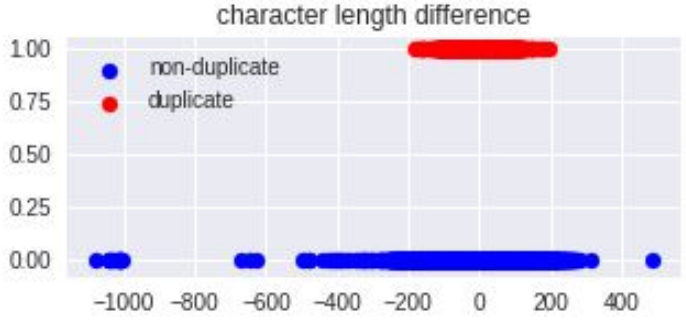
另外，假如在比對時多考慮字義相似度，便可以做到fuzzy 的common word match，效果如下：



可以預期fuzzy 的common word match 將 duplicate 錯分成 non-duplicate 的機率會比hard 的common word match 低很多。

五、基本的NLP feature

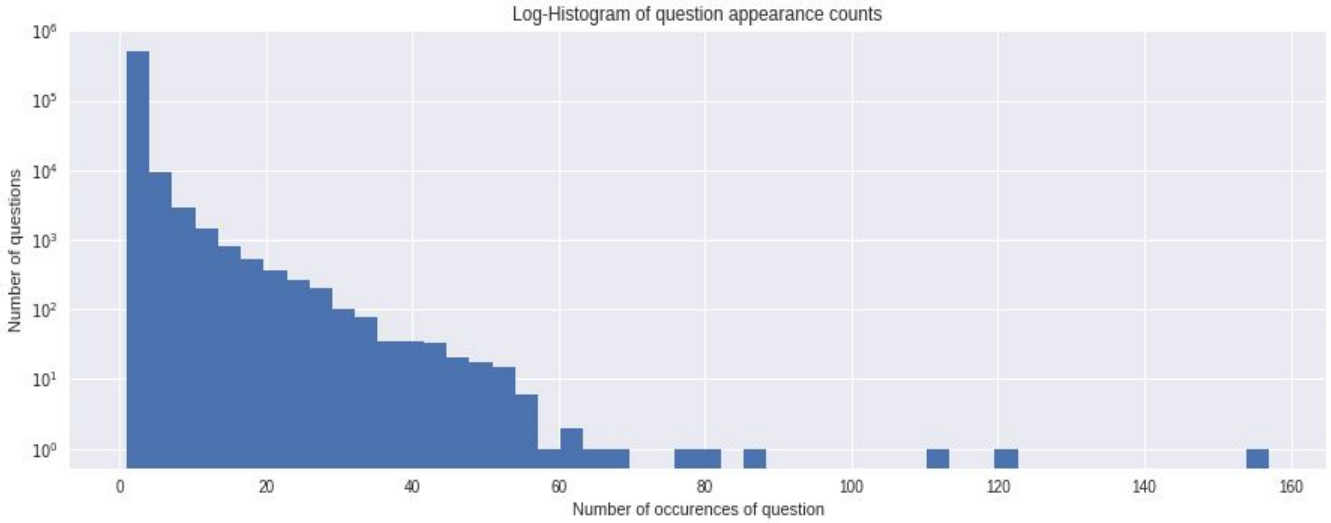
由於 dataset 中存在不完整的或是根本已經被刪掉的問題，所以做基本的NLP 分析，如：統計句子長度、計算問題pair 句子長度差距等，可以預期有用：



六、Magic Feature

這項feature 是在我初次上傳預測後，發現排名大幅落後其他選手，不甘

心去比賽討論頁面查看時發現的一張圖：

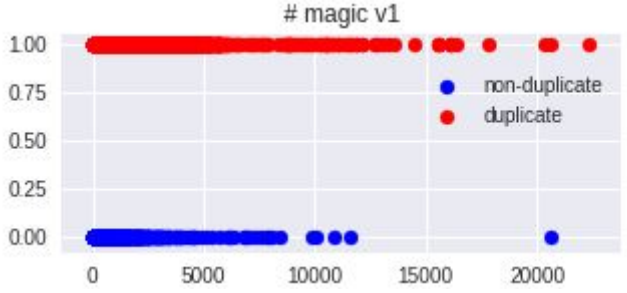


這張圖統計了每個問題出現在 dataset 中的次數，出現次數越多表示被挑出來做越多次 duplicate pair 的比較。但是，每個問題被比較的次數，假如問題是隨機挑選，照理說應該是差不多的。然而，統計結果說明和我們預期完全相反！大部分問題只被比較一次，而存在少數問題被比較非常多次 80次、120次、甚至是158 次！為何問題被比較次數會差這麼多？可以合理猜測這是因為問題pair 的產生是由Quora 內部某種演算法挑選的，所以假如一個問題被挑中越多次，就表示這個問題在那個演算法看來越有可能是 duplicate 的問題！

因此，我抽取的 magic feature 第一個就是

(1) 問題被比較的次數

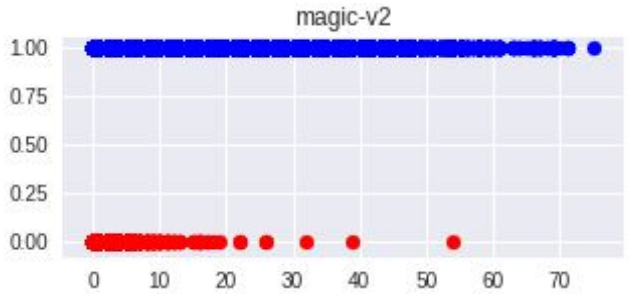
這部分的實作有個 trick 是因為我只有在training set 上有問題的ID 但testing set 沒有，為了能夠在 testing set 做 predict，我將問題文字先透過一個 hash function 得到hash 值，並用這個 hash 值當作問題 ID，如此在training set 和 testing set 都能 infer 每個問題被比較次數了。這項 feature （將兩問題出現次數相乘）和label 的相關度還行：



(2) 這組問題 pair 比較過的問題個數

另一個觀察是這張圖的y 軸是log-scale 的問題個數，對於x 軸的問題出現次數呈現power-law 分佈，由此我甚至猜測整個dataset 產生方法就是，初始時，使用幾個問題當作seed，接著使用 Quora 內部演算法篩出跟這些 seed 比較有可能是 duplicate 的candidate 問題，然後對於 candidate 問題再挑出跟其有可能是類似的 candidate 的問題，不斷重複以上步驟，直到挑出固定數量的問題 pair。所以，當作 seed 的或是本身有非常多 candidate 的問題會被比較非常多次，才會出現 power-law 分佈。假如每個問題是一個點，兩個問題被比較就有一條邊的話，那麼問題被比較的圖應該會聚集成一個個的 community，我們可以合理猜測假如兩問題出現在同個 community，它們是 duplicate 的機率比較高。根據這個想法，

我抽出第二個 magic feature，統計有幾個共同問題被拿來跟這兩個問題比較過，和label 的相關度：



比第一個 magic feature 高。

總計，抽取的 feature 共 49 個。

一、character-level similarity
- char_ratio, char_diff, char_diff_unq_stop, jaccard

二、word-level similarity
- 句子中心距離：cosine_distance, cityblock_distance, canberra_distance, minkowski_distance, braycurtis_distance
- WMD: wmd, norm_wmd, wmd_q2join, wmd_q2join

三、document embedding
- TF-IDF: tfidf_sim, tfidf_stop_sim
- doc2vec: d2v_dim

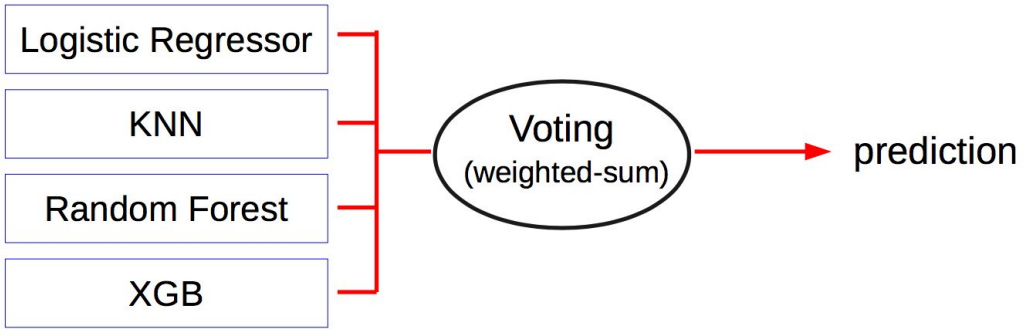
四、common word match
- Hard: word_match, wc_ratio, wc_diff, wc_diff_unique, wc_diff_ratio_unique, wc_diff_unq_stop, wc_ratio_unique_stop
- Fuzzy: fuzz_ratio, fuzz_qratio, fuzz_word_ratio, fuzz_partial_ratio, fuzz_partial_token_set_ratio, fuzz_partial_token_sort_ratio, fuzz_token_set_ratio, fuzz_token_sort_ratio

五、基本的NLP feature
- len_char_q1, len_char_q1, len_word_q1, len_word_q2, len_char_diff, len_word_diff, same_start, total_unique_words, total_unique_words_stop
- skew_q1vec, skew_q2vec, kur_q1vec, kur_q2vec

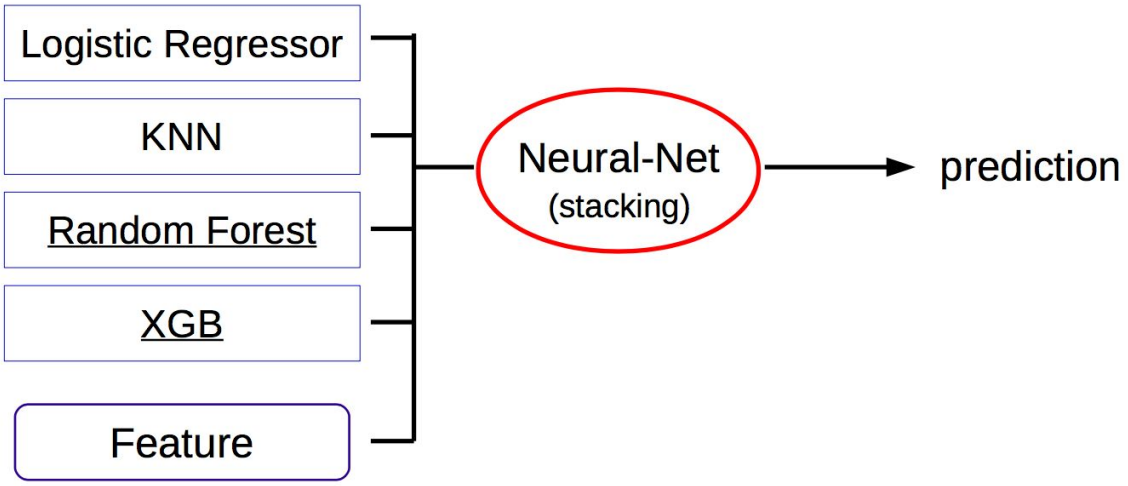
六、Magic Feature
- magic-v1: q1_freq, q2_freq
- magiv-v2: q1_q2_intersect

Classification Mode

我嘗試了兩種架構，以下為基本款：



我使用了4 個 classifier 用以上的 feature 訓練產生 prediction，接著將 4個 classifier 的結果用 weighted-sum 的 voting 結合出最終的 prediction。而這 4 個 classifier 是特意選擇的組合，因為 data 量大所以使用 logistic regressior 做訓練很有效率，而 KNN 本身是看 instance-level 的相似度，相比於另外 3個 classifier 看 feature，觀點上有互補的作用，所以可以預期做 voting ensemble 有機會表現很好。至於 Random Forest 和 Extreme Gradient Boosting Tree (XGB) 都是 Tree-based 的 classifier，而上個段落在 explore feature 時可看出這些 feature 相當適合找一個 threshold 分成兩半，所以可以預期這兩個 Tree-based classifier 會表現得很出色。
而進階版：



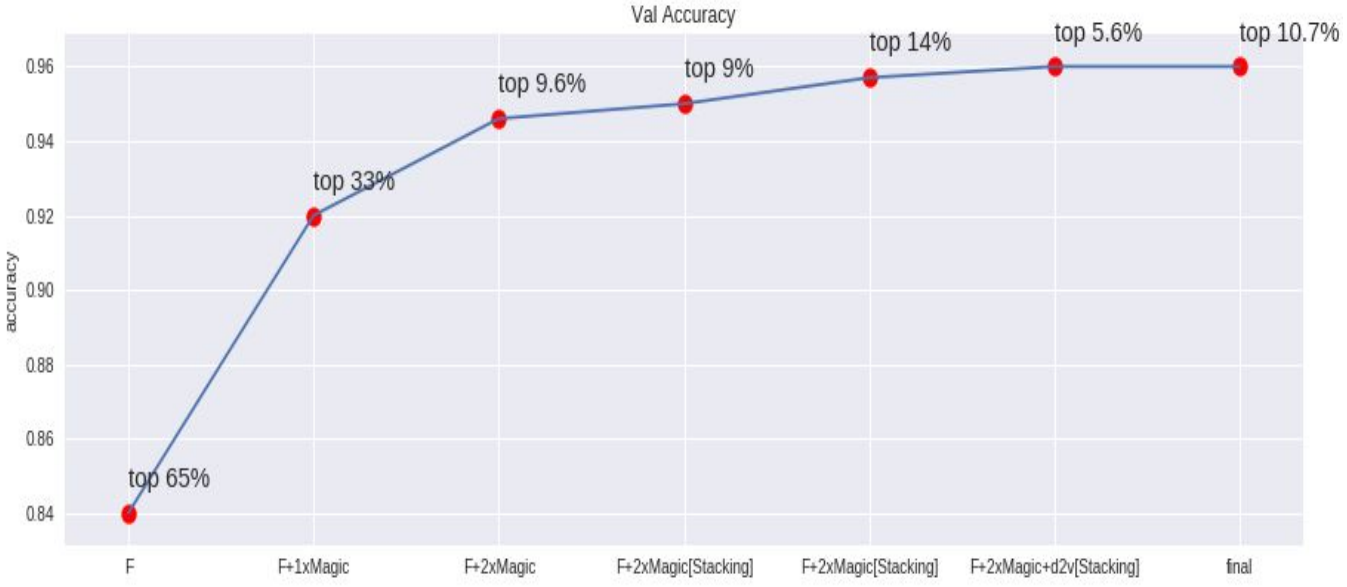
則是將 voting 的部分，改用一個 classifier 取代，相比於固定 ensemble weight，使用 classifier 更有彈性。基於 classifier 能力考量，我選擇使用 neural network 作為第二層的 ensemble classifier，其 input 除了和第一層 base classifier 相同的 feature、base classifier 的 prediction外，還多加上了 doc2vec 的 document 向量做訓練。

Evaluation

我使用 Holdout-CV 將 0.2 的 training set 作為 validation set。在使用基本 classification model 以及基本 feature (不包含 magic feature 以及 doc2vec feature) 時的 performance 是

	LR	KNN	RF	XGB
Val-AUC	0.75	0.79	0.84	0.83
weight	0	0	12	3
Voting Val-AUC	0.84			

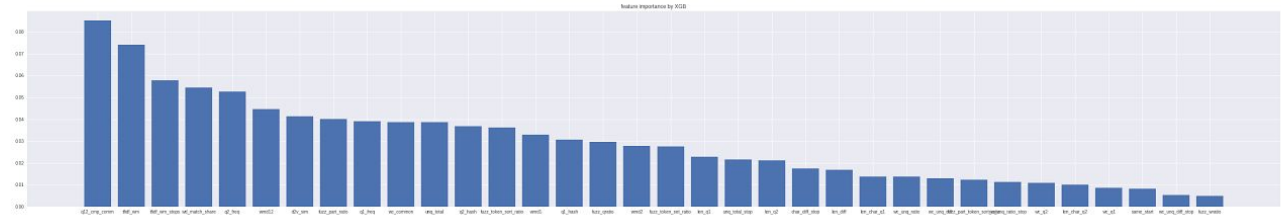
Validation accuracy 有84%，其中 tree-based 的 RF 和 XGB 果然表現比較好，但在比賽排行榜只有 top 65%。我陸續做了改進，以下為個階段的 progress 圖：



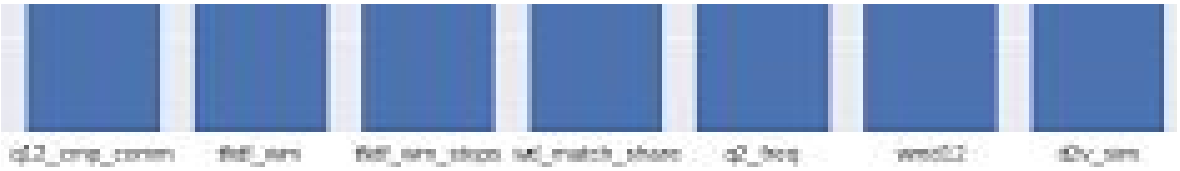
- F 為上述的基本款
- 使用第一個 magic feature 後，validation accuracy 到達 92% 在比賽排行榜晉升到 top 33%
- 繼續加入第二個 magic feature，validation accuracy 飆升 94% 排名進入 top 10%
- 將 classification model 換成使用NN 做 stacking 的改進版後，validation accuracy 上升 0.8% 排名小幅上升
- 但一段時間沒更新排名下降到top 14%。
- 最後一個改進是，加入doc2vec feature 做訓練，並且在訓練時調整 training set 的 label 分佈使之與 testing set一致，做完得到的 validation accuracy 上升到 96%，排名晉級到 top 5.6%
- 排名隨時間下降到最後比賽結束時，掉到top 10.7%

Feature Importance

我使用最終訓練好的 model 中的 XGB 為 feature 重要程度做排序：



XGB 認為重要程度最高的 feature 是 magic-v2、TF-IDF 相似度、common word match、magic-v2、WMD、以及 doc2vec 相似度。



Conclusion

在這次比賽中，使用的model 在加入合理的想法後，可以立即在 performance 上觀察到提升，比如：使用 stacking 做更好的 model ensemble、解決 label 分佈在training 和 testing 不一致問題後，確實得到了performance 的提升。

但需要特別說明的是，其中提升效果最明顯的是 magic feature，假如實務上會不斷有新的問題新增到dataset 的話，這項feature 必須不斷更新，已訓練好的 classifier 必須重新訓練，所以 magic feature 在實務上的使用情況可能非常有限。

Reference

[1] Kusner, M. J., et al. From Word Embeddings To Document Distances. ICML, 2015.