What's Cooking?

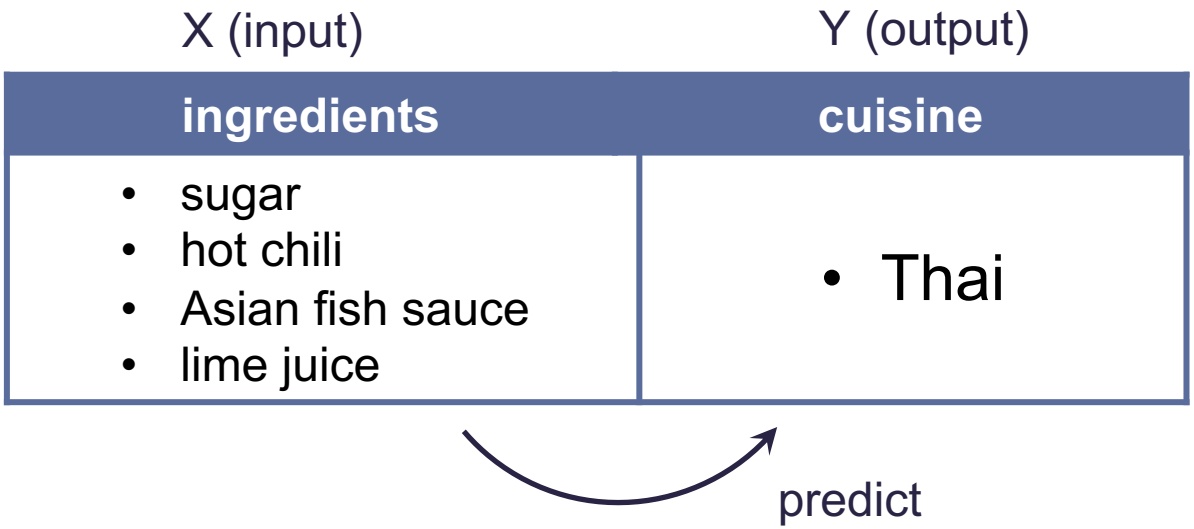
利用食材組成預測美食種類

統計112 宋穎恩
統計112 劉恩兆
財金產碩112 丁豪

Problem Statement

利用食材組成(ingredients)預測對應的美食種類(cuisine)

e.g.,



Competition on Kaggle：https://www.kaggle.com/competitions/whats-cooking/overview

Solved Challenges

資料維度的疑慮：

經計算，不重複的食材有6,714種，若是直接使用 one-hot encoding 可能會使資料維度過大而造成硬體不足

資料處理：

處理 input 資料有很多種方法（如:詞向量、one-hot encoding、label encoding 等)，需多加試驗以找出最合適的處理方法

模型選擇：

此資料集有許多模型種類可以使用，如何挑選適合的模型種類，以及如何調整模型以提高準確率都需要多加探索

Used Dataset

① 訓練集　(39,778 rows * 3 columns)

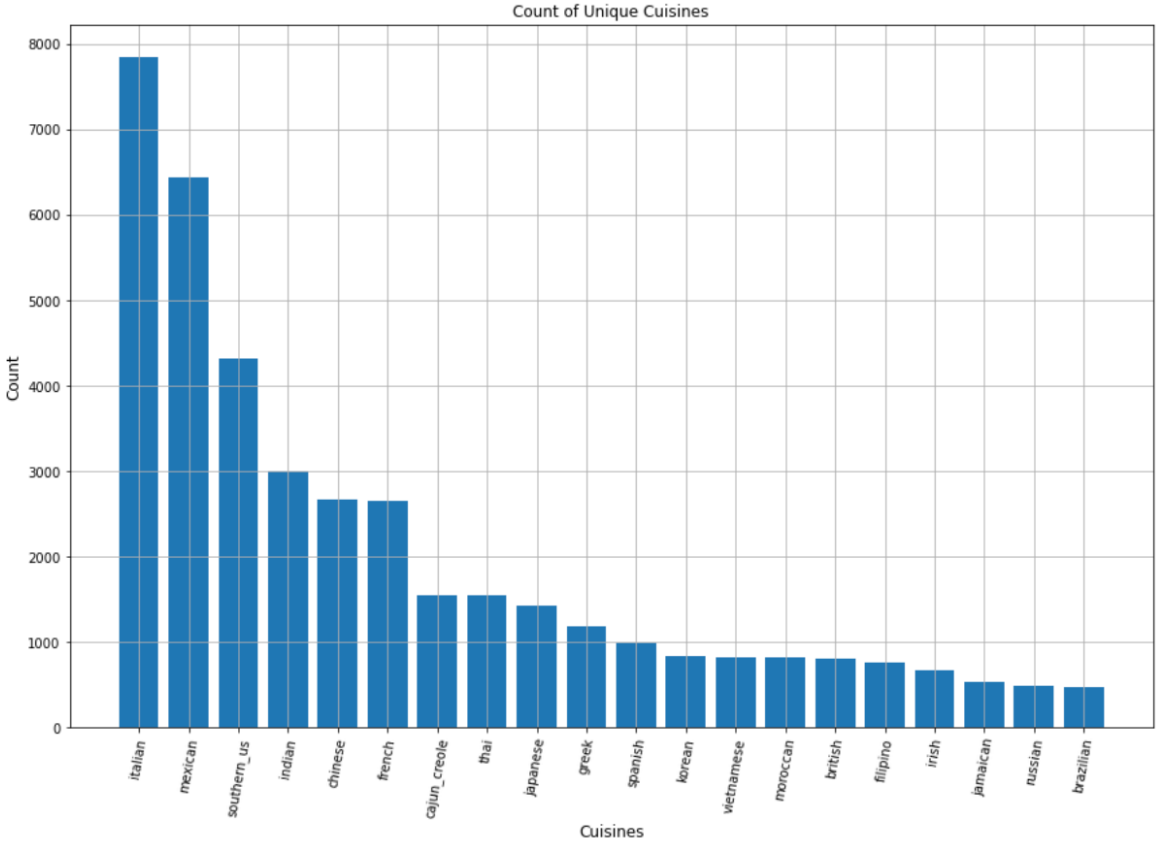
id	ingredients	cuisine
10259	[romaine lettuce, black olives, grape tomatoes, garlic, pepper, purple onion, seasoning…]	Greek

② 測試集　(9944 rows * 2 columns)

id	ingredients	cuisine
18009	[baking powder, eggs, all-purpose flour, raisins, milk, white sugar]	To be predicted

⬆ 預測值

Explorative Data Analysis

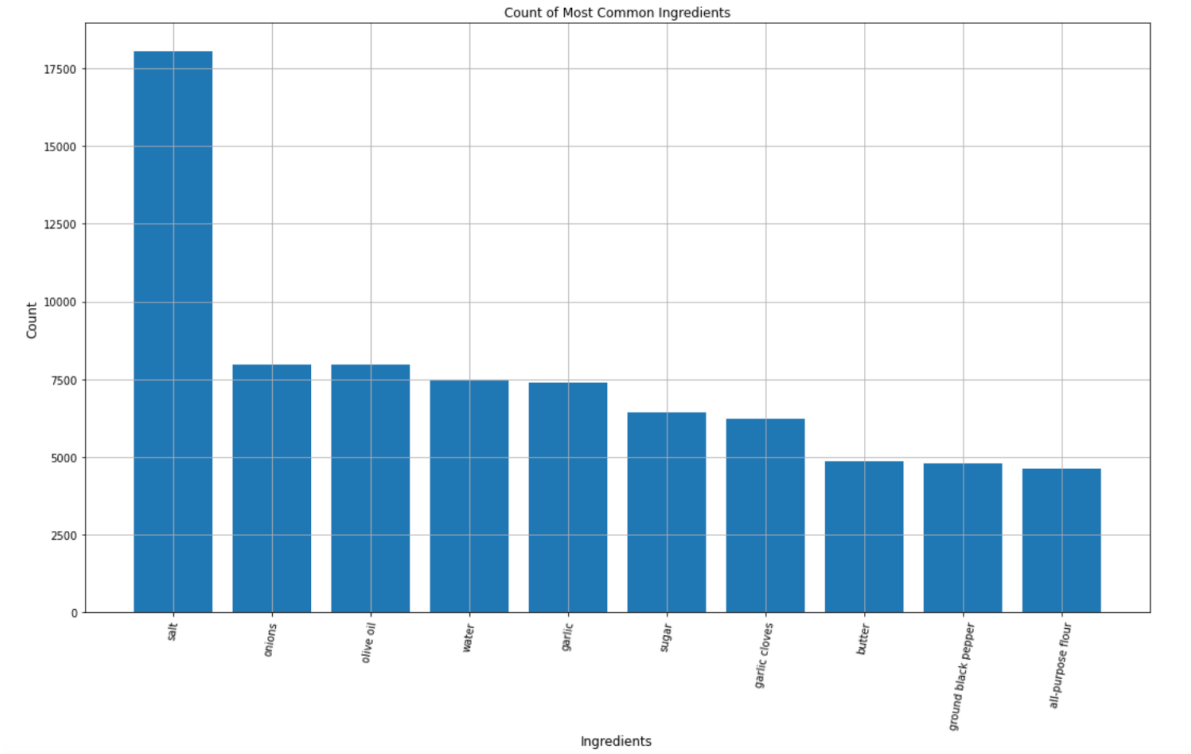


Cuisines 種類統計

- 總共有20種不同的 Cuisines

1. Italian
2. Mexican
3. Southern_us

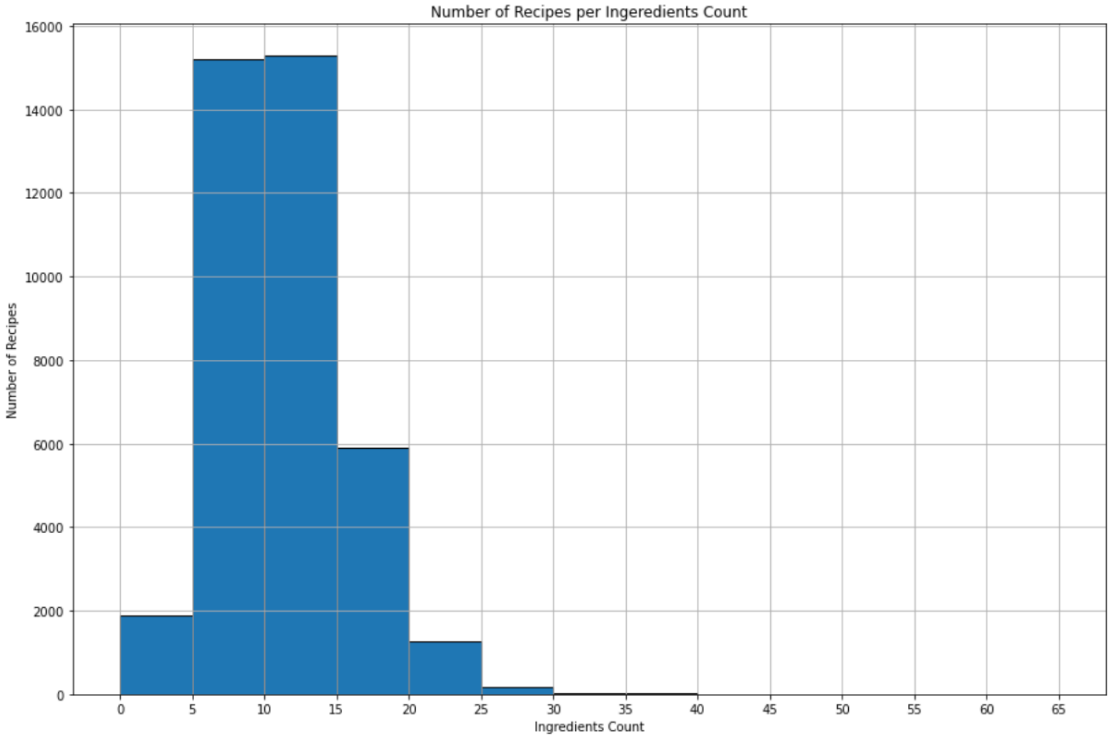
Explorative Data Analysis



Ingredients 種類統計

1. Salt
2. Onions
3. Olive oil

Explorative Data Analysis



Ingredients 使用個數統計

- 大多數的 Cuisines 由 5-15種 ingredients 構成

Proposed Methods

資料前處理：

- TF-IDF 、One-Hot Encoding

基本演算法：

- Logistic Regression 、 Decision Tree 、 Random Forest 、 Support Vector Machine

嘗試其他演算法與調整模型參數：

- LightGBM 、XGBoost 、Gradient Boosting

Data processing

Step 1：資料整理

- 將 list 轉成 string，用 join() 將字詞加入字串中，用空格做區隔

Step 2：檢查遺失值

- 使用is.null().sum()檢查是否有遺失值，於訓練集與測試集均未發現遺失值

Data processing

Step 3：資料前處理 TF-IDF

- Term Frequency (TF) 詞頻：找到出現次數最多的詞
- Inverse Document Frequency (IDF) 逆文檔頻率：在詞頻的基礎上，對每個詞分配一個重要性的權重，最常見的詞給予最小的權重，較少見的詞給予較大的權重，也就是大小與一個詞的常見程度成反比

  E.g., 像是 a/an/the 等常出現的詞語就會被分配到最低的分數
- 公式：*TF-IDF = TF * IDF*

  其大小與一個詞在文檔的出現次數成正比，與該詞在整個語言中的出現次數成反比

Data processing

Step 3：資料前處理 TF-IDF

- 從 sklearn 套件 import CountVectorizer 和 TfidfVectorizer
- 創建詞袋數據結構，透過 CountVectorizer 中的 transform 函數將文本中的詞語轉換成詞頻矩陣
- 利用 TfidfVectorizer 建構一個計算 TF-IDF 的函式，停用詞使用英語內建的停用詞列表，依詞頻給予各食材不同的權重，最終將詞頻矩陣轉換成 TF-IDF 權重矩陣
- 將文字資料轉為數字組成矩陣，方可以放進模型計算

Baseline Models

	Accuracy	
Model	TF-IDF	One-Hot Encoding
Logistic Regression	**0.7856**	0.1927
Decision Tree	**0.6258**	0.5413
Random Forest	**0.7556**	0.7098
SVM	**0.7816**	0.2358

註：參數皆使用預設參數

SVM 調整參數

- 根據上一頁 baseline models 的結果，我們挑選表現較佳的 SVM 來進行參數調整，以追求更高的準確率
- 調整參數的方法為 GridSearchCV ，而最終使用的參數分別為：

  C = 10

  kernel = ‘rbf’（ Gaussian 高斯 ）

  gamma = ‘scale’

  decision_function_shape = ‘ovo’（ one-vs-one 一對一 ）
- 最終結果 Score 從 0.7816 進步到 0.8108

Advanced Models

- 因為 baseline models 中 TF-IDF 表現明顯較 one-hot encoding 好，因此 advanced models 皆使用 TF-IDF 做資料前處理

Model	Accuracy
XGBoost	0.7325
LightGBM	**0.7764**
Gradient Boosting	0.7387

註：參數皆使用預設參數

LightGBM 調整參數

- 根據上一頁 advanced models 的結果，我們挑選表現結果較佳的 LightGBM 來進行參數調整，以追求更高的準確率
- LightGBM 主要調整的參數為 learning rate 以及 n_estimators ，而調參數方法為依上述順序一次調一種參數，同時其他參數固定不變，在最佳化此參數後即固定不動，依序調整完此兩種參數

LightGBM

Learning Rate

Learning rate	Accuracy	Precision
0.001	0.3094	0.314
0.01	0.6777	0.6774
0.05	0.7612	0.7618
0.1	0.7764	0.7788
0.25	**0.7803**	**0.7785**

LightGBM 調整 learning rate 的表現結果

LightGBM

n_estimators (learning rate = 0.25)

n_estimators	Accuracy	Precision
10	0.7231	0.7206
50	0.7721	0.7721
100	0.7803	0.7785
500	0.7845	0.7828
1000	**0.7862**	**0.7853**

LightGBM 調整 n_estimators 的表現結果

Experimental Results and Analysis

- 由初步模型分析結果可以得知，SVM 與 Logistic Regression 的表現結果較其他模型佳（包含 advanced models）
- One-hot encoding 在模型分析中皆表現較差，推測是因為資料處理後過於分散，較難抓到資料間的關聯性
- SVM 為表現最佳的模型，經參數調整後，LightGBM 最佳參數為 learning rate = 0.25 與 n_estimators = 1000，得到 0.7862 的分數，若再多嘗試其他參數的調整，有望達到更高的表現結果

Discussions

- 這個競賽排行榜最高分數為 0.8322，而我們最高分數為 0.8108，表示還有優化的空間。目前優化方向以選擇其他文字處理方法優先，而非模型
- TF-IDF 優點：簡單易理解，運算速度快
- TF-IDF 缺點：詞頻衡量並沒有很全面，沒有考慮到類別間的偏差

	THAI Lemon	THAI Pepper	SU Pepper
出現次數	100	200	200
TF-IDF分數	100*1=1	200*1/2=1	200*1/2=1

若檸檬只出現在泰國菜，而胡椒只出現在泰國與南美洲菜，如表格情況所示，TF-IDF 就會給予相同的權重，無法彰顯出檸檬對泰國菜獨特的重要性

Todo List for Final Reports

文字處理：

- 除了 one-hot encoding 跟 TF-IDF 以外，還可以嘗試 Word2Vec 跟 Bert 來訓練 word embedding

模型：

- LightGBM可以新增調整的參數種類，像是 objective、max_bin以及 max_depth，以達到更好的表現結果
- 嘗試使用 Neural Network

THE END

Thank You !