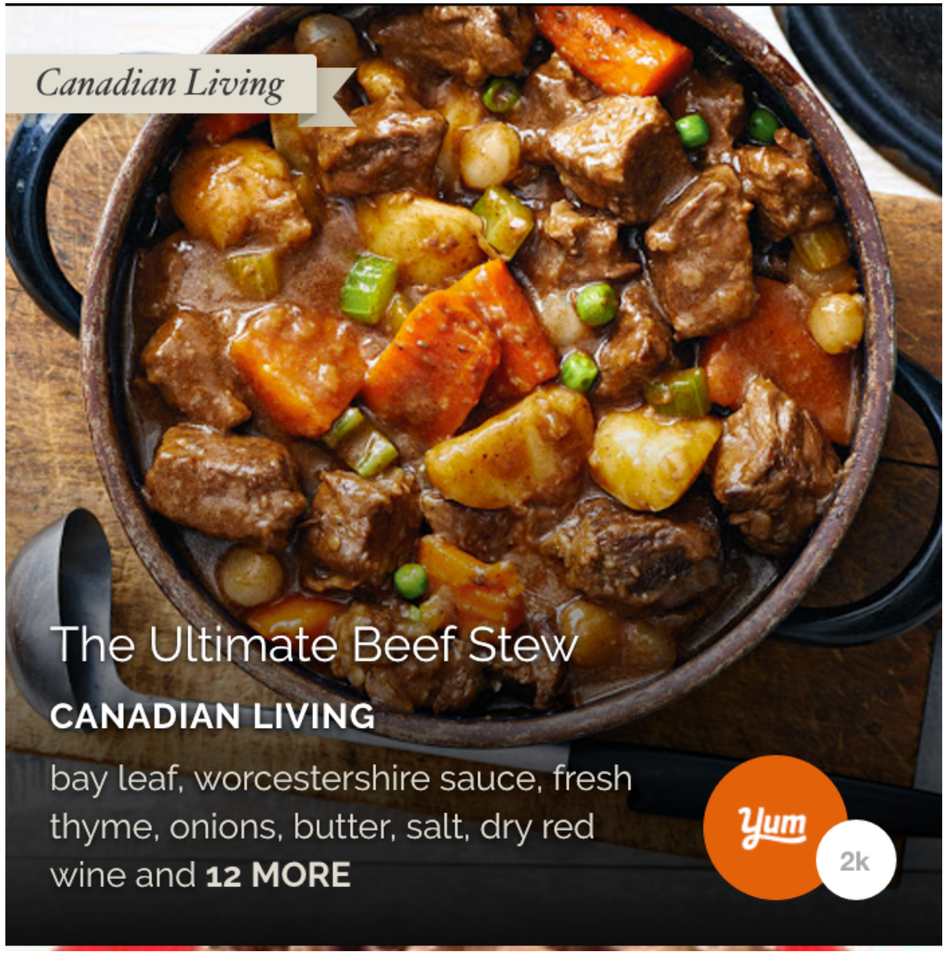
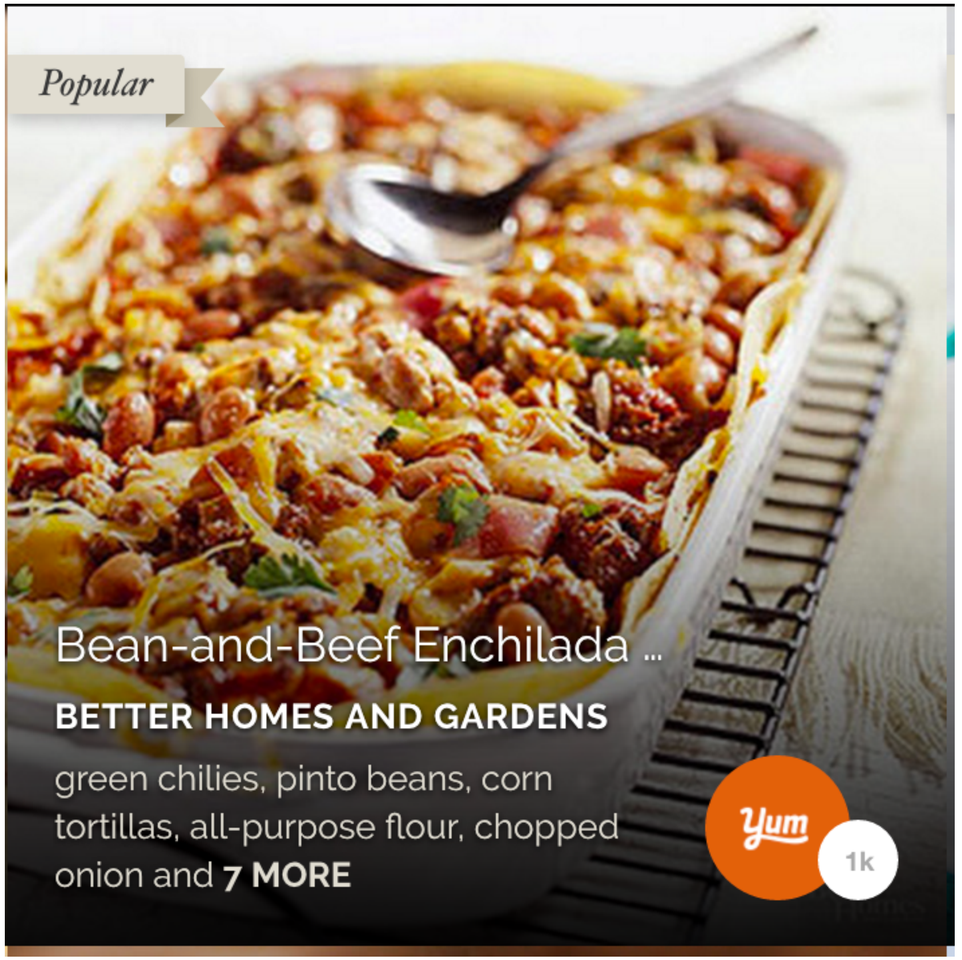
Cuisine Prediction

- Project Introduction
- Dataset
- Modules
- Future Improvements

- Project Introduction
    - Use recipe ingredients to categorize the cuisine
    - For example: if the ingredients contain things like pasta, olive oil, canned tomatoes and tomato paste, garlic, parmesan cheese, fresh basil, pesto, etc., it’s probably Italian food.
    - NLP project


Canadian Living
The Ultimate Beef Stew
CANADIAN LIVING
bay leaf, worcestershire sauce, fresh thyme, onions, butter, salt, dry red wine and 12 MORE
Yum
2k


Popular
Bean-and-Beef Enchilada ...
BETTER HOMES AND GARDENS
green chilies, pinto beans, corn tortillas, all-purpose flour, chopped onion and 7 MORE
Yum
1k

- Dataset
    - This dataset was from Kaggle. It's provided by Yummly
    - I divided its training dataset into my training (0.8) and testing dataset(0.2).

- Dataset

	cuisine	id	ingredients
0	greek	10259	[romaine lettuce, black olives, grape tomatoes...
1	southern_us	25693	[plain flour, ground pepper, salt, tomatoes, g...
2	filipino	20130	[eggs, pepper, salt, mayonaise, cooking oil, g...
3	indian	22213	[water, vegetable oil, wheat, salt]
4	indian	13162	[black pepper, shallots, cornflour, cayenne pe...

  - 39774 observations, 3 columns, no null values.
  - Features(X): ingredients (need to be fit and transformed)
  - Response(y): cuisine
  - Dumb Model Accuracy Rate: 0.19

- Dataset
    - Intuition: I feel using ‘black olives’ as the token is better than ‘black’ and ‘olives’.
    - Attention: Fit and transform the training data, transform the test data.
    - Using Bag of Words because it’s short text dataset.

- Models: Naive Bayes

```
cuisine_nb = Pipeline([('vect', CountVectorizer()),
                       ('nb', MultinomialNB())])

parameters = {'vect__token_pattern':[r"\b\w\w+\b", r"'([a-z ]+)'"],
              'vect__min_df':[1,2,3],
              'nb__alpha':[0,0.1,0.5,1]}

gs_nb = GridSearchCV(cuisine_nb, parameters, cv = 10, n_jobs = -1)
```

- Models: Naive Bayes
    - The accuracy score and best parameters:

```
fit_nb.score(X_test, y_test)
```

```
0.7537397862979258 2
```

```
fit_nb.best_params_
```

```
{'nb__alpha': 0.1, 'vect__min_df': 1, 'vect__token_pattern': "'([a-z ]
+)'"}
```

- Models: Random Forest

```
cuisine_rfc = Pipeline([('vect', CountVectorizer()),
                        ('rfc', RandomForestClassifier())])
parameters = {'vect__token_pattern':[r"'([a-z ]+)'"],
              'rfc__max_features' : [10,20],
              'rfc__n_estimators':[1000]}

gs_rfc = GridSearchCV(cuisine_rfc, parameters, cv = 5, n_jobs = -1)
```

- Models: Random Forest
    - The accuracy score and best parameters:

```
fit_rfc.score(X_test, y_test)
```

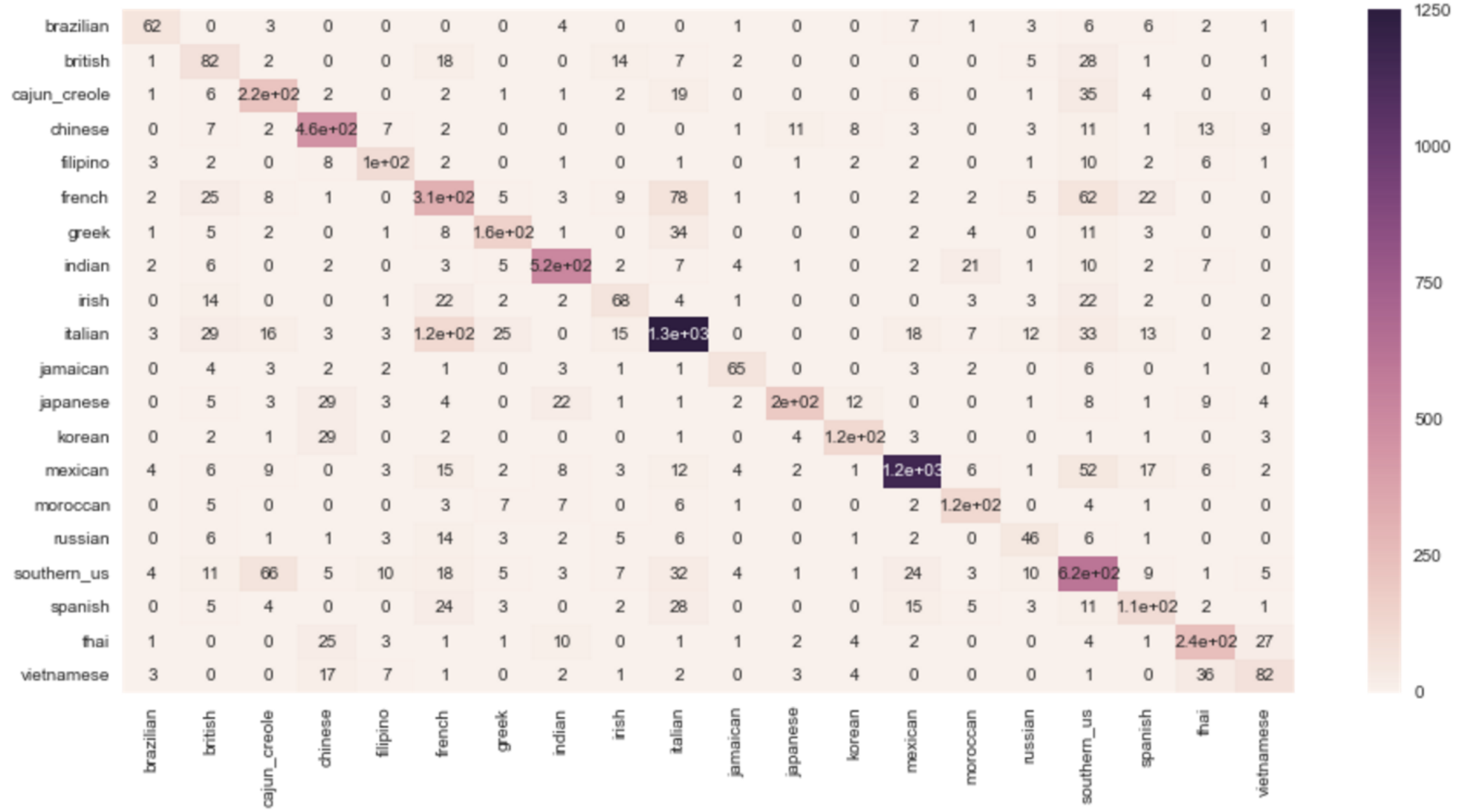
```
0.7285936580766812
```

```
fit_rfc.best_params_
```

```
{'rfc__max_features': 10,
 'rfc__n_estimators': 1000,
 'vect__token_pattern': "'([a-z ]+)'"}
```

- Models:
    - Model Comparison: Naive Bayes is better. So I use Naive Bayes model to predict the test data.

Confusion Matrix



- Future Improvements
    - Try to group the cuisine into similar groups based on their taste, then predict cuisine in each group. (Model Stacking)
    - Try other NLP packages to analyze the data.