Kaggle: What’s cooking?

CS570 Final Project
Qiyang Zhou, Rongmei Lin, Liquan Bai
Outline

- Introduction
- Data preprocessing
- Our proposed methods
- Experiments
Introduction

This study sought to apply algorithms learned from the data mining course to classify 20 forms of cuisines through the analysis of the ingredient lists.
Data description

The data we are using is from the Kaggle “What’s Cooking?” competition. There are training set and testing set in the data and both in JSON format.

- train.json – 39774 records containing recipe id, type of cuisine and list of ingredients
- test.json – 9942 records containing recipe id and list of ingredients

One example of recipe record in train.json:

{"id": 24717,"cuisine": "indian","ingredients": ["tumeric","vegetable stock","tomatoes","garam masala","naan","red lentils","red chili peppers","onions","spinach","sweet potatoes"],}
Data description

There are 20 different cuisines in the dataset.

Italian: 7838
Mexican: 6438
Southern US: 4320
Indian: 3003
Chinese: 2673
Others: 15502
There are 6714 unique ingredients in the dataset.

Top 1  ('salt', 3454)
Top 2  ('olive oil', 3111)
Top 3  ('garlic cloves', 1619)
Top 4  ('grated parmesan cheese', 1580)
Top 5  ('garlic', 1471)
Top 6  ('ground black pepper', 1444)
Top 7  ('extra-virgin olive oil', 1362)
Top 8  ('onions', 1240)
Top 9  ('water', 1052)
Top 10 ('butter', 1030)
Data description

For each cuisine we could also count the most frequent ingredients.

1. Most of frequent ingredients are salty seasonings.
2. Soy sauce is top 1 ingredient for chinese and for most asian cuisines.
3. Some ingredients have more than one name: e.g. garlic and garlic cloves.
Except the frequent ingredients in specific cuisines, there are some ingredients with high degree of uniqueness.

- Soy sauce (Asian cuisine)
- Sesame oil (Asian cuisine)
- Sake (Japanese cuisine)
- Garam masala (Indian cuisine)
- Ground ginger (Moroccan cuisine)
- Avocado (Mexican cuisine)
Data preprocessing

Text mining and Natural Language Processing techniques could be used to derive useful information from raw data.

- **Change the text case:** Data is often received in irregular formats. E.g. ‘Salt’ and ‘salt’ both means the same ingredient.
- **Deal with the punctuation:** Python would read ‘data mining’ and ‘data-mining’ as two different words.
- **Remove stopwords:** Stopwords are nothing but the words which don’t describe any sentiment. We could also remove words repeated but add no value.
- **Stemming:** This method aims at obtaining root of derived words. E.g. ‘play’, ‘playing’ and ‘played’ can be stemmed into ‘play’.
Data preprocessing example

**original**

In [55]:
```plaintext
words = "Hellmann's Mayonnaise with cheese. and tri@ color PASTA twists, cooked and drained"
print words
```

```
Hellmann's Mayonnaise with cheese. and tri@ color PASTA twists, cooked and drained
```

**remove actions**

In [56]:
```plaintext
words = words.split("",")[0]
print words
```

```
Hellmann's Mayonnaise with cheese. and tri@ color PASTA twists
```

**convert to lowercase**

In [57]:
```plaintext
words = words.lower()
print words
```

```
hellmann's mayonnaise with cheese. and tri@ color pasta twists
```
Data preprocessing example

split words using sklearn's default tokenizer pattern

```python
In [58]:
words = re.findall(r"\b\w+\b", words)
print words

['hellmann', 'mayonnaise', 'with', 'cheese', 'and', 'tri', 'color', 'pasta', 'twists']
```

remove stopwords

```python
In [59]:
words = [x for x in words if x not in stopwords]
print words

['hellmann', 'mayonnaise', 'cheese', 'tri', 'color', 'pasta', 'twists']
```

stemming

```python
In [60]:
words = [stemmer1.stem(w) for w in words]
words = [str(x) for x in words]
print words

['hellmann', 'mayonnais', 'chees', 'tri', 'color', 'pasta', 'twist']
```
Algorithm

Distance:
- Jacard
- TF-IDF

Evaluation:
- K-means purity

Classification:
- Nearest center
## Distance matrix

<table>
<thead>
<tr>
<th>ingredient</th>
<th>salt</th>
<th>doubanjiang</th>
<th>shirataki</th>
</tr>
</thead>
<tbody>
<tr>
<td>brazilian</td>
<td>0.01</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>british</td>
<td>0.02</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>cajun_creole</td>
<td>0.04</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>chinese</td>
<td>0.05</td>
<td>1000</td>
<td>-1000</td>
</tr>
<tr>
<td>filipino</td>
<td>0.02</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>french</td>
<td>0.07</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>greek</td>
<td>0.03</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>indian</td>
<td>0.12</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>irish</td>
<td>0.02</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>italian</td>
<td>0.24</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>jamaican</td>
<td>0.02</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>japanese</td>
<td>0.02</td>
<td>-1000</td>
<td>1000</td>
</tr>
<tr>
<td>korean</td>
<td>0.01</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>mexican</td>
<td>0.18</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>moroccan</td>
<td>0.02</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>russia</td>
<td>0.02</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>southern_us</td>
<td>0.15</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>spanish</td>
<td>0.03</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>thai</td>
<td>0.02</td>
<td>-1000</td>
<td>-1000</td>
</tr>
<tr>
<td>vietnamese</td>
<td>0.01</td>
<td>-1000</td>
<td>-1000</td>
</tr>
</tbody>
</table>
Results

Training accuracy: 96%

Testing accuracy: 75%

class_test.csv
6 days ago by Qiang Zhou

add submission details
## Kaggle Leaderboard

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Team Name</th>
<th>Kernel</th>
<th>Team Members</th>
<th>Score</th>
<th>Entries</th>
<th>Last</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>yilihome</td>
<td></td>
<td></td>
<td>0.83216</td>
<td>31</td>
<td>2y</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>pavelgonchar</td>
<td></td>
<td></td>
<td>0.82853</td>
<td>49</td>
<td>2y</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Kemal Atatürk</td>
<td></td>
<td></td>
<td>0.82461</td>
<td>163</td>
<td>2y</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>KKC</td>
<td></td>
<td></td>
<td>0.82441</td>
<td>11</td>
<td>2y</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Michal Tadeusiak</td>
<td></td>
<td></td>
<td>0.82300</td>
<td>11</td>
<td>2y</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Egor</td>
<td></td>
<td></td>
<td>0.82260</td>
<td>3</td>
<td>2y</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>Avishek</td>
<td></td>
<td></td>
<td>0.82210</td>
<td>73</td>
<td>2y</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>Saquib Ahmad</td>
<td></td>
<td></td>
<td>0.82089</td>
<td>11</td>
<td>2y</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>CMP_2016</td>
<td></td>
<td></td>
<td>0.82079</td>
<td>5</td>
<td>2y</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>Alaa Adel</td>
<td></td>
<td></td>
<td>0.82069</td>
<td>29</td>
<td>2y</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>Andrew Khalel</td>
<td></td>
<td></td>
<td>0.82049</td>
<td>84</td>
<td>2y</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>WaleedNader</td>
<td></td>
<td></td>
<td>0.82049</td>
<td>24</td>
<td>2y</td>
</tr>
</tbody>
</table>
Future Work

Try different classification methods

Optimized NLP process

Bag of words

Data mining on online recipe database
References

1.) one vs. All Classification: Train one classifier for each class, and assign samples to the class with the greatest probability.

2.) All vs. All Classification: Train one classifier for each pair of classes, and assign samples to the class pairs with the greatest probability.

3.) “Single Machine” approaches: Use the Bayes-optimal solution to train many binary classifiers.

4.) “Error Correcting Output Coding” approaches: Divide the multi-class categorization problem into a chain of binary classification issues)

5.) https://www.analyticsvidhya.com/blog/2015/12/kaggle-solution-cooking-text-mining-competition/

6.) http://www.akbarian.org/notes/text-mining-nlp-python/
Thank you!!