Descriptive_analysis

December 17, 2018

1 Descriptive analysis

Here is a little insight in to how the dataset looks

Most of this code is taken from:

https://www.kaggle.com/gloriahristova/a-walkthrough-eda-vizualizations-unigram-model/notebook

1.0.1 Imports

```
In [1]: # Data processing
        import pandas as pd
        import json
        from collections import Counter
        from itertools import chain
        from sklearn.feature_extraction.text import TfidfVectorizer
        import numpy as np
        import re

        # Data vizualizations
        import random
        import plotly
        from plotly import tools
        from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
        init_notebook_mode(connected=True)
        import plotly.offline as offline
        import plotly.graph_objs as go
```

1.0.2 Reading in the data file

```
In [2]: train_data = pd.read_json('train.json')
        print(train_data.head())
```

```
      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...
```

```
In [3]: print("The training data consists of {} recipes".format(len(train_data)))
```

```
The training data consists of 39774 recipes
```

A function for producing random colors for plots:

```
In [4]: def random_colours(number_of_colors):
            '''
            Simple function for random colours generation.
            Input:
                number_of_colors - integer value indicating the number of colours which are go
            Output:
                Color in the following format: ['#E86DA4'] .
            '''
            colors = []
            for i in range(number_of_colors):
                colors.append("#"+''.join([random.choice('0123456789ABCDEF') for j in range(6)]
            return colors
```

1.0.3 Number of recipes in each cuisine

```
In [5]: trace = go.Table(
                        header=dict(values=['Cuisine','Number of recipes'],
                        fill = dict(color=['#EABEB0']),
                        align = ['left'] * 5),
                        cells=dict(values=[train_data.cuisine.value_counts().index,train_data.
                       align = ['left'] * 5))

        layout = go.Layout(title='Number of recipes in each cuisine category',
                           titlefont = dict(size = 20),
                           width=500, height=650,
                           paper_bgcolor =  'rgba(0,0,0,0)',
                           plot_bgcolor = 'rgba(0,0,0,0)',
                           autosize = False,
                           margin=dict(l=30,r=30,b=1,t=50,pad=1),
                           )
        data = [trace]
        fig = dict(data=data, layout=layout)
        iplot(fig)
```

1.1 Percentage of each cuisine

```
In [6]: #  Label distribution in percents
        labelpercents = []
        for i in train_data.cuisine.value_counts():
            percent = (i/sum(train_data.cuisine.value_counts()))*100
            percent = "%.2f" % percent
            percent = str(percent + '%')
            labelpercents.append(percent)

        trace = go.Bar(
                    x=train_data.cuisine.value_counts().values[::-1],
                    y= [i for i in train_data.cuisine.value_counts().index][::-1],
                    text =labelpercents[::-1],  textposition = 'outside',
                    orientation = 'h',marker = dict(color = random_colours(20)))
        layout = go.Layout(title='Number of recipes in each cuisine category',
                           titlefont = dict(size = 25),
                           width=1000, height=450,
                           plot_bgcolor = 'rgba(0,0,0,0)',
                           paper_bgcolor = 'rgba(255, 255, 255, 0.88)',
                           margin=dict(l=75,r=110,b=50,t=60),
                           )
        data = [trace]
        fig = dict(data=data, layout=layout)
        iplot(fig, filename='horizontal-bar')
```

So the italian and the mexican cuisine represents around 36% of the data and the bottom 10 cuisines represents around 18 % of the training data. We can expect from our predictors that when it is uncertain it might predict italian or mexican by "default", since it will be more right to guess there than other cuisines.

1.1.1 Distribution of the Recipe Length

```
In [7]: trace = go.Histogram(
            x= train_data['ingredients'].str.len(),
            xbins=dict(start=0,end=90,size=1),
           marker=dict(color='#7CFDF0'),
            opacity=0.75)
        data = [trace]
        layout = go.Layout(
            title='Distribution of Recipe Length',
            xaxis=dict(title='Number of ingredients'),
            yaxis=dict(title='Count of recipes'),
            bargap=0.1,
            bargroupgap=0.2)

        fig = go.Figure(data=data, layout=layout)
        iplot(fig)
```

So the recipes has a mean around 10 ingridients and there are very few recipes which contains ingridients more than 30 ingridients.

1.2 Abnormal recipes

Some recipes have very short recipes with only 1 or two ingridients.

There are 22 recipes with only one ingridient:

```
In [8]: print("Explore the ingredients in the shortest recipes in our training set:" + "\n")
        print(train_data.loc[train_data.ingredients.str.len() == 1])

        #print(list(train_data[train_data['ingredients'].str.len() == 1].ingredients.values))
        #print("And there corresponding labels" + "\n")
        #print(list(train_data[train_data['ingredients'].str.len() == 1].cuisine.values))
```

```
Explore the ingredients in the shortest recipes in our training set:

           cuisine     id                 ingredients
940       japanese   4734                [sushi rice]
2088    vietnamese   7833        [dried rice noodles]
6787        indian  36818      [plain low-fat yogurt]
7011        indian  19772           [unsalted butter]
8181      japanese  16116                      [udon]
8852          thai  29738               [sticky rice]
8990        indian  41124                    [butter]
10506      mexican  32631            [corn tortillas]
13178         thai  29570                   [grained]
17804  southern_us  29849      [lemonade concentrate]
18136         thai  39186              [jasmine rice]
18324       indian  14335           [unsalted butter]
21008      italian  39221           [cherry tomatoes]
22119       french  41135                    [butter]
22387       indian  36874                [cumin seed]
23512       french  35028            [haricots verts]
26887      mexican  18593             [vegetable oil]
29294      spanish   7460           [spanish chorizo]
30636      spanish  32772  [sweetened condensed milk]
32105     japanese  12805                     [water]
34531        greek  10816                    [phyllo]
37220       indian  27192           [unsalted butter]
```

1.3 Most common ingridients

Lets have a look at the most common ingridients.

```
In [9]: allingredients = [] # this list stores all the ingredients in all recipes (with duplic
        for item in train_data['ingredients']:
            for ingr in item:
                allingredients.append(ingr)

        # Count how many times each ingredient occurs
```

```
countingr = Counter()
for ingr in allingredients:
     countingr[ingr] += 1

# Extract the first 20 most common ingredients in order to vizualize them for better u
mostcommon = countingr.most_common(20)
mostcommoningr = [i[0] for i in mostcommon]
mostcommoningr_count = [i[1] for i in mostcommon]

trace = go.Bar(
            x=mostcommoningr_count[::-1],
            y= mostcommoningr[::-1],
            orientation = 'h',marker = dict(color = random_colours(20),
))
layout = go.Layout(
    xaxis = dict(title= 'Number of occurences in all recipes (training sample)', ),
    yaxis = dict(title='Ingredient',),
    title= '20 Most Common Ingredients', titlefont = dict(size = 20),
    margin=dict(l=150,r=10,b=60,t=60,pad=5),
    width=800, height=500,
)
data = [trace]
fig = go.Figure(data=data, layout=layout)
iplot(fig, filename='horizontal-bar')
```

We can tell that salt is the most common ingridient by far. We can also assume that salt isn't very specific for a certain cuisine, so it will probably not be a good predicator.