

Mieux a Manger

User Manual

# Background Information

For your information, we are using Facebook's new JS framework React for this project. This framework effectively implements a Controlled View. The View is made of components, which have tools (e.g. on click) that interact with the user. These tools will then send logic to update the page. These updates, however, will only affect specific component of the DOM instead of the entire page. This is one of the reasons why React is special.

Setting up React, adding loaders, and building components turned out to be extremely time consuming. Thankfully, after building the JSON packages on our end, it is easy to for users to set up and load the web page. See below for how to set up and load the web page locally.

# Setting Up

Since Mieux a Manger is a web page/app that prompts user authentic restaurant suggestions, we must set up the web page locally in order to load the page.

First clone the Github repository and navigate to the “web” folder in command line. Make sure you have “npm” and admin rights as well (otherwise use “sudo npm”). We need to download all the libraries needed to run the web page. So first we install the relevant commands globally by typing:

npm install webpack --global

npm install webpack-dev-server --global

Then let’s download the package we spent hours on by simply typing:

npm install

We can now use “webpack,” a tool that helps bundle everything together, by typing:

webpack

Now let's load the server and keep it running:

webpack-dev-server

You can now locally load the web page via the prompted local host address. We loaded Bootstrap via cdn so the user no longer needs to set up Bootstrap. This setup will allow us to start using the end product, the web page. To test our python code, however, please refer to Testing the Code.

# Using the Web Page

Yelp only provided data for certain cities, and we chose to only present top restaurants in Las Vegas. We purposely chose 3 cuisine categories to present: “Chinese”, “Indian”, and “Japanese”. The user can use the drop down list to choose the category of interest.



For each category, we present the top 5 restaurants and their corresponding scores. In addition, we build word clouds to show the keywords extracted from reviews using NLP techniques. Simply click on the word cloud picture to see it full screen.

# Testing the Code

Testing our code, however, requires some additional setups. We assume that the TA already has nltk and scikit learn set up. If not, please refer to the following for installation:

<http://www.nltk.org/install.html>

<http://scikit-learn.org/stable/install.html>

In addition, we are using PIL, which is similar to matplotlib. As PIL is a bit out dated, please set it up via Pillow:

pip install Pillow

In addition, in order to generate the word cloud, we used a third party library as suggested by Swap. The library is effectively a single python file named WordCloud.py.

<https://github.com/roshansingh/python-tagcloud>

We built a homebrew tf-idf vectorizer and another using sklearn. We went with the latter as it is considerably faster. Then we built a custom stop word list to improve the results (our data set is too small for tf idf to completely replace stop words). These stop words include common stop words, nltk stop words, common names and etc. At the end, we use the third party file to build word clouds that reflect our tf idf scores.

In order to run the summarizer and generate a word cloud, use the run() function. This function requires at least one argument, the corpus, and the corpus must be a list of string reviews. You can also modify the number of words to display and the destination to store the word cloud jpg file.

For example, the following command will generate the word cloud as shown, which is indeed helpful to customers:

run(["This restaurant is great", "This restaurant is authentic!", "This Chinese restaurant is Tasty.", "This place is authentic!", "This place is tasty."], 50, "test.jpg")



Now putting it all together: in order to generate the web page, simply go to web\_builder.py and run the following:

**if** \_\_name\_\_ == '\_\_main\_\_':  
 build\_json("web/reviews.json")  
 build\_word\_cloud()  
 **print** "done"

The first command builds the JSON for the webpage to display and the the second one generates the word cloud jpg files.