**Recommending locations and categories relative to business requirements.**

**Team Members**:

1. Suyog Swami – 1001119101
2. Vineet Nair – 1001163230

**Motivation and Objective:**

Today setting up a new business or expanding the existing business to new locations is a challenging task. The success of the business depends on many factors like locations proximity from happening places like colleges, offices social gatherings etc. Also popularity (reviews) of the business matters extensively in its success. We try to build a user interactive site where the users can find out best suitable places to their desired businesses. Also we help them find out category of businesses in the user specified location which are more likely to succeed.

The Objectives of our project are as follows:

* To recommend appropriate business categories for a given location requirement :

We are using K-means clustering method to make clusters of Yelp Dataset. We will get centroids and their respective clusters. We are clustering using 5 properties of the Yelp Dataset i.e. Latitude, Longitude, Price Range, Stars, Review Count. User specified location will be converted to their respective geographical coordinates i.e. Longitude and Latitude. The centroids will further be used to compare with user specified Location’s Longitude and Latitude. Hence we will get cluster which is more appropriate with the location provided. Next we will use this cluster to find out groups in the cluster by business categories. Based on the number of records in each group of Business Category we will output the business categories in descending order.

* To recommend locations for a given business category input and suggest best area for the business:

We are using K-means clustering method to make clusters of Yelp Dataset. We are taking two inputs from user i.e. business category and its price range. We will get centroids and their respective clusters. We are clustering using 5 properties of the Yelp Dataset i.e. Latitude, Longitude, Price Range, Stars, Review Count. We are using the Price Range provided by the user to select appropriate cluster. The cluster will be further used to compare with user specified Business Category. Next we will use this cluster to find out groups in the cluster by Locations. For each group we will find out best area i.e. city name with the highest frequency. Based on the number of records in each group we will output the locations in descending order which will be the appropriate location for given business category.

**Data Mining Tasks:**

We are using clustering in this project as a major data mining task. K-means clustering method is used for the purpose of making clusters of Yelp Dataset. We have chosen 5 numeric attributes that effectively define the Yelp Dataset as well as are appropriate for achieving our objectives in this project. The five attributes chosen are Latitude, Longitude, Price Range, Stars and Review Count. We chose Latitude and Longitude because it helps to cluster the data by the location. And the conversion from area (categorical attribute) to its respective geographical coordinates (numeric attribute) helps us to give more appropriate location. The price range helps us to segregate the businesses by their service costs. The remaining two attributes i.e. Stars and Review Count helps us gauge the popularity of the business category.

We are using ‘k’ as 3 in this k-means clustering algorithm. We used this value because points in the cluster overlapping was low and the performance time using k as 3 compared to other values of k was also better.

**Design of Methods:**

The method where we calculate the K-means, we first assign K centroids which are points chosen at random from the dataset. Next we compare all the remaining instances of the

Dataset with the chosen centroid and assign these points to the cluster associated with the centroid with which they are most similar. After this we check in each cluster if there is a point which is more suitable to become centroid. If there is any such point then it is assigned as new centroid of that cluster. We again repeat the process of comparing the remaining instances with the new centroids. If the clusters do not change for the given centroids then we stop the process and we get the final set of clusters. However, if the clusters have changed then the given process is repeated until it remains across the iterations. We implement separate methods for calculating the centroids for clusters across the iterations. We have also used separate function to get Euclidean distances which is used to calculate the similarity between the instance of the dataset and the centroid of the cluster. We have stored the clusters and its centroids as the instances of the list.

We have created method to get the grouping of the clusters according to the given user specification i.e. location/business category. Then we output the top five results of the given query based on their frequency in the groups. In this, if we are given a location then we get the clusters by the location’s geographical coordinates and then we group the clusters by each unique business category and output the top 5 business category with highest frequency in the chosen cluster. Similarly, if we are given the business category and its price range then we get the cluster based on the price range. Then this cluster will be segregated based on the location and we get the top 5 location as the output with highest frequency in the chosen cluster.

**Implementation of methods:**

The source code is divided into a) Generation of clusters by K-means [1] b) Finding appropriate Location based on the cluster for given business category. c) Finding appropriate Business Category based on the cluster for given location.

* Generation of clusters by K-means[1] :
  1. Based on random k-points selected from the data set k-centroids are generated.
  2. Next step we assign points to each cluster containing the generated centroid. For the purpose of assigning points to each cluster we calculate Euclidean distance between centroid and each instance. The instance is assigned to the cluster which is most similar with it i.e. least Euclidean distance.
  3. Then we recalculate the centroid of each cluster, if the new centroid is different than the previous centroid then we again repeat step 2).
  4. Step 3) is repeated till new centroid is not equal to previous centroid.
* Finding appropriate Location based on the cluster for given business category:
  1. From the output of k-means clustering we obtain k-clusters from which we choose appropriate cluster. This is done by finding the cluster with centroid which is similar to the user provided price range. This is done by calculating the Euclidean distance between the users provided price range and centroids price range.
  2. Now, we group the cluster with respect to its location (city).
  3. We then sort the data in descending order to get the top 5 locations (cities) based on their number of occurrences.
  4. We have used following python libraries:

Pandas

Itemgetter from operator

Counter from collections

* Finding appropriate Business category based on the cluster for given location:
  1. From the output of k-means clustering we obtain k-clusters from which we choose appropriate cluster. This is done by finding the cluster with centroid which is similar to the user provided location. The location is converted to its geographical coordinated i.e. Latitude and Longitude. Finding the appropriate cluster is done by calculating the Euclidean distance between the users specified Latitude, Longitude and centroids Latitude, Longitude.
  2. Now, we group the cluster with respect to its Business categories.
  3. We then sort the data in descending order to get the top 5 business categories based on their number of occurrences.
  4. We have used following python libraries:

Pandas

Itemgetter from operator

Counter from collections

GoogleV3 from geopy.geocoders

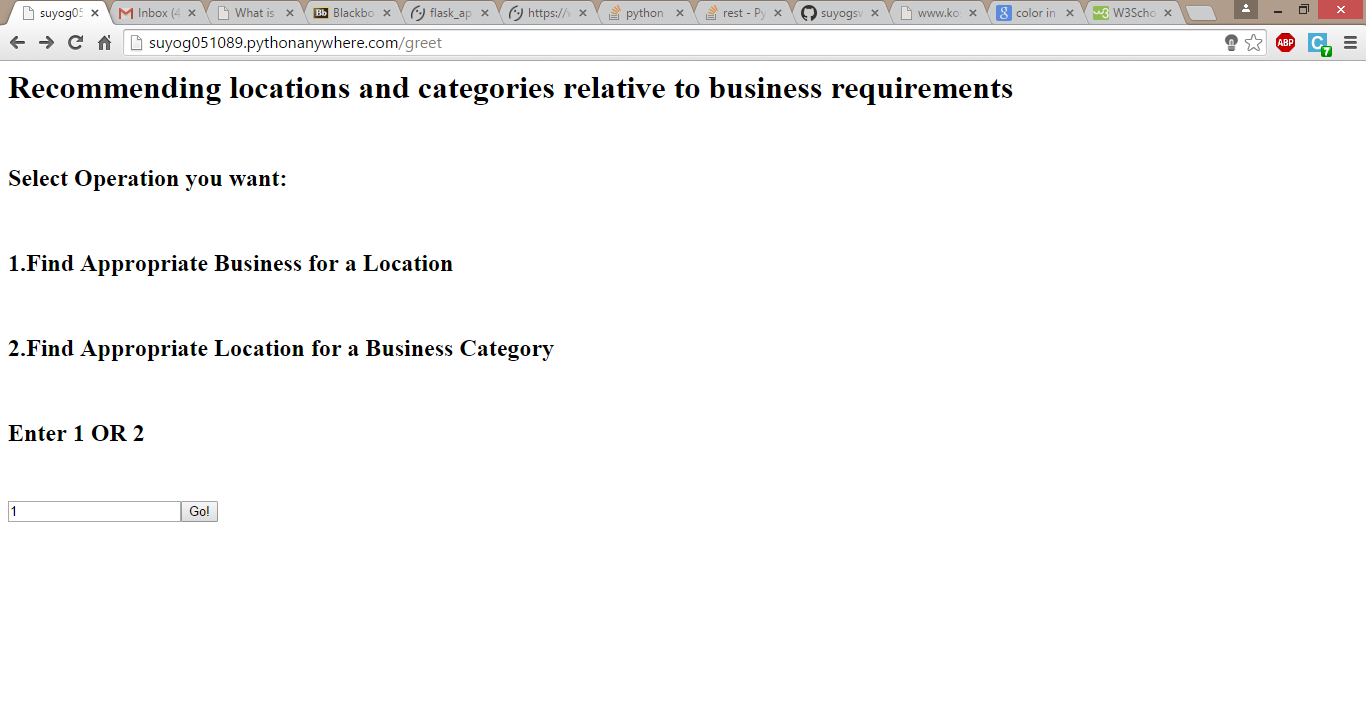
**Results and Evaluation:**

We have hosted website for our final project on pythonanywhere.com. You can view the website by following the link: <http://suyog051089.pythonanywhere.com>.

* The first page shows options to the user to choose between the two tasks of our project.

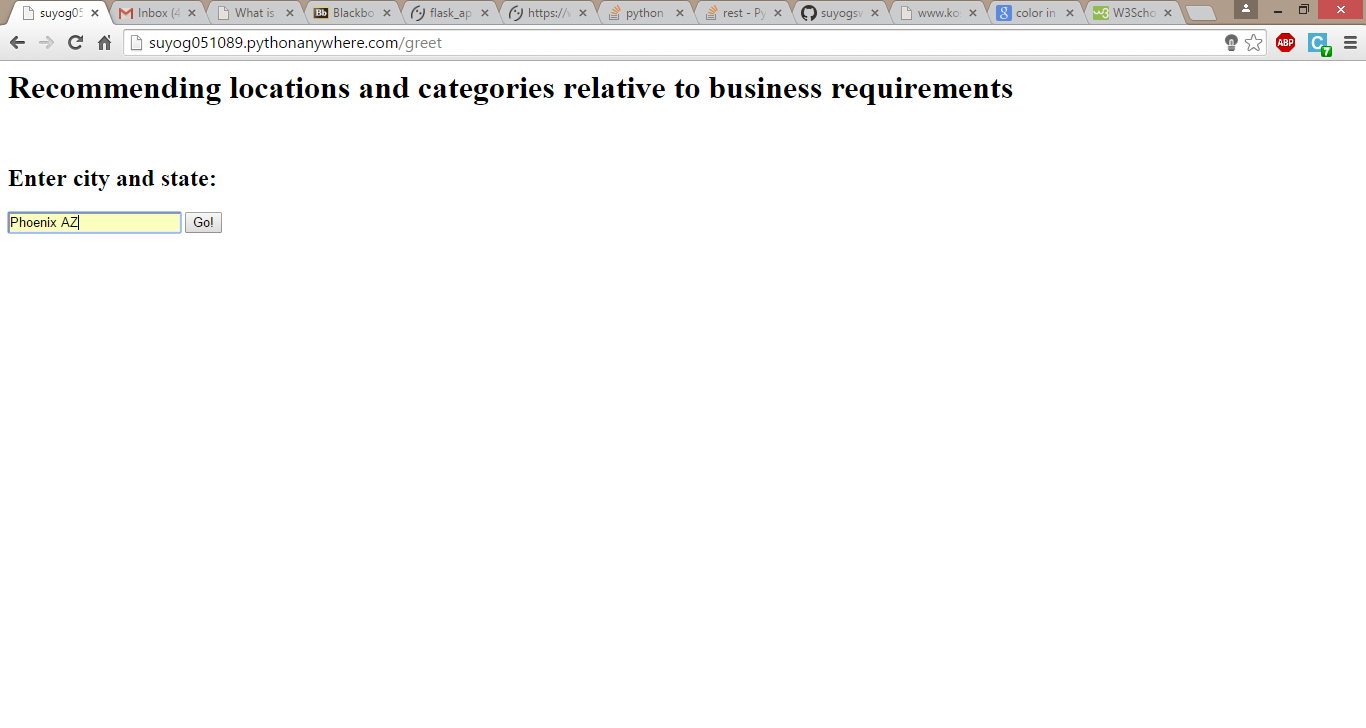
1. Finding appropriate Location based on the cluster for given business category.
2. Finding appropriate Business Category based on the cluster for given location.

Also an input field to accept users options and a “Go!” button to submit the query.

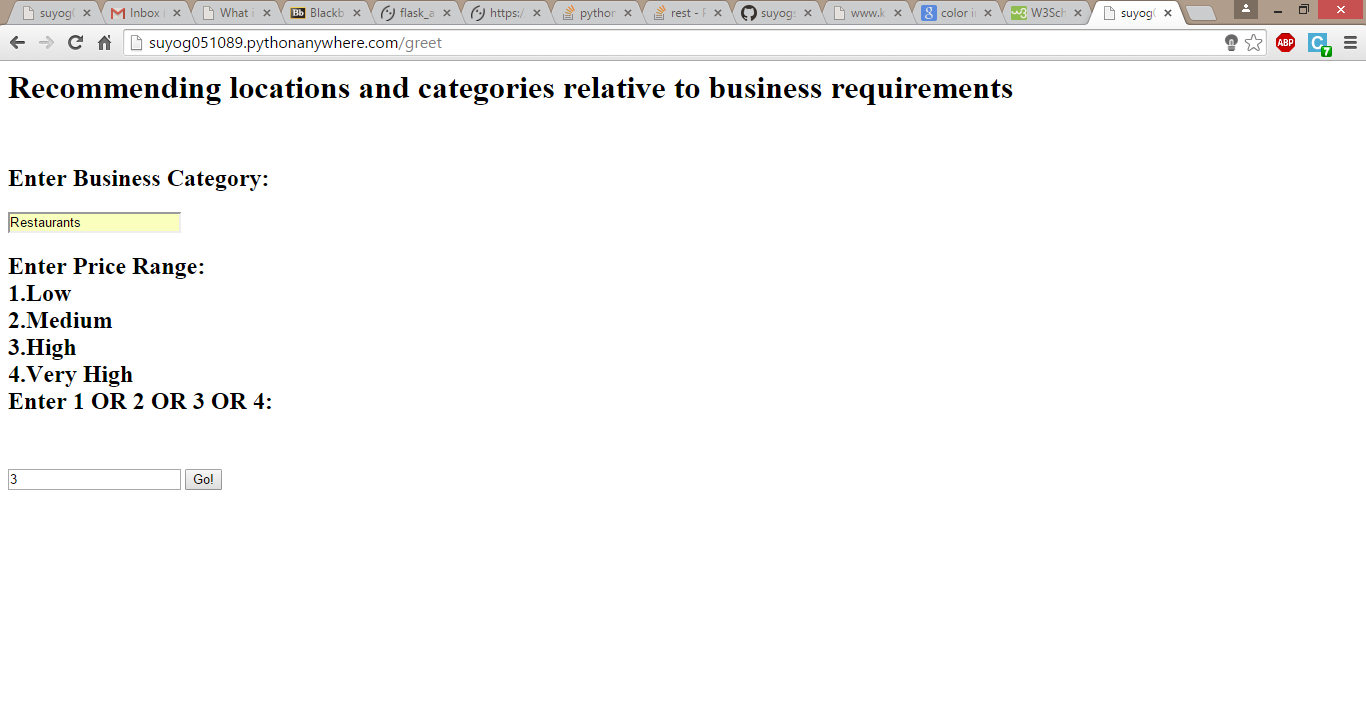


* Second Page depends on the users input. If user has chosen:

1: He has to enter the city and state name

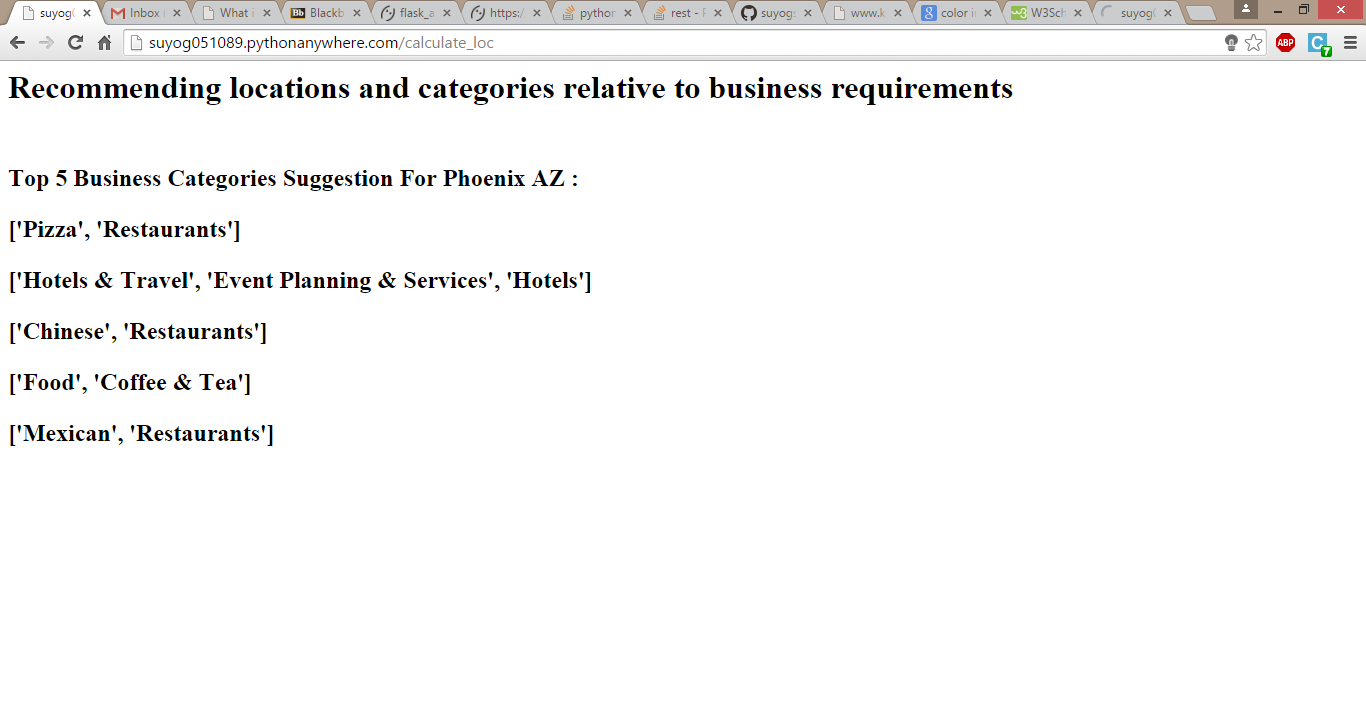


2: He has to enter the Business category and its expected price range.

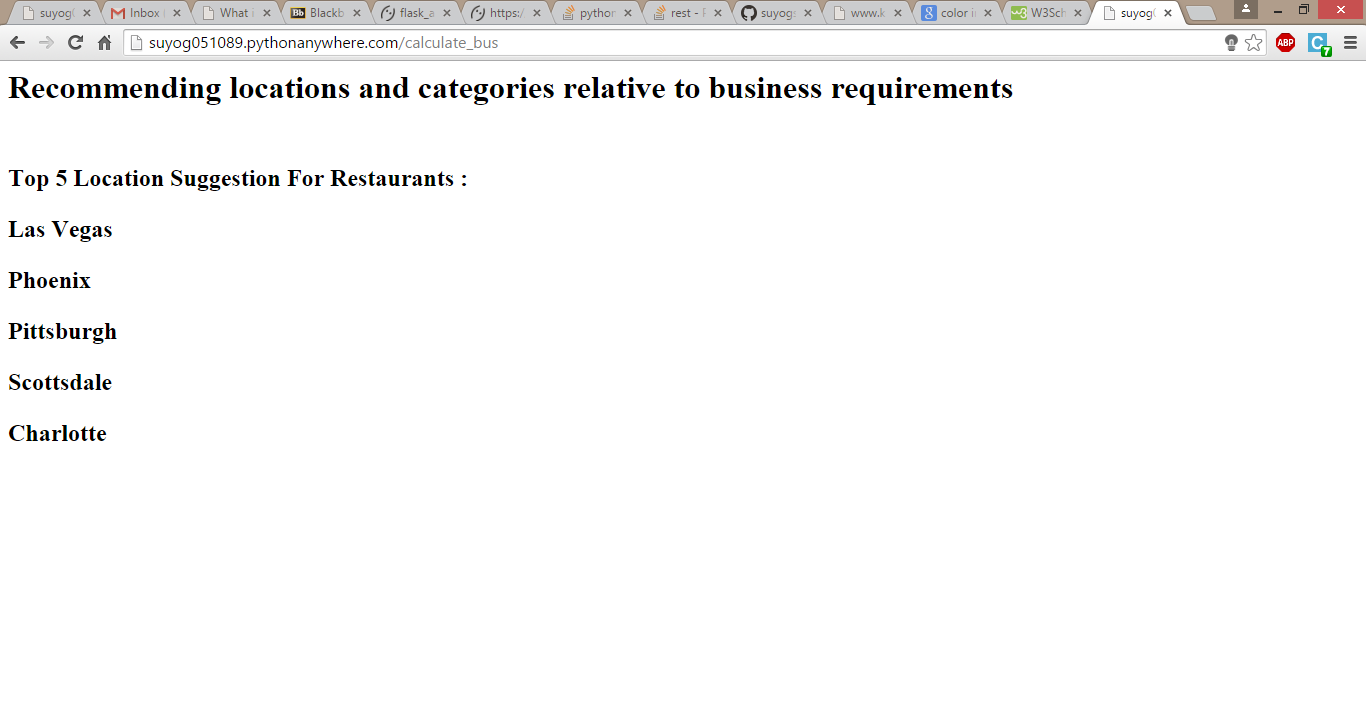


* Third pages gives user the output depending upon the chosen task i.e. Top 5 appropriate business category for user specified location or Top 5 appropriate location for user specified business category.

Result for “Top 5 Business Categories for Phoenix AZ”



Result for “Top 5 Locations for Restaurants”



Evaluation:

We tried for the areas which are known for being tourist locations E.g. Las Vegas, New York, etc. and we found out that similar suggestions of hotels, restaurants (pizza, Chinese, Italian) was given to the user.

For Second task we found out that business category such as bars, pubs and nightlife which are famous in locations such as Las Vegas etc. and the result also showed similar areas.

**Presentation and Visualization:**

We have used flask framework on the pythonanywhere.com. The website has 5 pages in total. First page for choosing the task, 2 pages for accepting respective task inputs and 2 pages for respective task outputs.

We had also made use of tkinter python package to display the cluster formation. But as the requirement of this project was to host a website, tkinter is not supported on web app. Hence we had to remove the cluster formation visualization.

**URL’s:**

<http://suyog051089.pythonanywhere.com>(Website)

<https://github.com/suyogswami/6339project> (Repository)

**References:**

1. http://www.kosbie.net/cmu/fall-10/15-110/handouts/notes-clustering/kmeans.py