Industrial Distribution Analyzer and Recommender

Lei Yang, Puxuan Qi, Shuhan Liu, Youcun Liu

Abstract

This paper discussed how to use different recommendation algorithms (Decision Tree, SVM, Softmax Regression, KNN & K-Means++, etc) to recommend companies for job seekers. Furthermore, we analyze the accuracy of each recommendation system on our dataset, and explain the reasons. In addition, we discussed how to use K-Means clusters to analyze the industrial distribution to provide the overview of leading Industrial Distribution in Massachusetts.

Section 1: Introduction

There are two parts of our program, one is the analysis of leading industry, the other is a recommender system for job seeker.

Our recommender aims at providing job seekers a clear overview of leading industrial distribution in Massachusetts, and recommending companies for job seeker basing on user's past behavior and basing on content respectively.

Typically, a recommender system produce a list of recommendation in two ways. One is building a model from a user's past behavior (in our project, we would use users' frequency of clicking different job information) as well as similar decisions made by other users. This model is then used to predict items (in our project, means companies) that the user may have an interest in. The other is utilizing a series of discrete characteristics of an item in order to recommend additional items with similar properties.

For the industrial distribution part, we first use TF-IDF vectorize the input features (city, job title). Adjust the weight of location and job title to 40% and 60%. Use K-Means++ to cluster the data. Then Plot the industrial distribution in Massachusetts.

Our dataset is crawled from indeed.com and divided into two parts. One contains jobs and their features. The other contains rows of user Id, job Id and frequency of the user clicking the job, in short How much the user is interested in that company.

Our project would combine those two approach and do some revision and comparison between different algorithm(i.e. Softmax regression and SVM, decision tree and KNN). The detail is: for old users which means we have the record of their clicking information, we would use SVM and Softmax regression and base on user's past behavior to do recommendation. And for newbies, we would use decision tree and KNN and base on user's requirement to do recommendation.

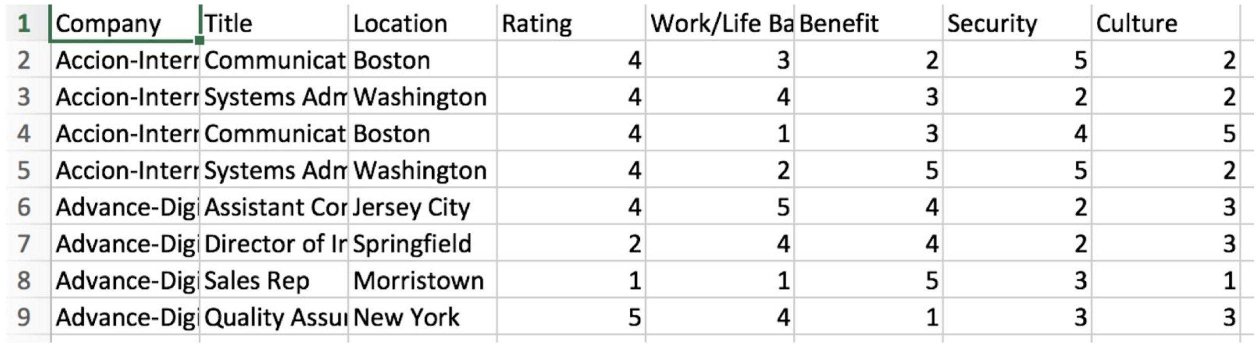
Finally, we would compare accuracies of those algorithms and analysis the advantages and disadvantage. Then we would offer some future work about our recommendation system.

In section 2, we will talk about the data set and the data preprocess
In section 3, we will take about the industrial distribution in Massachusetts
In section 4, the topic is about using Softmax regression and SVM algorithm and basing on user's past behavior to recommend company for job seeker.
In section 5,6 the topic covers that for newbie to the job, which we do not have any past behavior record, we would recommend basing on the content provided by job seeker.

In section 7, The conclusion and future work on our recommendation system will be discussed.

Section 2. Dataset

Our data set comes from Indeed.com. We implemented a Python web scraper and analyzed over two hundred company reviews from over ninety thousand users. Here is a glance of the data set.



1	Company	Title	Location	Rating	Work/Life Ba	Benefit	Security	Culture
2	Accion-Interr	Communicat	Boston	4	3	2	5	2
3	Accion-Interr	Systems Adm	Washington	4	4	3	2	2
4	Accion-Interr	Communicat	Boston	4	1	3	4	5
5	Accion-Interr	Systems Adm	Washington	4	2	5	5	2
6	Advance-Digi	Assistant Cor	Jersey City	4	5	4	2	3
7	Advance-Digi	Director of Ir	Springfield	2	4	4	2	3
8	Advance-Digi	Sales Rep	Morristown	1	1	5	3	1
9	Advance-Digi	Quality Assu	New York	5	4	1	3	3

It has eight columns and over ninety thousand rows. Each column represents one feature, and each row represents one user review details.

Specifically, the company, title and location columns are used as the feature of K-Means algorithm for Industrial distribution analyze in Massachusetts.

The company, rating, work/life balance, benefit, security and culture are used as the feature for the recommendation systems.

All the ratings are based on a 1 – 5 scale. 1 is lowest and 5 is highest.
Rating: The overall rating in all aspects of the company
Work/Life Balance: The balance of work time and leisure time
Benefit: The overall salary and compensation
Security: The safety of the work
Culture: The company culture

Section 3. Industrial Distribution Analyze in Massachusetts

*Assumption: we select 7 cities in Massachusetts and 1 area *Greater Boston Area" which has 6 extra cities.*
Greater Boston Area: Boston, Providence, Lowell, Cambridge, Quincy, Newton
Cities in Massachusetts: Greater Boston Area, Salem, Plymouth, Waltham, Framingham, Worcester. Lexington, Danvers

Algorithm:

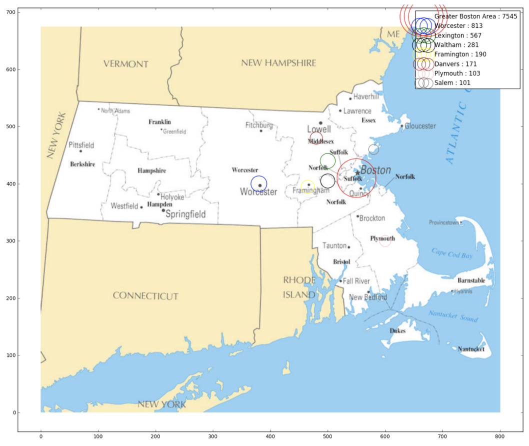
First we applied Tfidf vectorizer to vectorize the string features.

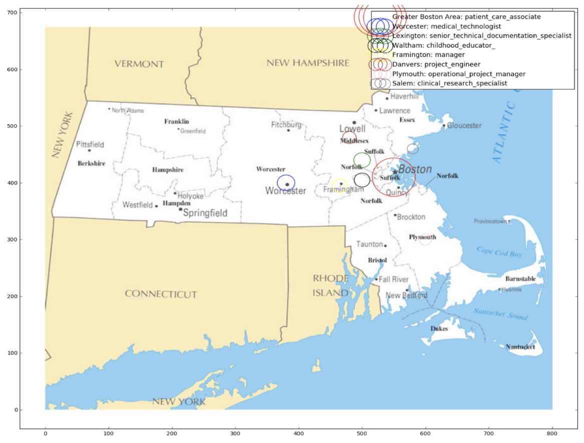
Adjust the weight of features, let location for 40% weight, and the rest 60% weight
Applied K-Means algorithm shared by job titles. The algorithm allows us to get the leading industries in each location.
Let c_1 be an entry in the document, Let X be the document
Choose an initial center uniformly at random from X. Compute the vector containing the Euclidean distances between all points in the dataset and C_1: $D_i^2 = ||\mathrm{x}_i - c_1||^2$
Choose a second center from X randomly drawn from the probability distribution $D_i^2 / \sum_j D_j^2$
Re-compute the distance vector as $D_i^2 = \min\left(||\mathrm{x}_i - c_1||^2, ||\mathrm{x}_i - c_2||^2\right)$
Pick a successive center and recompute by the distance $D_i^2 = \min\left(||\mathrm{x}_i - c_1||^2, \ldots, ||\mathrm{x}_i - c_l||^2\right)$

A glance of the K-Means result
Cluster 0: patient_care_associate, greater_boston_area, critical_care_technician…
Cluster 1: Worcester, medical_technologist, administrative_assistant, executive_administrative_assistant
Cluster 2: Lexington, security_guard, senior_technical_documentation_specialist

Result1: Visualize the leading industry in Massachusetts of 8 places

Here we can see the leading industries of each location in Massachusetts, for example, in Greater Boston Area, the leading industry is patient care associate. The size of circle represents how many jobs this area has. In the above map we can see the circle of Greater Boston Area is largest.

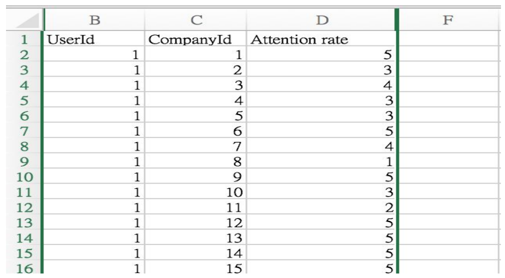



Greater Boston Area: patient_care_associate
Worcester: medical_technologist
Lexington: senior_technical_documentation_specialist
Waltham: childhood_educator_
Framington: manager
Danvers: project_engineer
Plymouth: operational_project_manager
Salem: clinical_research_specialist
VERMONT
NEW HAMPSHIRE
NEW YORK
CONNECTICUT
RHODE ISLAND
ATLANTIC
Boston
Worcester
Springfield

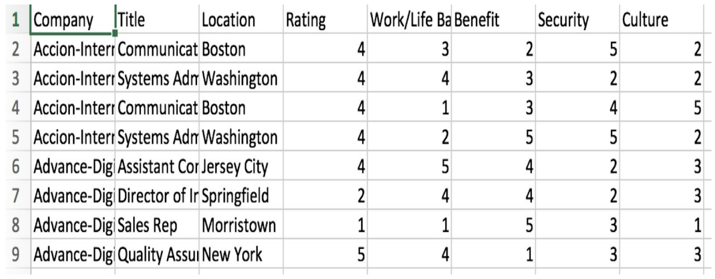
Visualize the leading industries in Greater Boston Area
Here we can see in Greater Boston Area, there are over 7,000 jobs, almost ten times over the second Worcester (813). So we also use count vectorize to get the leading industries in Greater Boston Area.

Section 4: Recommend company basing on user's past behavior

1. Data Set
In our complete data set, we have two data sheets. One records the company's feature and the other contains the userId, companyId and rate of the frequency of the user clicking the company's information, in short How much the user was interested in that company, e.g 5 means very interested in and focused more on the company, 1 means less interest and paid less attention to it. In addition, the rate is not a continuous value, thus we treat the rate as a label and then the problem becomes a classification problem.



	B	C	D	F
1	UserId	CompanyId	Attention rate	
2	1	1	5	
3	1	2	3	
4	1	3	4	
5	1	4	3	
6	1	5	3	
7	1	6	5	
8	1	7	4	
9	1	8	1	
10	1	9	5	
11	1	10	3	
12	1	11	2	
13	1	12	5	
14	1	13	5	
15	1	14	5	
16	1	15	5	



1	Company	Title	Location	Rating	Work/Life Ba	Benefit	Security	Culture
2	Accion-Interr	Communicat	Boston	4	3	2	5	2
3	Accion-Interr	Systems Adm	Washington	4	4	3	2	2
4	Accion-Interr	Communicat	Boston	4	1	3	4	5
5	Accion-Interr	Systems Adm	Washington	4	2	5	5	2
6	Advance-Digi	Assistant Cor	Jersey City	4	5	4	2	3
7	Advance-Digi	Director of Ir	Springfield	2	4	4	2	3
8	Advance-Digi	Sales Rep	Morristown	1	1	5	3	1
9	Advance-Digi	Quality Assu	New York	5	4	1	3	3

2. General Approach

we would train a model for each user who appears in upper sheet, then by using that model, we can predict a label for the companies of testing data and recommend the company with label 5 to the user. And we use two algorithms, i.e. SoftMax regression and SVM to implement our idea.

3. Softmax regression

Softmax regression is a general case of logistic regression, it can do multi-classification problem. We can get a theta matrix for every user, in other words the weights, then use the matrix to do prediction.

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1|x^{(i)};\theta) \\ p(y^{(i)} = 2|x^{(i)};\theta) \\ \vdots \\ p(y^{(i)} = k|x^{(i)};\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix} \qquad \theta = \begin{bmatrix} -\theta_1^T- \\ -\theta_2^T- \\ \vdots \\ -\theta_k^T- \end{bmatrix}$$

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$

(a)Initial parameters and input data

The size of input vector is 1*5, which means every row of our data has 5 features and the number classes is 5, i.e. 1, 2, 3, 4, 5, five kinds of different rate. Thus the size of our goal, the theta matrix or the weights is 5*5=25. In order to avoid the problem of local optimal solution, we initialize the theta matrix as a random number matrix.

(b) Compute the cost

- We would compute the ground truth matrix for input data, which means sparse function of each sample. Then we use sparse() and full() in matlab to convert sparse matrix to one hot encoding.
- Computing the gradient function and cost function

$$\frac{\partial J(\theta)}{\partial \theta_j} = -\frac{1}{m}\left[\sum_{i=1}^{m} x^{(i)} \cdot (1\{y^{(i)} = j\} - P(y^{(i)} = j|x^{(i)}\ ;\theta))\right] + \lambda\theta_j$$
$$= -\frac{1}{m}\left[\sum_{i=1}^{m} x^{(i)} \cdot (1\{y^{(i)} = j\} - \frac{e^{(\theta_j)^T \cdot x^{(i)}}}{\sum_{l=1}^{k} e^{(\theta_l)^T \cdot x^{(i)}}}\right] + \lambda\theta_j$$

(c) Matlab Automatic gradient optimization

The normal way is utilizing current weights minus product of step length and gradient, i.e. $\theta j = \theta i - \alpha \nabla \theta i$, but we would choosing another quick solution, using MATLAB's optimal function minFunc and LBFGS algorithm.

4. 1 vs rest SVM

1 vs rest SVM calculate an optimal hyper-plane to assign label. we would train 5 svm for each user and use RBF as kernel function, because the number of samples is much more than the number of features.

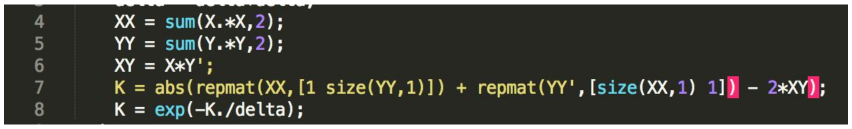
(a) Labels preprocess

For example, if we want to train a SVM of label 2 vs rest, we have to create a new label vector where label 2 would be converted to 1 and the others would be converted to -1.

(b) using quadprog() to solve the Lagrange function with constraints

$$\begin{cases} \max_{\alpha_i} \left\{ \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \left(x_i \cdot x_j \right) \right\} \\ \quad s.t. \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \\ \quad\quad\quad \alpha_i \geq 0 \end{cases}$$

(c) RBF - Kernel



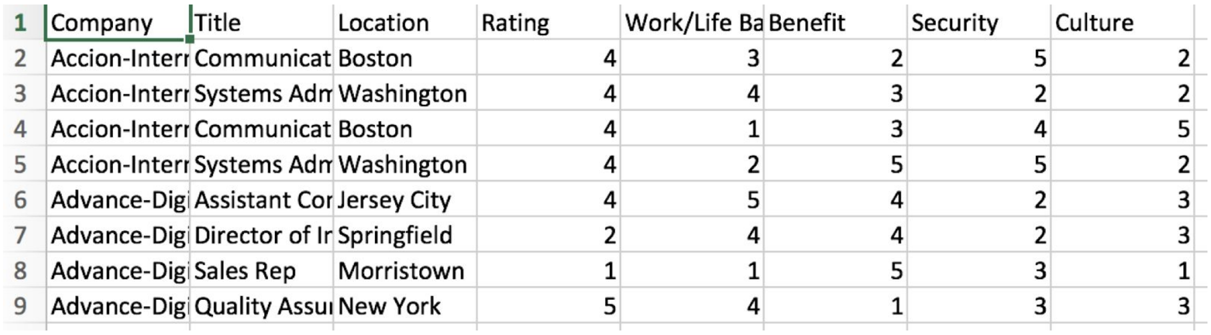
```
XX = sum(X.*X,2);
YY = sum(Y.*Y,2);
XY = X*Y';
K = abs(repmat(XX,[1 size(YY,1)]) + repmat(YY',[size(XX,1) 1]) - 2*XY);
K = exp(-K./delta);
```

5. Conclusion

Finally, we find that regression's performance is better than svm, although the accuracy of those two is similar(i.e. 85.46% and 85.43%). Because the number of feature in our data set is really small and the kernel function may be not the most suitable for our problem, also we cannot ignore the time-consuming. By contrast, Softmax regression is simple and fast. The most important issue is that it can return the classification possibility for our model such that we can do recommendation more precise, rather than giving a meaningless number, it can tell us the possibility of the user may interested in that company. In addition, regression algorithm is better for the data with less feature.

Section 5. Recommending companies to new users by using Kmeans and KNN.

The second part of our project focuses on how to recommend companies to new users that are most suitable for them. In order to get the most suitable one, a user's expectations, or in other word, their data is needed. However, the original dataset doesn't have this piece of information, therefore, our project needs users to give their expected rating for each attribute first, for example, work/life balance, benefit, security and culture.



1	Company	Title	Location	Rating	Work/Life Ba	Benefit	Security	Culture
2	Accion-Interr	Communicat	Boston	4	3	2	5	2
3	Accion-Interr	Systems Adm	Washington	4	4	3	2	2
4	Accion-Interr	Communicat	Boston	4	1	3	4	5
5	Accion-Interr	Systems Adm	Washington	4	2	5	5	2
6	Advance-Digi	Assistant Cor	Jersey City	4	5	4	2	3
7	Advance-Digi	Director of Ir	Springfield	2	4	4	2	3
8	Advance-Digi	Sales Rep	Morristown	1	1	5	3	1
9	Advance-Digi	Quality Assu	New York	5	4	1	3	3

Once the user gives his/her expected ratings, this program will not jump directly into the process of comparing user's data to each record to find the best matching, since it is not efficient and may have many outliers that will affect the accuracy of this recommendation method. Therefore, in order to make it efficient and accurate, a clustering method as the first step is needed.

In order to cluster properly, we need to choose the correct clustering method. Before start, there are two commonly used clustering method to choose, the K-means algorithm and hierarchical clustering. Comparing these two techniques, K-means algorithm works more efficiently when the dataset is large. As long as the dataset we have contains job reviews all over the country, choosing K-means can get the result in a shorter time. Another advantage is that K-means can produce tighter clusters than hierarchical, which is very important to us. The score for each attribute is ranging from one to five, which is a relatively small range, thus, we need clusters to be tight enough to get more accurate results.

However, K-means algorithm also has its drawbacks. The most obvious one is that it is hard to decide the "K", which is the number of clusters, and this method produces exactly *k* different clusters of greatest possible distinction. The best number of clusters k leading to the greatest separation (distance) is not known as a priori and must be computed from the data. To solve this problem, we take use of the squared

$$\text{objective function} \leftarrow J = \sum_{j=1}^{\overset{\text{number of clusters}}{k}} \sum_{i=1}^{\overset{\text{number of cases}}{n}} \underbrace{\left\| \overset{\text{case } i}{x_i^{(j)}} - \overset{\text{centroid for cluster } j}{c_j} \right\|^2}_{\text{Distance function}}$$

error function of K-means.
Therefore, we can use different possible k values to do the K-means clustering, and calculate the squared error for each time. By plotting these values out, it can help us to decide the K value. After implementing the clustering algorithm, the dataset is divided into different clusters. Each cluster has a distinct label and a centroid. Since each row of the dataset is a past user's rating to a specific company, which can also be considered as expectations or requirements of a user, therefore, the new user's expected rating for each attribute can be assigned to one of the clusters. To accomplish this, this program computes the new user's data with all cluster's centroids and compare the Euclidean distance of them. After that, the cluster with the smallest result will be considered as companies with same kind of properties that the user expected, and this user will be assigned to this cluster.

Then, inside the cluster, are the ratings to companies that has characteristics that matches the new user's requirements. One thing needs to be notified is that each record is a rating to a company, so that there will probably exist many records pointing to duplicate company names. This is reasonable because each company has many jobs and each job may have several employers. We can use this important characteristic to implement a k nearest neighbour algorithm. The KNN algorithm is one of those algorithms that are very simple to understand but works incredibly well in practice. It is a non parametric algorithm, which means it does not make any assumptions on the underlying data distribution. This is can be useful in this project since we don't really define what this new user needs, we just use data to find the similar points objectively. This can make the results more precise and accurate. Therefore, we can consider the rating of the user as a point in a K-dimensional space, then the distance between these two points in Euclidean distance can be calculated by the sum of difference between each two values of the same attribute squared. Calculating the distance here is much faster than comparing the new user's data with all other records, since you only have to use the records inside the cluster. As mentioned above, there will be duplicates in company name, and since all companies in this cluster has similar characters as the user expected, so the company name repeats the most means it is the best match for the user, therefore recommending this company should be suitable for the user. If the user wants more, we can recommend the second most repeated and so on.

The result of this experiment is very straightforward, it gives the company names that repeats the most after implementing both Kmeans and KNN algorithms with an accuracy of 83.5%. However, this algorithm is not perfect, one drawback of this method is that, the KNN is a black-box algorithm. Although it gives you results to keep points of the same characteristic together, it does not provide any instructions about what the rules it used to classify target points. This might be an disadvantage of this recommendation system since user will probably want to have an idea why this program comes with this result and what is the main feature of this kind of companies. Therefore, this project has a second approach to do that, that is using the decision tree.

Section 6. Decision tree

Decision trees are simple to understand and interpret. Besides, we could use it for both classify and regress. It is a white box model which means every step is interpretable. But decision tree also some disadvantages, for example, the multi class prediction will takes a lot of time. At worst, memory overflow may occur. The proportion of data items and features is also vital important, too. If data items overwhelm features, underfitting will happen. We firstly try logistic regression and it really performs bad, the accuracy is desperately low and training is about 6000 sec. Therefore, we try to use multi decision tree to solve this problem.

1. Data processing:

The total number of items of our dataset is 95842, the number of features is 6 and the number of classes of our label is about 248.
Before training, we need to do some job with the dataset.

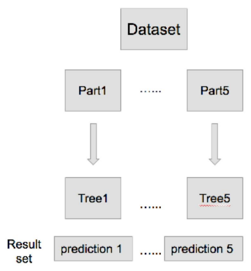
1.1 impute empty data cell:
Because we only use the non-empty feature, so Nan cells only appear in label region.
Given the consideration that the type of label is non-value, there is no means to calculate an average value and fill in by using it. Besides, if we use the label appears the most time, it is highly possible that we create an unbalanced dataset. Therefore, we have two other choices, the first is choose a company randomly and impute the Nan field. The second one is to create a new class that contains all Nan value.
In our project, we choose the latter one.

1.2 Changing the string type to numeric type:
Company is represented in natural language, only changing it into numerical value we can load it into our model. So we use LabelEncoder function in sklearn library to change it into numerical type.

1.3 Shuffle
Because we were using multi decision tree, without shuffling the dataset unbalanced will happened inside each tree and finally lead to bad performance.

2. Multi decision tree

Since using single decision tree will lead to memory overflow in predicting steps, we try to split the dataset into 5 different part. Therefore, each part contains 12,000 datas. Then we and use each part to train a decision tree. Then we use the five decision trees to give user a group of predictions.
We show the procedure in left figure and the prediction result looks like right:



```
real: [226 206 182 ..., 113 114 223]
[226 206  96 ..., 213 118  59]
real: [226 206 182 ..., 113 114 223]
[226 166 130 ...,  37   2 136]
real: [226 206 182 ..., 113 114 223]
[227 206 137 ...,  37  37  59]
real: [226 206 182 ..., 113 114 223]
[227 206 228 ...,  96 140  59]
real: [226 206 182 ..., 113 114 223]
[227 237 182 ...,  37 115 223]
```

Section 7. Conclusion

For industrial distribution analyze in Massachusetts, in Greater Boston Area, the leading industry is patient care associate. In Greater Boston Area, there are over 7,000 jobs, almost ten times over the second Worcester (813). For a more detailed analyze report, please refer the industrial distribution map in Section 3. It shows the leading industrial in each area of Massachusetts.

For recommending companies for old user. Softmax regression performance is better than svm, although the accuracy of those two is similar(i.e. 85.46% and 85.43%). Because the number of feature in our data set is small and the kernel function may be not the most suitable for our problem, also we cannot ignore the time-consuming. By contrast, Softmax regression is simple and fast. The most important issue is that it can return the classification possibility for our model such that we can do recommendation more precise, rather than giving a meaningless number, it can tell us the possibility of the user may interested in that company. In addition, regression algorithm is better for the data with less feature.

For recommending companies to new users, the two approaches we tried have their advantages and drawbacks. For the Kmeans and KNN approach, it has a higher accuracy, and is more efficient when implementing, but it cannot express the classification rule in an explicit way. Therefore the user might have many concerns. While decision trees' classification rule is more clear and the project can provide the user with what kind of characteristics the recommended companies have. Moreover, decision tree is sensitive with the proportion of data items and features. So, if we cannot extract more feature, it would be reasonable that split the dataset into different part and merge the result.