Computer Science
Spring 2023
CSCI-6401 (DataMining)
Phase 6–Data Modeling

GOOGLE MAPS

Submitted by:
Team Amigos

Submitted to:
Dr. Shivanjali Khare

Team Members:
Member 1: Jayanth Reddy Manda (00764288) jmand5@unh.newhaven.edu
Member 2: Snigdha Reddy Yeruva (00762316) syeru2@unh.newhaven.edu
Member 3: Jeevan Kumar Konduru (00806735) Jkond6@unh.newhaven.edu

GitHub link: https://github.com/JayanthReddy4/Phase-6-AMIGOS.git

Research question:
Recommendation to the customer in a path from source to destination (Ex: let us Assume google maps knows user is from India, assume user is travelling from "Newhaven railway station " to "Hartford Railway Station" via car. If the user needs coffee, we recommend best coffee available stores near user based on business data available in google maps such as rating, price levels, user rating, etc....)

Dataset:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	lat	lng	name	vicinity	Type1	Type2	Type3	Type4	Type5	Type6	rating	user_ratin	price_level
2	41.2705	-72.947	West Have	West Have	accounting	finance	point_of_i	establishm	NA	NA	0	0	0
3	41.2331	-73.026	Hilton Gar	291 Old G	accounting	finance	local_gove	point_of_i	establishm	NA	4	813	0
4	41.2312	-73.0299	Hyatt Plac	190 Old G	accounting	local_gove	finance	point_of_i	establishm	NA	4	664	0
5	41.2558	-73.0018	Courtyard	136 Marsh	accounting	finance	point_of_i	establishm	NA	NA	4	499	0
6	41.2232	-73.0771	Hampton I	129 Plains	accounting	finance	point_of_i	establishm	NA	NA	3.8	824	0
7	41.2315	-73.0463	Super 8 by	1015 Bost	accounting	finance	local_gove	point_of_i	establishm	NA	2.7	294	0
8	41.2357	-73.0356	Zumiez	1201 Bost	accounting	finance	point_of_i	establishm	NA	NA	4.4	19	2
9	41.236	-73.0355	Connecticu	1201 Bost	accounting	finance	point_of_i	establishm	NA	NA	4.3	6999	0
10	41.2352	-73.0372	AT&T Stor	1201 Bost	accounting	local_gove	finance	point_of_i	establishm	NA	4.8	799	2
11	41.2348	-73.0372	ALDO	1201 Bost	accounting	finance	point_of_i	establishm	NA	NA	3.7	47	2
12	41.2348	-73.037	Hollister C	1201 Bost	airport	point_of_i	establishm	NA	NA	NA	4.3	111	2
13	41.2351	-73.0378	Buffalo Wi	1201 Bost	airport	point_of_i	establishm	NA	NA	NA	4	1341	2
14	41.2356	-73.0366	Express	1201 Bost	airport	point_of_i	establishm	NA	NA	NA	4.1	88	2
15	41.2358	-73.0361	Torrid	1201 Bost	airport	point_of_i	establishm	NA	NA	NA	4.3	92	2
16	41.237	-73.0343	Diva Kidz	1201 Bost	airport	point_of_i	establishm	NA	NA	NA	2.9	33	0
17	41.2356	-73.0356	Undergrou	1201 Bost	amusemer	tourist_att	point_of_i	establishm	NA	NA	4.3	22	2
18	41.2509	-73.0239	Costco Pha	1718 Bost	amusemer	tourist_att	point_of_i	establishm	NA	NA	3.2	5	2
19	41.236	-73.0357	Pretzelmal	1201 Bost	aquarium	tourist_att	point_of_i	establishm	NA	NA	3.7	9	1
20	41.2513	-73.0178	Trader Joe	560 Bosto	aquarium	tourist_att	point_of_i	establishm	NA	NA	4.6	2243	2
21	41.2307	-73.064	Milford	Milford	art_gallery	bar	night_club	point_of_i	establishm	NA	0	0	0
22	41.236	-73.0356	rue21	1201 Bost	art_gallery	point_of_i	store	establishm	NA	NA	4.2	87	1
23	41.2193	-73.0128	Foran High	80 Foran F	atm	finance	point_of_i	establishm	NA	NA	2.9	11	0
24	41.2386	-73.02	Cracker Ba	30 Researc	atm	finance	point_of_i	establishm	NA	NA	4.3	4689	2
25	41.251	-73.0245	Cardtronic	1718 Bost	atm	finance	point_of_i	establishm	NA	NA	0	0	0

Full_data

As shown in the above image we deal with multiple attributes such as:
Latitude & Longitude: Coordinates of a business so that we able to predict if user is available in available range of coordinates.
Name: Name of different businesses at coordinates.
Vicinity: Human readable location of a business.
Type: Indicates different attributes of a store such as availability of food, coffee, restaurant….
Rating: Indicates the rating of a business by the customer, it's one of the key parameter for the business model, it's one of the key parameter whether a new customer is willing to visit or not.
User rating: Number of ratings users made on business. so that if the business has more ratings with God rating it indicates good business.

List of Data Mining Techniques used in optimization Phase:

1) Content Based Recommendation System.
2) Random Forest Classifier

Details of Model parameters and Hyperparameters used in Optimization datamining techniques:

Data Mining Techniques	Model Parameters	Hyperparameters
Content Based Recommendation System.	Features, rating	Name, storeId, userId
Random Forest Classifier	Rating, user_ratings_total	Price_level

Content Recommendation System:
The model parameters used in previous content recommendation system are Features and technique used is cosine similarity, in optimized technique the model parameters are Features and rating and technique used is cosine similarity with Tf-idf. we were able to recommend better stores by including rating with Features parameter.

In the optimized technique we will be using precision to evaluate the recommendation system with an value of 75% .

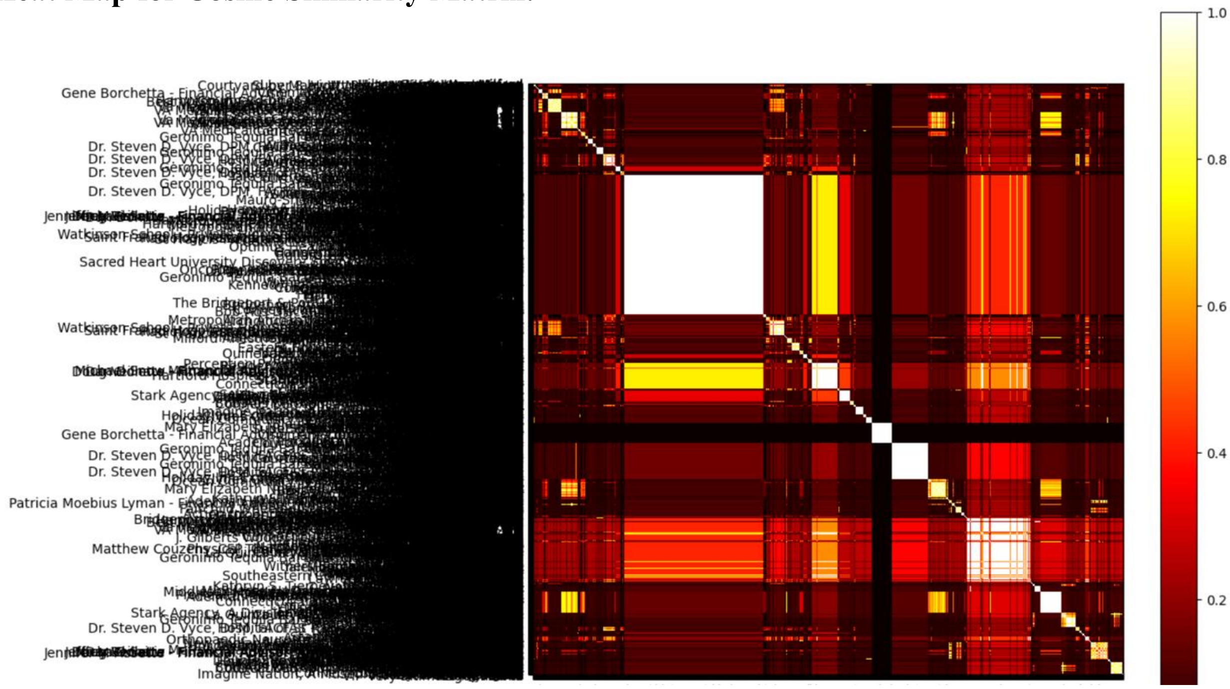
```
# 5. Test the recommendation system
movie_title = 'West Haven'  # Replace with the store title you want recommendations for
recommendations = recommend_movies(movie_title)
print("Recommended stores based on", movie_title, "are:")
print(recommendations)
```

```
Recommended stores based on West Haven are:
1                          Hilton Garden Inn Milford
9                                               ALDO
5                 Super 8 by Wyndham Milford/New Haven
6                                             Zumiez
7                               Connecticut Post Mall
8                                         AT&T Store
2                      Hyatt Place Milford / New Haven
10                                     Hollister Co.
3     Courtyard by Marriott New Haven Orange/Milford
4                                 Hampton Inn Milford
Name: name, dtype: object
```

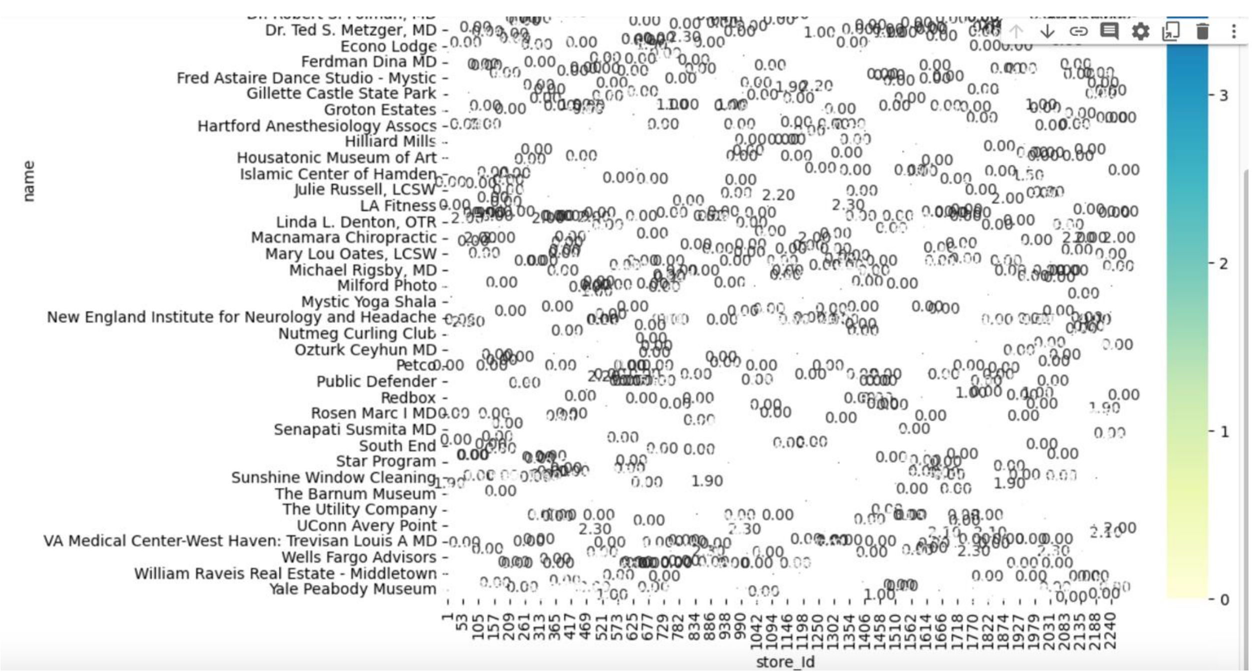
```
West Haven: Precision = 0.40, Recall = 0.67
Average Precision = 0.40, Average Recall = 0.67
```

From the above precision level west have has got the 40% precision rate and 67% Recall.

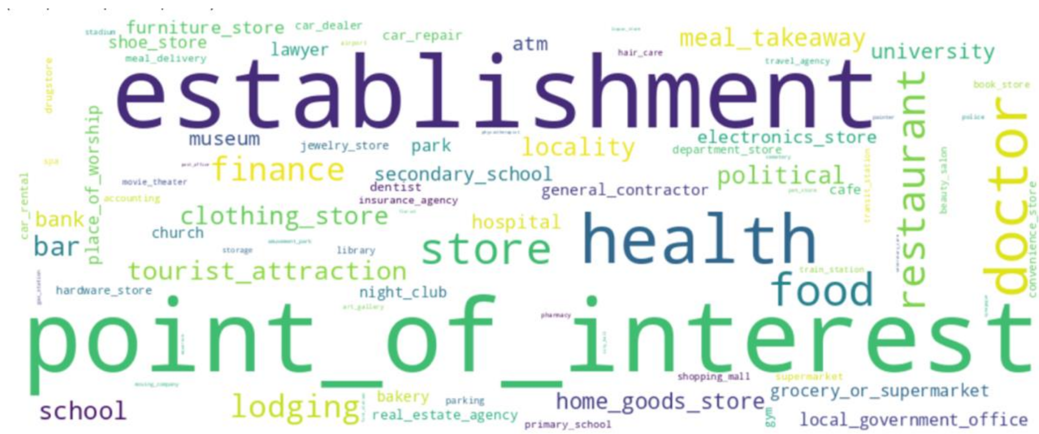
Heat Map for Cosine Similarity Matrix:



Heat map to visualize ratings data:



Feature Frequency:



Random Forest Classifier:

```
In [28]: #Create classifier
         rf_classifier = RandomForestClassifier(n_jobs=-1)

In [29]: # set different parameter values to tune
         param_grid = {
             "n_estimators": [100, 200, 300, 400],
             "max_depth": [1, 3, 5, 7, 9],
             "criterion": ["gini", "entropy"],
         }

In [31]: # train the model with gridserchCV
         model.fit(X_scaled,y)

         Fitting 5 folds for each of 40 candidates, totalling 200 fits
         [CV] END ......criterion=gini, max_depth=1, n_estimators=100; total time=
         3.4s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=100; total time=
         0.1s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=100; total time=
         0.2s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=100; total time=
         0.1s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=100; total time=
         0.1s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=200; total time=
         0.3s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=200; total time=
         0.4s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=200; total time=
         0.3s
         [CV] END ......criterion=gini, max_depth=1, n_estimators=200; total time=
         0.3s

In [32]: # print the best score and estimator
         print(model.best_score_)
         print(model.best_estimator_.get_params())

         0.9052631578947368
         {'bootstrap': True, 'ccp_alpha': 0.0, 'class_weight': None, 'criterion': 'gi
         ni', 'max_depth': 9, 'max_features': 'auto', 'max_leaf_nodes': None, 'max_sa
         mples': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 1, 'min_samp
         les_split': 2, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 200, 'n_job
         s': -1, 'oob_score': False, 'random_state': None, 'verbose': 0, 'warm_star
         t': False}
```

```
In [70]: # perform optimization
         result = gp_minimize(
             func=evaluate_model,
             dimensions=search_space,
             n_calls=30,
             random_state=42,
             verbose=True,
             n_jobs=1,
         )

         Iteration No: 1 started. Evaluating function at random point.
         Iteration No: 1 ended. Evaluation done at random point.
         Time taken: 3.8747
         Function value obtained: -0.8882
         Current minimum: -0.8882
         Iteration No: 2 started. Evaluating function at random point.
         Iteration No: 2 ended. Evaluation done at random point.
         Time taken: 3.8083
         Function value obtained: -0.8368
         Current minimum: -0.8882
         Iteration No: 3 started. Evaluating function at random point.
         Iteration No: 3 ended. Evaluation done at random point.
         Time taken: 2.3749
         Function value obtained: -0.8368
         Current minimum: -0.8882
         Iteration No: 4 started. Evaluating function at random point.
         Iteration No: 4 ended. Evaluation done at random point.
         Time taken: 3.8165
         Function value obtained: -0.8873

In [71]: # summarizing finding:
         print('Best Accuracy: %.3f' % (abs(result.fun)))
         print('Best Parameters: %s' % (result.x))

         Best Accuracy: 0.907
         Best Parameters: [400, 'gini', 9]
```

The above image depicts random forest with a best score when the evaluation pattern is “gini index” to build decision tree with 9 levels in building a decision tree.

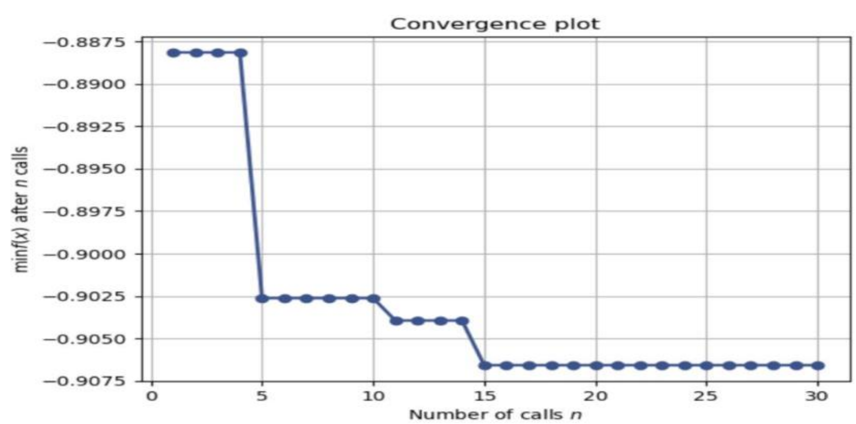
Convergence Plot for the Random Forest Classifier:

```
In [73]: # plot convergence

         from skopt.plots import plot_convergence

         plot_convergence(result)

Out[73]: <AxesSubplot:title={'center':'Convergence plot'}, xlabel='Number of calls $n
         $', ylabel='$\\min f(x)$ after $n$ calls'>
```



Conclusion:

The ability of Recommending a store to the user has improved by using content-based recommendation by tuning its model parameters with additional model parameters which gives a better efficient way compared to previous technique. The optimized content based recommendation system creates cosine similarity for an transformed data by using tf-idf which gives more accuracy, whereas in random forest classifies the usage of “gini” over “entropy ” for better results.