STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

Business Information Prediction for Yelp Restaurants

Course: BIA660-WS Web Analytics – Fall 2017

Guided By: Prof. Theodoros Lappas

INTRODUCTION

- This project is a Python implementation that can mine/scrape at least 30 reviews of more than 1000 restaurants in Europe from Yelp.com.
- Used multiple Classification Machine Learning Algorithms in Python, to predict the correct values and prediction accuracy for entries in the "Business info" section, listed on the right side of the screen of Yelp restaurant web page.
- Business Information prediction example: If this Restaurant is/has
  - **"Good For kids”**
  - **”Good For Groups”**
  - **“Wi-Fi”**
  - **“Private Parking available”**
  - **“Outdoor Seating available”**

WEB SCRAPING

- Scraped at least 30 reviews of more than 1000 restaurants in Europe from Yelp.com as below:

City	No of Restaurants
London	543
Paris	227
Berlin	93
Manchester	163
Total	**1026**

- We have stored first 2 pages for each restaurants for getting at least 30 reviews in English language as one page contains only 20 reviews.
- Stored all web pages of restaurants in HTML formal in local hard drive.
- Scraped data link: https://drive.google.com/open?id=1oK57zlwLWlh0ydmh8bwNYpWBvdfnL5AN

REVIEW TRANSFORMATION

- While scraping, created a tab separated text file for each city containing Restaurant name, Restaurant Location, 30 Reviews, and Business Information.
- Merged all text files for all cities into two train and test tab separated text files containing,
  - First element as merged all 30 reviews into one mega review
  - Second element as 0 if **Good for Kids** = No or as 1 if **Good for Kids** = Yes
  - Third element as 0 if **Wi-Fi** = No or as 1 if **Wi-Fi** = Yes/Free/Paid
  - Forth element as 0 if **Good for Groups**= No or as 1 if **Good for Groups**= Yes
  - Fifth element as 0 if **Parking**= No/Street or as 1 if **Parking**= Yes/Private/Valet/Garage
  - Sixth element as 0 if **Outdoor Seating**= No or as 1 if **Outdoor Seating**= Yes
- New created Train file contains 70% of the data and used for train of machine learning algorithm model.
- New created Test file contains 30% of the data and used for test and accuracy check of machine learning algorithm model.

CLASSIFIER: RandomForestClassifier

- Below is Accuracy Matrix for RandomForestClassifier using different Vectorizer and classifier Parameters.

Accuracy Matrix	Good for Kids	Wi-Fi	Good for Groups	Parking	Outdoor Seating
CountVectorizer() RandomForestClassifier(n_estimators=1400,criterion='entropy',max_features='log2', oob_score=True,max_depth=5000,min_samples_split=162,random_state=150, n_jobs=8)	0.68137254902	0.598039215686	**0.852941176471**	**0.833333333333**	0.696078431373
CountVectorizer() RandomForestClassifier(n_estimators=1550,criterion='gini',max_features='auto', oob_score=False, n_jobs=1, max_depth=5000,min_samples_split=162,random_state=None)	**0.78431372549**	0.637254901961	0.852941176471	**0.833333333333**	0.754901960784
TfidfVectorizer() RandomForestClassifier(n_estimators=1400,criterion='entropy',max_features='log2', oob_score=True,max_depth=5000,min_samples_split=162,random_state=150, n_jobs=8)	0.68137254902	**0.642156862745**	0.852941176471	**0.833333333333**	0.68137254902
TfidfVectorizer() RandomForestClassifier(n_estimators=1550,criterion='gini',max_features='auto', oob_score=False, n_jobs=1, max_depth=5000,min_samples_split=162,random_state=None)	0.754901960784	0.637254901961	0.852941176471	**0.833333333333**	**0.78431372549**
TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',ngram_range=(4,5), stop_words='english') RandomForestClassifier(n_estimators=1550,criterion='gini',max_depth=None,min_samples_ split=2,min_samples_leaf=1,min_weight_fraction_leaf=0.0,max_features='auto',max_leaf_n odes=None,bootstrap=True,oob_score=False,n_jobs=1,random_state=None,verbose=0,war m_start=False,class_weight=None)	0.563725490196	0.573529411765	0.852941176471	**0.833333333333**	0.602941176471

CLASSIFIER: RandomForestClassifier

- We get overall good accuracy using RandomForestClassifier with for both TfidfVectorizer and CountVectorizer.
- Runtime is very slow while we set TfidfVectorizer parameter ngram_range= (4,5) and Accuracy is also low.
- Highest Accuracy for Good for Kids:
  - Run 2 Using CountVectorizer and RandomForestClassifier; Accuracy: **0.78431372549**
- Highest Accuracy for Wi-Fi :
  - Run 3 Using TfidfVectorizer and RandomForestClassifier; Accuracy: **0.642156862745**
- Highest Accuracy for Good for Group:
  - For every combination we get same Accuracy: **0.852941176471**
- Highest Accuracy for Parking Availability:
  - For every combination we get same Accuracy: **0.833333333333**
- Highest Accuracy for Outdoor Seating:
  - Run 4 Using TfidfVectorizer and RandomForestClassifier; Accuracy: **0.78431372549**

CLASSIFIER: MultinomialNBClassifier

- Below is Accuracy Matrix for MultinomialNBClassifier using different Vectorizer and classifier Parameters.

Accuracy Matrix	Good for Kids	Wi-Fi	Good for Groups	Parking	Outdoor Seating
CountVectorizer() MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)	**0.828431372549**	0.642156862745	0.848039215686	**0.921568627451**	**0.686274509804**
TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',ngram_range=(1,5), stop_words='english') MultinomialNB(alpha=0.5, class_prior=None, fit_prior=True)	0.583333333333	0.602941176471	**0.852941176471**	0.833333333333	0.627450980392
TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',ngram_range=(4,5), stop_words='english') MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)	0.563725490196	0.578431372549	**0.852941176471**	0.833333333333	0.602941176471
TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',ngram_range=(1,2),s top_words='english') MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)	0.578431372549	0.602941176471	**0.852941176471**	0.833333333333	0.627450980392
CountVectorizer() MultinomialNB(alpha=2.0, class_prior=None, fit_prior=True)	0.823529411765	**0.647058823529**	**0.852941176471**	0.882352941176	0.68137254902

CLASSIFIER: MultinomialNBClassifier

- We get overall good accuracy using MultinomialNBClassifier Classifier with for both TfidfVectorizer and CountVectorizer
- Runtime is very slow while we set parameter of TfidfVectorizer as ngram_range= (4,5) and Accuracy is also low.
- Highest Accuracy for Good for Kids:
  - Run 1 Using CountVectorizer and MultinomialNBClassifier; Accuracy: **0.828431372549**
- Highest Accuracy for Wifi Available:
  - Run 5 Using CountVectorizer and MultinomialNBClassifier; Accuracy: **0.647058823529**
- Highest Accuracy for Good for Group:
  - For every combination except run1 we get same Accuracy: **0.852941176471**
- Highest Accuracy for Parking Availability:
  - Run 1 Using CountVectorizer and MultinomialNBClassifier; Accuracy: **0.921568627451**
- Highest Accuracy for Outdoor Seating:
  - Run 1 Using CountVectorizer and MultinomialNBClassifier Accuracy: **0.686274509804**

CLASSIFIER: LinearSVC

- Below is Accuracy Matrix for LinearSVC using different Vectorizer and classifier Parameters.

Accuracy Matrix	Good for Kids	Wi-Fi	Good for Groups	Parking	Outdoor Seating
TfidfVectorizer(stop_words=u'english',ngram_range=(2,3),lowercase=True) LinearSVC()	1.0	1.0	1.0	1.0	1.0
TfidfVectorizer() LinearSVC(C=1.0,dual=True,fit_intercept=True,intercept_scaling=1,loss='hinge', max_iter=1000,multi_class='ovr',penalty='l2',random_state=None, tol=0.0001,verbose=0)	0.892420537897	0.768948655257	0.864303178484	0.940097799511	0.784841075795
TfidfVectorizer(sublinear_tf=True, max_df=0.5, analyzer='word', stop_words='english') LinearSVC(C=1.0,dual=True,intercept_scaling=1,loss='hinge',max_iter=1000, multi_class='ovr',random_state=None,tol=0.0001,verbose=0)	0.996332518337	0.993887530562	0.988997555012	0.984107579462	0.996332518337
CountVectorizer(stop_words=u'english',ngram_range=(2,4),lowercase=True) LinearSVC(penalty='l1', loss='squared_hinge', random_state=6, dual=False, tol=1e-3, intercept_scaling=5)	1.0	1.0	1.0	1.0	1.0
CountVectorizer() LinearSVC()	1.0	1.0	1.0	1.0	1.0

CLASSIFIER: LinearSVC

- We get overall 100% accuracy using LinearSVC Classifier with for both TfidfVectorizer and CountVectorizer
- Accuracy is low when we set RandomState=None.
- Highest Accuracy for Good for Kids:
  - Run 1, Run4 and Run5 where random_state is not none, Accuracy: **1.0**
- Highest Accuracy for Wifi Available:
  - Run 1, Run4 and Run5 where random_state is not none, Accuracy: **1.0**
- Highest Accuracy for Good for Group:
  - Run 1, Run4 and Run5 where random_state is not none, Accuracy**: 1.0**
- Highest Accuracy for Parking Availability:
  - Run 1, Run4 and Run5 where random_state is not none, Accuracy: **1.0**
- Highest Accuracy for Outdoor Seating:
  - Run 1, Run4 and Run5 where random_state is not none, Accuracy: **1.0**

CLASSIFIER: GridSearch

- Below is Accuracy Matrix for GridSearch using different Vectorizer and classifier Parameters.

Accuracy Matrix		Good for Kids	Wi-Fi	Good for Groups	Parking	Outdoors Seating
CountVectorizer(stop_words= stopwords.words('english'))	MultinomialNB() 'alpha':[0.8, 0.85, 0.9, 0.95, 1.0], 'fit_prior' :[True,False]	0.833333333333	0.637254901961	0.857843137255	**0.921568627451**	0.686274509804
	KNeighborsClassifier() 'n_neighbors': [1,3,5,7,9,11,13,15,17], 'weights':['uniform','distance']	0.78431372549	0.588235294118	0.852941176471	0.877450980392	0.647058823529
	DecisionTreeClassifier() 'max_depth': [3,4,5,6,7,8,9,10,11,12],' criterion':['gini','entropy']	0.700980392157	0.519607843137	0.862745098039	0.887254901961	0.745098039216
	LogisticRegression() 'C':[0.5,1,1.5,2],'penalty':['l1','l2']	0.843137254902	0.563725490196	**0.872549019608**	0.897058823529	0.71568627451
	VotingClassifier(('knn',KNN_classifier),('lreg',LREG_classifier),('dt',DT_classifier), ('nb',NB_classifier))	**0.848039215686**	0.627450980392	0.862745098039	0.906862745098	**0.764705882353**
TfidfVectorizer(stop_words= stopwords.words('english'))	MultinomialNB()	0.651960784314	**0.656862745098**	0.852941176471	0.833333333333	0.666666666667
	KNeighborsClassifier()	0.78431372549	0.558823529412	0.852941176471	0.848039215686	0.671568627451
	DecisionTreeClassifier()	0.769607843137	0.558823529412	0.838235294118	0.877450980392	0.720588235294
	LogisticRegression()	0.843137254902	0.647058823529	0.852941176471	0.911764705882	0.769607843137
	VotingClassifier()	0.813725490196	0.642156862745	0.852941176471	0.833333333333	0.68137254902

CLASSIFIER: GridSearch

- Here We have used four predictor classifier MultinomialNB, KNeighborsClassifier, DecisionTreeClassifier, LogisticRegression and one voting classifier VotingClassifier to calculate accuracy.
- Highest Accuracy for Good for Kids:
  - Using CountVectorizer and VotingClassifier; Accuracy: **0.848039215686**
- Highest Accuracy for WIfi:
  - Using TfidfVectorizer and MultinomialNB; Accuracy: **0.656862745098**
- Highest Accuracy for Good for Groups :
  - Using CountVectorizer and LogisticRegression; Accuracy: **0.872549019608**
- Highest Accuracy for Parking:
  - Using CountVectorizer and MultinomialNB; Accuracy: **0.921568627451**
- Highest Accuracy for Outdoor Seating:
  - Using CountVectorizer and VotingClassifier; Accuracy: **0.764705882353**

CONCLUSION

- While Scraping, We have to check for multiple criteria;
  - Same name restaurant in different area should be scraped with number attached at the end of restaurant name
  - Only scrape that restaurant that have at least 30 English reviews, not 30 total reviews as in European cities have more reviews in other languages like French or German
- Grid Search is not providing best result accuracy every time.
- For reviews as input and Business Information as predict variable, **LinearSVC** Classifier gives 100% Accuracy.

stevens.edu

Thank You