ML4VA PROJECT

Andrew Lin
Department of Computer Science
University of Virginia
Charlottesville, VA
acl2mf@virginia.edu

Ali Ibrahim
Department of Computer Science
University of Virginia
Charlottesville, VA
ai9sk@virginia.edu

Kaleb Getachew
Department of Computer Science
University of Virginia
Charlottesville, VA
kg9rv@virginia.edu

November 23, 2020

1 Abstract

In order to alleviate some of the financial hardships felt by local restaurants during the pandemic, we will build a local restaurant recommender system that will be deployed to a website for people to use. Users will go on the website and enter some details about themselves and then predictions for compatible cuisines will be made. The website will then recommend three random restaurants, based on the user's location in Virginia, for each of the predicted cuisines.

2 Introduction

The main inspiration for this project was fueled by the dire financial situation of many local restaurants in Virginia due to the COVID-19 pandemic. A secondary inspiration was that each city has its own combination of unique and diverse cuisines to offer, we wanted to be able to highlight these local restaurants for residents as well as visitors of the region. This would make selecting restaurants as easy as entering information into a website.

The overall goal is to spotlight local restaurants since they have been one of the hardest hit during the COVID-19 pandemic. According to an article written on the impact of the pandemic on restaurants, the "National Restaurant Association, the restaurant and food service industry likely lost nearly $120 billion in sales during the first three months of the pandemic".

Specifically, we are focusing our efforts to the restaurants located in the major areas of Virginia, these are Northern Virginia (NoVA), Roanoke, Charlottesville, Richmond, Virginia Beach, and Blacksburg.

Previous research has been done to explore the effects of relevant contextual features on restaurant recommender systems. It highlighted a recommender system called *Surfeos*, which used collaborative filtering to give recommendations.

In addition, there have been similar efforts within the restaurant industry that have utilized Machine Learning algorithms. One such example was a *Stanford Paper* which predicted restaurant rating and conducted an analysis of the types of predictive models and how well they are able to predict the rating of a specific restaurant.

The idea being explored is if it is possible to accurately predict what cuisine a person would want eat depending on their personal attributes and restaurant atmosphere preferences. Therefore the question we will attempt to answer is: "To what extent can we predict people's cuisine preferences based on their attributes?"

3 Method

3.1 Original Dataset Description

Our User-Restaurant dataset can be found here: Restaurant Data with Consumer Ratings (hyperlink)

The dataset contains a total of nine csv files. Two of which, we believe that we will discard because they do not seem to be relevant for our purposes. The remaining seven csv files are what we will use. Four of the csv files pertain to

restaurant characteristics, ex. type of payment accepted, cuisine served, hours of operation, etc. The other three csv files pertain to the attributes and preferences of a customer. ex. cuisine preferred, dress preference, ambience, etc.

3.2 Data Cleaning and Survey

One of the most important tasks was to clean the data and turn it into a form we can use to train the machine learning models. The first thing we did was find and fill in all the missing values. We did the following to the columns with missing values: dress preference = no preference, ambience = family, transport = public, hijos = independent, budget = low, smoker = False, marital status = single.

We chose those values as the default because were the mode for that column or they weren't highly specific. The original data-set had a birth year column, which we transformed into an age column to keep the data brief and numeric. With the new age column, we used conditions to fill in the activity column's missing values:

if age <= 22: activity = student,

if age > 22 and age < 65: activity = professional,

if age >= 65: activity = unemployed.

One problem that arose during training and testing our baseline model was that of a single unique observation under the religion category. The only observation that had "Mormon" in the religion category was put in the test set randomly, thus the model kept getting errors because it had not recognized "Mormon" during the training portion. As a result, the religion observation of "Mormon" was set to "Other" to ensure overall success. We also did something similar with Widow in the marital status column. We switched the only two widows to single for the same reason.

While imputing values for missing data does introduce some bias, we believed that it is the best option for a data-set, especially where there are only 138 observations.

From the original data-set, we dropped the longitude, latitude, userID, and birth year features because we did not believe they would provide valuable information in predicting a favorite cuisine and would introduce too many features if we were to one hot encode some of them.

We then transformed the remaining column data using encoders and pipelines. We looked at the features and decided it would be beneficial to turn drink level and dress preference into ordinal columns since they can be subdivided into discrete labels . We then used OneHotEncoder to encode all the other categorical variables. For numerical data, we simply used the standard scaler to ensure an even scale for our numeric data.

All together we were able to implement a full pipeline that encoded the numerical and categorical columns, into relevant forms of information to achieve our goal of predicting cuisines.

However, with only 138 observations and a huge imbalance and scarcity of some labels, we had to brainstorm a quick way to fix this problem. We decided to make a survey that contained all the features we needed and sent it out to various places online. This included subreddits like r/UVA and r/SampleSize and other online communities.

We retrieved results of the survey in a google sheets form where we manually edited it in there to make the column names match and the values for columns match. We did not force the respondents to enter their weight and height, as a result for missing values in those categories, we imputed the sample median for weight and sample mean for height. Our new survey allowed us to include 2 new categories to the religion column. We were able to add "Muslim" and "Other" to the column.

Through the survey, we were able to get a more representative sample than what the original dataset provided. The original dataset had data that was gathered in Mexico, making the dataset heavily biased in terms. With our data, we have an American and quite possible a strong Virginian population, which is where this project is focused on.

In order to avoid multicollinearity, we decided to combine height and weight into a ratio: weight/height.

3.3 Stratified sampling, training and testing

We used Stratified sampling to make our train and test sets. We stratified on age and made 3 bins, $\leq 20, 21-23, 23 \geq$. The bins were found by making a histogram and finding out which bins made a decent Normal distribution. We decided to use this technique because it would be unwise not to at least make the train and test set models representative of the population. We decided "age" was the best feature to stratify because we believed that cuisine preferences would vary between different age groups. For example, older people's preferences may not vary much within the same age group, however, they could differ vastly when compared to a younger person's cuisine preferences.

3.4 Machine Learning Models Tried Out

We made a binary classifier for each of these 8 cuisine categories: "American", "Asian", "Austronesian", "Desserts/Cafe", "European", "Mediterranean/Middle East/African", "Mexican/Southwestern/Latin American", "Modern". We combined "African" with "Mediterranean/Middle East" and "Austronesian" with "Asian" because their models were still having a hard time learning them and those were the categories that we deemed "close enough."

The models we tried out were: RandomForestClassifier, Logistic Regression, Support Vector Machine, Adaboosted Decision Tree, GradientBoostingClassifier, Bagging Decision Tree Classifier, and Bernoulli Naive Bayes Classifier. Some combination of these models were tested and used in a VotingClassifier as the final model for each cuisine predictor.

We decided to use RandomForestClassifier because since it is a robust and uses both boosting and bagging. Logistic Regression is just a simpler model that can be applied to classify something. We did not use support vector machine in any VotingClassifier for a cuisine, but it was quick to train and find optimal hyperparameters, so we included it in the exploration. An Adaboosted Decision Tree helped us since it uses Adaptive Boosting. This would allow the final tree to be a strong classifier since it learned from the mistakes of the previous trees. GradientBoostingClassifier is a model we came about in our research for models that worked well with small datasets. Bagging Decision Tree just uses bagging, which is important for our dataset. The Bernoulli Naive Bayes Classifier is also a model that we came about in our research for models that worked well with small datasets.

We used GridSearchCV and RandomizedSearchCV where appropriate for each of the models trained on each of the cuisines. The loss we used was "balanced accuracy" because the imbalanced labels made it difficult to get both true positives and true negatives.

3.5 Deployment

Once the models were trained in Google Colab, we pickled the models one by one into byte-streams. In our Django environment, we uploaded the pickled models and imported them into our views.py. There we created various functions that allowed us to take in user input, run a prediction for each cuisine based on the user input, and return a list of local (to the user) restaurants that we think the users will like (corresponding to the selected predicted preferred cuisines).

4 Experiments:

The preliminary experiment we ran was to first test if our baseline model could predict if a person would want Mexican cuisine or not. We went through each observation's cuisine preferences and if "Mexican" was in it, then we labeled their cuisine '1', otherwise it was labeled '0' to represent 'Not Mexican'.

4.1 Baseline Model

The baseline model we used was a single Random Forest classifier. We used GridSearchCV to find the n estimators, max depth, and max features hyper-parameters. GridSearch found that n estimators = 10, max depth = 11, max features = 4 made the best model.

We then trained the Random Forest model on the new labels, tested it and calculated the confusion matrix, precision, recall, and accuracy scores. These were our results from the test set:

$$\text{Confusion Matrix } \begin{bmatrix} 2 & 1 \\ 5 & 13 \end{bmatrix}$$

Precision: 0.93 Recall: 0.72 Accuracy: 0.71

We knew we could do better, so we decided on making an ensemble model. So far, we have made a model that consists of 10 Random Forest models, all different using RandomizedSearchCV for 9 of them, 2 Logistic models, 2 Decision Tree models, 2 Adaboost models, and 2 Bagging models.

In the final voting classifier we made, we used all of these models and hard voting to predict the labels on the test set. These were our results from the test set:

$$\text{Confusion Matrix } \begin{bmatrix} 1 & 2 \\ 0 & 18 \end{bmatrix}$$

Precision: 0.9 Recall: 1.0 Accuracy: 0.9

Obviously the ensemble method did a lot better than just a single Random Forest model and we plan to continue toying around with the ensemble to possibly make it even better.

4.2 Problem

After the success of just predicting a binary outcome, we attempted to make it a multiclass classifier, but we ran into the problem of having a smaller dataset. The error is unclear, but we believe it is due to some labels appearing in the test set and not at all in the train set or some of the labels are not even being picked up in our method of making multi-class labels because there were so few of them.

We experimented with different losses for the GridSearchCV and RandomizedSearchCV and decided that "balanced accuracy" was the most appropriate. We also experimented with different combinations of models and chose the combination that had the highest AUC.

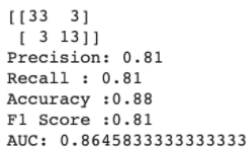
4.3 Reflection

We decided to not have as many models as we did in the VotingClassifier because we knew that it was better to have many different classifiers than just some similar ones together. So we made the limitation of one type of model in each VotingClassifier in our final models.

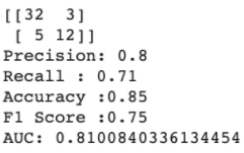
5 Results

We deployed the models to ml4va.herokuapp.com/predictor for everyone to use. The website carried out all of our functional needs and asked users for the necessary information needed for the models to predict cuisines. After submitting their information, the pickled models will process the inputs and output a list of cuisines that it thinks are compatible with the user's information. In addition, it will also randomly select some restaurant websites related to the cuisine to recommend to the user.

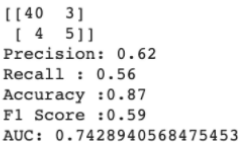
Here is how each individual model performed:



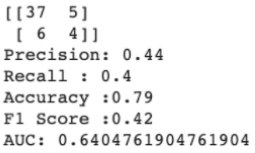
American Cuisine Model



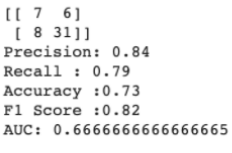
Desserts/Cafe Cuisine Model



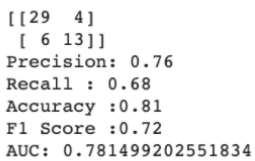
European Model



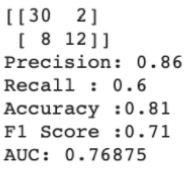
Modern Model



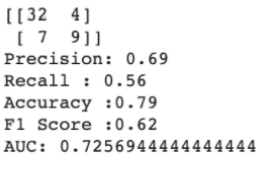
Mexican/Southwestern/ Latin American Cuisine Model



Asian/Austronesian Cuisine Model



Italian Cuisine Model



Mediterranean/ Middle Eastern/African Cuisine Model

Each cuisine model performed differently, as expected, depending on the balance of positive labels and zero labels. For example, the American cuisine model performed well because it was more balanced in its labeling unlike Mexican/Southwestern/Latin American cuisine, which was heavily weighted towards positive labels. The Modern cuisine model had the opposite problem of the Mexican/Southwestern/Latin American cuisine model as it was heavily overweight on the zero labels.

We allowed a few people to try the website to see if it was somewhat accurate. The testers said yes it was accurate. Obviously, this isn't a proper metric to go by, but at least it shows that the website predictor isn't totally off.

In the future, it would be ideal to gather more data on the lagging cuisines of Modern, Austronesian, and African.

6 Conclusion

6.1 Discussion of results

With the introduction of the data we had collected on our own, our models significantly improved, in terms of correctly classifying certain cuisines to certain people. We also achieved a diverse set of response in terms of cuisine preferences and user backgrounds so nearly all of our models improved. The question we set out to answer was "To what extent can we predict people's cuisine preferences based on their attributes" and based on the performance metrics of the model, we conclude that while predicting people's cuisine preferences is not a trivial task, it's not an impossible one either: we've found that we can predict people's preference for a single cuisine with about 80% accuracy, depending on the exact cuisine.

6.2 Impact

The impact that this could have on Virginia's economy, specifically the health of the states local businesses is huge. If enough people visit the website, get an appropriate cuisine predicted, and order food or go to one of the hand-picked local restaurants we recommend alongside each cuisine then we can significantly help restaurants regain some of their lost revenue during this pandemic and prevent them from going out of business until the pandemic is over.

6.3 Shortcomings

There are 2 major shortcomings with our results that are a consequence of our approach. Since we have 8 binary classifiers and predict whether someone will like a specific cuisine or not for each cuisine, in exceedingly rare cases we predict a 0 for every single cuisine for an individual (and thus don't predict any restaurants). The other major shortcoming is that each model's accuracy is about 80%. While this isn't a bad accuracy, especially for a nontrivial or solved task, the error compounds since we do 8 classifications.

6.4 Future Work

We've seen our modes improve significantly by just adding a few observations. We are confident that if we collect even more data and increase the number of observations, especially in the form of positive labels for cuisines that didn't have many positive labels, then we can improve the performance of our models even more and potentially eliminate or significantly reduce both shortcomings discussed above.

Furthermore, instead of using 8 classifiers, we could potentially use a neural net model to classify which cuisine an individual would like the most as opposed to predicting whether an individual would like a given cuisine or not.

In addition, it may be interesting to look into applying the same concepts to other regions/states across the country and potentially uncover any underlying trends between region and cuisine preferences.

7 References

7.1 Team Contributions

We worked closely on both the coding and report aspects of the project. We used pair programming to come up with code, and implement and deploy the model, frequently bouncing ideas and solutions off of each other. Lastly, we all participated in discussions regarding planning and implementation.

Andrew Lin: Made survey, cleaned data, feature engineered, trained and tuned models, helped deploy

Ali Ibrahim: Cleaned data, trained models, handled deployment

Kaleb Getachew: Made survey, cleaned data, trained models, helped deploy

7.2 Resources

References

[1] Blanca Vargas-Govea, Gabriel González-Serna, Rafael Ponce-Medellín : Effects of relevant contextual features in the performance of a restaurant recommender system
http://ceur-ws.org/Vol-791/paper8.pdf Centro Nacional de Investigación y Desarrollo Tecnológico, 2011

[2] I., Chris : Productionize a Machine Learning Model with a Django API,
https://towardsdatascience.com/productionize-a-machine-learning-model-with-a-django-api-c774cb47698c December 2019

[3] Jaitley, Urvashi : Comparing Different Classification Machine Learning Models for an imbalanced dataset,
https://towardsdatascience.com/comparing-different-classification-machine-learning-models-for-an-imbalanced-dataset-fdae1af3677f Feb 2019

[4] McCarthy, Kelly, *Restaurant, food service industry has lost nearly $120B due to pandemic* https://abcnews.go.com/Business/restaurant-food-service-industry-lost-120b-due-pandemic/story?id=71301061 ABC News, June 2020

[5] Metrics and scoring: quantifying the quality of predictions,
https://scikit-learn.org/stable/modules/model_evaluation.html

[6] Nguyen, Rich *CS 4774* https://www.cs.virginia.edu/ nn4pj/teaching University of Virginia, 2020.