GEBZE TECHNICAL UNIVERSITY

CSE 454
Data Mining Project

Yeşim Yalçın
200104004094

16.12.2022

Contents

1-) Project Topic

A. Problem Definition & Details

The problem is determining which cuisine a given recipe belongs to. The data objects are the recipes, the attributes are the ingredients and the class attribute is the cuisines they belong to.

For this project I found a dataset created for a research. The dataset is a dataset taken from Kaggle and Nature. The dataset format is like below. Each row represents a recipe. The first attribute determines the class and rest of the attributes are the ingredients of the recipe.

SoutheastAsian,salt,cassava,margarine,coconut,milk
EastAsian,eggs,pork,soy_sauce,water,sugar,wine,soy_sauce,tofu_puffs
NorthAmerican,baking_powder,baking_soda,salt,butter,all_purpose_flour,sugar,butter milk
SoutheastAsian,silken_tofu,ginger,shallots,brown_sugar,cilantro,chili_paste,lemongrass,coconut,vegetable,soy_sauce,lime,crimini_mushrooms

In the dataset there are recipes from 7 different cuisines in total. The distributions are like below:

['African', 'EastAsian', 'MiddleEastern', 'NorthernEuropean', 'SouthAsian', 'SoutheastAsian', 'WesternEuropean'] [1114, 6926, 555, 647, 3456, 3430, 6040]

B. How to Solve the Problem

The dataset is created for a research and this research uses machine learning algorithms like BERT. However, because I needed to use data mining algorithms I decided to use a classification algorithm to solve the problem. I chose naive bayes classification as my method of solution. The reason why I chose naive bayes method was the fact that the dataset is already properly labeled, naive bayes is believed to give good results with big datasets and it does not have a very complicated implementation.

However, I knew using naive bayes purely has disadvantages as well. First of all, there are 7 classes in the dataset and there is a huge imbalance in some of them. Naive bayes should give good accuracy even in a scenario like that however, for the imbalanced classes the individual accuracies will suffer a lot. Therefore, I needed to minimize the class imbalance as much as possible for the best results. I decided to do random undersampling on some classes and data augmentation for some other classes.

Another problem that comes with Naive Bayes is the zero probability problem. Because of this problem the accuracies in test set results suffer a lot. I fixed that. Moreover, to get even better test set accuracies I used cross validation and boosting.

C. Problem Solution Success Comments

In my project I used various methods to improve the initial results. The initial training set results are from using naive bayes directly on the given dataset. It is expected to have around %70 overall accuracy but bad individual accuracies for the imbalanced classes.

After each applied method, I recalculated the training set results to show how much effect they had. It is expected to get better results after each method.

In the end it is expected to get a better accuracy compared to the initial accuracy. Moreover, the individual accuracies for the imbalanced classes must be improved a lot. Improving individual accuracies is more important than improving the overall accuracy as long as the overall accuracy does not go below the initial accuracy.

The accuracies for the training sets and the test sets will be calculated separetly. It is important to see improvement in the test set results like explained above. However, test set results are expected to get better especially after zero probability fix and cross validation.

It is expected to have good results in the end but with naive bayes it is still not expected to have perfect/very good results as naive bayes ignores dependance between the ingredients.

The accuracies might slightly differ after each different training because of the random oversampling and undersampling methods.

2-) Problem Solution

A. Data Preprocessing

The dataset I found had some faulty rows. Some rows had the same ingredients included multiple times, some rows had only one ingredient written. I started with detecting and removing those rows.

Afterwards, I needed to reshape the dataset in a way that I could use classification on it. I reshaped it in a way like below:

Cuisine,chicken,cinnamon,soy_sauce,onion,ginger,cane_molasses,cumin,garlic…
African,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0…

In the new format the first line is to represent the attribute names. Rest of the rows belong to a data object. Ones and zeros represent if the data object has the recipe represented on that column.

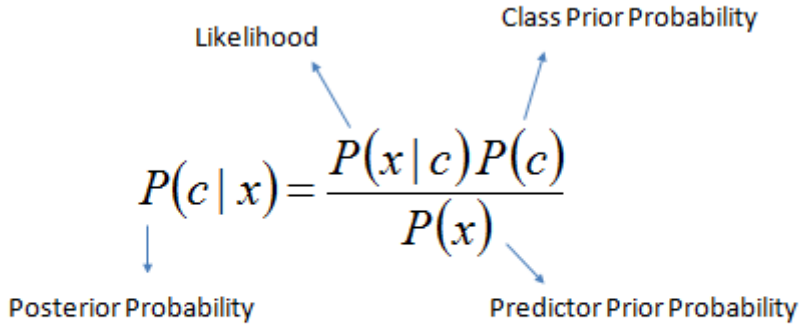
In the end, I created the training and test sets. The training and test sets differ in some of the techniques I used because of the undersampling, oversampling etc. They will be mentioned more in detail in their results sections.

All of the implementations for this part is in dataPreprocessing.py file.

B. Naive Bayes Classification

I implemented the naive bayes classification model myself. It is implemented in model.py file. It includes all the training and testing methods.

To implement the classification, I needed to calculate the probability of each class P(C). Afterwards, I needed to calculate the probabilty of each ingredient for each class P(I|C). Multiplying those two values gave me the probability of an ingredient belonging to the class P(C|I). After calculating all the probabilities, I saved them in a file. An example of it can be found in the file named trainResults.csv. This is the training part.



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

For a given recipe, it's probability to belong to each cuisine is looked from the trainResults.csv file. The cuisine with the highest probability is selected as the result. This is the testing part.

C. Enhancements to the Results

a. Undersampling to Fix Class Imbalance

I used random undersampling on the classes EastAsian, SouthAsian, SoutheastAsian and WeasternEuropean. Because undersampling might result in losing some important information, I did not want to overdo it. I cut around %50 of the classes. Moreover, I

wanted this process to be as random as possible not to have a bias in the results. To achieve that, I used a randomizer function randrange.

The implementaiton of undersampling is in dataPreprocessing.py.

b. Oversampling to Fix Class Imbalance

I used a data augmentation method on the classes African, MiddleEastern and NorthernEuropean. To implement the data augmentation method, I counted what ingredients are used how many times in each of these classes. Afterwards with the found ingredients and count weights, I created new augmented recipes randomly using numpy.random.choice() function. I did not want to overdo this either because I did not want to populate my classes with so many artificial data. I added different amounts to each of these 3 classes ranging from +%10 to +%60. I assumed having artificial data would not create many problems as naive bayes classification does not care about dependance between ingredients. Moreover the ingredients used to create the artifical data are already the ingredients heavily used in their respected cuisines.

The implementaiton of oversampling is in dataPreprocessing.py.

c. Zero Probabilty Fix

There is a possibilty that there can be a recipe with ingredient X belonging to cuisine A. However, that ingredient might never be found in any of cuisine A recipes in the training set. In a situation like that, that recipe will always be put in a wrong class. To fix that during training, I never set a possibility as 0. Instead I gave them some very low values depending on the classes. I needed to test to determine what values to give because giving very big values result in even worse classification, giving very low values result in almost no difference from not adding the fix at all.

The implementation of zero probability fix is in model.py.

d. Cross Validation

For the previous methos, I used %90 of the data for training and %10 of the data for testing. The data are randomly chosen each time for training and testing. However, this resulted in biased training, zero probabilities etc. To have even better results both for training and testing I decided to create my model with cross validation instead. I created 10 different test sets and 10 different corresponding train sets. After training a model with each of them, combined their results and used the combined results as a new model. Thanks to this, biases are reduced.

The implementation of cross validation is in trainAndTest4.py.

e. Boosting

Lastly, I thought about using boosting for the 10 different trained models of cross validation. I gave different weights to each models while combining them according to their train set result accuracies.

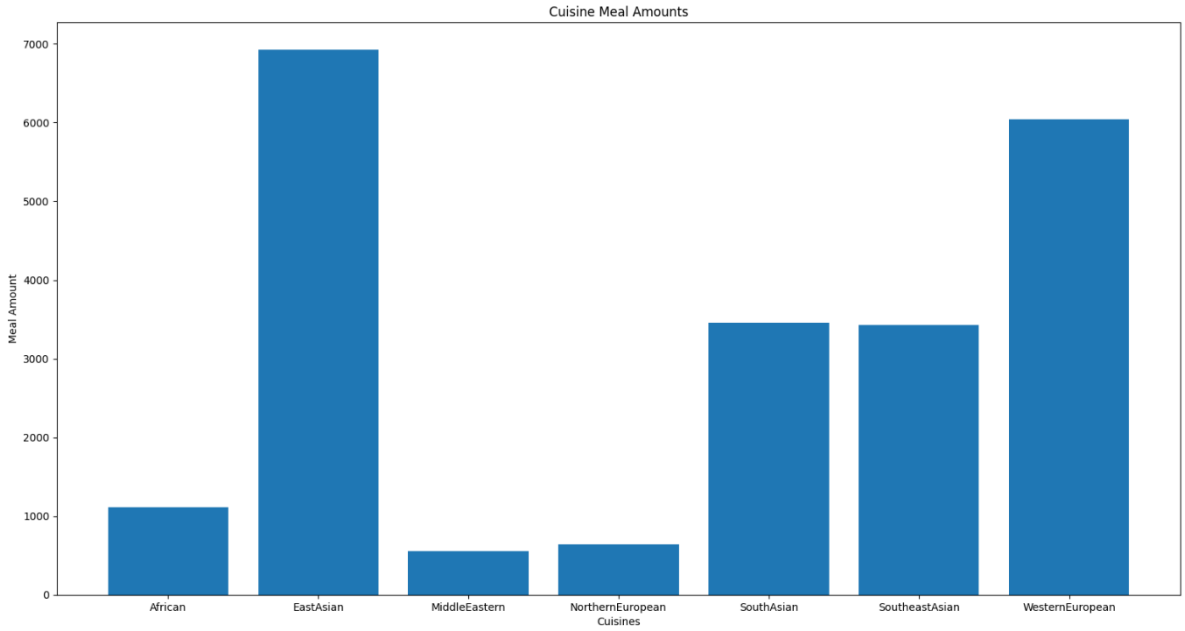
The implementation of boosting is in trainAndTest5.py.

3-) Results

A. Initial Results

The initial results are from applying naive bayes classification directly on the preprocessed dataset.

Dataset Details:



Cuisine Counts:
['African', 'EastAsian', 'MiddleEastern', 'NorthernEuropean', 'SouthAsian', 'SoutheastAsian', 'WesternEuropean']
[1114, 6926, 555, 647, 3456, 3430, 6040]

Ingredient Count: 2753

Accuracy Results:
TrainSet:
Accurate: 14270, False: 5684 -> %71.51448331161671

African: Total:1003, Accurate:109, False:4 -> %10.867397806580259
EastAsian: Total:6234, Accurate:6001, False:3334 -> %96.26243182547321
MiddleEastern: Total:500, Accurate:3, False:0 -> %0.6
NorthernEuropean: Total:583, Accurate:56, False:0 -> %9.605488850771868
SouthAsian: Total:3111, Accurate:2064, False:309 -> %66.34522661523626
SoutheastAsian: Total:3087, Accurate:841, False:30 -> %27.243278263686427
WesternEuropean: Total:5436, Accurate:5196, False:2007 -> %95.58498896247241

TestSet:
Accurate: 1454, False: 760 -> %65.67299006323395

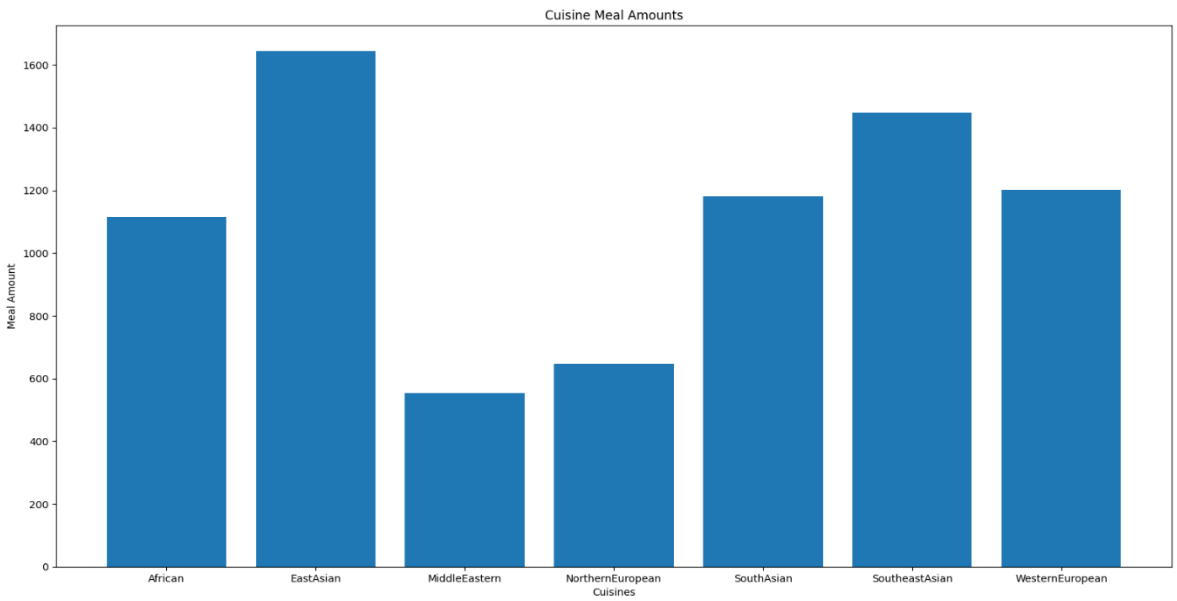
African: Total:111, Accurate:16, False:107 -> %14.414414414414415
EastAsian: Total:692, Accurate:625, False:393 -> %90.3179190751445
MiddleEastern: Total:55, Accurate:0, False:0 -> %0.0
NorthernEuropean: Total:64, Accurate:0, False:0 -> %0.0
SouthAsian: Total:345, Accurate:214, False:29 -> %62.02898550724638
SoutheastAsian: Total:343, Accurate:67, False:7 -> %19.533527696793
WesternEuropean: Total:604, Accurate:532, False:224 -> %88.0794701986755

It can be seen that the total accuracy for the train set is not bad however, the imbalanced classes have very bad percentages. Moreover, the test set accuracy suffers even more.

B. Undersampled Results

The undersampled results are from applying undersampling to the previous method's dataset.

Dataset Details:



Cuisine Counts:
['African', 'EastAsian', 'MiddleEastern', 'NorthernEuropean', 'SouthAsian', 'SoutheastAsian', 'WesternEuropean']
[1114, 1644, 555, 647, 1182, 1447, 1202]

Ingredient Count: 1967

Accuracy Results:

TrainSet:
Accurate: 5401, False: 1614 -> %76.99215965787599

African: Total:1003, Accurate:815, False:238 -> %81.25623130608174
EastAsian: Total:1480, Accurate:1421, False:416 -> %96.01351351351352
MiddleEastern: Total:500, Accurate:9, False:1 -> %1.7999999999999998
NorthernEuropean: Total:583, Accurate:130, False:1 -> %22.29845626072041
SouthAsian: Total:1064, Accurate:942, False:140 -> %88.53383458646617
SoutheastAsian: Total:1303, Accurate:1082, False:165 -> %83.03914044512663
WesternEuropean: Total:1082, Accurate:1002, False:653 -> %92.60628465804066

TestSet:
Accurate: 474, False: 302 -> %61.08247422680413

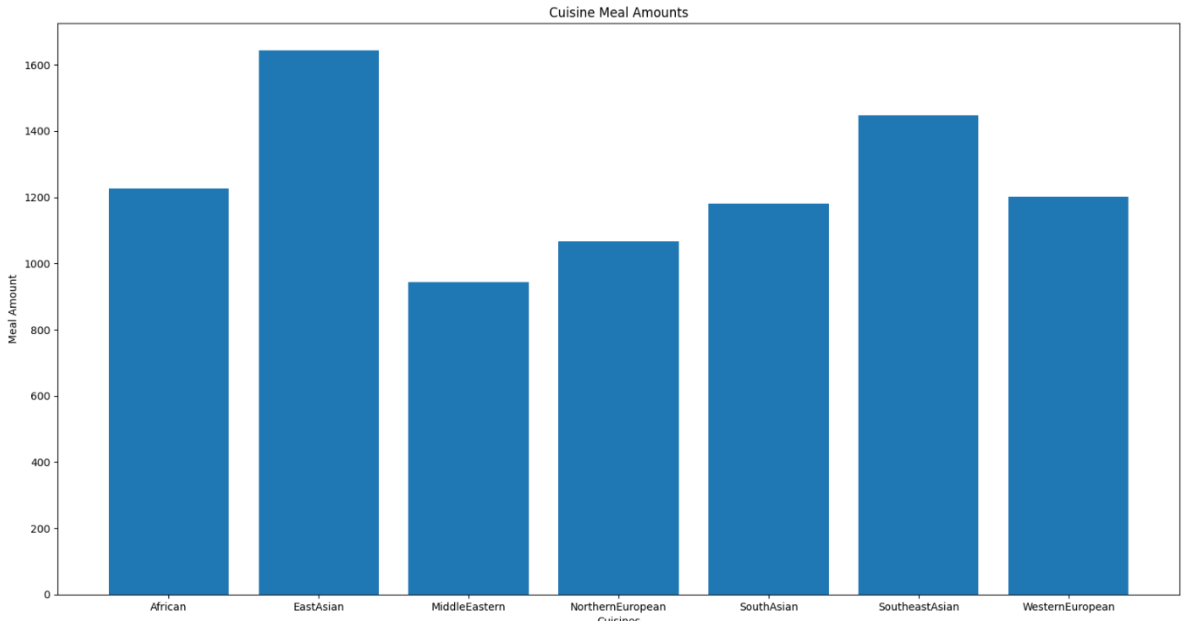
African: Total:111, Accurate:88, False:131 -> %79.27927927927928
EastAsian: Total:164, Accurate:135, False:61 -> %82.3170731707317
MiddleEastern: Total:55, Accurate:1, False:1 -> %1.8181818181818181
NorthernEuropean: Total:64, Accurate:5, False:1 -> %7.8125
SouthAsian: Total:118, Accurate:86, False:12 -> %72.88135593220339
SoutheastAsian: Total:144, Accurate:84, False:28 -> %58.333333333333336
WesternEuropean: Total:120, Accurate:75, False:68 -> %62.5

It can be seen that random undersampling had positive effect both on training set overall accuracy and imbalanced classes percentages. However, there is still a big imbalance and it effects the 3rd and 4th classes a lot. Moreover, the test set result accuracy got lower. This is not a direct effect of undersampling but effect of having more zero probability problems after undersampling. Still even in the test set, imbalanced class percentages improved.

C. Oversampled Results

The oversampled results are from applying oversampling to the previous method's dataset.

Dataset Details:



Cuisine Counts:
['African', 'EastAsian', 'MiddleEastern', 'NorthernEuropean', 'SouthAsian', 'SoutheastAsian', 'WesternEuropean']
[1226, 1644, 944, 1068, 1182, 1447, 1202]
Ingredient Count: 1938

Accuracy Results:

TrainSet:
Accurate: 6362, False: 1483 -> %81.09623964308477

African: Total:1104, Accurate:961, False:312 -> %87.04710144927536
EastAsian: Total:1480, Accurate:1405, False:400 -> %94.93243243243244
MiddleEastern: Total:850, Accurate:461, False:50 -> %54.23529411764706
NorthernEuropean: Total:962, Accurate:583, False:9 -> %60.6029106029106
SouthAsian: Total:1064, Accurate:928, False:103 -> %87.21804511278195
SoutheastAsian: Total:1303, Accurate:1089, False:128 -> %83.57636224098235
WesternEuropean: Total:1082, Accurate:935, False:481 -> %86.41404805914972

TestSet:
Accurate: 607, False: 261 -> %69.93087557603687

African: Total:122, Accurate:107, False:116 -> %87.70491803278688
EastAsian: Total:164, Accurate:129, False:46 -> %78.65853658536585
MiddleEastern: Total:94, Accurate:51, False:6 -> %54.25531914893617
NorthernEuropean: Total:106, Accurate:63, False:1 -> %59.43396226415094
SouthAsian: Total:118, Accurate:78, False:11 -> %66.10169491525424
SoutheastAsian: Total:144, Accurate:99, False:22 -> %68.75
WesternEuropean: Total:120, Accurate:80, False:59 -> %66.66666666666666

It can be seen that data augmentation helped to fix the class imbalance very well. Both the training set overall accuracy and class accuracies improved significantly. The test set results are improved significantly as well however, it still needs to be improved.

D. Zero Probability Fixed Results

The zero probability fixed results are from applying the fix to the previous method's dataset. Because the dataset will not change from now on, I will not be adding any new graphs or amounts.

Accuracy Results:

TrainSet:
Accurate: 6361, False: 1484 -> %81.08349267049076

African: Total:1104, Accurate:961, False:312 -> %87.04710144927536
EastAsian: Total:1480, Accurate:1405, False:401 -> %94.93243243243244
MiddleEastern: Total:850, Accurate:461, False:50 -> %54.23529411764706
NorthernEuropean: Total:962, Accurate:583, False:9 -> %60.6029106029106
SouthAsian: Total:1064, Accurate:928, False:103 -> %87.21804511278195
SoutheastAsian: Total:1303, Accurate:1089, False:128 -> %83.57636224098235
WesternEuropean: Total:1082, Accurate:934, False:481 -> %86.32162661737523

TestSet:
Accurate: 669, False: 199 -> %77.07373271889401

African: Total:122, Accurate:104, False:38 -> %85.24590163934425
EastAsian: Total:164, Accurate:148, False:49 -> %90.2439024390244
MiddleEastern: Total:94, Accurate:51, False:8 -> %54.25531914893617
NorthernEuropean: Total:106, Accurate:63, False:2 -> %59.43396226415094
SouthAsian: Total:118, Accurate:92, False:16 -> %77.96610169491525
SoutheastAsian: Total:144, Accurate:114, False:25 -> %79.16666666666666
WesternEuropean: Total:120, Accurate:97, False:61 -> %80.83333333333333

The main aim of the zero probability fix is to have better test set result. It can be seen that the test set accuracies improved significantly. However, it did not contribute into fixing class imbalance.

E. Cross Validation Results

The cross validation results are from applying cross validation and creating 10 new models then combining them from the previous method's dataset.

Accuracy Results:
TrainSet:
Accurate: 6388, False: 1454 -> %81.46065266460079

African: Total:1103, Accurate:965, False:299 -> %87.43882544861337
EastAsian: Total:1480, Accurate:1408, False:391 -> %95.14733711814004
MiddleEastern: Total:850, Accurate:460, False:44 -> %54.201977401129945
NorthernEuropean: Total:961, Accurate:583, False:9 -> %60.61173533083646
SouthAsian: Total:1064, Accurate:928, False:98 -> %87.25324309080655
SoutheastAsian: Total:1302, Accurate:1100, False:130 -> %84.48898103355602
WesternEuropean: Total:1082, Accurate:944, False:483 -> %87.23423923091144

TestSet:
Accurate: 704, False: 164 -> %81.10599078341014

African: Total:122, Accurate:108, False:36 -> %88.52459016393442
EastAsian: Total:164, Accurate:158, False:45 -> %96.34146341463415
MiddleEastern: Total:94, Accurate:51, False:1 -> %54.25531914893617
NorthernEuropean: Total:106, Accurate:61, False:0 -> %57.54716981132076
SouthAsian: Total:118, Accurate:101, False:12 -> %85.59322033898306
SoutheastAsian: Total:144, Accurate:125, False:16 -> %86.80555555555556
WesternEuropean: Total:120, Accurate:100, False:54 -> %83.33333333333334

From the results we can see that cross validation removed any possible bias during training therefore increased the test set accuracies. Training set accuracies only slightly increased.

F. Boosted Cross Validation Results

The boosted cross validation results are from adding weights during combination to the different models of cross validation from the previous model.

Accuracy Results:
TrainSet:
Accurate: 6395, False: 1445 -> %81.56540533520713

African: Total:1103, Accurate:966, False:295 -> %87.59724419831223
EastAsian: Total:1479, Accurate:1407, False:389 -> %95.12216104611149
MiddleEastern: Total:849, Accurate:464, False:43 -> %54.57425403484738
NorthernEuropean: Total:961, Accurate:583, False:8 -> %60.64035120917383
SouthAsian: Total:1064, Accurate:928, False:97 -> %87.26240940790373
SoutheastAsian: Total:1302, Accurate:1102, False:131 -> %84.6332164951333
WesternEuropean: Total:1082, Accurate:945, False:482 -> %87.36006992301745

TestSet:
Accurate: 704, False: 164 -> %81.10599078341014

African: Total:122, Accurate:108, False:36 -> %88.52459016393442
EastAsian: Total:164, Accurate:158, False:45 -> %96.34146341463415
MiddleEastern: Total:94, Accurate:51, False:1 -> %54.25531914893617
NorthernEuropean: Total:106, Accurate:61, False:0 -> %57.54716981132076
SouthAsian: Total:118, Accurate:101, False:12 -> %85.59322033898306
SoutheastAsian: Total:144, Accurate:125, False:16 -> %86.80555555555556
WesternEuropean: Total:120, Accurate:100, False:54 -> %83.33333333333334

The boosted results improved the training set results slightly but did not change the test set results. It gave results different than my expectations. I expected to get slight improvements on the test set as well.

G. Final Analysis on the Results

Excluding boosting method, all the methods imrpoved the accuracies as expected. The first methods fixed the class imbalance and the later ones improved the test set results. There is still a visible class imbalance for the 3rd and 4th classes. However, their accuracies significantly improved and around %55 after the last method. Meanwhile all the other classes have above %80 accuracies. The class imbalance fix worked best on the first class.

The previous percentages were calculated according to the formula. It showed the recall values: (Accurate/Total)*100

Now I will calculate the evaluation metrics.

Precision Predictions/ Classifications $\frac{\text{True Positive}}{\text{True Positive + False Positive}}$

Recall Predictions/ Classifications $\frac{\text{True Positive}}{\text{True Positive + False Negative}}$

F1 Predictions/ Classifications $\frac{2 * \text{True Positive}}{\text{True Positive + 0.5 (False Positive + False Negative)}}$

Final Metrics for each Class:

TrainSet:

Cuisine	Precision	Recall	F-Measure
African	%77	%88	%82
EastAsian	%78	%95	%86
MiddleEastern	%91	%55	%69
NorthernEuropean	%99	%61	%75
SouthAsian	%96	%87	%91
SoutheastAsian	%89	%85	%87
WesternEuropean	%66	%87	%75

TestSet:

Cuisine	Precision	Recall	F-Measure
African	%75	%89	%81
EastAsian	%78	%96	%86
MiddleEastern	%98	%54	%70
NorthernEuropean	%100	%58	%73
SouthAsian	%89	%86	%87
SoutheastAsian	%87	%87	%87
WesternEuropean	%65	%83	%73

4-) Resources

A. Dataset:
https://raw.githubusercontent.com/warcraft12321/HyperFoods/master/data/kaggle_and_nature/kaggle_and_nature.csv

B. Dataset Research:
https://towardsdatascience.com/recipe-cuisine-classification-278ea0837c94

C. Data Mining Techniques & Evaluation Metrics:
https://hanj.cs.illinois.edu/bk3/bk3_slidesindex.htm