Food Group Classifier

Predicting “animal-based” vs. “plant-based”

Team Group 1

**Group Members:**

Chris Ben

Angelica Semenec

Muhammad Soroya

**Project Type:**

Type B: Analytics Project

**Ch.1 Introduction**

Food contains many different chemical compounds that can vary drastically in ratio to one another. We wanted to create a machine learning model that can detect the significant differences between foods that are plant-based and foods that are animal-based. This model can be used to determine whether a food is plant-based or animal-based given their chemical makeup and nutritional information and the ratios of the different compounds.

The dataset is analyzed to find important as well as correlated features. This dataset is preprocessed in order to remove descriptive attributes as well as attributes that are highly correlated. Furthering preprocessing will be done, which is specific to each of the three different machine learning algorithms used to create predictive models, such as standardization, normalization and principal component analysis. The algorithms used in this project include the Random Forest Classifier, Guassian Naive Bayes, and K-Nearest Neighbors.

**Ch.2 System Design & Implementation details**

**Data engineering/preprocessing**

*Data Simplification*

The target variable was not readily available in the dataset and needed to be created. The foods were organized according to the listed food group number. Some food groups containing hundreds of instances that were more difficult to label were omitted. These included different food groups such as fats and oils, baby food, restaurant food, and other categories that were more difficult to classify as “plant-based” or “animal-based”, which included a lot of pre-processed products. These instances contributed to just under half of the dataset. Other categories such as cereals and grains were sorted through to remove some harder to classify instances.

*Missing Data Points*

For some columns, the food data was not input. The missing attribute instances seemed to correspond to the feature not being important to that particular food item or that food item not containing that specific compound. Because of this, the missing attribute instances were filled with 0.

*Removal of Attributes*

Descriptive attributes such as food group number, food description, and serving or weight descriptions are omitted from the dataset during preprocessing. The descriptive attributes were unique to each food, and so they did not create any effectively usable patterns and included the attributes *NDB\_No, Shrt\_Desc, GmWt\_Desc1, GmWt\_Desc2, Refuse\_Pct.*

A correlation map was generated to show the levels of correlation between the different attributes (See Appendix, Figure 2.1). This map showed that three instances contained a correlation of greater than 0.9 and revealed that the dataset contained three different instances of folate. These had nearly the same data for the different attribute columns. Two of the folate instances, *Food\_Folate\_(µg)* and *Folate\_DFE\_(µg)*, were removed while *Folate\_Tot\_(µg)* was left in the dataset.

**Additional Preprocessing (Algorithm-specific)**

*Standardization*

To effectively utilize Principle Component Analysis (PCA) with the Random Forest Classifier, a standardized dataset was created. This dataset instance was the input into the PCA functions. The standardized dataset was also used to create a predictive model without undergoing PCA.

*Principal Component Analysis (PCA)*

Principal Component Analysis was utilized with the Random Forest Classifier and the K-Nearest Neighbors algorithm and used the standardized dataset as input. Two variations of PCA were used. One variation created enough principal components to cover 95% of the variance within the dataset and the other variation created enough to cover 99% of the variance.

*Normalization*

Normalization is used for the Gaussian Naive Bayes algorithm. The algorithm assumes that the input dataset is normalized and so that was included in the preprocessing for this specific algorithm.

**Algorithm Considered/ Selected**

*Random Forest Classifier*

The Random Forest Classifier consists of an ensemble of decision trees. The algorithm reflects a decision-making process similar to how humans make decisions and can produce easy to visualize results, such as feature importance. Feature importance shows which features were most important in the model’s decision-making process, which can provide a good way to visualize which attributes had the greatest impact on the decision-making process. The Random Forest Classifier also creates models through the process of bagging, which is the creation of a dataset with replacement and can reduce the model’s sensitivity to the input data. This process creates models with a lower correlation to one another and the final decision is made by comparing the results of all of the individual trees in the forest and choosing what the classification is through majority voting and so this algorithm would provide a good method for building a food classifier model.

For this project, the random forest was populated with 4 different versions of the input dataset. Two different datasets were input using PCA, one set to capture 95% of the variance with 30 principal components and another set to capture 99% of the variance with 38 principle components. Another random forest model was created using the standardized input dataset and a fourth was created using a dataset with no additional preprocessing past the attribute removal step.

The number of trees in the forest (*n\_estimators*) was adjusted to improve the accuracy of the models. The models were initially trained using 100 trees and the accuracy for all 4 models was between 97% and 99%. The number of estimators was reduced to 10, which improved the accuracy of both the standardized set without PCA and the non-standardized set.

The maximum depth (*max\_depth*) of each tree was also tuned to improve accuracy and reduce overfitting. The optimal value for this was determined to be 9.

*Naive Bayes*

Naive Bayes is a fast classification algorithm. It predicts the probability of each data point belonging to a target class and then chooses the class with the highest probability as the predictive outcome. We chose it simply due to its naive characteristic, which means the model is based off of conditional probability, but treats each attribute as independent when modeling. We know that real world application of dataset attributes primarily interact with each other and should not be treated independently, yet the model still surprisingly effective.

When implementing Naive Bayes, we used the *GaussianNB()* model from *sk.learn.naive\_bayes*. We used Gaussian because our data is primarily in continuous values. After the normal preprocessing steps were applied, the dataset was normalized before being input to the Guassian Naive Bayes function. The hyper-parameters of the Guassian Naive Bayes algorithm were not adjusted during the building of the model.

*K-Nearest Neighbors*

K-Nearest Neighbors is one of the simplest, yet one of the most used machine learning algorithms. It has a large array of uses such as intrusion detection and data mining. The K-Nearest Neighbors algorithm falls under the classification category. The algorithm works better with datasets that don’t contain a lot of noise. To calculate the distance between a datapoint to the other points within the set and to determine the correct classification through majority voting of the nearest *k* neighbors. We use the Euclidean distance formula, distance =√((X-A)^2+(Y-B)^2). For this project we did not use a large dataset that would make the algorithm compute slow. This is important because K-Nearest Neighbors requires a lot of computational power to function.

The *n\_neighbors* parameter was adjusted to improve the accuracy of the dataset. The best accuracy was found when *k* was set to 3.

**Technologies & Tools Used**

Jupyter Notebook provides an easy to use interface and an excellent environment for data visualization. It was useful for cleaning and preprocessing through the display of dataframes and correlation mapping. It was always good for visualizing patterns in the data and models, such as feature importance.

Many different Python libraries were used for creating the machine learning models and visualizing the data. These libraries include Pandas, Numpy, Matplotlib, Seaborn, and Sklearn.

Excel provided an easy way to create our target attribute and modify the dataset before preprocessing with Python was executed.

Google was another very helpful tool. For all our problems with coding, we could solve them with the massive amounts of forums available in the web search.

Canvas allowed us to communicate with the class TA, Charles Patel, who also helped with debugging issues.

**Ch.3 Experiments / Proof of concept evaluation**

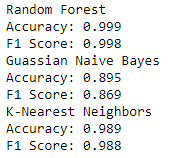
*Dataset*

The *Composition of Foods Raw, Processed, Prepared USDA National Nutrient Database for Standard Reference* contains 8,618 instances of food items with 53 different attributes. The food group number (*NDB\_No*) corresponds to one of many different food group categories such as Poultry Products, Beef Products, and Vegetables and Vegetable Products. The short description (*Shrt\_Desc*) corresponds to the name and description of a product such as what cuts of meat or how the food was prepared. The set also contained 3 more descriptive attributes and 49 numeric attributes that described the quantities of various compounds in the food such as protein, cholesterol, various vitamins and minerals, and others. (Preprocessing methods are discussed in Chapter 2).

*Methodology*

The dataset was split using the *train\_test\_split()* function after separating the source variables (the 49 remaining attributes) and the target variable (*Based*). The *train\_test\_split()* function was used twice to create three different sets of data: training, testing, and validation. For each of the splits, the random\_state was set to 0 and the split percentages corresponded to 25% for the testing set, 18.75% for validation, and 56.25% for the training data of the overall dataset.

*Graphs/ Diagrams*



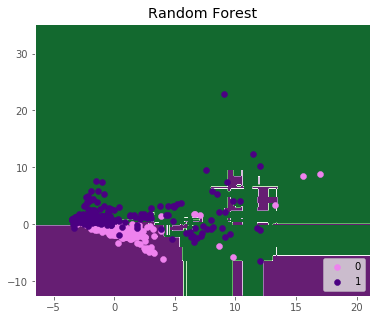
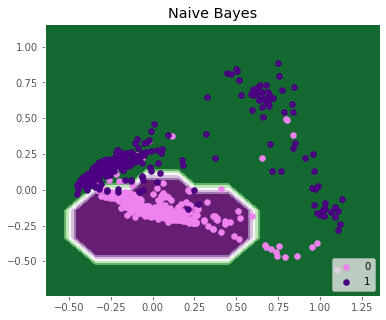
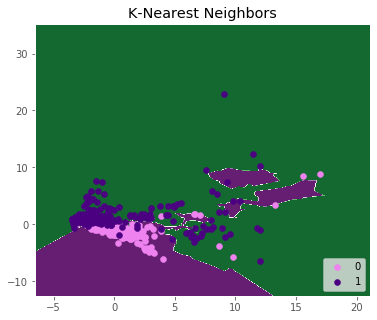
*Figure 1.1: The accuracy and F1 score of each algorithm’s model that scored the highest overall accuracy.*

Four different variations of the Random Forest model were generated using data that was preprocessed different. The model that scored the highest accuracy on the validation set included only standardization of the dataset in addition to the general preprocessing methods.

The K-Nearest Neighbor algorithm of the four generated that scored the highest accuracy also utilized standardization for its preprocessing method.

The Guassian Naive Bayes model that scored the highest, as expected, utilized a normalized dataset.

Overall, the model generated by the Random Forest Classifier scored the highest accuracy as well as F1 score as shown in Figure 1.1.



*Figure 1.2: Visualization of the most accurate model variations.*

The above diagrams show the decision boundaries generated by the different algorithms. The number 0 corresponds to foods that are animal-based and the number 1 corresponds to foods that are plant-based. The models were generated using PCA in order to reduce the dimensions down to 2. The Guassian Naive Bayes shows that it was not able to capture many of the points due to its radial decision boundary. Both the K-Nearest Neighbors and Random Forest algorithm are able to generalize the datasets well without overfitting, although both models show several data points that were not classified correctly.

*Analysis of results*

The Random Forest classifier resulted in the best model. The model that was created using a standardized dataset resulted in the highest overall accuracy of 99.9% compared to the other algorithms as well as the highest F1 score. While the Random Forest model did the best, the K-Nearest Neighbors algorithm also produced good results with 98.8% accuracy. The Guassian Naive Bayes algorithm resulted in the lowest accuracy at 89.5%, which is approximately 10% different from the top-scoring algorithms.

The visualization of the algorithms in Figure 1.2 also reflect how well the different algorithms were able to generalize the data. For the Guassian Naive Bayes, the decision boundaries radiate from a central point, and so they were not able to capture data points that did not follow some sort of a radial trend. Random Forests and K-Nearest Neighbors had decision boundaries that did not stem from a central point and so were able to generalize points that were not part of the central group.

**Ch.4 Discussion & Conclusions**

*Decisions made*

* How to edit down the input dataset
* Choosing initial algorithms
* Feature removal
  + Based on relevance
  + Based on correlation mapping

*Difficulties faced*

* Creating the target variable
* Tuning hyper-parameters
* Debugging errors
* Learning dataframe syntax

*Things that worked*

* Standardizing for Random Forests
* Creating a correlation map for attribute removal

*Things that didn’t work well*

* PCA for Random Forests

*Conclusion*

The three machine learning algorithms that were chosen were implemented successfully and produced satisfactory results. Using Sklearn and other supporting libraries, we were able to easily implement random Forest, Guassian Naive Bayes, and K-Nearest Neighbors.

The feature importances of the Random Forest model with the highest accuracy were also visualized to show interesting patterns in the food data. Figure 2.2 of the Appendix shows the attributes that had the greatest impact in determining whether a food was plant-based or animal-based. The two most important attributes were protein and cholesterol, which had a feature importance of approximately 0.4.

Random Forests yielded the best results, however K-Nearest Neighbor also produced great results. Gaussian Naive Bayes did not perform as well as its counterparts. As it yielded the worst results of the three because its radial nature was not able to generalize the data that did not fit into that pattern.

**Ch.5 Project Plan / Task Distribution**

*Who was assigned to what task*

Chris assigned to create models using K-Nearest Neighbors.

Angelica assigned to create models using Random Forest Classifier and conduct preprocessing on the dataset such as attribute removal and filling in missing data points.

Muhammad assigned to create models using Naive Bayes and conduct preprocessing on the dataset such as data simplification and creation of the target variable.

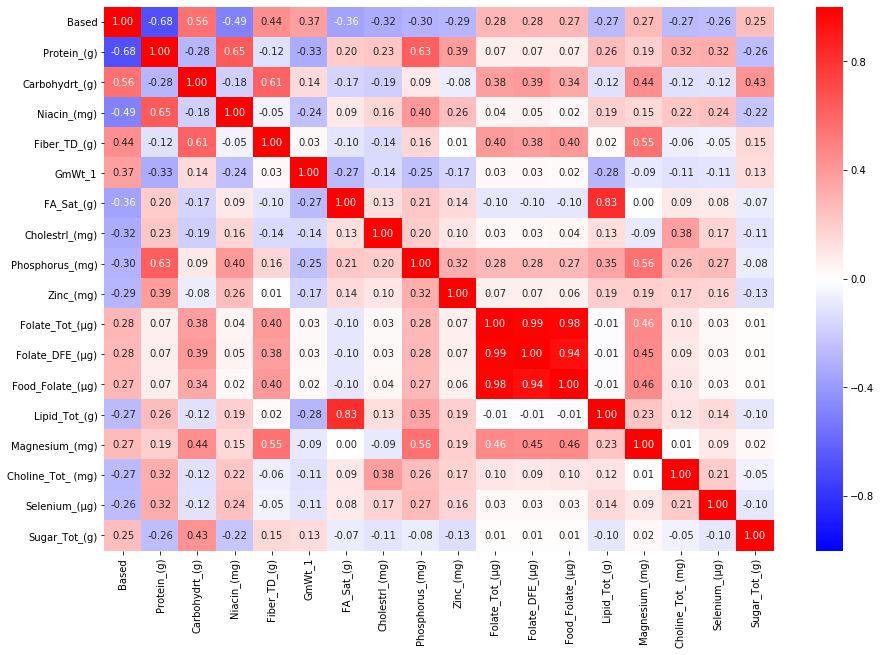
*Who ended up doing what task (justify as applicable)*

Chris implemented the K-Nearest Neighbors algorithm and data visualization for the model as well as documentation.

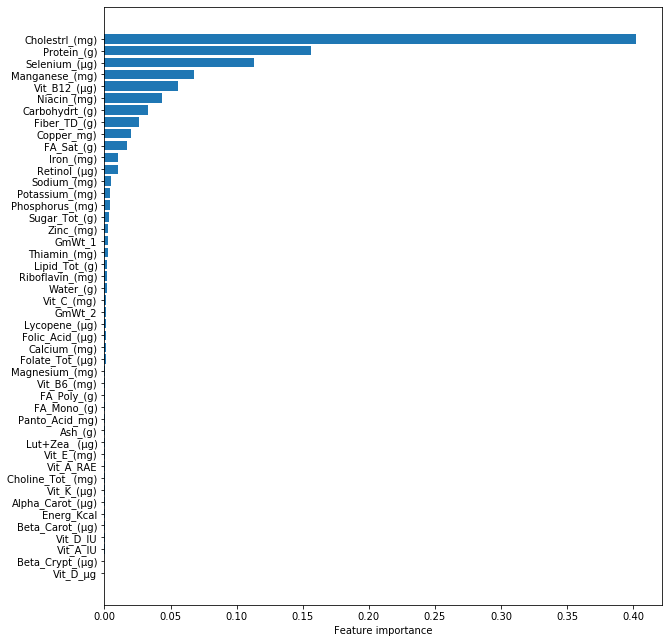
Angelica did preprocessing including attribute removal, correlation mapping, PCA, and standardization. Angelica also implemented the Random Forest Classifier algorithm and data visualization for the model as well as documentation. Angelica also did Notebook organization for readability.

Muhammad did preprocessing in Excel to edit food groups for data simplification and created the target variable as well as contributed to the preprocessing step of attribute removal through correlation mapping. Muhammad also implemented Guassian Naive Bayes and contributed to the documentation.

**Appendix**



*Figure 2.1: The above figure shows features with a correlation of greater than 0.25 with the target class,* Based*. The red square in the middle of the diagram shows three highly correlated attributes (3 variations of folate)*.



*Figure 2.2: Feature importance generated by the Random Forest model with the highest accuracy. The attributes that had the greatest impact on the decision-making process were cholesterol, protein and selenium.*