# Using Food Science for Data Science Instruction

Sudheeksha Garg
Department of Computer Science
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, NY 14586
sg3932@cs.rit.edu

***Abstract*—Using difficult topics when teaching machine learning can induce anxiety in students and cause them to mentally distant from the subject matter resulting in a lack of teaching effectiveness. Topics such as cancer, pandemics, mortality can distract the listener from the key points. Using familiar examples such as food, the students can learn the methodology without becoming distracted because it is more relatable. This paper demonstrates methods of teaching machine learning using examples drawn from food and cooking.**

***Index Terms*—Machine Learning; Data Science; Food science**

## I. Introduction

Data science is used to extract insights, information, or knowledge out of the data. It can be used to create recommender systems, speech recognition systems, fraud and risk detection, etc. It has become an essential feature of every organization because it helps business leaders to make decisions based on facts, statistical numbers, and trends. It is evolving fast as the size of data grows bigger each day.

Understanding data science is not easy as a statistical and mathematical approach to teaching data science can be too abstract and often is hard to understand. Also, topics such as pandemics, biological processes such as cancer, or climate change that are traditionally used to teach data science can be grim and cause communication and learning blocks in students by making them mentally distracted. We hypothesize that using topics such as food can make data science more approachable and easier to understand.

Data science considers topics that can save lives. These include mortality rates, demographics, climate change, and other grim topics. However, young impressionable students in Data Science classes have a difficult time talking about these topics.

To avoid these topics, we instead turn to the familiar subject matter. Here the focus is on food. Instead of talking about people expiring, we talk about fresh food rotting or going bad. Instead of talking about the probability of people contracting COVID-19 (Coronavirus) from each other, we can instead discuss the topic of the chances of popcorn popping in a microwave. Instead of discussing using classifiers to distinguish malignant cancer from benign cancer, we can discuss classifiers to tell cupcakes from muffins. By using food, we avoid difficult topics and focus on familiar and happy topics.

The reason food is a good topic to teach data science is that it is something that everyone understands, and it does not cause communication blocks. Another good reason to use food is that in the United States, it is a 5.75 trillion-dollar industry [1] that uses Data Science in different sectors such as food supply chain and production.

The paper is structured as follows: In the first section, traditional topics and the issues involving them are discussed. In the next section, related work is presented. In the following section, the results of a survey are analyzed to show that there is a problem. A subset of the student population is identified who are uncomfortable talking about traditional topics. Section four presents machine learning algorithms using food science and provides an alternative method of teaching data science. Finally, Section five consists of closing discussion and conclusions. We have shown that difficult topics that might "turn off" students can be avoided by talking about food instead.

## II. Background

Traditionally, machine learning has been used for cancer diagnoses, climate change, classifying a virus, and other grim topics. For example, here is a paper [2] on a Comparison of Machine Learning algorithms on breast cancer diagnosis. They have applied popular machine learning techniques to classify Wisconsin Breast Cancer dataset [3], and the classification performance of methods was compared with each other using accuracy, precision, recall, and ROC Area. The Support Vector Machines obtained the best performance with the highest accuracy.

An example of climate change in Machine learning is this paper [4]. The authors of this paper derive a hierarchical cohort of high-risk senior citizens, outpatients, clusters, and proactively deliver population health insights and timely health services through alerting governmental and non-governmental agencies. This system is developed using Apriori, Naive Bayesian, and artificial intelligence algorithms.

In this paper [5], the authors used artificial neural networks for the prediction of important influenza virus antigenic types and the hosts, to create a computational system for influenza surveillance. They found that the accuracy of classification can vary based on the host and gene segment. Compared to virus such avian and swine flu, human viruses can be classified with a higher accuracy. This indicates that human viruses have adapted to their human hosts. All of these results show that Data Science can save lives. While these topics

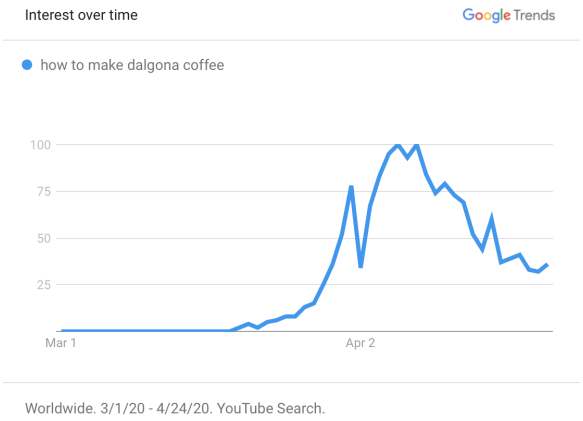


Fig. 1. Interest in Dalgona Coffee. This shows an overall surge and decay of Dalgona coffee. Image was generated on Google Trends [6].

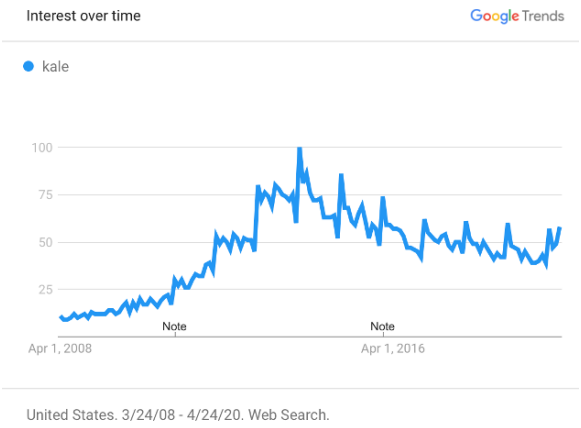


Fig. 2. Rise and Fall of interest in Kale over time in the United States. Image was generated on Google Trends [6].

are incredibly important, it can be difficult for a student in a learning environment.

Due to the current pandemic, there is a lot to be learned from the spread of viruses such as modeling viral spreading, exponential growth, and geometric decay. However, obviously, many people find it disturbing to talk about the virus because of the mortality rate associated with it. Instead of talking about a virus such as COVID-19, we can talk about food trends or fads. Fads can behave like a virus. They have a quick rise and steep decline in a short amount of time. Here are two or three food fads that demonstrate this. The following examples are based on the presumption that when someone searches on a topic that they are interested in it.

An example of food fad is Dalgona coffee induced by the isolation during COVID-19. As per Youtube searches on Google Trends [6], *"How to make Dalgona coffee"* has become the most searched type of coffee recipe worldwide. As shown in Figure 1, searches worldwide for Dalgona coffee recipe peaked in mid-April and declined by the end of April.

Another example of a food fad was the increase in popularity of Kale. It was coined as "superfood" and is found now as chips or in salads. From Figure 2, it can be seen that Kale became very popular in the United States in mid-2014 when young-adults where being drawn to a more health-conscious diet. It is now slowly declining, and its popularity is reduced by half since 2014.

A traditional Japanese tea, Matcha, has been gaining a lot of popularity lately as well. Matcha has become attractive to millennials because of green color of the drink being aesthetically pleasing as well the health benefits such as weight loss. In Figure 3, it can be seen how the popularity of Matcha has been increasing over the last few years, even though it has been around since the 7th Century.

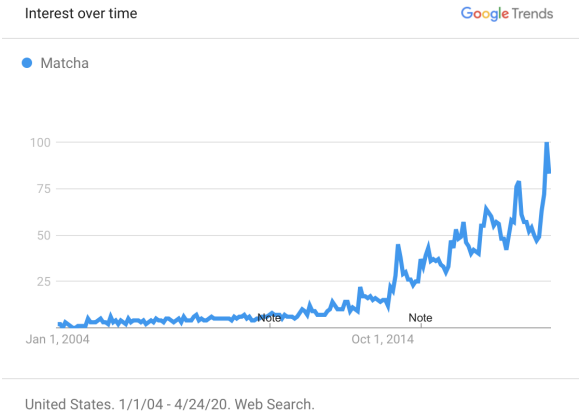


Fig. 3. This is growth and interest in Matcha tea. Additionally, we can see another cycle is superimposed. Image was generated on Google Trends [6].

## III. Related Work

A study that is most closely related to the work done in this paper [7] is using the UCI machine learning repository's wine dataset to understand what makes wine taste good. The author of this paper aims to make his readers understand why data science is important and what its applications are. After plotting a heat map to find features that are related to each other, plots are made with those features to spot any trends. A pattern that was found was people buy wines with more alcohol content is found.

Food computing [8] can help resolve food related health problems, gatronomy, and biology by using machine learning and data science to perform food analysis. The food analysis is performed on data collected from sources such as food blogs and books, social networks or IoT.

Food pairing [9] is a hybrid method of using consumer behavior and scientific analysis. The approach developed by the author is named Consumer Flavor Intelligence (CFI). The purpose of food pairing is to ensure that the right flavors come to the market at the right time. It can be used by Chefs, food bloggers, and other people in the food industry to create recipes. Based on consumer data, the authors of the paper discovered that consumers are not loyal to brands. The millennial consumer is open to explore food pairing that is new and has not been done before. First, they mine the food data online to find out to current trends in food. Then its aromas

TABLE I
RESULT OF K MEANS IN WEKA

| Attribute | Cluster 0 | Cluster 1 | Cluster 2 |
|---|---|---|---|
| Food over Biological Processes | 0 | 0.9 | 1 |
| Food over Pandemics or Epidemics | 0.03 | 1 | 0 |

and measures are used to find compatibility with other food items. Then perform a complete brand analysis.

Another approach to the previous work is food market analysis. Here, trends in the food and drink market are found by mining data. Based on this information, restaurants can decide which items to add to their menu, and stores can determine what food items to stock up on, and which seeds the farmers should plant.

Food and supply chain optimization is a vital part, and machine learning algorithms can make forecasts as to pricing and inventory, which preserves extra costs. For example, in food manufacturing it can help track the entire process. From when the product is grown, to processed to manufactured. It can help in predicting the cost of the product.

Data Science in the Food Industry can be extremely beneficial. It can help one stay ahead of the competition. There are many companies today that provide meal kits. These meal kits contain all ingredients such as vegetables, spices, and set of instructions to create a meal. These meals are catered to a particular consumer based on their dietary restrictions and previously liked meals.

## IV. SURVEY

To validate our belief, we conducted a survey in which we asked the students various questions about their topic preferences when learning Data Science. They were questioned whether they would prefer learning about food topics or cancer, food or pandemics, and food or climate change.

Using Weka, K means was performed on data collected, and on analysis, three clusters were found. From Table I, we can see the information generated by Weka. If the value in the table is zero or closer to zero, it indicates that the preference is not food. If the value is closer to 1 or 1, the preference is food. It shows that there are 43% of students in the class who are comfortable talking about grim topics as data science can be used to save lives. Also, it can be seen that 26% of the students do not want to talk about morbid topics and it can cause communication and learning blocks in these students. The rest of the students were neural in their preference.

The above information tells us that there is a mixture model of students in the class. While some have a morbid curiosity, others prefer not to talk about grim topics. Also, from Figure 4, we can observe that most students like food. In order to include the 26% of the students and promote learning without any communication block, data science should be taught with topics such as food. Thus, proving our original hypothesis.

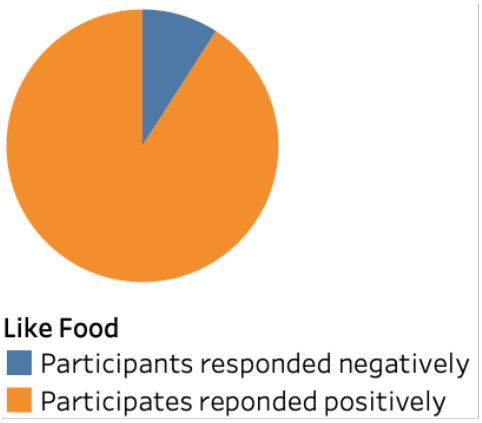


Fig. 4. Pie chart depicting the number of people who like food

## V. EXAMPLE MACHINE LEARNING ALGORITHMS USING FOOD

### A. Background on Gradient Descent

Gradient Descent is an iterative optimization algorithm that takes more significant steps where the amount of gradient is higher or moves in the direction of steepest descent. It is used to find the parameters of a model in machine learning algorithms such as Logistic Regression, SVM, etc. An analogy for gradient descent would be a ball rolling down a hill. The ball would initially roll down faster till it slows down when it is about to reach a dip similar to the one shown in Figure 5. The goal of gradient descent is finding this *dip*, known as the local minimum. The first step is to initialize a starting point. Then steps are taken in path of the greatest descent. The rate at which these *steps* are taken is called the learning rate.
The two parameters required are -

- Learning Rate: It is used to control the change in each coefficient. It should not be too high as it is possible to jump over the bottom, and if it too low, then the algorithm takes a longer time to run.
- Epochs: The number of iterations that need to be performed. This amount can vary from dataset to dataset.

The Gradient Descent calculated also depends on the cost function. The cost function tells how *good* the model is. The two parameters of the cost function are weight and bias. The slope of the cost function informs on how to update the parameters. Notice that the shape of the bowl in Figure 5, it shows that gradient descent is a convex function.

Similar to Gradient Descent, we can use as Grid Search as well to find parameters. In a grid search, a rough set of all parameters is tried. Then, once a global best fit is found, the parameters in that region are iteratively changed to improve the fit to the local model, and the precision of the model. Figure 6 a SIR (Susceptible-Infected-and-Recovered) model [10] is used to predict Dangola coffee.

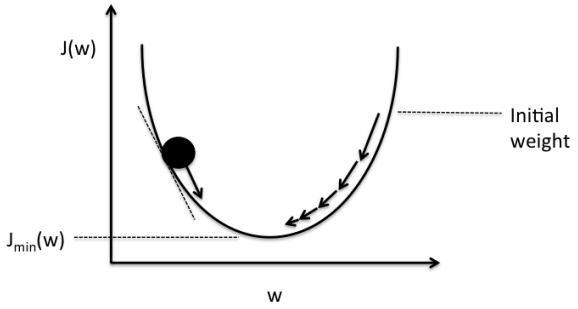


Fig. 5. Schematic of Gradient Descent

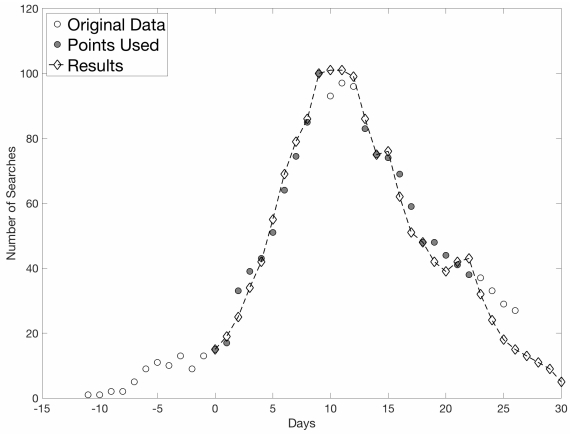


Fig. 6. Grid Search [10]

### *B. Logistic Regression*

Before getting into Logistic Regression, it is vital to understand the concept of Odds and Odds Ratio. Odds are the ratio of success to the ratio of failure. The range of odds can be in the range of 0 to $\infty$. If we take a natural logarithm of such numbers, for a number $x \geq 0$, $\log(x)$ is in the range of $[-\infty, \infty]$. Odds ratios, as the name suggests, is a ratio of odds. The odds ratio can vary between 0 to positive infinity, $\log(OddsRatio)$ can vary between $[-\infty, \infty]$. Specifically, when the odds ratio lies between $[0, 1]$, $\log(OddsRatio)$ is negative.

The term logistic regression is confusing as it is neither logical nor a typical regression. Logistic regression is a statistical classification method for predicting binary cases, i.e., the target class is either 1 or 0. To explain it better, we are using data on popcorn to predict two scenarios: if the popcorn is cooked or not, and if the popcorn is burnt or not. If the popcorn is cooked, then the target class is 1, and if uncooked, it is 0. Since the values predicted by logistic regression are binary, they are mainly used for classification purposes. However, the output of logistic regression, is the probability of the prediction i.e values are in the range of 0 to 1. This is used to find how similar the predicted value is to the expected value. It is assumed that all values above 0.5 belong to Class 1, and all values below 0.5 belong to Class 0. Here, we are trying to

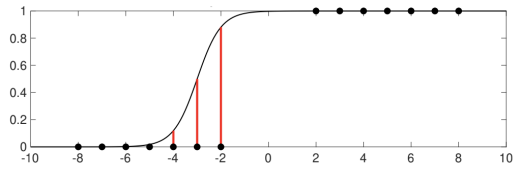

Fig. 7. An example of Sigmoid function. The vertical red lines show the error between the function and the data.

classify for a given time in seconds, what is the probability that the popcorn is cooked. Suppose value for x1 value is probable to 0.8, then there is an 80% probability that it is cooked.

As the values that need to be predicted by logistic regression should be in the range of 0 to 1, we use sigmoid function. Here $w$ is the weight, and $b$ is a bias.

$$h_{w,b}(x) = \frac{1}{1 + e^{-(w^\top . x + b)}}$$

The sigmoid function can be used to create a sigmoidal curve, as shown in Figure 7. By examining the curve in Figure 7, we can see that the error would be decreased if the sigmoid function was shifted further to the right. The problem with machine learning is to figure out how to get a computer to examine this situation and then have the algorithm decide to shift the curve to the right. The secret is to have the algorithm consider the amount of error and the slope of the sigmoid function at which it occurs. In this case, since the error occurs where the sigmoid function is going up, the curve should be shifted to the right such that error lines intersect sigmoid at a lower value.

The cost function of logistic regression is $Cost(h_\theta(x), y) = -y\ log(h_\theta(x)) - (1 - y) log(1 - h_\theta(x))$ [11]

It is a convex function. The right and left part of the cost function join together to form a bowl shape, like the one in Figure 5.

$$Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta x), & \text{if y=1.} \\ -\log(1 - h_\theta x), & \text{if y = 0.} \end{cases} \quad [11]$$

The intention is to minimize the cost function. In order to achieve this, we repeatedly update the weights and bias using gradient descent until the global minimum is reached. In other words, to fix the error in Figure 7, we can use gradient descent.

Every iteration of logistic regression is used to update the coefficients. In each iteration, the error is found between the predicted value and expected value. Based on this error and the learning rate, the coefficients are updated. Eventually, the error lowers to an insignificant amount. Once the lowest cost is reached, the final coefficients can be used to make predictions. The above algorithm was written for the popcorn data, and it generated an accuracy of 0.83.

Figure 8 shows the result of logistic regression on popcorn data. The blue sigmoidal curve is for the cooked data, and the orange is for burnt data. Two logistic regression curves can be

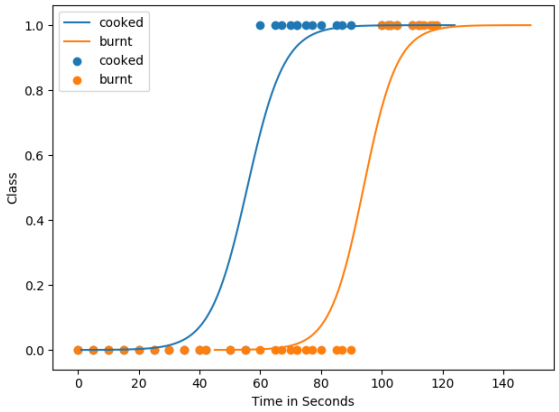


Fig. 8. Logistic Regression is to classify if Popcorn is cooked or not.

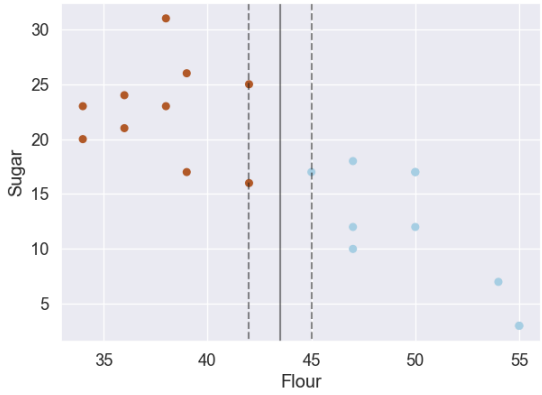


Fig. 9. Support Vector Machines used to classify if given recipe is a Muffin or Cupcake. The bold line represents the Hyperplane.

used to give a clever function. This is what artificial neurons do. A delta function can be drawn between the two curves. Using the delta function, any curve can be generated. This is why artificial neural networks can be used as a universal approximator.

### C. Support Vector Machines

Support Vector Machines are a supervised machine learning algorithm that is used to make binary classification. The SVM algorithm tries to find a decision boundary using support vectors that separates the data such that on the side lies one category, and on the other side lies the other category. These decision boundaries are called Hyperplanes. There can be multiple hyperplanes that separate the data, but the ideal one should satisfy the following conditions:

- It creates divide between the classes with a maximum margin.
- Its equation generates a value greater than 1.

Hyperplanes ensures that future data is classified with more confidence. The data points that are closest to the hyperplane are support vectors. They can change the orientation or maximize the margin between the hyperplane and the data points. The hyperplane can be found by finding ideal parameters, *weight* and *bias* by minimizing the cost function. The cost function [12] is

$$J(w) = \frac{1}{2}||w||^2 + C\left[\frac{1}{N}\sum_{i}^{n} max(0, 1 - y_i * (w.x_i + b))\right]$$

We can minimize the cost function by minimizing $||w||^2$, which minimizes the margin, or we can minimize the sum of hinge loss [12] $max(0, 1–y_i * (w.x_i + b)$ . We can minimize either of the two by using gradient descent.

Consider the graph in Figure 9; here SVM is used to determine if a given recipe is that of a cupcake or a muffin. The features used are the amount of Sugar and Flour in a recipe. The prediction is labeled as 0 and 1 for Muffin and Cupcake respectively. The accuracy of the model resulted was 1.0. And the ideal hyperplane is the one shown in Figure 9.

### D. K means

K means is a clustering algorithm; a clustering algorithm divides the data into subgroups where a point in a subgroup is similar to other points in that subgroup. This similarity is found using distance metrics such as Euclidean distance. Unlike the supervised methods mentioned above, clustering is unsupervised. K means is an iterative method that divides the data into *k* pre-defined cluster/subgroups, where each point in data belongs to exactly one cluster. Each cluster has a center called centroid: it is the average of all the points in the cluster. Once centroid is initialized, points are assigned to each cluster such that the sum of squared distance between all points in that cluster and center is at a minimum. The center is re-calculated, and this process is repeated iteratively till the center of the clusters stops changing. The algorithm in steps:

1) Specify *k*.
2) Initialize the centroids and assign points to the closest cluster.
3) Keep iterating till the center of the cluster stops changing, or a seed limit is reached.
4) Compute the sum of squared distance between the center and all data points.
5) Assign points to the closest center.
6) Compute the center.

It is essential to select the correct *k* to get homogeneous clusters. A *knee* graph (some call it an elbow curve) can be plotted to find the ideal *k*. The idea is to find the sum of squared error for a range of k values and then plot the sum of squared error for each value of k. The ideal *k* is point right after which there is a steady, almost linear decrease in the graph. Observe, Figure 10, notice that six is ideal *k*. Here, K means was used to do market basket analysis. The data consist of different shoppers that buy items from a store. The goal is to find the food groups that the shoppers belong to, such as keto, vegan, gluten-free, etc. Based on Figure 10, we see six groups emerge.

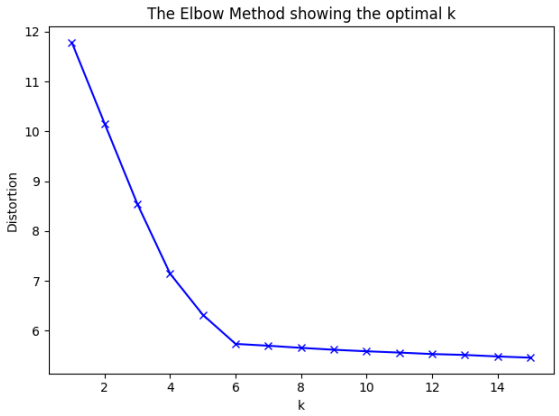


Fig. 10. Inflection point detection used to find optimal value of $k$. The inflection point is the point of diminishing returns. The optimal $k$ is 6 for Market Basket Analysis.

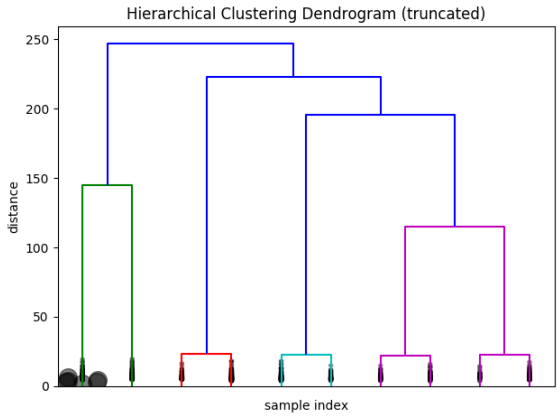


Fig. 11. The dendrogram for Market Basket Analysis, truncated to show the last 10 merges. The lower leftmost merges are difficult to discern.

### *E. Agglomeration*

Agglomeration or agglomerative clustering is a bottom-up approach to hierarchical clustering. Here, each point in the data is considered as an individual cluster and merged until there is only one cluster. One of the key decisions is to choose a linkage. It is required to determine distance between the points. Different types of linkage are the ward's method, complete linkage, single linkage, and average linkage. Another decision is to select the metric to calculate the linkage. It is usually a Euclidean distance. Agglomeration algorithm is as follows:

1) Compute the distance between each data point.
2) Each data point is its own cluster at the start.
3) Merge two closest clusters and re-compute the distance.
4) Repeat the previous step till all the clusters are merged into one.

Dendrograms help understand agglomeration in a much better and more visual way. It shows in a static way how the aggregations are performed or how each cluster is joined together until there is only cluster left. The height of the dendrogram indicates the order in which the clusters were joined, and it can also be used to determine how far apart each cluster is. The more the height before joining, the further apart they are. Dendrograms can also be used to find the number of clusters by cutting it by drawing a horizontal line across it. It is important to know where to cut the line. The line must be cut where the difference is most significant, or clusters are far apart—this line when drawn across dendrogram cut lines in the dendrogram. The number of lines it cuts is the number of clusters.

When deciding between K means or Hierarchical clustering, usually, Hierarchical Clustering is preferable. It is because Hierarchical clustering has fewer hidden assumptions about the distribution of data. With K means clustering, the desired number of clusters should be known. Also, k-means will often give unintuitive results if the data is not well-separated into sphere-like clusters, $k$ picked is not well-suited to the shape of the data or initial values for the cluster centroids are weird.

The only requirement (which k-means also shares) of hierarchical clustering is that a distance has to be calculated between each pair of data points. Hierarchical clustering typically joins nearby points into a cluster, and then successively adds nearby points to the nearest group. A dendrogram can be used to decide how many clusters your data has, by cutting the dendrogram at different heights. Of course, it has to be predecided on how many clusters one wants.

### *F. Artificial Neural Networks*

Artificial Neural Networks is a computational model that has a similar structure of the neural network of a human brain. Like in the human brain, there is a neuron called a node. There can be multiple layers in the network. The node in one layer is connected to all the nodes in the subsequent layer. The signal in this network flows from left to right. The node, when given an input, implements a function and forwards the output to the next layer. Each input is given a weight. This weight depends on the significance of the input. The resulting output is generated after all the nodes complete the above process. There are various kinds of neural networks, but the most commonly used ones are:

1) Feedforward
2) Recurrent

Artificial Neural Network is used to find the classification between muffins and cupcakes. It similarly classifies the data as Support Vector Machines but takes slightly longer. This shows that we should not use ANN unless it is required, and it better to go with the more straightforward option of SVM (Occam's Razor).

## VI. Conclusion

We investigated the hypothesis that students would not want to discuss grim topics. In order to do that, we conducted an experiment where the students answered questions based on their topic preferences. K means was performed on the data collected from the experiment. We identified that there were three clusters of students with different preferences. As expected the first cluster did not want to talk about sickness, mortality, extinction, or anything grim. However, this cluster comprised only 26% of the students. A second cluster of students either had morbid curiosity, or who want to use data science to save lives. This consisted of 46% of students. The remaining cluster of student were neutral.

In this paper, we have shown five examples demonstrating the use of food to teach machine learning. These include commonly used classification and clustering algorithms. The classification methods include: Logistic Regression, Support Vector Machines,Artificial Neural Network. The clustering techniques demonstrated are K means, Agglomeration.

We used logistic regression to classify how long it takes for a bag of popcorn to get cooked. Support Vector Machines are used to classify a given recipe as a cupcake recipe or a muffin recipe. For comparison purposes, Artificial Neural Networks were used on the same cupcake recipe or muffin recipe task. This helps students compare and contrast two algorithms. For finding if a recipe is a muffin or cupcake, it is better to use support vector machines as they are easier to implement and generate the same classification as an Artificial Neural Network. Additionally, Support Vector Machines run much faster.

To demonstrate K means and Agglomeration, we used market basket analysis to identify subgroups of shoppers in a grocery store. This helps in recommending products to a particular subset of shoppers. This type of clustering algorithms can be used to create recommender algorithms that help in issuing coupons to shoppers.

Overall, we found a cluster of students who did not mind discussing unpleasant topics (43%), and we identified a cluster of around one quarter of students who prefer learning data science while avoiding unpleasant topics (26%). In order to help the latter students understand machine learning better, we have given recommended approaches using pleasant topics. These example given include the use of food, cooking, recipes, and shopping. Using these techniques, data science can be taught to the full spectrum of students while avoiding communication and learning blocks. At the start of this project, there was no indication that there would be a Coronovirus pandemic. The mortality of the pandemic further justifies the need to avoid communication blocks.

## Acknowledgment

I would like to thank Dr. Thomas B. Kinsman for taking me on as a Capstone student. This work above and beyond his usual plan of work. His continuous support and guidance, lead to the successful completion of this project. His knowledge and constructive feedback helped in gaining a bigger perspective and better results. Few other Professors would have co-authored a paper [10] with me. His dedication and determination to improving his students was evident at every step of the way.

I would also like to thank the Computer Science department for providing the academic resources I required to complete this project.

## References

[1] E. Duncan, "Topic: Food retail industry." [Online]. Available: https://www.statista.com/topics/1660/food-retail/

[2] E. A. Bayrak and P. Kırcı and T. Ensari, "Comparison of Machine Learning Methods for Breast Cancer Diagnosis," in *2019 Scientific Meeting on Electrical-Electronics Biomedical Engineering and Computer Science (EBBT)*, 2019, pp. 1–3.

[3] [Online]. Available: http://archive.ics.uci.edu/ml/datasets/breast cancer wisconsin (diagnostic)

[4] Wenyu Zhang, Xuan Liu, Wei Xiao, and Dezhong Chi, "Software design of sand-dust strom warning system based on grey correlation analysis and particle swarm optimization support vector machine," in *2009 2nd International Conference on Power Electronics and Intelligent Transportation System (PEITS)*, vol. 2, 2009, pp. 47–50.

[5] P. K. Attaluri, Z. Chen, and G. Lu, "Applying neural networks to classify influenza virus antigenic types and hosts," in *2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 2010, pp. 1–6.

[6] [Online]. Available: https://trends.google.com/trends/?geo=US

[7] A. Hariharan, "How to use data science to understand what makes wine taste good," Aug 2019. [Online]. Available: https://www.freecodecamp.org/news/using-data-science-to-understand-what-makes-wine-taste-good-669b496c67ee/what-is-machine-learning

[8] W. Min, S. Jiang, L. Liu, Y. Rui, and R. Jain, "A survey on food computing," *ACM Computing Surveys*, vol. 52, no. 5, p. 1–36, 2019.

[9] "Discover exciting pairings." [Online]. Available: https://www.foodpairing.com/en/home

[10] Garg and Kinsman, *Simulating the Exponential Rise and Fall of Fads and Rumors*. [Online]. Available: http://www.cs.rit.edu/ tbk

[11] T. V. Ganesh, "logistic regression," Mar 2019. [Online]. Available: https://gigadom.in/tag/logistic-regression/

[12] R. Zha, "5," Jun 2019. [Online]. Available: https://zharuosi.github.io/blog/2018/06/09/machine-learning-5/

[13] D. Sahoo, W. Hao, S. Ke, W. Xiongwei, H. Le, P. Achananuparp, E.-P. Lim, and S. C. H. Hoi, "Foodai," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining - KDD 19*, 2019.

[14] R. Chuprina, "Machine learning and ai in food industry: Solutions and potential." [Online]. Available: https://www.datasciencecentral.com/profiles/blogs/machine-learning-and-ai-in-food-industry-solutions-and-potential

[15] G. Van Rossum and F. L. Drake Jr, *Python tutorial*. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.

[16] T. pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3509134

[17] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[18] T. E. Oliphant, *A guide to NumPy*. Trelgol Publishing USA, 2006, vol. 1.
[19] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing In Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
[20] F. Chollet *et al.*, "Keras," https://keras.io, 2015.
[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

## APPENDIX

All the code in implemented in Python 3.6 [15] and using packages [16], SciPy [17], NumPy [18], Matplotlib [19], Keras [20], and Scikit learn [21].

### *A. Code for Logistic Regression*

```
class LogisticRegression:
    def __init__(self, b0 = 0, b1 = 0,
    L= 0.01, num_iter=100000):
        self.b0 = b0
        self.b1 = b1
        self.L = L
        self.epochs = num_iter

    def get_loss(self, X, Y):
        return (np.sum(np.power((Y - self
            .predict(X)),2)))

    def predict(self, X):
        return np.array([1 / (1 +
        exp(-1 * self.b0 + -1 *
        self.b1 * x)) for x in X])

    def logistic_regression(self,
    X, Y):

        costs = [], loss = []
        size = X.shape[0]

        # while self.get_loss(X,Y) >
            0.35:
        for epoch in range(self.epochs):
            y_pred = self.predict(X)
            cost = (-1/size)*(np.sum((Y.T
                * np.log(y_pred)) + ((1-Y
                .T)*(np.log(1-y_pred))
            D_b0 = -2 * sum((Y - y_pred)
            * y_pred * (1 - y_pred))
            # Derivative of loss wrt b0
            D_b1 = -2 * sum(X * (Y -
            y_pred) * y_pred * (1 -
            y_pred))

            # Update b0 and b1
            self.b0 = self.b0 - self.L *
                D_b0
            self.b1 = self.b1 - self.L *
                D_b1
```

### *B. Code for SVM*

```
def svm(features, outputs):
    """
    SVM using gradient descent
    :param features: The features Sugar
        and Flour
    :param outputs: Tells if its a
        cupcake or muffin in 1 or -1
    :return: The weights calculated
    """
    count = 0
    prev_cost = inf
    cost_thres = 0.1

    # gradient descent begins
    for indx in range(1, max_iter):

        for index, x in enumerate(
            features):

            gradient = calculate_gradient
                (weights, x, output[index
                ])

            # calculate the new weight
            weights = weights - (l_r *
                gradient)

        if indx == 2 ** count:

            # in case we reach the local
                minimum before
            # end of iterations
            if prev_cost - cost <
                cost_thre * prev_cost:
                return weights

            prev_cost = cost
            count += 1

    return weights
```

*C. Code for K means*

```
from sklearn.cluster import KMeans
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_csv('SHOPPING_CART_v892.csv'
    )
df = df.drop(['ID'], axis=1)
print(df.head())
kmeans = KMeans(n_clusters=6).fit(df)

# Plot Knee Graph
K = range(1,16)
for i in range K:
    kmean = KMeans(n_clusters=i, init='
        random', n_init=1000)
    kmean.fit(df.values)
    sse_list.append(kmean.inertia_)

plt.plot(K, sse_list)
plt.xlabel('k')
plt.ylabel('SSE')
plt.title('The Elbow Method showing the
    optimal k')
plt.show()
```

*D. Code for Agglomeration*

```
    import pandas as pd
    import numpy as np
    from scipy.cluster.hierarchy import
        linkage
    from scipy.cluster.hierarchy import
        dendrogram
    clustering = AgglomerativeClustering(
        n_clusters=6, 'euclidean', linkage
        ='ward').fit(df)

    Z = linkage(df, 'ward')

    #plotting the dendrogram
    plt.title('Dendrogram (truncated)')
    plt.xlabel('index')
    plt.ylabel('distance')
    dendrogram(
    Z,
    truncate_mode='lastp',  # show only
        the last p merged clusters
    p=10,
    show_leaf_counts=False,
    leaf_rotation=90.,
    # leaf_font_size=12,
    show_contracted=True )
    plt.show()
```

*E. Code for Artificial Neural Network*

```
# Install Tensorflow
import numpy as np
import pandas as pd
from sklearn.model_selection import
    train_test_split
from keras import Sequential
from keras.layers import Dense

recipes = pd.read_csv('
    recipes_muffins_cupcakes.csv')
X = recipes[['Flour','Sugar']]
y = np.where(recipes['Type']=='Muffin',
    0, 1)

X_train, X_test, y_train, y_test =
    train_test_split(X, y, test_size=0.3)

clf = Sequential()

#Hidden Layer
clf.add(Dense(4, 'relu', 'random_normal')
    )

#Output Layer
clf.add(Dense(1, 'sigmoid', '
    random_normal'))

clf.compile('adam','binary_crossentropy',
     metrics =['accuracy'])

clf.fit(X_train,y_train)

model= clf.evaluate(X_train, y_train)

predictions = clf.predict(X_test)
```