



UCD Michael Smurfit  
Graduate Business School

**MIS41270 – Data Management and Mining**  
**Assignment 3 – Classification**

**Jesus Cortina Bernal**  
**21200889**

**Course Name:**  
**MIS41270 – Data Management and Mining**

**Lecturer:**  
**Dr. Elayne Ruane**

**Date:**  
**April 24th, 2022**

## **Classification**

Machine Learning technologies are becoming ever prevalent in the retail industry. This has motivated T-Superstore, a large retailer located in the US to request consulting services to assess the opportunity to implement this technology in their processes. In this report, it will be explained how implementing Machine Learning algorithms for classification can improve the bottom-line of T-Superstore and its competitive edge in the market. This report will cover a classification use case in one of the business units of T-Superstore (E-commerce) which will be explored with 3 different classification models (Naïve Bayes, Decision Tree, Support Vector Machine), comparing their benefits, limitations, and results. Eventually, the report will present the financial value of implementing these technologies, plus certain considerations that T-Superstore's management must take into account when implementing Machine Learning, AI, and related methods.

### **Task 1: Describing the Use Case**

#### **Company Description**

T-Superstore is a major retail chain with operations based in the US. This company has over 1000 physical locations distributed around the country. Within each store, the company sells all types of consumer goods including, but not limited, to clothes, food, household appliances, and medicines. All stores possess a pharmacy within their facilities, with some locations having extra services depending on the store category and customer base. The company has major distribution centers for its location, plus a thriving E-commerce online storefront. From its sales, around 25% are completed online, with a yearly increase averaging 3% during the past 4 years. Finally, just like any large organization it has internal functions such as Human Resources, Finance, and Logistics.

#### **Department Description**

A retailer is an organization with multiple departments, and as such, each one of them has its own needs and pain points that Machine Learning (ML) can alleviate. In the coming sections, I will provide a clear view of how ML algorithms can provide value to one of these departments. For the use case in question, the department chosen has been the E-commerce division within Sales and Marketing.

E-commerce is the company branch dedicated to selling products in a digital storefront. This unit was chosen because of the close relationship between technology and this department. E-commerce only exists thanks to the proliferation of technology, with consumers having millions of stores in the palm of their hands with their smartphones (Shaw, n.d.). E-commerce has changed the relationship between buyers and sellers with consumer anonymity and personalization, while also bringing interest into what key factors prompts purchases, which eventually lead to higher




revenue (Policarpo et. al, 2021). In addition, E-commerce is a major source of big data for any ML project, proving both structured (addresses, prices, time) and unstructured data (reviews, descriptions, clicks) (Apparatus, n.d.), with companies who have invested in big data analytics enjoying both higher productivity and growth (Akter and Fosso Wamba, 2016).

For T-Superstore, the E-commerce division handles its digital storefront located at [www.t-superstore.com](http://www.t-superstore.com) and all supporting operations. The site in question offers the option for any consumer to create a user profile that stores its personal information in a protected cloud server. Upon entering at the website, the customer receives a view of some of the most popular products in the store plus some recommendations based on their search history. For each product, the consumer can review, rate, and put it in an online shopping cart. Eventually, the consumer can finish the purchasing process online and choose whether to pick up the product in store or to have it delivered to their address.


Images 1. Pictures of T-Superstore's online storefront

The screenshot displays the T-Superstore website's online storefront. At the top is a navigation bar with the Target logo, links for Categories, Deals, What's New, Pickup & Delivery, a search bar, and a Sign in button. Below the navigation bar is a promotional banner for "Easter surprises" with the text "Get hopping to find all you need to celebrate on Sunday, 4/17." and "Easter Basket Gift Ideas". The banner features images of an Easter basket, a pink instant camera, and a dog with a bunny. Below the banner are three product listings:

- Quest Tortilla Style Protein Chips - Nacho Cheese - 4ct/4.5oz**  
Quest Nutrition  
★★★★☆ 381  
\$8.99  
at Cedar Rapids South  
Only ships with \$35 orders  
Free 2-day shipping with \$35 orders  
In stock at Cedar Rapids South  
Ready within 2 hours with pickup  
[Pick it up](#)
- Dymatize Nutrition ISO100 Hydrolyzed 100% Whey Protein Isolate, Protein Powder**  
Dymatize Nutrition  
★★★★☆ 36  
\$39.84 - \$94.98  
When purchased online  
Sold and shipped by iHerb  
a Target Plus™ partner  
Free standard shipping  
[Add to cart](#)
- Quest Tortilla Style Protein Chips - Loaded Taco - 4ct/4.5oz**  
Quest Nutrition  
★★★★☆ 459  
\$8.99  
at Cedar Rapids South  
Only ships with \$35 orders  
Free 2-day shipping with \$35 orders  
In stock at Cedar Rapids South  
Ready within 2 hours with pickup  
[Pick it up](#)



Share your pic



When purchased online <sup>①</sup>

★★★★★ <sup>36</sup> ▼

size **5 lb (2.3 kg)**

**5 lb (2.3 kg)** 5 lbs (2.3 kg) 3 lb (1.4 kg) 3 lb <sup>②</sup>

color **gourmet chocolate**

**gourmet chocolate** peanut butter chocolate p <sup>②</sup>

**Ship to 52404**

[Edit location](#)


Get it by **Fri, Apr 22**

Ships free


This delivery date includes extra time for the weekend.

This item isn't sold in stores

Qty 1 ▼ **Add to cart**

 **Sold and shipped by**  
iHerb >

[Report this item](#)

 Not eligible for registries <sup>①</sup>

### About this item

- Details
- Label info
- Shipping & Returns
- Q&A

#### Specifications

Contains: Milk

Weight: 5.82 ounces

Protein Grams per Serving: 25 g

**Product Warning:** Keep out of reach of children


Product Form: Powder

Show more

#### Description

If your goal is gains in muscle size and strength, then ISO100 is your perfect workout partner. Loaded with muscle building amino acids, ISO100 can support even the most serious resistance-training programs. Known worldwide for quality, taste and purity, ISO100 is produced to our highest quality standards. ISO100 is formulated using a cross-flow microfiltration, multi-step purification process that preserves important muscle-building protein fractions while removing excess carbohydrates, fat, lactose and cholesterol. ISO100 is made with pre-hydrolyzed protein sources to ensure fast digestion and absorption. Available in variety of delicious indulgent flavors, ISO100 is the perfect before-workout, after-workout, anytime protein. Contains both Milk and Soy.


### You might also want



**\$94.98**  
Dymatize Nutrition ISO100  
Hydrolyzed, 100% Whey...

✓


+



**\$12.89**  
OLLY Women's Multivitamin  
Gummies - Berry - 90ct

✓


+



**\$7.99**  
Premier Protein Shake -  
Chocolate - 11oz/4pk

✓

+



**\$8.99**  
Blender Bottle 20oz  
Portable Drinkware

✓

Subtotal: **\$115.86** (4 items) **Add all 4 to cart**

### More to consider



\$32.99



\$24.99



\$18.99



\$63.71



\$24.99



\$20.99



\$18.99

>

Images from Target.com (n.d.)

## Use Case Description

There are multiple ways to apply ML in E-Commerce. Some of them include data exploration for discovering underlying purchasing patterns, purchase prediction, repurchase prediction, recommendation systems, customer relationship management and fraud detection (Policarpo et al., 2021). This document will present a use case showcasing a classification ML algorithm for classifying products. However, unlike the previous examples that handle products already out in the market, the ML case explored here will target the problem of advertising new products.

Selling new products is a huge challenge for any company, especially retailers. According to an article from Harvard Business Review (Schneider and Hall, 2011), 75% of consumer goods and retail products won't earn even \$7.5 million USD in their first year; meanwhile, only 3% reach the benchmark value of \$50 million USD for a successful launch. In contrast, other sources estimate that the actual failure rate of new products falls around 40%, with some industries having lower or higher percentages such as consumer goods with 45% (Castellion and Markham, 2013). Regardless of the exact failure rate, the point is that launching a new product is a risky business.

Furthermore, according to the Harvard article (Schneider and Hall, 2011), American households buy roughly the same 150 products every time. Plus, even if a product launches successfully, low sustained sales could make it flop. Among the most important reasons for these flops is the lack of preparation for launch, especially when it comes to advertisement (Schneider and Hall, 2011).

As one can deduce, companies spend too much time developing new products only to not know how to properly advertise them, and most importantly to whom advertise them. In addition, while market surveys and other similar market research methods do provide insights, these methods have some limitations. For instance, the answers from the surveys might be too old to be used or from a very small sample that doesn't even represent your customer base (Magusara, 2019). Similarly, they can also be very expensive, with the cost ranging from approximately \$20000 USD to \$60000 USD per month, with the average project lasting between 2 to 8 weeks and some programs lasting even longer (Frederiksen and Waffle, 2021).

As seen from these values, market research can become a very expensive endeavor considering the number of products and consumer niches needed to be researched. To make matters worse, as trends become more ephemeral traditional market research methods won't hold to the pace of our current world. Now, for a retailer like T-Superstore this challenge is an everyday task since it must decide constantly which new products to advertise and which to ignore.

This is where ML technology can provide an edge. Using these techniques, the Sales and Marketing team at T-Superstore can explore in real-time customer behavior and identify new sales opportunities. Unlike traditional research methods, ML methods can give you faster answers with potentially lower costs. Similarly, the data used for these models would come directly from the clients' activities at T-Superstore's online website, giving more confidence that the data represents the actual customer-base. Ultimately, developing the capacity to identify new products with the potential to be top sellers fast and reliably can lead to a competitive advantage in the retail sector.

## **Task 2: Describing the Proof of Concept**

### **Proof of Concept Description**

To demonstrate the business and financial value of implementing ML in T-Superstore's E-commerce division, in the next sections I will apply ML algorithms to the new product advertising dilemma. As explained before, the use case for ML will involve deciding which new product needs to be advertised to maximize sales. To achieve this, the ML algorithm will attempt to label a set of products as Hot or non-Hot. A Hot product in this case is one that the consumer will adore, thus naturally having a larger potential for sales. In contrast, a non-Hot product will not be as well received as a Hot one. To be more specific, non-Hot products involve those goods that the consumer doesn't like, but also those that are good but not as stellar as the Hot ones. In addition, the task will be done with 3 different approaches to demonstrate the value and limitations of different classification machines.

### **Data Description**

To demonstrate the value of this technology, I will use it on a dataset with product information. Given that the company has decided not to provide actual data from new products, the data used for this demonstration comes from a public source. In this case, the data used was obtained from Kaggle.com (Saja, Yip and Madi, 2019). The data consists of product information from fitness and health supplements, such as protein drinks and powders. The data was scrapped from an online business dedicated to selling this kind of goods, Bodybuilding.com. The clean data from this source contains information for 303 different products, with the key attributes used in this demonstration being:

- Brand Name: brand name of the good's manufacturer.
- Number of Flavors: number of flavors offered for that good.
- Price: unit price in USD.
- Product Category: product category in the online store.
- Product Description: text description of the product in the online website.
- Overall Rating: consumer rating from 0 to 10, and the variable used for labelling.

This data was used because it contained a good mix of numerical and categorical variables. Plus, it also offered a text variable in the product description for Natural Language Processing. As for the information in the variables, the data offers attributes that are available before a product is released, resembling the real implementation of the model. Similarly, it offers a variable that can be used for labelling purposes, in the form of the Overall Rating. Finally, this data is appropriate to reflect the behavior of a consumer segment since the rating score is from a niche site for fitness goods. This assures some degree of homogeneity from the review scores that will reflect the behavior of a dedicated consumer base.

## **Limitations and Assumptions**

To properly use this data, it first required to have a label. Since the original data didn't have a label, the Overall Rating was used for labelling. In this case, a Hot product is one which will have a User Rating of at least 9. This means that we are assuming that a product with a higher user score leads to more sales in the long run, which can make sense given word of mouth and users favoring well-reviewed products. In this scenario, we are also assuming that the entirety of a product's performance depends on its inherent attributes prior to entering the market, which isn't that true. Viral marketing, scandals, and some types of offers can boost a product's reputation and change its sales. Finally, one last key assumption is that the Overall Rating scores come from actual consumers. In real life, there is the possibility that these numbers came from bots and given that the numbers are aggregates there is no way to validate this claim. So, for now, the best one can do is to assume that humans are the ones rating.

As for limitations, the first one is that these models can account for a product's description at an exact time. For instance, one of the variables is the number of flavors. At release a product can start with one flavor but by the end of the year the product might end up with many flavors. Similarly, changes in prices will require the model to be changed. In addition, this model is limited to only one category of products, the health and fitness supplements. It would be valuable to have a model coming from a mix of products from different categories for comparison's sake. Thought, having data from just a specific category could lead to more accurate results for that category.

## Task 3: Describe the Data

### Dataset Description

The raw dataset used for the demonstration contained information for over 800 products. However, most of these records were incomplete. That is, that they had missing information in one of the relevant attributes. Thus, they needed to be removed along with some of the unnecessary columns. These actions were done using the following code.

#### Snippet 1. Importing and removing nulls and unnecessary attributes

```
raw_prod = pd.read_csv("products.csv")

raw_prod.drop(['average_flavor_rating', 'link', 'number_of_reviews', 'price_per_serving',
'product_name', 'top_flavor_rated', 'verified_buyer_number', 'verified_buyer_rating'], axis = 1,
inplace = True)

raw_prod.dropna(inplace = True)
```

After cleaning the dataset, the data ended with 303 records. Here is a brief description of some of the relevant statistics for each attribute:

- Brand Name:
  - Number of Brands: 58
  - Most Frequent Brand Count: 33 (Optimum Nutrition)
  - Less Frequent Brand Count: 1 (16 brands)
- Number of Flavors:
  - Max: 43
  - Min: 1
  - Mean: 7.03
  - St. Deviation: 7.71
  - Median: 5
- Price (USD and per unit):
  - Max: 119.53
  - Min: 3.05
  - Mean: 34.33
  - St. Deviation: 19.15
  - Median: 31.45
- Product Category:
  - Number of Categories: 36
  - Most Frequent Category Count: 68 (Whey Protein)



- Less Frequent Category Count: 1 (9 categories)
- Overall Rating:
  - Max: 10
  - Min: 5.7
  - Mean: 9.97
  - St. Deviation: 0.55
  - Median: 9.1

From these, we can say that most of these products have a price tag of around \$30 USD, with a rating of 9 out of 10, and coming in around 5-7 flavors. In terms of categories, the most frequent brand and category are Optimum Nutrition and Whey Protein.

## Labelling

Moving on, the next step was labelling. As mentioned before, the labels were created using the overall rating score with the following code. In this snippet, Hot products received the label of 1.

### Snippet 2. Labelling

```
raw_prod["label"] = raw_prod["overall_rating"].map(lambda x: 1 if x >= 9 else 0)
```

The partition value was 9 because it provided a decently balanced number of labels, with 184 products labelled as Hot out of the 303. Also, because it could be argued that many people would consider a score of 9 to be representative of a very good product. Besides 9, other values tried were 8.5 and 9.5, but they resulted in over 250 labels for the Hot category which could lead to biases. Plus, going for values beyond that range would include products that do not represent the Hot category.

## Data Wrangling

After labelling, the next steps for cleaning depended on the model used. For each of the three models, a deep copy of the original data was used to avoid any unintended consequence. For two of the models, Naïve Bayes and Decision Tree, the categorical variables were transformed into numerical one. In this case, the variables for the brand and the product category were One Hot encoded. These categories were created using the code in Snippet 3. Once the variables were One Hot encoded, the dummy columns for the brands and categories were put together with the price, and the number of flavors variables, leaving this data ready for splitting into the test and training datasets. Also, for the Naïve Bayes implementation, the data required standardization with a scaler

after the split into the training and test datasets, done as shown in Snippet 4. The standardization code was based on Surbhi\_22 (2021).

### Snippet 3. One Hot Encoding

```
#One_hot
onehot_brand = pd.get_dummies(dt_prod["brand_name"], prefix = "Brand: ")
onehot_category = pd.get_dummies(dt_prod["product_category"], prefix = "Category: ")
dt_prod = dt_prod.join(onehot_brand)
dt_prod = dt_prod.join(onehot_category)
```

### Snippet 4. Standardization

```
#Standardizing
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

For the third method, the Supply Vector Machine, the only column used was the product description. To clean this column, the text description was transformed to its most basic components. To achieve this, the cleaning activities required were:

- Transforming all words to lowercase, since algorithms can mix-up lower and upper cases.
- Tokenization to transform each sentence into a list of words.
- Removing stop words (like to, the, or), since they are mostly useless for the analysis.
- Removing punctuation, such as exclamation points, and numbers.
- Lemmatization, to keep the core piece of the words.

The sources used for the NLP data wrangling process are Bedi (2018), GeeksforGeeks(2021), Shukla and Iriondo (2020), Malik (n.d.), and Jablonski (2021). After the text was cleaned, the words were put back into a single string for the training-test split. The code for these processes is found in Snippet 5.

## Snippet 5. Cleaning the Product Descriptions

#Text Preprocessing

#Putting all in lowercase

```
svm_prod["product_description"] = [entry.lower() for entry in svm_prod["product_description"]]
```

#Tokenization

```
svm_prod["product_description"] = [word_tokenize(entry) for entry in svm_prod["product_description"]]
```

#Removing stop words

```
def remove_stop(entry):
    stop = stopwords.words("english")
    word_list = []
    for word in entry:
        if word not in stop:
            word_list.append(word)
    return word_list
```

```
svm_prod["tkn_no_sw"] = svm_prod["product_description"].apply(
    lambda entry: remove_stop(entry))
```

#Removing Special characters

```
def remove_punct(entry):
    sp_chars = punctuation
    word_list = []
    for word in entry:
        true_list = []
        for char in word:
            if char in punctuation:
                true_list.append(False)
            else:
                true_list.append(True)
        if False not in true_list:
            word_list.append(word)
    return word_list
```

```
svm_prod["tkn_no_sw_p"] = svm_prod["tkn_no_sw"].apply(  
    lambda entry: remove_punct(entry))
```

#Removing numbers

```
def remove_numb(entry):  
    numb_chars = "0123456789"  
    word_list = []  
    for word in entry:  
        true_list = []  
        for char in word:  
            if char in numb_chars:  
                true_list.append(False)  
            else:  
                true_list.append(True)  
        if False not in true_list:  
            word_list.append(word)  
    return word_list
```

```
svm_prod["tkn_no_sw_p_nb"] = svm_prod["tkn_no_sw_p"].apply(  
    lambda entry: remove_numb(entry))
```

#Lemmatization

```
def WNL(entry):  
    lemmatizer = WordNetLemmatizer()  
    word_list = []  
    for word in entry:  
        lem_word = lemmatizer.lemmatize(word)  
        word_list.append(lem_word)  
    return word_list  
svm_prod["tkn_lemm"] = svm_prod["tkn_no_sw_p_nb"].apply(  
    lambda entry: WNL(entry))
```

#Joining the cleaned words

```
svm_prod["clean"] = svm_prod["tkn_lemm"].apply(lambda entry: " ".join(entry))  
svm_final = svm_prod[["label", "clean"]]
```

Table 1. Example of the Raw Text versus the Clean Text

Raw Text	Clean Text
'24g of Pure, Quality Protein in Every Scoop with No Added Amino Acids or Filler Nutrients'	'pure quality protein every scoop added amino acid filler nutrient'

After splitting the clean text, one final transformation was needed: the Term Frequency Inverse Document Matrix, which is a metric that counts how important is a word in a document within a collection of documents (Stecanella, 2019). This needed to be done separately for the training and testing datasets using the TfidfVectorizer function as shown in Snippet 6. Once the TFID process was finished, the text data was ready for classification.

Snippet 6. TFID Matrix

```
#Word Vectorization aka TermDocumentMatrix and Term Frequency Inverse Document
Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(svm_final['clean']) #TFID tokenizes on itself, so need to regroup

#Creating the TFID frames
X_train_Tfidf = Tfidf_vect.transform(X_train)
X_test_Tfidf = Tfidf_vect.transform(X_test)
```

## Task 4. Classification Experimentation & Results

### Model Usage

Having cleaned the data, it is time to apply the ML algorithms. In this use case, there will be 3 algorithms used: Naïve Bayes, Decision Tree, and Support Vector Machine. Each of them has their own section describing their benefits and limitations, plus the way to interpret them. All 3 classification techniques were applied using the same training data split parameters as shown in Snippet 7. After having discussed each algorithm individually, there will be a section comparing all 3 of them and their results.

Snippet 7. Training and Test Data Split

```
#Building the training dataset
X_train, X_test, Y_train, Y_test = train_test_split(dt_X, dt_Y, test_size = 0.3, random_state = 1996)
```

### Model 1. Naive Bayes

The first model implemented is Naïve Bayes (NB). NB is based on applying the Bayes Theorem, a theory that explores the relationship between conditional probabilities (Scikit-learn, n.d.D). Historically, this model has been very successful for document classification and spam detection since it doesn't need a lot of training data. NBs are also extremely fast and very cheap in computational costs (Scikit-learn, n.d.D).

In E-commerce, NB has been used for predicting user behavior and profiling; specially, there is a case in which NB was used to understand the consumer's decision style by using it to complete a consumer's set of decision (Policarpo et al., 2021). In this case, I'll be using NB to explore whether a product is Hot given its price, product category, brand, and number of flavors.

The model was implemented using the Python module called scikit-learn, which possesses multiple modules for different classification machines. In this case, the machine used is called Gaussian Naïve Bayes, a convenient NB machine that doesn't require complex parameter tuning as shown in the snippet below. This implementation was based on Surbhi\_22 (2021).

Snippet 8. Gaussian Naïve Bayes

```
#Doing Naive Bayesi
NB_machine = GaussianNB()
NB_machine = NB_machine.fit(X_train, Y_train)
```

```
Y_pred = NB_machine.predict(X_test)
```

While NB is quite reliable, straightforward, and simple to use, it does come with some considerations. First, this model performs well at classifying objects, but it doesn't have a good performance when it comes to probabilities (Scikit-learn, n.d.D). In other words, it's very good at telling you if a product is Hot, but not why that product is Hot. Is it because of the price, the brand, or the category? The model doesn't know.

Still, there is a workaround to this problem. Using a process known as Permutation Importance, it is possible to identify how important is a variable to a specific model. This technique does it by checking how much a model's score drops by removing that variable (Scikit-learn, n.d.E). The snippet and the table below show the implementation and the results of this technique, which was based on Kumar (2020).

#### Snippet 9. Permutation Importance

```
imps = sklearn.inspection.permutation_importance(NB_machine, X_test, Y_test)
importances = imps.importances_mean
std = imps.importances_std
indices = np.argsort(importances)[-1:]

#Feature Ranking
print("Feature ranking:")
for f in range(X_test.shape[1]):
    print("%d. %s (%f)" % (f + 1, nb_feature_cols[indices[f]], importances[indices[f]]))
```

Table 2. Top and Worst features - Permutation Importance

Rank	Feature	Importance Score
1	Brand: EVLUTION NUTRITION	0.035
2	Brand: JYM Supplement Science	0.021
3	Brand: Cellucor	0.017
4	Brand: Labrada	0.015
93	Category: D-Aspartic Acid	-0.008
94	Category: Whey Protein Blends	-0.013
95	Brand: Gaspari Nutrition	-0.018
96	Brand: Met-Rx	-0.021

Based on the table above, we see that in this model, some of the most important variables are Brands. For example, if we remove the brand EVLUTION NUTRITION, the model's score drops by 0.035 percentage points. On the other side, by removing Met-Rx the model's score improves. In any case, while this approach can tell you how important a brand or a category is to this model, these inferences will only work for this model. That is, running a new NB machine on another training data will yield a different set of permutation rankings. For instance, maybe EVLUTION NUTRITION could end up as the worst ranked variable in another model, yet in real-life it might be a crucial factor when classifying. In the end, while we can patch some of the limitations of NB, using it is still a Black Box approach. It's like knowing that a tsunami will come when the shore recedes, but not knowing why it does so.

## Model 2. Decision Tree

While Naïve Bayes won't tell you which factors make a product Hot, there are some models that can provide insights on that and more. This is the case of the Decision Tree (DT), a model which tries to classify the objects in the model by obtaining simple decision rules learnt from the training data (Scikit-learn, n.d.A). Eventually, those rules can be visualized in a decision tree for everyone to see and understand. In E-commerce, DT and its related methods have been used for predicting repurchase in customers, fraud detection and identifying user behavior (Policarpo et al., 2021).

Among the benefits of using a DT, according to the documentation (Scikit-learn, n.d.A), are:

- Easy and simple to understand
- Useful for data exploration
- Visualizations for the Tree's rules
- Relatively cheap to compute
- Handles multiple data types
- Not only for binary classification but for multiple labels too
- White-box model i.e., the user can see how the model works behind the scenes

As for its implementation, DTs can be implemented using scikit-learn too in the python environment. The next snippets show both the implementation of the DT and the creation of the DT chart on the product dataset consisting of the Brand, Category, Price, and Number of Flavors variables. The snippets are then followed by the image of the DT itself. The code for implementing the DT was based on Navlani (2018) and Scikit-learn (n.d.A).



### Snippet 10. Decision Tree

```
#Summoning Machine

#Criterion and Max_Depth
tree_prod = DecisionTreeClassifier(max_depth = 4, random_state = 1996)

#Fitting the data
tree_prod = tree_prod.fit(X_train, Y_train)

#Predicting the response for test dataset
Y_pred = tree_prod.predict(X_test)
```

### Snippet 11. Building the Decision Tree Visualization

```
dot_data = StringIO()

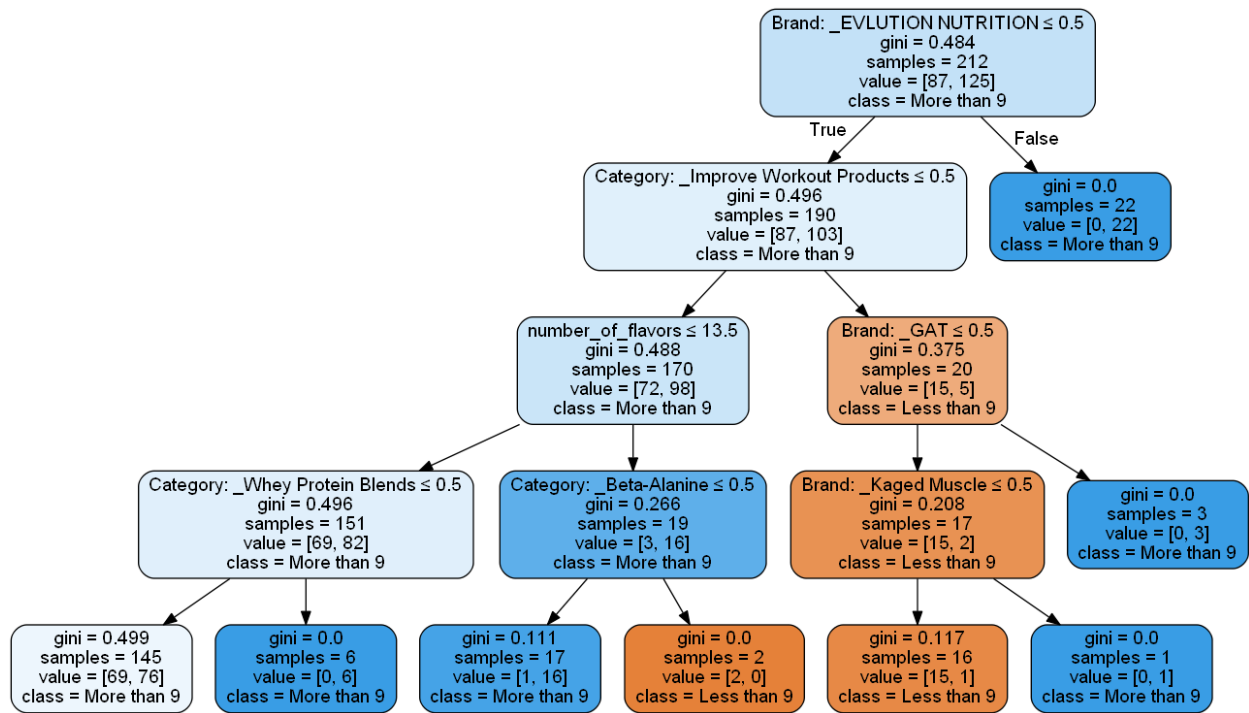
export_graphviz(tree_prod, out_file = dot_data, filled = True, rounded = True,
                special_characters = True, feature_names = dt_feature_cols,
                class_names = ["Less than 9", "More than 9"])

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())

graph.write_png("Products.png")

Image(graph.create_png())
```

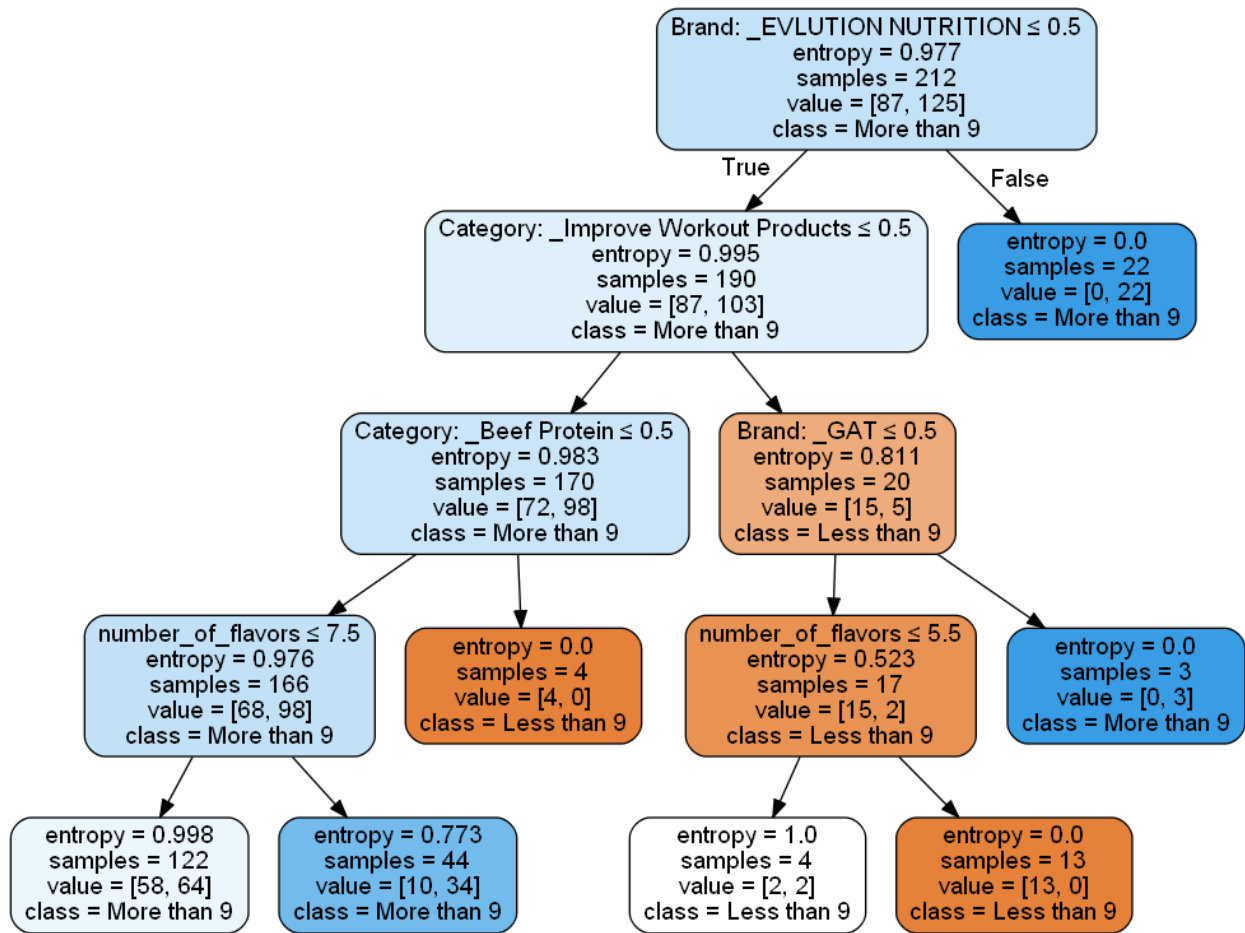
Image 2. Decision Tree with Gini criterion and maximum depth of 4.



Analyzing the DT above, we can deduce some of the factors that make a product Hot. For instance, we see that the model discovered that a really important factor to determine whether a product is Hot or not is if they are from the EVLUTION NUTRITION brand. The rule in the tree, represented by the first node, indicates that if a product is from this brand, they are Hot. If they are not, you move to the next node to apply another rule. Similarly, a product in the Category: Improve Workout Products, that is neither from the Brand: GAT nor Brand: Kaged Muscle will not be a Hot product. These simple rules can be easily understood by non-experts, but most importantly they can be applied with ease. For instance, with this rule we see that advertising a new product from EVLUTION NUTRITION is a favorable bet.

Still, while DTs are quite useful and approachable, there are some considerations in their use. First, DT's can be done in two methods according to the metric used to create them: GINI, which maximizes purity, or Entropy, which minimizes disorder (Aznar, 2020). In essence, both are quite similar. However, they can net very different results when building the tree. Just look at the tree Image 3 and compare it with the tree in Image 2. While some of the starting nodes are similar, as one digs deeper in the tree some of the rules change.

Image 3. Decision Tree with Entropy criterion



Besides that, DTs have a major problem known as overfitting. In simple words, overfitting means that the model works too well for the data it was created from, but for other data it starts to tumble (Scikit-learn, n.d.A). In fact, it has been found that in isolation DTs tend to underperform compared to other methods (Policarpo et al., 2021). Fortunately, there is a technique called Random Forests that counterbalances this issue by essentially creating multiple trees and obtaining the average tree out of them (Scikit-learn, n.d.B). Regardless of this issue, one can see the value of this technique when it comes to making ML findings accessible to the non-expert manager with clear rules and visualizations.

### Model 3. Support Vector Machine

The final classification model implemented is the Support Vector Machine (SVM). SVM is a classification method that essentially creates zones where each item in that zone belongs to that category (Scikit-learn, n.d.G). In E-commerce, SVMs have been used for fraud detection and product recommendations (Policarpo et al., 2021). Among its advantages, SVM is known to work

well with datasets with a lot of variables (high dimensions), its versatility for some of the parameters, and for being relatively cheap in computational costs (Scikit-learn, n.d.G).

The benefit of being quite effective for data with a lot of variables makes them a very popular tool for text processing. In fact, the implementation shown here will consist in applying a SVM to the clean product descriptions, hoping to identify if certain attributes shown in the description make the products more likely to be Hot. The next snippet illustrates the code for creating the SVM model which was applied on the clean descriptions that were transformed into the term matrix format. The implementation was based on Bedi (2018).

#### Snippet 12. Support Vector Machine

```
#SVM Machine
SV_prod = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SV_prod.fit(X_train_Tfidf, Y_train)

#prediction
Y_pred = SV_prod.predict(X_test_Tfidf)
```

Regarding its interpretation, SVM is a bit similar to Naïve Bayes since it's a bit hard to interpret the weight of the variables in the model. However, unlike Naïve Bayes, it is possible to extract those values from the model. In other words, we can discover which words in the product description makes them more likely to be Hot. In the next table, we can see the top and bottom 5 words by their weight in the model, out of the 277 words considered by the SVM.

Table 3. SVM Words and Weights

Word	Weight
Support	1.536
BCAAS	1.515
Protein	1.186
Energy	1.186
Ultimate	1.164
Nighttime	-1.078
Weight	-1.218
Promote	-1.221
Performance	-1.260
Isolate	-1.291

Based on the model, we see that some product features such as Protein and BCAAS (Branched-Chain Amino Acids) seem to be quite popular, and that new products with these components are less risky to promote. On the other side, words like Weight, Nighttime and Isolate (from a

particular type of protein powder) don't make the product Hot. So, if a new product arrives with a description of Whey Protein Isolate for Nighttime, it is fair for the E-commerce unit to not advertise it as much.

In any case, just like the other models, SVMs have their considerations. While it can deal with many variables, having too many of them compared to the sample data can lead to biases (Scikit-learn, n.d.G). Furthermore, some SVM models can be quite heavy in computational costs as the amount of data increases. In addition, if the SVM works with numeric data it is necessary to standardize it to avoid biases. Finally, while SVM has plenty of flexibility incorporated thanks to its parameters, this same flexibility requires the knowledge to adjust them according to the business case (Scikit-learn, n.d.G).

## **Model Comparison and Result Interpretation**

Now that the 3 models have been run, it is time to compare their effectiveness to discover which model is the best for the E-commerce division. That is why in the next table the results for the 3 models previously used will be displayed. To make this comparison, there are some metrics that are commonly used (Scikit-learn, n.d.C). The metrics displayed in the table, along with their meaning in the context of the use case are the following:

- True Positive: amount of actual Hot products classified as Hot.
- True Negative: amount of actual non-Hot products classified as non-Hot (or actual non-Hot not classified as Hot)
- False Positive: amount of actual non-Hot products classified as Hot.
- False Negative: amount of actual Hot products not classified as Hot.
- Precision: ability of a machine to identify actual positives (how many predicted Hot products are actually Hot?)
- Recall: ability of a machine to find all positive samples (did it identify all the actual Hot products?).
- Accuracy: ability of a machine to correctly identify all objects (did it identify all products correctly, Hot and non-Hot?)
- Error Rate: tendency of a machine to make mistakes at labelling all objects (how many products were labeled incorrectly?)
- F-Score: Weighted Mean of the Precision and Recall. A metric to facilitate comparisons with machines.

Table 4. Model Metrics for the 3 Implementations

Metric / Model	Naïve Bayes	Decision Tree	Support Vector Machine
True Positive	24	57	46
True Negative	26	3	16
False Positive	6	29	16
False Negative	35	2	13
Precision	80%	66.27%	74.19%
Recall	40.67%	96.61%	77.96%
Accuracy	54.94%	65.93%	68.13%
Error Rate	45.05%	34.06%	31.86%
F-Score	53.93%	78.62%	76.03%

Now, some of the findings of these comparison are:

- For this data, DT and SVM seem to have vastly outperformed NB in all metrics except for the precision score. That is, that while NB seems to not identify all the Hot products, it is the best at avoiding the non-Hot products.
- On the other side, the DT was the best at identifying all the Hot products with the highest recall score very close to 100%. However, this high recall came at the cost of having the highest number of False Positives. In other words, this model is more open to identify a non-Hot product as Hot if it means having as many Hot products as possible.
- As for the SVM, it seems that it is the best performing model of the three. I consider this because its F-score is quite higher than NB and very close to SVM. Similarly, it isn't as extreme as the DT when it comes to False Positives by keeping a high Precision score without labelling too many non-Hot products as Hot. Finally, it can avoid plenty of False Negatives as shown with its high Recall Score of 77%, meaning that it can detect plenty of the potential Hot products.

Overall, in this use case, the recommendation for the E-Commerce team would be to use the Support Vector Machine model. Thanks to this exercise in ML algorithms, the E-commerce has now discovered the insight that product descriptions are the best predictors for knowing if a product will be a hit and if it deserves to receive attention from marketing.

## Task 5. Next Steps & Recommendations

### Financial Value of the Model

To start this section, I will present an estimate of the potential value of using ML for advertising new products. As explained, a Hot product is one that will be well received by the consumer, unlike a non-Hot one that will be received decently. Thus, a Hot product will have a larger increase in revenue than others when given the same amount of advertisement. The next table compares the performance of both products, with average prices coming from the dataset while the increase in sales is around 20% based on an article by Harvard Business Review (Abraham and Lodish, 1990). Furthermore, it is assumed that both products will sell the same without advertising, while the Hot product will have double the increase in sales when advertised.

Table 5. Revenue by Advertisement Level and Product Type

Metric	Hot	Non-Hot	Difference (H > C)
Quantity	184	119	
Avg. Price (USD)	\$33.85	\$35.08	
Avg. Unit Sales W/O Advertisement	1000	1000	
Avg. Yearly Revenue W/O Advertisement	\$33,850	\$35,080	-\$1,230
Avg. Unit Sales W/Advertisement	1400	1200	200
Avg. Yearly Revenue W/Advertisement	\$47,390	\$42,096	\$5,294
Increase in Revenue due to Advertisement	\$13,540	\$7,016	\$6,524

Based on the table, we can see that even if a non-Hot product has a larger revenue without advertisement, advertising a Hot product drastically improves the revenue compared to the other goods. In this scenario and assuming an unlimited advertising space, by not advertising a Hot product T-superstore misses on \$13,540 USD. To complement, if there is a limited amount of advertising space, the opportunity cost of advertising a non-Hot product rather than a Hot one is of \$6,524 USD. This means that if you advertise the wrong product you are missing on that amount. As one might think, while this amount seems small, in the real-world units and sales are not in thousands but in millions. Outside, those \$6000 USD could be \$6 million USD, and missing on that amount is a terrible mistake.

## Limitations and Considerations for ML Implementations

As demonstrated during this paper, ML can provide new insights, discover untapped revenue, and turn into a competitive advantage for T-Superstore. However, implementing ML is not as simple as it sounds. For a company to begin using ML, AI, and any other similar technologies it needs to be aware of not only the benefits it can net, but the costs, potential pitfalls and risks involving their use. In this final section, I will discuss some of the considerations, limits, and risks that T-Superstore's management must consider before adopting these types of technologies.

### -Model Limitations

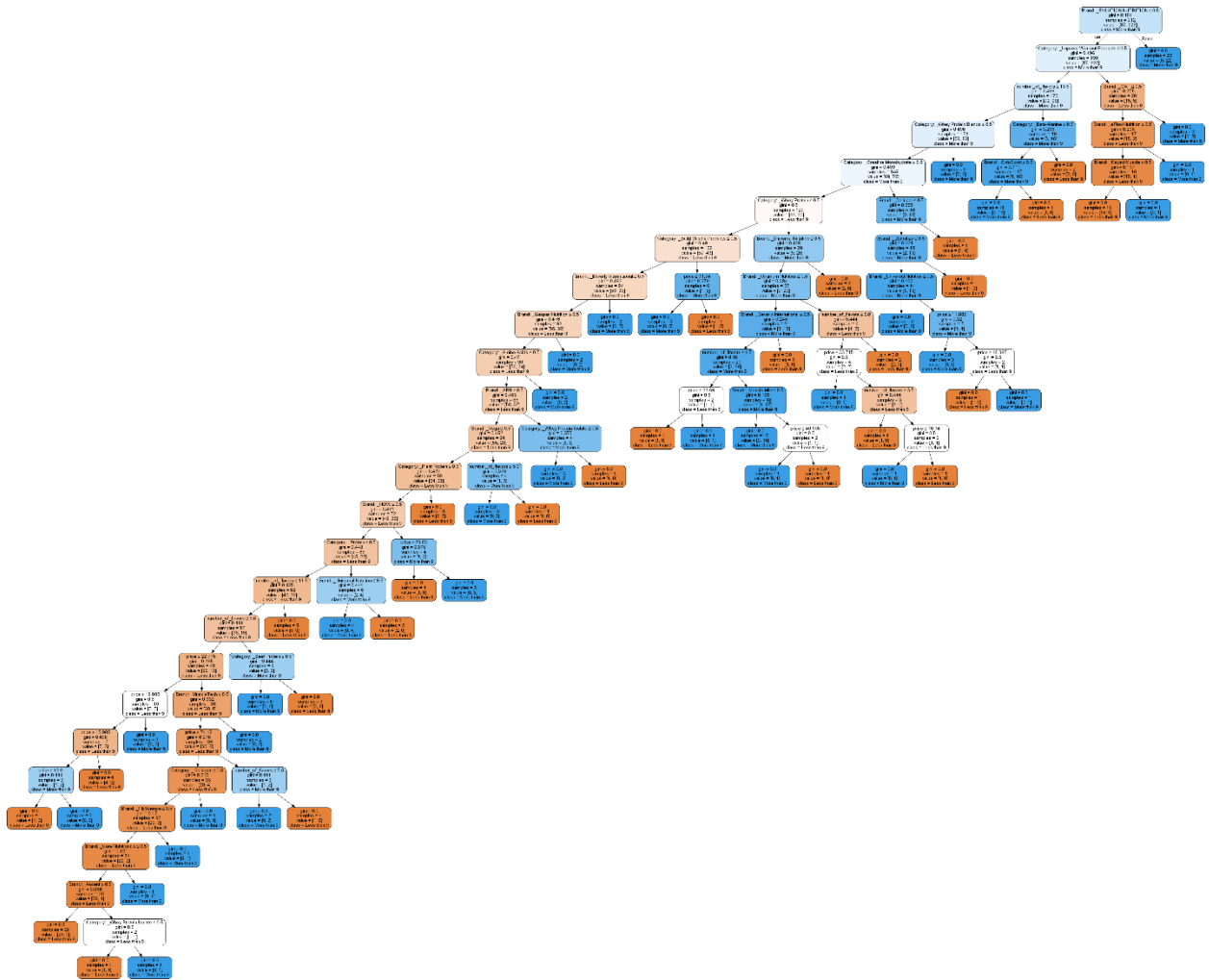
To begin with, it should be noted that even though these types of models are often referred to as Artificial Intelligence, they aren't as smart as one might think. For starters, models can require calibration. When it comes to classification, there are two things to consider: the label (is it Hot or not), and the confidence on the label (how likely is it to be Hot). By calibrating a model, one can obtain reasonable and interpretable probabilities of this label, and maybe even increasing the model's performance (Scikit-learn, n.d.F). For instance, in the next table there is a comparison of the results for the non-calibrated Naïve Bayes versus the calibrated one. As one can see, the calibrated model performs better compared to the original one based on the F-Score and many other improved metrics, while also providing much better likelihoods.

Table 6. Performance of Non-Calibrated vs Calibrated Naïve Bayes

Metric / Model	Naïve Bayes	Calibrated Naïve Bayes
True Positive	24	34
True Negative	26	19
False Positive	6	13
False Negative	35	25
Precision	80%	72.34%
Recall	40.67%	57.62%
Accuracy	54.94%	58.24%
Error Rate	45.05%	41.75%
F-Score	53.93%	64.15%



Image 4. Complete Decision Tree for GINI method



Adding on model interpretability, even some of the easier models to understand can become tricky to read. For instance, in the model section, the Decision Trees shown in Images 2 and 3 are reasonably small. However, they are only so because they have been cut. The actual complete tree for Image 2 is shown in Image 4. As one can easily see, traversing a tree of that size can become confusing and time consuming for any kind of employee. In fact, the tree is so big that displaying it in this report reduces its image quality, so care is needed even for some of the most accessible models.

Next, when it comes to data, a model is like a baby. It needs to be fed well and taken care of so it can become a healthy person. If you feed a model very poor data, the model will just throw at you mixed signals and garbage conclusions. Similarly, a model cannot be left unattended. As shown by the Covid pandemic, things can change drastically from one day to another. And when things change, the model needs to change to. Models must be kept up-to-date, ready to be modified or

replaced when needed to. Furthermore, models, especially the classification ones, have troubles working with unseen categories. For instance, what if a new brand appears in the market. The model has no idea what to do with it.

Finally, one should always remember that there are things that machines cannot see. A machine will not detect if a dataset is biased, nor if it is unbalanced, or if there are any legal troubles in the data. That is why T-Superstore needs to have experts in this area so the company can extract the maximum benefits of these algorithms, while avoiding potential dangers. In a way, it can be said that machines are very reliable. You always know what to expect from a machine since it can only do what it is programmed to do. Nothing less, but nothing more.

### **-Problem Definitions**

In the section where I compared the results of the algorithms, I proposed that the Support Vector Machine was the most ideal one for labelling Hot products. I considered this because it seemed to be the best one at identifying Hot products without risking too many False Negatives nor False Positives. However, in real life companies can have different targets and circumstances. For example, what if the decision is to promote or not to promote a product without any space constraint. In that situation, we don't care about having some False Positives if we get as many True Positives as possible, turning the Decision Tree into the best model.

Similarly, the product mix can drastically affect the preferred model. For instance, if historically a product category has few Hot products, you will want a model with a high Recall and minimum False Negatives. This is because if you only have so few good products, you don't want to miss on any of them. On the other hand, for a category with a lot of Hot products having a high count of False Positives is worse than having many False Negatives. If I have a lot of good products, then I don't want to promote a bad one, especially if there are space constraints. These types of dilemmas can only be solved by the management team, and they need to be translated to model specifications so these machines can actually be valuable.

Finally, it is important to truly identify what is the label. In this report, I have talked about Hot products and non-Hot products, with the Hot ones being labelled. I used this classification to easily identify the two categories within the text. But what would have happened if the non-Hot products were called Cold? Is it right to call them that way? It is important to determine whether to label the non-labeled products properly. In this case, Cold and Non-Hot have different implications which can lead to misunderstandings. Furthermore, what if the goal had been to identify the non-Hot products? In this scenario, it is likely that the final models would have ended up vastly different than the ones obtained. Only by management working together with the analytics team it is possible to define the problem properly so the models can be interpreted properly.

### **-Infrastructure and Data Capabilities**

To be able to effectively implement ML algorithms, it is necessary to have the data capabilities to carry them. As demonstrated by the SVM model, data needs to be cleaned thoroughly or else the

models will fail. But even before that, the company must have access to the data it finds relevant. Without having the data available, it's like having the best chef in the world in your kitchen with no ingredients, or worse, with rancid ones.

In terms of storage, among the requirements that T-Superstore must consider for how to store its data are scalability, security, cost efficiency, flexibility for different software, and capacity for integration with cloud servers (Clouddian, n.d.). Once the company has the data, it will be necessary to prepare it for the models by performing Data Wrangling and Data Preprocessing. These two steps involve exploring, preparing, and modifying the data for the model and require competent data scientists or analysts to be done properly (Wahner, 2017). And, as shown with the SVM model, it can get really messy and tricky to do this.

Besides having the data ready, T-Superstore will need enough computer power to run these models. While the models shown in this report are relatively cheap in computational costs, there are some that are more costly. Furthermore, unlike this test case which worked with a very small dataset, in real life datasets could have millions of observations with hundreds of variables. With many of the industry providers recognizing the value of having faster memory for ML activities (Farber, 2017), T-Superstore cannot overlook computational costs.

Finally, it is worth mentioning that data and technology capabilities do not only involve the computers and the machines. It involves having the right people in the right place. As identified by Cosic et al. (2015), analytics capabilities require governance, culture, and people besides technology. None of these 3 factors depend on having the strongest computer but having the right mentality in management for adopting ML and other technologies in their processes, for having people that will champion these initiatives, and for having a culture that will support and embrace this shift within T-Superstore.

## **-Legal Requirements and Ethical Concerns**

To close this section, the final consideration for management when implementing ML involve legal requirements and ethical conflicts. Starting with the law, we live in a time where technology is no longer the wild west. Regulations such as the GDPR and the upcoming Digital Markets Act and Digital Services Act have changed the landscape of the technology industry. For instance, the GDPR covers personal data, and personal data is essentially defined as anything that can identify an individual (European Commission, n.d.B), and there are hefty violations for not protecting it or for using it in models without caution. Furthermore, with the European Union working intensively on regulating AI and its related techniques, including ML (European Commission, n.d.A), having a legal team specialized in this topic will become imperative for T-Superstore.

Besides legal concerns, the use of ML can bring upon ethical dilemmas. From job creation, privacy concerns, models that reinforce discrimination, to lack of accountability, AI and ML techniques can be an ethical minefield (IBM Cloud Education, 2020). For instance, the SVM discovered some words that affect the classification method. What would happen if one of the suppliers discovered these words, and started to swarm its products with them? Or similarly if the employees or management themselves start changing the descriptions to favor certain products over others?

Similarly, as explained in the report, the Naïve Bayes machine doesn't tell you how important the features are. So, what would happen if a supplier complained about its products not getting advertised and not being able to explain them why? These kinds of issues are the ethical dilemmas involving ML, and small issues can easily escalate into big problems. That is why T-Superstore should work on building a strong ethical foundation with clear goals and principles that will prevent this organization from becoming the next big controversy.

## **Final Notes**

To finalize this report, there are two more things that management must be aware of when it comes to Machine Learning and AI if it wants them to succeed. First, implementing new technologies takes time and escalating efforts. T-Superstore will not become Amazon the day after implementing its first model. Building a model that is not only reliable but fulfilling of its legal requirements requires effort. It will also require slow steps rather than a daunting dive into automatizing all the company's processes. The journey towards building a competitive advantage with ML is one of thousand steps, and a firm cannot rush them.

The second aspect to consider is that Machine Learning, AI, Neural Networks, any of these technologies are not for predicting the future. All these are tools for managing risk and making informed decisions. Models return labels, but inside of them there are probabilities, and the model is only so sure of the labels it returned. For example, maybe some of the labels will have a 99% level of confidence, but some others will have 50% or even less. That is why management must accept that even a very dependable model will fail from time to time. It might be even more strange that it does not. Anyways, the goal of the model is not to discover the fate of every product that enters but being able to assess the likelihood of a new product becoming a hot seller by checking thousands of records. In the end, more than the labels, the real value of the model is the support that it gives to the decision maker.

## References

- Abraham, M. and Lodish, L. M. (1990) *Getting the Most Out of Advertising and Promotion*. Harvard Business Review. Available at: <https://hbr.org/1990/05/getting-the-most-out-of-advertising-and-promotion?registration=success> (Accessed 16 April 2022)
- Akter, S. and Fosso Wamba, S. (2016) "Big data analytics in E-commerce: a systematic review and agenda for future research". *Electronic Markets*. 26, pp. 173-194. Available at: <https://link.springer.com/article/10.1007/s12525-016-0219-0> (Accessed 14 April 2022)
- Apparatus (n.d.) *The application of Big Data for ecommerce*. Apparatus. Available at: <https://www.apparatus.io/The-application-of-Big-Data-for-ecommerce#gref> (Accessed 14 April 2022)
- Aznar, P. (2020) "Decision Trees: Gini vs Entropy". *Quantdare*. 12 February. Available at: <https://quantdare.com/decision-trees-gini-vs-entropy/> (Accessed 09 April 2022)
- Bedi, G. (2018) "A guide to Text Classification (NLP) using SVM and Naïve Bayes with Python". *Medium*. 09 November. Available at: <https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34> (Accessed 09 April 2022)
- Castellion, G. and Markham, S. (2013) "Perspective: New Product Failure Rates: Influence of Argumentum ad Populum and Self-Interest". *Journal of Product Innovation and Management*. 30, pp. 976-979. Available at: <https://newproductsuccess.org/new-product-failure-rates-2013-jpim-30-pp-976-979/> (Accessed 14 April 2022)
- Cloudian (n.d.) *Eight Storage Requirements for Artificial Intelligence and Deep Learning*. Cloudian. Available at: <https://cloudian.com/resource/data-sheets/eight-storage-requirements-artificial-intelligence-deep-learning/> (Accessed 17 April 2022)
- Cosic et al. (2015) "A business analytics capability framework". *Australasian Journal of Information Systems*. 19. Pp. S5-S19. Available at: <https://journal.acs.org.au/index.php/ajis/article/view/1150> (Accessed 17 April 2022)
- European Commission (n.d.A) *A European approach to artificial intelligence*. European Commission. Available at: <https://digital-strategy.ec.europa.eu/en/policies/european-approach-artificial-intelligence> (Accessed 17 April 2022)
- European Commission (n.d.B) *What is personal data?* European Commission. Available at: [https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-personal-data\\_en](https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-personal-data_en) (Accessed 17 April 2022)
- Farber, R. (2017) "Technology Requirements for Deep and Machine Learning". *The Next Platform*. 14 July. Available at: <https://www.nextplatform.com/2017/07/14/technology-requirements-deep-machine-learning/> (Accessed 17 April 2022)
- Frederiksen, L and Waffle, K. (2021) "Cost and Benefits of Market Research". *Hinge Marketing*. 16 August. Available at:

[https://hingemarketing.com/blog/story/cost\\_and\\_benefits\\_of\\_market\\_research](https://hingemarketing.com/blog/story/cost_and_benefits_of_market_research) (Accessed 14 April 2022)

GeeksforGeeks (2021) *Python / Lemmatization with NLTK*. Available at: <https://www.geeksforgeeks.org/python-lemmatization-with-nltk/> (Accessed 10 April 2022)

IBM Cloud Education (2020) “Machine Learning”. IBM. 15 July. Available at: <https://www.ibm.com/cloud/learn/machine-learning> (Accessed 17 April 2022)

Jablonski, J. (2021) “Natural Language Processing With Python’s NLTK Package”. *Real Python*. 05 May. Available at: <https://realpython.com/nltk-nlp-python/> (Accessed 10 April 2022)

Kumar, S. (2020) “Feature Importance in Naïve Bayes Classifiers”. *inBlog*. 31 October. Available at: <https://blog.ineuron.ai/Feature-Importance-in-Naive-Bayes-Classifiers-5qob5d5sFW> (Accessed 10 April 2022)

Magusara, M. (2019) “Advantages and Disadvantages of Doing Market Research Before Starting a Business”. *Business 2 Community*. 17 April. Available at: <https://www.business2community.com/strategy/advantages-and-disadvantages-of-doing-market-research-before-starting-a-business-02192153> (Accessed 14 April 2022)

Malik, U. (n.d.) “Removing Stop Words from Strings in Python”. *StackAbuse*. Available at: <https://stackabuse.com/removing-stop-words-from-strings-in-python/> (Accessed 10 April 2022)

Navlani, A. (2018) “Decision Tree Classification in Python Tutorial”. *Datacamp*. 28 December. Available at: <https://www.datacamp.com/community/tutorials/decision-tree-classification-python> (Accessed 09 April 2022)

Policarpo, L.M. et. al (2021) “Machine learning through the lens of e-commerce initiatives: An up-to-date systematic literature review”. *Computer Science Review*. 41. Available at: <https://www.sciencedirect.com/science/article/pii/S157401372100054X> (Accessed 14 April 2022)

Saja, A., Yip, A. and Madi, E. (2019) *Workout supplement and nutrition products*. Kaggle. Available at: <https://www.kaggle.com/datasets/afsaja/workout-supplements-and-nutrition-products> (Accessed 09 April 2022)

Scikit-learn (n.d.A) *Decision Trees*. Scikit-learn. Available at: <https://scikit-learn.org/stable/modules/tree.html> (Accessed 09 April 2022)

Scikit-learn (n.d.B) *Ensemble methods*. Scikit-learn. Available at: <https://scikit-learn.org/stable/modules/ensemble.html#forest> (Accessed 09 April 2022)

Scikit-learn (n.d.C) *Metrics and scoring: quantifying the quality of predictions*. Scikit-learn. Available at: [https://scikit-learn.org/stable/modules/model\\_evaluation.html](https://scikit-learn.org/stable/modules/model_evaluation.html) (Accessed 09 April 2022)

Scikit-learn (n.d.D) *Naïve Bayes*. Scikit-learn. Available at: [https://scikit-learn.org/stable/modules/naive\\_bayes.html](https://scikit-learn.org/stable/modules/naive_bayes.html) (Accessed 09 April 2022)

Scikit-learn (n.d.E) *Permutation Feature Importance*. Scikit-learn. Available at: [https://scikit-learn.org/stable/modules/permutation\\_importance.html#permutation-importance](https://scikit-learn.org/stable/modules/permutation_importance.html#permutation-importance) (Accessed 10 April 2022)

Scikit-learn (n.d.F) *Probability Calibration*. Scikit-learn. Available at: <https://scikit-learn.org/stable/modules/calibration.html> (Accessed 09 April 2022)

Scikit-learn (n.d.G) *Support Vector Machines*. Scikit-learn. Available at: <https://scikit-learn.org/stable/modules/svm.html#scores-probabilities> (Accessed 10 April 2022)

Schneider, J. and Hall, J. (2011) *Why Most Product Launches Fail*. Harvard Business Review. Available at: <https://hbr.org/2011/04/why-most-product-launches-fail> (Accessed 14 April 2022)

Shaw, N. (n.d.) “Ecommerce Machine Learning: AI’s Role in the Future of Online Shopping” *BIGCOMMERCE*. Available at: <https://www.bigcommerce.com/blog/ecommerce-machine-learning/#conclusion> (Accessed 14 April 2022)

Shukla, P. and Iriondo, R. (2020) “Natural Language Processing (NLP) with Python – Tutorial”. *Towards AI*. 22 July. Available at: <https://towardsai.net/p/nlp/natural-language-processing-nlp-with-python-tutorial-for-beginners-1f54e610a1a0#7458> (Accessed 10 April 2022)

Stecanella, B. (2019) “Understanding TF-IDF: A Simple Introduction”. *MonkeyLearn*. 10 May. Available at: <https://monkeylearn.com/blog/what-is-tf-idf/> (Accessed 18 April 2022)

Surbhi\_22 (2021) “A Guide to the Naïve Bayes Algorithm”. *Analytics Vidhya*. 16 January. Available at: <https://www.analyticsvidhya.com/blog/2021/01/a-guide-to-the-naive-bayes-algorithm/> (Accessed 09 April 2022)

Target (n.d.) *Target*. Available at: <https://www.target.com/> (Accessed 14 April 2022)

Wahner, K. (2017) “Data Preprocessing vs. Data Wrangling in Machine Learning Projects”. *InfoQ*. 05 March. Available at: <https://www.infoq.com/articles/ml-data-processing/> (Accessed 17 April 2022)