Classification

May 28, 2022

[190]:
```
#pip install --upgrade scikit-learn
```

```
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/9d/20/0ffe8665a44bce7616bd
33d4368a198fecad3b226bcafa38c63ef0f6286f/scikit_learn-1.0.2-cp37-cp37m-win_amd64
.whl (7.1MB)
Requirement already satisfied, skipping upgrade: numpy>=1.14.6 in
c:\users\jesus\anaconda3\lib\site-packages (from scikit-learn) (1.21.5)
Requirement already satisfied, skipping upgrade: scipy>=1.1.0 in
c:\users\jesus\anaconda3\lib\site-packages (from scikit-learn) (1.3.1)
Collecting threadpoolctl>=2.0.0 (from scikit-learn)
  Downloading https://files.pythonhosted.org/packages/61/cf/6e354304bcb9c6413c4e
02a747b600061c21d38ba51e7e544ac7bc66aecc/threadpoolctl-3.1.0-py3-none-any.whl
Requirement already satisfied, skipping upgrade: joblib>=0.11 in
c:\users\jesus\anaconda3\lib\site-packages (from scikit-learn) (0.13.2)
Installing collected packages: threadpoolctl, scikit-learn
  Found existing installation: scikit-learn 0.21.3
    Uninstalling scikit-learn-0.21.3:
      Successfully uninstalled scikit-learn-0.21.3
Note: you may need to restart the kernel to use updated packages.

ERROR: Could not install packages due to an EnvironmentError: [WinError 5]
Access is denied: 'c:\\users\\jesus\\anaconda3\\lib\\site-
packages\\~klearn\\datasets\\_svmlight_format.cp37-win_amd64.pyd'
Consider using the `--user` option or check the permissions.
```

[1]:
```
import nltk
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Jesus\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Jesus\AppData\Roaming\nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Jesus\AppData\Roaming\nltk_data…
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Jesus\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

[1]: True

[40]:
```
#pip install pydotplus
```

```
Requirement already satisfied: pydotplus in c:\users\jesus\anaconda3\lib\site-
packages (2.0.2)
Requirement already satisfied: pyparsing>=2.0.1 in
c:\users\jesus\anaconda3\lib\site-packages (from pydotplus) (2.4.2)
Note: you may need to restart the kernel to use updated packages.
```

[39]:
```
#pip install graphviz
```

```
Requirement already satisfied: graphviz in c:\users\jesus\anaconda3\lib\site-
packages (0.19.2)
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
import pandas as pd
import numpy as np
from sklearn import metrics
import matplotlib.pyplot as plt
#Modules for training data
from sklearn.model_selection import train_test_split
#Tree modules
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.tree import export_graphviz
from six import StringIO
from IPython.display import Image
import pydotplus
#Naive Bayes Module
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import OneHotEncoder
from sklearn.calibration import CalibratedClassifierCV
import sklearn.inspection
```

[3]:
```
raw_prod = pd.read_csv("products.csv")
```

[4]:
```
raw_prod.loc[4,"product_description"]
```

[4]:
```
'24g of Pure, Quality Protein in Every Scoop with No Added Amino Acids or Filler
Nutrients'
```

[5]:
```
raw_prod.columns
```

[5]:
```
Index(['average_flavor_rating', 'brand_name', 'link', 'number_of_flavors',
       'number_of_reviews', 'overall_rating', 'price', 'price_per_serving',
       'product_category', 'product_description', 'product_name',
       'top_flavor_rated', 'verified_buyer_number', 'verified_buyer_rating'],
      dtype='object')
```

[6]:
```
#Cleaning columns only to keep those with the desired variables and with␣
 ↪complete information
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)
```

[7]:
```
#Dropping columns without complete values
raw_prod.dropna(inplace = True)
```

[8]:
```
#Final PreData
raw_prod
```

[8]:
```
                 brand_name  number_of_flavors  overall_rating  price  \
0         EVLUTION NUTRITION               29.0             9.4  19.99
1          Optimum Nutrition               43.0             9.3  57.99
2      JYM Supplement Science                9.0             9.1  48.99
4      JYM Supplement Science               14.0             9.2  56.98
7         EVLUTION NUTRITION                6.0             9.3  34.99
..                       …                  …               …      …
819                   Ascent                2.0             8.8  47.18
824                 Cellucor               10.0             9.4  16.99
825                   Ascent                4.0             9.6  75.80
826                  Isopure                2.0             8.4  41.07
830                     Vega                4.0             9.0  32.87

             product_category  \
0                       BCAAs
1        Build Muscle Products
2      Improve Workout Products
4          Whey Protein Isolate
7             Betaine Anhydrous
..                          …
819     Micellar Casein Protein
824               Beta-Alanine
825        Whey Protein Isolate
826        Whey Protein Isolate
```

```
830             Plant Protein

                                          product_description
0    BCAA Powder with Natural Energizers Sourced fr…
1    24g of Whey Protein with Amino Acids for Muscl…
2    Pre-Workout Powder Powerhouse Packed with 13-H…
4    24g of Pure, Quality Protein in Every Scoop wi…
7             Advanced Pre-Workout + Weight Management
..                                                 …
819  Slow And Sustained Release To Keep Muscles Fed…
824  Pre-Mix Pre-Workout for Energy, Focus and Ulti…
825  Made with Zero Artificial Ingredients and Nati…
826                                           Natural!
830                                Plant-based Protein!

[303 rows x 6 columns]
```

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

```
[10]: raw_prod["label"].value_counts()
```

```
[10]: 1    184
      0    119
      Name: label, dtype: int64
```

```
[11]: raw_prod
```

```
[11]:                brand_name  number_of_flavors  overall_rating  price  \
      0       EVLUTION NUTRITION               29.0             9.4  19.99
      1         Optimum Nutrition               43.0             9.3  57.99
      2    JYM Supplement Science                9.0             9.1  48.99
      4    JYM Supplement Science               14.0             9.2  56.98
      7       EVLUTION NUTRITION                6.0             9.3  34.99
      ..                      …                 …               …      …
      819                  Ascent                2.0             8.8  47.18
      824                Cellucor               10.0             9.4  16.99
      825                  Ascent                4.0             9.6  75.80
      826                 Isopure                2.0             8.4  41.07
      830                    Vega                4.0             9.0  32.87

                   product_category  \
      0                       BCAAs
      1        Build Muscle Products
      2    Improve Workout Products
      4          Whey Protein Isolate
      7            Betaine Anhydrous
      ..                          …
```

```
819   Micellar Casein Protein
824              Beta-Alanine
825      Whey Protein Isolate
826      Whey Protein Isolate
830             Plant Protein

                                   product_description  label
0    BCAA Powder with Natural Energizers Sourced fr…      1
1    24g of Whey Protein with Amino Acids for Muscl…      1
2    Pre-Workout Powder Powerhouse Packed with 13-H…      1
4    24g of Pure, Quality Protein in Every Scoop wi…      1
7             Advanced Pre-Workout + Weight Management      1
..                                                 …    …
819  Slow And Sustained Release To Keep Muscles Fed…      0
824  Pre-Mix Pre-Workout for Energy, Focus and Ulti…      1
825  Made with Zero Artificial Ingredients and Nati…      1
826                                           Natural!      0
830                               Plant-based Protein!      1

[303 rows x 7 columns]
```

1 Decision Tree

```
[12]: #Columns for the Decision Tree
dt_prod = raw_prod.copy()
dt_prod.drop(["overall_rating", "product_description"], axis = 1, inplace =␣
 ↪True)
```

```
[13]: dt_prod
```

```
[13]:                brand_name  number_of_flavors  price  \
0          EVLUTION NUTRITION               29.0  19.99
1           Optimum Nutrition               43.0  57.99
2     JYM Supplement Science                9.0  48.99
4     JYM Supplement Science               14.0  56.98
7          EVLUTION NUTRITION                6.0  34.99
..                       …                  …    …
819                   Ascent                2.0  47.18
824                 Cellucor               10.0  16.99
825                   Ascent                4.0  75.80
826                  Isopure                2.0  41.07
830                     Vega                4.0  32.87

             product_category  label
0                       BCAAs      1
1        Build Muscle Products      1
```

```
2    Improve Workout Products      1
4        Whey Protein Isolate      1
7           Betaine Anhydrous      1
..                        …    …
819   Micellar Casein Protein      0
824               Beta-Alanine      1
825       Whey Protein Isolate      1
826       Whey Protein Isolate      0
830              Plant Protein      1

[303 rows x 5 columns]
```

[14]:
```
#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)
dt_prod.columns
```

[14]:
```
Index(['brand_name', 'number_of_flavors', 'price', 'product_category', 'label',
       'Brand: _ABB', 'Brand: _AST', 'Brand: _AllMax Nutrition',
       'Brand: _Animal', 'Brand: _Ascent', 'Brand: _BSN',
       'Brand: _Beast Sports Nutrition', 'Brand: _Betancourt Nutrition',
       'Brand: _Beverly International', 'Brand: _Body Nutrition',
       'Brand: _Bodybuilding.com Signature', 'Brand: _COBRA LABS',
       'Brand: _Cellucor', 'Brand: _Celsius', 'Brand: _Core Nutritionals',
       'Brand: _CytoSport', 'Brand: _Dymatize', 'Brand: _EFX Sports',
       'Brand: _EVLUTION NUTRITION', 'Brand: _FINAFLEX', 'Brand: _GAT',
       'Brand: _Gamma Labs', 'Brand: _Garden Of Life',
       'Brand: _Gaspari Nutrition', 'Brand: _Grenade', 'Brand: _Isopure',
       'Brand: _JYM Supplement Science', 'Brand: _Kaged Muscle',
       'Brand: _Labrada', 'Brand: _Lenny & Larry's', 'Brand: _MET-Rx',
       'Brand: _MHP', 'Brand: _MRM', 'Brand: _Magnum Nutraceuticals',
       'Brand: _Muscle Beach Nutrition', 'Brand: _Muscle Milk',
       'Brand: _MuscleMeds', 'Brand: _MuscleTech', 'Brand: _NLA for Her',
       'Brand: _NOW', 'Brand: _NutraBio', 'Brand: _ONE',
       'Brand: _OhYeah! Nutrition', 'Brand: _Optimum Nutrition',
       'Brand: _PEScience', 'Brand: _PrimaForce', 'Brand: _Pro Supps',
       'Brand: _Quest Nutrition', 'Brand: _RSP Nutrition', 'Brand: _S.A.N.',
       'Brand: _Six Star Pro Nutrition', 'Brand: _Sports Research',
       'Brand: _Top Secret Nutrition', 'Brand: _Universal Nutrition',
       'Brand: _Vega', 'Brand: _eFlow Nutrition', 'Brand: _iForce Nutrition',
       'Brand: _iSatori', 'Category: _Agmatine', 'Category: _Amino Acids',
       'Category: _BCAAs', 'Category: _Beef Protein',
       'Category: _Beta-Alanine', 'Category: _Betaine Anhydrous',
       'Category: _Build Muscle Products', 'Category: _Caffeine',
       'Category: _Carbohydrates', 'Category: _Citrulline',
       'Category: _Collagen', 'Category: _Creatine', 'Category: _Creatine HCl',
       'Category: _Creatine Malate', 'Category: _Creatine Monohydrate',
       'Category: _D-Aspartic Acid', 'Category: _Egg Protein',
       'Category: _GABA', 'Category: _Glutamine',
       'Category: _Green Coffee Extract', 'Category: _Green Tea',
       'Category: _Hydrolyzed Whey Protein',
       'Category: _Improve Workout Products', 'Category: _Kre-Alkalyn',
       'Category: _L-Arginine', 'Category: _L-Taurine',
       'Category: _Micellar Casein Protein', 'Category: _Plant Protein',
       'Category: _Protein', 'Category: _Waxy Maize',
       'Category: _Weight Loss Products', 'Category: _Whey Protein',
       'Category: _Whey Protein Blends', 'Category: _Whey Protein Concentrate',
       'Category: _Whey Protein Isolate', 'Category: _Yerba Mate'],
      dtype='object')
```

[15]:
```
#Creating the training data for the decision tree
dt_feature_cols = ['number_of_flavors', 'price','Brand: _ABB', 'Brand: _AST',␣
 ↪'Brand: _AllMax Nutrition',
       'Brand: _Animal', 'Brand: _Ascent', 'Brand: _BSN',
       'Brand: _Beast Sports Nutrition', 'Brand: _Betancourt Nutrition',
       'Brand: _Beverly International', 'Brand: _Body Nutrition',
       'Brand: _Bodybuilding.com Signature', 'Brand: _COBRA LABS',
       'Brand: _Cellucor', 'Brand: _Celsius', 'Brand: _Core Nutritionals',
       'Brand: _CytoSport', 'Brand: _Dymatize', 'Brand: _EFX Sports',
       'Brand: _EVLUTION NUTRITION', 'Brand: _FINAFLEX', 'Brand: _GAT',
       'Brand: _Gamma Labs', 'Brand: _Garden Of Life',
       'Brand: _Gaspari Nutrition', 'Brand: _Grenade', 'Brand: _Isopure',
       'Brand: _JYM Supplement Science', 'Brand: _Kaged Muscle',
       'Brand: _Labrada', '''Brand: _Lenny & Larry's''', 'Brand: _MET-Rx',
       'Brand: _MHP', 'Brand: _MRM', 'Brand: _Magnum Nutraceuticals',
       'Brand: _Muscle Beach Nutrition', 'Brand: _Muscle Milk',
       'Brand: _MuscleMeds', 'Brand: _MuscleTech', 'Brand: _NLA for Her',
       'Brand: _NOW', 'Brand: _NutraBio', 'Brand: _ONE',
       'Brand: _OhYeah! Nutrition', 'Brand: _Optimum Nutrition',
       'Brand: _PEScience', 'Brand: _PrimaForce', 'Brand: _Pro Supps',
       'Brand: _Quest Nutrition', 'Brand: _RSP Nutrition', 'Brand: _S.A.N.',
       'Brand: _Six Star Pro Nutrition', 'Brand: _Sports Research',
       'Brand: _Top Secret Nutrition', 'Brand: _Universal Nutrition',
       'Brand: _Vega', 'Brand: _eFlow Nutrition', 'Brand: _iForce Nutrition',
       'Brand: _iSatori', 'Category: _Agmatine', 'Category: _Amino Acids',
       'Category: _BCAAs', 'Category: _Beef Protein',
       'Category: _Beta-Alanine', 'Category: _Betaine Anhydrous',
       'Category: _Build Muscle Products', 'Category: _Caffeine',
       'Category: _Carbohydrates', 'Category: _Citrulline',
       'Category: _Collagen', 'Category: _Creatine', 'Category: _Creatine HCl',
       'Category: _Creatine Malate', 'Category: _Creatine Monohydrate',
       'Category: _D-Aspartic Acid', 'Category: _Egg Protein',
       'Category: _GABA', 'Category: _Glutamine',
       'Category: _Green Coffee Extract', 'Category: _Green Tea',
       'Category: _Hydrolyzed Whey Protein',
       'Category: _Improve Workout Products', 'Category: _Kre-Alkalyn',
       'Category: _L-Arginine', 'Category: _L-Taurine',
       'Category: _Micellar Casein Protein', 'Category: _Plant Protein',
       'Category: _Protein', 'Category: _Waxy Maize',
       'Category: _Weight Loss Products', 'Category: _Whey Protein',
       'Category: _Whey Protein Blends', 'Category: _Whey Protein Concentrate',
       'Category: _Whey Protein Isolate', 'Category: _Yerba Mate']
dt_X = dt_prod[dt_feature_cols]
dt_Y = dt_prod.label
```

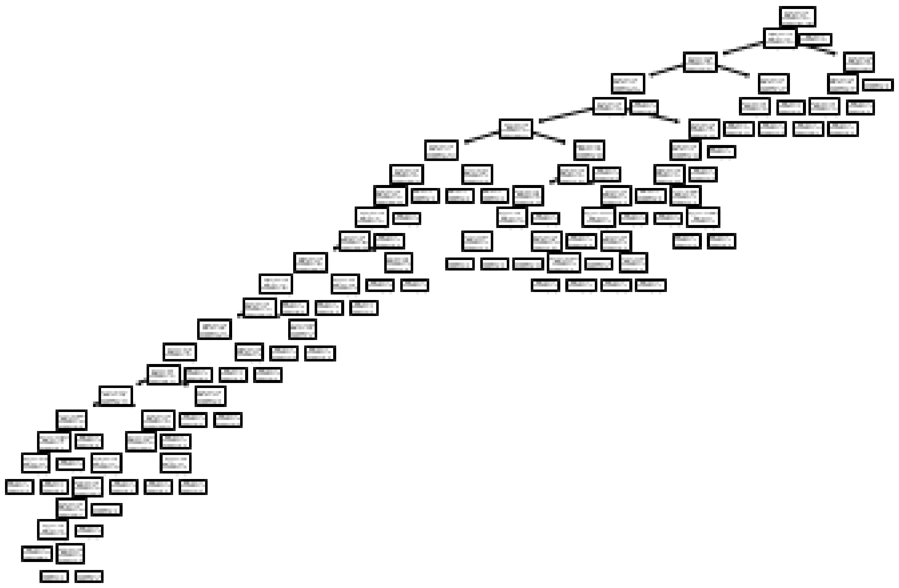
[16]:
```
#Building the training dataset

X_train, X_test, Y_train, Y_test = train_test_split(dt_X, dt_Y, test_size = 0.
 ↪3, random_state = 1996)
```

[17]:
```
#Decision Tree (If reset, run above before)
#Summoning Machine
#Criterion and Max_Depth
tree_prod = DecisionTreeClassifier(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)
```

[18]:
```
#Checking accuracy for products with review higher than 9
print("Accuracy: ", metrics.accuracy_score(Y_test, Y_pred))
```

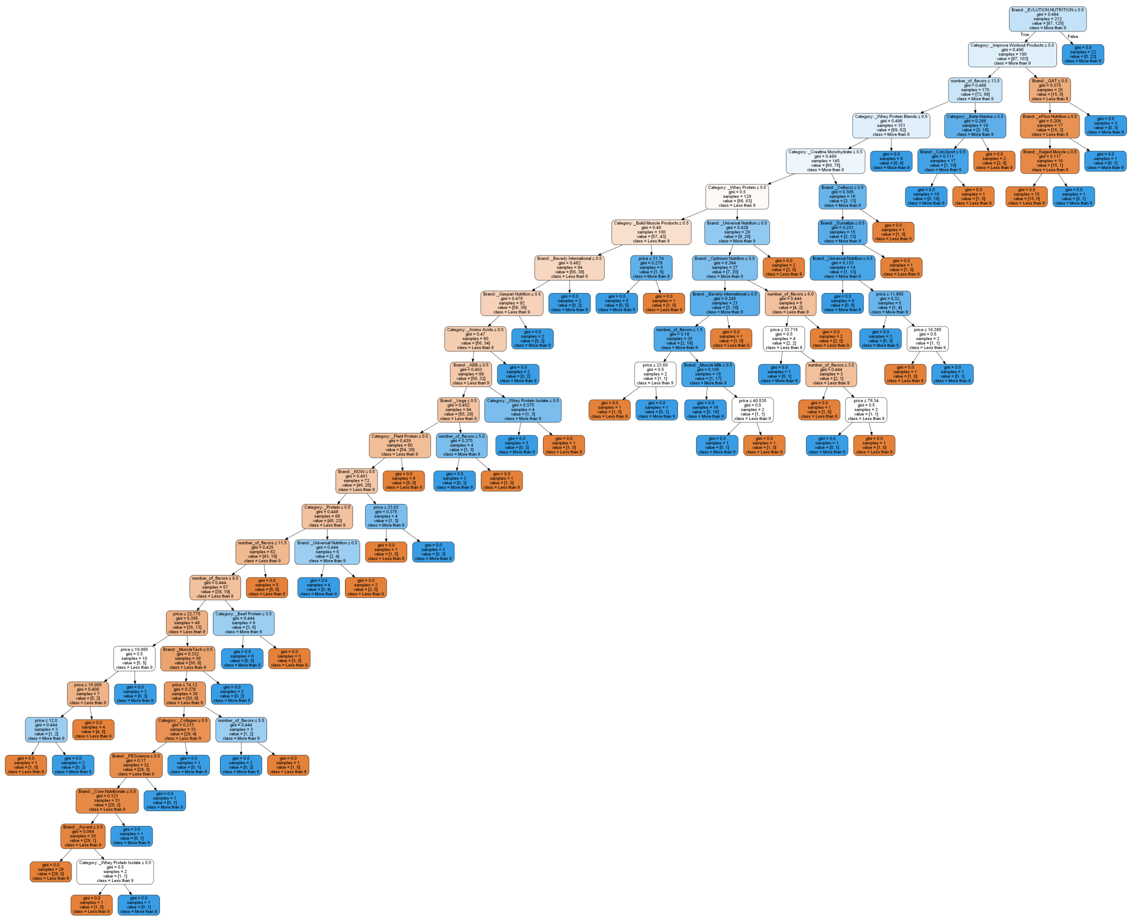
```
Accuracy:  0.6813186813186813
```

[19]:
```
tree.plot_tree(tree_prod)
plt.show()
```



```
[20]: 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_large.png")
      Image(graph.create_png())
```

[20]:



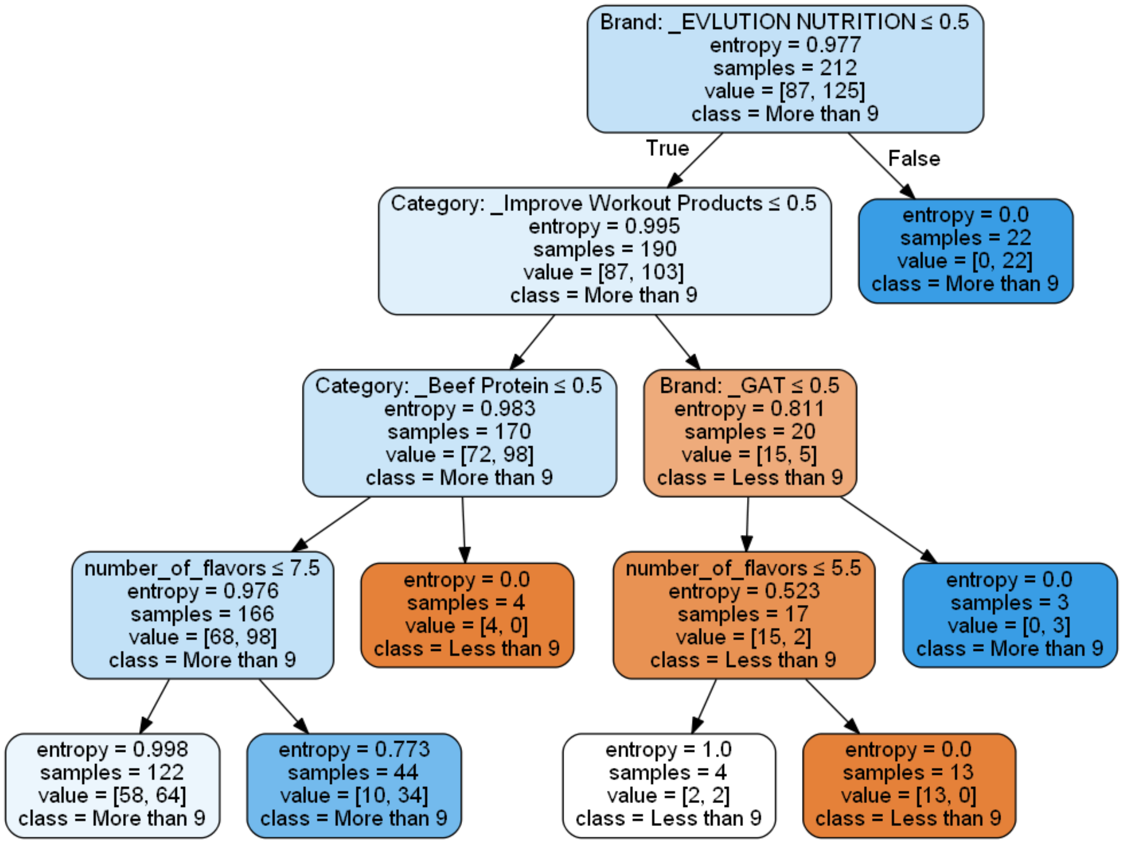
[21]:
```
#Evaluating the tree
print("Confusion Matrix Tree : \n", metrics.confusion_matrix(Y_test,␣
↪Y_pred),"\n")
print("The precision for Tree is ", metrics.precision_score(Y_test, Y_pred))
print("The recall for Tree is ", metrics.recall_score(Y_test, Y_pred),"\n")
print("The accuracy for Tree is ", metrics.accuracy_score(Y_test, Y_pred),"\n")
print("The error rate for Tree is ", (1 - metrics.accuracy_score(Y_test,␣
↪Y_pred)),"\n")
print("The F-score for Tree is ", metrics.f1_score(Y_test, Y_pred),"\n")
```

```
Confusion Matrix Tree :
 [[17 15]
 [14 45]]

The precision for Tree is  0.75
The recall for Tree is  0.7627118644067796

The accuracy for Tree is  0.6813186813186813
The error rate for Tree is  0.31868131868131866

The F-score for Tree is  0.7563025210084034
```

[22]:
```
#Entropy
#Decision Tree (If reset, run above before)
#Summoning Machine
#Criterion and Max_Depth
tree_prod = DecisionTreeClassifier(criterion = "entropy", 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)

#Checking accuracy for products with review higher than 9
print("Accuracy: ", metrics.accuracy_score(Y_test, Y_pred))

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("Products2.png")
Image(graph.create_png())
```

```
Accuracy:  0.6703296703296703
```

[22]:



2 Naive Bayesian

[23]:
```
#Preparing data
nb_prod = raw_prod.copy()
nb_prod.drop(["overall_rating", "product_description"], axis = 1, inplace =␣
 ↪True)
```

[24]:
```
nb_prod
```

[24]:
```
                  brand_name  number_of_flavors  price  \
0         EVLUTION NUTRITION               29.0  19.99
1          Optimum Nutrition               43.0  57.99
2      JYM Supplement Science               9.0  48.99
4      JYM Supplement Science              14.0  56.98
7         EVLUTION NUTRITION                6.0  34.99
..                       …                  …    …
819                   Ascent                2.0  47.18
824                 Cellucor               10.0  16.99
825                   Ascent                4.0  75.80
```

```
826                Isopure              2.0  41.07
830                   Vega              4.0  32.87

              product_category  label
0                        BCAAs      1
1         Build Muscle Products      1
2      Improve Workout Products      1
4          Whey Protein Isolate      1
7             Betaine Anhydrous      1
..                         …      …
819     Micellar Casein Protein      0
824                Beta-Alanine      1
825        Whey Protein Isolate      1
826        Whey Protein Isolate      0
830               Plant Protein      1

[303 rows x 5 columns]
```

[25]:
```
#Label Encoder for categories
#One_hot
onehot_brand = pd.get_dummies(nb_prod["brand_name"], prefix = "Brand: ")
onehot_category = pd.get_dummies(nb_prod["product_category"], prefix =␣
 ↪"Category: ")
nb_prod = nb_prod.join(onehot_brand)
nb_prod = nb_prod.join(onehot_category)
nb_prod.columns
```

[25]:
```
Index(['brand_name', 'number_of_flavors', 'price', 'product_category', 'label',
       'Brand: _ABB', 'Brand: _AST', 'Brand: _AllMax Nutrition',
       'Brand: _Animal', 'Brand: _Ascent', 'Brand: _BSN',
       'Brand: _Beast Sports Nutrition', 'Brand: _Betancourt Nutrition',
       'Brand: _Beverly International', 'Brand: _Body Nutrition',
       'Brand: _Bodybuilding.com Signature', 'Brand: _COBRA LABS',
       'Brand: _Cellucor', 'Brand: _Celsius', 'Brand: _Core Nutritionals',
       'Brand: _CytoSport', 'Brand: _Dymatize', 'Brand: _EFX Sports',
       'Brand: _EVLUTION NUTRITION', 'Brand: _FINAFLEX', 'Brand: _GAT',
       'Brand: _Gamma Labs', 'Brand: _Garden Of Life',
       'Brand: _Gaspari Nutrition', 'Brand: _Grenade', 'Brand: _Isopure',
       'Brand: _JYM Supplement Science', 'Brand: _Kaged Muscle',
       'Brand: _Labrada', 'Brand: _Lenny & Larry's', 'Brand: _MET-Rx',
       'Brand: _MHP', 'Brand: _MRM', 'Brand: _Magnum Nutraceuticals',
       'Brand: _Muscle Beach Nutrition', 'Brand: _Muscle Milk',
       'Brand: _MuscleMeds', 'Brand: _MuscleTech', 'Brand: _NLA for Her',
       'Brand: _NOW', 'Brand: _NutraBio', 'Brand: _ONE',
       'Brand: _OhYeah! Nutrition', 'Brand: _Optimum Nutrition',
       'Brand: _PEScience', 'Brand: _PrimaForce', 'Brand: _Pro Supps',
       'Brand: _Quest Nutrition', 'Brand: _RSP Nutrition', 'Brand: _S.A.N.',
       'Brand: _Six Star Pro Nutrition', 'Brand: _Sports Research',
       'Brand: _Top Secret Nutrition', 'Brand: _Universal Nutrition',
       'Brand: _Vega', 'Brand: _eFlow Nutrition', 'Brand: _iForce Nutrition',
       'Brand: _iSatori', 'Category: _Agmatine', 'Category: _Amino Acids',
       'Category: _BCAAs', 'Category: _Beef Protein',
       'Category: _Beta-Alanine', 'Category: _Betaine Anhydrous',
       'Category: _Build Muscle Products', 'Category: _Caffeine',
       'Category: _Carbohydrates', 'Category: _Citrulline',
       'Category: _Collagen', 'Category: _Creatine', 'Category: _Creatine HCl',
       'Category: _Creatine Malate', 'Category: _Creatine Monohydrate',
       'Category: _D-Aspartic Acid', 'Category: _Egg Protein',
       'Category: _GABA', 'Category: _Glutamine',
       'Category: _Green Coffee Extract', 'Category: _Green Tea',
       'Category: _Hydrolyzed Whey Protein',
       'Category: _Improve Workout Products', 'Category: _Kre-Alkalyn',
       'Category: _L-Arginine', 'Category: _L-Taurine',
       'Category: _Micellar Casein Protein', 'Category: _Plant Protein',
       'Category: _Protein', 'Category: _Waxy Maize',
       'Category: _Weight Loss Products', 'Category: _Whey Protein',
       'Category: _Whey Protein Blends', 'Category: _Whey Protein Concentrate',
       'Category: _Whey Protein Isolate', 'Category: _Yerba Mate'],
      dtype='object')
```

[26]:
```
#Creating Data
nb_feature_cols = ['number_of_flavors', 'price','Brand: _ABB', 'Brand: _AST',␣
 ↪'Brand: _AllMax Nutrition',
       'Brand: _Animal', 'Brand: _Ascent', 'Brand: _BSN',
       'Brand: _Beast Sports Nutrition', 'Brand: _Betancourt Nutrition',
       'Brand: _Beverly International', 'Brand: _Body Nutrition',
       'Brand: _Bodybuilding.com Signature', 'Brand: _COBRA LABS',
       'Brand: _Cellucor', 'Brand: _Celsius', 'Brand: _Core Nutritionals',
       'Brand: _CytoSport', 'Brand: _Dymatize', 'Brand: _EFX Sports',
       'Brand: _EVLUTION NUTRITION', 'Brand: _FINAFLEX', 'Brand: _GAT',
       'Brand: _Gamma Labs', 'Brand: _Garden Of Life',
       'Brand: _Gaspari Nutrition', 'Brand: _Grenade', 'Brand: _Isopure',
       'Brand: _JYM Supplement Science', 'Brand: _Kaged Muscle',
       'Brand: _Labrada', '''Brand: _Lenny & Larry's''', 'Brand: _MET-Rx',
       'Brand: _MHP', 'Brand: _MRM', 'Brand: _Magnum Nutraceuticals',
       'Brand: _Muscle Beach Nutrition', 'Brand: _Muscle Milk',
       'Brand: _MuscleMeds', 'Brand: _MuscleTech', 'Brand: _NLA for Her',
       'Brand: _NOW', 'Brand: _NutraBio', 'Brand: _ONE',
       'Brand: _OhYeah! Nutrition', 'Brand: _Optimum Nutrition',
       'Brand: _PEScience', 'Brand: _PrimaForce', 'Brand: _Pro Supps',
       'Brand: _Quest Nutrition', 'Brand: _RSP Nutrition', 'Brand: _S.A.N.',
       'Brand: _Six Star Pro Nutrition', 'Brand: _Sports Research',
       'Brand: _Top Secret Nutrition', 'Brand: _Universal Nutrition',
       'Brand: _Vega', 'Brand: _eFlow Nutrition', 'Brand: _iForce Nutrition',
       'Brand: _iSatori', 'Category: _Agmatine', 'Category: _Amino Acids',
       'Category: _BCAAs', 'Category: _Beef Protein',
       'Category: _Beta-Alanine', 'Category: _Betaine Anhydrous',
       'Category: _Build Muscle Products', 'Category: _Caffeine',
       'Category: _Carbohydrates', 'Category: _Citrulline',
       'Category: _Collagen', 'Category: _Creatine', 'Category: _Creatine HCl',
       'Category: _Creatine Malate', 'Category: _Creatine Monohydrate',
       'Category: _D-Aspartic Acid', 'Category: _Egg Protein',
       'Category: _GABA', 'Category: _Glutamine',
       'Category: _Green Coffee Extract', 'Category: _Green Tea',
       'Category: _Hydrolyzed Whey Protein',
       'Category: _Improve Workout Products', 'Category: _Kre-Alkalyn',
       'Category: _L-Arginine', 'Category: _L-Taurine',
       'Category: _Micellar Casein Protein', 'Category: _Plant Protein',
       'Category: _Protein', 'Category: _Waxy Maize',
       'Category: _Weight Loss Products', 'Category: _Whey Protein',
       'Category: _Whey Protein Blends', 'Category: _Whey Protein Concentrate',
       'Category: _Whey Protein Isolate', 'Category: _Yerba Mate']
nb_X = dt_prod[dt_feature_cols]
nb_Y = dt_prod.label
```

[27]:
```
#Splitting
X_train, X_test, Y_train, Y_test = train_test_split(nb_X, nb_Y, test_size = 0.
 ↪3, random_state = 1996)
```

[28]:
```
#Standardizing
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

[29]:
```
#Doing Naive Bayesian
NB_machine = GaussianNB()

NB_machine = NB_machine.fit(X_train, Y_train)

Y_pred = NB_machine.predict(X_test)
```

[30]:
```
#Evaluating the NB
print("Confusion Matrix NB : \n", metrics.confusion_matrix(Y_test, Y_pred),"\n")
print("The precision for NB is ", metrics.precision_score(Y_test, Y_pred))
print("The recall for NB is ", metrics.recall_score(Y_test, Y_pred),"\n")
print("The accuracy for NB is ", metrics.accuracy_score(Y_test, Y_pred),"\n")
print("The error rate for NB is ", (1 - metrics.accuracy_score(Y_test,␣
 ↪Y_pred)),"\n")
print("The F-score for NB is ", metrics.f1_score(Y_test, Y_pred),"\n")
```

```
Confusion Matrix NB :
 [[26  6]
 [35 24]]

The precision for NB is  0.8
The recall for NB is  0.4067796610169492

The accuracy for NB is  0.5494505494505495

The error rate for NB is  0.4505494505494505

The F-score for NB is  0.5393258426966292
```

[31]:
```
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]]))
```

```
Feature ranking:
1. Brand: _EVLUTION NUTRITION (0.035165)
2. Brand: _JYM Supplement Science (0.021978)
3. Brand: _Muscle Beach Nutrition (0.019780)
4. Brand: _ABB (0.017582)
5. Brand: _Cellucor (0.017582)
6. Category: _Whey Protein (0.015385)
7. Category: _Plant Protein (0.010989)
8. Brand: _Labrada (0.010989)
9. Brand: _eFlow Nutrition (0.010989)
10. Category: _Glutamine (0.010989)
11. Brand: _Core Nutritionals (0.010989)
12. Brand: _Quest Nutrition (0.008791)
13. Brand: _Beast Sports Nutrition (0.008791)
14. Brand: _Six Star Pro Nutrition (0.008791)
15. Category: _Creatine HCl (0.008791)
16. Category: _Green Tea (0.006593)
17. Brand: _Dymatize (0.006593)
18. Category: _Caffeine (0.006593)
19. Category: _Build Muscle Products (0.004396)
20. Brand: _Grenade (0.004396)
21. Brand: _CytoSport (0.004396)
22. number_of_flavors (0.004396)
23. Category: _Green Coffee Extract (0.004396)
```

```
24. Category: _Amino Acids (0.002198)
25. Category: _Creatine Monohydrate (0.002198)
26. Brand: _MRM (0.002198)
27. Brand: _RSP Nutrition (0.002198)
28. Category: _Beef Protein (0.002198)
29. Category: _Beta-Alanine (0.002198)
30. Brand: _Muscle Milk (0.002198)
31. Brand: _MuscleMeds (0.002198)
32. Brand: _Kaged Muscle (0.000000)
33. Brand: _Magnum Nutraceuticals (0.000000)
34. price (0.000000)
35. Brand: _AST (0.000000)
36. Brand: _MuscleTech (0.000000)
37. Brand: _MHP (0.000000)
38. Brand: _AllMax Nutrition (0.000000)
39. Brand: _Lenny & Larry's (0.000000)
40. Brand: _Animal (0.000000)
41. Brand: _Betancourt Nutrition (0.000000)
42. Brand: _Ascent (0.000000)
43. Brand: _Celsius (0.000000)
44. Brand: _Beverly International (0.000000)
45. Brand: _Garden Of Life (0.000000)
46. Brand: _Gamma Labs (0.000000)
47. Brand: _GAT (0.000000)
48. Brand: _FINAFLEX (0.000000)
49. Brand: _Body Nutrition (0.000000)
50. Brand: _Bodybuilding.com Signature (0.000000)
51. Brand: _NOW (0.000000)
52. Brand: _EFX Sports (0.000000)
53. Brand: _COBRA LABS (0.000000)
54. Brand: _NLA for Her (0.000000)
55. Category: _Yerba Mate (0.000000)
56. Brand: _NutraBio (0.000000)
57. Category: _Improve Workout Products (0.000000)
58. Category: _Citrulline (0.000000)
59. Category: _Collagen (0.000000)
60. Category: _Creatine (0.000000)
61. Category: _Creatine Malate (0.000000)
62. Category: _Egg Protein (0.000000)
63. Category: _GABA (0.000000)
64. Category: _Hydrolyzed Whey Protein (0.000000)
65. Category: _Kre-Alkalyn (0.000000)
66. Category: _Betaine Anhydrous (0.000000)
67. Category: _L-Arginine (0.000000)
68. Category: _L-Taurine (0.000000)
69. Category: _Micellar Casein Protein (0.000000)
70. Category: _Protein (0.000000)
71. Category: _Waxy Maize (0.000000)
```

```
72. Category: _Weight Loss Products (0.000000)
73. Category: _Whey Protein Concentrate (0.000000)
74. Brand: _ONE (0.000000)
75. Category: _Carbohydrates (0.000000)
76. Category: _BCAAs (0.000000)
77. Brand: _Top Secret Nutrition (0.000000)
78. Brand: _OhYeah! Nutrition (0.000000)
79. Brand: _Optimum Nutrition (0.000000)
80. Category: _Whey Protein Isolate (0.000000)
81. Brand: _Pro Supps (0.000000)
82. Brand: _S.A.N. (0.000000)
83. Category: _Agmatine (0.000000)
84. Brand: _Sports Research (0.000000)
85. Brand: _Universal Nutrition (0.000000)
86. Brand: _Vega (0.000000)
87. Brand: _iForce Nutrition (0.000000)
88. Brand: _iSatori (0.000000)
89. Brand: _PrimaForce (0.000000)
90. Brand: _PEScience (-0.002198)
91. Brand: _Isopure (-0.002198)
92. Brand: _BSN (-0.002198)
93. Category: _D-Aspartic Acid (-0.006593)
94. Category: _Whey Protein Blends (-0.013187)
95. Brand: _Gaspari Nutrition (-0.017582)
96. Brand: _MET-Rx (-0.028571)
```

[32]:
```
#Calibration

Cal_NB_machine = CalibratedClassifierCV(NB_machine, cv = None, method =␣
 ↪"isotonic")

Cal_NB_machine = Cal_NB_machine.fit(X_train, Y_train)

Y_pred = Cal_NB_machine.predict(X_test)
```

[33]:
```
#Evaluating the NB
print("Confusion Matrix for Calibrated NB : \n", metrics.
 ↪confusion_matrix(Y_test, Y_pred),"\n")
print("The precision for NB is ", metrics.precision_score(Y_test, Y_pred))
print("The recall for NB is ", metrics.recall_score(Y_test, Y_pred),"\n")
print("The accuracy for NB is ", metrics.accuracy_score(Y_test, Y_pred),"\n")
print("The error rate for NB is ", (1 - metrics.accuracy_score(Y_test,␣
 ↪Y_pred)),"\n")
print("The F-score for NB is ", metrics.f1_score(Y_test, Y_pred),"\n")
```

```
Confusion Matrix for Calibrated NB :
 [[19 13]
 [25 34]]

The precision for NB is  0.723404255319149
The recall for NB is  0.576271186440678

The accuracy for NB is  0.5824175824175825

The error rate for NB is  0.41758241758241754

The F-score for NB is  0.6415094339622641
```

3 SVM Text

[34]:
```
#Preparing data
svm_prod = raw_prod.copy()
svm_prod.drop(["brand_name", "number_of_flavors", "price",␣
 ↪"product_category","overall_rating",], axis = 1, inplace = True)
```

[35]:
```
svm_prod
```

[35]:
```
                                   product_description  label
0    BCAA Powder with Natural Energizers Sourced fr…      1
1    24g of Whey Protein with Amino Acids for Muscl…      1
2    Pre-Workout Powder Powerhouse Packed with 13-H…      1
4    24g of Pure, Quality Protein in Every Scoop wi…      1
7             Advanced Pre-Workout + Weight Management      1
..                                                 …    …
819  Slow And Sustained Release To Keep Muscles Fed…      0
824  Pre-Mix Pre-Workout for Energy, Focus and Ulti…      1
825  Made with Zero Artificial Ingredients and Nati…      1
826                                           Natural!      0
830                               Plant-based Protein!      1

[303 rows x 2 columns]
```

[36]:
```
#NLP and SVM packages
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from string import punctuation
import re
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import model_selection, svm
```

[37]:
```
#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"]]
```

[38]:
```
#Removing stop words
stop = stopwords.words("english")


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))
```

[39]:
```
#Removing Scpecial 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))
```

[40]:
```
#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))
```

[41]:
```
#Stemming?

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))
```

[42]:
```
svm_prod
```

[42]:
```
                                   product_description  label  \
0     [bcaa, powder, with, natural, energizers, sour…      1
1     [24g, of, whey, protein, with, amino, acids, f…      1
2     [pre-workout, powder, powerhouse, packed, with…      1
4     [24g, of, pure, ,, quality, protein, in, every…      1
7        [advanced, pre-workout, +, weight, management]      1
..                                                 …    …
819   [slow, and, sustained, release, to, keep, musc…      0
824   [pre-mix, pre-workout, for, energy, ,, focus, …      1
825   [made, with, zero, artificial, ingredients, an…      1
826                                        [natural, !]      0
830                             [plant-based, protein, !]      1

                                              tkn_no_sw  \
0    [bcaa, powder, natural, energizers, sourced, g…
1    [24g, whey, protein, amino, acids, muscle, rec…
2    [pre-workout, powder, powerhouse, packed, 13-h…
4    [24g, pure, ,, quality, protein, every, scoop,…
7       [advanced, pre-workout, +, weight, management]
..                                                 …
819  [slow, sustained, release, keep, muscles, fed,…
824  [pre-mix, pre-workout, energy, ,, focus, ultim…
825  [made, zero, artificial, ingredients, native, …
826                                     [natural, !]
830                          [plant-based, protein, !]

                                         tkn_no_sw_p  \
0    [bcaa, powder, natural, energizers, sourced, g…
1    [24g, whey, protein, amino, acids, muscle, rec…
2    [powder, powerhouse, packed, picked, ingredien…
4    [24g, pure, quality, protein, every, scoop, ad…
7                    [advanced, weight, management]
..                                                 …
819  [slow, sustained, release, keep, muscles, fed,…
824            [energy, focus, ultimate, convenience]
825  [made, zero, artificial, ingredients, native, …
826                                        [natural]
830                                        [protein]

                                      tkn_no_sw_p_nb  \
0    [bcaa, powder, natural, energizers, sourced, g…
1      [whey, protein, amino, acids, muscle, recovery]
2    [powder, powerhouse, packed, picked, ingredien…
4    [pure, quality, protein, every, scoop, added, …
7                    [advanced, weight, management]
..                                                 …
819  [slow, sustained, release, keep, muscles, fed,…
824            [energy, focus, ultimate, convenience]
825  [made, zero, artificial, ingredients, native, …
826                                        [natural]
830                                        [protein]

                                            tkn_lemm
0    [bcaa, powder, natural, energizer, sourced, gr…
1       [whey, protein, amino, acid, muscle, recovery]
2    [powder, powerhouse, packed, picked, ingredien…
4    [pure, quality, protein, every, scoop, added, …
7                    [advanced, weight, management]
..                                                 …
819  [slow, sustained, release, keep, muscle, fed, …
824            [energy, focus, ultimate, convenience]
```

```
825  [made, zero, artificial, ingredient, native, w…
826                                         [natural]
830                                         [protein]

[303 rows x 6 columns]
```

[43]:
```
svm_prod["clean"] = svm_prod["tkn_lemm"].apply(lambda entry: " ".join(entry))
svm_final = svm_prod[["label", "clean"]]
```

[44]:
```
svm_final.loc[4, "clean"]
```

[44]:
```
'pure quality protein every scoop added amino acid filler nutrient'
```

[45]:
```
#Splitting the data
X_train, X_test, Y_train, Y_test = model_selection.
 ↪train_test_split(svm_final['clean'],svm_final['label'],test_size=0.3,␣
 ↪random_state = 1996)
```

[46]:
```
#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

X_train_Tfidf = Tfidf_vect.transform(X_train)
X_test_Tfidf = Tfidf_vect.transform(X_test)
```

[47]:
```
#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)
```

[48]:
```
print(str(SV_prod.coef_))
```

```
  (0, 222)      0.29390150060925657
  (0, 81)       0.27885273924492837
  (0, 80)       0.1224871425698392
  (0, 273)      0.5409880529664709
  (0, 151)      0.5045743527574137
  (0, 72)       0.45869851596911654
  (0, 183)      0.43234100460133473
  (0, 104)      0.43234100460133473
  (0, 54)       0.22118319795237343
  (0, 189)      0.43521502188024225
  (0, 169)      0.43521502188024225
  (0, 135)      0.43521502188024225
  (0, 212)      0.24963008926939578
  (0, 11)       0.5797706277264328
  (0, 2)        0.5797706277264328
  (0, 238)      0.9557344456476344
  (0, 260)      0.39321623882873713
  (0, 91)       0.612842645455558
  (0, 6)        0.39321623882873713
  (0, 28)       0.9478126505413083
  (0, 26)       0.9478126505413083
  (0, 216)      0.4096420241255165
  (0, 109)      0.4096420241255165
  (0, 84)       0.4096420241255165
  (0, 165)      0.1272219760674679
  :     :
  (0, 44)       0.43383194066677977
  (0, 246)      -0.31006586531856106
  (0, 240)      -0.7839057086182566
  (0, 173)      -0.5164734447656214
  (0, 139)      -0.0060401626250568
  (0, 87)       0.684008245363187
  (0, 68)       -0.9270619355709491
  (0, 39)       -0.5164734447656214
  (0, 270)      -1.2182955300032619
  (0, 191)      0.5914764393972491
  (0, 276)      0.7602867014364114
  (0, 128)      -0.6378959025690238
  (0, 40)       -0.17855834085172095
  (0, 271)      0.6032755970487345
  (0, 127)      -1.291921332293117
  (0, 35)       0.021424671823774555
  (0, 161)      -0.46658713489606846
  (0, 223)      -0.9301437431898613
  (0, 121)      0.4911857237569417
  (0, 95)       -0.2744768141718483
  (0, 76)       1.1865432594983318
  (0, 13)       -0.9301437431898613
  (0, 7)        -0.8825172043774686
  (0, 201)      1.1865520847253441
  (0, 51)       -0.297364908860267
```

[49]:
```
sorted_coeff = SV_prod.coef_.toarray()
coeff_df = pd.DataFrame(sorted_coeff, index = ["Coefficient"])
coeff_df = coeff_df.T
dict_code = pd.Series(range(0,277))
coeff_df["dict_code"] = dict_code

#Unwrangling the Vocabulary of the matrix
```

```python
res = dict((v,k) for k,v in Tfidf_vect.vocabulary_.items())

coeff_df["word"] = coeff_df["dict_code"].map(lambda code : res[code])

#Order of columns and sorting
#coeff_df.order(by = "")
```

[50]:
```python
coeff_df.sort_values(by = "Coefficient")
```

[50]:
```
     Coefficient  dict_code         word
127    -1.291921        127      isolate
180    -1.260142        180  performance
199    -1.221891        199      promote
270    -1.218296        270       weight
164    -1.078959        164    nighttime
..           …          …          …
257     1.164544        257     ultimate
76      1.186543         76       energy
201     1.186552        201      protein
23      1.515921         23        bcaas
242     1.536012        242      support

[277 rows x 3 columns]
```

[51]:
```python
coeff_df
```

[51]:
```
     Coefficient  dict_code     word
0      -0.575407          0  absorbs
1       0.000000          1  achieve
2       0.579771          2     acid
3      -0.682450          3   acting
4      -0.512045          4   active
..           …          …      …
272     0.000000        272    white
273     0.540988        273    whole
274     0.531346        274  workout
275    -0.549636        275   worthy
276     0.760287        276     zero

[277 rows x 3 columns]
```

[52]:
```python
print(Tfidf_vect.vocabulary_)
```

```
{'bcaa': 22, 'powder': 191, 'natural': 161, 'energizer': 75, 'sourced': 234,
'green': 107, 'coffee': 50, 'tea': 248, 'support': 242, 'focus': 93, 'whey':
271, 'protein': 201, 'amino': 11, 'acid': 2, 'muscle': 158, 'recovery': 212,
'powerhouse': 194, 'packed': 175, 'picked': 185, 'ingredient': 122, 'improved':
```

```
118, 'pure': 205, 'quality': 207, 'every': 81, 'scoop': 222, 'added': 5,
'filler': 91, 'nutrient': 167, 'advanced': 7, 'weight': 270, 'management': 143,
'intense': 125, 'increased': 121, 'energy': 76, 'power': 192, 'bcaas': 23,
'zero': 276, 'sugar': 238, 'calorie': 38, 'essential': 80, 'formulated': 95,
'caffeine': 37, 'source': 233, 'building': 35, 'lean': 133, 'crispy': 62, 'bar':
20, 'provides': 203, 'gram': 105, 'per': 178, 'turn': 256, 'workout': 274,
'intensity': 126, 'increase': 120, 'performance': 180, 'revolutionary': 219,
'formula': 94, 'bigger': 28, 'better': 26, 'science': 221, 'based': 21,
'testosterone': 250, 'fuel': 97, 'pharmaceutical': 183, 'grade': 104,
'micronized': 150, 'creatine': 61, 'anyone': 13, 'seeking': 223, 'complex': 52,
'milkshake': 152, 'taste': 246, 'glutamine': 101, 'result': 218, 'hydrolyzed':
115, 'build': 34, 'serious': 225, 'complete': 51, 'multistage': 157,
'thermogenic': 251, 'fat': 87, 'loss': 139, 'clean': 48, 'gluten': 102, 'free':
96, 'high': 112, 'nighttime': 164, 'use': 262, 'explosive': 83, 'cutting': 64,
'isolate': 127, 'optimal': 169, 'level': 135, 'post': 189, 'patented': 177,
'hydrochloride': 114, 'uncompromised': 259, 'purity': 206, 'carb': 40,
'isopure': 128, 'original': 172, 'igniter': 116, 'deliciously': 67, 'crunchy':
63, 'supplement': 241, 'extreme': 84, 'requirement': 216, 'hardcore': 109,
'unflavored': 260, 'additive': 6, 'strength': 237, 'ultra': 258, 'premium': 195,
'mass': 144, 'worthy': 275, 'gold': 103, 'standard': 236, 'name': 159, 'great':
106, 'tasting': 247, 'minimal': 153, 'carbs': 42, 'digestive': 70, 'potent':
190, 'powerful': 193, 'ultimate': 257, 'plus': 188, 'vegan': 264, 'cookie': 59,
'delicious': 66, 'way': 269, 'insane': 123, 'maximize': 147, 'perfect': 179,
'comprehensive': 53, 'period': 181, 'designed': 68, 'enhance': 78, 'micellar':
149, 'casein': 44, 'low': 140, 'diet': 69, 'contains': 56, 'help': 111,
'massive': 145, 'ph': 182, 'correct': 60, 'professional': 197, 'blend': 30,
'sustaining': 245, 'pump': 204, 'super': 239, 'fast': 85, 'acting': 3, 'burner':
36, 'meal': 148, 'replacement': 215, 'egg': 72, 'endurance': 74, 'boosting': 33,
'gainer': 100, 'mrp': 156, 'matrix': 146, 'mix': 154, 'clinically': 49,
'proven': 202, 'carbohydrate': 41, 'serving': 226, 'growth': 108, 'training':
255, 'shake': 227, 'leaner': 134, 'macronutrient': 141, 'profile': 198,
'athlete': 16, 'looking': 138, 'achieve': 1, 'shredded': 228, 'physique': 184,
'convenient': 58, 'healthy': 110, 'joint': 129, 'satisfying': 220, 'promote':
199, 'repair': 214, 'monohydrate': 155, 'lab': 131, 'tested': 249, 'raw': 209,
'plant': 186, 'aiding': 8, 'exercise': 82, 'supporting': 243, 'bioengineered':
29, 'beef': 24, 'capsule': 39, 'originally': 173, 'drink': 71, 'vegetable': 265,
'engineered': 77, 'feed': 89, 'plasma': 187, 'volumizer': 267, 'ratio': 208,
'promotes': 200, 'booster': 32, 'optimum': 170, 'reach': 210, 'appearance': 14,
'beta': 25, 'alanine': 9, 'carnosine': 43, 'superb': 240, 'nitric': 166,
'oxide': 174, 'instantly': 124, 'dairy': 65, 'lactose': 132, 'ready': 211,
'size': 230, 'torque': 253, 'includes': 119, 'total': 254, 'active': 4,
'fitness': 92, 'partner': 176, 'nutrition': 168, 'absorbs': 0, 'faster': 86,
'load': 136, 'cholesterol': 46, 'elite': 73, 'series': 224, 'unparalleled': 261,
'balanced': 19, 'vasodilator': 263, 'thirteen': 252, 'sprouted': 235,
'concentrate': 54, 'reservoir': 217, 'made': 142, 'artificial': 15, 'native':
160, 'cell': 45, 'volumizing': 268, 'preworkout': 196, 'awareness': 18, 'milk':
151, 'fusion': 99, 'nitrate': 165, 'sleep': 231, 'citrulline': 47, 'improve':
117, 'amazing': 10, 'white': 272, 'six': 229, 'convenience': 57, 'beyond': 27,
'organic': 171, 'whole': 273, 'highest': 113, 'loaded': 137, 'vitamin': 266,
'antioxidant': 12, 'athletic': 17, 'enhancer': 79, 'concentrated': 55, 'full':
98, 'fiber': 90, 'net': 162, 'boost': 31, 'slow': 232, 'sustained': 244,
'release': 213, 'keep': 130, 'fed': 88, 'night': 163}
```

[53]:
```
res = dict((v,k) for k,v in Tfidf_vect.vocabulary_.items())
```

[54]:
```
#Evaluating the NB
print("Confusion Matrix for SVM : \n", metrics.confusion_matrix(Y_test,␣
 ↪Y_pred),"\n")
print("The precision for SVM is ", metrics.precision_score(Y_test, Y_pred))
print("The recall for SVM is ", metrics.recall_score(Y_test, Y_pred),"\n")
print("The accuracy for SVM is ", metrics.accuracy_score(Y_test, Y_pred),"\n")
print("The error rate for SVM is ", (1 - metrics.accuracy_score(Y_test,␣
 ↪Y_pred)),"\n")
print("The F-score for SVM is ", metrics.f1_score(Y_test, Y_pred),"\n")
```

```
Confusion Matrix for SVM :
 [[16 16]
 [13 46]]

The precision for SVM is  0.7419354838709677
The recall for SVM is  0.7796610169491526

The accuracy for SVM is  0.6813186813186813

The error rate for SVM is  0.31868131868131866

The F-score for SVM is  0.7603305785123968
```

[55]:
```
raw_prod["label"].value_counts()
```

[55]:
```
1    184
0    119
Name: label, dtype: int64
```

[56]:
```
raw_prod.groupby("label")["price"].mean()
```

[56]:
```
label
0    35.088487
1    33.855870
Name: price, dtype: float64
```

[57]:
```
#Opportunity Cost
raw_prod["Potential Sales"] = raw_prod["label"].map(lambda x: 1400 if x == 1␣
 ↪else 1000)
```

```
raw_prod["Total Revenue"] = raw_prod["Potential Sales"] * raw_prod["price"]
raw_prod.groupby("label")["Total Revenue"].mean()
```

```
[57]: label
      0    35088.487395
      1    47398.217391
      Name: Total Revenue, dtype: float64
```

```
[58]: raw_prod.describe()
```

```
[58]:        number_of_flavors  overall_rating       price       label  \
      count         303.000000      303.000000  303.000000  303.000000
      mean            7.033003        8.976568   34.339967    0.607261
      std             7.718918        0.550928   19.152406    0.489168
      min             1.000000        5.700000    3.050000    0.000000
      25%             2.000000        8.700000   20.480000    0.000000
      50%             5.000000        9.100000   31.450000    1.000000
      75%             9.000000        9.300000   43.895000    1.000000
      max            43.000000       10.000000  119.530000    1.000000

             Potential Sales  Total Revenue
      count       303.000000     303.000000
      mean       1242.904290   42563.702970
      std         195.667005   25307.356457
      min        1000.000000    4270.000000
      25%        1000.000000   26960.000000
      50%        1400.000000   39186.000000
      75%        1400.000000   53186.000000
      max        1400.000000  167342.000000
```

```
[59]: raw_prod["brand_name"].value_counts()
```

```
[59]: Optimum Nutrition          33
      EVLUTION NUTRITION         27
      Universal Nutrition        18
      AllMax Nutrition           14
      Isopure                    14
      Cellucor                   13
      BSN                        12
      Dymatize                   10
      RSP Nutrition               9
      GAT                         9
      Animal                      8
      MuscleTech                  8
      PEScience                   7
      MET-Rx                      6
      NOW                         6
```

```
MRM                            6
ABB                            6
NutraBio                       5
MuscleMeds                     5
Kaged Muscle                   5
Quest Nutrition                5
Beverly International          4
Vega                           4
Muscle Milk                    4
JYM Supplement Science         4
Six Star Pro Nutrition         4
Muscle Beach Nutrition         3
Ascent                         3
Labrada                        3
Gaspari Nutrition              3
EFX Sports                     3
COBRA LABS                     3
NLA for Her                    3
Grenade                        3
Body Nutrition                 3
Beast Sports Nutrition         2
CytoSport                      2
Celsius                        2
eFlow Nutrition                2
S.A.N.                         2
Core Nutritionals              2
Garden Of Life                 2
PrimaForce                     1
Bodybuilding.com Signature     1
Top Secret Nutrition           1
Gamma Labs                     1
iSatori                        1
Pro Supps                      1
Magnum Nutraceuticals          1
MHP                            1
FINAFLEX                       1
OhYeah! Nutrition              1
Betancourt Nutrition           1
Lenny & Larry's                1
iForce Nutrition               1
Sports Research                1
ONE                            1
AST                            1
Name: brand_name, dtype: int64
```

[60]:
```
raw_prod["product_category"].value_counts()
```

[60]:
```
Whey Protein                 68
Creatine Monohydrate         31
Whey Protein Isolate         29
Improve Workout Products     26
Beta-Alanine                 19
Build Muscle Products        15
Plant Protein                13
Protein                       9
Micellar Casein Protein       8
Caffeine                      8
Whey Protein Blends           7
Citrulline                    6
BCAAs                         6
Amino Acids                   6
Glutamine                     5
Whey Protein Concentrate      5
Beef Protein                  5
Weight Loss Products          4
Hydrolyzed Whey Protein       4
D-Aspartic Acid               3
Kre-Alkalyn                   3
Green Tea                     3
L-Arginine                    3
Green Coffee Extract          2
Creatine HCl                  2
Agmatine                      2
Egg Protein                   2
Waxy Maize                    1
GABA                          1
L-Taurine                     1
Creatine                      1
Yerba Mate                    1
Betaine Anhydrous             1
Collagen                      1
Carbohydrates                 1
Creatine Malate               1
Name: product_category, dtype: int64
```

[ ]:

[ ]: