

# Ingredient Analyser for Consumables



Presented By:  
Swathi

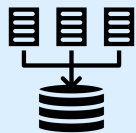


# Problem Statement

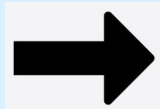
Additives in consumable products listed in ingredients are hard to identify for the average consumer. Too many food safety standards and clever labeling makes it tedious for customers to make the right choice.

A simple, easy to understand scale of safety levels can be devices to classify products based on the nature of additives present. A machine learning model can be trained using this labeled data to predict the classification of new products, thereby helping shoppers choose safer consumable products.

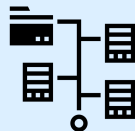
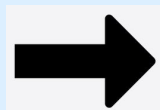
# Project stages



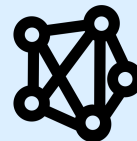
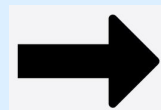
Data Collection



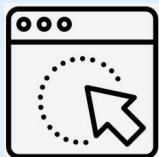
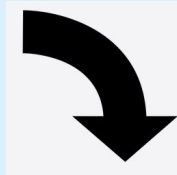
Data preparation



Labeling Records  
and identifying  
features.



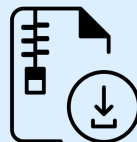
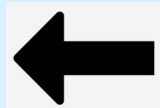
Training classification  
models



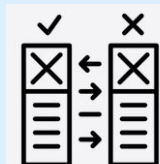
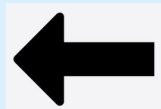
Deploying the tool



Design a web  
service




Serialization



Choosing best  
model





# Data Collection & Preparation

A dataset of 10000 popular consumable products with their ingredients was collected. Missing values were handled by finding the ingredients wherever applicable and dropping some records.



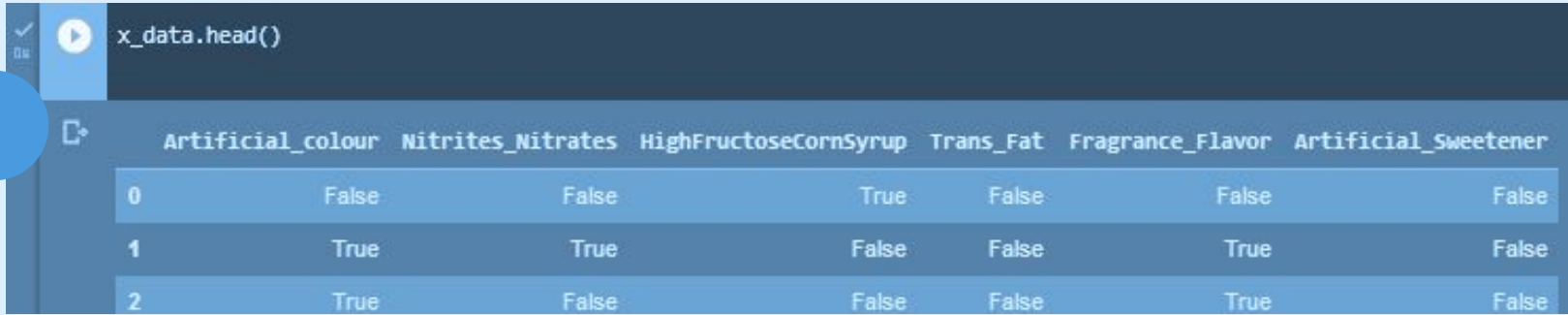
# Labeling & Feature Identification

A new column “Class” was added and each record was labeled between 1-5 based on the severity of food additives and their position in the full ingredient list.

The most prominent additives observed during labeling were grouped into 6 categories. Based on this the presence/absence of any additive in each category of additive was stored in 6 feature columns.

For example: The Artificial\_colour column will be set to true if any of the dyes like Blue 1 (Brilliant Blue), Blue 2 (Indigo Carmine), Citrus Red 2, Green 3 (Fast Green FCF), or any other color is present in the ingredient list. If no such ingredient is found then the column value will be set to False.

# Categories/Features



```
x_data.head()
```

	Artificial_colour	Nitrites_Nitrates	HighFructoseCornSyrup	Trans_Fat	Fragrance_Flavor	Artificial_Sweetener
0	False	False	True	False	False	False
1	True	True	False	False	True	False
2	True	False	False	False	True	False

01 — 02 — 03 — 04 — 05 — 06

## Artificial Colour

Example ingredients:  
Blue, Red, Yellow 5

## Nitrites\_Nitrates

This will be true for any nitrite/nitrate.

## HighFructoseCornSyrup

This feature will be set if corn syrup is present in the list.

## Trans\_Fat

Examples:  
Hydrogenated oil,  
hydrogenated soy

## Fragrance\_Flavour

Any artificial flavouring agent or fragrance wil.

## Artificial sweetener

Only harmful Artificial sweeteners like Aspartame.



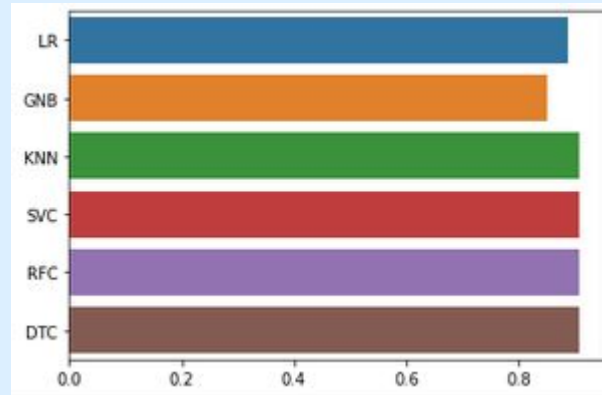
# Training Classification Models

The data set was split into test and train sets and the training sets were used to train the classification models.

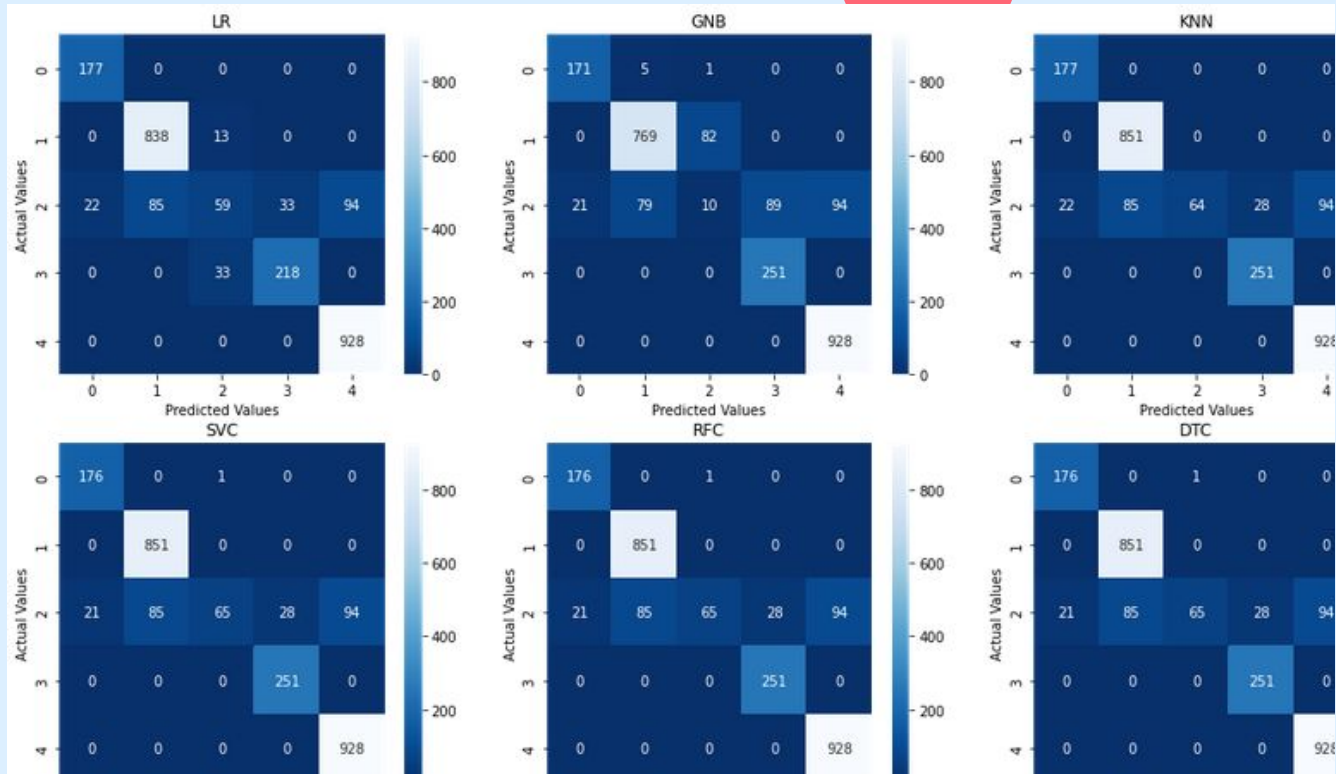
Logistic Regression, Naive-Bayes Classifier, K-Nearest Neighbors, SVM, Random Forest Classifier and Decision Tree Classifier models were trained.

# Choosing the right model

K-Nearest Neighbors, SVM, Random Forest Classifier and Decision Tree Classifier models were trained. K-Nearest Neighbors, SVM, Random Forest Classifier and Decision Tree Classifier models had a significantly higher accuracy rate compared to Logistic Regression and Naive Bayes Classifier models.







K-Nearest Neighbors, SVM, Random Forest Classifier and Decision Tree Classifier models perform better.



# Serialization/Pickling

Random Forest Classifier was chosen and saved into a .pkl file to use it on another platform. Python pickle module is used for serialising and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. Pickle “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.



# Designing a Web Service & Deployment

A simple API was designed using Flask to take ingredients as inputs and use the trained model to output the classification results. HTML/CSS was used to design the web page. The web page is hosted on python anywhere. To use the web tool:

1. Copy the ingredient list of the chosen product from any shopping service/ingredient database and click on the “Classifier” link.  
OR  
Click on the “Sample ingredients” link and copy any ingredient list from the list provided and click on the “Go to Classifier” link below the table.
2. Paste the ingredient list in the form box and click classify to see how safe the product is ranked on a scale of 1-5 where 5 is the safest.



# What the tool **CAN** do

1. Help people with disorders like thyroid and heart conditions to avoid risky additives by choosing better products.
2. Provides a quick web service which is easy to use to compare two similar products.

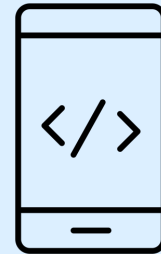
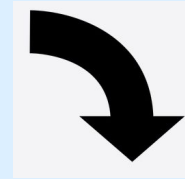
# What the tool **CAN'T** do

1. It classifies low risk ingredients like sugar in class 5 along with sugar free options, so it would not be useful for identifying high amounts of risk free/low risk ingredients.
2. There is no option to extract ingredients from the product directly.

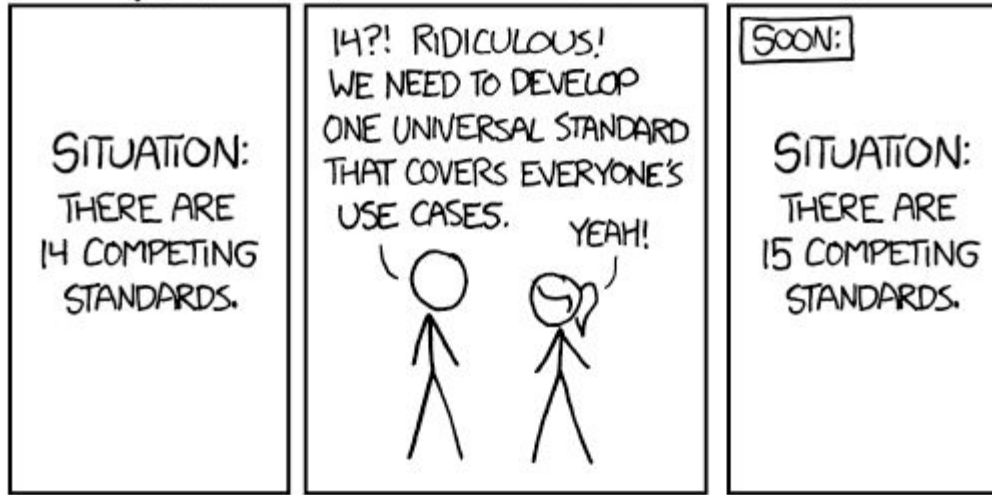
# Future Scope

A mobile application can be built using the existing API with the additional features of image to text recognition so that customers can scan ingredients using cameras.

More samples should be added along with more features to catch more additives.



## HOW STANDARDS PROLIFERATE: (SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)



**Final Note:** *The model and the tool do not aim to create a new food standard, instead aims to help compare available options.*