THINGS TO DO FOR SENTIMENT ANALYSIS OF TWEETS:

**Learn about Tweet Tokenizer:**

Tobacco main keyword with focus on product in the tweet data.

*In manual twitter analysis*

*Indicators for JUUL data*:

\* Personal experience

\* Humor

\* News media

\* Marketing

\* Health

\* Industry

\* Positive sentiment

\* Negative sentiment

\* Neutral sentiment

To keep in excel sheet

\* Language

\* Tweet id

\* Text

\* Tweet type (filter retweets)

\*Hashtags

\*Mentions

\*Media type

\*Sentiments

**Check for duplication**

**LEARN ABOUT Machine learning algos:**

* Logistic regression
* Naïve Bayes
* Random forest Classifier

**Read theorems:**

- Accuracy comparison

- Basic commands and syntax of theorems

- Introduction of all three

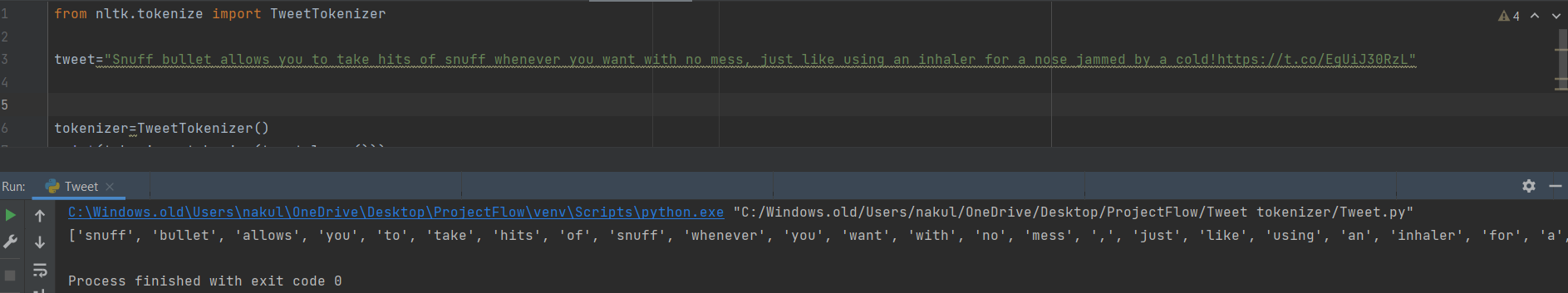
- Comparison of all three

**OBJECTIVE**

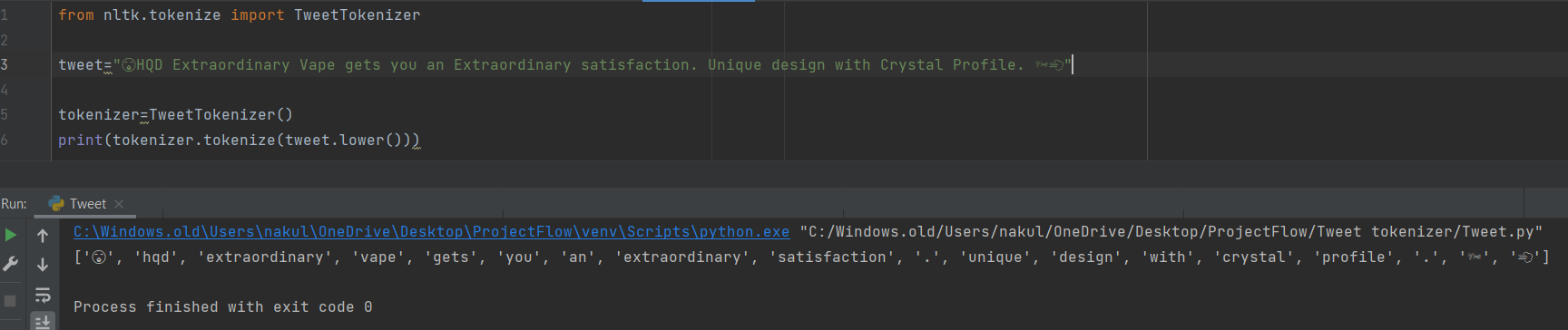
My primary objective was to further understand salient themes and topics related to JUUL use on Twitter with particular foci on underage JUUL use and health perceptions. My secondary objective was to use natural language processing (NLP) methods to develop machine learning–based classifiers capable of automatically identifying and evaluating underage-related JUUL mentions as well as positive and negative sentiments towards JUUL. In doing so, we hoped to provide optimally performing classifiers to be further validated and applied to additional work relating to underage JUUL use and its representation on Twitter.

# **TWEET TOKENIZER**

* **NLTK** has this special method called **TweetTokenizer()** that helps to tokenize Tweet Corpus into relevant tokens.
* **Tokenization** refers to a process by which a piece of sensitive data, such as a credit card number, is replaced by a surrogate value known as a token. The sensitive data still generally needs to be stored securely at one centralized location for subsequent reference and requires strong protections around it. The security of a tokenization approach depends on the security of the sensitive values and the algorithm and process used to create the surrogate value and map it back to the original value.
* The advantage of using **TweetTokenizer()** compared to regular **word\_tokenize** is that, when processing tweets, we often come across emojis, hashtags that need to be handled differently.
* I have used this on one of the tweets on my Python compiler which is IntelliJ Idea:

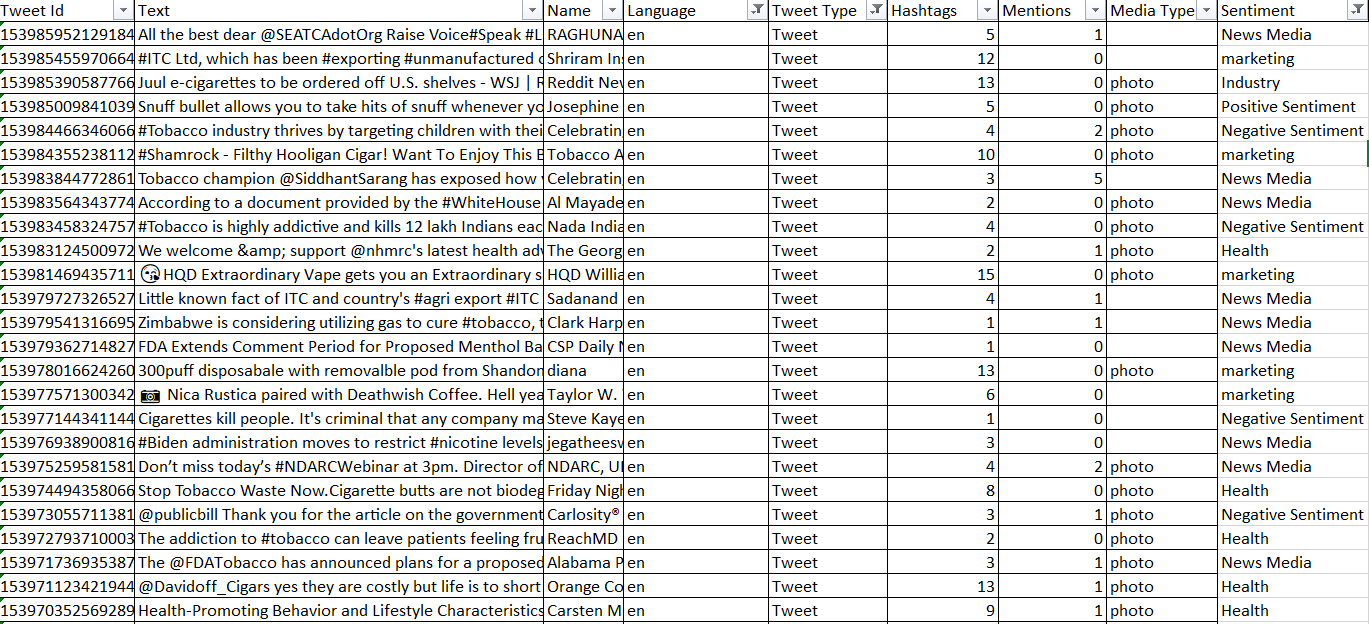


* This helped me find out what were the sentiments in the tweet.
* TweetTokenizer considers hashtag along with the words as one token while word\_tokenize splits it into 2 different tokens. Similarly with username and emoji in the corpus. Here is an example of tweetTokenizer taking the emojis in consideration in my tweets.



With the help of Tweet Tokenizer and my Python Compiler IntelliJ Idea, I was able to do sentiment analysis of over 2000+ Twitter data manually. After tokenizing every character, I was able to point out the key characters which helped me find out the real sentiments of the author of the tweet.

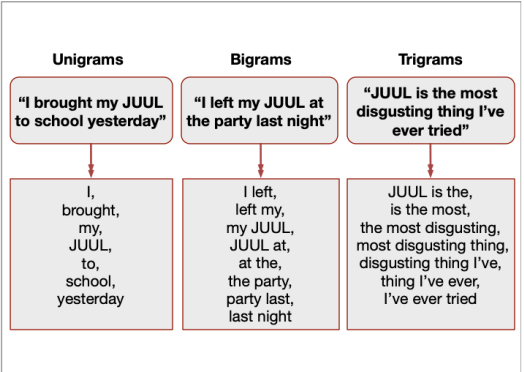
Here is some of the entries of the excel sheet, which contains all the tweets and the required information with it, that is mentioned at the starting.



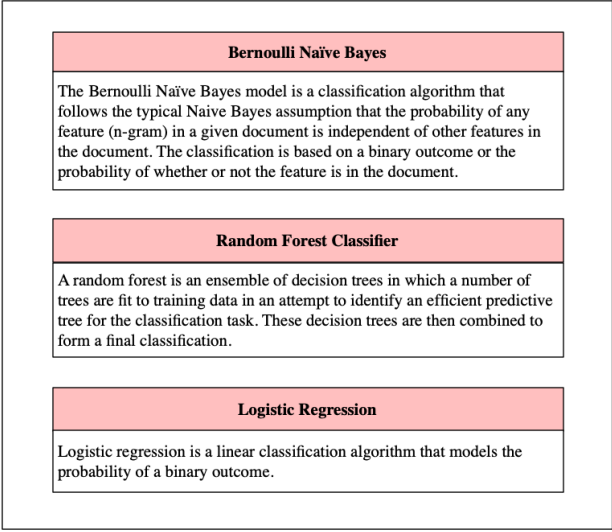
# 

# **MACHINE LEARNING CLASSIFICATION**

* All tokens were then converted into n-gram text sequences. An n-gram (ie, unigram, bigram, trigram) is a contiguous sequence of n features used in NLP to transform raw text into features that can be readily processed by a machine learning algorithm.



* In an attempt to automatically classify JUUL related tweets, we applied supervised machine learning algorithms to identify tweets related to underage JUUL use, positive sentiment, and negative sentiment. The goal of this machine learning–based approach was to identify a predictive function of the data in which unseen data can be accurately classified as containing either underage JUUL use, positive sentiment, or negative sentiment.
* The algorithms we used for classification were a logistic regression, Bernoulli naïve Bayes, and random forest classifier. These models were selected because of their computational simplicity and efficiency in Twitter-based classification tasks. The input of each classifier consisted of the most salient features determined by feature selection.
* 4 metrics were used to evaluate the performance of the various models: accuracy, precision (positive predictive value), recall (sensitivity), and F1 score (the harmonic mean of precision and recall). These metrics are standard in NLP and reflect a classifier’s ability to classify the task at hand effectively.



# **1.LOGISTIC REGRESSION**

# What is it?

* Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
* Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
* Logistic regression uses the concept of predictive modeling as regression; therefore, it is called logistic regression, but is used to classify samples; Therefore, it falls under the classification algorithm.

# Types of Logistic Regression:

* **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
* **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
* **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

# Steps in Logistic Regression in Python:

* Data Pre-processing step
* Fitting Logistic Regression to the Training set
* Predicting the test result
* Test accuracy of the result (Creation of Confusion matrix)
* Visualizing the test set result.

# **2.NAIVE BAYES CLASSIFIER**

* Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
* It is mainly used in *text classification* that includes a high-dimensional training dataset.
* Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
* **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object**.
* Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles**.

# Bayes' Theorem:

* Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
* The formula for Bayes' theorem is given as:

Naïve Bayes Classifier Algorithm

* **Where,**
  + **P(A|B) is Posterior probability**: Probability of hypothesis A on the observed event B.
  + **P(B|A) is Likelihood probability**: Probability of the evidence given that the probability of a hypothesis is true.
  + **P(A) is Prior Probability**: Probability of hypothesis before observing the evidence.
  + **P(B) is Marginal Probability**: Probability of Evidence.

# Types of Naïve Bayes Model:

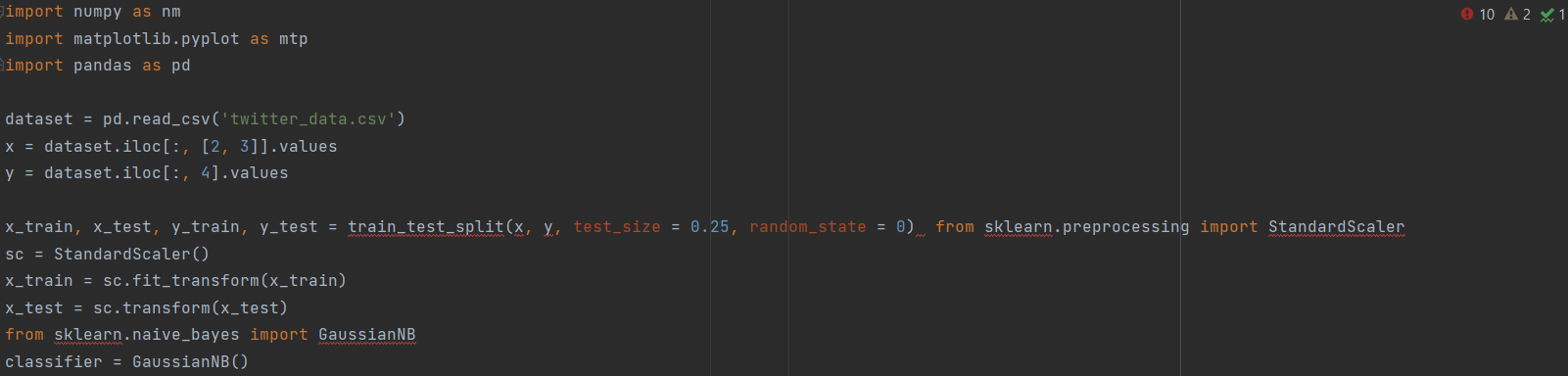
There are three types of Naive Bayes Model, which are given below:

* **Gaussian**: The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.
* **Multinomial**: The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.  
  The classifier uses the frequency of words for the predictors.
* **Bernoulli**: The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

# Python Implementation of the Naïve Bayes algorithm:

## Steps to implement:

* Data Pre-processing step
* Fitting Naive Bayes to the Training set
* Predicting the test result
* Test accuracy of the result (Creation of Confusion matrix)
* Visualizing the test set result.



# **3.RANDOM FOREST CLASSIFIER**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,** which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

# Working:



# Why use Random Forest?

Below are some points that explain why we should use the Random Forest algorithm:

* It takes less training time as compared to other algorithms.
* It predicts output with high accuracy, even for the large dataset it runs efficiently.
* It can also maintain accuracy when a large proportion of data is missing.

# Applications of Random Forest

There are mainly four sectors where Random forest mostly used:

1. **Banking:** Banking sector mostly uses this algorithm for the identification of loan risk.
2. **Medicine:** With the help of this algorithm, disease trends and risks of the disease can be identified.
3. **Land Use:** We can identify the areas of similar land use by this algorithm.
4. **Marketing:** Marketing trends can be identified using this algorithm.

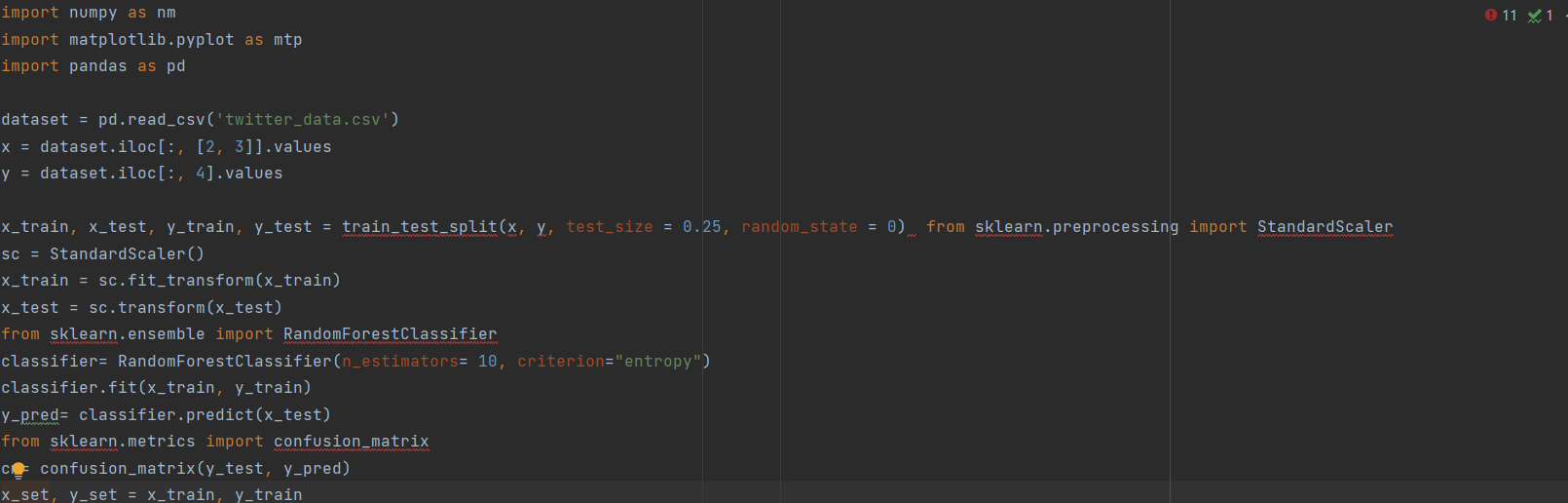
# Advantages of Random Forest

* Random Forest is capable of performing both Classification and Regression tasks.
* It is capable of handling large datasets with high dimensionality.
* It enhances the accuracy of the model and prevents the overfitting issue.

# Python Implementation of Random Forest Algorithm

# Implementation Steps are given below:

* Data Pre-processing step
* Fitting the Random forest algorithm to the Training set
* Predicting the test result
* Test accuracy of the result (Creation of Confusion matrix)
* Visualizing the test set result.



# COMPARISON OF ALL THREE

In all 3 classification tasks, the random forest model outperformed the logistic regression and Bernoulli naïve Bayes models. When classifying tweets related to underage usage of JUUL, the random forest model yielded a higher accuracy (99% accuracy) when compared to the logistic regression model (94% accuracy) and substantially higher accuracy than the Bernoulli naïve Bayes model (78% accuracy; Figure 4).

When comparing the models’ performance for classifying positive and negative tweet sentiment, the random forest model performed considerably better (82% and 91% accuracy, respectively) than the logistic regression model (72% and 78% accuracy, respectively) and the Bernoulli naïve Bayes model (69% and 62% accuracy, respectively).

When applying our random forest classifier to additional unseen data (7356 unannotated tweets), our model classified 109 of 7356 tweets as underage-related (1.48%). This proportion is lower than that of the tweets classified as underage-related during the manual annotation process (190/3152, 6.03%), perhaps due to the presence of previously unseen terms related to underage JUUL use

