1. INTRODUCTION

1.1. MACHINE LEARNING

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that which makes it more similar to humans: the ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

1.2. APPLICATIONS

- **Web Search Engine:** One of the reasons why search engines like google, bing etc work so well is because the system has learnt how to rank pages through a complex learning algorithm.
- **Photo tagging Applications:** Be it facebook or any other photo tagging application, the ability to tag friends makes it even more happening. It is all possible because of a face recognition algorithm that runs behind the application.
- **Spam Detector:** Our mail agent like Gmail or Hotmail does a lot of hard work for us in classifying the mails and moving the spam mails to spam folder.

1.3. HOW ML WORKS?

- Gathering past data in any form suitable for processing.The better the quality of data, the more suitable it will be for modeling
- Data Processing – Sometimes, the data collected is in the raw form and it needs to be pre-processed.
- Divide the input data into training,cross-validation and test sets. The ratio between the respective sets must be 6:2:2
- Building models with suitable algorithms and techniques on the training set.
- Testing our conceptualized model with data which was not fed to the model at the time of training and evaluating its performance using metrics such as F1 score, precision and recall

1.4.ML | TYPES OF LEARNING

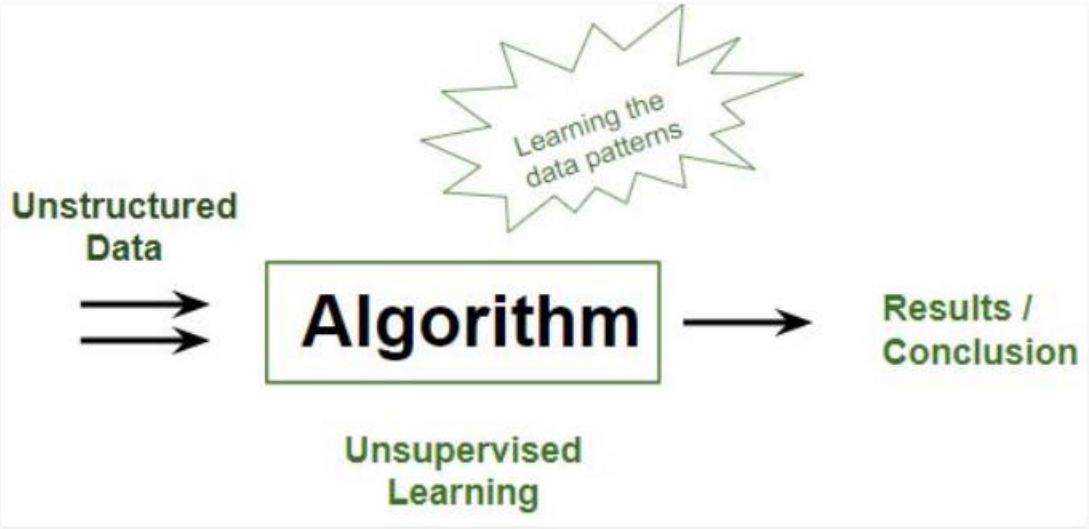


Fig 1.1 types of machine learning

1.4.1Unsupervised Learning:

It's a type of learning where we don't give target to our model while training i.e. training model has only input parameter values. The model by itself has to find which way it can learn.

Types of Unsupervised Learning :-

Clustering: Broadly this technique is applied to group data based on different patterns, our machine model finds.

Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Some algorithms:

- K-Means Clustering
- DBSCAN – Density-Based Spatial Clustering of Applications with Noise
- BIRCH – Balanced Iterative Reducing and Clustering using Hierarchies
- Hierarchical Clustering

For instance, suppose it is given an image having both dogs and cats which have not seen ever.

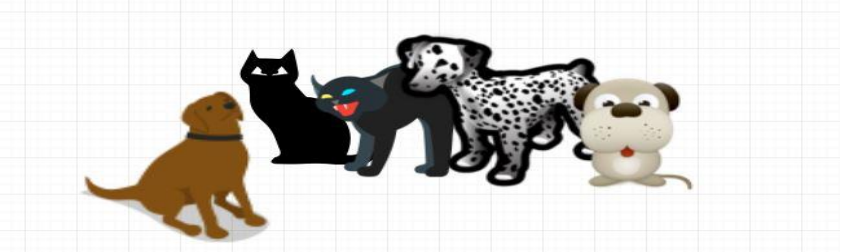


Fig 1.2 explanation of unsupervised learning

Thus the machine has no idea about the features of dogs and cat so we can't categorize it in dogs and cats. But it can categorize them according to their similarities, patterns, and differences i.e., we can easily categorize the above picture into two parts

1.4.2Supervised learning:

The majority of practical machine learning uses supervised learning. Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output **Y=f(X)** .

Types of Supervised Learning:

Classification : It is a Supervised Learning task where output is having defined labels(discrete value).
It can be either binary or multi class classification. In binary classification, model predicts either 0 or 1 yes or no but in case of multi class classification, model predicts more than one class.

Regression : It is a Supervised Learning task where output is having continuous value.

Supervised Learning Algorithms:

- Linear Regression
- Nearest Neighbor
- Guassian Naive Bayes
- Decision Trees
- Support Vector Machine (SVM)
- Random Forest

For instance, suppose you are given an basket filled with different kinds of fruits. Now the first step is to train the machine with all different fruits one by one like this:

- If shape of object is rounded and depression at top having color Red then it will be labelled as –**Apple**.
- If shape of object is long curving cylinder having color Green-Yellow then it will be labelled as –**Banana**.

2.Best Python libraries for Machine Learning

- Numpy
- Scipy
- Scikit-learn
- Pandas
- Matplotlib

Numpy: NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning.

Scikit learn: Skikit-learn is one of the most popular ML libraries for classical ML algorithms. It is built on top of two basic Python libraries, viz., NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms. Scikit-learn can also be used for data-mining and data-analysis, which makes it a great tool who is starting out with ML.

Pandas: Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training. In this case, Pandas comes handy as it was developed specifically for data extraction and preparation. It provides high-level data structures and wide variety tools for data analysis.

Matplotlib: Matpoltlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc.

Keras:keras is a very popular Machine Learning library for Python. It is a high-level neural networks API capable of running on top of TensorFlow, CNTK, or Theano. It can run seamlessly on both CPU and GPU. Keras makes it really for ML beginners to build and design a Neural Network.

2.1.Introduction to Data in Machine Learning

DATA : It can be any unprocessed fact, value, text, sound or picture that is not being interpreted and analyzed. Data is the most important part of all Data Analytics, Machine Learning, Artificial Intelligence. Without data, we can't train any model and all modern research and automation will go vain.

INFORMATION: Data that has been interpreted and manipulated and has now some

Meaningful inference for the users.

KNOWLEDGE : Combination of inferred information, experiences, learning and insights. Results in awareness or concept building for an individual or organization.

2.2.How we split data in Machine Learning?

- **Training Data:** The part of data we use to train our model. This is the data which your model actually sees(both input and output) and learn from.
- **Validation Data:** The part of data which is used to do a frequent evaluation of model, fit on training dataset along with improving involved hyperparameters (initially set parameters before the model begins learning). This data plays it's part when the model is actually training.
- **Testing Data:** Once our model is completely trained, testing data provides the unbiased evaluation. When we feed in the inputs of Testing data, our model will predict some values(without seeing actual output). After prediction, we evaluate our model by comparing it with actual output present in the testing data.

2.3.Properties of Data –

1. **Volume :** Scale of Data. With growing world population and technology at exposure, huge data is being generated each and every millisecond.
2. **Variety :** Different forms of data – healthcare, images, videos, audio clippings.
3. **Velocity :** Rate of data streaming and generation.
4. **Value :** Meaningfulness of data in terms of information which researchers can infer from it.
5. **Veracity :** Certainty and correctness in data we are working on.

2.4.Confusion Matrix in Machine Learning

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.It allows easy identification of confusion between classes e.g. one class is commonly mislabeled as the other. Most performance measures are computed from the confusion matrix.

ConfusionMatrix:

A confusion matrix is a summary of prediction results on a classification problem.

The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix.

The confusion matrix shows the ways in which your classification model is confused when it makes predictions.It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

	Class 1 Predicted	Class 2 Predicted
Class 1 Actual	TP	FN
Class 2 Actual	FP	TN

Fig 2.1 Confusion matrix

Positive (P) : Observation is positive (for example: is an apple).
• Negative (N) : Observation is not positive (for example: is not an apple).
• True Positive (TP) : Observation is positive, and is predicted to be positive.
• False Negative (FN) : Observation is positive, but is predicted negative.
• True Negative (TN) : Observation is negative, and is predicted to be negative.
• False Positive (FP) : Observation is negative, but is predicted positive.

Classification Rate/Accuracy:

Classification Rate or Accuracy is given by the relation:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Recall:

Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples. High Recall indicates the class is correctly recognized (small number of FN).

Recall is given by the relation:

$$\textit{Recall} = \frac{TP}{TP + FN}$$

Precision:

To get the value of precision we divide the total number of correctly classified positive examples by the total number of predicted positive examples. High Precision indicates an example labeled as positive is indeed positive (small number of FP).

Precision is given by the relation:

$$Precision = \frac{TP}{TP + FP}$$

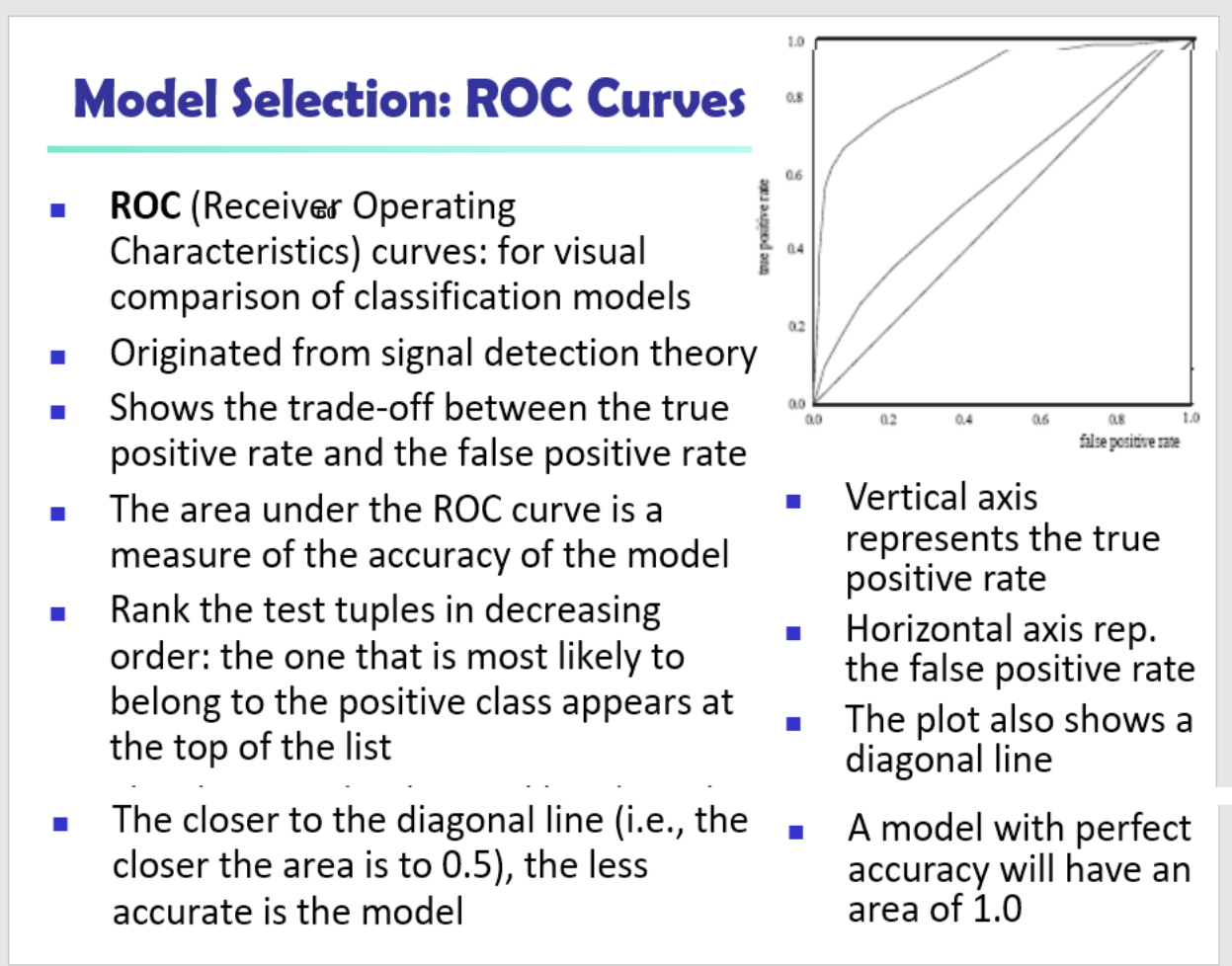


Fig 2.2 ROC curves

2.5.K-Nearest Neighbours

K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection.

It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data.

2.6. Logistic Regression

The name of this algorithm could be a little confusing in the sense that Logistic Regression machine learning algorithm is for classification tasks and not regression problems. The name 'Regression' here implies that a linear model is fit into the feature space. This algorithm applies a logistic function to a linear combination of features to predict the outcome of a categorical dependent variable based on predictor variables.

The odds or probabilities that describe the outcome of a single trial are modelled as a function of explanatory variables. Logistic regression algorithms helps estimate the probability of falling into a specific level of the categorical dependent variable based on the given predictor variables.

Based on the nature of categorical response, logistic regression is classified into 3 types –

Binary Logistic Regression – The most commonly used logistic regression when the categorical response has 2 possible outcomes i.e. either yes or not. Example –Predicting whether a student will pass or fail an exam, predicting whether a student will have low or high blood pressure, predicting whether a tumor is cancerous or not.

Multi-nominal Logistic Regression - Categorical response has 3 or more possible outcomes with no ordering. Example- Predicting what kind of search engine (Yahoo, Bing, Google, and MSN) is used by majority of US citizens.

Ordinal Logistic Regression - Categorical response has 3 or more possible outcomes with natural ordering. Example- How a customer rates the service and quality of food at a restaurant based on a scale of 1 to 10.

2.6.1 Advantages of Using Logistic Regression

- Easier to inspect and less complex.
- Robust algorithm as the independent variables need not have equal variance or normal distribution.
- These algorithms do not assume a linear relationship between the dependent and independent variables and hence can also handle non-linear effects.
- Controls confounding and tests interaction.

2.6.2.Drawbacks of Using Logistic Regression:

- When the training data is sparse and high dimensional, in such situations a logistic model may over fit the training data.
- Logistic regression algorithms cannot predict continuous outcomes. For instance, logistic regression cannot be applied when the goal is to determine how heavily it will rain because the scale of measuring rainfall is continuous. Data scientists can predict heavy or low rainfall but this would make some compromises with the precision of the dataset.

- Logistic regression algorithms require more data to achieve stability and meaningful results. These algorithms require minimum of 50 data points per predictor to achieve stable outcomes.
- It predicts outcomes depending on a group of independent variables and if a data scientist or a machine learning expert goes wrong in identifying the independent variables then the developed model will have minimal or no predictive value.
- It is not robust to outliers and missing values.

2.6.3.Applications of Logistic Regression:

- Logistic regression algorithm is applied in the field of epidemiology to identify risk factors for diseases and plan accordingly for preventive measures.
- Used to predict whether a candidate will win or lose a political election or to predict whether a voter will vote for a particular candidate.
- Used to classify a set of words as nouns, pronouns, verbs, adjectives.
- Used in weather forecasting to predict the probability of rain.
- Used in credit scoring systems for risk management to predict the defaulting of an account.

2.7.Support Vector Machine(SVM):

It is a black box machine technique.It uses a technique called hyperplain.Kernel with rbf produces more accuracy than all the models .It gives more accuracy for training data than testing data.It is used in both classification and regression.An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.In addition to performing linear classification, SVMs can efficiently perform a non-linear classification, implicitly mapping their inputs into high-dimensional feature spaces.

2.8.Random Forest (Breiman 2001):

- **Random Forest:**
  - Each classifier in the ensemble is a *decision tree* classifier and is generated using a random selection of attributes at each node to determine the split
  - During classification, each tree votes and the most popular class is returnd Two Methods to construct Random Forest:
  - Forest-RI (*random input selection*): Randomly select, at each node, F attributes as candidates for the split at the node. The CART methodology isused to grow the trees to maximum size
  - Forest-RC (*random linear combinations*)*:* Creates new attributes (or features) that are a linear combination of the existing attributes (reduces the correlation between individual classifiers)
  - Comparable in accuracy to Adaboost, but more robust to errors and outliers

- Insensitive to the number of attributes selected for consideration at each split, and faster than bagging or boosting.

2.8.1.Bagging: Boostrap Aggregation:

- Each classifier M_i returns its class prediction
- The bagged classifier M* counts the votes and assigns the class with the most votes to **X**
- Prediction: can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple.
- Often significantly better than a single classifier derived from D
- For noise data: not considerably worse, more robust
- Proved improved accuracy in prediction

2.8.2.How boosting works?

- **Weights** are assigned to each training tuple
- A series of k classifiers is iteratively learned
- After a classifier M_i is learned, the weights are updated to allow the subsequent classifier, M_{i+1}, to **pay more attention to the training tuples that were misclassified** by M_i
- The final **M* combines the votes** of each individual classifier, where the weight of each classifier's vote is a function of its accuracy.
- Boosting algorithm can be extended for numeric prediction
- Comparing with bagging: Boosting tends to have greater accuracy, but it also risks overfitting the model to misclassified data.

2.8.3.Adaboost (Freund and Schapire, 1997)

- Given a set of *d* class-labeled tuples, $(\mathbf{X_1}, y_1), \ldots, (\mathbf{X_d}, y_d)$

- Initially, all the weights of tuples are set the same (1/d)
- Generate k classifiers in k rounds. At round i,
- Tuples from D are sampled (with replacement) to form a training set D_i of the same size
- Each tuple's chance of being selected is based on its weight
- A classification model M_i is derived from D_i
- Its error rate is calculated using D_i as a test set
- If a tuple is misclassified, its weight is increased, it is decreased
- Error rate: err($\mathbf{X_j}$) is the misclassification error of tuple $\mathbf{X_j}$. Classifier M_i error rate is the sum of the weights of the misclassified tuples:

2.9.Decision Tree

Decision Tree: Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

2.9.1.Decision tree implementation using Python

Prerequisites: Decision Tree, Decision Tree Classifer , sklearn , numpy , pandas
Decision Tree is one of the most powerful and popular algorithm. Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables.

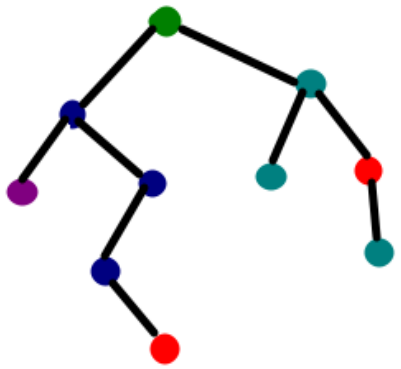


Fig 2.3 Decision tree

In this article, We are going to implement a Decision tree algorithm on the Balance scale weight and Distance DataBase presented on the UCI.

Installation of the packages :

In Python, sklearn is the package which contains all required packages to implement Machine learning algorithm. You can install the sklearn package by following the commands given below.

2.9.2.using pip :

```
pip install -U scikit-learn
```

Before using the above command make sure you have *scipy* and *numpy* packages installed.
If you don't have pip. You can install it using

```
python get-pip.py
```

2.9.3.using conda :

```
conda install scikit-learn
```

3.Analysis of dataset:

First we import the os to read the file. here the file format is csv extension file using this we can import the file

3.1 ML | Linear Regression

Linear Regression is a machine learning algorithm based on **supervised learning**. It performs a **regression task**. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

Hypothesis function for Linear Regression :

$$y = \theta_1 + \theta_2.x$$

While training the model we are given :
x: input training data (univariate – one input variable(parameter))
y: labels to data (supervised learning)

When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ_1 and θ_2 values.
θ_1: intercept
θ_2: coefficient of x

Once we find the best θ_1 and θ_2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

How to update θ_1 and θ_2 values to get the best fit line ?

Cost Function(J):

By achieving the best-fit regression line, the model aims to predict y value such that the error difference between predicted value and true value is minimum. So, it is very important to update the θ_1 and θ_2 values, to reach the best value that minimize the error between predicted y value (pred) and true y value (y).

$$minimize \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

$$J = \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

Cost function(J) of Linear Regression is the **Root Mean Squared Error (RMSE)** between predicted y value (pred) and true y value (y).

3.2 Multiple Linear Regression:

What Is Multiple Linear Regression – MLR?

Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression (MLR) is to model the linear relationship between the explanatory (independent) variables and response (dependent) variable.

In essence, multiple regression is the extension of ordinary least-squares (OLS) regression that involves more than one explanatory variable.

Explaining Multiple Linear Regression

A simple linear regression is a function that allows an analyst or statistician to make predictions about one variable based on the information that is known about another variable. Linear regression can only be used when one has two continuous variables—an independent variable and a dependent variable. The independent variable is the parameter that is used to calculate the dependent variable or outcome. A multiple regression model extends to several explanatory variables.

The multiple regression model is based on the following assumptions:

- There is a linear relationship between the dependent variables and the independent variables.
- The independent variables are not too highly correlated with each other.
- y_i observations are selected independently and randomly from the population.
- Residuals should be normally distributed with a mean of 0 and variance σ.

The coefficient of determination (R-squared) is a statistical metric that is used to measure how much of the variation in outcome can be explained by the variation in the independent variables. R^2 always increases as more predictors are added to the MLR model even though the predictors may not be related to the outcome variable.

R^2 by itself can't thus be used to identify which predictors should be included in a model and which should be excluded. R^2 can only be between 0 and 1, where 0 indicates that the outcome cannot be predicted by any of the independent variables and 1 indicates that the outcome can be predicted without error from the independent variables.

When interpreting the results of a multiple regression, beta coefficients are valid while holding all other variables constant ("all else equal"). The output from a multiple regression can be displayed horizontally as an equation, or vertically in table form.

3.3 Classification in ML:

A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”. A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes.
For example, when filtering emails “spam” or “not spam”, when looking at transaction data, “fraudulent”, or “authorized”. In short Classification either predicts categorical class labels or

classifies data (construct a model) based on the training set and the values (class labels) in classifying attributes and uses it in classifying new data. There are a number of classification models. Classification models include logistic regression, decision tree, random forest, gradient-boosted tree, multilayer perceptron, one-vs-rest, and Naive Bayes.

example:
Which of the following is/are classification problem(s)?

- Predicting the gender of a person by his/her handwriting style
- Predicting house price based on area
- Predicting whether monsoon will be normal next year
- Predict the number of copies a music album will be sold next month

Solution : Predicting the gender of a person Predicting whether monsoon will be normal next year. As we discussed classification with some examples. Now there is an example of classification in which we are performing classification on the iris dataset using *Random Forest Classifier* in python.

3.4 Sentiment Analysis:

Sentiment Analysis also known as *Opinion Mining* is a field within Natural Language Processing (NLP) that builds systems that try to identify and extract opinions within text. Usually, besides identifying the opinion, these systems extract attributes of the expression e.g.:

- *Polarity*: if the speaker express a *positive* or *negative* opinion,
- *Subject*: the thing that is being talked about,
- *Opinion holder*: the person, or entity that expresses the opinion.

Currently, sentiment analysis is a topic of great interest and development since it has many practical applications. Since publicly and privately available information over Internet is constantly growing, a large number of texts expressing opinions are available in review sites, forums, blogs, and social media.

With the help of sentiment analysis systems, this unstructured information could be automatically transformed into structured data of public opinions about products, services, brands, politics, or any topic that people can express opinions about. This data can be very useful for commercial applications like marketing analysis, public relations, product reviews, net promoter scoring, product feedback, and customer service.

What Is an Opinion?

Text information can be broadly categorized into two main types: *facts* and *opinions*. Facts are *objective* expressions about something. Opinions are usually *subjective* expressions that describe people's sentiments, appraisals, and feelings toward a subject or topic.

Sentiment analysis, just as many other NLP problems, can be modeled as a classification problem where two sub-problems must be resolved:

- Classifying a sentence as *subjective* or *objective,* known as **subjectivity classification**.
- Classifying a sentence as expressing a *positive, negative* or *neutral* opinion, known as **polarity classification**.

In an opinion, the entity the text talks about can be an object, its components, it's aspects, its attributes, or its features. It could also be a product, a service, an individual, an organization, an event, or a topic. As an example, take a look at the opinion below:

"The battery life of this camera is too short."

A negative opinion is expressed about a feature (battery life) of an entity (camera).

Direct vs Comparative Opinions

There are two kinds of opinions: *direct* and *comparative*. Direct opinions give an opinion about a entity directly, for example:

"The picture quality of camera A is poor."

This direct opinion states a negative opinion about camera A.

In comparative opinions, the opinion is expressed by comparing an entity with another, for example:

"The picture quality of camera A is better than that of camera B."

Usually, comparative opinions express similarities or differences between two or more entities using a comparative or superlative form of an adjective or adverb. In the previous example, there's a positive opinion about camera A and, conversely, a negative opinion about camera B.

Explicit vs Implicit Opinions

An explicit opinion on a subject is an opinion explicitly expressed in a subjective sentence. The following sentence expresses an explicit positive opinion:

"The voice quality of this phone is amazing."

An implicit opinion on a subject is an opinion implied in an objective sentence. The following sentence expresses an implicit negative opinion:

"The earphone broke in two days."

Within implicit opinions we could include *metaphors* that may be the most difficult type of opinions to analyze as they include a lot of semantic information.

Sentiment Analysis Scope

Sentiment analysis can be applied at different levels of scope:

- **Document level** sentiment analysis obtains the sentiment of a complete document or paragraph.
- **Sentence level** sentiment analysis obtains the sentiment of a single sentence.
- **Sub-sentence level** sentiment analysis obtains the sentiment of sub-expressions within a sentence.

Why sentiment analysis is important?

It's estimated that **80%** of the world's data is *unstructured* and *not organized* in a pre-defined manner. Most of this comes from text data, like emails, support tickets, chats, social media, surveys, articles, and documents. These texts are usually difficult, time-consuming and expensive to analyze, understand, and sort through.

Sentiment analysis systems allows companies to make sense of this sea of unstructured text by automating business processes, getting actionable insights, and saving hours of manual data processing, in other words, by making teams more efficient.

Some of the advantages of sentiment analysis include the following:

- **Scalability**:

Can you imagine manually sorting through thousands of tweets, customer support conversations, or customer reviews? There's just too much data to process manually. Sentiment analysis allows to process data at scale in an efficient and cost-effective way.

- **Real-time analysis**:

We can use sentiment analysis to identify critical information that allows situational awareness during specific scenarios in real-time. Is there a PR crisis in social media about to burst? An angry customer that is about to churn? A sentiment analysis system can help you immediately identify these kinds of situations and take action.

- **Consistent criteria**:

Humans don't observe clear criteria for evaluating the sentiment of a piece of text. It's estimated that different people only agree around 60–65% of the times when judging the sentiment for a particular piece of text. It's a subjective task which is heavily influenced by personal experiences,

thoughts, and beliefs. By using a centralized sentiment analysis system, companies can apply the same criteria to all of their data. This helps to reduce errors and improve data consistency

4. Functionality of Algorithms:

Classification:

Can you predict the patient's condition based on the review?

Regression:

Can you predict the rating of the drug based on the review?

Sentiment analysis:

What elements of a review make it more helpful to others?

Which patients tend to have more negative reviews?

Can you determine if a review is positive, neutral, or negative?

5. IMPLEMENTATION

The exchange of a commodity for money like clothes, furniture for money, the action of selling something is called Sales and data is the quantities, characters or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic etc., In other words simply we can say that things known or assumed as facts, making the basis of reasoning.

There are mainly two types of data

5.1 Categorical data: Data that is collected can be either categorical or numerical data. Numbers often don't make sense unless you assign meaning to those numbers. Categorical data helps you do that Categorical is when numbers are collected in groups or categories. It is further divided into two types

- Dichotomous: Dichotomous variables are nominal variables which have only two categories (Binary) or levels. For example, if we were looking at gender, we would most probably categorize somebody as either “male” or “female”.
- Non Dichotomous: Non Dichotomous are nominal variables which have more than two categories or levels. For example, if we were looking at grade in a semester, we would categorize many grades (A,B,C,D….).

5.2 Numeric data: Numeric data is a data that is measurable, such as time, height, weight, amount, and so on. You can help yourself identify numeric data by seeing if you can average or order the data in either ascending or descending order.

The Sales data we can assume the facts for making few computations and predictions, also reduce the relation between feature and parameters that leads to higher turnover of the Super Store.

Our data is consistent of both Categorical and Numerical. Here our aim is to drop all the unnecessary features and assign numerical equivalence to categorical features and to perform linear regression to predict the Profits of the company.

6.DATA USED

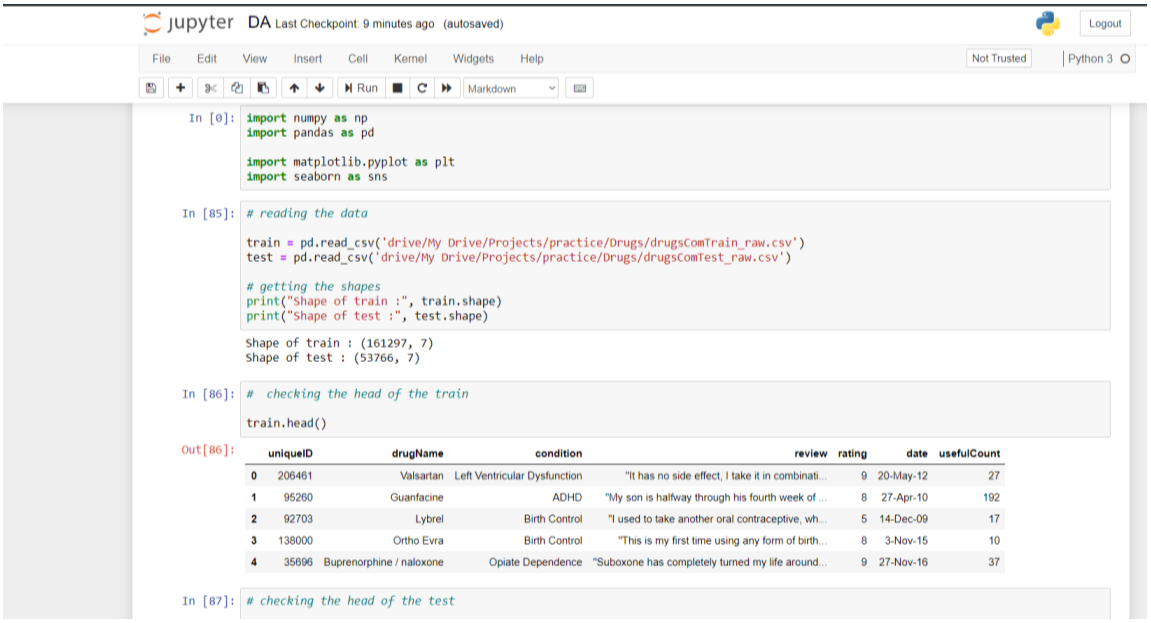
- Machine learning has permeated nearly all fields and disciplines of study. One hot topic is using natural language processing and sentiment analysis to identify, extract, and make use of subjective information.
- The UCI ML Drug Review dataset provides patient reviews on specific drugs along with related conditions and a 10-star patient rating system reflecting overall patient satisfaction. The data was obtained by crawling online pharmaceutical review sites.
- This data was published in a study on sentiment analysis of drug experience over multiple facets, ex. sentiments learned on specific aspects such as effectiveness and side effects (see the acknowledgments section to learn more).

7.TOOLS AND TECHNOLOGIES:

- Jupyter Notebook
- Anacoda navigator
- Python compiler

Presentation.pdf

8)OUTPUTS:



jupyter DA Last Checkpoint: 9 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Markdown

In [0]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
```

In [85]:
```
# reading the data

train = pd.read_csv('drive/My Drive/Projects/practice/Drugs/drugsComTrain_raw.csv')
test = pd.read_csv('drive/My Drive/Projects/practice/Drugs/drugsComTest_raw.csv')

# getting the shapes
print("Shape of train :", train.shape)
print("Shape of test :", test.shape)
```

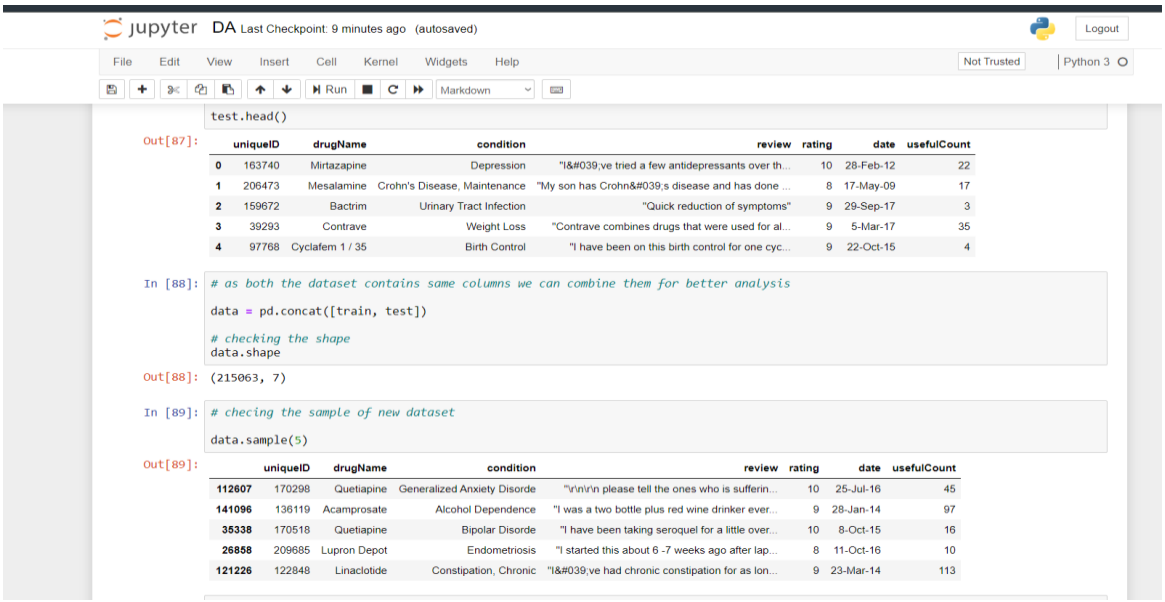
```
Shape of train : (161297, 7)
Shape of test : (53766, 7)
```

In [86]:
```
#  checking the head of the train

train.head()
```

Out[86]:

	uniqueID	drugName	condition	review	rating	date	usefulCount
0	206461	Valsartan	Left Ventricular Dysfunction	"It has no side effect, I take it in combinati...	9	20-May-12	27
1	95260	Guanfacine	ADHD	"My son is halfway through his fourth week of ...	8	27-Apr-10	192
2	92703	Lybrel	Birth Control	"I used to take another oral contraceptive, wh...	5	14-Dec-09	17
3	138000	Ortho Evra	Birth Control	"This is my first time using any form of birth...	8	3-Nov-15	10
4	35696	Buprenorphine / naloxone	Opiate Dependence	"Suboxone has completely turned my life around...	9	27-Nov-16	37

In [87]:
```
# checking the head of the test
```



jupyter DA Last Checkpoint: 9 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Markdown

```
test.head()
```

Out[87]:

	uniqueID	drugName	condition	review	rating	date	usefulCount
0	163740	Mirtazapine	Depression	"I've tried a few antidepressants over th...	10	28-Feb-12	22
1	206473	Mesalamine	Crohn's Disease, Maintenance	"My son has Crohn's disease and has done ...	8	17-May-09	17
2	159672	Bactrim	Urinary Tract Infection	"Quick reduction of symptoms"	9	29-Sep-17	3
3	39293	Contrave	Weight Loss	"Contrave combines drugs that were used for al...	9	5-Mar-17	35
4	97768	Cyclafem 1 / 35	Birth Control	"I have been on this birth control for one cyc...	9	22-Oct-15	4

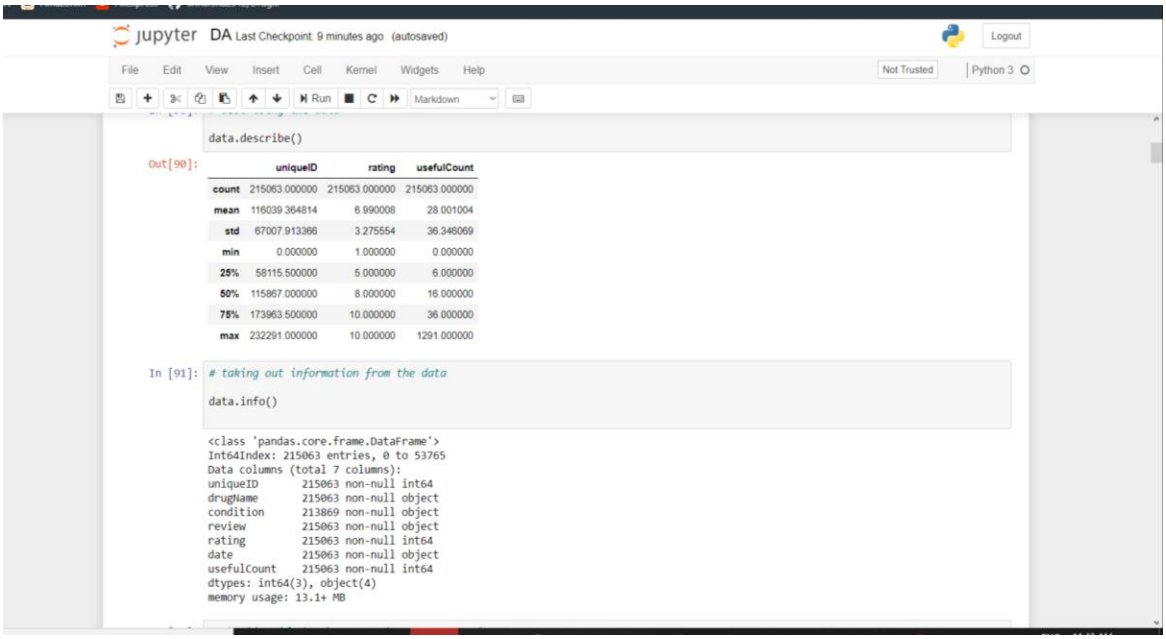
In [88]:
```
# as both the dataset contains same columns we can combine them for better analysis

data = pd.concat([train, test])

# checking the shape
data.shape
```

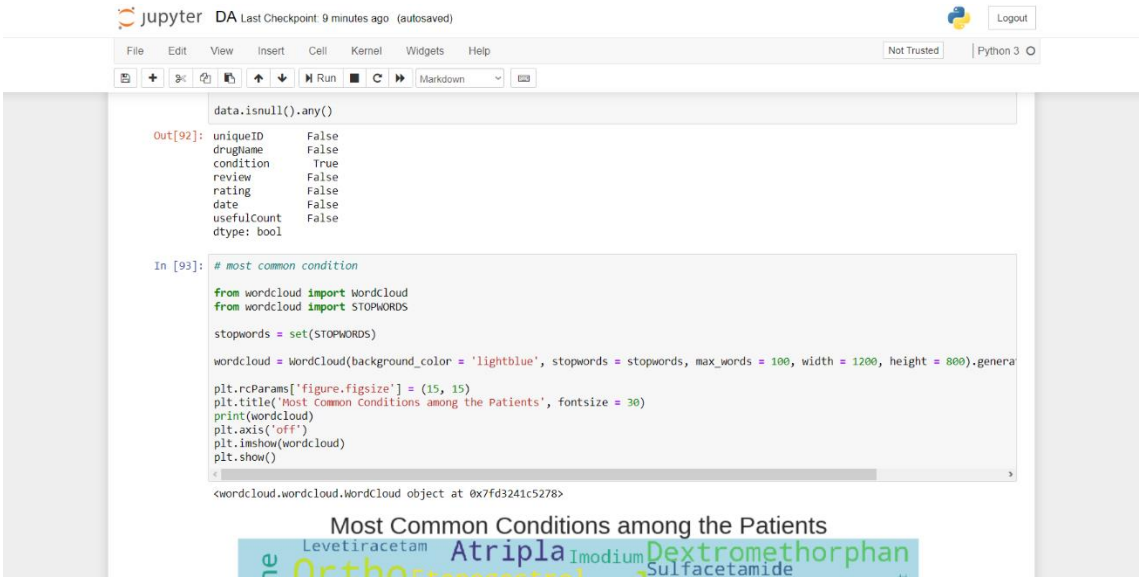
Out[88]:
```
(215063, 7)
```

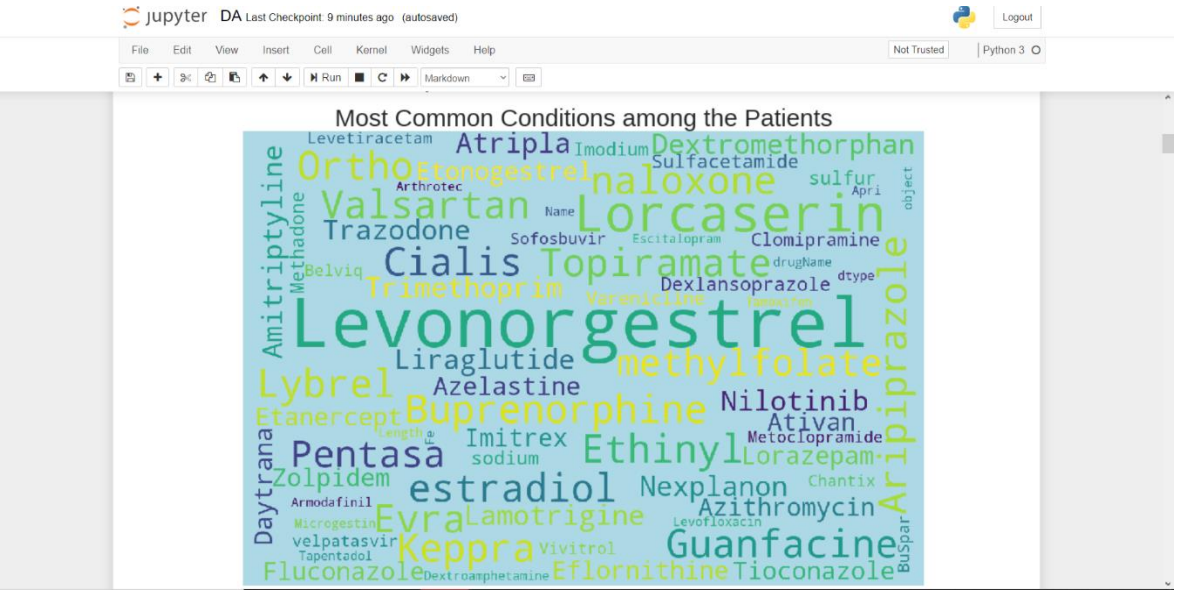
In [89]:
```
# checing the sample of new dataset

data.sample(5)
```

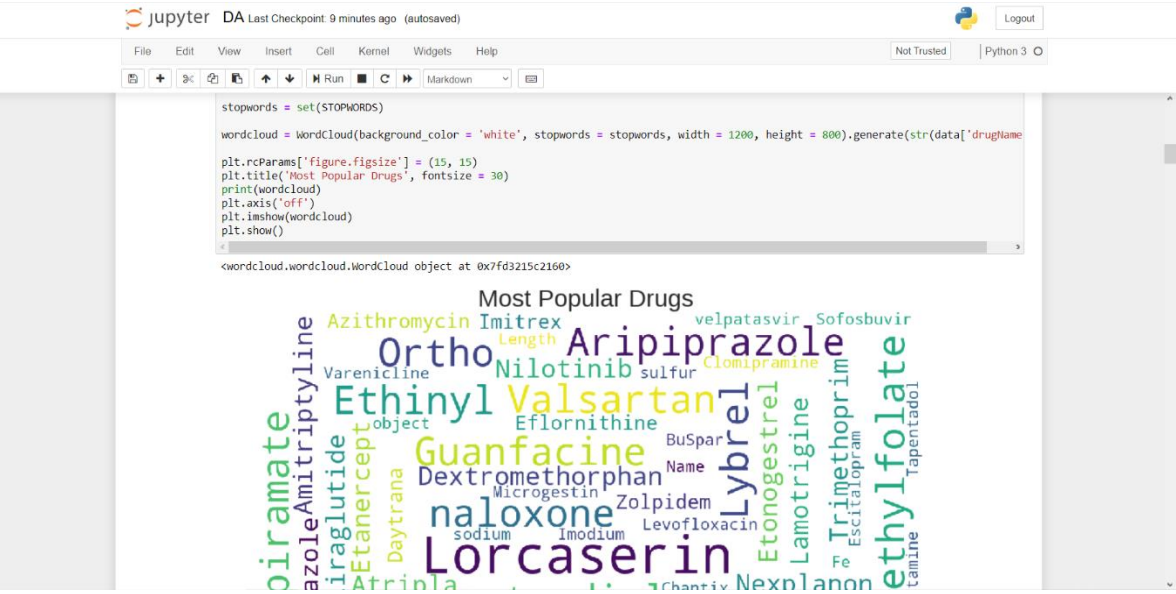
Out[89]:

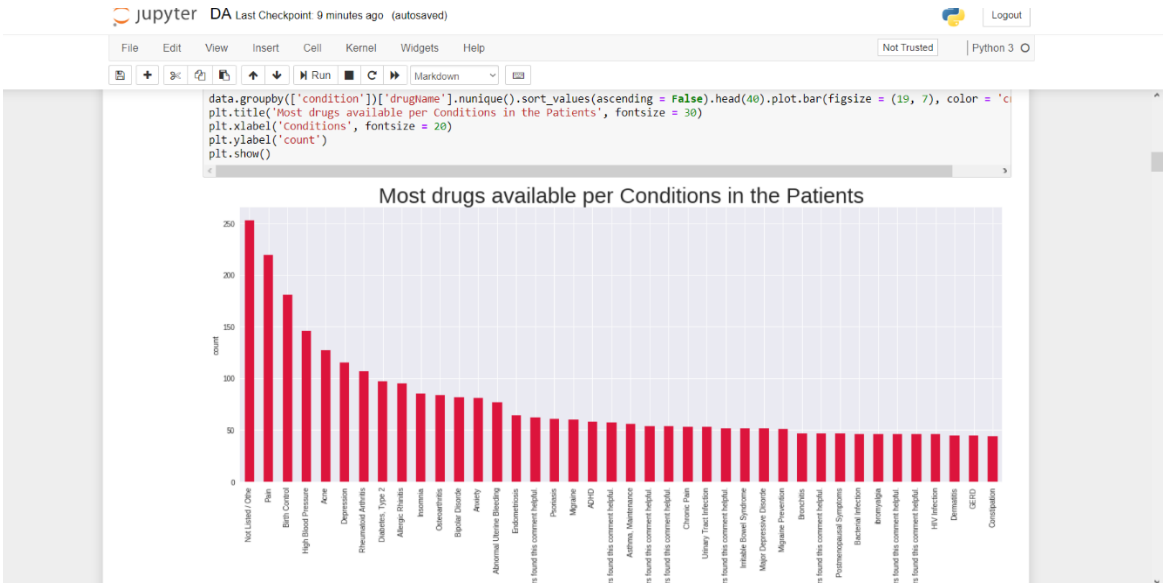
	uniqueID	drugName	condition	review	rating	date	usefulCount
112607	170298	Quetiapine	Generalized Anxiety Disorde	"\r\n\r\n please tell the ones who is sufferin...	10	25-Jul-16	45
141096	136119	Acamprosate	Alcohol Dependence	"I was a two bottle plus red wine drinker ever...	9	28-Jan-14	97
35338	170518	Quetiapine	Bipolar Disorde	"I have been taking seroquel for a little over...	10	8-Oct-15	16
26858	209685	Lupron Depot	Endometriosis	"I started this about 6 -7 weeks ago after lap...	8	11-Oct-16	10
121226	122848	Linaclotide	Constipation, Chronic	"I've had chronic constipation for as lon...	9	23-Mar-14	113

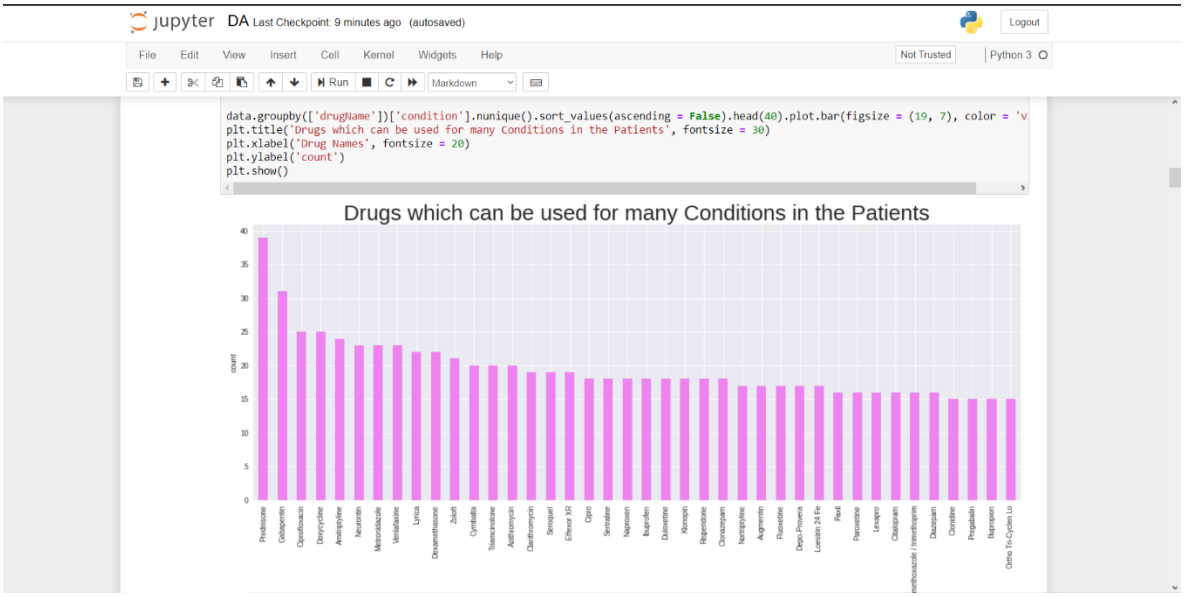

jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
data.describe()
Out[90]:
uniqueID rating usefulCount
count 215063.000000 215063.000000 215063.000000
mean 116039.364814 6.990008 28.001004
std 67007.913366 3.275554 36.346069
min 0.000000 1.000000 0.000000
25% 58115.500000 5.000000 6.000000
50% 115867.000000 8.000000 16.000000
75% 173963.500000 10.000000 36.000000
max 232291.000000 10.000000 1291.000000
In [91]:
# taking out information from the data
data.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 215063 entries, 0 to 53765
Data columns (total 7 columns):
uniqueID 215063 non-null int64
drugName 215063 non-null object
condition 213869 non-null object
review 215063 non-null object
rating 215063 non-null int64
date 215063 non-null object
usefulCount 215063 non-null int64
dtypes: int64(3), object(4)
memory usage: 13.1+ MB

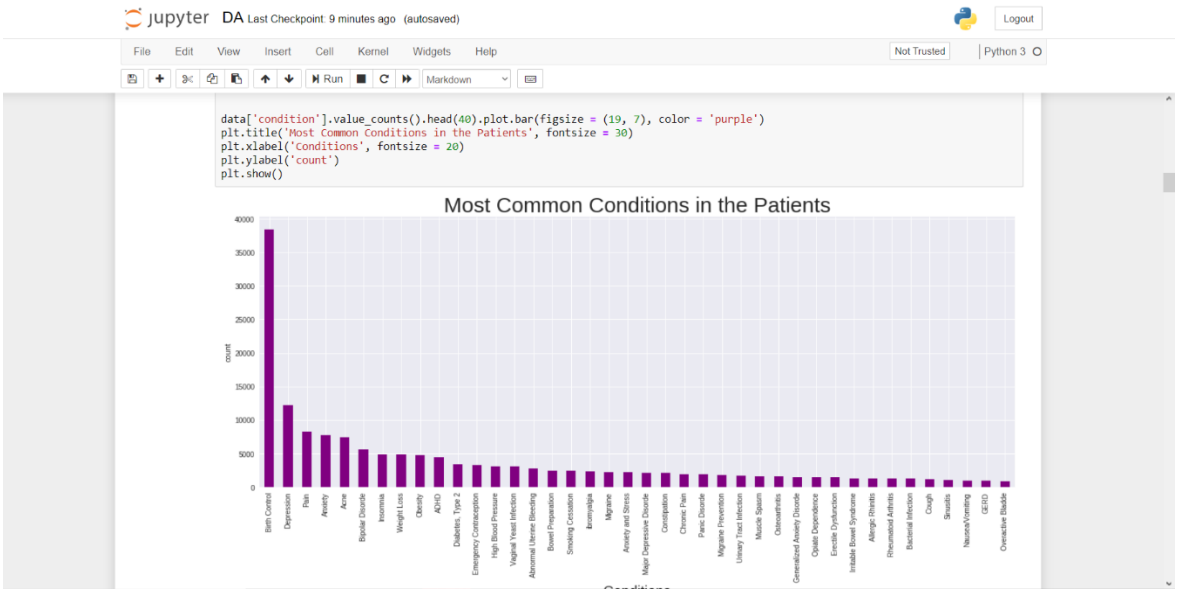

jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
data.isnull().any()
Out[92]:
uniqueID False
drugName False
condition True
review False
rating False
date False
usefulCount False
dtype: bool
In [93]:
# most common condition
from wordcloud import WordCloud
from wordcloud import STOPWORDS
stopwords = set(STOPWORDS)
wordcloud = WordCloud(background_color = 'lightblue', stopwords = stopwords, max_words = 100, width = 1200, height = 800).genera
plt.rcParams['figure.figsize'] = (15, 15)
plt.title('Most Common Conditions among the Patients', fontsize = 30)
print(wordcloud)
plt.axis('off')
plt.imshow(wordcloud)
plt.show()
<wordcloud.wordcloud.WordCloud object at 0x7fd3241c5278>
Most Common Conditions among the Patients
Levetiracetam
Atripla
Imodium
Dextromethorphan
Sulfacetamide

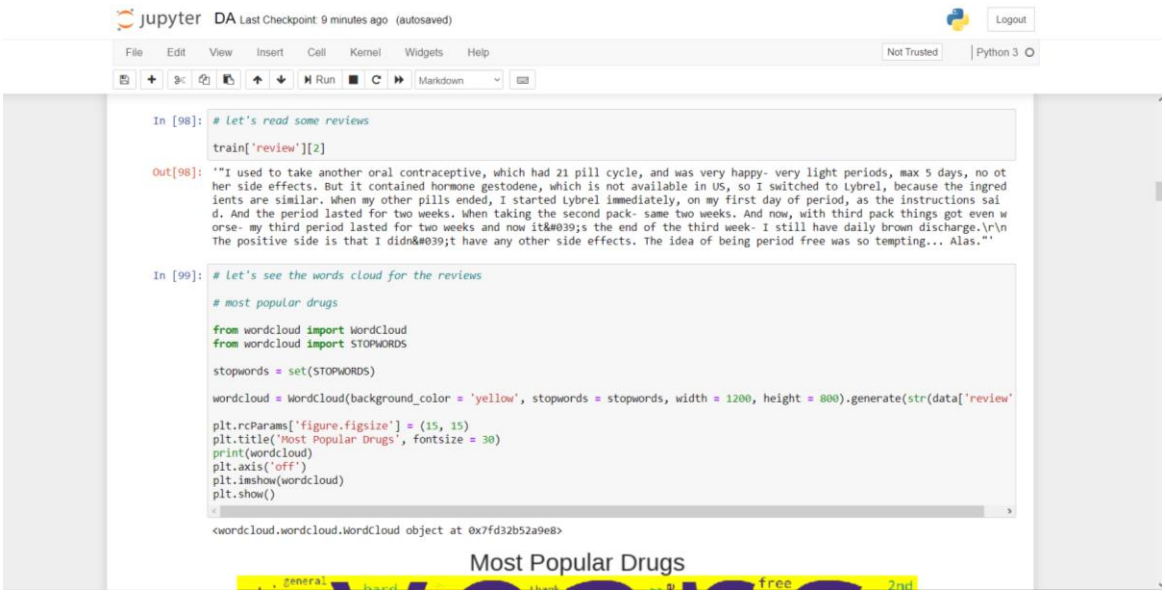

jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
Most Common Conditions among the Patients
Levonorgestrel
Lorcaserin
Valsartan
Cialis
Topiramate
Ortho
naloxone
Dextromethorphan
Atripla
Levetiracetam
Imodium
Sulfacetamide
Arthrotec
Etonogestrel
sulfur
Trazodone
Sofosbuvir
Escitalopram
Clomipramine
Belviq
Trimethoprim
Dexlansoprazole
drugName
dtype
Varenicline
Liraglutide
methylfolate
Lybrel
Azelastine
Nilotinib
Ativan
Etanercept
Buprenorphine
Imitrex
Metoclopramide
Pentasa
sodium
Ethinyl
Lorazepam
Zolpidem
estradiol
Nexplanon
Chantix
Armodafinil
Azithromycin
Evra
Lamotrigine
Microgestin
Levofloxacin
velpatasvir
Keppra
Guanfacine
Tapentadol
Vivitrol
Fluconazole
Dextroamphetamine
Eflornithine
Tioconazole
Amitriptyline
Methadone
Daytrana
Aripiprazole
BuSpar
object


jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
stopwords = set(STOPWORDS)
wordcloud = WordCloud(background_color = 'white', stopwords = stopwords, width = 1200, height = 800).generate(str(data['drugName
plt.rcParams['figure.figsize'] = (15, 15)
plt.title('Most Popular Drugs', fontsize = 30)
print(wordcloud)
plt.axis('off')
plt.imshow(wordcloud)
plt.show()
<wordcloud.wordcloud.WordCloud object at 0x7fd3215c2160>
Most Popular Drugs
Aripiprazole
Ortho
Ethinyl
Valsartan
Guanfacine
naloxone
Lorcaserin
Lybrel
Nilotinib
Dextromethorphan
Azithromycin
Imitrex
velpatasvir
Sofosbuvir
Length
Varenicline
sulfur
Clomipramine
Eflornithine
object
BuSpar
Name
Microgestin
Zolpidem
Levofloxacin
sodium
Imodium
Etonogestrel
Lamotrigine
Trimethoprim
Escitalopram
Tapentadol
Daytrana
Etanercept
Amitriptyline
Chantix
Nexplanon


jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
data.groupby(['condition'])['drugName'].nunique().sort_values(ascending = False).head(40).plot.bar(figsize = (19, 7), color = 'c
plt.title('Most drugs available per Conditions in the Patients', fontsize = 30)
plt.xlabel('Conditions', fontsize = 20)
plt.ylabel('count')
plt.show()
Most drugs available per Conditions in the Patients
count
Not Listed / Othe
Pain
Birth Control
High Blood Pressure
Acne
Depression
Rheumatoid Arthritis
Diabetes, Type 2
Allergic Rhinitis
Insomnia
Osteoarthritis
Bipolar Disorde
Anxiety
Abnormal Uterine Bleeding
Endometriosis
Psoriasis
Migraine
ADHD
Asthma, Maintenance
Chronic Pain
Urinary Tract Infection
Irritable Bowel Syndrome
Major Depressive Disorde
Migraine Prevention
Bronchitis
Postmenopausal Symptoms
Bacterial Infection
Dermatitis
GERD
Constipation


jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
data.groupby(['drugName'])['condition'].nunique().sort_values(ascending = False).head(40).plot.bar(figsize = (19, 7), color = 'v
plt.title('Drugs which can be used for many Conditions in the Patients', fontsize = 30)
plt.xlabel('Drug Names', fontsize = 20)
plt.ylabel('count')
plt.show()
Drugs which can be used for many Conditions in the Patients
count
Prednisone
Gabapentin
Ciprofloxacin
Doxycycline
Amitriptyline
Neurontin
Metronidazole
Venlafaxine
Lyrica
Dexamethasone
Zoloft
Cymbalta
Triamcinolone
Azithromycin
Clarithromycin
Seroquel
Effexor XR
Cipro
Sertraline
Naproxen
Ibuprofen
Duloxetine
Klonopin
Risperidone
Clonazepam
Nortriptyline
Augmentin
Fluoxetine
Depo-Provera
Lorazepam 24 Fe
Paxil
Paroxetine
Lexapro
Citalopram
methocarbamol / trimethoprim
Diazepam
Clonidine
Pregabalin
Bupropion
Ortho Tri-Cyclen Lo


jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
data['condition'].value_counts().head(40).plot.bar(figsize = (19, 7), color = 'purple')
plt.title('Most Common Conditions in the Patients', fontsize = 30)
plt.xlabel('Conditions', fontsize = 20)
plt.ylabel('count')
plt.show()
Most Common Conditions in the Patients
count
Conditions


jupyter DA Last Checkpoint: 9 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
In [98]:
let's read some reviews
train['review'][2]
Out[98]:
'"I used to take another oral contraceptive, which had 21 pill cycle, and was very happy- very light periods, max 5 days, no ot
her side effects. But it contained hormone gestodene, which is not available in US, so I switched to Lybrel, because the ingred
ients are similar. When my other pills ended, I started Lybrel immediately, on my first day of period, as the instructions sai
d. And the period lasted for two weeks. When taking the second pack- same two weeks. And now, with third pack things got even w
orse- my third period lasted for two weeks and now it's the end of the third week- I still have daily brown discharge.\r\n
The positive side is that I didn't have any other side effects. The idea of being period free was so tempting... Alas."'
In [99]:
# let's see the words cloud for the reviews
# most popular drugs
from wordcloud import WordCloud
from wordcloud import STOPWORDS
stopwords = set(STOPWORDS)
wordcloud = WordCloud(background_color = 'yellow', stopwords = stopwords, width = 1200, height = 800).generate(str(data['review'
plt.rcParams['figure.figsize'] = (15, 15)
plt.title('Most Popular Drugs', fontsize = 30)
print(wordcloud)
plt.axis('off')
plt.imshow(wordcloud)
plt.show()
<wordcloud.wordcloud.WordCloud object at 0x7fd32b52a9e8>
Most Popular Drugs
general
free
2nd

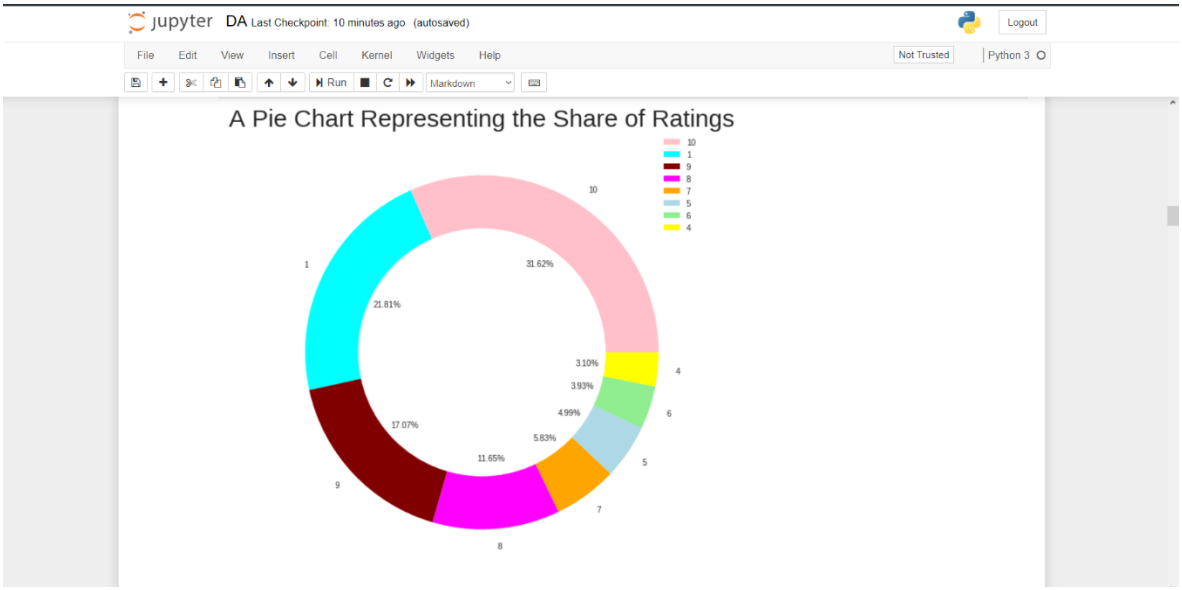
jupyter DA Last Checkpoint: 9 minutes ago (autosaved)

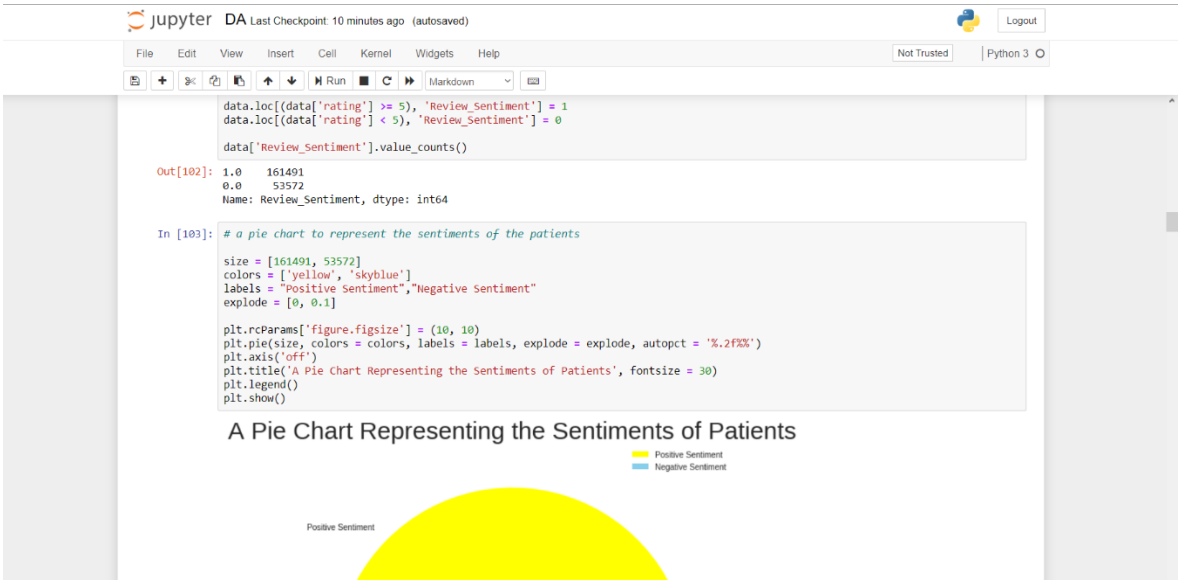
jupyter DA Last Checkpoint: 10 minutes ago (autosaved)

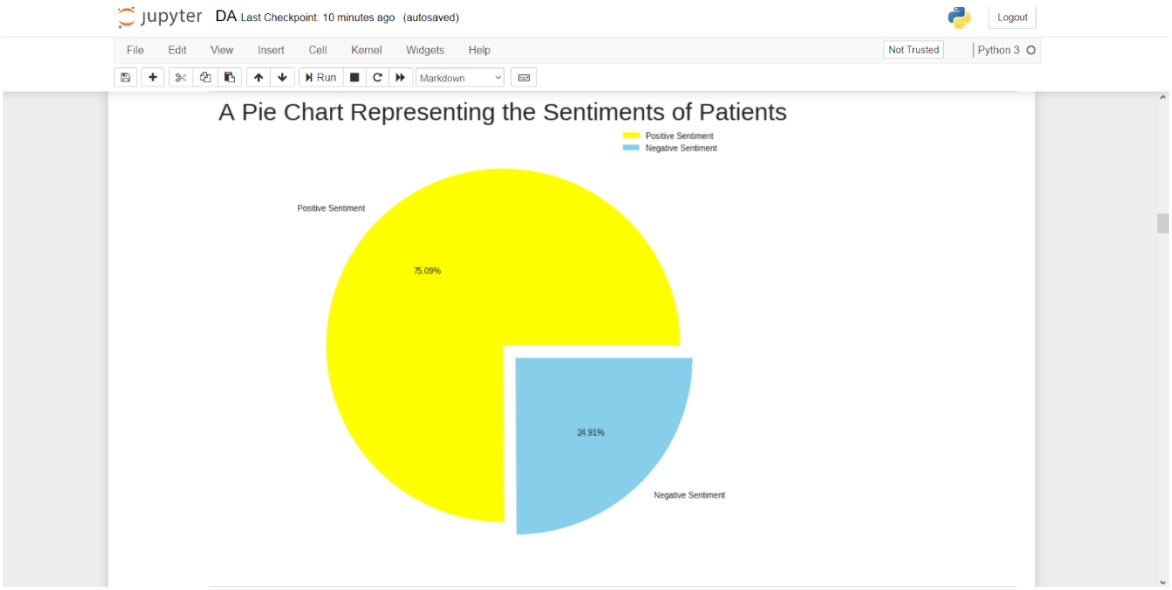
In [100]:
```
data['rating'].value_counts()
```

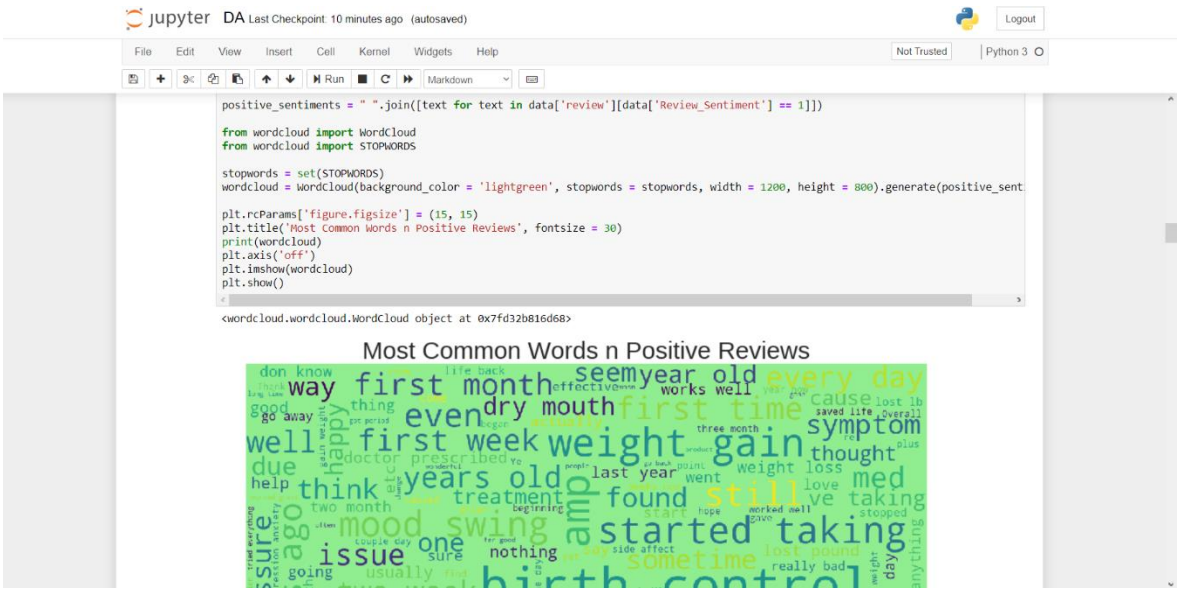
Out[100]:
```
10    68005
9     36708
1     28918
8     25046
7     12547
5     10723
2      9265
3      8718
6      8462
4      6671
Name: rating, dtype: int64
```

In [101]:
```
# making a donut chart to represent share of each ratings

size = [68005, 46901, 36708, 25046, 12547, 10723, 8462, 6671]
colors = ['pink', 'cyan', 'maroon',  'magenta', 'orange', 'lightblue', 'lightgreen', 'yellow']
labels = "10", "1", "9", "8", "7", "5", "6", "4"

my_circle = plt.Circle((0, 0), 0.7, color = 'white')

plt.rcParams['figure.figsize'] = (10, 10)
plt.pie(size, colors = colors, labels = labels, autopct = '%.2f%%')
plt.axis('off')
plt.title('A Pie Chart Representing the Share of Ratings', fontsize = 30)
p = plt.gcf()
plt.gca().add_artist(my_circle)
plt.legend()
plt.show()
```


jupyter DA Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
A Pie Chart Representing the Share of Ratings
10
1
9
8
7
5
6
4
31.62%
21.81%
17.07%
11.65%
5.83%
4.99%
3.93%
3.10%


jupyter DA Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
data.loc[(data['rating'] >= 5), 'Review_Sentiment'] = 1
data.loc[(data['rating'] < 5), 'Review_Sentiment'] = 0
data['Review_Sentiment'].value_counts()
Out[102]:
1.0 161491
0.0 53572
Name: Review_Sentiment, dtype: int64
In [103]:
# a pie chart to represent the sentiments of the patients
size = [161491, 53572]
colors = ['yellow', 'skyblue']
labels = "Positive Sentiment","Negative Sentiment"
explode = [0, 0.1]
plt.rcParams['figure.figsize'] = (10, 10)
plt.pie(size, colors = colors, labels = labels, explode = explode, autopct = '%.2f%%')
plt.axis('off')
plt.title('A Pie Chart Representing the Sentiments of Patients', fontsize = 30)
plt.legend()
plt.show()
A Pie Chart Representing the Sentiments of Patients
Positive Sentiment
Negative Sentiment
Positive Sentiment


jupyter DA Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
A Pie Chart Representing the Sentiments of Patients
Positive Sentiment
Negative Sentiment
Positive Sentiment
75.09%
24.91%
Negative Sentiment


jupyter DA Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
positive_sentiments = " ".join([text for text in data['review'][data['Review_Sentiment'] == 1]])
from wordcloud import WordCloud
from wordcloud import STOPWORDS
stopwords = set(STOPWORDS)
wordcloud = WordCloud(background_color = 'lightgreen', stopwords = stopwords, width = 1200, height = 800).generate(positive_sent
plt.rcParams['figure.figsize'] = (15, 15)
plt.title('Most Common Words n Positive Reviews', fontsize = 30)
print(wordcloud)
plt.axis('off')
plt.imshow(wordcloud)
plt.show()
<wordcloud.wordcloud.WordCloud object at 0x7fd32b816d68>
Most Common Words n Positive Reviews

jupyter DA Last Checkpoint: 10 minutes ago (autosaved)

Logout

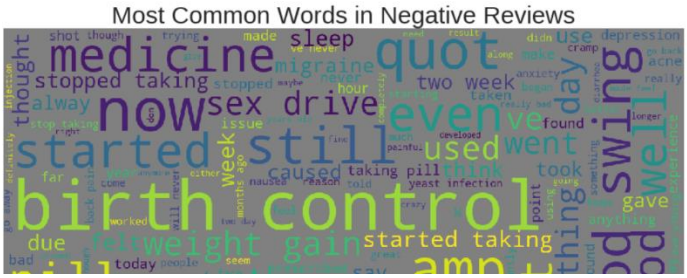
File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Markdown

```
from wordcloud import WordCloud
from wordcloud import STOPWORDS

stopwords = set(STOPWORDS)
wordcloud = WordCloud(background_color = 'grey', stopwords = stopwords, width = 1200, height = 800).generate(negative_sentiments

plt.rcParams['figure.figsize'] = (15, 15)
plt.title('Most Common Words in Negative Reviews', fontsize = 30)
print(wordcloud)
plt.axis('off')
plt.imshow(wordcloud)
plt.show()
```

```
<wordcloud.wordcloud.WordCloud object at 0x7fd32b39e898>
```



jupyter DA Last Checkpoint: 10 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Markdown

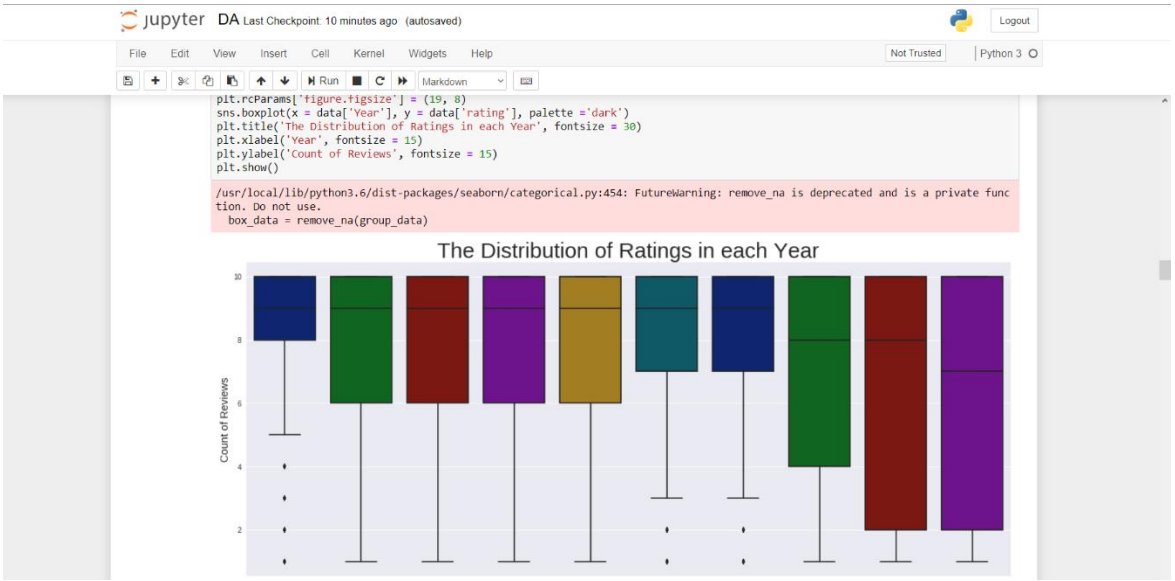
In [0]:

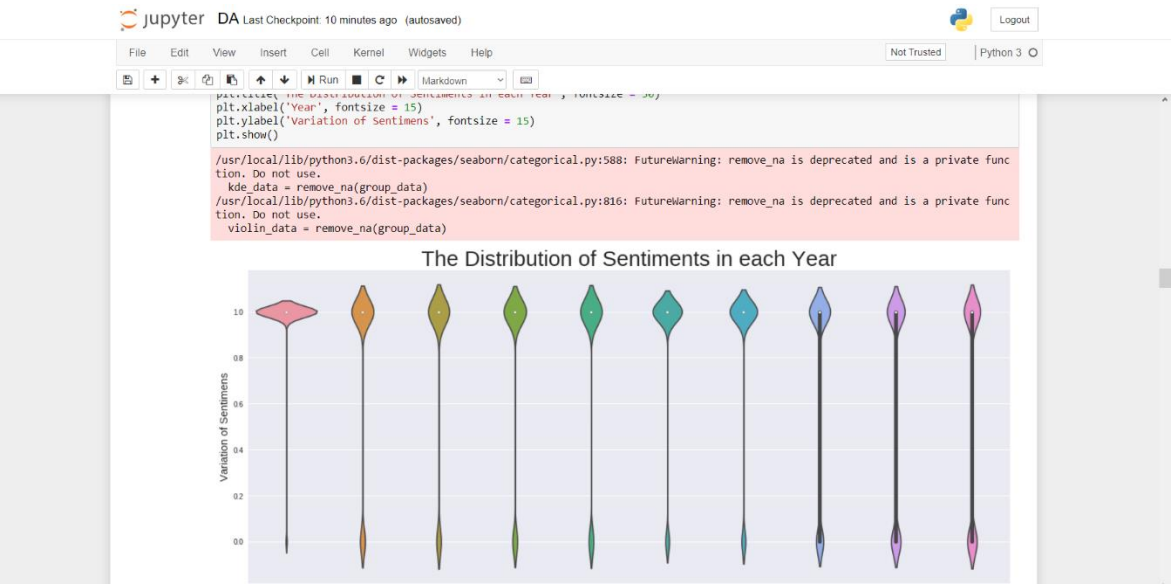
```
# converting the date into datetime format
data['date'] = pd.to_datetime(data['date'], errors = 'coerce')

# now extracting year from date
data['Year'] = data['date'].dt.year

# extracting the month from the date
data['month'] = data['date'].dt.month

# extracting the days from the date
data['day'] = data['date'].dt.day
```

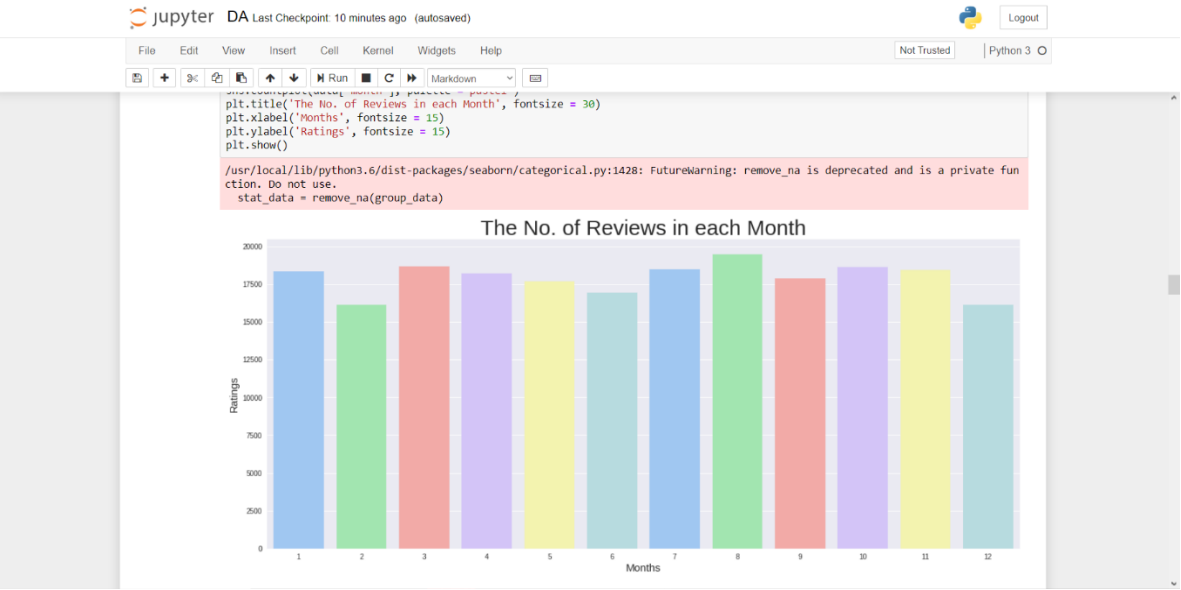
In [107]:

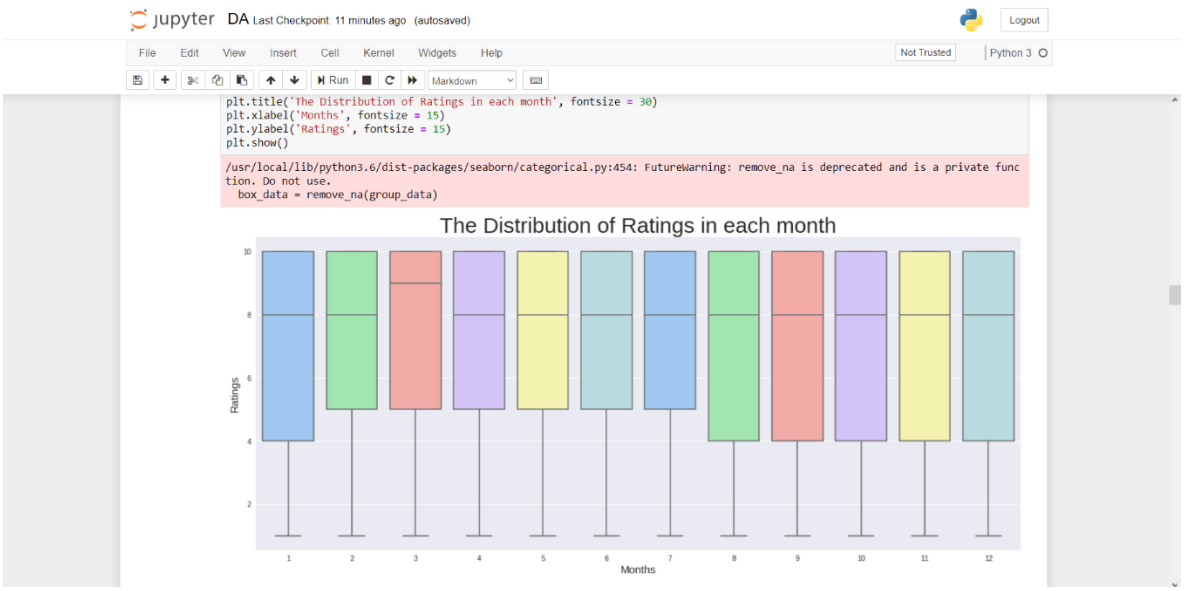
```
# looking at the no. of reviews in each of the year

plt.rcParams['figure.figsize'] = (19, 8)
sns.countplot(data['Year'], palette ='dark')
plt.title('The No. of Reviews in each year', fontsize = 30)
plt.xlabel('Year', fontsize = 15)
plt.ylabel('Count of Reviews', fontsize = 15)
plt.show()
```

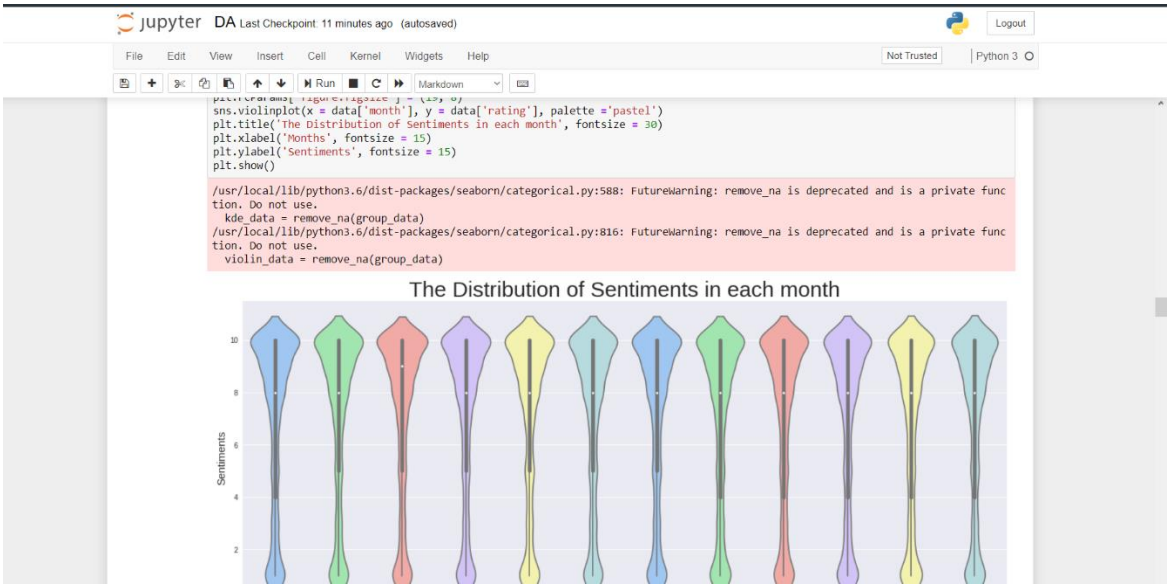
In [108]:
```
# looking at the no. of ratings in each of the year

plt.rcParams['figure.figsize'] = (19, 8)
sns.boxplot(x = data['Year'], y = data['rating'], palette ='dark')
plt.title('The Distribution of Ratings in each Year', fontsize = 30)
plt.xlabel('Year', fontsize = 15)
plt.ylabel('Count of Reviews', fontsize = 15)
```




jupyter DA Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
plt.xlabel('Year', fontsize = 15)
plt.ylabel('Variation of Sentimens', fontsize = 15)
plt.show()
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:588: FutureWarning: remove_na is deprecated and is a private function. Do not use.
kde_data = remove_na(group_data)
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:816: FutureWarning: remove_na is deprecated and is a private function. Do not use.
violin_data = remove_na(group_data)
The Distribution of Sentiments in each Year
Variation of Sentimens
1.0
0.8
0.6
0.4
0.2
0.0


jupyter DA Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
plt.title('The No. of Reviews in each Month', fontsize = 30)
plt.xlabel('Months', fontsize = 15)
plt.ylabel('Ratings', fontsize = 15)
plt.show()
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:1428: FutureWarning: remove_na is deprecated and is a private function. Do not use.
stat_data = remove_na(group_data)
The No. of Reviews in each Month
Ratings
20000
17500
15000
12500
10000
7500
5000
2500
0
1
2
3
4
5
6
7
8
9
10
11
12
Months


jupyter DA Last Checkpoint: 11 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
plt.title('The Distribution of Ratings in each month', fontsize = 30)
plt.xlabel('Months', fontsize = 15)
plt.ylabel('Ratings', fontsize = 15)
plt.show()
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:454: FutureWarning: remove_na is deprecated and is a private function. Do not use.
box_data = remove_na(group_data)
The Distribution of Ratings in each month
Ratings
Months


jupyter DA Last Checkpoint: 11 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
sns.violinplot(x = data['month'], y = data['rating'], palette ='pastel')
plt.title('The Distribution of Sentiments in each month', fontsize = 30)
plt.xlabel('Months', fontsize = 15)
plt.ylabel('Sentiments', fontsize = 15)
plt.show()
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:588: FutureWarning: remove_na is deprecated and is a private function. Do not use.
kde_data = remove_na(group_data)
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:816: FutureWarning: remove_na is deprecated and is a private function. Do not use.
violin_data = remove_na(group_data)
The Distribution of Sentiments in each month
Sentiments

jupyter DA Last Checkpoint: 11 minutes ago (autosaved)

Logout

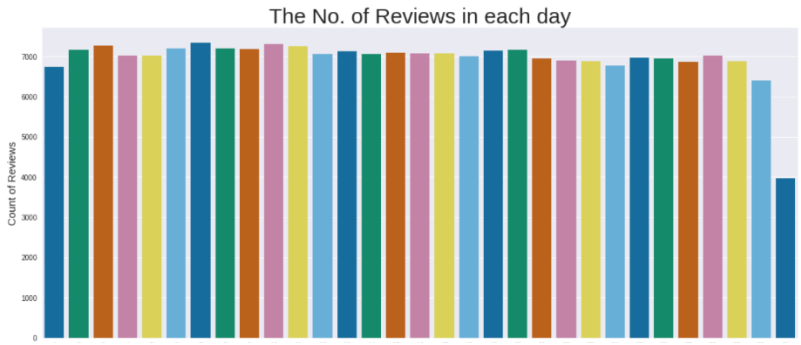
File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Markdown

```
plt.rcParams['figure.figsize'] = (19, 8)
sns.countplot(data['day'], palette ='colorblind')
plt.title('The No. of Reviews in each day', fontsize = 30)
plt.xlabel('Days', fontsize = 15)
plt.ylabel('Count of Reviews', fontsize = 15)
plt.show()
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:1428: FutureWarning: remove_na is deprecated and is a private fun
ction. Do not use.
  stat_data = remove_na(group_data)
```



jupyter DA Last Checkpoint: 11 minutes ago (autosaved)

Logout

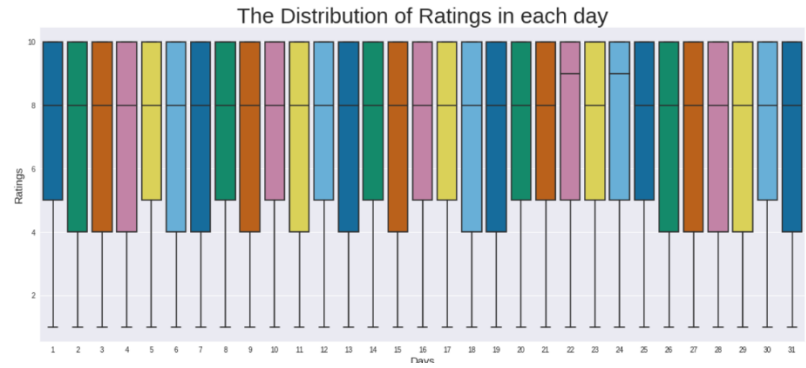
File Edit View Insert Cell Kernel Widgets Help

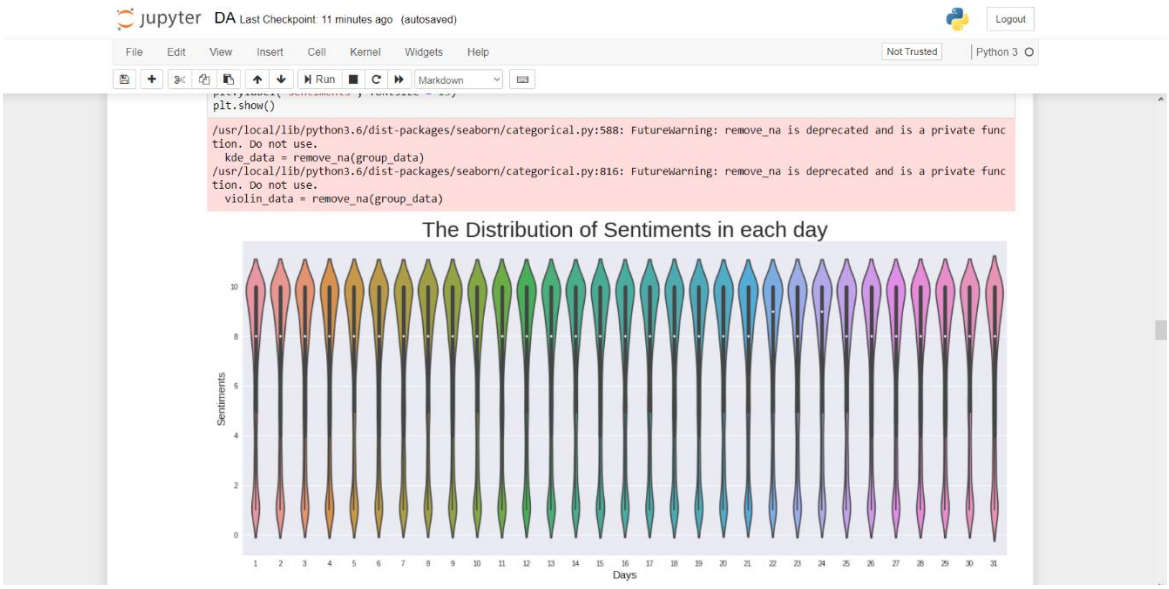
Not Trusted | Python 3

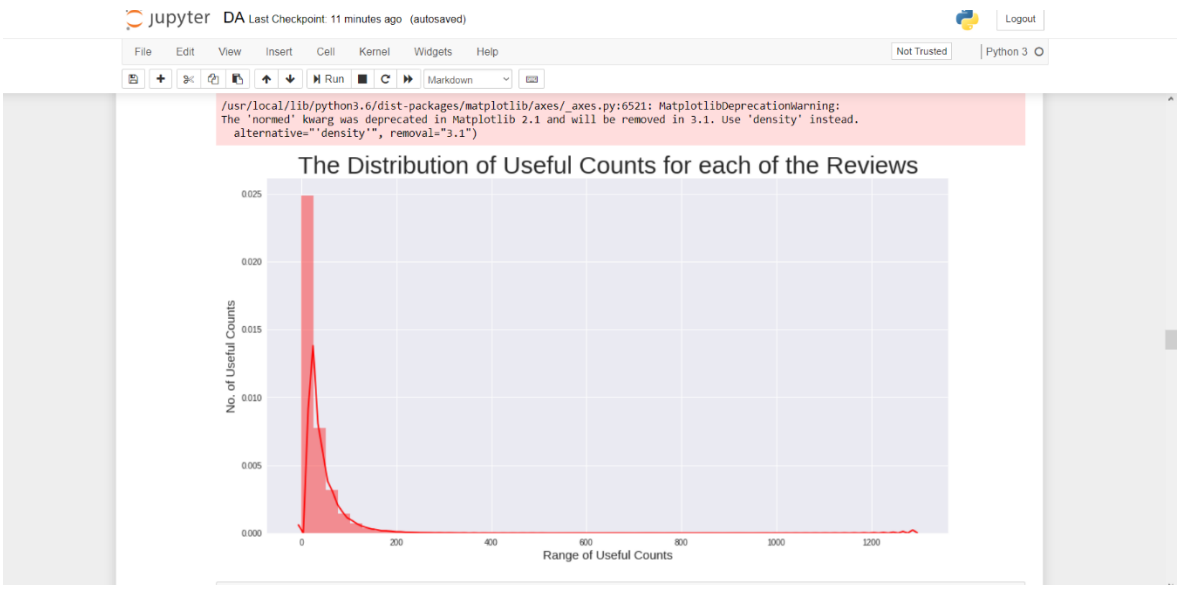
Run Markdown

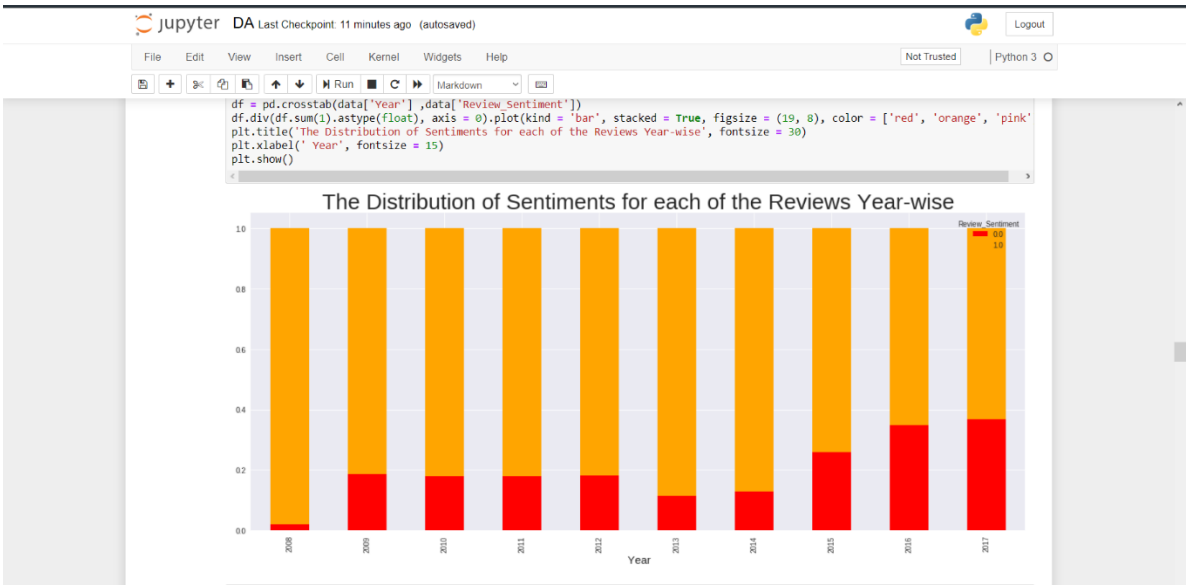
```
sns.boxplot(x = data['day'], y = data['rating'], palette ='colorblind')
plt.title('The Distribution of Ratings in each day', fontsize = 30)
plt.xlabel('Days', fontsize = 15)
plt.ylabel('Ratings', fontsize = 15)
plt.show()
```

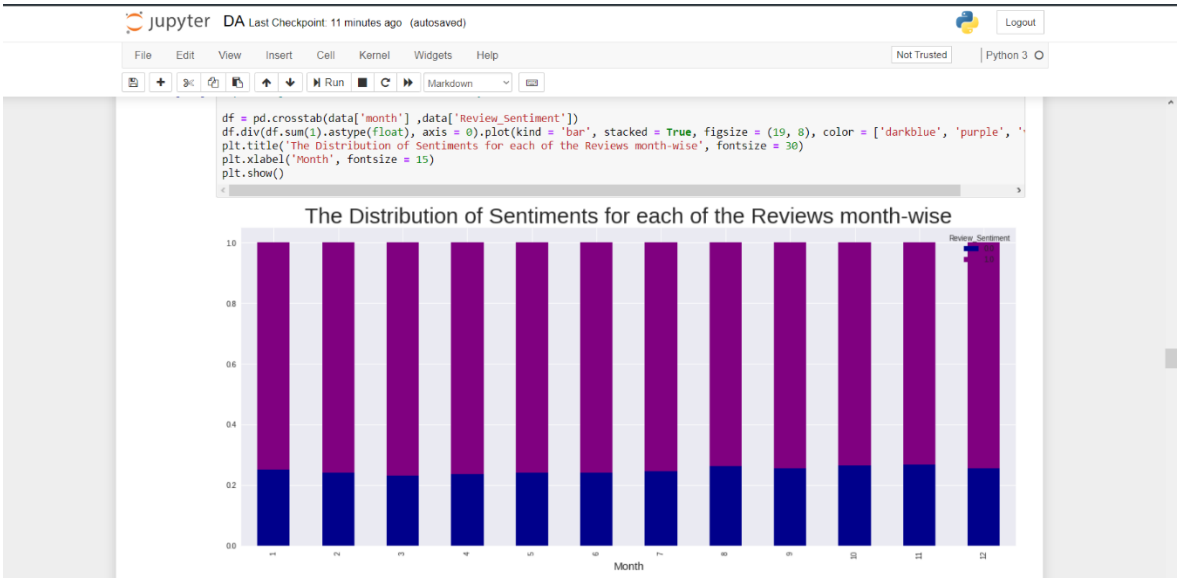
```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:454: FutureWarning: remove_na is deprecated and is a private func
tion. Do not use.
  box_data = remove_na(group_data)
```




jupyter DA Last Checkpoint: 11 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
plt.show()
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:588: FutureWarning: remove_na is deprecated and is a private function. Do not use.
kde_data = remove_na(group_data)
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:816: FutureWarning: remove_na is deprecated and is a private function. Do not use.
violin_data = remove_na(group_data)
The Distribution of Sentiments in each day
Sentiments
Days


jupyter DA Last Checkpoint: 11 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
/usr/local/lib/python3.6/dist-packages/matplotlib/axes/_axes.py:6521: MatplotlibDeprecationWarning:
The 'normed' kwarg was deprecated in Matplotlib 2.1 and will be removed in 3.1. Use 'density' instead.
alternative="'density'", removal="3.1")
The Distribution of Useful Counts for each of the Reviews
No. of Useful Counts
Range of Useful Counts


jupyter DA Last Checkpoint: 11 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
df = pd.crosstab(data['Year'] ,data['Review_Sentiment'])
df.div(df.sum(1).astype(float), axis = 0).plot(kind = 'bar', stacked = True, figsize = (19, 8), color = ['red', 'orange', 'pink'
plt.title('The Distribution of Sentiments for each of the Reviews Year-wise', fontsize = 30)
plt.xlabel(' Year', fontsize = 15)
plt.show()
The Distribution of Sentiments for each of the Reviews Year-wise
Review_Sentiment
0.0
1.0
1.0
0.8
0.6
0.4
0.2
0.0
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
Year


jupyter DA Last Checkpoint: 11 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
df = pd.crosstab(data['month'] ,data['Review_Sentiment'])
df.div(df.sum(1).astype(float), axis = 0).plot(kind = 'bar', stacked = True, figsize = (19, 8), color = ['darkblue', 'purple',
plt.title('The Distribution of Sentiments for each of the Reviews month-wise', fontsize = 30)
plt.xlabel('Month', fontsize = 15)
plt.show()
The Distribution of Sentiments for each of the Reviews month-wise
Review_Sentiment
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
7
8
9
10
11
12
Month

jupyter DA Last Checkpoint: 11 minutes ago (autosaved)

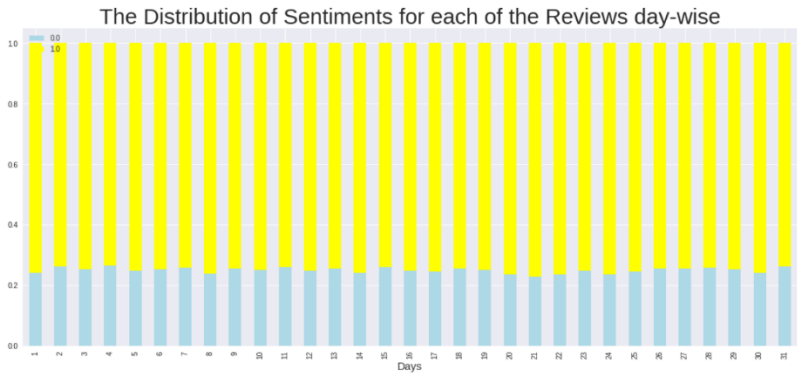
Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Markdown

```
df = pd.crosstab(data['day'] ,data['Review_Sentiment'])
df.div(df.sum(1).astype(float), axis = 0).plot(kind = 'bar', stacked = True, figsize = (19, 8), color = ['lightblue', 'yellow',
plt.title('The Distribution of Sentiments for each of the Reviews day-wise', fontsize = 30)
plt.xlabel('Days', fontsize = 15)
plt.legend(loc = 2)
plt.show()
```



jupyter DA Last Checkpoint: 11 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Markdown

In [120]:
```
data['condition'].isnull().sum()
```

Out[120]: 1194

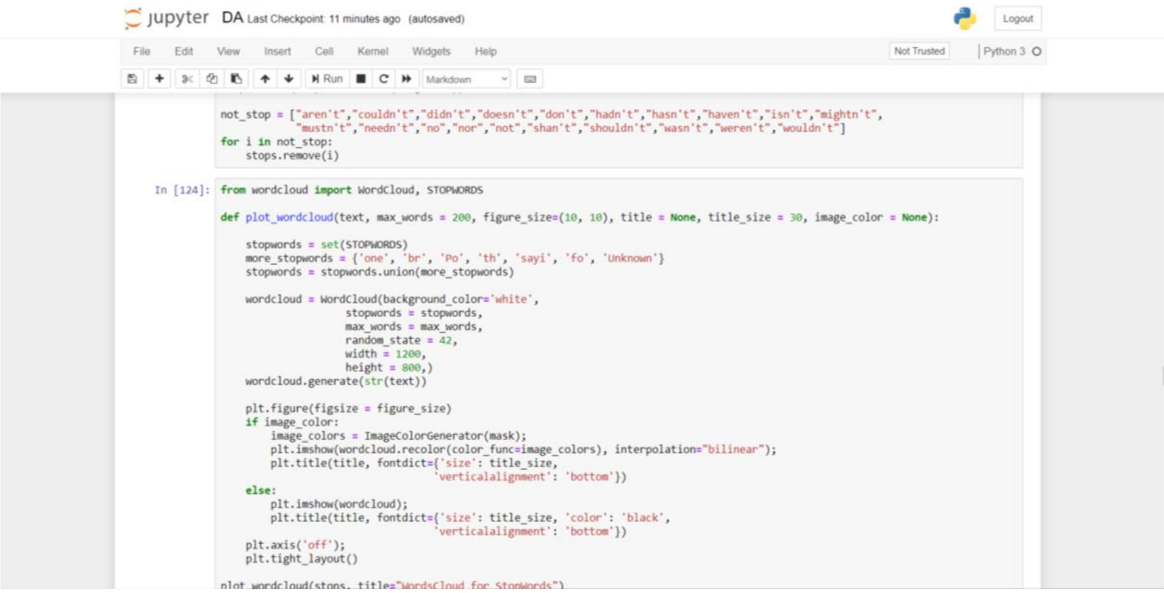
In [121]:
```
# we will delete the rows so that the data does not overfits

data = data.dropna(axis = 0)

# checking the new shape of the data
data.shape
```

Out[121]: (213869, 11)

In [122]:
```
# importing the important libraries

import re
from bs4 import BeautifulSoup
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.snowball import SnowballStemmer
from nltk.stem.porter import PorterStemmer
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

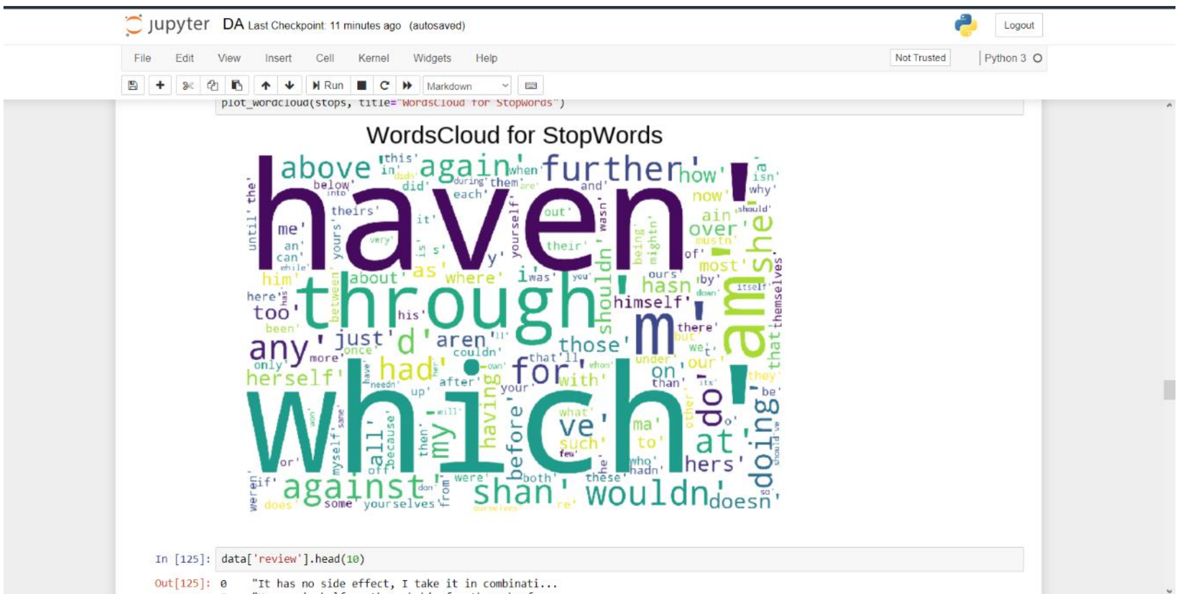
In [0]:
```
# removing some stopwords from the list of stopwords as they are important for drug recommendation

stops = set(stopwords.words('english'))

not_stop = ["aren't","couldn't","didn't","doesn't","don't","hadn't","hasn't","haven't","isn't","mightn't",
            "mustn't","needn't","no","nor","not","shan't","shouldn't","wasn't","weren't","wouldn't"]
for i in not_stop:
    stops.remove(i)
```



```python
not_stop = ["aren't","couldn't","didn't","doesn't","don't","hadn't","hasn't","haven't","isn't","mightn't",
            "mustn't","needn't","no","nor","not","shan't","shouldn't","wasn't","weren't","wouldn't"]
for i in not_stop:
    stops.remove(i)
```

In [124]:

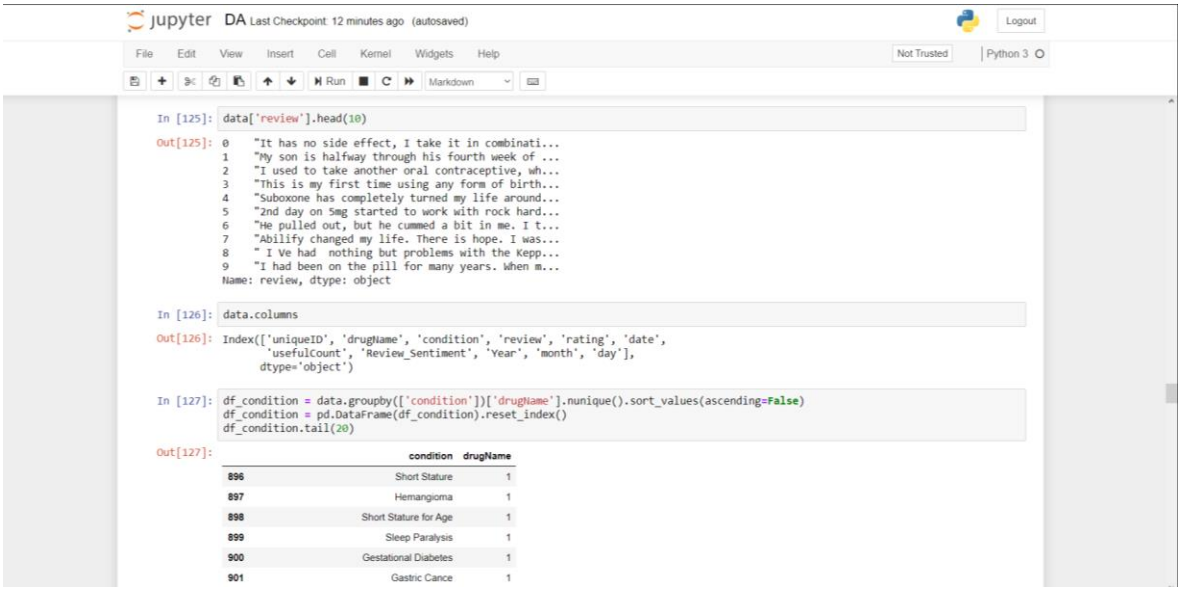
```python
from wordcloud import WordCloud, STOPWORDS

def plot_wordcloud(text, max_words = 200, figure_size=(10, 10), title = None, title_size = 30, image_color = None):

    stopwords = set(STOPWORDS)
    more_stopwords = {'one', 'br', 'Po', 'th', 'sayi', 'fo', 'Unknown'}
    stopwords = stopwords.union(more_stopwords)

    wordcloud = WordCloud(background_color='white',
                    stopwords = stopwords,
                    max_words = max_words,
                    random_state = 42,
                    width = 1200,
                    height = 800,)
    wordcloud.generate(str(text))

    plt.figure(figsize = figure_size)
    if image_color:
        image_colors = ImageColorGenerator(mask);
        plt.imshow(wordcloud.recolor(color_func=image_colors), interpolation="bilinear");
        plt.title(title, fontdict={'size': title_size,
                                  'verticalalignment': 'bottom'})
    else:
        plt.imshow(wordcloud);
        plt.title(title, fontdict={'size': title_size, 'color': 'black',
                                  'verticalalignment': 'bottom'})
    plt.axis('off');
    plt.tight_layout()

plot_wordcloud(stops, title="WordsCloud for StopWords")
```

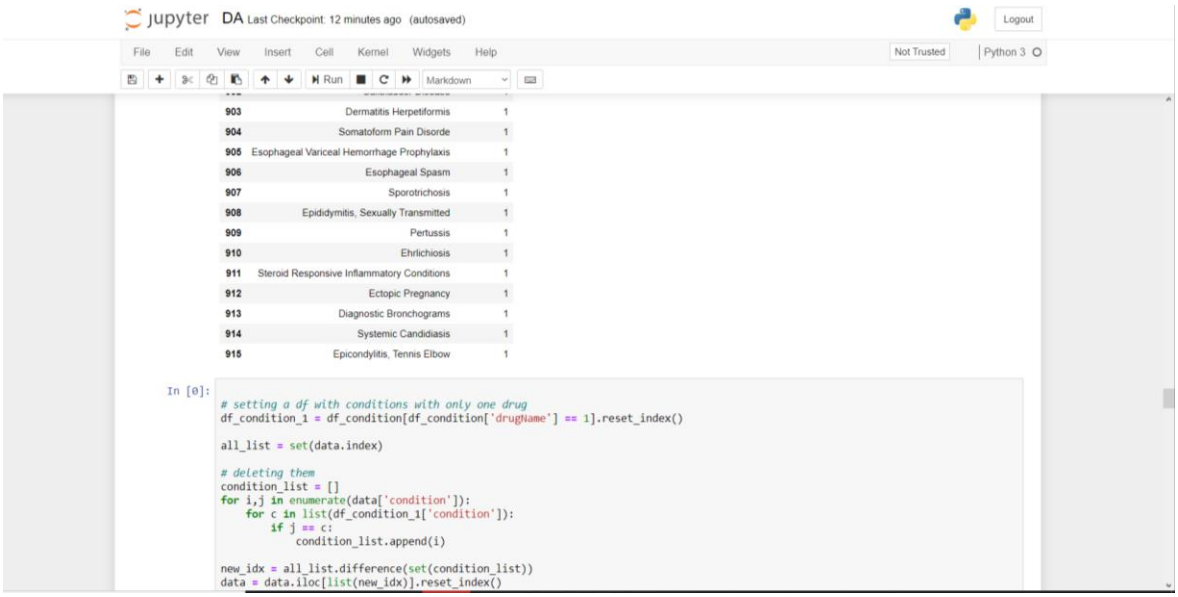


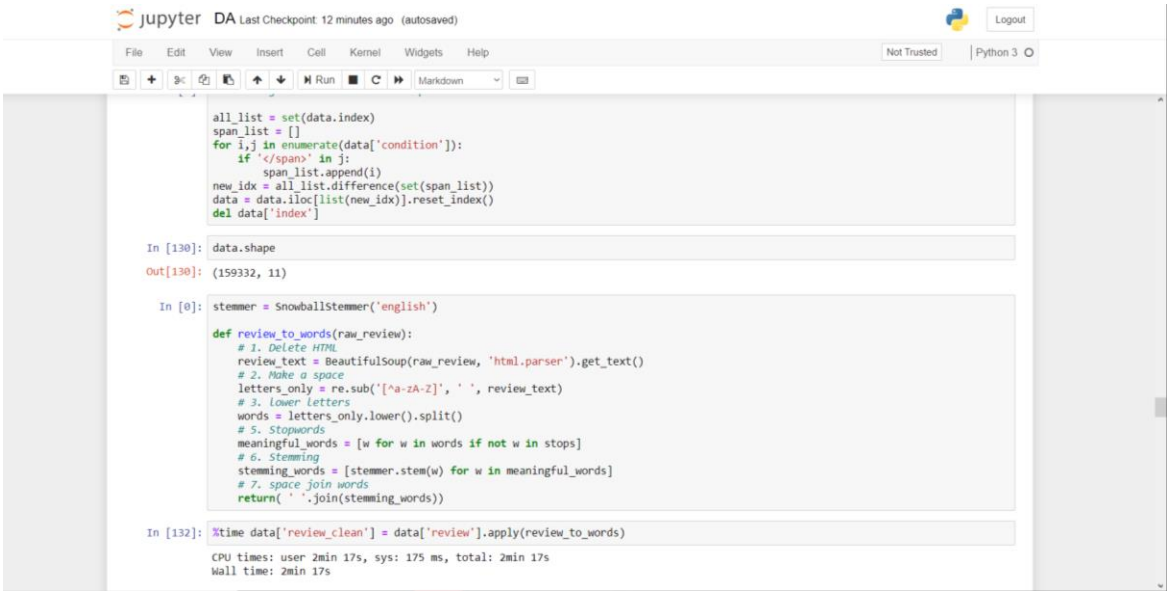
In [125]:

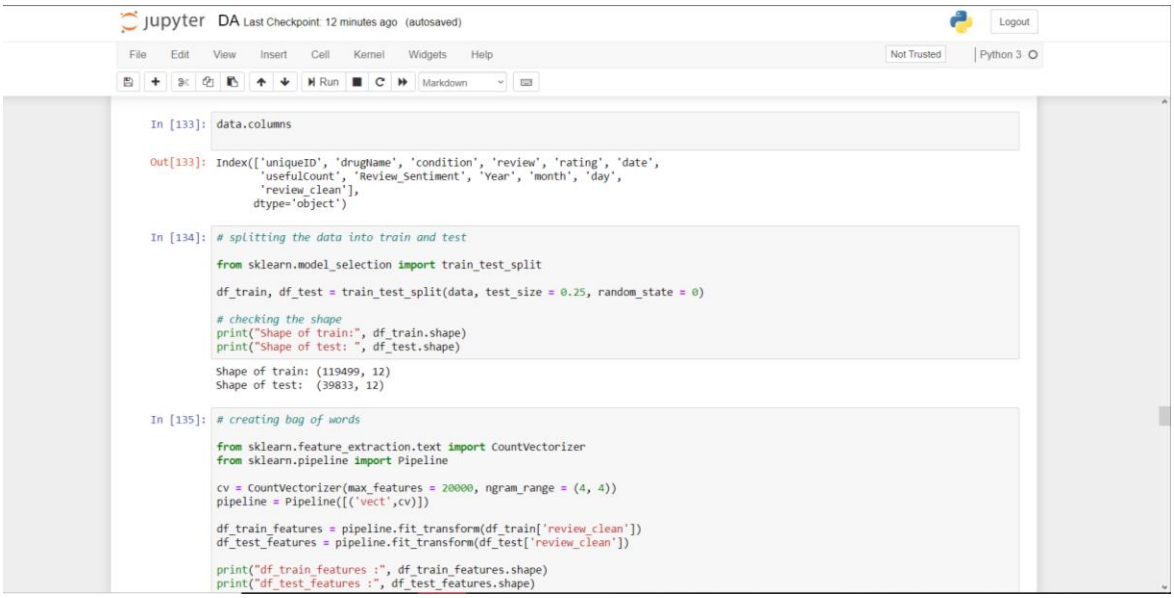
```python
data['review'].head(10)
```

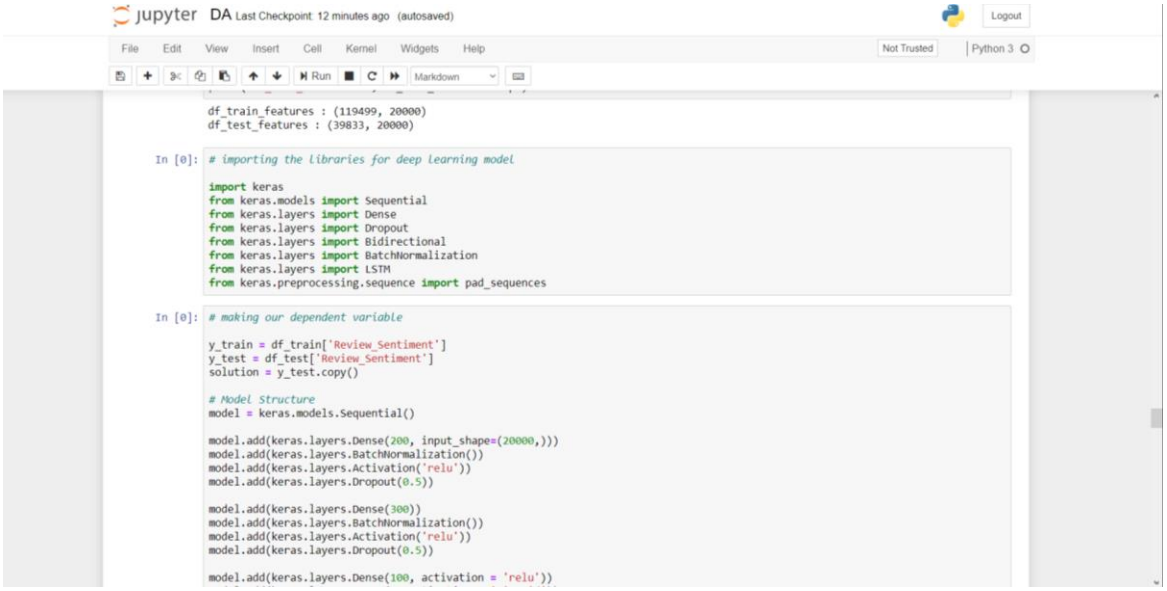
Out[125]: 0   "It has no side effect, I take it in combinati...

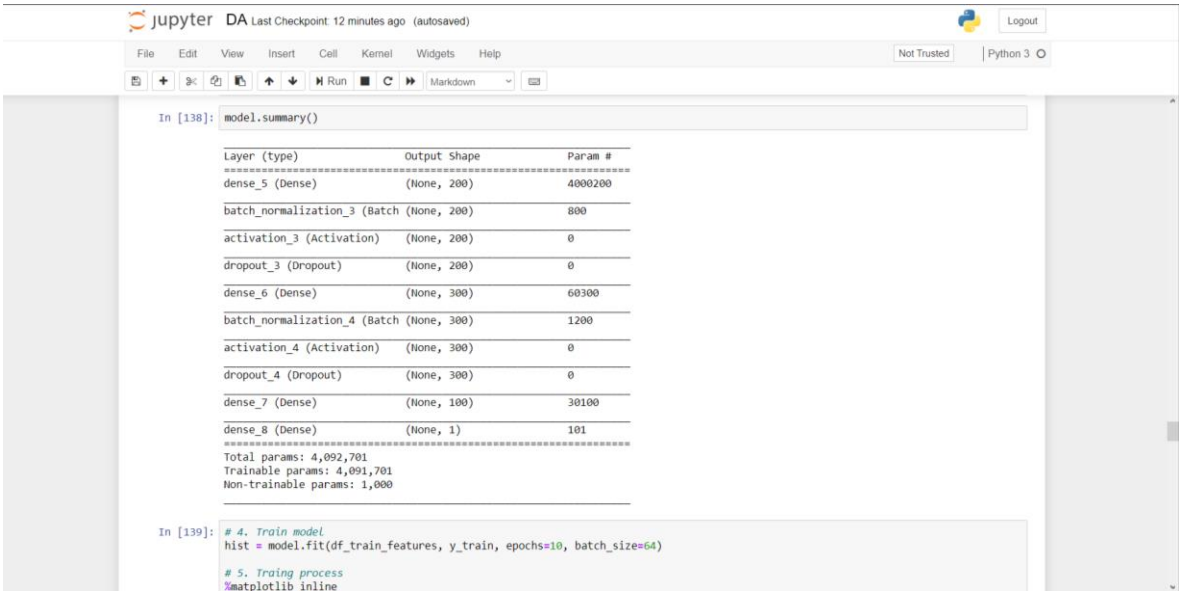

jupyter DA Last Checkpoint: 12 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
In [125]: data['review'].head(10)
Out[125]: 0 "It has no side effect, I take it in combinati...
1 "My son is halfway through his fourth week of ...
2 "I used to take another oral contraceptive, wh...
3 "This is my first time using any form of birth...
4 "Suboxone has completely turned my life around...
5 "2nd day on 5mg started to work with rock hard...
6 "He pulled out, but he cummed a bit in me. I t...
7 "Abilify changed my life. There is hope. I was...
8 " I Ve had nothing but problems with the Kepp...
9 "I had been on the pill for many years. When m...
Name: review, dtype: object
In [126]: data.columns
Out[126]: Index(['uniqueID', 'drugName', 'condition', 'review', 'rating', 'date',
'usefulCount', 'Review_Sentiment', 'Year', 'month', 'day'],
dtype='object')
In [127]: df_condition = data.groupby(['condition'])['drugName'].nunique().sort_values(ascending=False)
df_condition = pd.DataFrame(df_condition).reset_index()
df_condition.tail(20)
Out[127]:
condition drugName
896 Short Stature 1
897 Hemangioma 1
898 Short Stature for Age 1
899 Sleep Paralysis 1
900 Gestational Diabetes 1
901 Gastric Cance 1


jupyter DA Last Checkpoint: 12 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
903 Dermatitis Herpetiformis 1
904 Somatoform Pain Disorde 1
905 Esophageal Variceal Hemorrhage Prophylaxis 1
906 Esophageal Spasm 1
907 Sporotrichosis 1
908 Epididymitis, Sexually Transmitted 1
909 Pertussis 1
910 Ehrlichiosis 1
911 Steroid Responsive Inflammatory Conditions 1
912 Ectopic Pregnancy 1
913 Diagnostic Bronchograms 1
914 Systemic Candidiasis 1
915 Epicondylitis, Tennis Elbow 1
In [0]:
# setting a df with conditions with only one drug
df_condition_1 = df_condition[df_condition['drugName'] == 1].reset_index()
all_list = set(data.index)
# deleting them
condition_list = []
for i,j in enumerate(data['condition']):
for c in list(df_condition_1['condition']):
if j == c:
condition_list.append(i)
new_idx = all_list.difference(set(condition_list))
data = data.iloc[list(new_idx)].reset_index()


jupyter DA Last Checkpoint: 12 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
all_list = set(data.index)
span_list = []
for i,j in enumerate(data['condition']):
if '' in j:
span_list.append(i)
new_idx = all_list.difference(set(span_list))
data = data.iloc[list(new_idx)].reset_index()
del data['index']
In [130]: data.shape
Out[130]: (159332, 11)
In [0]: stemmer = SnowballStemmer('english')
def review_to_words(raw_review):
# 1. Delete HTML
review_text = BeautifulSoup(raw_review, 'html.parser').get_text()
# 2. Make a space
letters_only = re.sub('[^a-zA-Z]', ' ', review_text)
# 3. lower letters
words = letters_only.lower().split()
# 5. Stopwords
meaningful_words = [w for w in words if not w in stops]
# 6. Stemming
stemming_words = [stemmer.stem(w) for w in meaningful_words]
# 7. space join words
return(' '.join(stemming_words))
In [132]: %time data['review_clean'] = data['review'].apply(review_to_words)
CPU times: user 2min 17s, sys: 175 ms, total: 2min 17s
Wall time: 2min 17s


jupyter DA Last Checkpoint: 12 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
In [133]: data.columns
Out[133]: Index(['uniqueID', 'drugName', 'condition', 'review', 'rating', 'date',
'usefulCount', 'Review_Sentiment', 'Year', 'month', 'day',
'review_clean'],
dtype='object')
In [134]: # splitting the data into train and test
from sklearn.model_selection import train_test_split
df_train, df_test = train_test_split(data, test_size = 0.25, random_state = 0)
# checking the shape
print("Shape of train:", df_train.shape)
print("Shape of test: ", df_test.shape)
Shape of train: (119499, 12)
Shape of test: (39833, 12)
In [135]: # creating bag of words
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.pipeline import Pipeline
cv = CountVectorizer(max_features = 20000, ngram_range = (4, 4))
pipeline = Pipeline([('vect',cv)])
df_train_features = pipeline.fit_transform(df_train['review_clean'])
df_test_features = pipeline.fit_transform(df_test['review_clean'])
print("df_train_features :", df_train_features.shape)
print("df_test_features :", df_test_features.shape)


jupyter DA Last Checkpoint: 12 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
df_train_features : (119499, 20000)
df_test_features : (39833, 20000)
In [0]:
# importing the libraries for deep learning model
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Bidirectional
from keras.layers import BatchNormalization
from keras.layers import LSTM
from keras.preprocessing.sequence import pad_sequences
In [0]:
# making our dependent variable
y_train = df_train['Review_Sentiment']
y_test = df_test['Review_Sentiment']
solution = y_test.copy()
# Model Structure
model = keras.models.Sequential()
model.add(keras.layers.Dense(200, input_shape=(20000,)))
model.add(keras.layers.BatchNormalization())
model.add(keras.layers.Activation('relu'))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Dense(300))
model.add(keras.layers.BatchNormalization())
model.add(keras.layers.Activation('relu'))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Dense(100, activation = 'relu'))


jupyter DA Last Checkpoint: 12 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
In [138]:
model.summary()
Layer (type) Output Shape Param #
dense_5 (Dense) (None, 200) 4000200
batch_normalization_3 (Batch (None, 200) 800
activation_3 (Activation) (None, 200) 0
dropout_3 (Dropout) (None, 200) 0
dense_6 (Dense) (None, 300) 60300
batch_normalization_4 (Batch (None, 300) 1200
activation_4 (Activation) (None, 300) 0
dropout_4 (Dropout) (None, 300) 0
dense_7 (Dense) (None, 100) 30100
dense_8 (Dense) (None, 1) 101
Total params: 4,092,701
Trainable params: 4,091,701
Non-trainable params: 1,000
In [139]:
# 4. Train model
hist = model.fit(df_train_features, y_train, epochs=10, batch_size=64)
# 5. Traing process
%matplotlib inline

jupyter DA Last Checkpoint: 12 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Markdown

In [139]:
```
# 4. Train model
hist = model.fit(df_train_features, y_train, epochs=10, batch_size=64)

# 5. Traing process
%matplotlib inline
import matplotlib.pyplot as plt

fig, loss_ax = plt.subplots()

acc_ax = loss_ax.twinx()

loss_ax.set_ylim([0.0, 1.0])
acc_ax.set_ylim([0.0, 1.0])

loss_ax.plot(hist.history['loss'], 'y', label='train loss')
acc_ax.plot(hist.history['acc'], 'b', label='train acc')

loss_ax.set_xlabel('epoch')
loss_ax.set_ylabel('loss')
acc_ax.set_ylabel('accuray')

loss_ax.legend(loc='upper left')
acc_ax.legend(loc='lower left')

plt.show()
```

```
Epoch 1/10
119499/119499 [==============================] - 43s 361us/step - loss: 0.5444 - acc: 0.7567
Epoch 2/10
119499/119499 [==============================] - 42s 355us/step - loss: 0.4638 - acc: 0.7912
Epoch 3/10
119499/119499 [==============================] - 42s 354us/step - loss: 0.4250 - acc: 0.8085
Epoch 4/10
119499/119499 [==============================] - 40s 331us/step - loss: 0.4054 - acc: 0.8163
Epoch 5/10
119499/119499 [==============================] - 36s 304us/step - loss: 0.3927 - acc: 0.8215
```

jupyter DA Last Checkpoint: 12 minutes ago (autosaved)

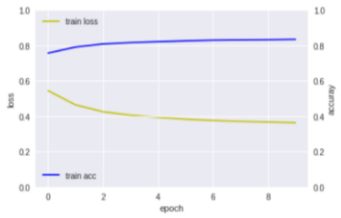
Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

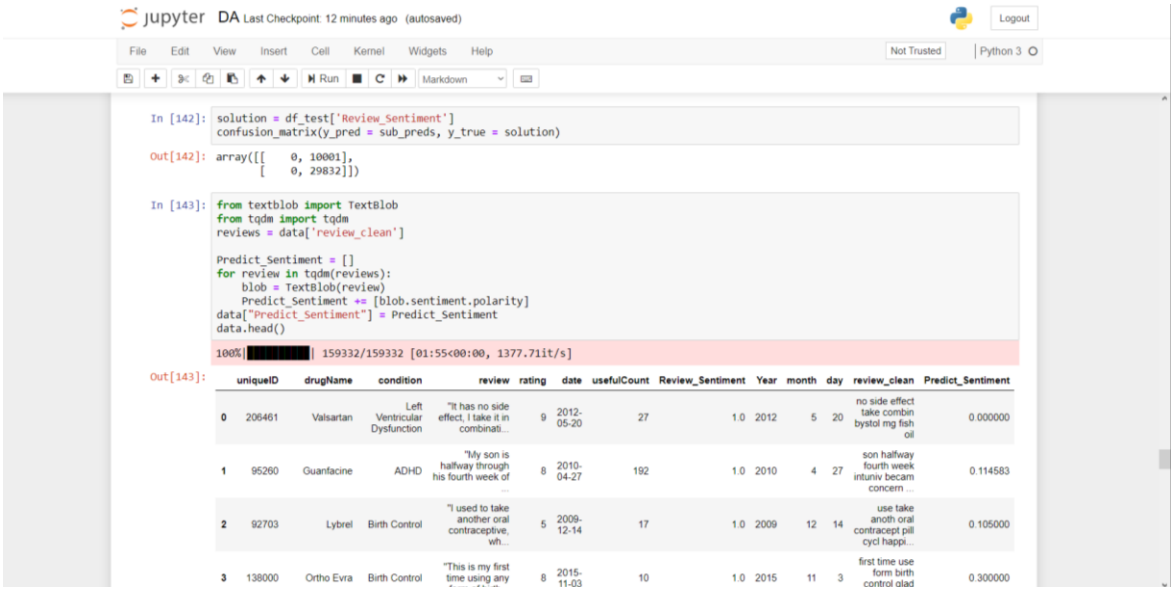
Run Markdown

```
119499/119499 [==============================] - 37s 312us/step - loss: 0.3671 - acc: 0.8325
Epoch 10/10
119499/119499 [==============================] - 38s 315us/step - loss: 0.3629 - acc: 0.8347
```



In [0]:
```
sub_preds_deep = model.predict(df_test_features,batch_size=32)
```

In [141]:
```
from sklearn.metrics import roc_auc_score, precision_recall_curve, roc_curve, average_precision_score
from sklearn.model_selection import KFold
from lightgbm import LGBMClassifier
from sklearn.metrics import confusion_matrix

#folds = KFold(n_splits=5, shuffle=True, random_state=546789)
target = df_train['Review_Sentiment']
feats = ['usefulCount']

sub_preds = np.zeros(df_test.shape[0])

trn_x, val_x, trn_y, val_y = train_test_split(df_train[feats], target, test_size=0.2, random_state=42)
```



In [142]:
```
solution = df_test['Review_Sentiment']
confusion_matrix(y_pred = sub_preds, y_true = solution)
```

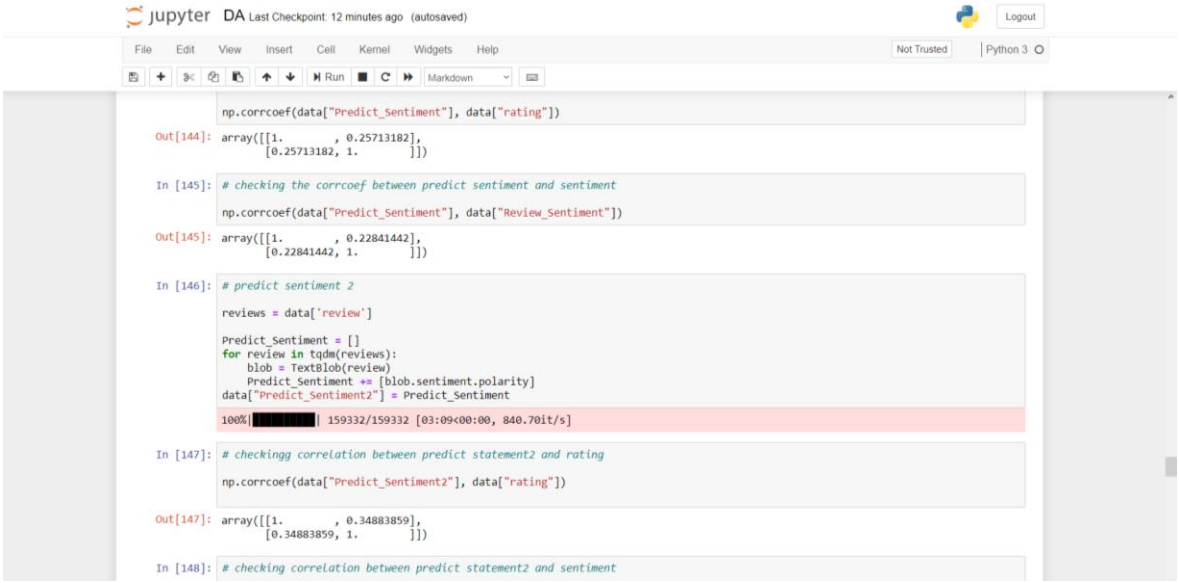
Out[142]:
```
array([[    0, 10001],
       [    0, 29832]])
```

In [143]:
```
from textblob import TextBlob
from tqdm import tqdm
reviews = data['review_clean']

Predict_Sentiment = []
for review in tqdm(reviews):
    blob = TextBlob(review)
    Predict_Sentiment += [blob.sentiment.polarity]
data["Predict_Sentiment"] = Predict_Sentiment
data.head()
```

```
100%|██████████| 159332/159332 [01:55<00:00, 1377.71it/s]
```

Out[143]:

	uniqueID	drugName	condition	review	rating	date	usefulCount	Review_Sentiment	Year	month	day	review_clean	Predict_Sentiment
0	206461	Valsartan	Left Ventricular Dysfunction	"It has no side effect, I take it in combinati...	9	2012-05-20	27	1.0	2012	5	20	no side effect take combin bystol mg fish oil	0.000000
1	95260	Guanfacine	ADHD	"My son is halfway through his fourth week of ...	8	2010-04-27	192	1.0	2010	4	27	son halfway fourth week intuniv becam concern ...	0.114583
2	92703	Lybrel	Birth Control	"I used to take another oral contraceptive, wh...	5	2009-12-14	17	1.0	2009	12	14	use take anoth oral contracept pill cycl happi...	0.105000
3	138000	Ortho Evra	Birth Control	"This is my first time using any	8	2015-11-03	10	1.0	2015	11	3	first time use form birth control glad	0.300000




```
np.corrcoef(data["Predict_Sentiment"], data["rating"])
```

Out[144]:
```
array([[1.        , 0.25713182],
       [0.25713182, 1.        ]])
```

In [145]:
```
# checking the corrcoef between predict sentiment and sentiment

np.corrcoef(data["Predict_Sentiment"], data["Review_Sentiment"])
```

Out[145]:
```
array([[1.        , 0.22841442],
       [0.22841442, 1.        ]])
```

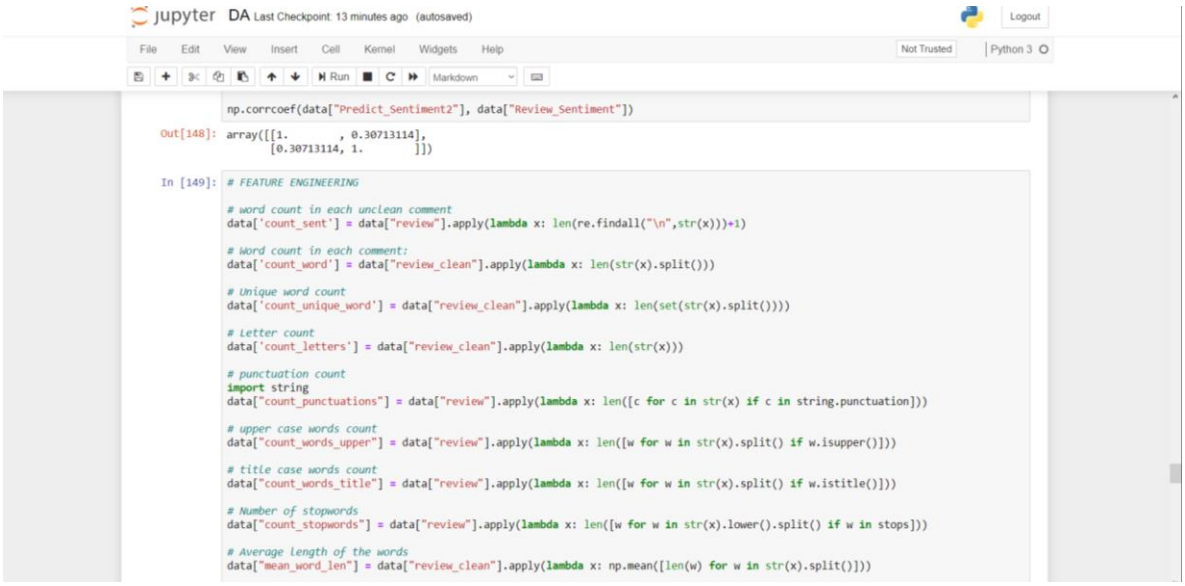
In [146]:
```
# predict sentiment 2

reviews = data['review']

Predict_Sentiment = []
for review in tqdm(reviews):
    blob = TextBlob(review)
    Predict_Sentiment += [blob.sentiment.polarity]
data["Predict_Sentiment2"] = Predict_Sentiment
```

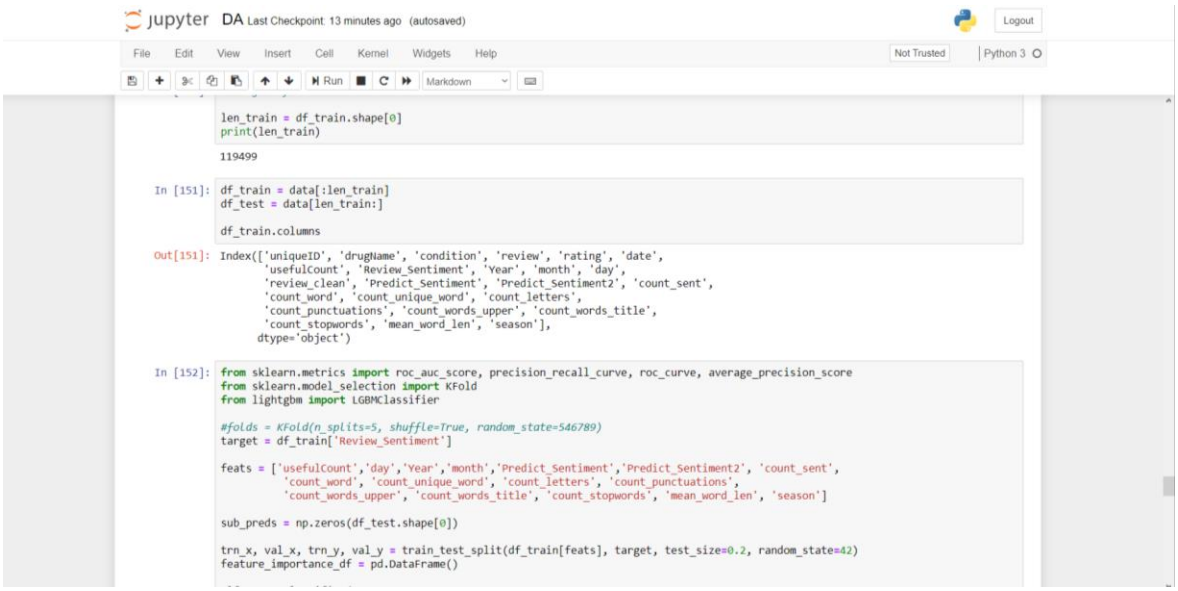
```
100%|██████████| 159332/159332 [03:09<00:00, 840.70it/s]
```

In [147]:
```
# checkingg correlation between predict statement2 and rating

np.corrcoef(data["Predict_Sentiment2"], data["rating"])
```

Out[147]:
```
array([[1.        , 0.34883859],
       [0.34883859, 1.        ]])
```

In [148]:
```
# checking correlation between predict statement2 and sentiment
```


jupyter DA Last Checkpoint: 13 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
np.corrcoef(data["Predict_Sentiment2"], data["Review_Sentiment"])
Out[148]: array([[1. , 0.30713114],
[0.30713114, 1.]])
In [149]:
FEATURE ENGINEERING
word count in each unclean comment
data['count_sent'] = data["review"].apply(lambda x: len(re.findall("\n",str(x)))+1)
Word count in each comment:
data['count_word'] = data["review_clean"].apply(lambda x: len(str(x).split()))
Unique word count
data['count_unique_word'] = data["review_clean"].apply(lambda x: len(set(str(x).split())))
Letter count
data['count_letters'] = data["review_clean"].apply(lambda x: len(str(x)))
# punctuation count
import string
data["count_punctuations"] = data["review"].apply(lambda x: len([c for c in str(x) if c in string.punctuation]))
upper case words count
data["count_words_upper"] = data["review"].apply(lambda x: len([w for w in str(x).split() if w.isupper()]))
title case words count
data["count_words_title"] = data["review"].apply(lambda x: len([w for w in str(x).split() if w.istitle()]))
Number of stopwords
data["count_stopwords"] = data["review"].apply(lambda x: len([w for w in str(x).lower().split() if w in stops]))
Average length of the words
data["mean_word_len"] = data["review_clean"].apply(lambda x: np.mean([len(w) for w in str(x).split()]))


jupyter DA Last Checkpoint: 13 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
len_train = df_train.shape[0]
print(len_train)
119499
In [151]:
df_train = data[:len_train]
df_test = data[len_train:]
df_train.columns
Out[151]: Index(['uniqueID', 'drugName', 'condition', 'review', 'rating', 'date',
'usefulCount', 'Review_Sentiment', 'Year', 'month', 'day',
'review_clean', 'Predict_Sentiment', 'Predict_Sentiment2', 'count_sent',
'count_word', 'count_unique_word', 'count_letters',
'count_punctuations', 'count_words_upper', 'count_words_title',
'count_stopwords', 'mean_word_len', 'season'],
dtype='object')
In [152]:
from sklearn.metrics import roc_auc_score, precision_recall_curve, roc_curve, average_precision_score
from sklearn.model_selection import KFold
from lightgbm import LGBMClassifier
#folds = KFold(n_splits=5, shuffle=True, random_state=546789)
target = df_train['Review_Sentiment']
feats = ['usefulCount','day','Year','month','Predict_Sentiment','Predict_Sentiment2', 'count_sent',
'count_word', 'count_unique_word', 'count_letters', 'count_punctuations',
'count_words_upper', 'count_words_title', 'count_stopwords', 'mean_word_len', 'season']
sub_preds = np.zeros(df_test.shape[0])
trn_x, val_x, trn_y, val_y = train_test_split(df_train[feats], target, test_size=0.2, random_state=42)
feature_importance_df = pd.DataFrame()

jupyter DA Last Checkpoint: 13 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Markdown

In [152]:
```
from sklearn.metrics import roc_auc_score, precision_recall_curve, roc_curve, average_precision_score
from sklearn.model_selection import KFold
from lightgbm import LGBMClassifier

#folds = KFold(n_splits=5, shuffle=True, random_state=546789)
target = df_train['Review_Sentiment']

feats = ['usefulCount','day','Year','month','Predict_Sentiment','Predict_Sentiment2', 'count_sent',
         'count_word', 'count_unique_word', 'count_letters', 'count_punctuations',
         'count_words_upper', 'count_words_title', 'count_stopwords', 'mean_word_len', 'season']

sub_preds = np.zeros(df_test.shape[0])

trn_x, val_x, trn_y, val_y = train_test_split(df_train[feats], target, test_size=0.2, random_state=42)
feature_importance_df = pd.DataFrame()

clf = LGBMClassifier(
        n_estimators=10000,
        learning_rate=0.10,
        num_leaves=30,
        #colsample_bytree=.9,
        subsample=.9,
        max_depth=7,
        reg_alpha=.1,
        reg_lambda=.1,
        min_split_gain=.01,
        min_child_weight=2,
        silent=-1,
        verbose=-1,
        )
clf.fit(trn_x, trn_y,
        eval_set= [(trn_x, trn_y), (val_x, val_y)],
        verbose=100, early_stopping_rounds=100  #30
    )
```

jupyter DA Last Checkpoint: 13 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

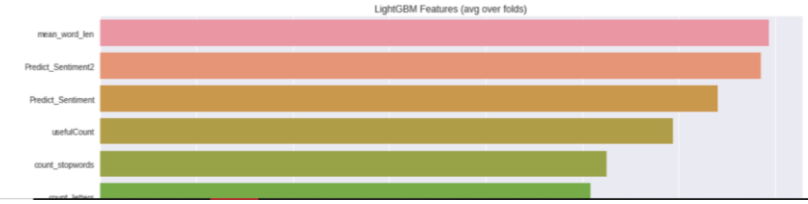
Run Markdown

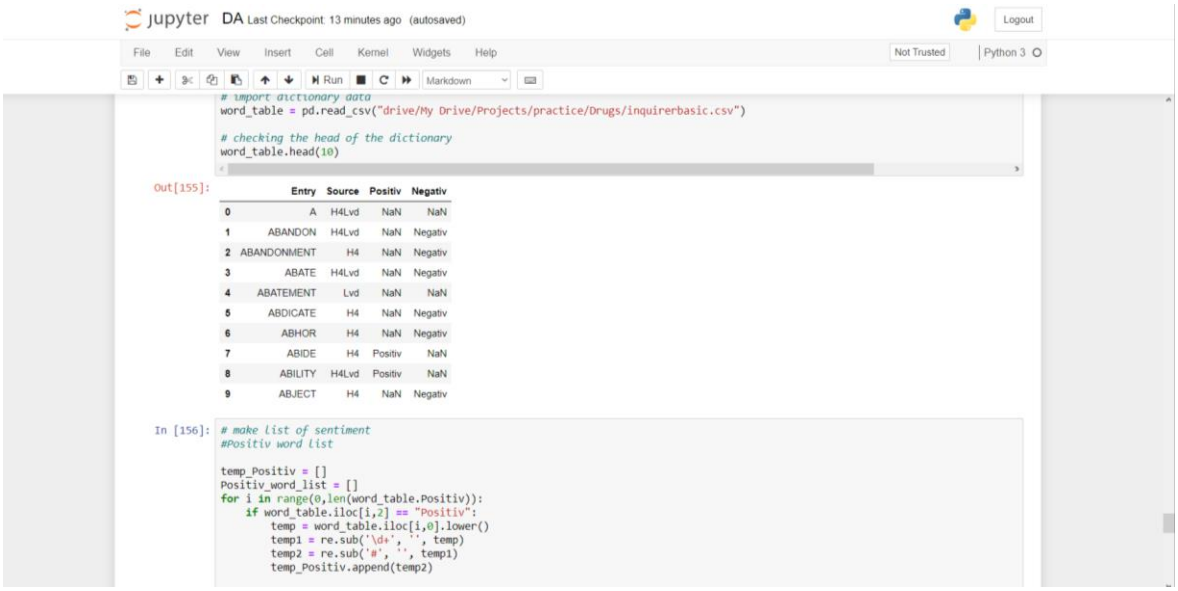
In [153]:
```
solution = df_test['Review_Sentiment']
confusion_matrix(y_pred=sub_preds, y_true=solution)
```

Out[153]:
```
array([[ 5408,  4383],
       [ 1687, 28355]])
```

In [154]:
```
cols = feature_importance_df[["feature", "importance"]].groupby("feature").mean().sort_values(
    by="importance", ascending = False)[:50].index

best_features = feature_importance_df.loc[feature_importance_df.feature.isin(cols)]

plt.figure(figsize=(14,10))
sns.barplot(x="importance", y="feature", data = best_features.sort_values(by="importance", ascending=False))
plt.title('LightGBM Features (avg over folds)')
plt.tight_layout()
plt.savefig('lgbm_importances.png')
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:1428: FutureWarning: remove_na is deprecated and is a private fun
ction. Do not use.
  stat_data = remove_na(group_data)
```




jupyter DA Last Checkpoint: 13 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
# import dictionary data
word_table = pd.read_csv("drive/My Drive/Projects/practice/Drugs/inquirerbasic.csv")
# checking the head of the dictionary
word_table.head(10)
Out[155]:
Entry Source Positiv Negativ
0 A H4Lvd NaN NaN
1 ABANDON H4Lvd NaN Negativ
2 ABANDONMENT H4 NaN Negativ
3 ABATE H4Lvd NaN Negativ
4 ABATEMENT Lvd NaN NaN
5 ABDICATE H4 NaN Negativ
6 ABHOR H4 NaN Negativ
7 ABIDE H4 Positiv NaN
8 ABILITY H4Lvd Positiv NaN
9 ABJECT H4 NaN Negativ
In [156]:
make list of sentiment
#Positiv word list
temp_Positiv = []
Positiv_word_list = []
for i in range(0,len(word_table.Positiv)):
if word_table.iloc[i,2] == "Positiv":
temp = word_table.iloc[i,0].lower()
temp1 = re.sub('\d+', '', temp)
temp2 = re.sub('#', '', temp1)
temp_Positiv.append(temp2)

Conclusion

Health information needs are also changing the information seeking behavior and can be observed around the globe. Challenges faced by many people are looking online for health information regarding diseases, diagnoses and different treatments. If a recommendation system can be made for doctors and medicine while using review mining will save a lot of time. In this type of system, the user face problem in understanding the heterogeneous medical vocabulary as the users are laymen. User is confused because a large amount of medical information on different mediums are available. By Using this project you would be able to find the solution of the disease or problem upto a minor extent as the 100% surety of the medicine can be given with only through Diagnosis with a Doctor. So this might be helpful till without a Doctor diagnosis. As the Drugs are recommended only through the reviews that are can be given by the people from the data sets downloaded from the Kaggle. To keep on updating the reviews with the change in Time the maximum accuracy achieved through by implementing on the web based platform or mobile apps.

References

- Cotton, J.L. and Tuttle, J.M., 1986. “Employee turnover: A metaanalysis and review with implications for research” Academy of management review, pp.55-70.

- Liu, D., Mitchell, T.R., Lee, T.W., Holtom, B.C. and Hinkin, T.R., 2012. “When employees are out of step with coworkers: How job satisfaction trajectory and dispersion influence individual-and unit-level voluntary turnover”. Academy of Management Journal, pp.1360-1380.

- Heckert, T.M. and Farabee, A.M., 2006. “Turnover intentions of the faculty at a teaching-focused university”. Psychological reports, pp.39-45.

- Rish, Irina, “An empirical study of the naive bayes classifier”,IJCAI Workshop on Empirical Methods in AI.