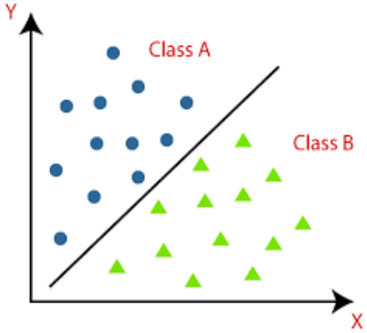
MACHINE LEARNING

CLASSIFICATION

CONTENTS

SENTIMENT ANALYSIS USING CLASSIFICATION

- The **Classification algorithm** is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data.
- In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups.

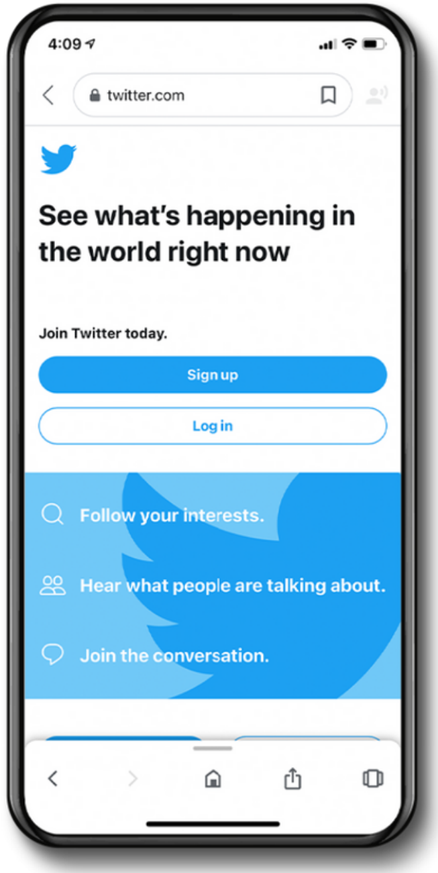


- **Sentiment analysis** refers to identifying as well as classifying the sentiments that are expressed in the text source.
- These data are useful in understanding the opinion of the people about a variety of topics.

TWITTER SENTIMENT ANALYSIS

In this project, we try to implement a Twitter sentiment analysis model that helps to overcome the challenges of identifying the sentiments of the tweets.
The dataset used contains 14640 tweets and 12 columns
The various columns present in the dataset are:

- **tweet_id-** unique twitter ID
- **airline_sentiment-**label whether the text is positive neg or neutral
- **airline_sentiment_confidence-**probability of airline sentiment
- **negative_confidence**-probability of tweet being negative
- **negativereason-** the keyword because of which the tweet is negative
- **airline-** airline name
- **name-** user name
- **retweet_count-**number of times a tweet is retweeted
- **text-**the actaual tweet
- **tweet_created-**date and time when the tweet was created
- **tweet_location-** place from where the tweet was created
- **user_timezone-** timezone in which the place falls



TWITTER SENTIMENT ANALYSIS USE CASE

Social Media Monitoring

Online reputation is one of the most precious assets for brands. A bad review on social media can be costly to a company if it’s not handled effectively and swiftly.

Customer Service

It has become increasingly important for customer service agents to be present on Twitter. They need to engage with customers and respond quickly to customer queries.

Market Research

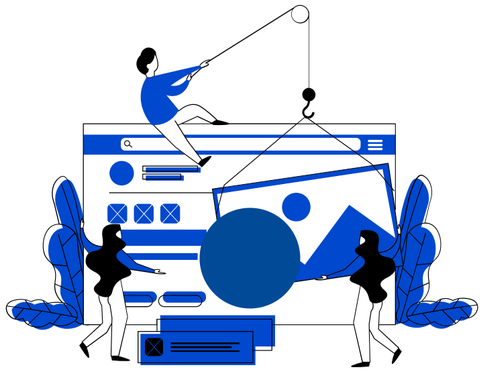
You can use this to track specific keywords and topics to detect customer trends and interests. Understanding what customers like, and how this changes over time is essential if you are planning to launch a new product.

Brand Monitoring

Whether you are launching a new feature on your platform, a site redesign, or a new marketing campaign, you may want to track customer reactions on Twitter.

CLASSIFICATION TYPES

We have used 2 classification techniques:



TEXT CLASSIFICATION

- Text classification is a machine learning technique that assigns a set of predefined categories to open-ended text. Text classifiers can be used to organize, structure, and categorize any kind of text – from documents, medical studies and files, and all over the web.
- **Algorithms used for Text Classification** - Naive Bayes Model, Logistic Regression, Support Vector Classifier.
- **Attribute of the dataset used to classify text** - text

NUMERIC CLASSIFICATION

- Numeric classification technique uses only numeric data to predict the output. The output can be a text, but the input has to be numeric.
- **Algorithms used for Numeric Classification** - k-Nearest Neighbor (KNN).
- **Attributes of the dataset used to classify the classes** - airline_sentiment_confidence, negativereason_confidence, retweet_count

Splitting Of training and testing dataset and vectorization

```
y=data['airline_sentiment']
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(
...     X, y, test_size=0.2, random_state=101)
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
tfidf = TfidfVectorizer(stop_words='english')
```

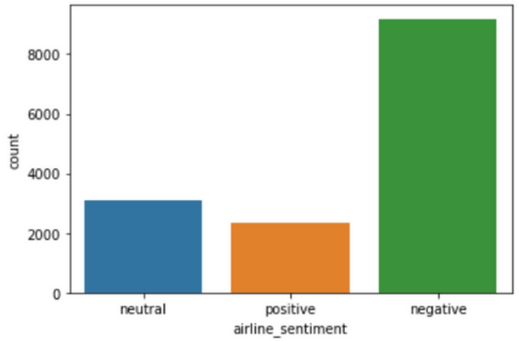
```
tfidf.fit(X_train)
```

```
TfidfVectorizer(stop_words='english')
```

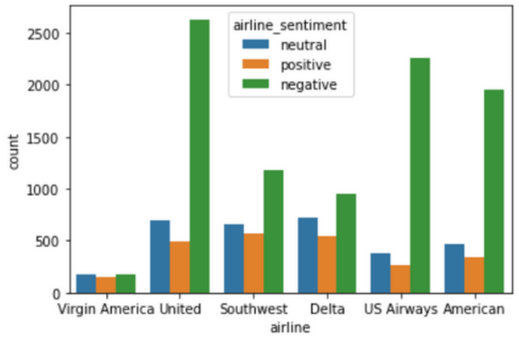
```
X_train_tfidf=tfidf.transform(X_train)
```

```
X_test_tfidf=tfidf.transform(X_test)
```

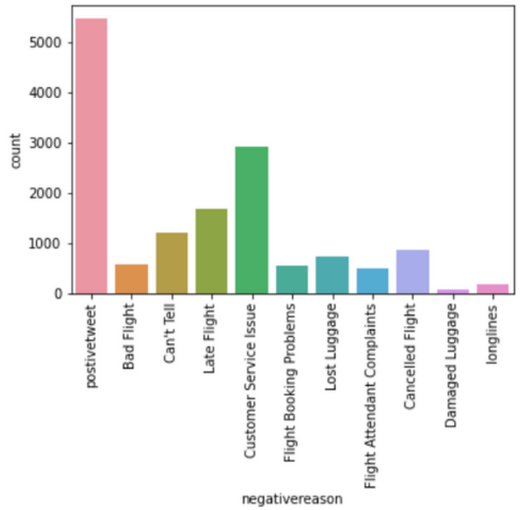
TWITTER DATASET ANALYSIS



The graph above represents the distribution of 'Neutral', 'Positive', and 'Negative' tweets of the dataset. 'Negative' class has the majority of data instances.



The graph above represents the distribution of 'Neutral', 'Positive' and 'Negative' tweets wrt location.
We observe a huge difference in Negative and Neutral/Positive tweets in most locations



The graph above represents the count of 'negative reason', i.e. the no. of times a negative feedback was given.

DRAWBACK OF THE DATASET

A huge drawback of this dataset is that the data is imbalanced. More than 65% of the tweets belong to the 'Negative' class, whereas the 'Positive' and 'Neutral' classes contain less instances of data. This might result in wrong prediction of the classes with less data.

Dateset used for Text Classification

```
data=td[['airline_sentiment','text']]
data
```

	airline_sentiment	text
0	neutral	@VirginAmerica What @dhepburn said.
1	positive	@VirginAmerica plus you've added commercials t...
2	neutral	@VirginAmerica I didn't today... Must mean I n...
3	negative	@VirginAmerica it's really aggressive to blast...
4	negative	@VirginAmerica and it's a really big bad thing...
...	...	...
14635	positive	@AmericanAir thank you we got on a different f...
14636	negative	@AmericanAir leaving over 20 minutes Late Flig...
14637	neutral	@AmericanAir Please bring American Airlines to...
14638	negative	@AmericanAir you have my money, you change my ...
14639	neutral	@AmericanAir we have 8 ppl so we need 2 know h...

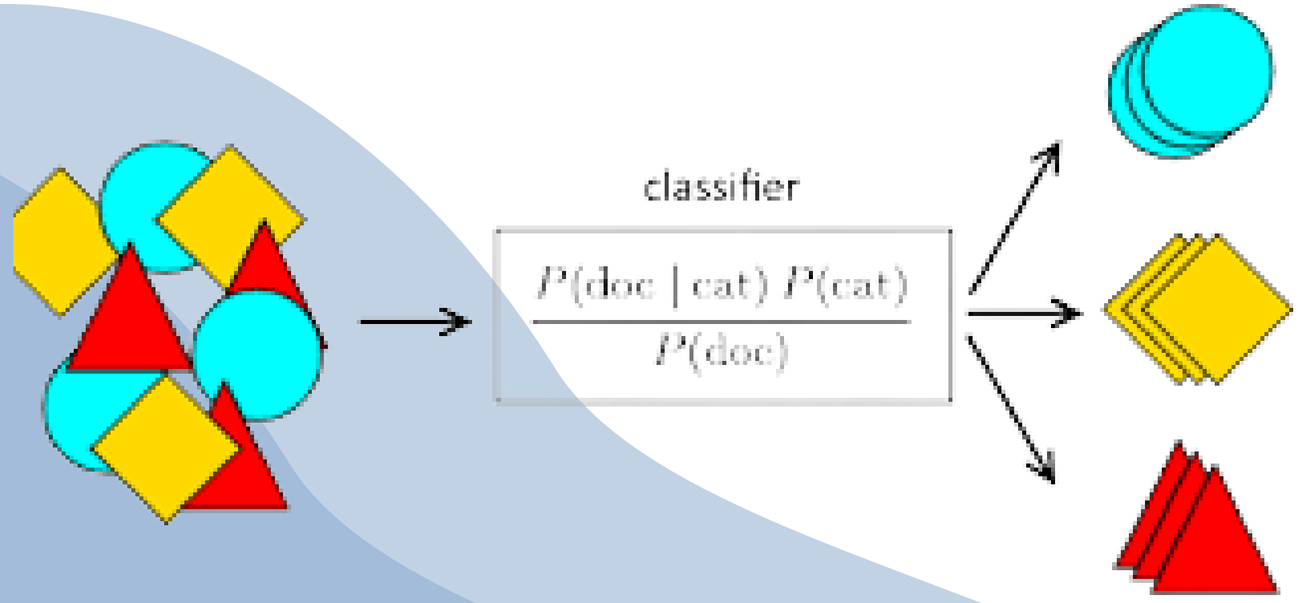
14640 rows × 2 columns

NAIVE BAYES ALGORITHM

- It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors.
- In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

MULTINOMIAL NAIVE BAYES



How does MULTINOMIAL NAIVE BAYES Work?

- Step-1: create a histogram for all the words occurring in the text. All these words are now the attributes and what we need to classify them into are the class labels.
- Step-2: Calculate prior probability for given class labels.
- Step-3: Find likelihood probability with each attribute for each class.
- Step-4: put these values in bayes formula explained in the next slide and calculate posterior probability.
- Step-5: see which class has a higher probability, given the input belongs to the higher probability class
- Step-6: Our model is ready.

BAYES FORMULA

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$

- P(A|B) is the posterior probability of class (A, target) given predictor (B, attributes).
- P(A) is the prior probability of class.
- P(B|A) is the likelihood which is the probability of predictor given class.
- P(B) is the prior probability of predictor.

NAIVE BAYES ALGORITHM

ADVANTAGES

- It is not only a simple approach but also a fast and accurate method for prediction.
- Naive Bayes has very low computation cost.
- It can efficiently work on a large dataset.
- It can be used with multiple class prediction problems.
- When the assumption of independence holds, a Naive Bayes classifier performs better compared to other models like logistic regression.

DISADVANTAGES

- The assumption of independent features. In practice, it is almost impossible that model will get a set of predictors which are entirely independent.
- If there is no training tuple of a particular class, this causes zero posterior probability. In this case, the model is unable to make predictions. This problem is known as Zero Probability/Frequency Problem.

Naive bayes code

```
from sklearn.metrics import plot_confusion_matrix,classification_report
```

```
def report(model):
    preds = model.predict(X_test_tfidf)
    print(classification_report(y_test,preds))
    plot_confusion_matrix(model,X_test_tfidf,y_test)
```

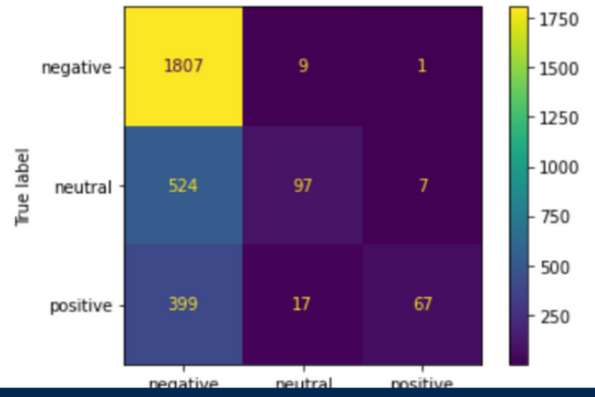
```
from sklearn.naive_bayes import MultinomialNB
nb=MultinomialNB()
nb.fit(X_train_tfidf,y_train)
```

```
MultinomialNB()
```

NAIVE BAYES CLASSIFIER

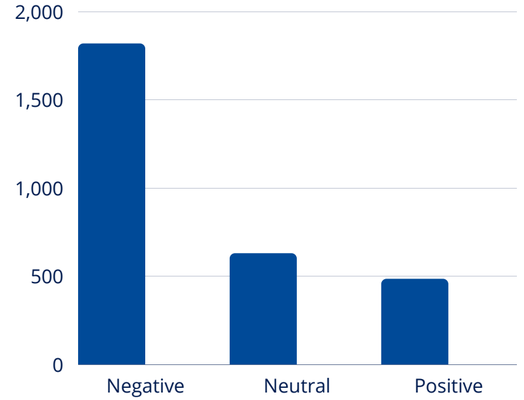
CLASSIFICATION REPORT

```
In [36]: print("NB MODEL")
         report(nb)

NB MODEL
                precision    recall  f1-score   support

    negative       0.66      0.99      0.79      1817
     neutral       0.79      0.15      0.26       628
    positive       0.89      0.14      0.24       483

    accuracy                           0.67      2928
   macro avg       0.78      0.43      0.43      2928
weighted avg       0.73      0.67      0.59      2928
```



ACCURACY

- The accuracy is 0.67 that means the model is 67% accurate.

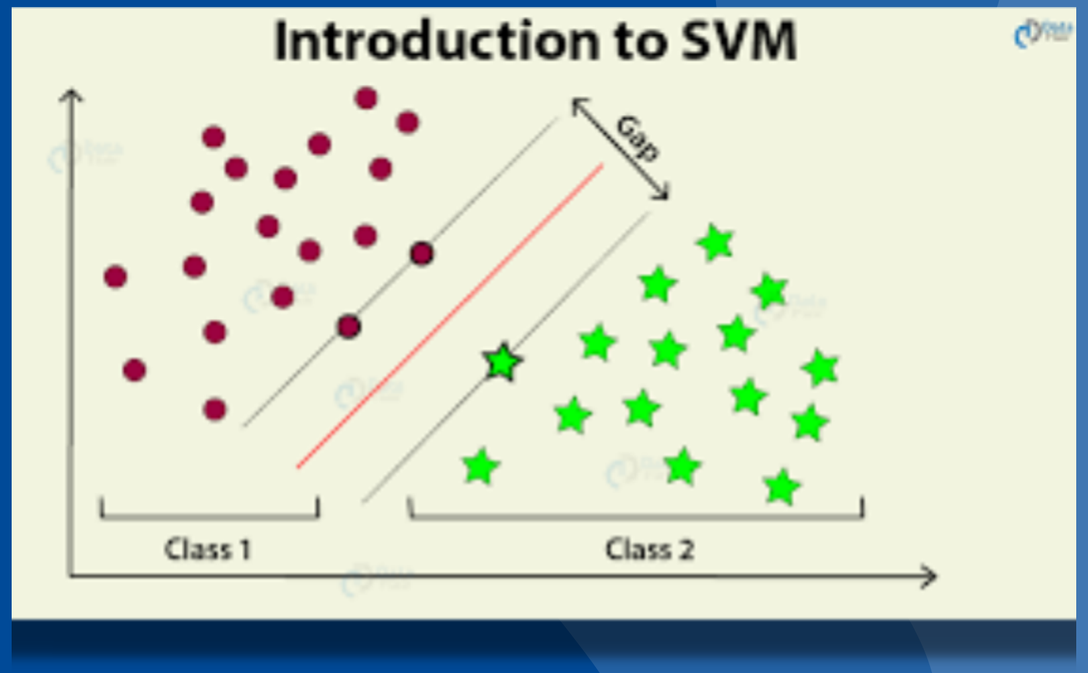


REPORT ANALYSIS

- **Negative** : We get precision(0.66) and recall(0.99) for this class.
- **Neutral** : The precision(0.79) and recall(0.15) for this class is not ideal.
- **Positive** : We get 0.89 precision for this class, which shows that 89% of the predicted outcomes for class 'Positive' were correct. But only 14% of the 'Positive' cases were correctly predicted. So the predictions were not upto mark.

LINEAR SUPPORT VECTOR CLASSIFICATION

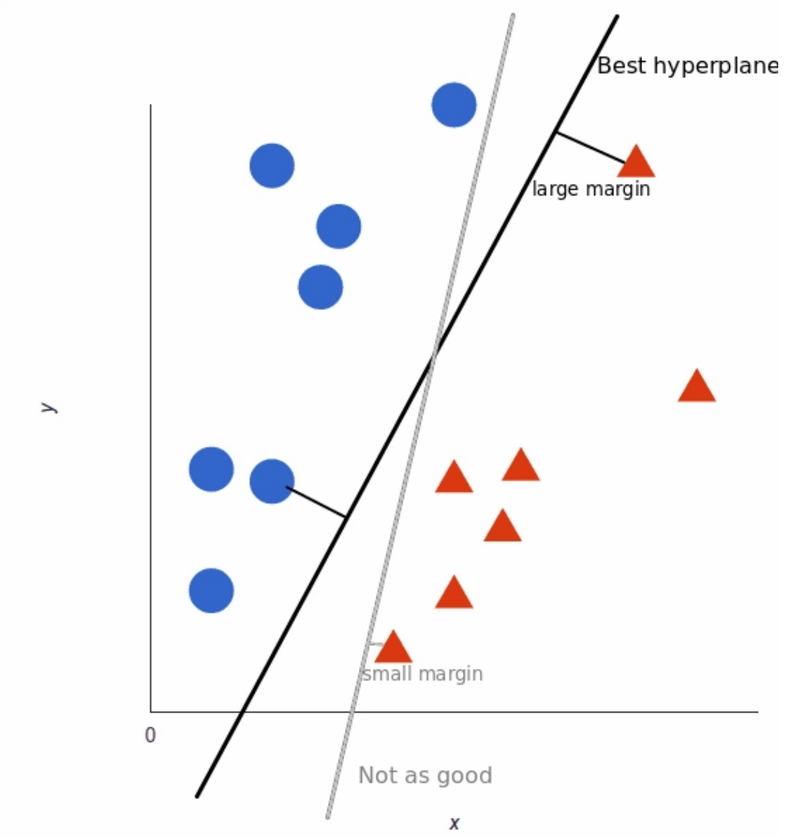
- A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems.
- After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.



SUPPORT VECTOR CLASSIFIERS

How does support vector classifier work?

- SVM draws a line or “hyperplane” that divides a space into two subspaces. One subspace contains vectors (tags) that belong to a group, and another subspace contains vectors that do not belong to that group.
- The optimal hyperplane is the one with the largest distance between each tag. In two dimensions it looks like the figure shown



SUPPORT VECTOR CLASSIFIERS

How to find the hyperplane ?

- There is no way we can separate these 2 classes without allowing for misclassification

- if we do this ,it fits too much too the blue dots but misclassifies only one blue dot. This lowers bias and becomes high variance data because if a new test point having features CLOSE to the blue point is put it may get classified into red points

- Therefore we use bias variance trade off where we increase the bias a little bit to decrease the variance to lead to better long term results on future data.

WE TRY TO FIND A LINE TO SEPERATE BLUE AND RED DOTS ,BUT WE TOLERATE A FEW MISCLASSIFIED ONES THIS IS CALLED SOFT MARGIN.IT TRIES TO MAXIMIZE THE MARGIN AND MINIMIZE THE MISCLASSIFICATION

WE USE CROSS VALIDATION TO DETERMINE THE OPTIMAL SIZE OF THE MARGINS I.E WE TEST DIFFERENT LEVELS OF MISCLASSIFICATION ALLOWED TO FIND THE BEST OVERALL MODEL.

SUPPORT VECTOR CLASSIFIER

ADVANTAGES

- works on two classes are not easily separable by a hyperplane.
- Effective on datasets with multiple features, like financial or medical data.
- Effective in cases where number of features is greater than the number of data points.
- only observations that either lie on the margin or that violate the margin will affect the hyperplane, and hence the classifier obtained
- Greater robustness to individual observations

DISADVANTAGES

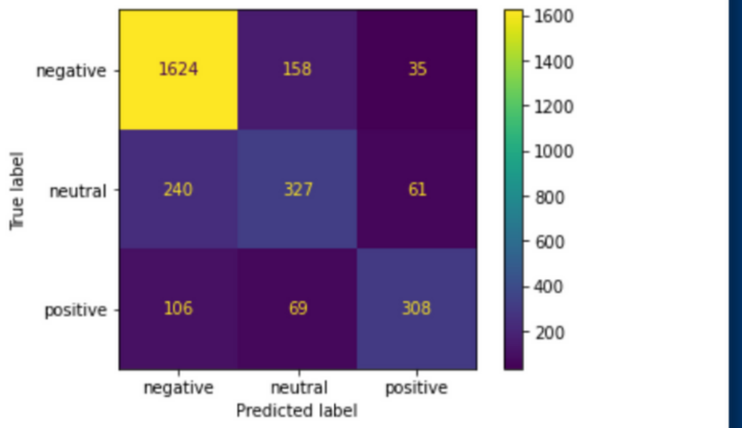
- If the number of features is a lot bigger than the number of data points, avoiding over-fitting when regularization term is crucial.
- Does not work with non-linear class boundaries
- Works better on small sample sets because of its high training time.

SUPPORT VECTOR CLASSIFIERS

CLASSIFICATION REPORT

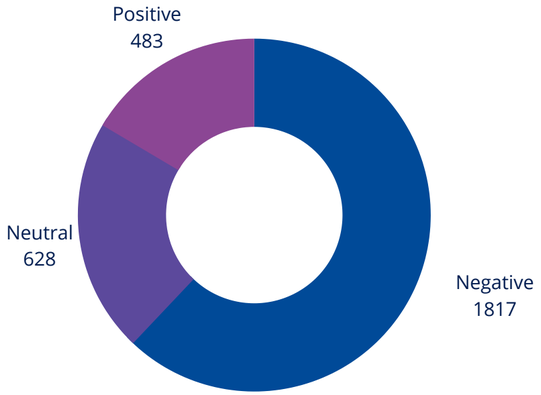
SVC

	precision	recall	f1-score	support
negative	0.82	0.89	0.86	1817
neutral	0.59	0.52	0.55	628
positive	0.76	0.64	0.69	483
accuracy			0.77	2928
macro avg	0.73	0.68	0.70	2928
weighted avg	0.76	0.77	0.77	2928



ACCURACY

- The accuracy is 0.77 that means the model is 77% accurate



REPORT ANALYSIS

- **Negative** : We get good precision(0.82) and recall(0.89) for this class.
- **Neutral** : The precision(0.59) and recall(0.52) for this class is not ideal.
- **Positive** : We get 0.76 precision for this class, which shows that 76% of the predicted outcomes for class 'Positive' were correct. But 64% of the 'Positive' cases were correctly predicted. So even though we got a good precision, the model is not an ideal choice as we get a low recall value.

TEXT CLASSIFICATION

LOGISTIC REGRESSION

- Logistic regression is a process of modeling the probability of a discrete outcome given an input variable.
- The most common logistic regression models a binary outcome; something that can take two values such as true/false, yes/no, and so on.
- Multinomial logistic regression can model scenarios where there are more than two possible discrete outcomes.

LOGISTIC REGRESSION

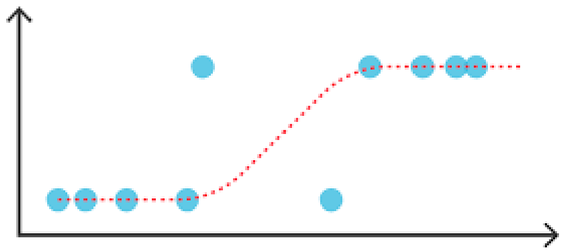
How does Logistic Regression Work?

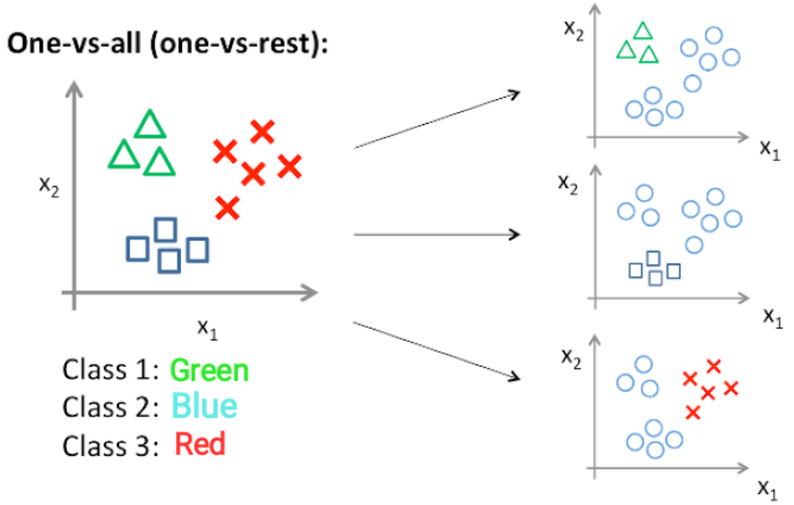
- Logistic regression does not require a linear relationship between inputs and output variables.
- Logistic regression uses an equation as the representation.
- In simple words, it tries to convert the independent variable into an expression of probability using the **Sigmoid Function**.

Why Sigmoid Function?

Sigmoid Function acts as an activation function in machine learning which is used to add non-linearity in a machine learning model,

$$f(x) = \frac{1}{1 + e^{-(x)}}$$





Multi-Class Classification in Logistic Regression

- Logistic regression, by default, is limited to two-class classification problems.
- Extensions like one-vs-rest can allow logistic regression to be used for multi-class classification problems, although they require that the classification problem first be transformed into multiple binary classification problems.

LOGISTIC REGRESSION

ADVANTAGES

- It is easier to implement, interpret, and very efficient to train.
- It makes no assumptions about distributions of classes in feature space.
- It can easily extend to multiple classes(multinomial regression) and a natural probabilistic view of class predictions.
- It is very fast at classifying unknown records.
- Good accuracy for many simple data sets and it performs well when the dataset is linearly separable.

DISADVANTAGES

- The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables.
- It can only be used to predict discrete functions. Hence, the dependent variable of Logistic Regression is bound to the discrete number set.
- Non-linear problems can’t be solved with logistic regression because it has a linear decision surface.
- Logistic Regression requires average or no multicollinearity between independent variables.

Logistic Regression Code

```
from sklearn.linear_model import LogisticRegression
log = LogisticRegression(max_iter=1000)
log.fit(X_train_tfidf,y_train)
```

```
LogisticRegression(max_iter=1000)
```

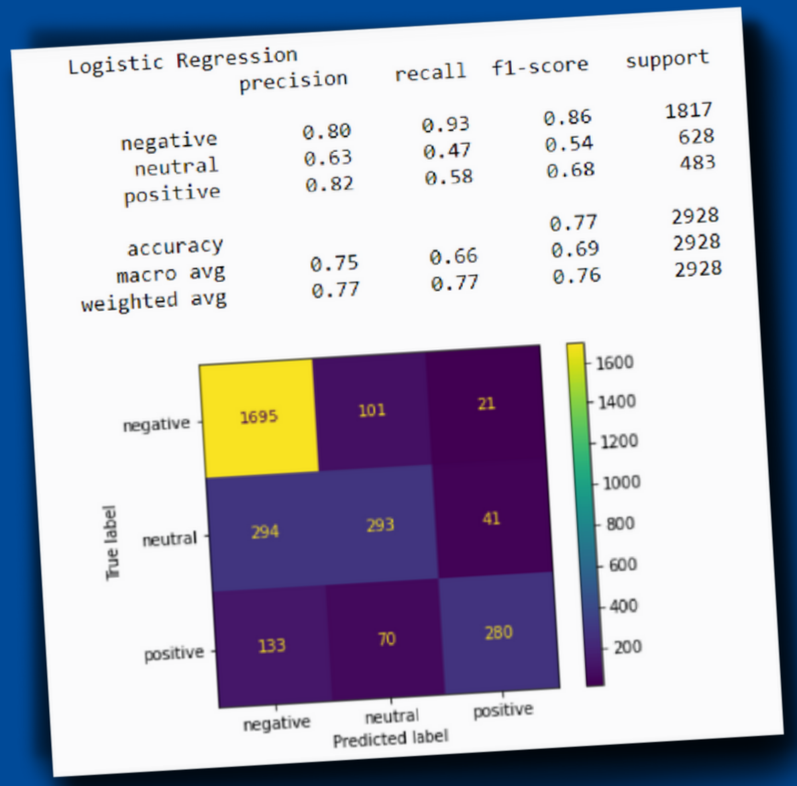
Linear SVC Code

```
from sklearn.svm import LinearSVC
svc = LinearSVC()
svc.fit(X_train_tfidf,y_train)
```

```
LinearSVC()
```

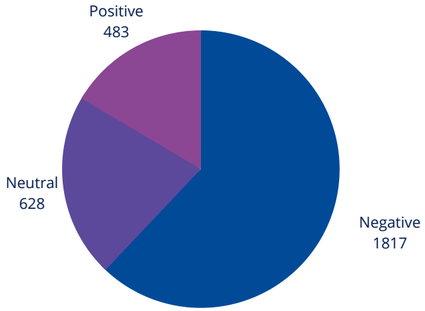
LOGISTIC REGRESSION

CLASSIFICATION REPORT



TYPE USED

- The target variable has three nominal categories: Negative, Neutral and Positive.
- Thus, Multi-Class Logistic Regression was used.



ACCURACY

- The Logistic Regression Algorithm gave 77% accuracy.

REPORT ANALYSIS

- **Negative** : We get good precision(0.80) and recall(0.93) for this class.
- **Neutral** : The precision(0.63) and recall(0.47) for this class is not ideal.
- **Positive** : We get 0.82 precision for this class, which shows that 82% of the predicted outcomes for class 'Positive' were correct. But only 58% of the 'Positive' cases were correctly predicted. So even though we got a good precision, the model is not an ideal choice as we get a low recall value.

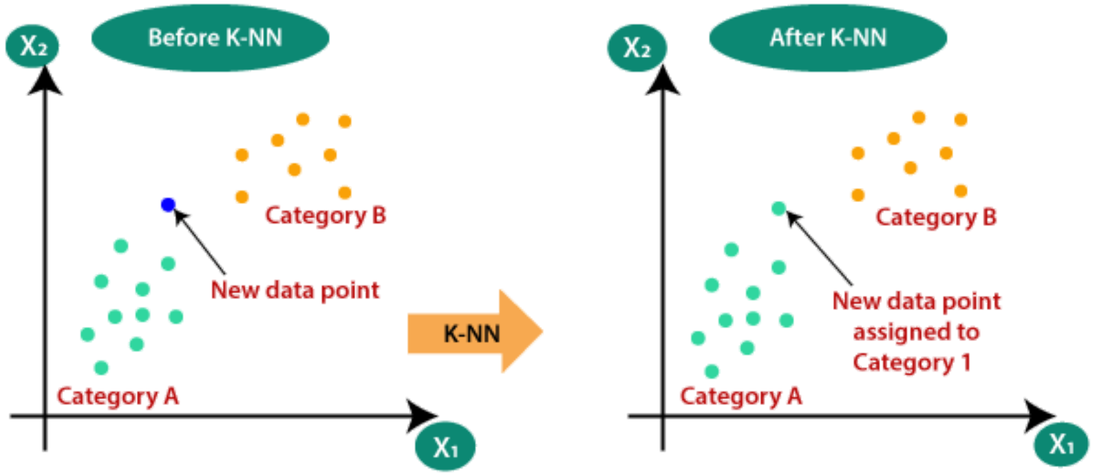
NUMERIC CLASSIFICATION

K NEAREST NEIGHBOR (KNN Classifier)

- K Nearest Neighbour is a simple algorithm that stores all the available cases and classifies the new data or case based on a similarity measure.
- It is mostly used to classify the data point based on how its neighbours are classified.

K-NN is also called a lazy learner because it doesn't learn a discriminative function from the training data but memorizes the training dataset instead. There is no training phase in K-NN.

K NEAREST NEIGHBOR



How does K-NN Work?

- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step-6: Our model is ready.

Selection of K Value

The number of neighbors is the core deciding factor. K is generally an odd number if the number of classes is 2

k = sqrt(N) , where N stands for the number of samples in the training dataset.

K NEAREST NEIGHBOR

ADVANTAGES

- K-NN is pretty intuitive and simple.
- K-NN has no assumptions.
- No Training Step. Thus new data entry would be tagged with majority class in the nearest neighbor.
- It constantly evolves.
- Very easy to implement for multi-class problem.

DISADVANTAGES

- KNN is a slow algorithm.
- It works well with small number of input variables but as the numbers of variables grow K-NN algorithm struggles to predict the output of new data point.
- K-NN needs homogeneous features.
- It doesn’t perform well on imbalanced data.

KNN Code

```
from sklearn.model_selection import train_test_split

X = df.loc[:,['airline_sentiment_confidence','negativereason_confidence','retweet_count']]
y = df.loc[:,'airline_sentiment']

# The data is split into the ratio 80:20
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2)
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

KNN Code for training and testing

```
# Define the model: Init K - NN
from sklearn.neighbors import KNeighborsClassifier

classifier = KNeighborsClassifier(n_neighbors=53 )
```

```
# Fit Model
classifier.fit(X_train, y_train )
```

```
KNeighborsClassifier(n_neighbors=53)
```

```
y_pred = classifier.predict(X_test)
```

```
from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(y_test,y_pred))
print(confusion_matrix(y_test,y_pred))
```

N=53

K NEAREST NEIGHBOR

CLASSIFICATION REPORT

	precision	recall	f1-score	support
negative	1.00	1.00	1.00	1846
neutral	0.61	0.48	0.54	576
positive	0.52	0.64	0.57	506
accuracy			0.84	2928
macro avg	0.71	0.71	0.70	2928
weighted avg	0.84	0.84	0.83	2928

```
[[1846    0    0]
 [   2  277  297]
 [   3  180  323]]
```

```
from sklearn.metrics import accuracy_score
print(accuracy_score(y_test,y_pred))
```

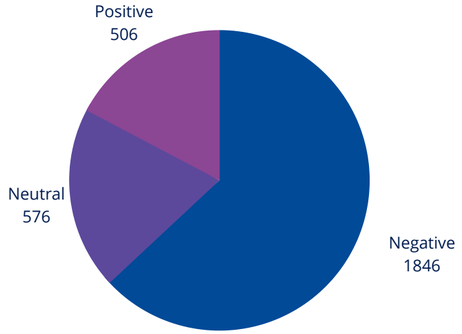
```
0.8353825136612022
```

K - VALUE

- The value of K was taken as 53.
- K value : the closest odd number to the square root of testing data.

ACCURACY

- The accuracy at K= 53 is 0.84
- Thus, the KNN Algorithm gave 84% accuracy

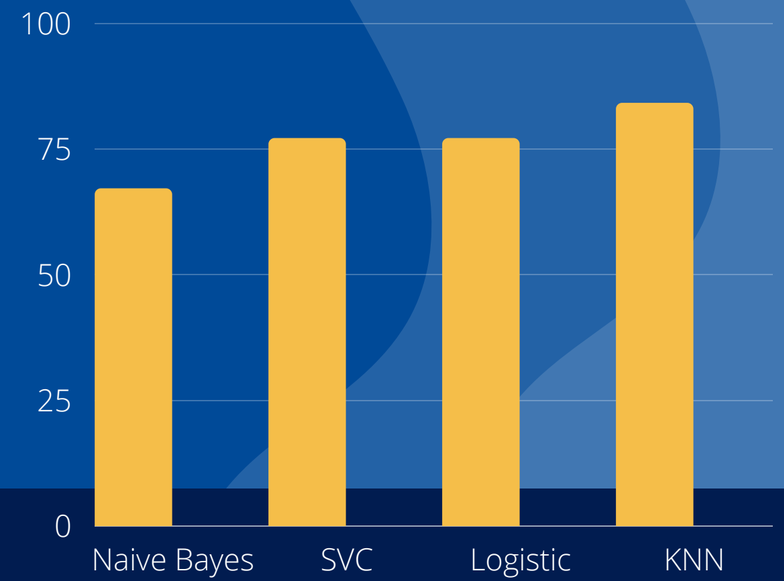


REPORT ANALYSIS

- **Negative** : We get a very good precision(1.00) and recall(1.00) for this class.
- **Neutral** : The precision(0.61) and recall(0.48) for this class is not ideal. This shows that 61% of the predicted outcomes for class 'Neutral' were correct, whereas only 48% of the 'Neutral' cases were correctly predicted.
- **Positive** : We get 0.52 precision for this class, which shows that 52% of the predicted outcomes for class 'Positive' were correct. And only 64% of the 'Positive' cases were correctly predicted.

CONCLUSION

ALGORITHM	ACCURACY
NAIVE BAYES	67%
SUPPORT VECTOR CLASSIFIER	77%
LOGISTIC REGRESSION	77%
K NEAREST NEIGHBOR	84%



The **KNN Algorithm** gives the highest accuracy, which is **84%**.

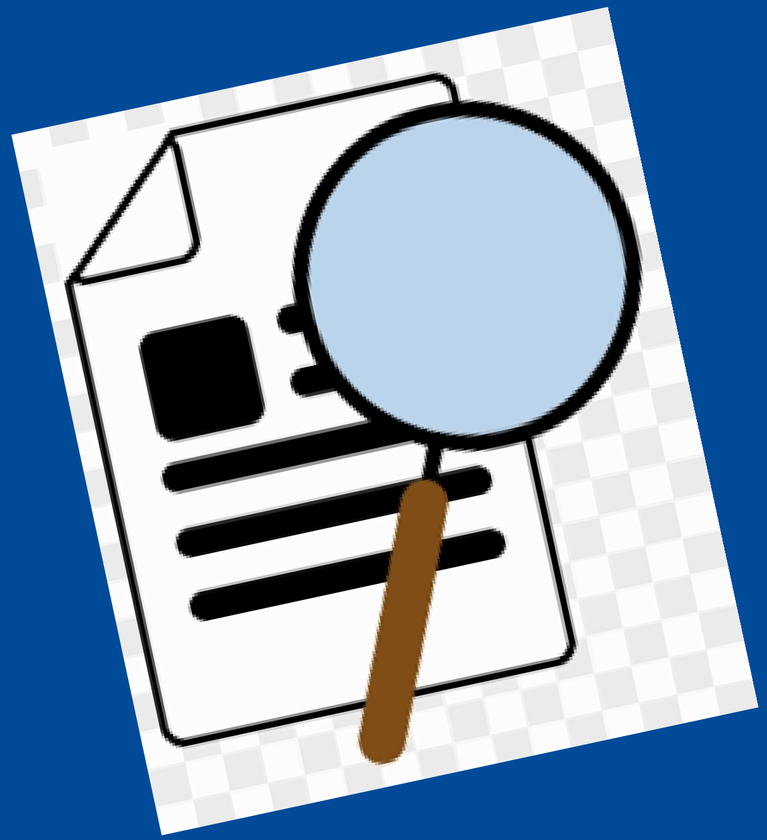
We conclude that **the KNN Algorithm** is the best fit, out of the 4 algorithms, for this particular dataset because:

- Most of the classifier algorithms are easy to implement for binary problems and need effort to implement for multi class whereas **K-NN adjusts to multi class** without any extra efforts.
- **K-NN has no assumptions**: K-NN is a non-parametric algorithm which means there are assumptions to be met to implement K-NN. Parametric models have lots of assumptions to be met by data before it can be implemented which is not the case with K-NN.
- KNN Algorithm works well even with less attributes.

***Note**: K-NN doesn't perform well on imbalanced data. More than 65% of the data belongs to 'Negative' class. Thus, the model will ultimately give a lot of preference to 'Negative' class. This results in getting the less common classes wrongly classified. Though the K-NN Model is best fit for this dataset, it is not an ideal model as it gives low accuracy to the 'Neutral' and 'Positive' classes.*

REFERENCES

1. https://monkeylearn.com/
2. https://www.analyticsvidhya.com/
3. https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc
4. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (Book)
5. https://arxiv.org/ftp/arxiv/papers/1601/1601.06971.pdf
6. https://scikit-learn.org/stable/modules/naive_bayes.html



Dataset Link:

https://www.kaggle.com/crowdflower/twitter-airline-sentiment?select=Tweets.csv