MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY

TWITTER SENTIMENT ANALYSIS

UNDER THE GUIDANCE OF:
DR. GHANSHYAM SINGH THAKUR

Presented By: -
Parvati sidu(212120030)
Navin kumar Sinku(212120049)
sunny Horo(212120055)

CONTENT INDEX

- INTRODUCTION
- OBJECTIVE
- FLOW CHART

- MATERIAL AND METHODS
- MODELS

- RESULTS
- MODEL EVALUATION
- CONCLUSION

SENTIMENT ANALYSIS

Senti ment analysis is the process of determining the sentiment behind the tweets, whether a piece of written tweet is **positive**, **neutral** or **negative**. it's also called opinion mining.
it's a really useful analysis since we could possibly determine the overall opinion about a selling object, or predict stock markets for the given company.

APPLICATION

1 **BUSINESS**

To understand customers' feeling towads product and brands.

2 **POLITICS**

Keep track of political view, to detect consistency and inconsistency between statements of the political parties.

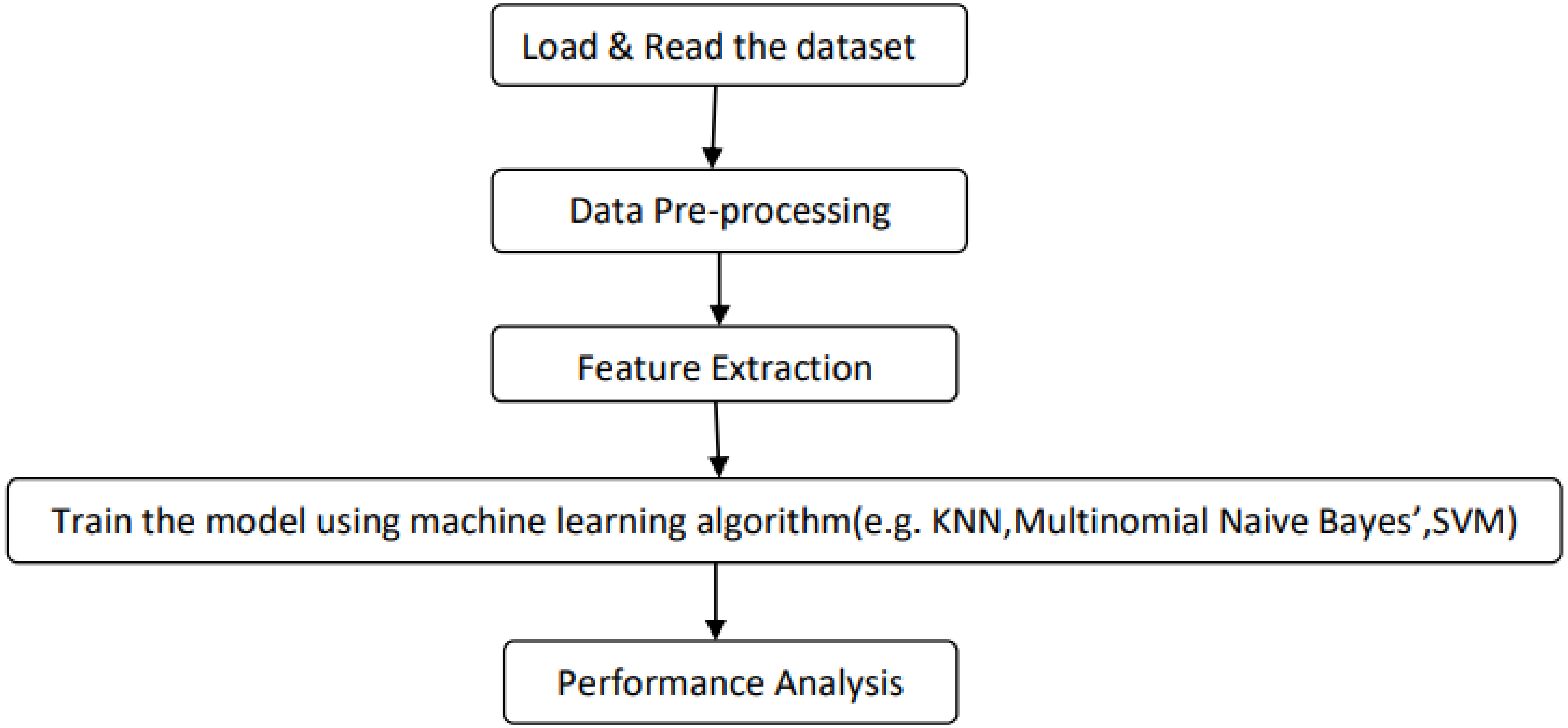
3 **PUBLIC ACTIONS**

Monitor and analyse social phenomena.

OBJECTIVE

1. Sentiment Analysis to determine the attitude of the mass is +ve,-ve or neutral.
2. Graphical representation of the sentiment in form of confusion matrix,ROC-AUC curve,etc.
3. To find the machine learning algorithm that best fits the twitter data set.
4. To achieve the accuracy of our model greater than 95%.

PROCESS FLOWCHART



SOURCE OF TWITTER DATA SET

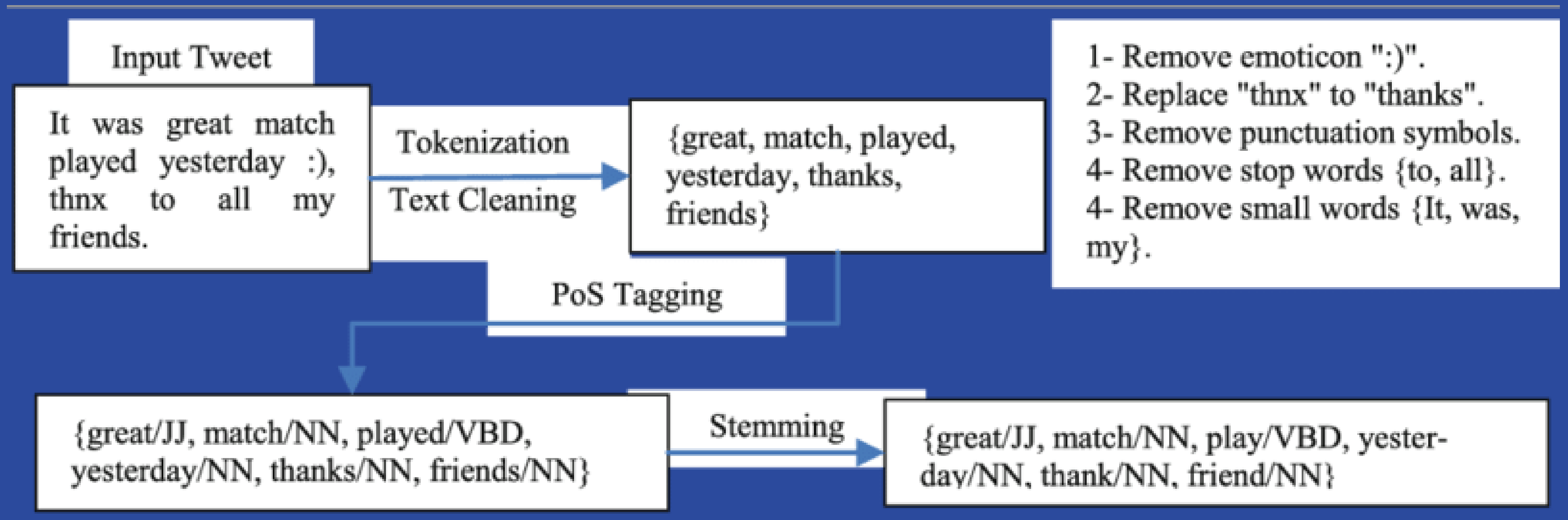
HTTPS://WWW.KAGGLE.COM/DATASETS/JP797498E/TWITTER-ENTITY-SENTIMENT-ANALYSIS

Twitter sentiment Analysis Dataset is taken from Kaggle Website that consists 69491 of rows and 4 columns.

- **Tweet id:** Unique id of the tweet.
- **Entity:** It represents the type of game.
- **Sentiment:** the polarity of the tweet (positive, negative or neutral).
- **Tweet content:** It refers to the text of the tweet.

2.DATA PREPROCESSING

(a). Remove unnecessary columns(features) from data frame that do not contribute in determining the sentiment of the twitter text.



(b).drop duplicate data from the data frame

	Text	Target
0	im getting on borderlands and i will murder yo...	Positive
1	I am coming to the borders and I will kill you...	Positive
2	im getting on borderlands and i will kill you ...	Positive
3	im coming on borderlands and i will murder you...	Positive
4	im getting on borderlands 2 and i will murder ...	Positive

1

TOKENIZATION

I love this game → **TOKENIZATION** →

"I"
"love"
"this"
"game

2

REMOVE PUNCTUATION

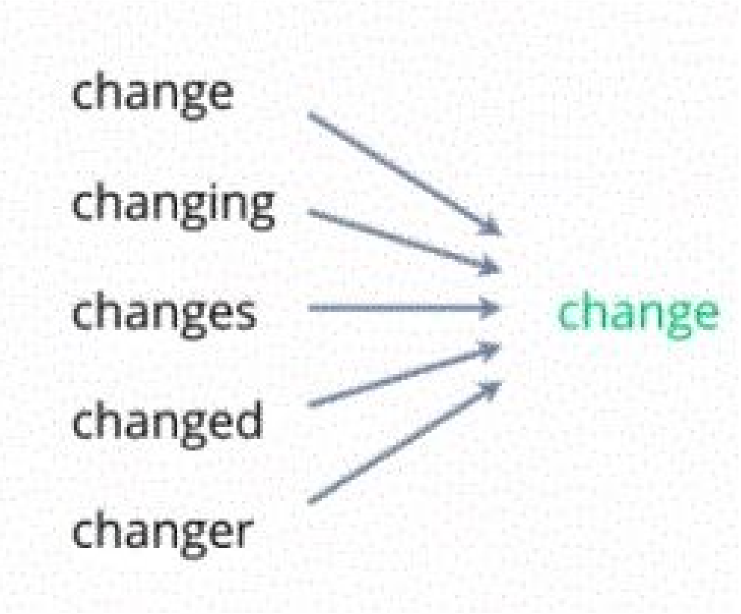
Punctuation marks, such as commas(,), periods(.), and question marks(?), can add clarity and structure to text when we read it, but they can also add noise when we try to analyze or process text computationally.

3

STOPWORDS

"the", "and", "a", "an", "in", "of", and "to".

4 LEMMATIZE



5 STEMMING

Before cleaning

	Text	Target	Sentiment
0	im getting on borderlands and i will murder yo...	Positive	1
1	I am coming to the borders and I will kill you...	Positive	1
2	im getting on borderlands and i will kill you ...	Positive	1
3	im coming on borderlands and i will murder you...	Positive	1
4	im getting on borderlands 2 and i will murder ...	Positive	1

After cleaning

```
0        get borderland murder
1             come border kill
2          get borderland kill
3       come borderland murder
4        get borderland murder
Name: Text, dtype: object
```

3.FEATURE EXTRACTION

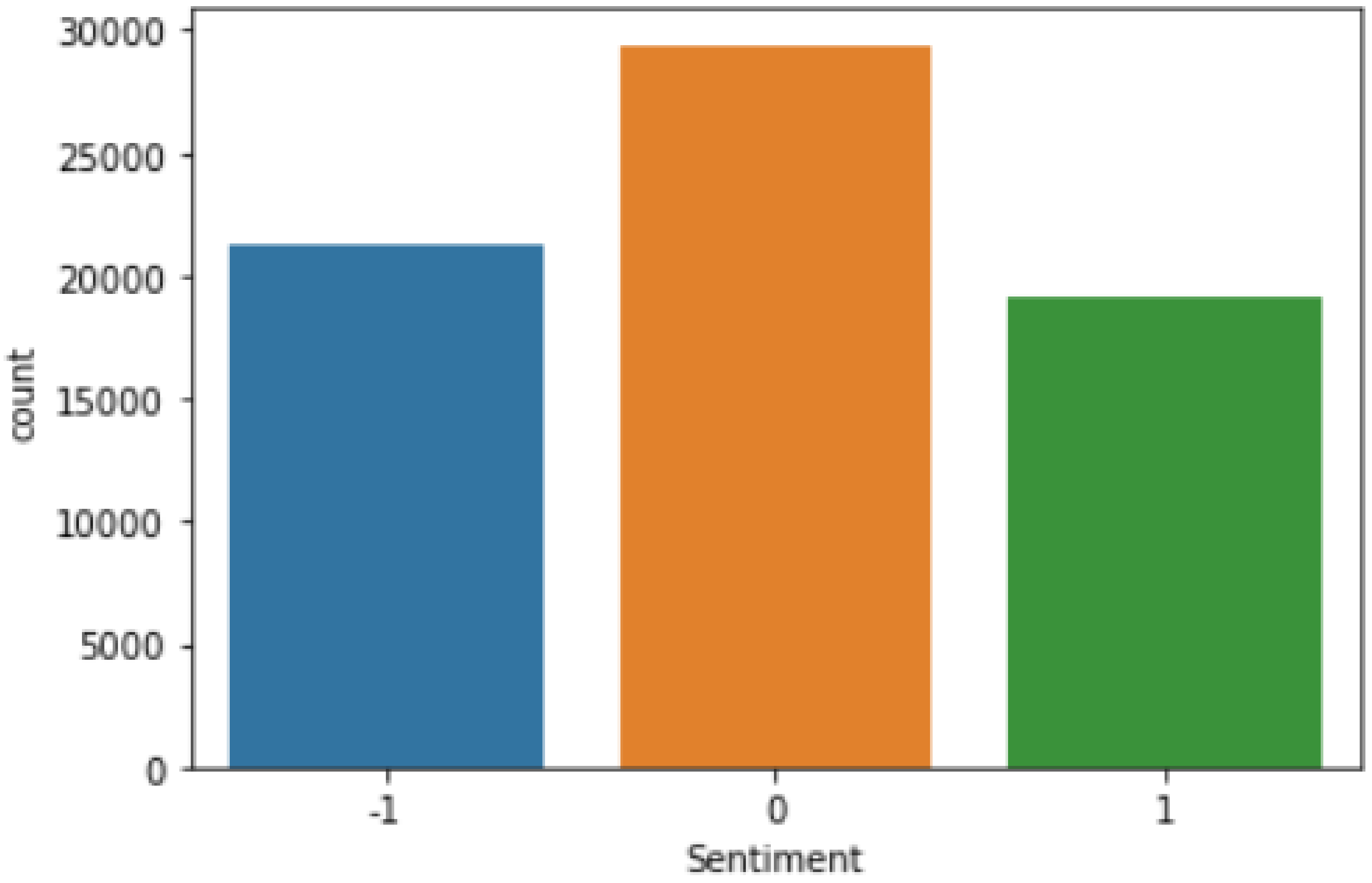
Since our machine cannot understand text data,therefore we have to convert text data into numerical data.

```
sentiment = []

for i in df["Target"]:
    if i == "Positive":
        sentiment.append(1)
    elif (i == "Irrelevant") or (i == "Neutral"):
        sentiment.append(0)
    else:
        sentiment.append(-1)
df["Sentiment"] = sentiment
```

Sentiment column is added

	Text	Target	Sentiment
0	get borderland murder	Positive	1
1	come border kill	Positive	1
2	get borderland kill	Positive	1
3	come borderland murder	Positive	1
4	get borderland murder	Positive	1
...	...	...	...
74677	realiz window partit mac like year behind nvid...	Positive	1
74678	realiz mac window partit year behind nvidia dr...	Positive	1
74679	realiz window partit mac year behind nvidia dr...	Positive	1
74680	realiz window partit mac like year behind nvid...	Positive	1
74681	like window partit mac like year behind driver...	Positive	1

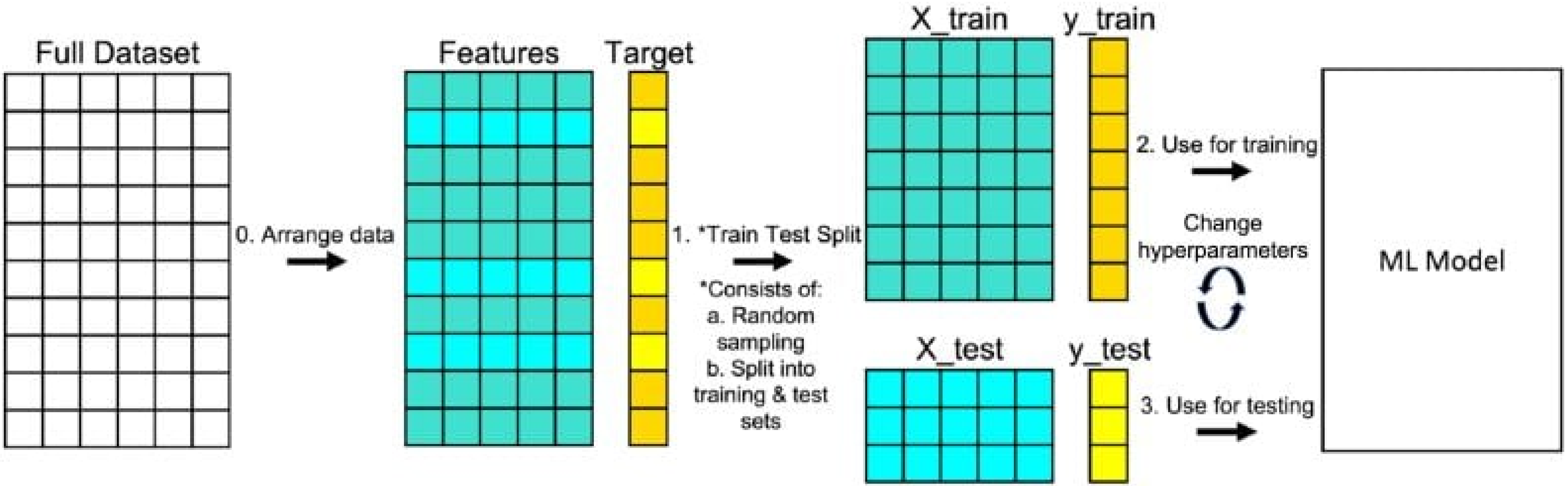
```
sns.countplot(y,data=df)
```




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

```
X_train,X_test,y_train,y_test = train_test_split(X,y, test_size = 0.30,random_state= 42,stratify = y)
```

e.g. of Train_Test_Split :



(a)Vectorization:We use count vectorizer to convert the text column into numerical form

```
from sklearn.feature_extraction.text import CountVectorizer
```

```python
vt = CountVectorizer(analyzer="word")
X_train_count = vt.fit_transform(X_train)
X_test_count = vt.transform(X_test)
print(vt.vocabulary_)
```

```
{'covid': 2565, 'big': 1152, 'shock': 10932, 'hiv': 5540, 'vaccin': 13066, 'johnson': 6318, 'stop': 11647, 'trial': 12561, 't
hank': 12177, 'nk': 8235, 'rockstarsupport': 10247, 'hi': 5489, 'screen': 10641, 'comput': 2379, 'name': 8014, 'zen': 14032,
'new': 8136, 'grand': 5050, 'theft': 12203, 'auto': 758, 'onlin': 8535, 'bought': 1426, 'whale': 13526, 'cash': 1842, 'insu
r': 5983, 'card': 1807, 'exact': 3920, 'th': 12172, 'april': 540, 'havent': 5339, 'entir': 3769, 'receiv': 9810, 'bonu': 137
1, 'yet': 13928, 'whole': 13572, 'day': 2883, 'andov': 419, 'hour': 5662, 'latest': 6771, 'fabien': 4008, 'henon': 5457, 'l
e': 6809, 'journal': 6355, 'elixirtip': 3670, 'cant': 1779, 'wait': 13349, 'lick': 6913, 'gipper': 4848, 'boot': 1392, 'sic
k': 10997, 'waifu': 13346, 'pistol': 9072, 'skin': 11087, 'sell': 10726, 'whiten': 13561, 'cream': 2616, 'concern': 2389, 'ra
cism': 9639, 'hors': 5644, 'behind': 1046, 'punk': 9526, 'tool': 12439, 'destruct': 3079, 'exactli': 3921, 'democrat': 3016,
'care': 1810, 'hard': 5289, 'work': 13710, 'christian': 2105, 'think': 12268, 'radic': 9646, 'jewish': 6260, 'compani': 2349,
```

```python
print(X_train_count.toarray())
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

```
<48841x14108 sparse matrix of type '<class 'numpy.int64'>'
	with 480354 stored elements in Compressed Sparse Row format>
```

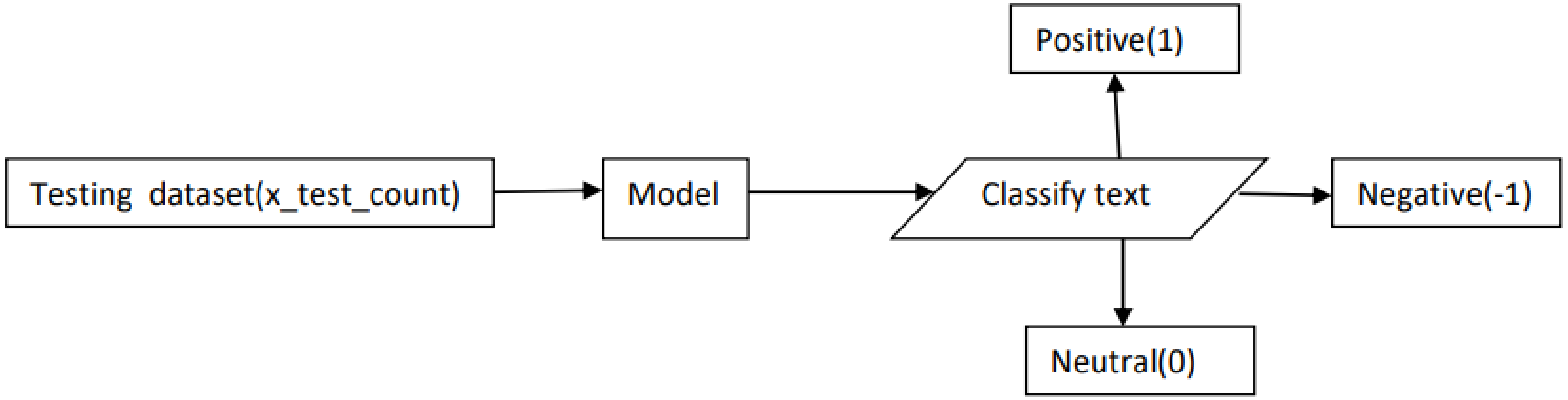
4.Train the model using classification algorithm

(a)

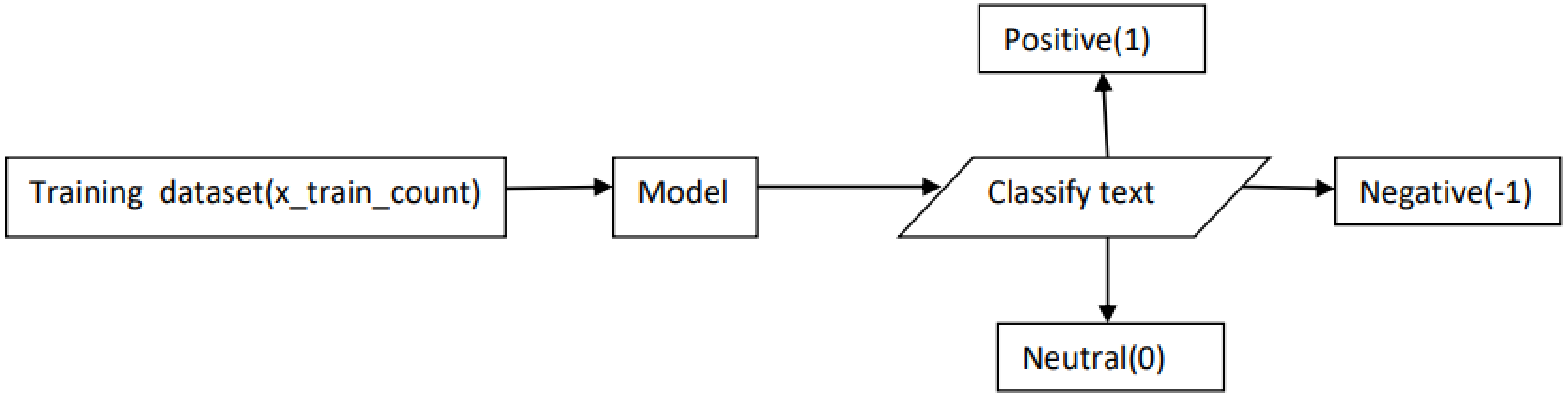
```
#import
from sklearn.branch import model_name

#create instance
model = model_name()

#fit model
model.fit(X_train, y_train)
```

#We train the model using fit() function

(b)Now,our model will predict the sentiment based training data(x_train) as well as testing data(x_test) y_test_prediction=model_name.predict(x_test_count)



y_train_prediction=model_name.predict(x_train_count)



Performance Metrics(Explaination)

Truth	1	0	1	-1	0	1	-1	-1	0	1
Pred.	1	0	1	1	0	0	1	-1	1	-1
	✓	✓	✓		✓			✓		

#Class 1 : Positive Text, **Class 0 :** Neutral Text **& Class -1 :** Negative Text

Accuracy = (Total correct predictions)/(Total no. of predictions)

=5/10 = 0.5

(a)For class 1(positive) :

Precision = True(positive)/[True(positive)+False(positive)]

= 2/(2+3) = 2/5

(#Precision is out of all positive predictions,how many you got it right?)

Recall = True(positive)/[Total(positive)(in truth)]

= 2/4 = ½

(#Recall is out of all positive(in truth),how many you got it right?)

(b)For class -1(negative) :

Precision = True(negative)/[True(negative)+False(negative)]

= 1/(1+1) = ½

(#Precision is out of all negative predictions,how many you got it right?)

Recall = True(negative)/[Total(negative)(in truth)]

= 1/3

(#Recall is out of all negative(in truth),how many you got it right?)

(c)For class 0(neutral) :

Precision = True(neutral)/[True(neutral)+False(neutral)]

= 2/(2+1) = 2/3

(#Precision is out of all neutral predictions,how many you got it right?)

Recall = True(neutral)/[Total(neutral)(in truth)

= 2/3

(#Recall is out of all neutral(in truth),how many you got it right?)

The F_1 Score is given by :

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

MULTINOMIAL NAIVE BAYES'

Multinomial Naive Bayes is a variant of the Naive Bayes algorithm that is commonly used for text classification problems, where the features are discrete counts (such as the counts of words in a document) rather than continuous numerical values.

In Multinomial Naive Bayes, the assumption is that the probability distribution of the features (i.e., the word counts) for each class follows a multinomial distribution. In other words, each class has a probability distribution over the different possible feature values, and this distribution is estimated from the training data.

During training, the algorithm estimates the probability distribution of the features for each class by counting the number of occurrences of each feature in the training documents of that class. These counts are then smoothed using techniques such as Laplace smoothing to avoid zero probabilities.

During prediction, the algorithm calculates the probability of the document belonging to each class using Bayes' theorem and the estimated probability distributions. The class with the highest probability is then selected as the predicted class.

MODEL EVALUATION

(a)On training data

```
nb_model = MultinomialNB()
nb_model.fit(X_train_count,y_train)
```

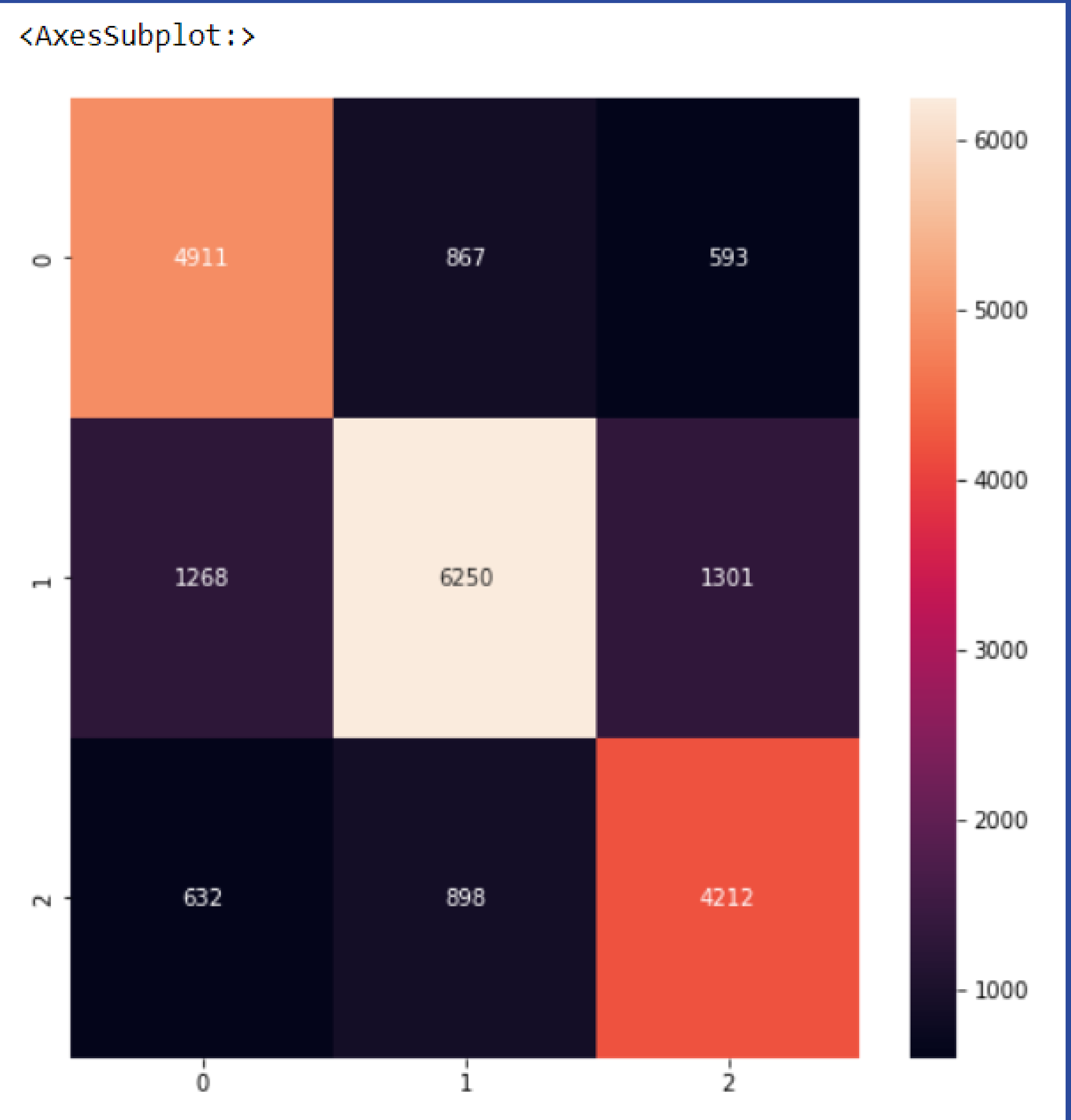
```
X Train
              precision    recall  f1-score   support

          -1       0.76      0.81      0.78     14867
           0       0.81      0.75      0.78     20577
           1       0.74      0.78      0.76     13397

    accuracy                           0.77     48841
   macro avg       0.77      0.78      0.77     48841
weighted avg       0.78      0.77      0.77     48841
```

(b)On testing data

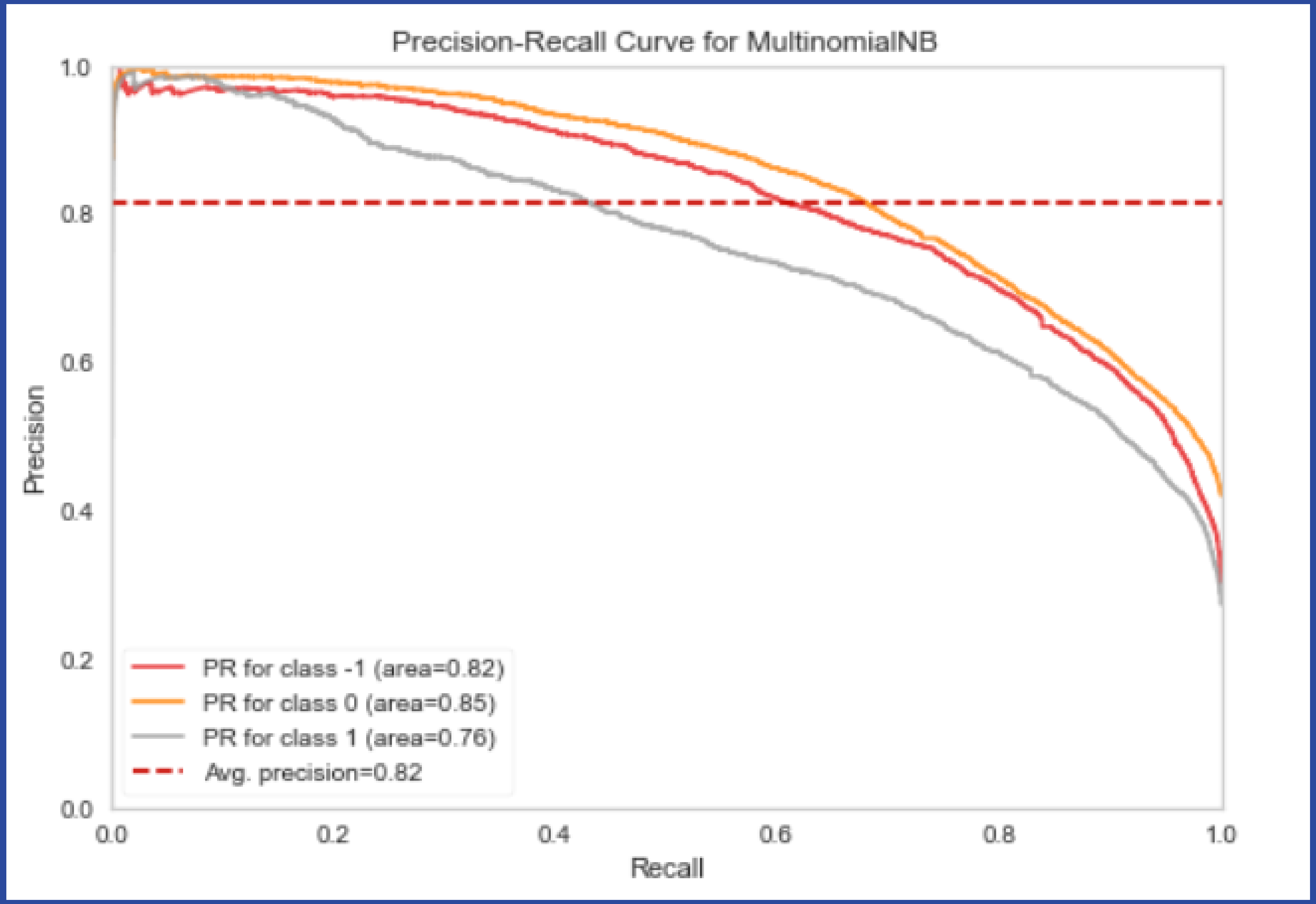
```
nb_pred = nb_model.predict(X_test_count)
nb_train_pred = nb_model.predict(X_train_count)
```

```
X Test
              precision    recall  f1-score   support

          -1       0.72      0.77      0.75      6371
           0       0.78      0.71      0.74      8819
           1       0.69      0.73      0.71      5742

    accuracy                           0.73     20932
   macro avg       0.73      0.74      0.73     20932
weighted avg       0.74      0.73      0.73     20932
```

(b). Confusion Matrix

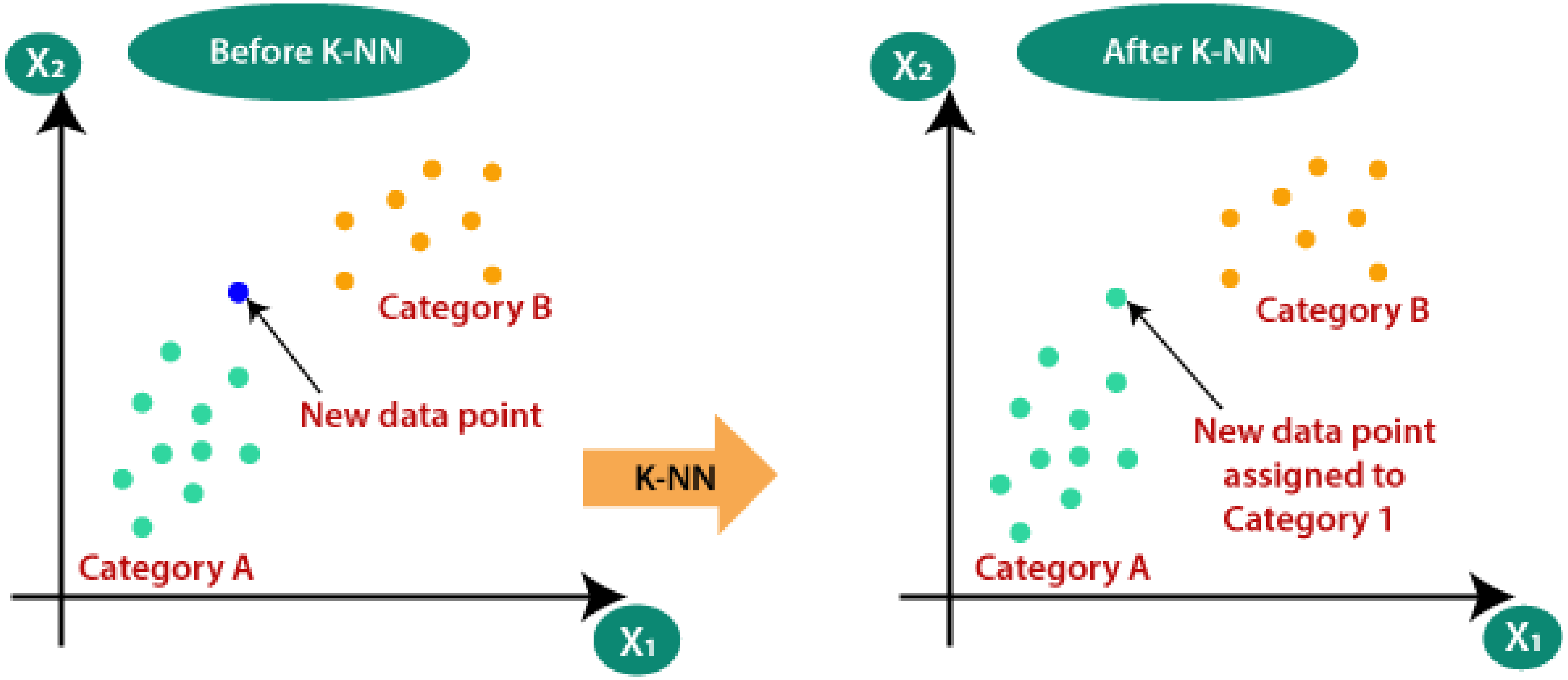


(c). Precision recall curve



KNN(K-NEAREST NEIGHBOR)

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.



MODEL EVALUATION

(a)On training data

X Train

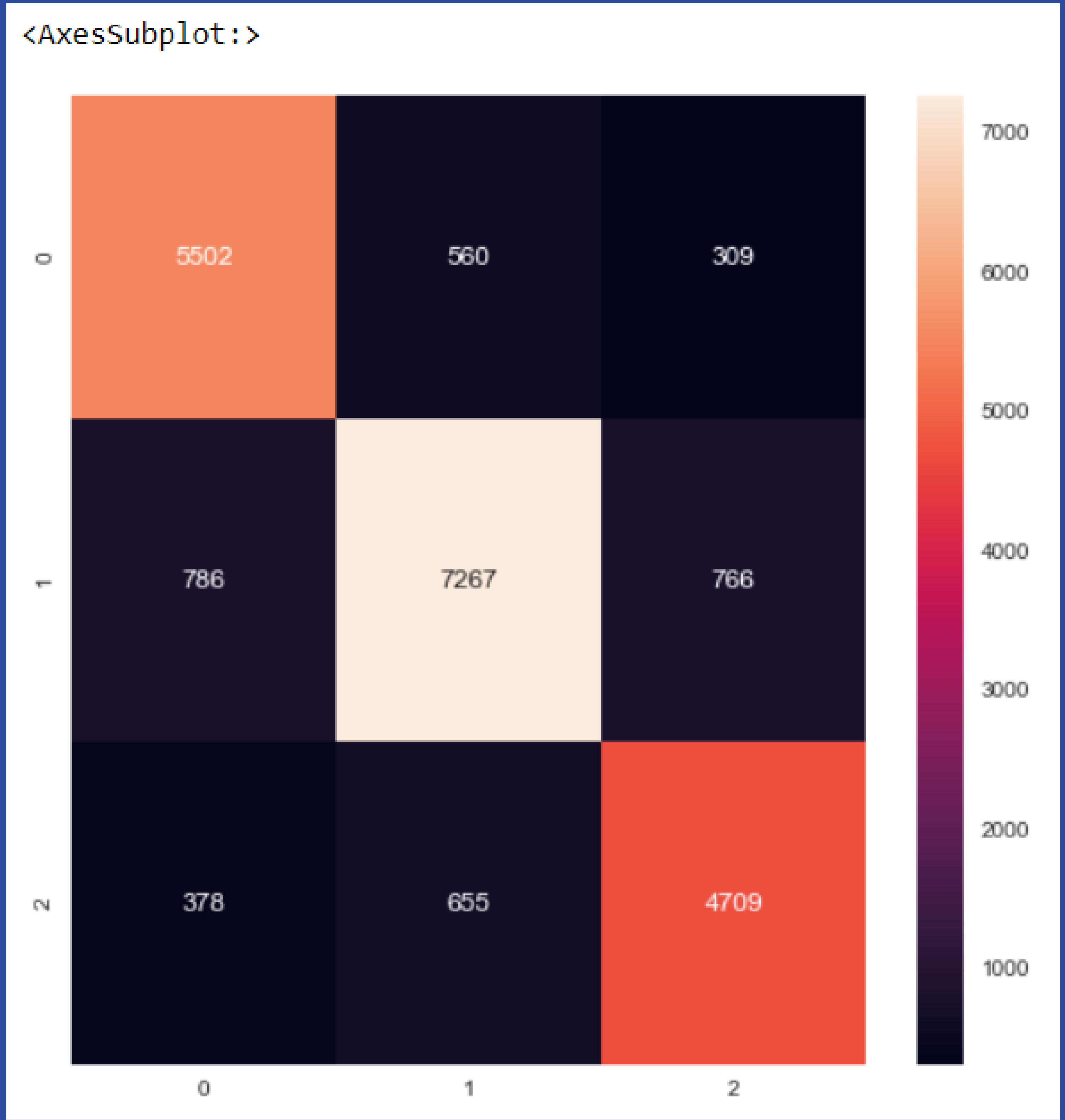
	precision	recall	f1-score	support
-1	0.92	0.94	0.92	14867
0	0.93	0.92	0.92	20577
1	0.92	0.91	0.91	13397
accuracy			0.92	48841
macro avg	0.92	0.92	0.92	48841
weighted avg	0.92	0.92	0.92	48841

(b)On testing data

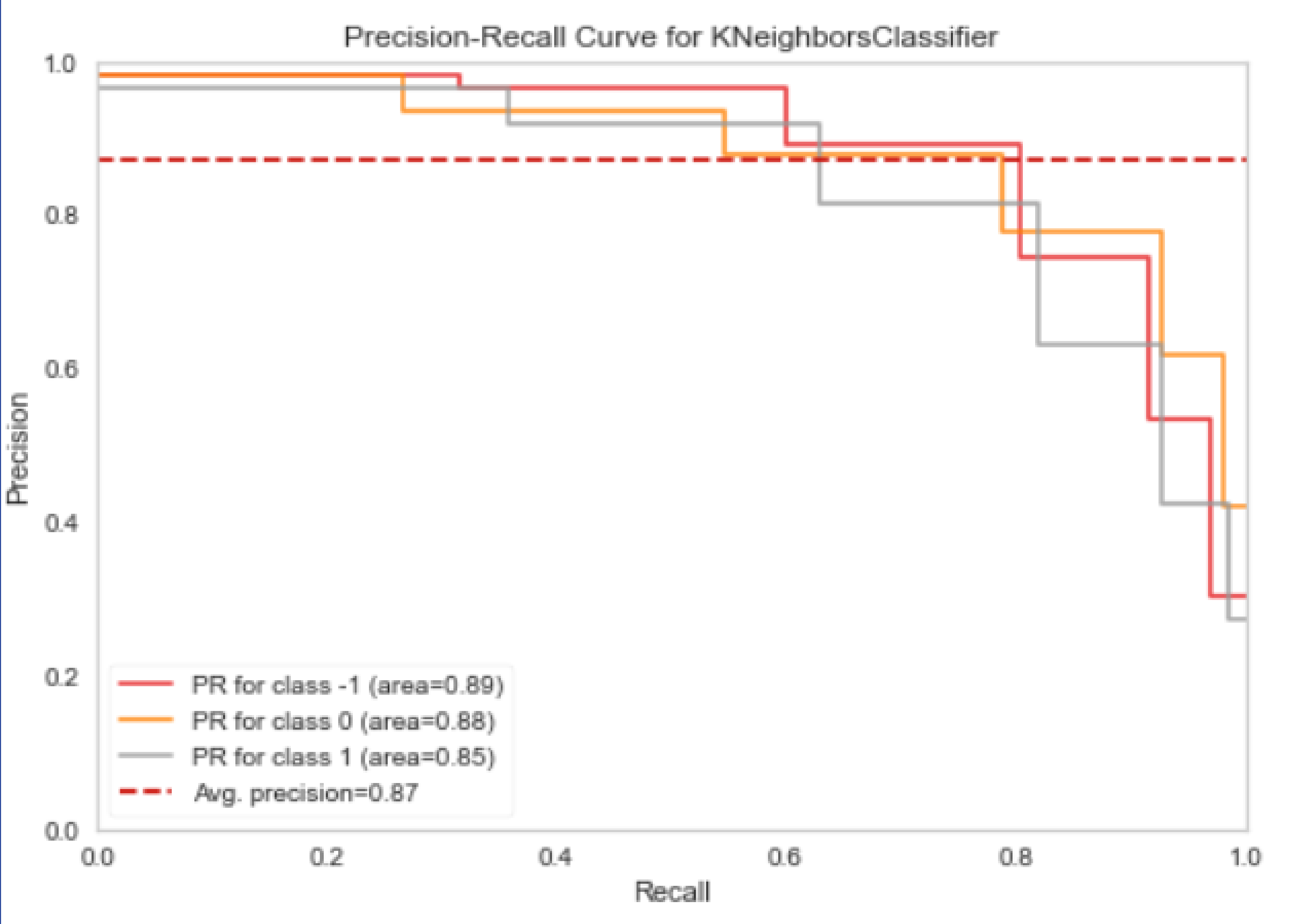
X Test

	precision	recall	f1-score	support
-1	0.83	0.86	0.84	6371
0	0.86	0.82	0.84	8819
1	0.81	0.82	0.82	5742
accuracy			0.83	20932
macro avg	0.83	0.84	0.83	20932
weighted avg	0.84	0.83	0.83	20932

(b). Confusion Matrix

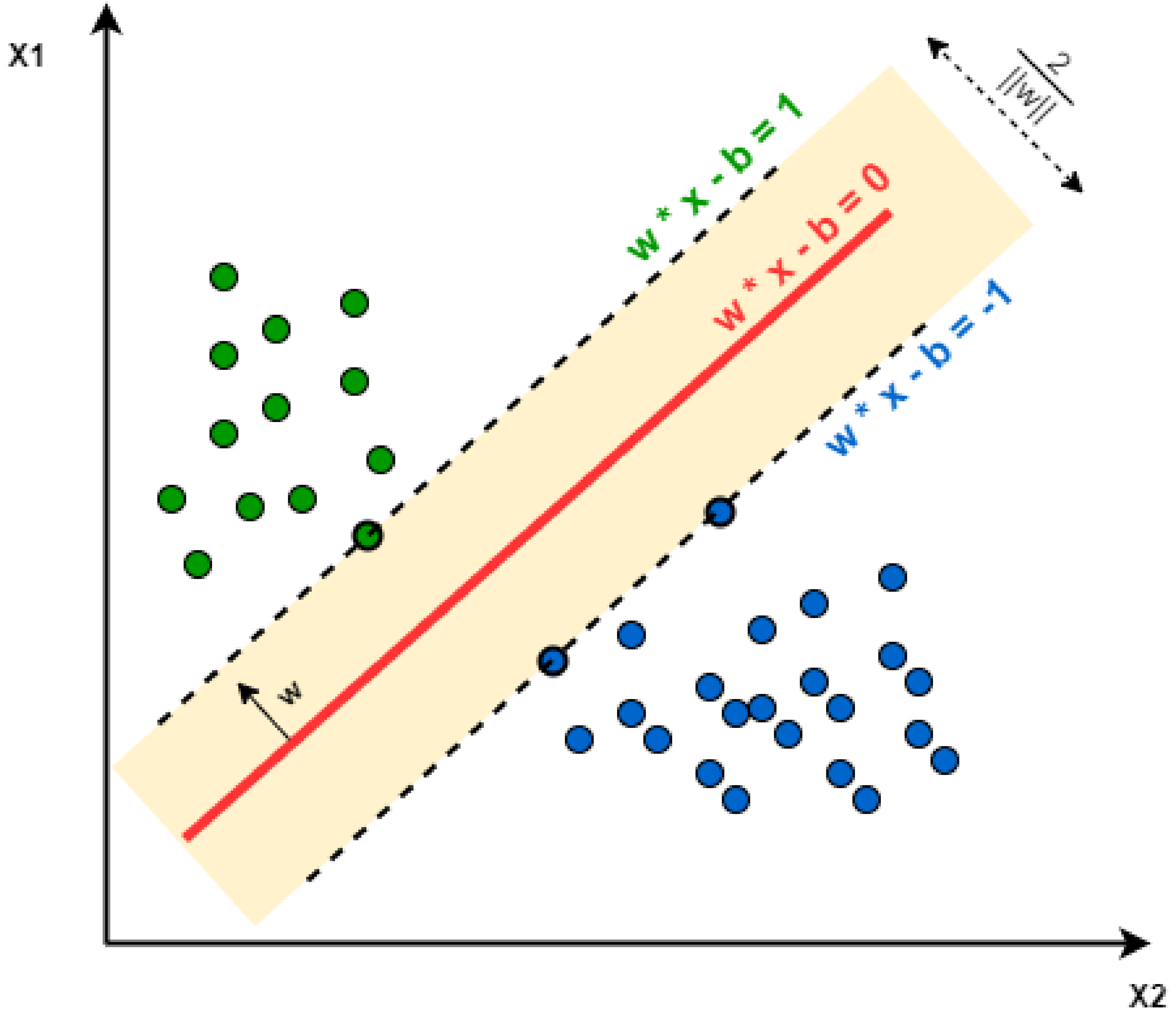


(c). Precision recall curve

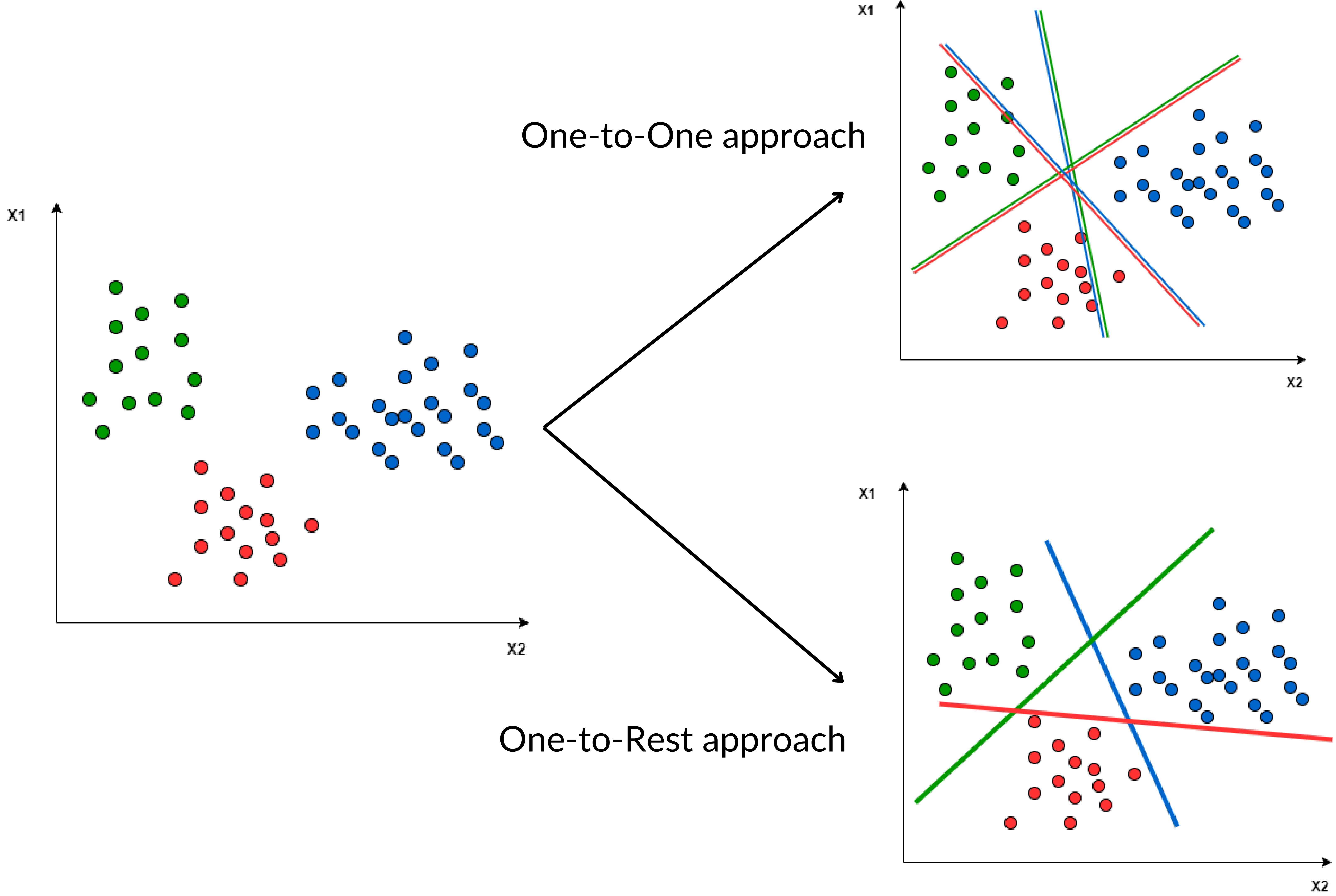


SVM(SUPPORT VECTOR MACHINE)

SVM tries to find a line that maximizes the separation between a two-class data set of 2-dimensional space points. To generalize, the objective is to find a hyperplane that maximizes the separation of the data points to their potential classes in an n-dimensional space. The data points with the minimum distance to the hyperplane (closest points) are called Support Vectors.



- Red line is hyperplane
- Scattered lines are margine
- There are three support vector 2 is blue and 1 is green toching scatter line


X1
X2
One-to-One approach
X1
X2
One-to-Rest approach
X1
X2

MODEL EVALUATION

(a)On training data

X Train

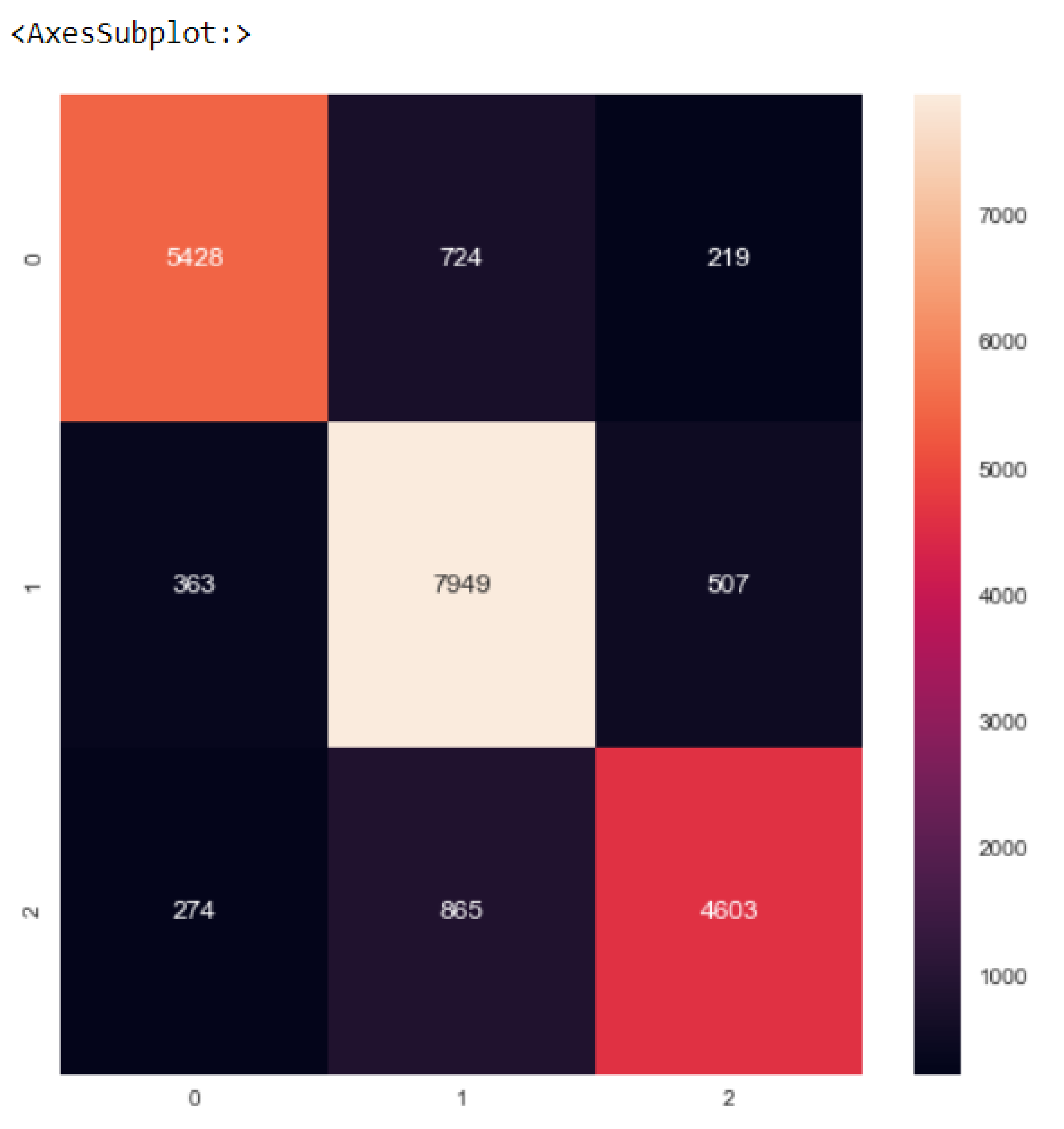
	precision	recall	f1-score	support
-1	0.94	0.90	0.92	14867
0	0.89	0.95	0.92	20577
1	0.93	0.88	0.90	13397
accuracy			0.92	48841
macro avg	0.92	0.91	0.92	48841
weighted avg	0.92	0.92	0.92	48841

(b)On testing data

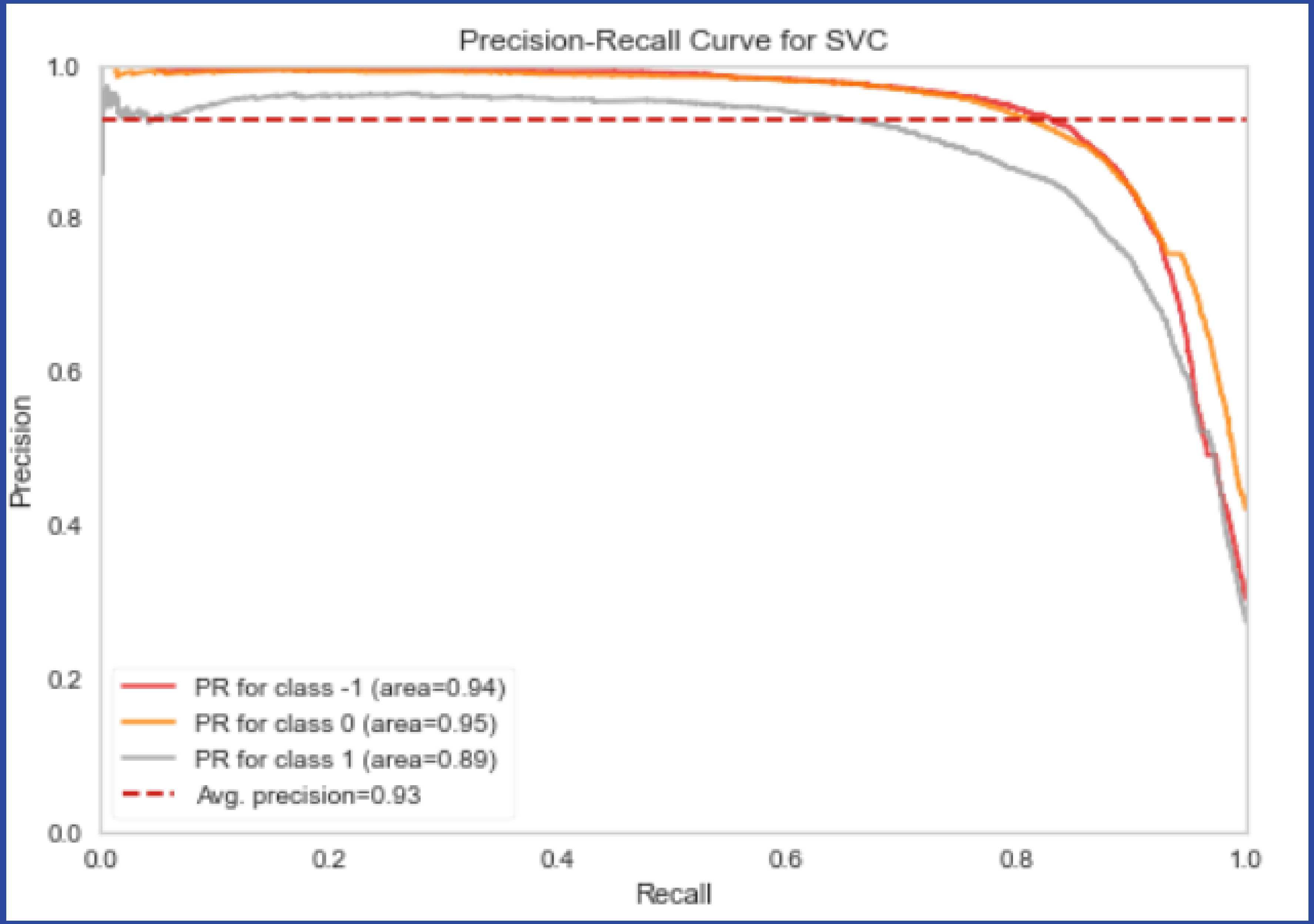
X Test

	precision	recall	f1-score	support
-1	0.89	0.85	0.87	6371
0	0.83	0.90	0.87	8819
1	0.86	0.80	0.83	5742
accuracy			0.86	20932
macro avg	0.86	0.85	0.86	20932
weighted avg	0.86	0.86	0.86	20932

(b). Confusion Matrix

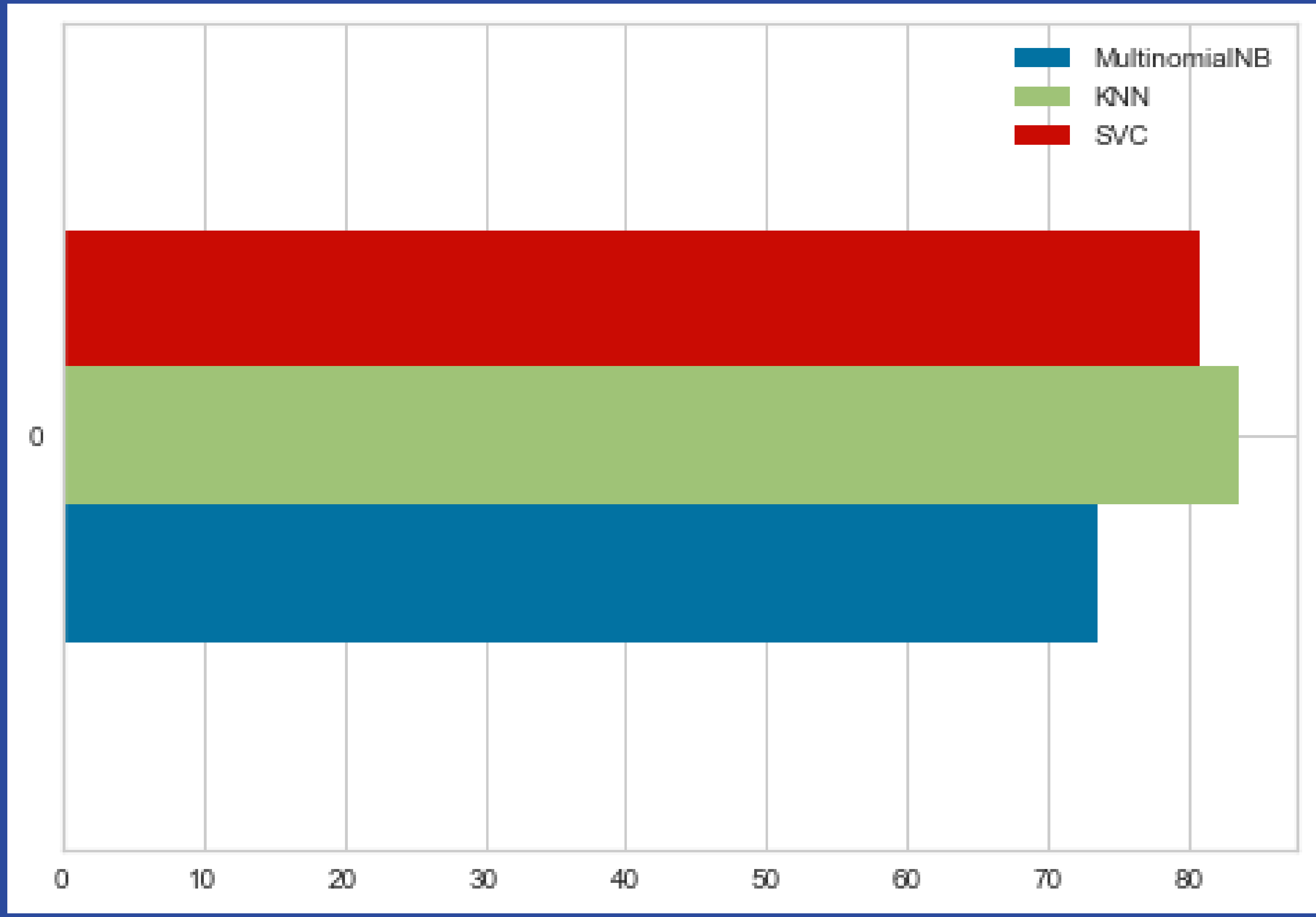


(c). Precision recall curve

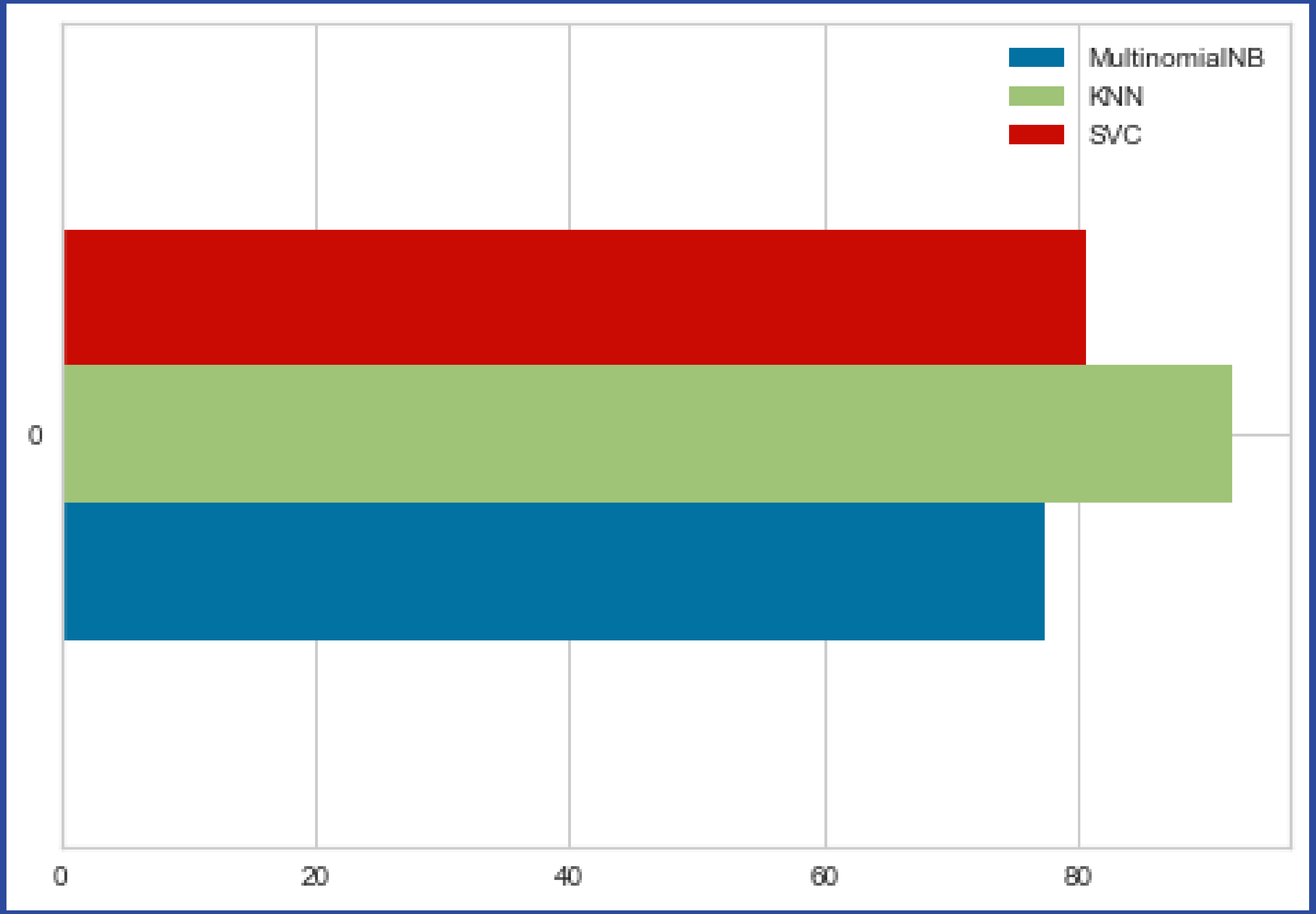


COMPARISON BETWEEN ALL

(a)On testing data



(b)On training data



CONCLUSION

- Twitter Sentimental Analysis is used to identify as well as classify the sentiments that are expressed in the text source.
- MultinomialNB, KNN, and SVC are some of the ML algorithms that can be used for Twitter Sentimental Analysis.
- After observing the accuracy of each classification algorithm, we can conclude that the model which is trained on KNN has the highest accuracy of **92%** on training data.

REFERENCES

- Brijesh Bakariya and G.S. Thakur, "An Efficient Algorithm for Extracting Infrequent Itemsets from Weblog", The International Arab Journal of Information Technology (IAJIT),2019 (2)
- Meylan Wongkar , Apriandy Angdresey, “Sentiment Analysis Using Naive Bayes Algorithm Of The Data Crawler: Twitter”, 2019 Fourth International Conference on Informatics and Computing (ICIC).
- Aliza Sarlan, Chayanit Nadam, Shuib Basri, “Twitter Sentiment Analysis”, International Conference on Information Technology and Multimedia (ICIMU) 2014.
- Ms.R.Monika, Dr.S.Deivalakshmi, Dr.B.Janet, “Sentiment Analysis of US Airlines Tweets using LSTM/RNN”, IEEE 9th International Conference on Advanced Computing (IACC) 2019.
- Brijesh Bakariya and G.S. Thakur, "Mining Rare Itemsets from Weblog Data",National Academy Science Letters, 2015 page 1-5.

THANK YOU