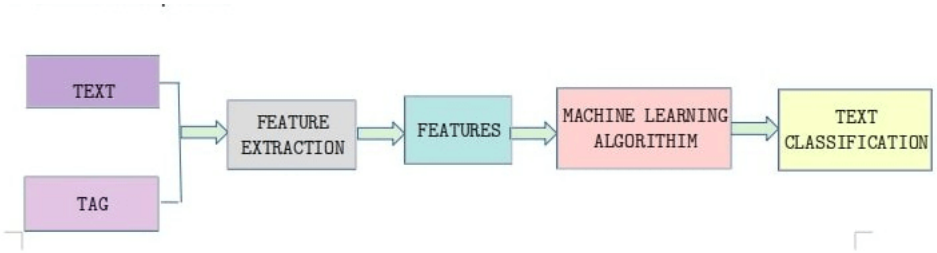
Text Classification - Natural Language Processing

Text classification also known as *text tagging* or *text categorization* is the process of categorizing text into organized groups. By using Natural Language Processing (NLP), text classifiers can automatically analyze text and then assign a set of pre-defined tags or categories based on its content.

This notebook gives a brief overview of performing ***literature text tagging*** using Naive Bayes, Logistic Regression, Support Vector Machines and Decision Tree Classifier. The data consists of approximately 1500 text pieces, which were tagged as ***cult**, **paranormal**, *and* **dramatic***. Our goal in this kernel is to explore the process of training and testing text classifiers for this dataset.



Import Required Libraries

In [1]:
```
import sys
import nltk
import sklearn
import pandas as pd
import numpy as np

# import matplotlib as mpl
# import matplotlib.cm as cm
import matplotlib.pyplot as plt
import plotly.graph_objects as go
from dash import Dash, dcc, html, Input, Output
import plotly.express as px
from plotly.offline import init_notebook_mode
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

import spacy

from sklearn.preprocessing import LabelEncoder

from tqdm import tqdm

from nltk.corpus import stopwords
from nltk.classify.scikitlearn import SklearnClassifier
from nltk.stem import WordNetLemmatizer



from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC


from nltk.classify.scikitlearn import SklearnClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.ensemble import VotingClassifier

from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix

from time import time

# import warnings
# warnings.filterwarnings("ignore")

tqdm.pandas()
```

```
lemm = WordNetLemmatizer()
init_notebook_mode(connected=True)
sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (20,8)
plt.rcParams['font.size'] = 18
```

Analysis of Data

In [2]:
```
data = pd.read_csv('../Data/task.csv', encoding='utf-8')
data.head()
```

Out[2]:

	Unnamed: 0	Title	Synopsis	Tag
0	0	I tre volti della paura	Note: this synopsis is for the orginal Italian...	cult
1	1	Mitt liv som hund	The action takes place in the years 1958-1959 ...	cult
2	2	The Brood	At the Somafree Institute, Dr. Hal Raglan humi...	cult
3	3	The Haunted	This creepy and scary story centers around The...	paranormal
4	4	The Frozen Ground	The film opens in an Anchorage motel room in 1...	dramatic

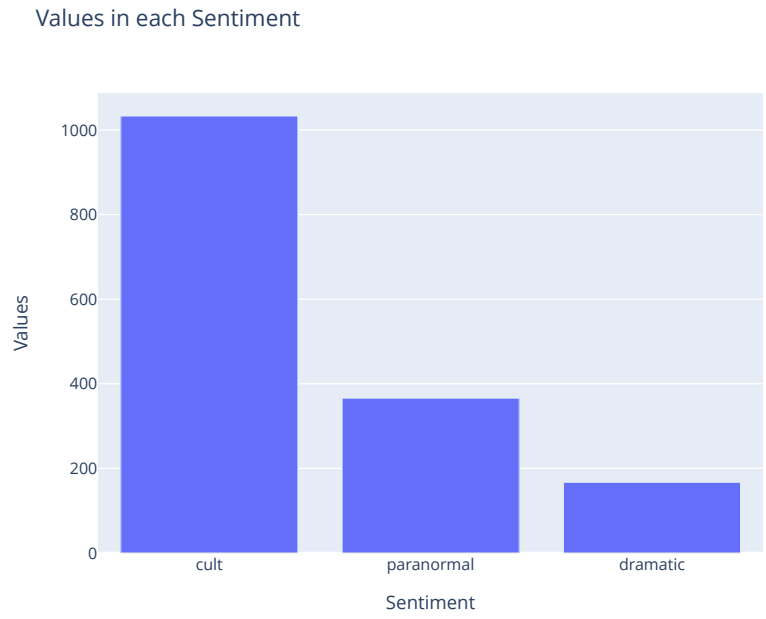
In [3]:
```
round(data["Tag"].value_counts()/len(data), 2)
```

Out[3]:
```
cult          0.66
paranormal    0.23
dramatic      0.11
Name: Tag, dtype: float64
```

There is an imbalance in the data with cultt being 66% in the dataset. We should keep this class imbalance mind when interpreting the classifier performance later. Let us first convert the class labels into numeric outcome variables for our ML methods.

In [ ]:
```
```

In [4]:
```
fig = go.Figure([go.Bar(x=data['Tag'].value_counts().index, y=data['Tag'].value_counts().tolist())])
fig.update_layout(
    title="Values in each Sentiment",
    xaxis_title="Sentiment",
    yaxis_title="Values")
fig.show()
```



In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1566 entries, 0 to 1565
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Unnamed: 0  1566 non-null   int64 
 1   Title       1566 non-null   object
 2   Synopsis    1566 non-null   object
 3   Tag         1566 non-null   object
dtypes: int64(1), object(3)
memory usage: 49.1+ KB
```

In [6]:
```
wordcloud = WordCloud(max_words=100, width=600, background_color='white').generate(" ".join(data['Synopsis']))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

Text Pre-processing

In [7]:
```
# tags = [ 0 if x=='cult'  else 1 if x=='paranormal' else 2 for x in data['Tag']]
# data['Tag'] = tags
# round(data["Tag"].value_counts()/len(data), 2)
```

In [8]:
```
enc = LabelEncoder()
label = enc.fit_transform(data['Tag'])
data['Tag'] = label
round(data["Tag"].value_counts()/len(data), 2)
```

Out[8]:
```
0    0.66
2    0.23
1    0.11
Name: Tag, dtype: float64
```

In [9]:
```
data.head()
```

Out[9]:

	Unnamed: 0	Title	Synopsis	Tag
0	0	I tre volti della paura	Note: this synopsis is for the orginal Italian...	0
1	1	Mitt liv som hund	The action takes place in the years 1958-1959 ...	0
2	2	The Brood	At the Somafree Institute, Dr. Hal Raglan humi...	0
3	3	The Haunted	This creepy and scary story centers around The...	2
4	4	The Frozen Ground	The film opens in an Anchorage motel room in 1...	1

In [10]:
```
data['Synopsis'][0]
```

Out[10]: 'Note: this synopsis is for the orginal Italian release with the segments in this certain order.Boris Karloff introduces three horror tales of the macabre and the supernatural known as the \'Three Faces of Fear\'.THE TELEPHONERosy (Michele Mercier) is an attractive, high-priced Parisian call-girl who returns to her spacious, basement apartment after an evening out when she immediately gets beset by a series of strange phone calls. The caller soon identified himself as Frank, her ex-pimp who has recently escaped from prison. Rosy is terrified for it was her testimony that landed the man in jail. Looking for solace, Rosy phones her lesbian lover Mary (Lynda Alfonsi). The two women have been estranged for some time, but Rosy is certain that she is the only one who can help her. Mary agrees to come over that night. Seconds later, Frank calls again, promising that no matter who she calls for protection, he will have his revenge. Unknown to Rosy, Mary is the caller impersonating Frank. Marry arrives at Rosy\'s apartment soon after, and does her best to calm Rosy\'s nerves. She gives the panic-struck woman a tranquillizer and puts her to bed.Later that night as Rosy sleeps, Mary gets up out of bed, and pens a note of confession: she was the one making the strange phone calls when she learned of Franks escape from prison. Knowing that Rosy would call on her for help, she explains that she felt it was her way of coming back into her life after their breakup. While she is busy writing, she fails to notice an intruder in the apartment. This time it is Frank, for real. He creeps up behind Mary and strangles her to death with one of Rosys nylon stockings. The sound of the struggle awaken Rosy and she gasps in fright. The murderous pimp realizes that he just killed the wrong woman, and slowly makes his way to Rosy\'s bed. However, earlier that night, Rosy had placed a butcher knife under her pillow at Mary\'s suggestion. Rosy seizes the knife and stabs Frank with it as he\'s beginning to strangle her. Rosy drops the knife and breaks down in hysteria, surrounded by the two corpses of her former lovers.THE WURDALAKIn 19th Century Russia, Vladimir D\'Urfe is a young nobleman on a long trip. During the course of his journey, he finds a beheaded corpse with a knife plunged into its heart. He withdraws the blade and takes it as a souvenir.Later that night, Vladimir stops at a small rural cottage to ask for shelter. He notices several daggers hanging up on one of the walls, and a vacant space that happens to fit the one he has discovered. Vladimir is surprised by the entrance of Giorgio (Glauco Onorato), who explains that the knife belongs to his father, who has not been seen for five days. Giorgio offers a room to the young count, and subsequently introduces him to the rest of the family: his wife (Rika Dialina), their young son Ivan, Giorgio\'s younger brother Pietro (Massimo Righi), and sister Sdenka (Susy Anderson). It subsequently transpires that they are eagerly anticipating the arrival of their father, Gorcha, as well as the reason for his absence: he has gone to do battle with the outlaw and dreaded wurdalak Ali Beg. Vladimir is confused by the term, and Sdenka explains that a wurdalak is a walking cadaver who feeds on the blood of the living, preferably close friends and family members. Giorgio and Pietro are certain that the corpse Vladimir had discovered is that of Ali Beg, but also realize that there is a strong possibility that their father has been infected by the blood curse too. They warn the count to leave, but he decides to stay and await the old mans return.At the stroke of midnight, Gorcha (Boris Karloff) returns to the cottage. His sour demeanor and unkempt appearance bode the worse, and the two brothers are torn: they realize that it is their duty to kill Gorcha before he feeds on the family, but their love for him makes it difficult to reach a decision. Later that night, both Ivan and Pietro are attacked by Gorcha who drains them of blood, and then flees the cottage. Giorgio stakes and beheads Pietro to prevent him from reviving as a wurdalak. But he is prevented from doing so to Ivan when his wife threatens to commit suicide. Reluntantly, he agrees to bury the child without taking the necessary precautions.That same night, the child rises from his grave and begs to be invited into the cottage. The mother runs to her son\'s aid, stabbing Giorgio when he attempts to stop her, only to be greeted at the front door by Gorcha. The old man bits and infects his daughter-in-law, who then does the same for her husband. Vladimir and Sdenka flee from the cottage and go on the run and hide out in the ruins of an abandoned cathedral as dawn breaks. Vladimir is optimistic that a long and happy life lies with them. But Sdenka is reluctant to relinquish her family ties. She believes that she is meant to stay with the family.Sdenka\'s fears about her family are confirmed when that evening, Gorcha and her siblings show up at the abandoned Abby. As Vladimir sleeps, Sdenka is lured into their loving arms where they bite to death. Awakened by her screams, Vladimir rushes to her aid, but the family has already taken her home, forcing the lover to follow suite. The young nobleman finds her, lying motionless on her bed. Sdenka awakens, and a distinct change is visible on her face. No longer caring, Vladimir embraces her, and she bites and infects him too.THE DROP OF WATERIn Victorian London, England, Nurse Helen Chester (Jacqueline Pierreux) is called to a large house to prepare the corpse of an elderly medium for her burial. As she dressed the body, she notices an elaborate diamond ring on its finger. Tempted by greed, Nurse Chester steals it. As she does, a glass tips over, and drops of water begin to splash on the floor. She is also assailed by a fly, no doubt attracted by the odor of the body. Unsettled but pleased by her acquisition, she finishes the job and returns home to her small East End flat.After returning home, Nurse Chester is assailed by strange events. The buzzing fly returns and continues to pester her. Then the lights in her apartment go out, and the sounds of the dripping water continues with maddening regularity. She sees the old womans corpse lying on her bed, and coming towards her. The terrified woman begs for forgiveness, but she ultimately strangles herself, imaging that the medium\'s hands are gripping her throat.The next morning, the concierge (Harriet White Medin) discovers Nurse Chester\'s body and calls the police. The investigator on the scene (Gustavo de Nardo) quickly concludes that its a simple case and that Nurse Chester "died of fright". The pathologist arrives on the scene to examine the body before it\'s taken away and he notes that the only sign of violence is a small bruise on her left finger, mostly likely caused when someone pried a ring from her finger. As the doctor makes this observation, the concierge appears distressed, as she has apparently took the ring from the dead Nurse Chester, and is further distracted by the sound of a fly swooping about in the air....Boris Karloff makes a final appearance as Gorcha riding on his horse as he concludes the three tales of fear and tells the viewers to be careful while walking home at night for ghosts and vampires have no fear. The image pulls back to actually reveal him sitting on a prop fake horse with a camera crew and various crewmen moving branches around to simulate the scene of riding through the forest from the Wurdalak segment.'

Text Cleaning

Using regular expression to cleanup the texs

In [11]:
```
text = data['Synopsis']
# Replace email addresses with 'email'
processed = text.str.replace(r'^.+@[^\.].*\.[a-z]{2,}$', 'emailaddress', regex=True)

# Replace URLs with 'webaddress'
processed = processed.str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',
                                  'webaddress', regex=True)

# Replace money symbols with 'moneysymb' (£ can by typed with ALT key + 156)
processed = processed.str.replace(r'£|\$', 'moneysymb', regex=True)

# Replace 10 digit phone numbers (formats include paranthesis, spaces, no spaces, dashes) with 'phonenumber'
processed = processed.str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',
                                  'phonenumbr', regex=True)
```

```
# Replace numbers with 'numbr'
processed = processed.str.replace(r'\d+(\.\d+)?', 'numbr', regex=True)

# Removeing useless characters like whitespace, punctuation and so on
processed = processed.str.replace(r'[^\w\d\s]', ' ', regex=True)

# Replace whitespace between terms with a single space
processed = processed.str.replace(r'\s+', ' ', regex=True)

# Remove leading and trailing whitespace
processed = processed.str.replace(r'^\s+|\s+?$', '', regex=True)
```

Lower case sentence

In [12]:
```
processed = processed.str.lower()
processed
```

Out[12]:
```
0       note this synopsis is for the orginal italian ...
1       the action takes place in the years numbr numb...
2       at the somafree institute dr hal raglan humili...
3       this creepy and scary story centers around the...
4       the film opens in an anchorage motel room in n...
                              ...
1561    buck o brien mike white is a numbr year old am...
1562    american foreign news correspondent larry stan...
1563    two children jacques mayol jean marc barr and ...
1564    bernard chanticleer peter kastner called big b...
1565    petey wheatstraw rudy ray moore is born during...
Name: Synopsis, Length: 1566, dtype: object
```

Removeing stopwords from the text by using the list from NLTK.

In [13]:
```
# In case you do not have the stopwords already, you can run the below line
nltk.download('stopwords')
```

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

Out[13]:
```
True
```

In [14]:
```
from nltk.corpus import stopwords

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

processed = processed.apply(lambda x: ' '.join(term for term in x.split() if term not in stop_words))
```

Using PorterStemmer, we can extract stem of each word.

In [15]:
```
ps = nltk.PorterStemmer()

processed = processed.apply(lambda x: ' '.join(ps.stem(term) for term in x.split()))
```

In [16]:
```
processed
```

Out[16]:
```
0       note synopsi orgin italian releas segment cert...
1       action take place year numbr numbr sweden trou...
2       somafre institut dr hal raglan humili patient ...
3       creepi scari stori center around smurl famili ...
4       film open anchorag motel room numbr numbr year...
                              ...
1561    buck brien mike white numbr year old amateur p...
1562    american foreign news correspond larri stanfor...
1563    two children jacqu mayol jean marc barr enzo m...
1564    bernard chanticl peter kastner call big boy pa...
1565    petey wheatstraw rudi ray moor born great miam...
Name: Synopsis, Length: 1566, dtype: object
```

In [17]:
```
processed[0]
```

Out[17]:
```
'note synopsi orgin italian releas segment certain order bori karloff introduc three horror tale macabr supernatur known
three face fear telephonerosi michel mercier attract high price parisian call girl return spaciou basement apart even imm
edi get beset seri strang phone call caller soon identifi frank ex pimp recent escap prison rosi terrifi testimoni land m
an jail look solac rosi phone lesbian lover mari lynda alfonsi two women estrang time rosi certain one help mari agre com
e night second later frank call promis matter call protect reveng unknown rosi mari caller imperson frank marri arriv ros
i apart soon best calm rosi nerv give panic struck woman tranquil put bed later night rosi sleep mari get bed pen note co
nfess one make strang phone call learn frank escap prison know rosi would call help explain felt way come back life break
up busi write fail notic intrud apart time frank real creep behind mari strangl death one rosi nylon stock sound struggl
awaken rosi gasp fright murder pimp realiz kill wrong woman slowli make way rosi bed howev earlier night rosi place butch
er knife pillow mari suggest rosi seiz knife stab frank begin strangl rosi drop knife break hysteria surround two corps f
ormer lover wurdalakin numbrth centuri russia vladimir urf young nobleman long trip cours journey find behead corps knife
plung heart withdraw blade take souvenir later night vladimir stop small rural cottag ask shelter notic sever dagger hang
one wall vacant space happen fit one discov vladimir surpris entranc giorgio glauco onorato explain knife belong father s
een five day giorgio offer room young count subsequ introduc rest famili wife rika dialina young son ivan giorgio younger
brother pietro massimo righi sister sdenka susi anderson subsequ transpir eagerli anticip arriv father gorcha well reason
absenc gone battl outlaw dread wurdalak ali beg vladimir confus term sdenka explain wurdalak walk cadav feed blood live p
refer close friend famili member giorgio pietro certain corps vladimir discov ali beg also realiz strong possibl father i
nfect blood curs warn count leav decid stay await old man return stroke midnight gorcha bori karloff return cottag sour d
emeanor unkempt appear bode wors two brother torn realiz duti kill gorcha feed famili love make difficult reach decis lat
er night ivan pietro attack gorcha drain blood flee cottag giorgio stake behead pietro prevent reviv wurdalak prevent iva
n wife threaten commit suicid reluntantli agre buri child without take necessari precaut night child rise grave beg invit
cottag mother run son aid stab giorgio attempt stop greet front door gorcha old man bit infect daughter law husband vladi
mir sdenka flee cottag go run hide ruin abandon cathedr dawn break vladimir optimist long happi life lie sdenka reluct re
linquish famili tie believ meant stay famili sdenka fear famili confirm even gorcha sibl show abandon abbi vladimir sleep
sdenka lure love arm bite death awaken scream vladimir rush aid famili alreadi taken home forc lover follow suit young no
bleman find lie motionless bed sdenka awaken distinct chang visibl face longer care vladimir embrac bite infect drop wate
rin victorian london england nurs helen chester jacquelin pierreux call larg hous prepar corps elderli medium burial dres
s bodi notic elabor diamond ring finger tempt greed nurs chester steal glass tip drop water begin splash floor also assai
l fli doubt attract odor bodi unsettl pleas acquisit finish job return home small east end flat return home nurs chester
assail strang event buzz fli return continu pester light apart go sound drip water continu madden regular see old woman c
orps lie bed come toward terrifi woman beg forgiv ultim strangl imag medium hand grip throat next morn concierg harriet w
hite medin discov nurs chester bodi call polic investig scene gustavo de nardo quickli conclud simpl case nurs chester di
e fright pathologist arriv scene examin bodi taken away note sign violenc small bruis left finger mostli like caus someon
pri ring finger doctor make observ concierg appear distress appar took ring dead nurs chester distract sound fli swoop ai
r bori karloff make final appear gorcha ride hors conclud three tale fear tell viewer care walk home night ghost vampir f
ear imag pull back actual reveal sit prop fake hors camera crew variou crewmen move branch around simul scene ride forest
wurdalak segment'
```

In [ ]:

Feature extraction

After preprocessing, we need to extract feature from text message. To do this, it will be necessary to tokenize each words.

In case of error while runing tokenizer that you do not have the punkt package, you can run the below line.

In [18]:
```
# nltk.download('punkt')
```

In [19]:
```
# from nltk.tokenize import word_tokenize

# all_words = []

# for message in processed:
#     words = word_tokenize(message)
#     for w in words:
#         all_words.append(w)

# all_words_freq = nltk.FreqDist(all_words)

# # Print the result
# print('Number of words: {}'.format(len(all_words_freq)))
# print('Most common words: {}'.format(all_words_freq.most_common(15)))
```

In [20]:
```
# list(all_words_freq.keys())[:20]
```

Based on the below analysis length of feature vector will be around the median (2350.5). We select the lenght 2400 for simplisity.

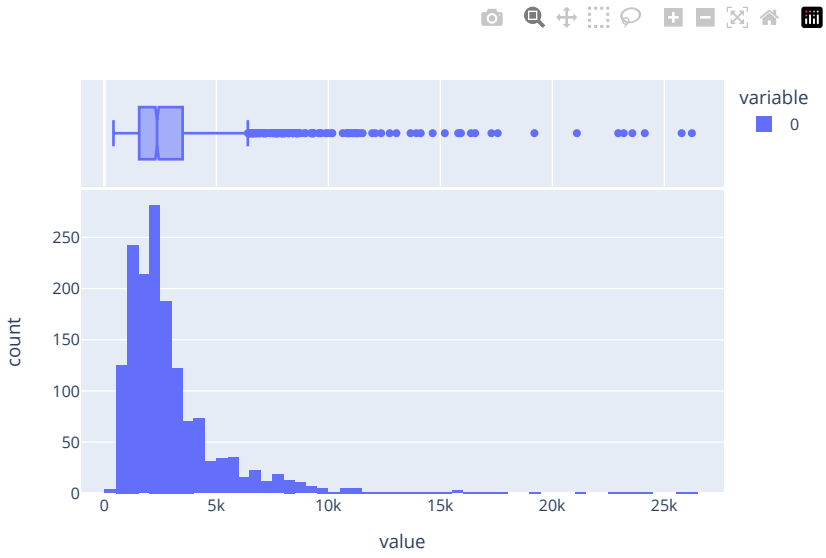
In [16]:
```
textlenght = [len(x) for x in processed]
# textlenght
app = Dash("lenght of Synopsis text analysis")

app.layout = html.Div([
    html.H4("Analysis of Synopsis text"),
    html.P("Select Distribution:"),
    dcc.RadioItems(
        id='distribution',
        options=['box', 'violin', 'rug'],
        value='box', inline=True
    ),
    dcc.Graph(id="graph"),
])
```

```
@app.callback(
    Output("graph", "figure"),
    Input("distribution", "value"))
def display_graph(distribution):
    df = px.data.tips() # replace with your own data source
    fig = px.histogram(
        textlenght,
        marginal=distribution,
)
    return fig


app.run_server(debug=True)
```

Analysis of Synopsis text

Select Distribution:

◉ box ○ violin ○ rug

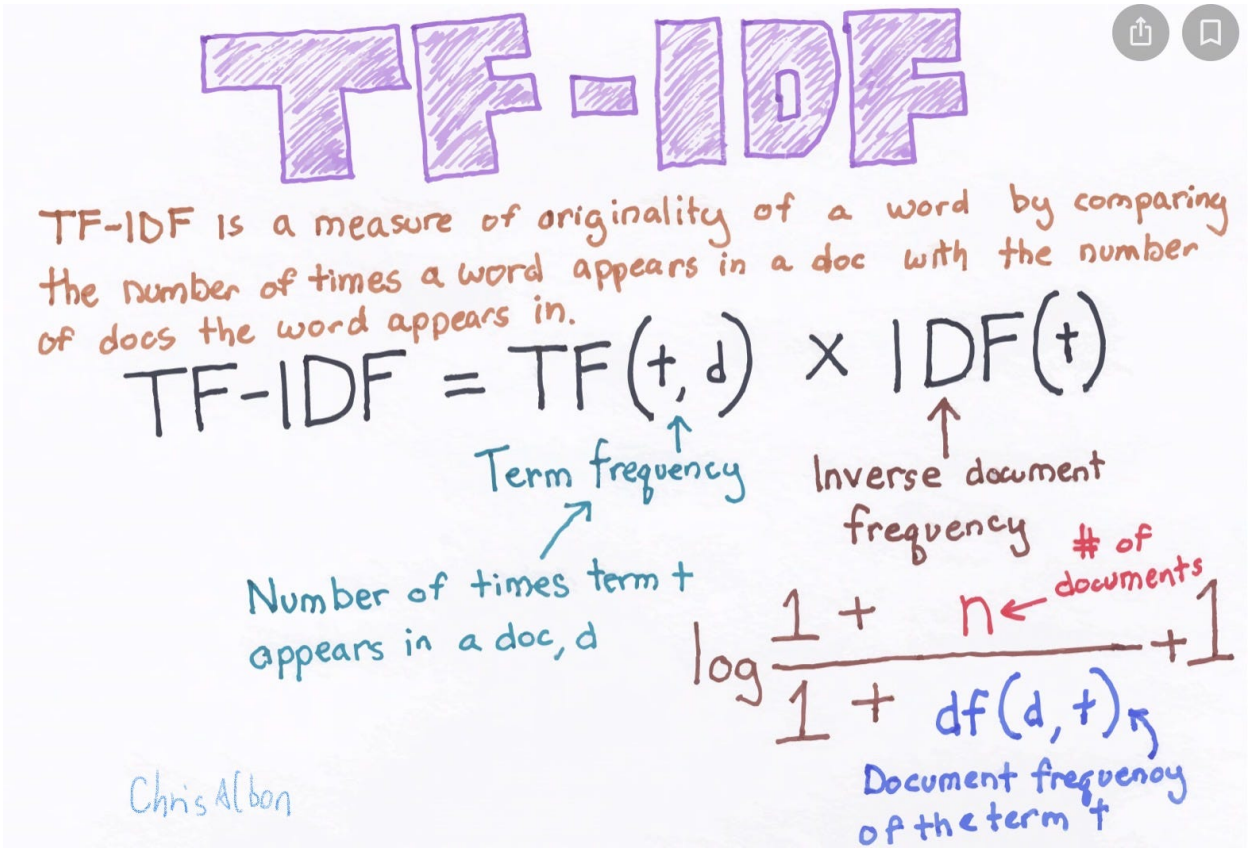


In [ ]:

Now we are ready for the modeling. We are going to use algorithms from sklearn package. We will go through the following steps:

1. Split the data into training and test sets (80% train, 20% test)
2. Extract features from the training data using TfidfVectorizer.
3. Transform the test data into the same feature vector as the training data.
4. Train the classifier
5. Evaluate the classifier

TF-IDF Vectorizer

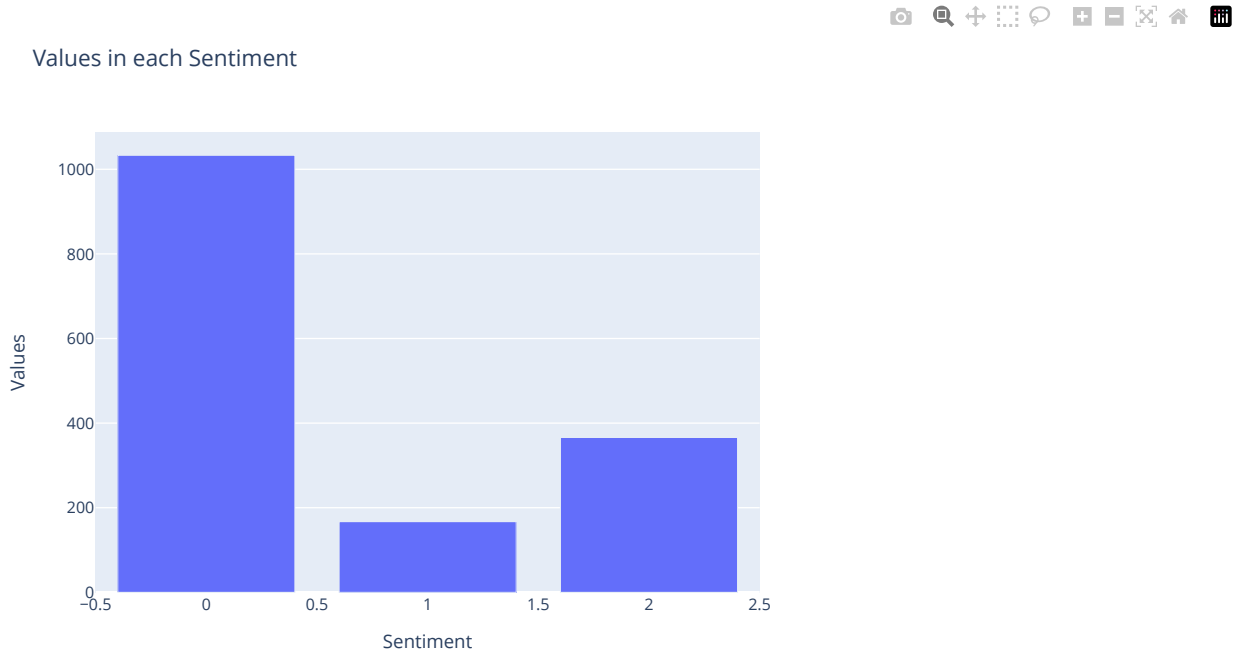


In [22]:
```
docs = list(processed)
tfidf_vectorizer = TfidfVectorizer(use_idf=True, max_features = 2400)
tfidf_vectorizer_vectors = tfidf_vectorizer.fit_transform(docs)
docs = tfidf_vectorizer_vectors.toarray()
```

In [23]:
```
X = docs
y = data['Tag']
print(X.shape, y.shape)
```

```
(1566, 2400) (1566,)
```

In [24]:
```
fig = go.Figure([go.Bar(x=y.value_counts().index, y=y.value_counts().tolist())])
fig.update_layout(
    title="Values in each Sentiment",
    xaxis_title="Sentiment",
    yaxis_title="Values")
fig.show()
```



Train-Test Split

In [27]:
```
X_train,X_test,y_train,y_test=train_test_split(X, y, test_size=0.2, stratify=y)
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(1252, 2400) (1252,)
(314, 2400) (314,)
```

Naive Bayes Classifier

Gaussian Naive Bayes

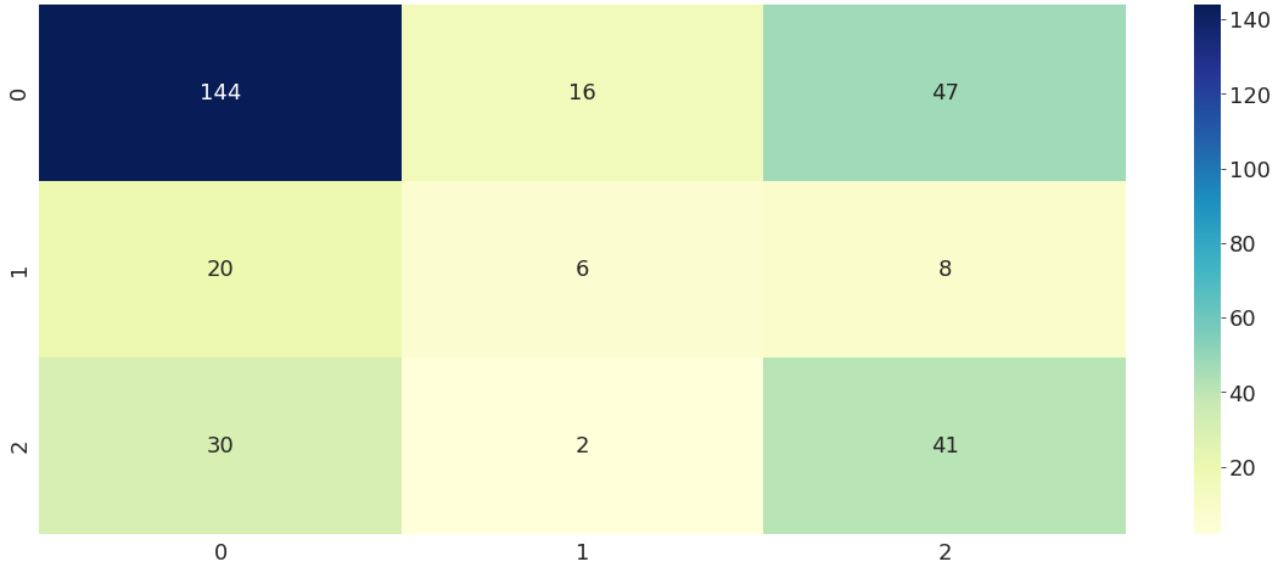
In [28]:
```
gnb = GaussianNB()
%time gnb.fit(X_train, y_train)

y_pred_train = gnb.predict(X_train)
y_pred_test = gnb.predict(X_test)
print("\nTraining Accuracy score:",accuracy_score(y_train, y_pred_train))
print("Testing Accuracy score:",accuracy_score(y_test, y_pred_test))
```

```
CPU times: user 17 ms, sys: 11.7 ms, total: 28.7 ms
Wall time: 27.6 ms

Training Accuracy score: 0.8817891373801917
Testing Accuracy score: 0.60828025477707
```

In [29]:
```
print(classification_report(y_test, y_pred_test, target_names=['Cult', 'paranormal', 'dramatic']))
```

```
              precision    recall  f1-score   support

        Cult       0.74      0.70      0.72       207
  paranormal       0.25      0.18      0.21        34
    dramatic       0.43      0.56      0.49        73

    accuracy                           0.61       314
   macro avg       0.47      0.48      0.47       314
weighted avg       0.62      0.61      0.61       314
```

In [30]:
```
cm = confusion_matrix(y_test, y_pred_test)
# print('Confusion matrix\n', cm)

cm_matrix = pd.DataFrame(data=cm)
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.show()
```



Gaussian Naive Bayes performs poorly in this case because of the prior and posterior probability condition

Multinomial Naive Bayes

In [31]:
```
mnb = MultinomialNB()
%time mnb.fit(X_train, y_train)

y_pred_train = mnb.predict(X_train)
y_pred_test = mnb.predict(X_test)
print("\nTraining Accuracy score:",accuracy_score(y_train, y_pred_train))
print("Testing Accuracy score:",accuracy_score(y_test, y_pred_test))
```

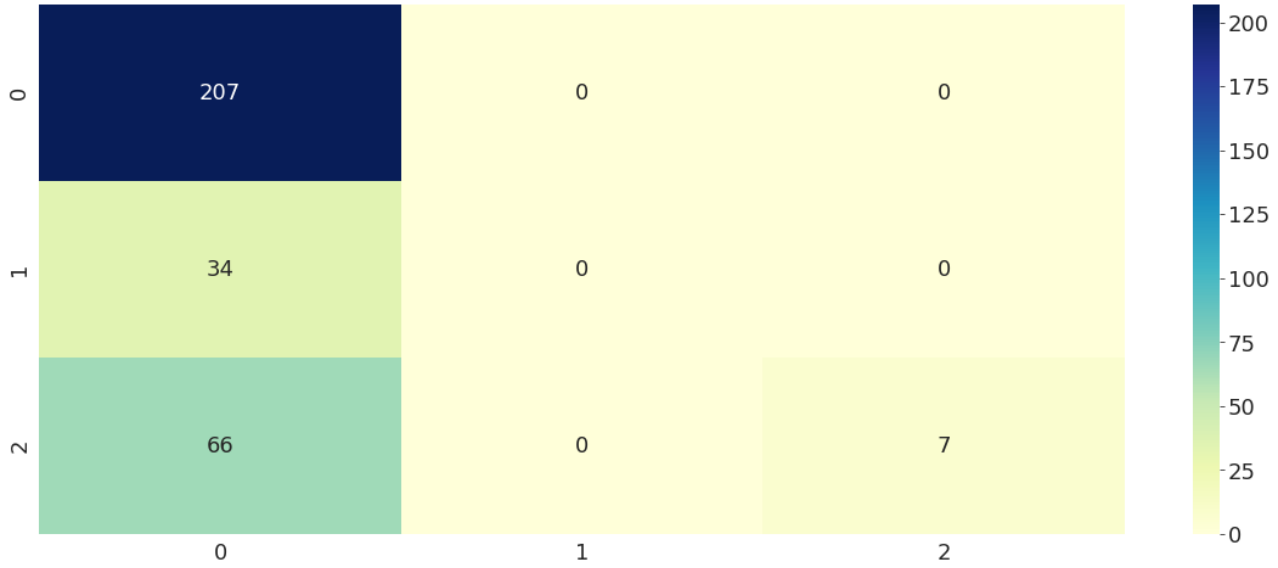
```
CPU times: user 33.6 ms, sys: 63.2 ms, total: 96.7 ms
Wall time: 20.1 ms

Training Accuracy score: 0.7204472843450479
Testing Accuracy score: 0.6815286624203821
```

In [32]:
```
print(classification_report(y_test, y_pred_test, target_names=['Cult', 'paranormal', 'dramatic']))
```

```
              precision    recall  f1-score   support

        Cult       0.67      1.00      0.81       207
  paranormal       0.00      0.00      0.00        34
    dramatic       1.00      0.10      0.17        73

    accuracy                           0.68       314
   macro avg       0.56      0.37      0.33       314
weighted avg       0.68      0.68      0.57       314
```

```
/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.
```

In [33]:
```
cm = confusion_matrix(y_test, y_pred_test)
# print('Confusion matrix\n', cm)

cm_matrix = pd.DataFrame(data=cm)
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.show()
```



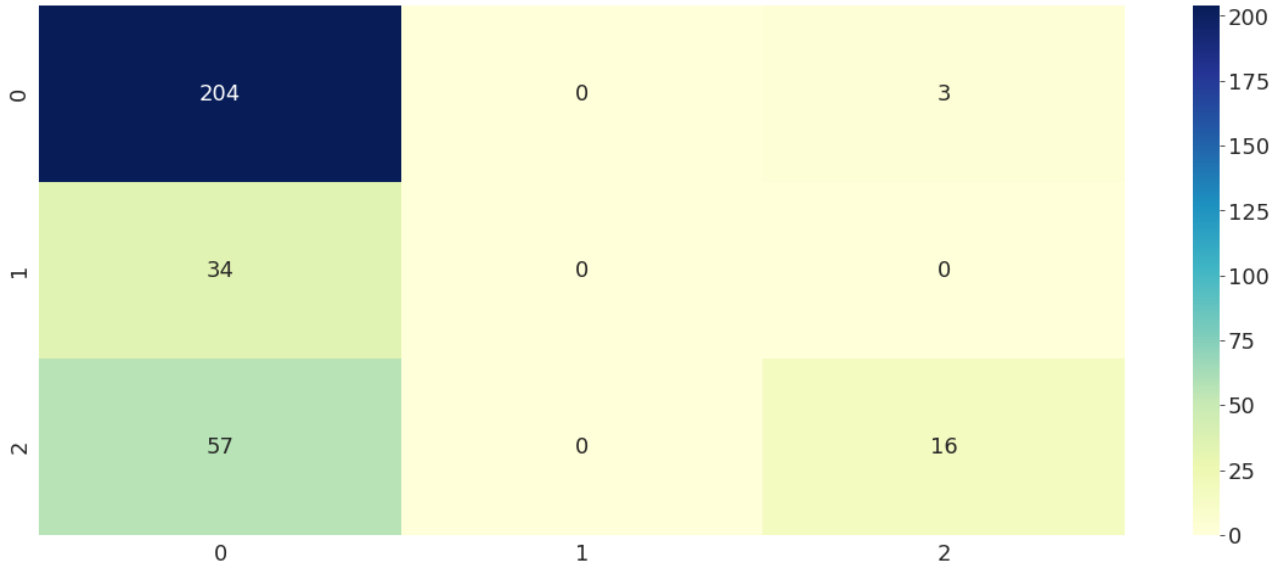
Multinomial Naive Bayes performs slightly worse than Gaussian Naive Bayes, because the size of feature vector is really big and Bayes Algorythm works better for small number of features. Let's check out results of Logistic Regression, Support Vector Machines and Decision Tree Classifier.

Logistic Regression Classifier

In [34]:
```
lr = LogisticRegression()
%time lr.fit(X_train, y_train)

y_pred_train = lr.predict(X_train)
y_pred_test = lr.predict(X_test)
print("\nTraining Accuracy score:",accuracy_score(y_train, y_pred_train))
print("Testing Accuracy score:",accuracy_score(y_test, y_pred_test))
```

```
CPU times: user 1.37 s, sys: 2.63 s, total: 4 s
Wall time: 611 ms

Training Accuracy score: 0.8075079872204473
Testing Accuracy score: 0.7006369426751592
```

In [35]:
```
cm = confusion_matrix(y_test, y_pred_test)
#print('Confusion matrix\n', cm)

cm_matrix = pd.DataFrame(data=cm)
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.show()
```



In [36]:
```
print(classification_report(y_test, y_pred_test, target_names=['Cult', 'paranormal', 'dramatic']))
```

```
              precision    recall  f1-score   support

        Cult       0.69      0.99      0.81       207
  paranormal       0.00      0.00      0.00        34
    dramatic       0.84      0.22      0.35        73

    accuracy                           0.70       314
   macro avg       0.51      0.40      0.39       314
weighted avg       0.65      0.70      0.62       314
```

```
/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.
```

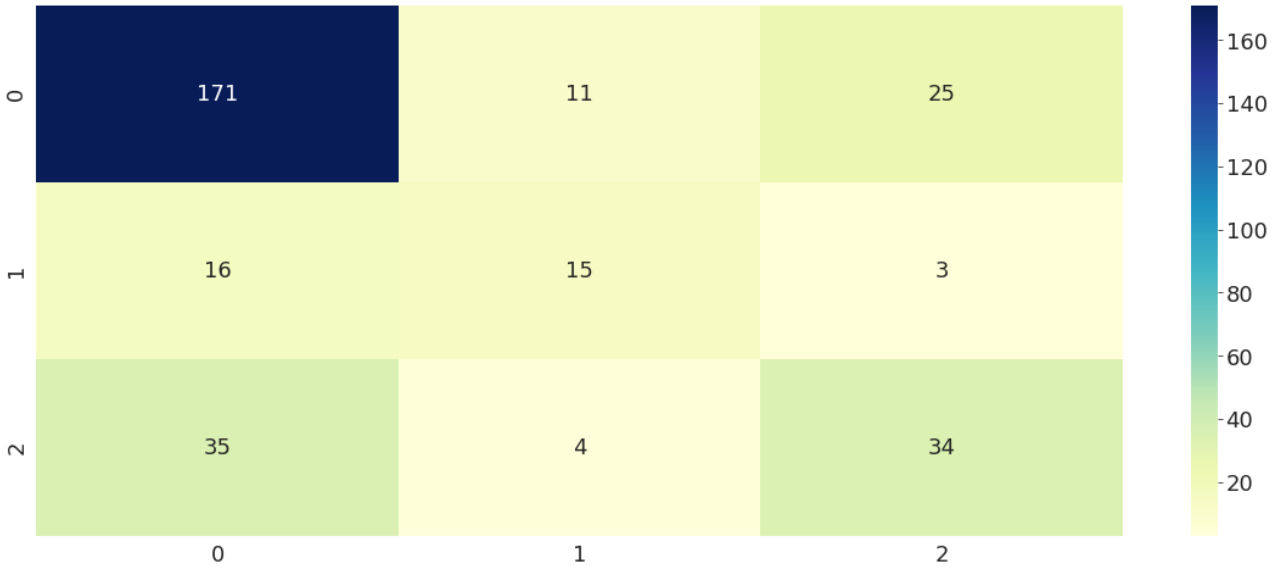
Support Vector Machines

In [37]:
```
svc =  LinearSVC(class_weight='balanced')
%time svc.fit(X_train, y_train)

y_pred_train = svc.predict(X_train)
y_pred_test = svc.predict(X_test)
print("\nTraining Accuracy score:",accuracy_score(y_train, y_pred_train))
print("Testing Accuracy score:",accuracy_score(y_test, y_pred_test))
```

```
CPU times: user 79.4 ms, sys: 3.76 ms, total: 83.2 ms
Wall time: 79.2 ms

Training Accuracy score: 0.9864217252396166
Testing Accuracy score: 0.7006369426751592
```

```
/home/amin/.local/lib/python3.8/site-packages/sklearn/svm/_classes.py:32: FutureWarning:

The default value of `dual` will change from `True` to `'auto'` in 1.5. Set the value of `dual` explicitly to suppress th
e warning.
```

In [38]:
```
cm = confusion_matrix(y_test, y_pred_test)
# print('Confusion matrix\n', cm)

cm_matrix = pd.DataFrame(data=cm)
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.show()
```



In [39]:
```
print(classification_report(y_test, y_pred_test, target_names=['Cult', 'paranormal', 'dramatic']))
```

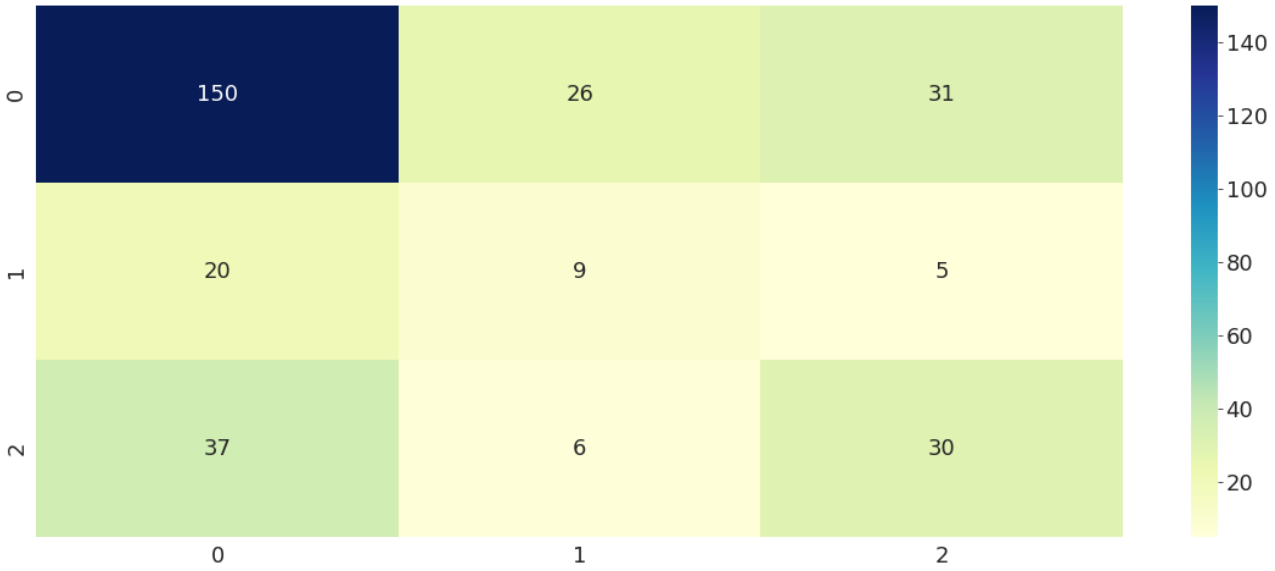
```
              precision    recall  f1-score   support

        Cult       0.77      0.83      0.80       207
  paranormal       0.50      0.44      0.47        34
    dramatic       0.55      0.47      0.50        73

    accuracy                           0.70       314
   macro avg       0.61      0.58      0.59       314
weighted avg       0.69      0.70      0.69       314
```

Decision Tree Classifier

In [40]:
```
dt = DecisionTreeClassifier()
%time dt.fit(X_train, y_train)

y_pred_train = dt.predict(X_train)
y_pred_test = dt.predict(X_test)
print("\nTraining Accuracy score:",accuracy_score(y_train, y_pred_train))
print("Testing Accuracy score:",accuracy_score(y_test, y_pred_test))
```

```
CPU times: user 959 ms, sys: 7.09 ms, total: 966 ms
Wall time: 957 ms

Training Accuracy score: 0.9960063897763578
Testing Accuracy score: 0.6019108280254777
```

In [41]:
```
cm = confusion_matrix(y_test, y_pred_test)
# print('Confusion matrix\n', cm)

cm_matrix = pd.DataFrame(data=cm, )
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.show()
```



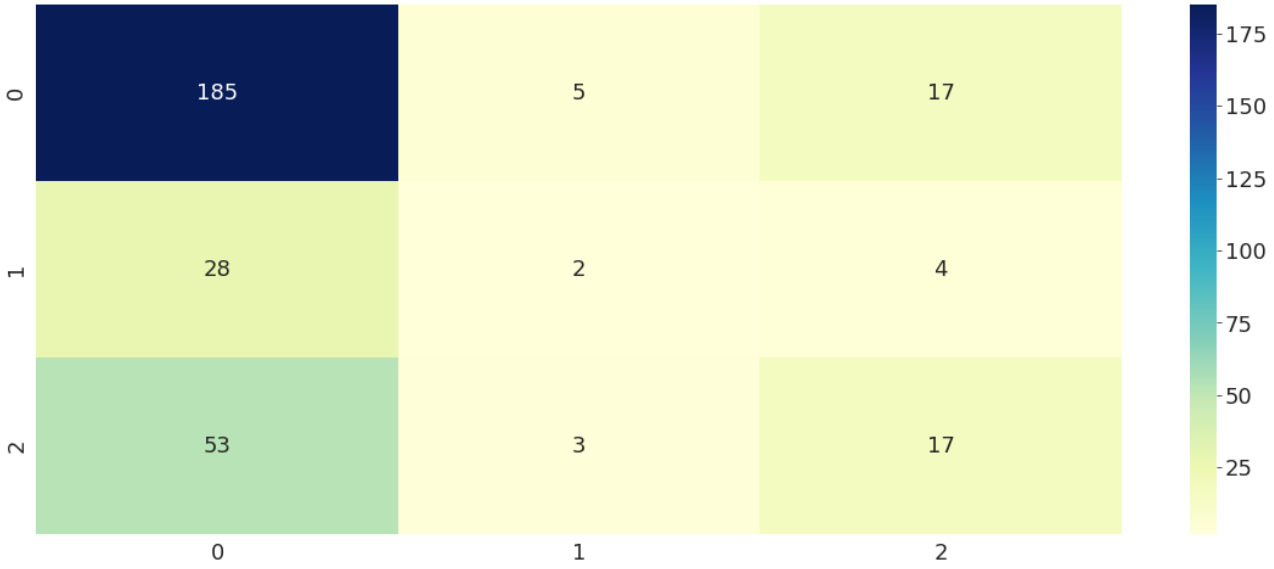
In [42]:
```
print(classification_report(y_test, y_pred_test, target_names=['Cult', 'paranormal', 'dramatic']))
```

```
              precision    recall  f1-score   support

        Cult       0.72      0.72      0.72       207
  paranormal       0.22      0.26      0.24        34
    dramatic       0.45      0.41      0.43        73

    accuracy                           0.60       314
   macro avg       0.47      0.47      0.47       314
weighted avg       0.61      0.60      0.60       314
```

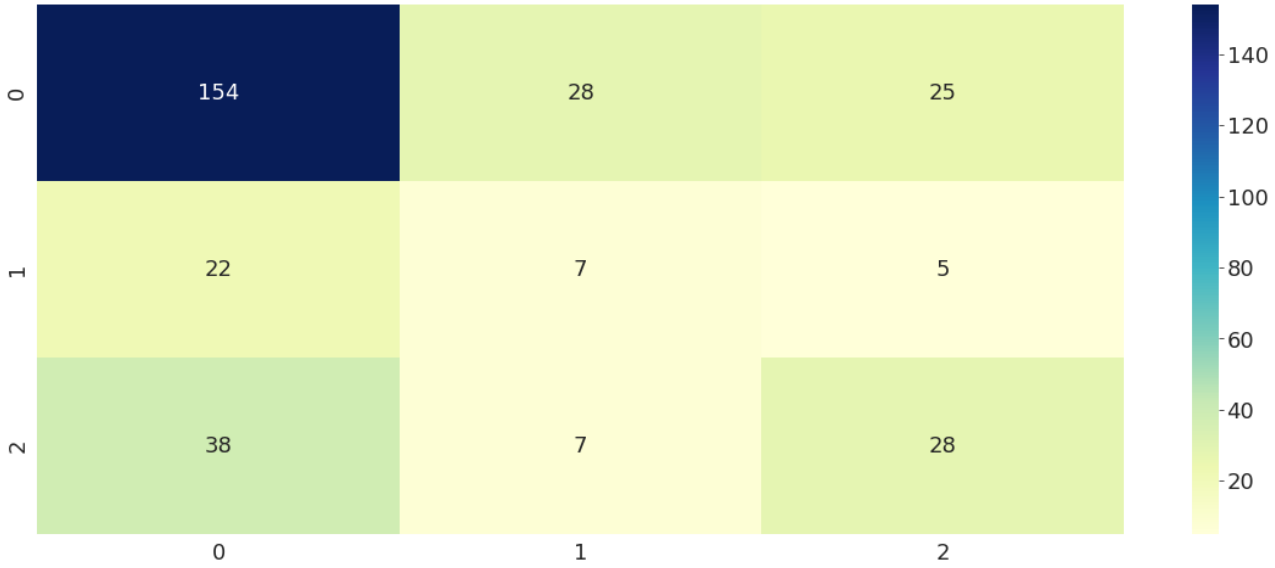
In [43]:
```
names = ['K Nearest Neighbors', 'Decision Tree', 'Random Forest', 'Logistic Regression', 'SGD Classifier',
         'Naive Bayes', 'Support Vector Classifier']

classifiers = [
    KNeighborsClassifier(),
    DecisionTreeClassifier(),
    RandomForestClassifier(),
    LogisticRegression(),
    SGDClassifier(max_iter=100),
    MultinomialNB(),
```

```
    SVC(kernel='linear')
]

models = zip(names, classifiers)

for name, model in models:
    nltk_model = SklearnClassifier(model)
    model.fit(X_train, y_train)
    y_pred_train= model.predict(X_train)
    y_pred_test = model.predict(X_test)
    print("################\n### %s\n################"%name)
    print("Training Accuracy score:",accuracy_score(y_train, y_pred_train))
    print("Testing Accuracy score:",accuracy_score(y_test, y_pred_test))
    cm = confusion_matrix(y_test, y_pred_test)

    cm_matrix = pd.DataFrame(data=cm, )
    sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
    plt.show()
    print(classification_report(y_test, y_pred_test, target_names=['Cult', 'paranormal', 'dramatic']))
```

```
################
### K Nearest Neighbors
################
Training Accuracy score: 0.7268370607028753
Testing Accuracy score: 0.6496815286624203
```

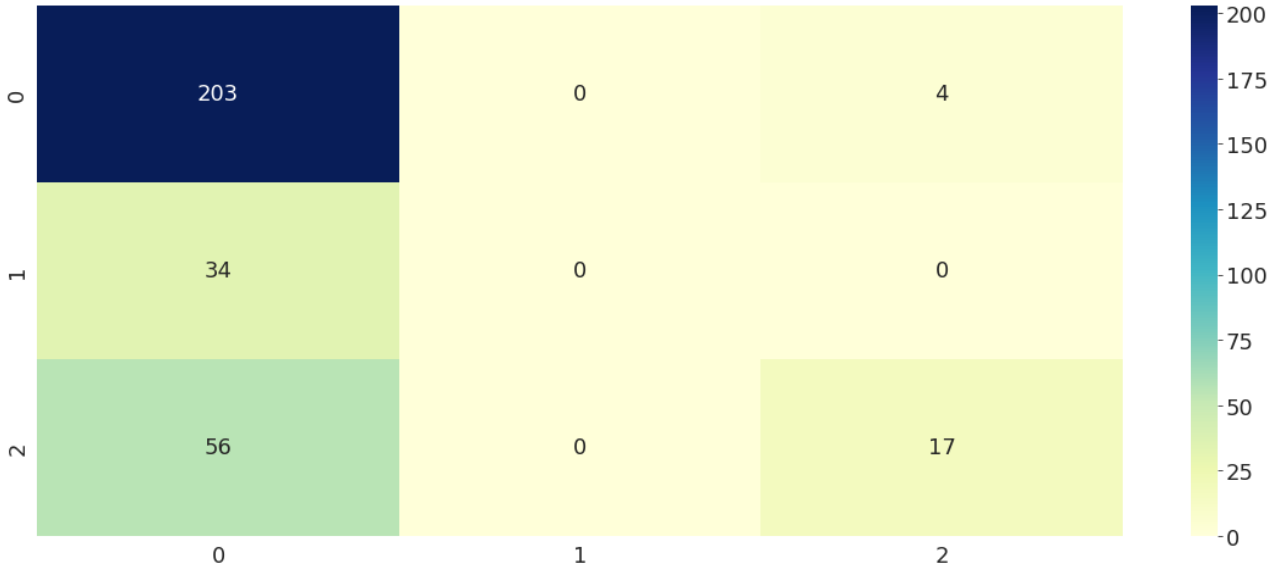



```
              precision    recall  f1-score   support

        Cult       0.70      0.89      0.78       207
  paranormal       0.20      0.06      0.09        34
    dramatic       0.45      0.23      0.31        73

    accuracy                           0.65       314
   macro avg       0.45      0.40      0.39       314
weighted avg       0.58      0.65      0.60       314

################
### Decision Tree
################
Training Accuracy score: 0.9960063897763578
Testing Accuracy score: 0.6019108280254777
```



	precision	recall	f1-score	support
Cult	0.72	0.74	0.73	207
paranormal	0.17	0.21	0.18	34
dramatic	0.48	0.38	0.43	73
accuracy			0.60	314
macro avg	0.46	0.44	0.45	314
weighted avg	0.60	0.60	0.60	314

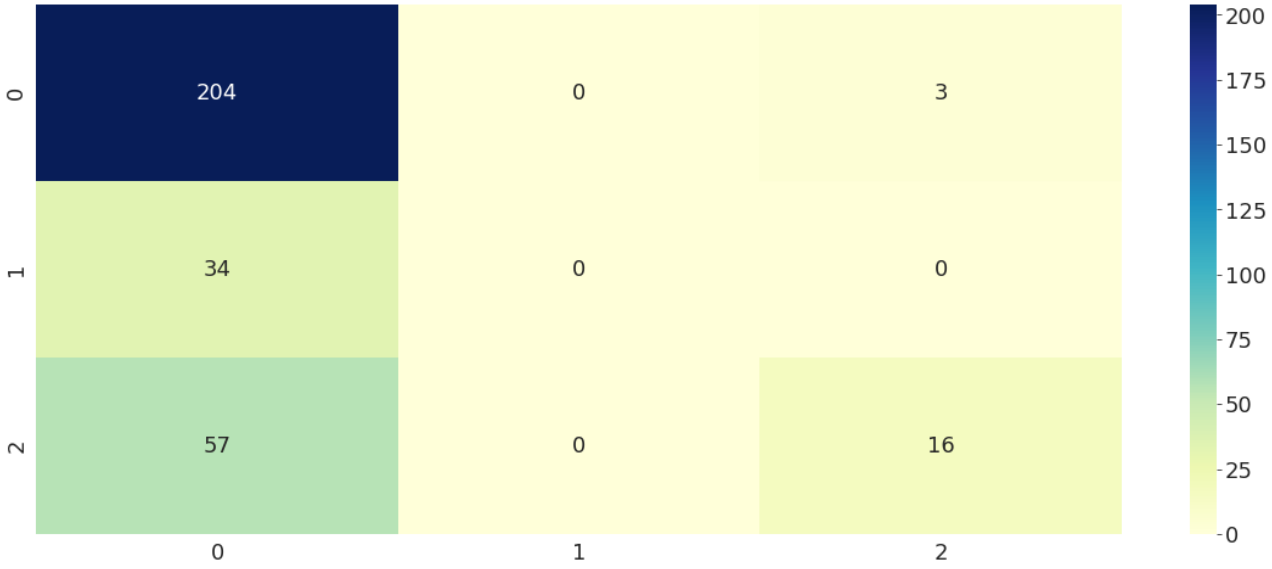
```
################
### Random Forest
################
Training Accuracy score: 0.9960063897763578
Testing Accuracy score: 0.7006369426751592
```



	precision	recall	f1-score	support
Cult	0.69	0.98	0.81	207
paranormal	0.00	0.00	0.00	34
dramatic	0.81	0.23	0.36	73
accuracy			0.70	314
macro avg	0.50	0.40	0.39	314
weighted avg	0.64	0.70	0.62	314

```
/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.
```

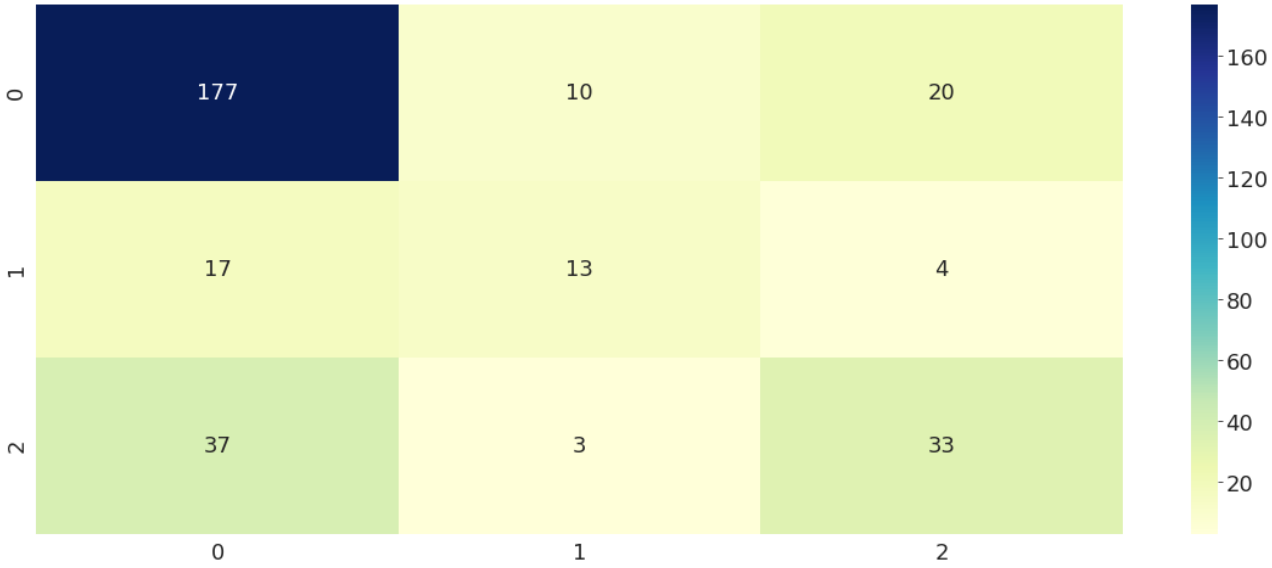
```
################
### Logistic Regression
################
Training Accuracy score: 0.8075079872204473
Testing Accuracy score: 0.7006369426751592
```



	precision	recall	f1-score	support
Cult	0.69	0.99	0.81	207
paranormal	0.00	0.00	0.00	34
dramatic	0.84	0.22	0.35	73
accuracy			0.70	314
macro avg	0.51	0.40	0.39	314
weighted avg	0.65	0.70	0.62	314

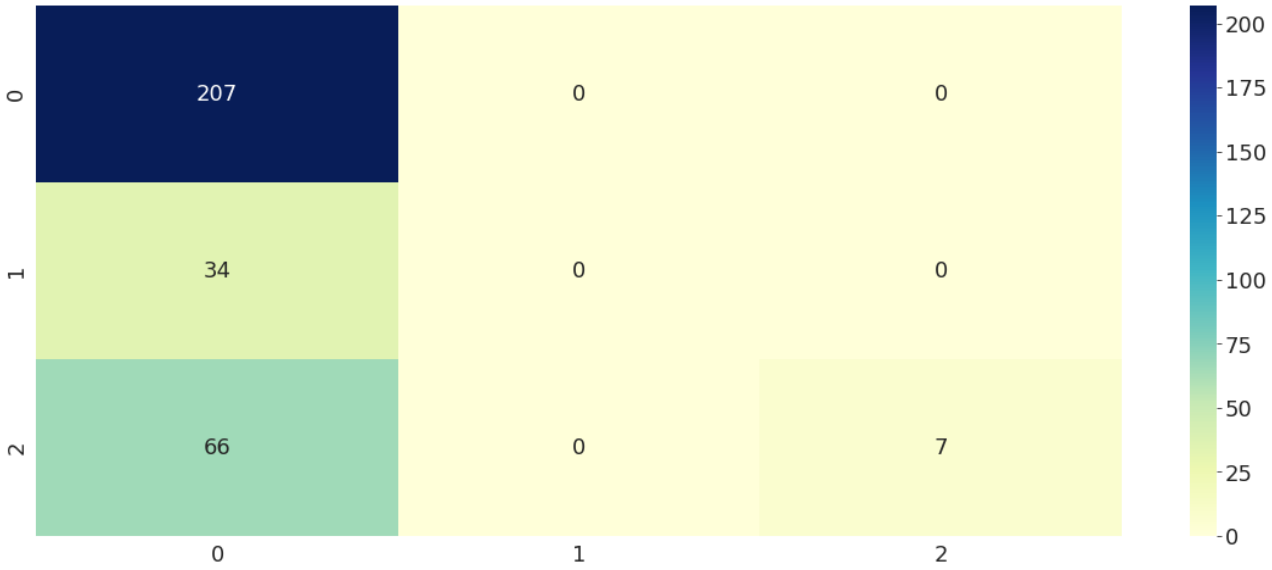
```
/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.
```

```
################
### SGD Classifier
################
Training Accuracy score: 0.994408945686901
Testing Accuracy score: 0.7101910828025477
```



	precision	recall	f1-score	support
Cult	0.77	0.86	0.81	207
paranormal	0.50	0.38	0.43	34
dramatic	0.58	0.45	0.51	73
accuracy			0.71	314
macro avg	0.62	0.56	0.58	314
weighted avg	0.69	0.71	0.70	314

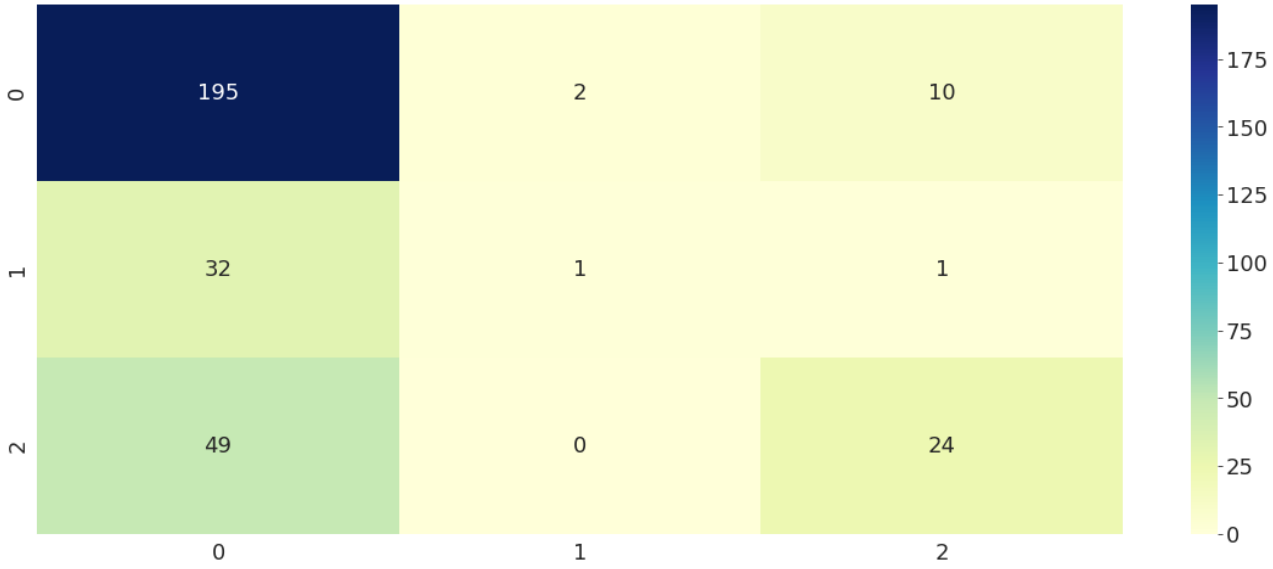
```
################
### Naive Bayes
################
Training Accuracy score: 0.7204472843450479
Testing Accuracy score: 0.6815286624203821
```



	precision	recall	f1-score	support
Cult	0.67	1.00	0.81	207
paranormal	0.00	0.00	0.00	34
dramatic	1.00	0.10	0.17	73
accuracy			0.68	314
macro avg	0.56	0.37	0.33	314
weighted avg	0.68	0.68	0.57	314

```
/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.
```

```
################
### Support Vector Classifier
################
Training Accuracy score: 0.8690095846645367
Testing Accuracy score: 0.7006369426751592
```

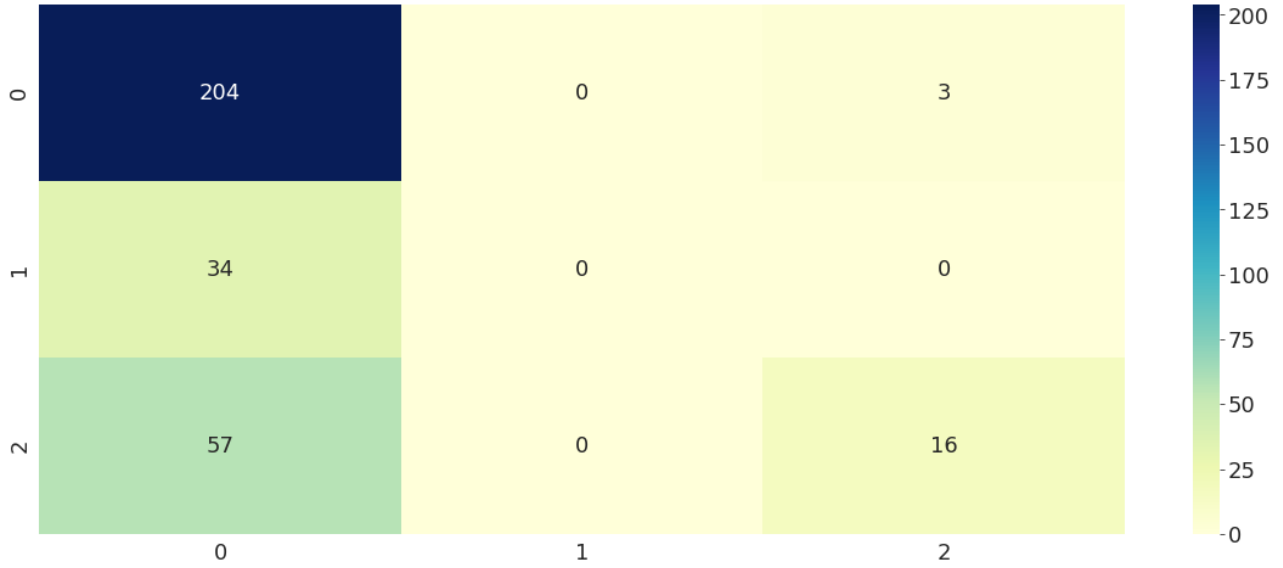


	precision	recall	f1-score	support
Cult	0.71	0.94	0.81	207
paranormal	0.33	0.03	0.05	34
dramatic	0.69	0.33	0.44	73
accuracy			0.70	314
macro avg	0.58	0.43	0.44	314
weighted avg	0.66	0.70	0.64	314

Ensembleing

In [44]:
```
models = list(zip(names, classifiers))

ensemble = VotingClassifier(estimators=models, voting='hard', n_jobs=-1)
ensemble.fit(X_train, y_train)
y_pred_train= ensemble.predict(X_train)
y_pred_test = ensemble.predict(X_test)
print("################\n### %s\n################"%'Ensemble method')
print("Training Accuracy score:",accuracy_score(y_train, y_pred_train))
print("Testing Accuracy score:",accuracy_score(y_test, y_pred_test))
cm = confusion_matrix(y_test, y_pred_test)

cm_matrix = pd.DataFrame(data=cm, )
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.show()
print(classification_report(y_test, y_pred_test, target_names=['Cult', 'paranormal', 'dramatic']))
```

```
################
### Ensemble method
################
Training Accuracy score: 0.8865814696485623
Testing Accuracy score: 0.7006369426751592
```




```
              precision    recall  f1-score   support

        Cult       0.69      0.99      0.81       207
  paranormal       0.00      0.00      0.00        34
    dramatic       0.84      0.22      0.35        73

    accuracy                           0.70       314
   macro avg       0.51      0.40      0.39       314
weighted avg       0.65      0.70      0.62       314
```

```
/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.
```

So, how do we choose whats the best? If we look at overall accuracy alone, we should be choosing the very first classifier in this notebook. However, that is also doing poorly with identifying "paranormal", and "dramatic" texts. If we choose purely based on how good it is doing with "paranormal", and "dramatic" category, we should choose the Decesion Tree or VotingClassifier (Ensembleing) we built.

In [45]:
```
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras import layers
from tensorflow.keras.layers import Embedding, Layer, Dense, Dropout, MultiHeadAttention, LayerNormalization, Input, Globa
from tensorflow.keras.layers import LSTM, Bidirectional
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
from sklearn.model_selection import train_test_split
```

In [46]:
```
y = data['Tag']
```

Train - Test Splitting (80:20)

In [47]:
```
X_train,X_test,y_train,y_test=train_test_split(processed, y, test_size=0.2, stratify=y)
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(1252,) (1252,)
(314,) (314,)
```

In [48]:
```
y_train = tf.keras.utils.to_categorical(
    y_train, num_classes=None, dtype='float32'
)
y_val = tf.keras.utils.to_categorical(
```

```
    y_test, num_classes=None, dtype='float32'
)
```

Tokenization

. Splitting sentences into words

. Finding the vocab size

In [49]:
```
max_len = 2000
oov_token = '00_V'
padding_type = 'post'
trunc_type = 'post'

tokenizer = Tokenizer()
# tokenizer.fit_on_texts(X_train)
vocab_size = len(tokenizer.word_index) + 1
print("Vocab Size: ",vocab_size)
```

```
Vocab Size:  26070
```

In [50]:
```
train_sequences = tokenizer.texts_to_sequences(X_train)
X_train = pad_sequences(train_sequences, maxlen=max_len, padding=padding_type, truncating=trunc_type)


test_sequences = tokenizer.texts_to_sequences(X_test)
X_test = pad_sequences(test_sequences, maxlen=max_len, padding=padding_type, truncating=trunc_type)
```

In [51]:
```
# X_train
```

Multi-Headed Attention

. Multi-head Attention is a module for attention mechanisms which runs through an attention mechanism several times in parallel. The independent attention outputs are then concatenated and linearly transformed into the expected dimension.

. The Self Attention mechanism (illustrated in picture above next to the picture of encoder block) is used several times in parallel in Multi-Head attention

. Multiple attention heads allows for attending to parts of the sequence differently

. During self attention a word's attention score with itself will be the highest, therefore by using mutli-head attention a word can establish its relationship with other words in the sequence by calculating the attention scores with them in parallel

In [52]:
```
class TransformerEncoder(layers.Layer):
    def __init__(self, embed_dim, heads, neurons):
        super(TransformerEncoder, self).__init__()
        self.att = layers.MultiHeadAttention(num_heads=heads, key_dim=embed_dim)
        self.ffn = Sequential(
            [layers.Dense(neurons, activation="relu"), layers.Dense(embed_dim),]
        )
        self.layernorm1 = layers.LayerNormalization(epsilon=1e-6)
        self.layernorm2 = layers.LayerNormalization(epsilon=1e-6)
        self.dropout1 = layers.Dropout(0.5)
        self.dropout2 = layers.Dropout(0.5)

    def call(self, inputs, training):
        attn_output = self.att(inputs, inputs)
        attn_output = self.dropout1(attn_output, training=training)
        out1 = self.layernorm1(inputs + attn_output)
        ffn_output = self.ffn(out1)
        ffn_output = self.dropout2(ffn_output, training=training)
        return self.layernorm2(out1 + ffn_output)

class TokenAndPositionEmbedding(layers.Layer):
    def __init__(self, maxlen, vocab_size, embed_dim):
        super(TokenAndPositionEmbedding, self).__init__()
        self.token_emb = layers.Embedding(input_dim=vocab_size, output_dim=embed_dim)
        self.pos_emb = layers.Embedding(input_dim=maxlen, output_dim=embed_dim)

    def call(self, x):
        maxlen = tf.shape(x)[-1]
        positions = tf.range(start=0, limit=maxlen, delta=1)
        positions = self.pos_emb(positions)
        x = self.token_emb(x)
        return x + positions
```

Model definition

In [65]:
```
embed_dim = 20
heads = 2
neurons = 10
# maxlen = 2000
# vocab_size = vocab_size

inputs = layers.Input(shape=(max_len,))
embedding_layer = TokenAndPositionEmbedding(max_len, vocab_size, embed_dim)
x = embedding_layer(inputs)
transformer_block = TransformerEncoder(embed_dim, heads, neurons)
x = transformer_block(x)
x = layers.GlobalAveragePooling1D()(x)
x = Dropout(0.2)(x)
outputs = layers.Dense(3, activation="sigmoid")(x)
model = Model(inputs=inputs, outputs=outputs)
```

In [66]:
```
model.compile(optimizer=tf.keras.optimizers.Adam(0.0001), loss='binary_crossentropy', metrics=['accuracy'])
model.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 2000)]            0

 token_and_position_embeddin  (None, 2000, 20)         561400
 g_1 (TokenAndPositionEmbedd
 ing)

 transformer_encoder_1 (Tran  (None, 2000, 20)         3850
 sformerEncoder)

 global_average_pooling1d_1   (None, 20)               0
 (GlobalAveragePooling1D)

 dropout_5 (Dropout)         (None, 20)                0

 dense_5 (Dense)             (None, 3)                 63

=================================================================
Total params: 565,313
Trainable params: 565,313
Non-trainable params: 0
_________________________________________________________________
```

In [67]:
```
model_name = "model.h5"
checkpoint = ModelCheckpoint(model_name,
                             monitor="val_loss",
                             mode="min",
                             save_best_only = True,
                             verbose=1)

earlystopping = EarlyStopping(monitor='val_loss',min_delta = 0.0001, patience = 1, verbose = 1)

learning_rate_reduction = ReduceLROnPlateau(monitor='val_loss',
                                            patience=3,
                                            verbose=1,
                                            factor=0.2,
                                            min_lr=0.00000001)
```

In [68]:
```
history = model.fit(X_train,y_train,
                    validation_data=(X_test,y_val),
                    epochs=25,
                    batch_size=32,
                    callbacks=[earlystopping])
```

```
Epoch 1/25
40/40 [==============================] - 50s 1s/step - loss: 0.5843 - accuracy: 0.6014 - val_loss: 0.5194 - val_accuracy:
0.6592
Epoch 2/25
40/40 [==============================] - 49s 1s/step - loss: 0.5302 - accuracy: 0.6573 - val_loss: 0.5169 - val_accuracy:
0.6592
Epoch 3/25
40/40 [==============================] - 48s 1s/step - loss: 0.5184 - accuracy: 0.6565 - val_loss: 0.5139 - val_accuracy:
0.6592
Epoch 4/25
40/40 [==============================] - 47s 1s/step - loss: 0.5182 - accuracy: 0.6597 - val_loss: 0.5142 - val_accuracy:
0.6592
Epoch 4: early stopping
```

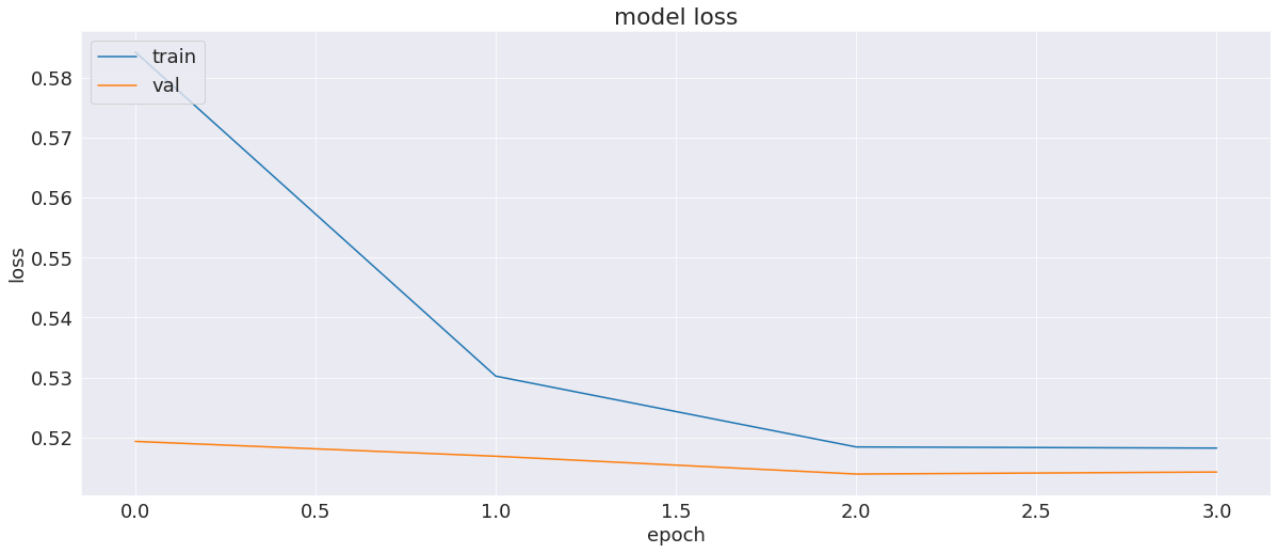
Model Evaluation

Learning Curves

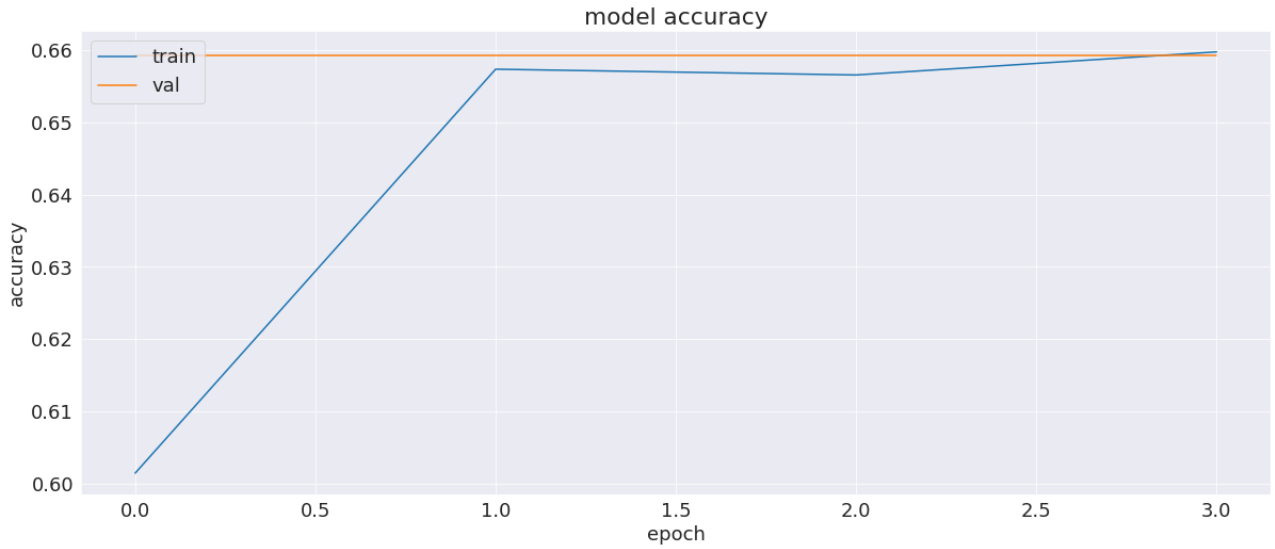
. Loss Curve

. Accuracy Curve

In [69]:
```
plt.figure(figsize=(20,8))
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```



In [70]:
```
plt.figure(figsize=(20,8))
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```



In [84]:
```
y_pred = model.predict(X_test)
cult = [int(x) for x in y_pred[:,0]>.6]
```

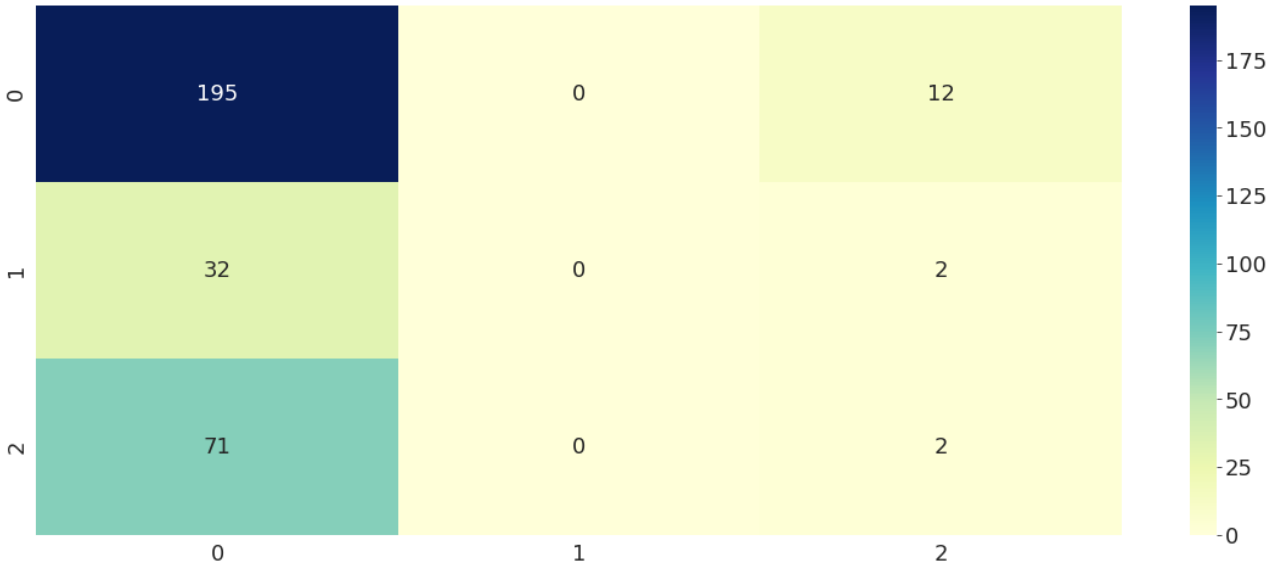
```
10/10 [==============================] - 5s 548ms/step
```

In [85]:
```
print(sum(y_pred[:,0]>.6))

y_pred1[:,0] = [int(x) for x in y_pred[:,0]>.6]
```

```
print(sum(y_pred[:,1]>.26))
y_pred1[:,1] = [int(x) for x in y_pred[:,1]>.15]

print(sum(y_pred[:,2]>.33))
y_pred1[:,2] = [int(x) for x in y_pred[:,2]>.23]

y_pred2 = [1 if x[1]==1 else 2 if x[2]==1 else 0 for x in y_pred1]
```

```
314
0
10
```

In [77]:
```
cm = confusion_matrix(y_test, y_pred2)
cm_matrix = pd.DataFrame(data=cm, )
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.show()
print(classification_report(y_test, y_pred2, target_names=['Cult', 'paranormal', 'dramatic']))
```



	precision	recall	f1-score	support
Cult	0.65	0.94	0.77	207
paranormal	0.00	0.00	0.00	34
dramatic	0.12	0.03	0.04	73
accuracy			0.63	314
macro avg	0.26	0.32	0.27	314
weighted avg	0.46	0.63	0.52	314

```
/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.

/home/amin/.local/lib/python3.8/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` param
eter to control this behavior.
```

In [74]:
```
model.save('../Model weights/transformer_weights.h5')
```

In [ ]:
```
```