



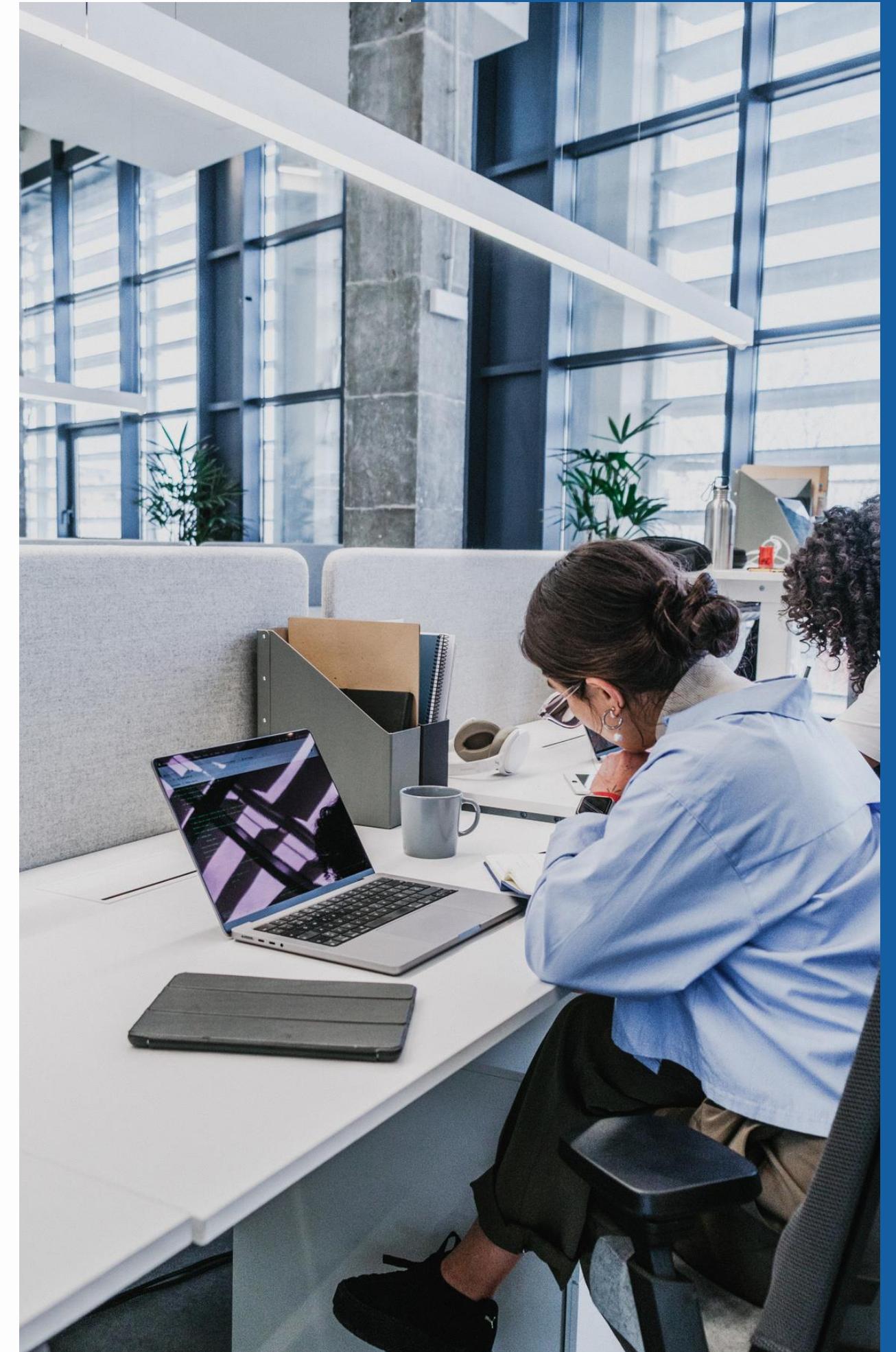
Customer Sentiment and Trend Analysis

Supervisor: Basma Reda



Overview

- Introduction
- Team Members
- Datasets
 - Yelp Review
 - IMDB Reviews
 - Twitter sentiment
 - Amazon Product Reviews
- MLFlow
- Streamlit and Azure ML Deployments
- Conclusion



Introduction

We performed sentiment analysis on ***IMDb reviews, Twitter sentiment analysis, yelp reviews, and Amazon product reviews.***

The project involves using Machine learning and deep learning models, including LSTM and Transformers models like Bert, to achieve high accuracy in predicting customer sentiment. Additionally, we implemented platforms like HuggingFace, MLFlow, and Streamlit for model optimization, tracking, and deployment



Team Members

- Mohamed Mostafa Abdelhamed
- Moaz Mohamed Tawfik
- Mohamed Talaat Abo Elftouh
- Mohamed Alaa Elsayad
- Amr Khaled Mostafa
- Mahmoud Mohammed Abdelmawgoud

Datasets

Amazon Product Reviews

Total Records: 568,454

Available Fields: Id, ProductId, UserId, ProfileName, HelpfulnessNumerator, HelpfulnessDenominator, Score, Time, Summary, Text

Yelp Review

Number of rows: 700,000
The Yelp reviews dataset consists of reviews from Yelp. It is extracted from the Yelp Dataset Challenge 2015 data.

IMDB Reviews

Total Records: 50K

This is a dataset for binary sentiment classification containing substantially more data than previous benchmark datasets. We provide a set of 25,000 highly polar movie reviews for training and 25,000 for testing.

Twitter sentiment

This is the sentiment140 dataset. It contains **1,600,000 tweets** extracted using the twitter api . The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment .

Yelp Review

The Yelp reviews dataset consists of reviews from Yelp.
It is extracted from the Yelp Dataset Challenge 2015 data.

Data Fields

- 'text': The review texts are escaped using double quotes ("), and any internal double quote is escaped by 2 double quotes (""").
New lines are escaped by a backslash followed with an "n" character, that is "\n".
- 'label': Corresponds to the score associated with the review (between 1 and 5).

Data Splits

The Yelp reviews full-star dataset is constructed by randomly taking 130,000 training samples and 10,000 testing samples for each review star from 1 to 5. In total, there are 650,000 training samples and 50,000 testing samples.

	label	text
0	4	dr. goldberg offers everything i look for in a...
1	1	Unfortunately, the frustration of being Dr. Go...
2	3	Been going to Dr. Goldberg for over 10 years. ...
3	3	Got a letter in the mail last week that said D...
4	0	I don't know what Dr. Goldberg was like before...
...
649995	4	I had a sprinkler that was gushing... pipe bro...
649996	0	Phone calls always go to voicemail and message...
649997	0	Looks like all of the good reviews have gone t...
649998	4	I was able to once again rely on Yelp to provi...
649999	0	I have been using this company for 11 months. ...

650000 rows × 2 columns

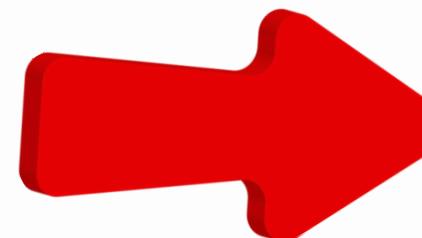
Yelp Review

To convert Yelp ratings into categorical values like "Positive," "Negative," and "Neutral," you could define rules based on the rating scale (often from 0 to 4). Here's a common way to categorize these ratings:

- Ratings 3 and 4: Positive
- Rating 2: Neutral
- Ratings 0 and 1: Negative

	label	text
0	4	dr. goldberg offers everything i look for in a...
1	1	Unfortunately, the frustration of being Dr. Go...
2	3	Been going to Dr. Goldberg for over 10 years. ...
3	3	Got a letter in the mail last week that said D...
4	0	I don't know what Dr. Goldberg was like before...
...
649995	4	I had a sprinkler that was gushing... pipe bro...
649996	0	Phone calls always go to voicemail and message...
649997	0	Looks like all of the good reviews have gone t...
649998	4	I was able to once again rely on Yelp to provi...
649999	0	I have been using this company for 11 months. ...

650000 rows × 2 columns

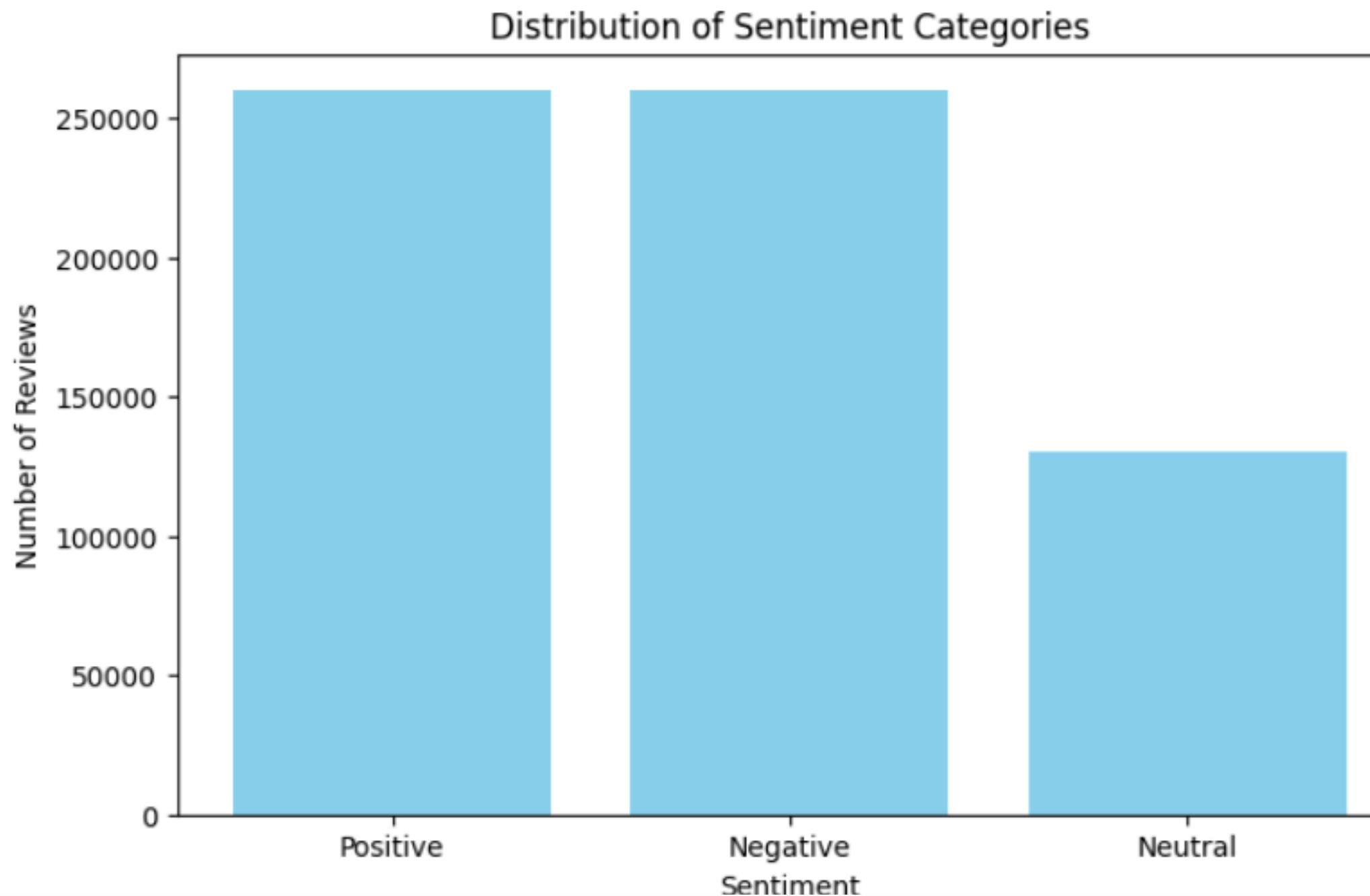


	text	sentiment
0	dr. goldberg offers everything i look for in a...	Positive
1	Unfortunately, the frustration of being Dr. Go...	Negative
2	Been going to Dr. Goldberg for over 10 years. ...	Positive
3	Got a letter in the mail last week that said D...	Positive
4	I don't know what Dr. Goldberg was like before...	Negative
...
649995	I had a sprinkler that was gushing... pipe bro...	Positive
649996	Phone calls always go to voicemail and message...	Negative
649997	Looks like all of the good reviews have gone t...	Negative
649998	I was able to once again rely on Yelp to provi...	Positive
649999	I have been using this company for 11 months. ...	Negative

650000 rows × 2 columns

Yelp Review

visualize the distribution of reviews (Positive, Neutral, Negative)

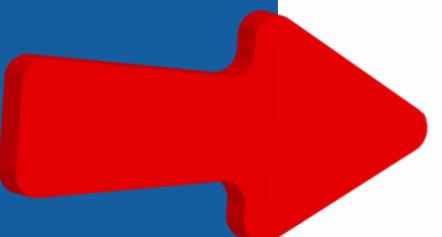


Preprocessing and Cleaning Data

Using nltk and regular expression

	text	sentiment
0	dr. goldberg offers everything i look for in a...	Positive
1	Unfortunately, the frustration of being Dr. Go...	Negative
2	Been going to Dr. Goldberg for over 10 years. ...	Positive
3	Got a letter in the mail last week that said D...	Positive
4	I don't know what Dr. Goldberg was like before...	Negative
...
649995	I had a sprinkler that was gushing... pipe bro...	Positive
649996	Phone calls always go to voicemail and message...	Negative
649997	Looks like all of the good reviews have gone t...	Negative
649998	I was able to once again rely on Yelp to provi...	Positive
649999	I have been using this company for 11 months. ...	Negative

650000 rows × 2 columns



	text	sentiment
0	dr goldberg offers everything look general pra...	Positive
1	unfortunately frustration dr goldberg patient ...	Negative
2	going dr goldberg years think one st patients ...	Positive
3	got letter mail last week said dr goldberg mov...	Positive
4	know dr goldberg like moving arizona let tell ...	Negative
...
649995	sprinkler gushing pipe broken way ground turne...	Positive
649996	phone calls always go voicemail messages return...	Negative
649997	looks like good reviews gone head place jason ...	Negative
649998	able rely yelp provide needed response leaking...	Positive
649999	using company months ryan would come every wee...	Negative

650000 rows × 2 columns

Yelp Review

Process The Dataset

```
new_data.isnull().sum()  
0  
text      48  
sentiment  0  
dtype: int64  
new_data.duplicated().sum()  
590
```

	text	sentiment
0	dr goldberg offers everything look general pra...	Positive
1	unfortunately frustration dr goldberg patient ...	Negative
2	going dr goldberg years think one st patients ...	Positive
3	got letter mail last week said dr goldberg mov...	Positive
4	know dr goldberg like moving arizona let tell ...	Negative
...
649995	sprinkler gushing pipe broken way ground turne...	Positive
649996	phone calls always go voicemail messages return...	Negative
649997	looks like good reviews gone head place jason ...	Negative
649998	able rely yelp provide needed response leaking...	Positive
649999	using company months ryan would come every wee...	Negative

649407 rows × 2 columns

usage to take 40,000 samples from each class

```
sentiment  
Positive    40000  
Negative   40000  
Neutral    40000  
Name: count, dtype: int64
```

	text	sentiment
417633	bestie recently bought jeep wanted go mini roa...	Positive
488716	charlotte wake get face food halcyon amazing s...	Positive
122370	husband found engagement ring years ago shoppi...	Positive
294942	perfect mocha yesterday special whole milk cho...	Positive
108228	never realy go always shop fry redbox used mon...	Positive

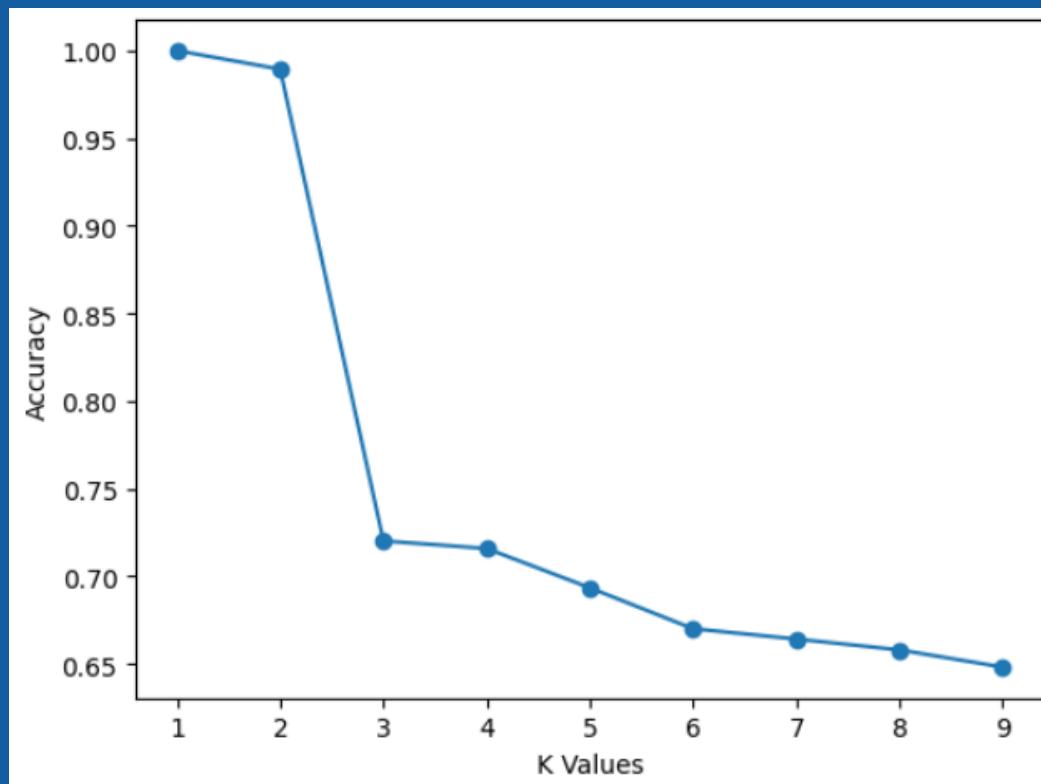
516943	went yesterday place wanted go closed handy dr...	Neutral
140162	stopped something different looked menu online...	Neutral
41391	sticker shock n nthree us ate really enjoyed s...	Neutral
195385	cute french setting turkey despite adding grav...	Neutral
390601	ak chin bless n nit sneeze throat clear new na...	Neutral

120000 rows × 2 columns

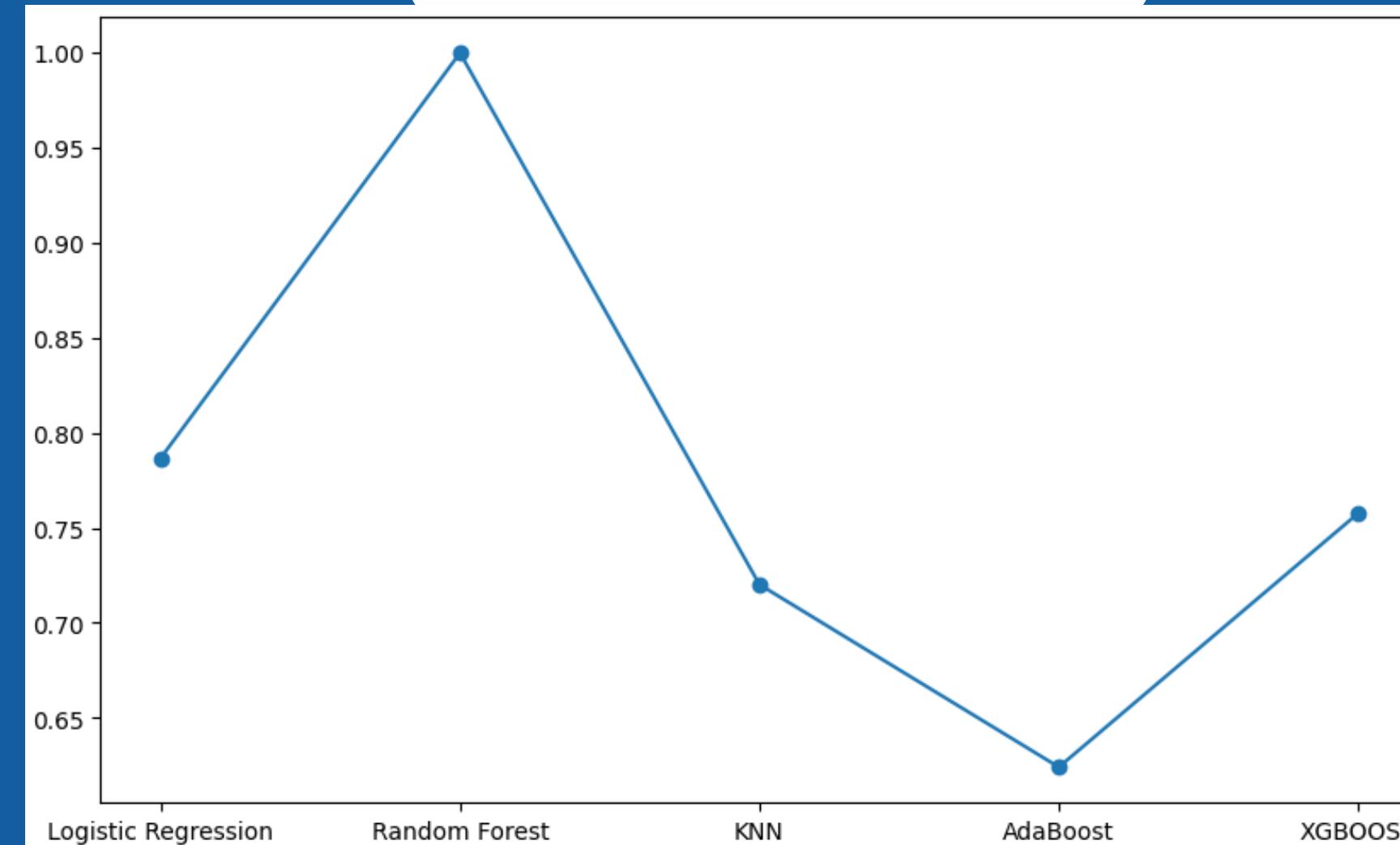
Machine Learning models

Use Cleaning Data and Use TF-IDF

KNN



Model Test Scores		
0	Logistic Regression	0.786700
1	Random Forest	0.999933
2	KNN	0.720200
3	AdaBoost	0.624133
4	XGBOOST	0.757733



Deep Learning Model

- Apply LabelEncoder on "sentiment" Column
- Tokenize The Text

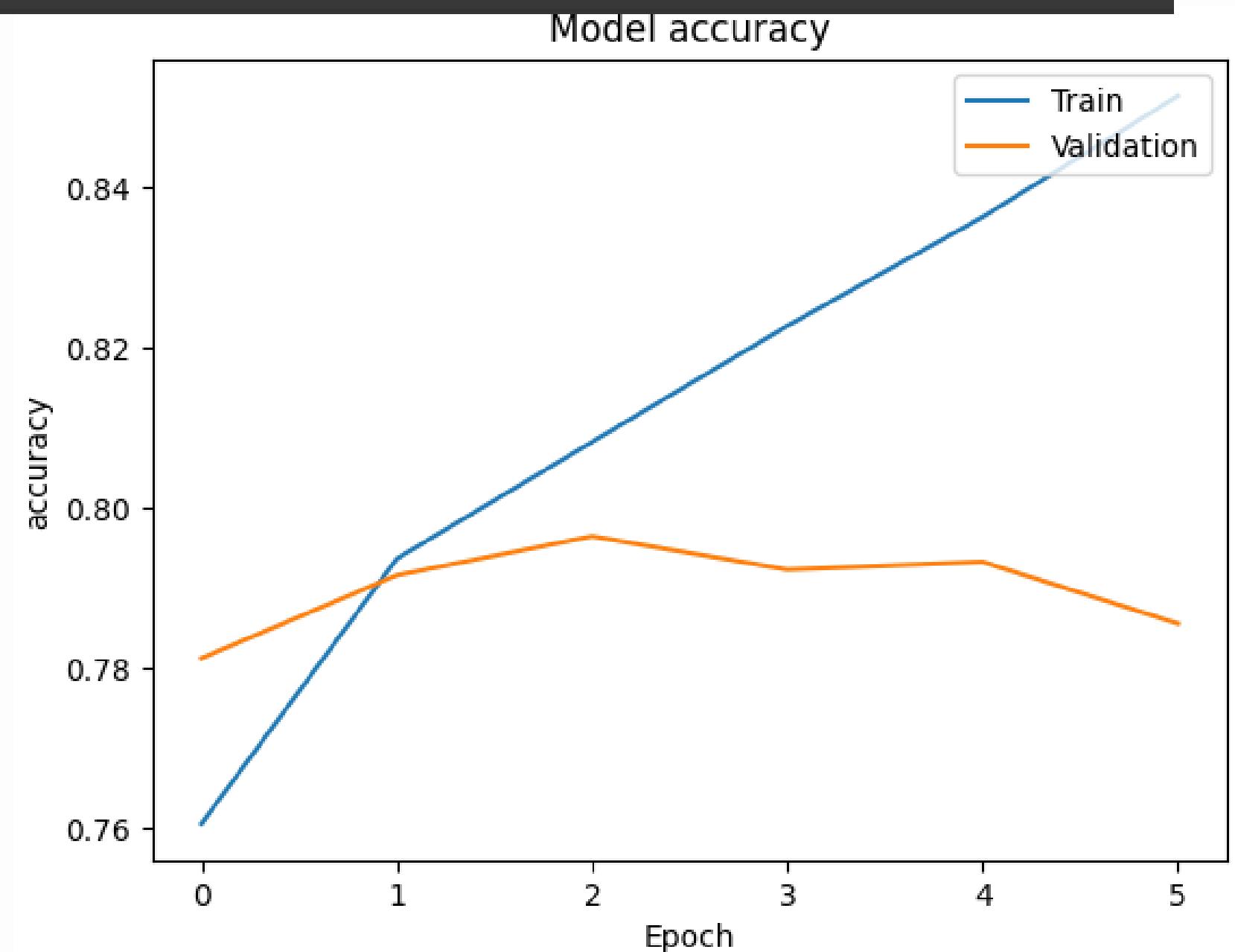
→ The 10 most repeated words are: ['n', 'food', 'good', 'place', 'like', 'get', 'one', 'time', 'would', 'great']

Build The LSTM Model

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	?	0 (unbuilt)
bidirectional (Bidirectional)	?	0 (unbuilt)
dropout (Dropout)	?	0 (unbuilt)
bidirectional_1 (Bidirectional)	?	0 (unbuilt)
dropout_1 (Dropout)	?	0 (unbuilt)
dense (Dense)	?	0 (unbuilt)
dense_1 (Dense)	?	0 (unbuilt)

Total params: 0 (0.00 B)
Trainable params: 0 (0.00 B)
Non-trainable params: 0 (0.00 B)

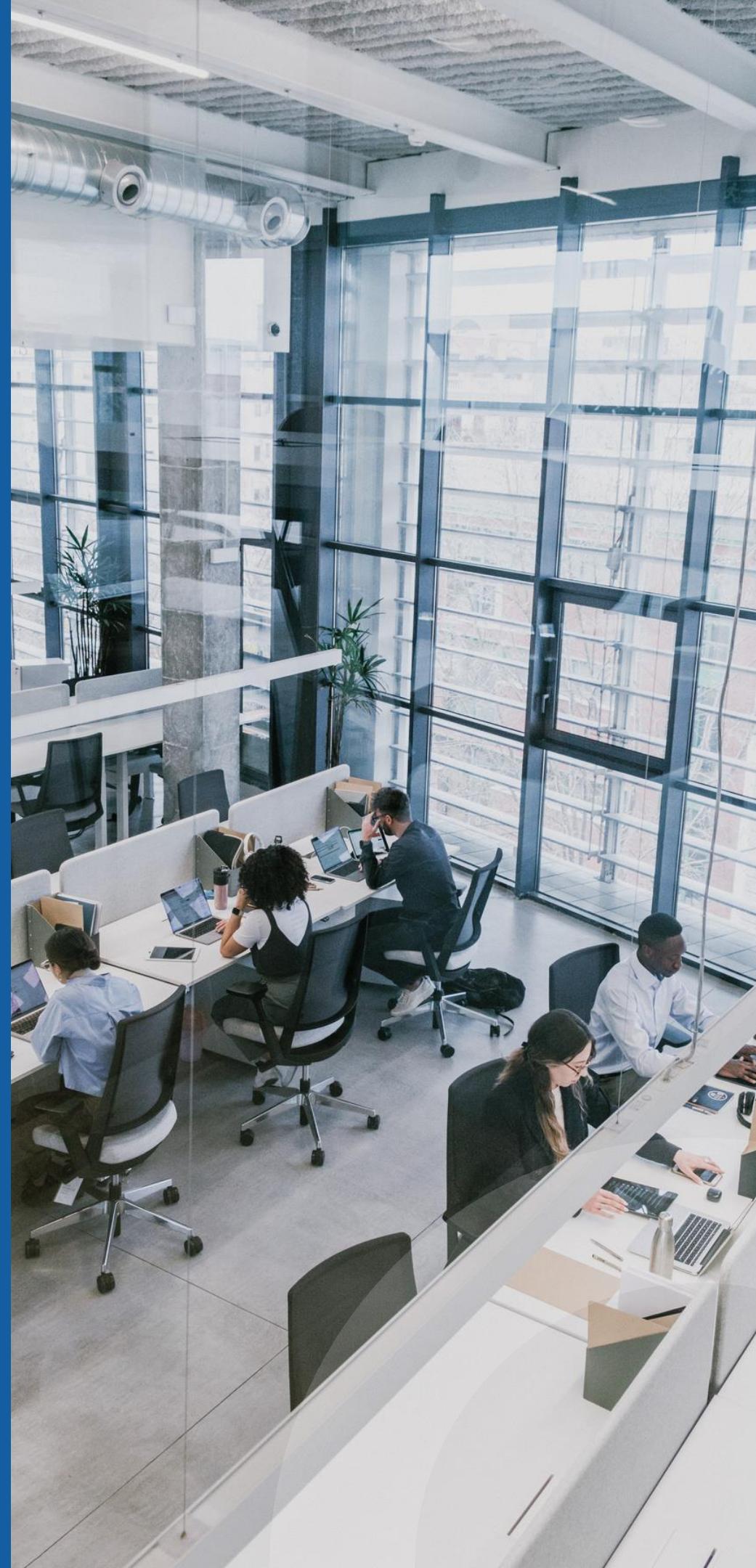


We can See From This Plot The Overfitting

IMDB Reviews

This is a dataset for binary sentiment classification containing substantially more data than previous benchmark datasets. We provide a set of 25,000 highly polar movie reviews for training and 25,000 for testing. So, predict the number of positive and negative reviews using either classification or deep learning algorithms.

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive



Importing Dataset & Preprocessing

```
print('Number Of Duplications is :', df.duplicated().sum())
df.drop_duplicates(inplace=True)
```

```
Number Of Duplications is : 418
```

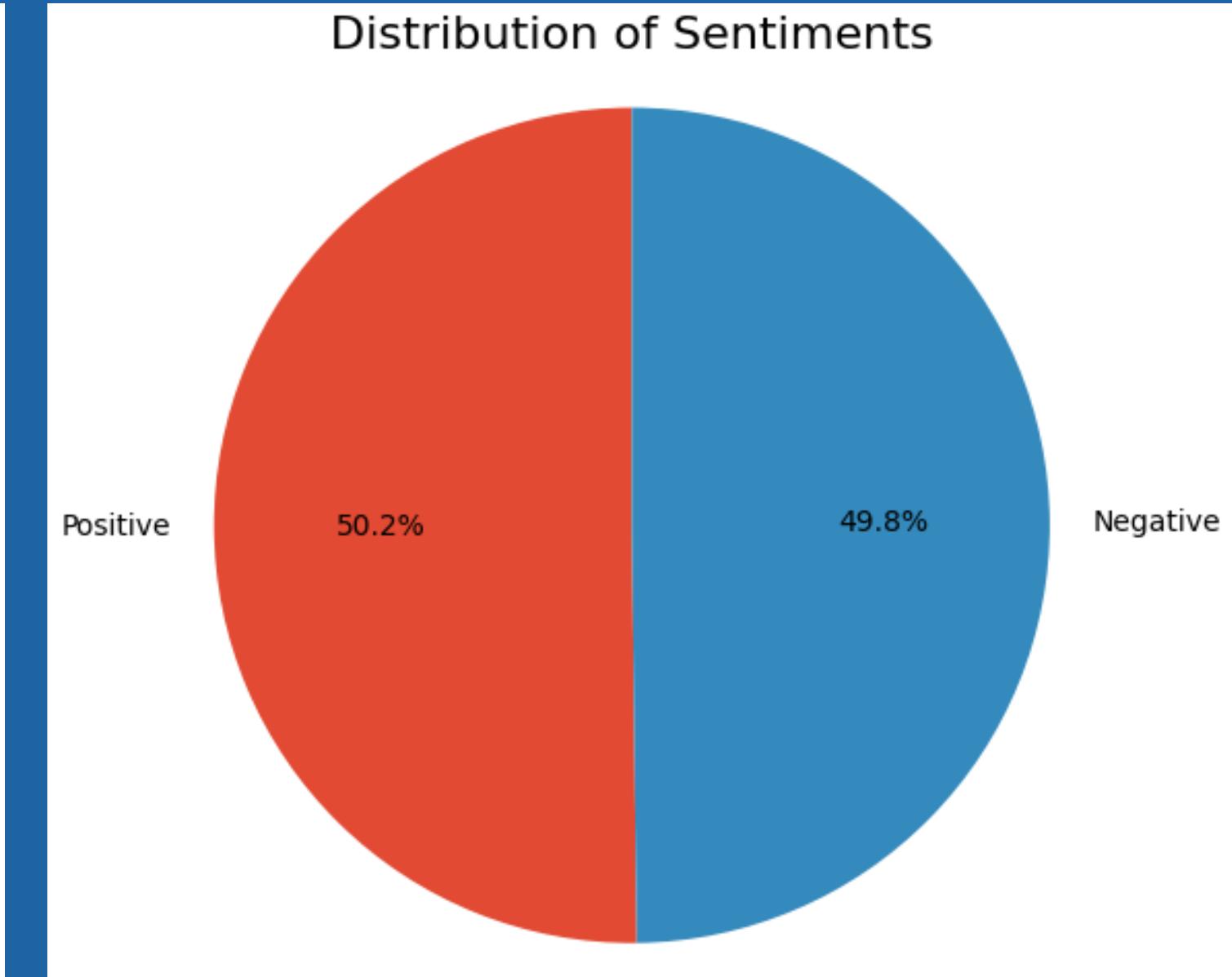
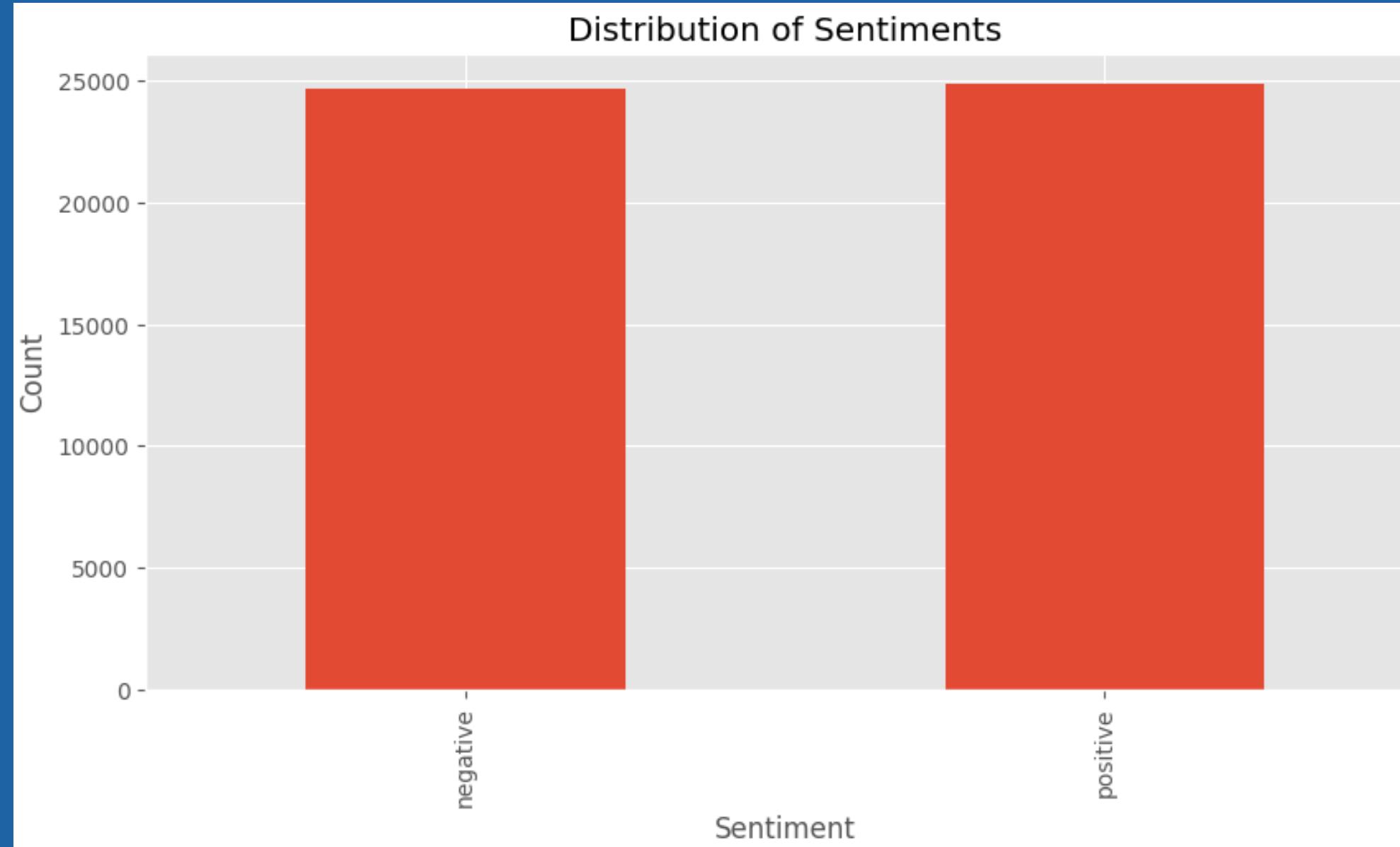
```
df.isna().sum()
```

```
review      0
sentiment   0
dtype: int64
```

```
df["review"][40]
```

```
"It had all the clichés of movies of this type and no substance. The plot went nowhere and at the end of the movie I felt like a sucker for watching it. The production was good; however, the script and acting were B-movie quality. The casting was poor because there were good actors mixed in with crumby actors. The good actors didn't hold their own nor did they lift up the others. <br /><br />This movie is not worthy of more words, but I will say more to meet the minimum requirement of ten lines. James Wood and Cuba Gooding, Jr. play caricatures of themselves in other movies. <br /><br />If you are looking for mindless entertainment, I still wouldn't recommend this movie."
```

Importing Dataset & Preprocessing

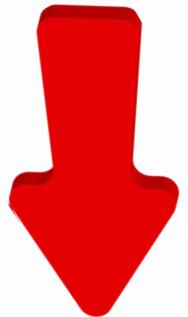


balanced data

After Cleaning

```
df["review"][40]

"It had all the clichés of movies of this type and no substance. The plot went nowhere and at the end of the movie I felt like a sucker for watching it. The production was good; however, the script and acting were B-movie quality. The casting was poor because there were good actors mixed in with crumby actors. The good actors didn't hold their own nor did they lift up the others. <br /><br />This movie is not worthy of more words, but I will say more to meet the minimum requirement of ten lines. James Wood and Cuba Gooding, Jr. play caricatures of themselves in other movies. <br /><br />If you are looking for mindless entertainment, I still wouldn't recommend this movie."
```



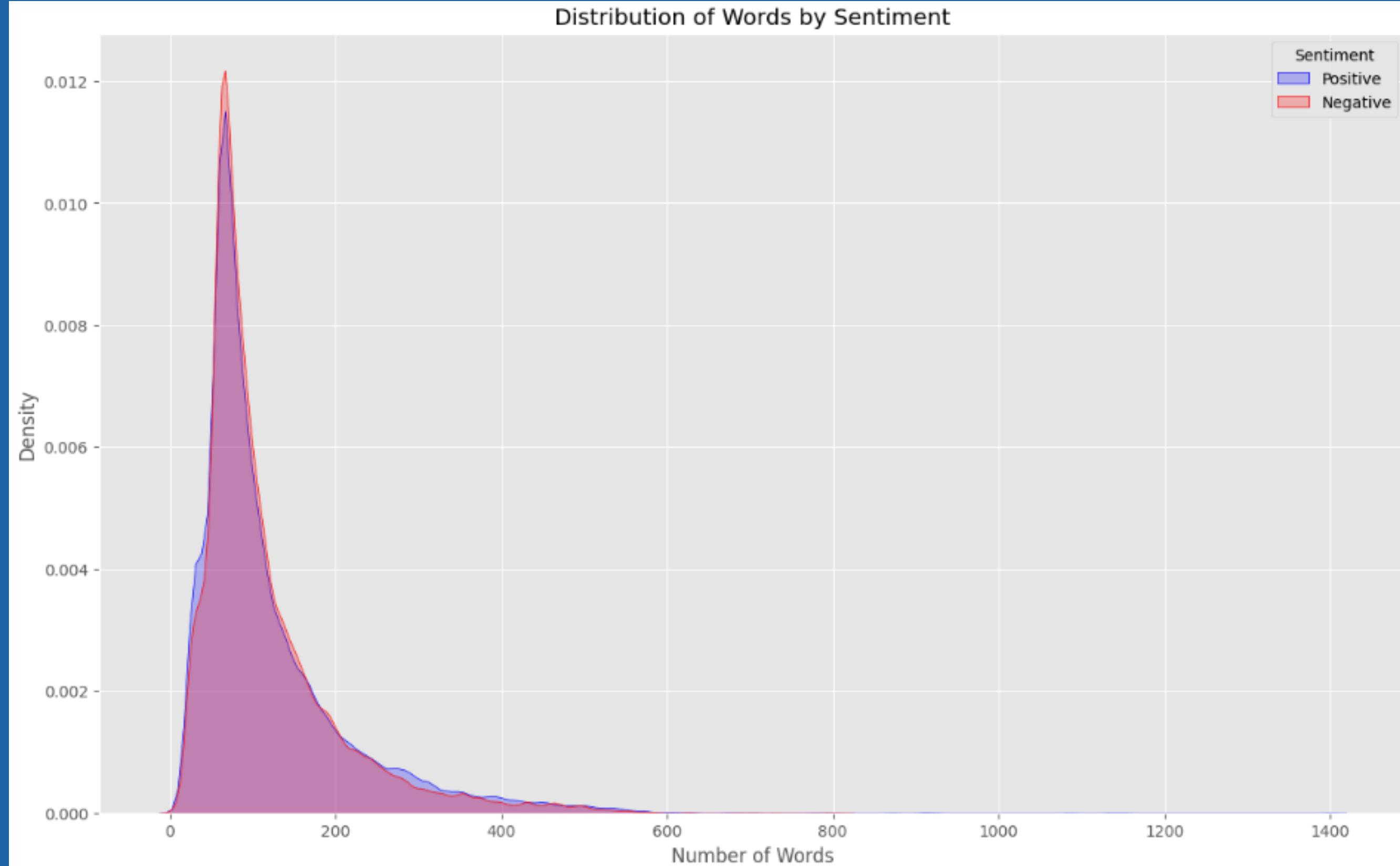
```
df['review'][40]

'clich movies type substance plot went nowhere end movie felt like sucker watching production good however script acting movie quality casting poor good actors mixed crumby actors good actors hold lift others movie worthy words say meet minimum requirement ten lines james wood cuba gooding jr play caricatures movies looking mindless entertainment still recommend movie'
```

convert labels into integers. convert them into integers by replacing "positive" --> 1 and negative --> 0

	review	sentiment	word_length
0	one reviewers mentioned watching oz episode ho...	1	162
1	wonderful little production filming technique ...	1	86
2	thought wonderful way spend time hot summer we...	1	84
3	basically family little boy jake thinks zombie...	0	64
4	petter mattei love time money visually stunnin...	1	125

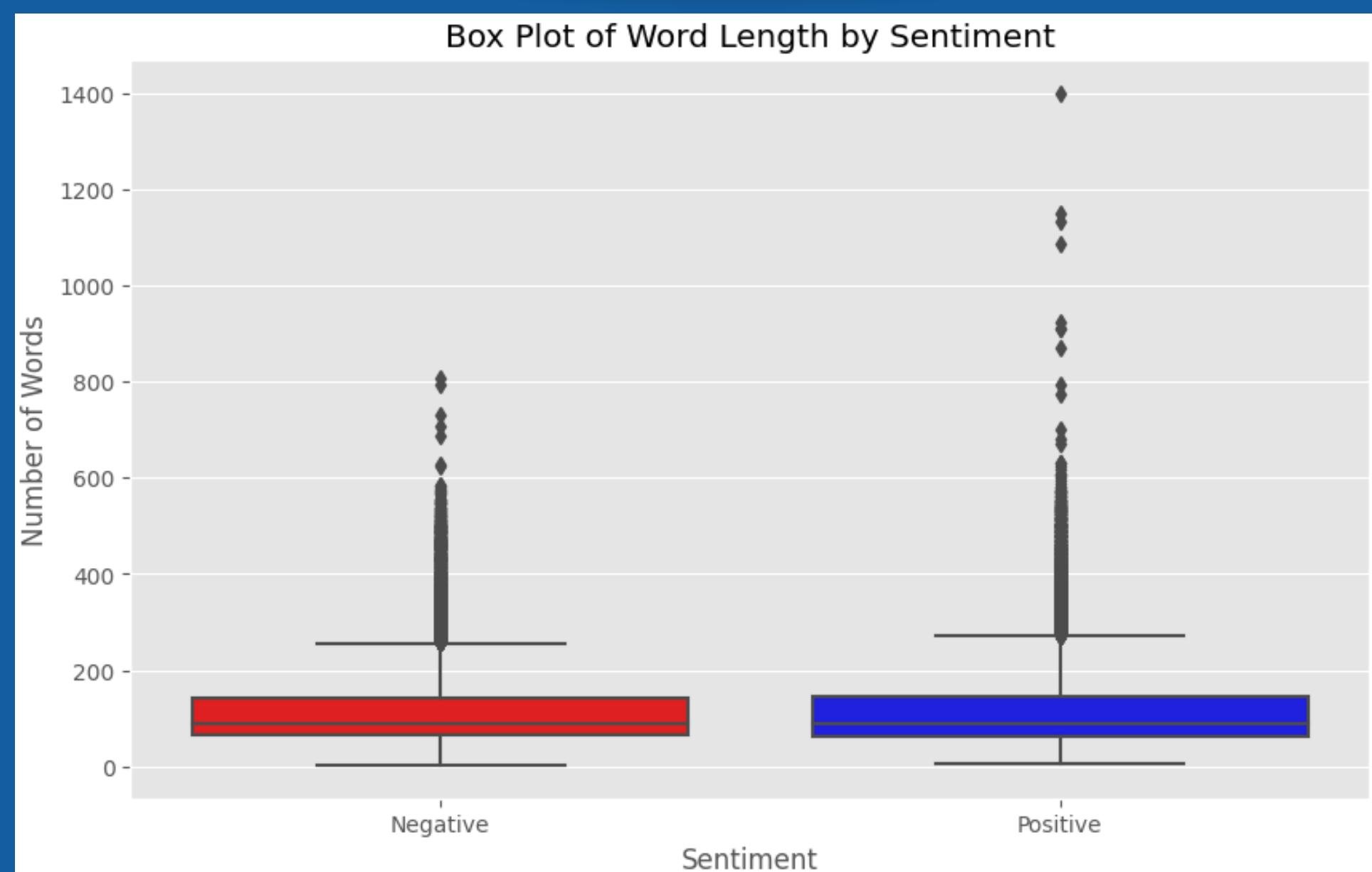
Plot KDE for Positive and Negative sentiment



Average review length

```
average_length = df.groupby('sentiment')['word_length'].mean()  
print("Average review length:")  
print(average_length)  
  
Average review length:  
sentiment  
0    116.133128  
1    119.834392  
Name: word_length, dtype: float64
```

Number of Words

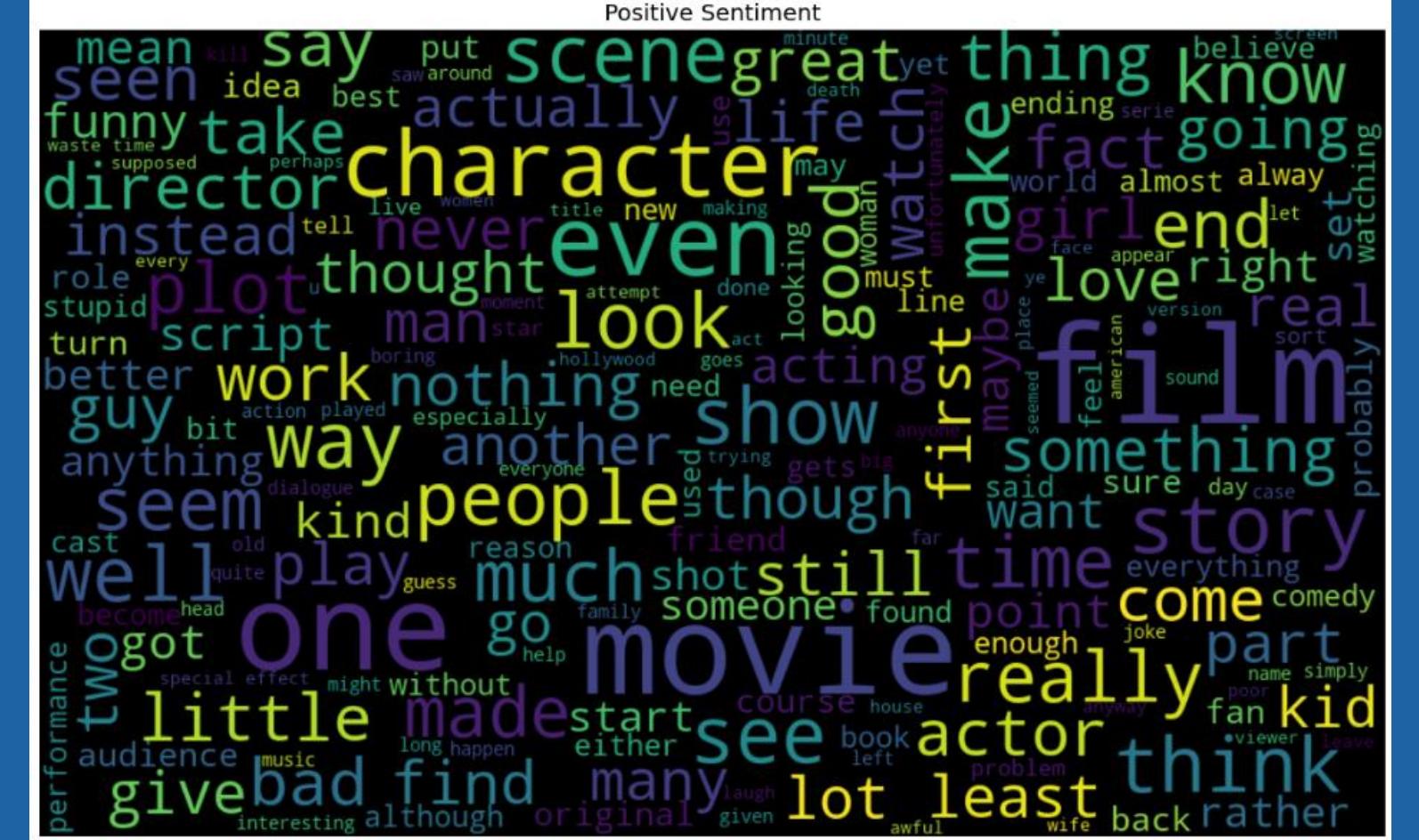


Combine all reviews labeled with sentiment

(Negative sentiment)

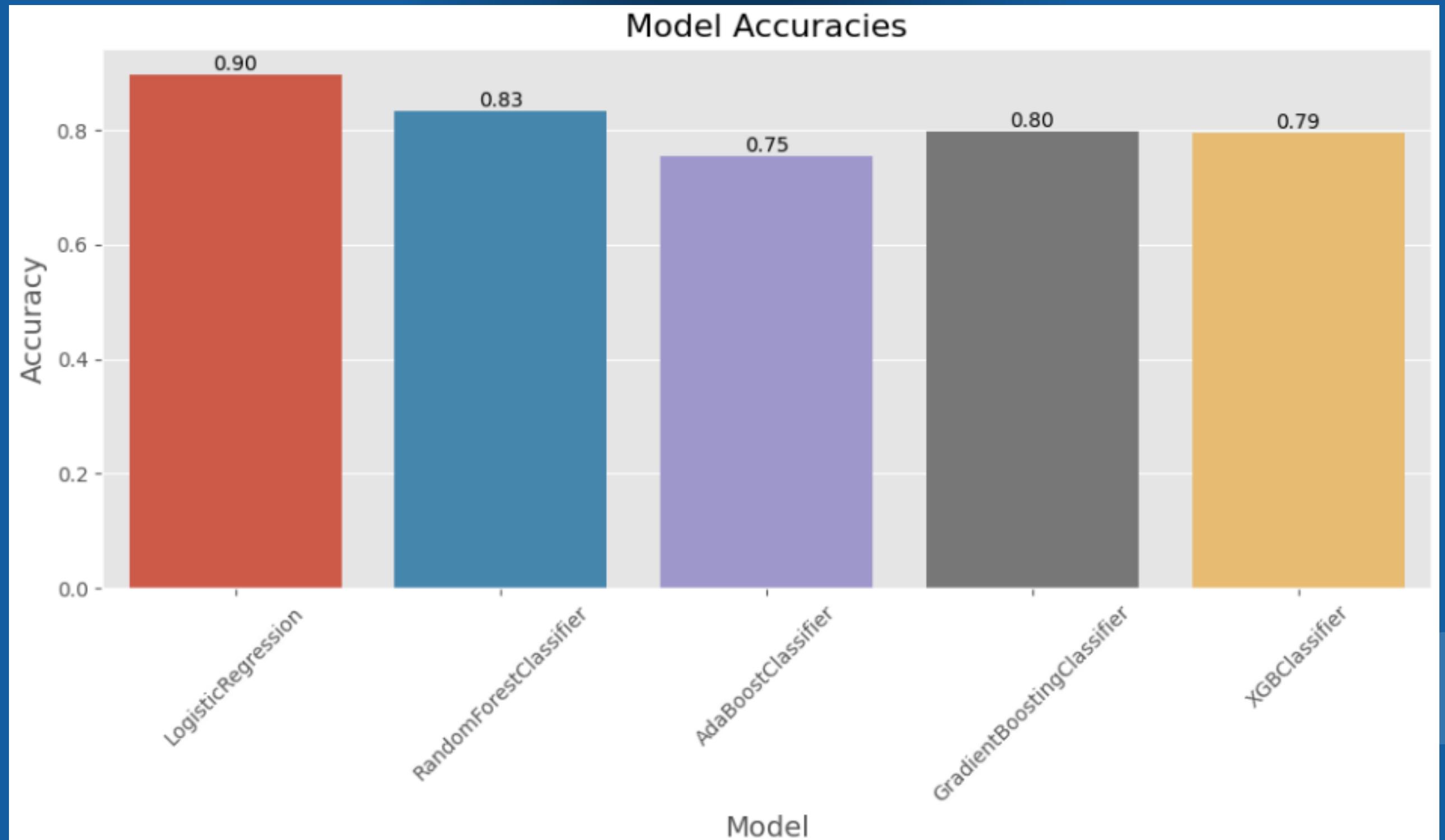


(Positive sentiment)



Machine Learning models

	Model	Test Scores
0	LogisticRegression	0.896844
1	RandomForestClassifier	0.833619
2	AdaBoostClassifier	0.754361
3	GradientBoostingClassifier	0.797419
4	XGBClassifier	0.794293



Deep Learning Model

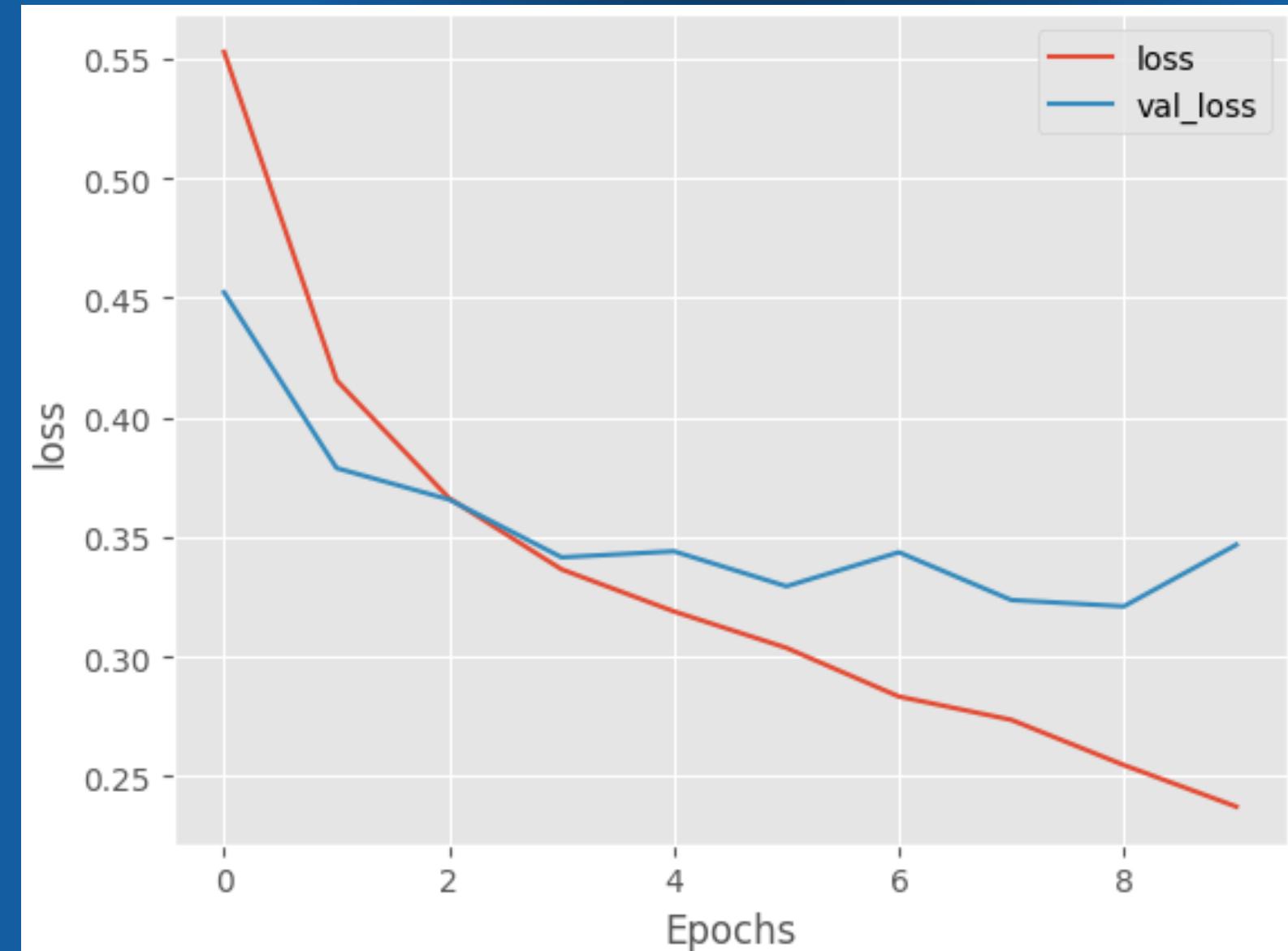
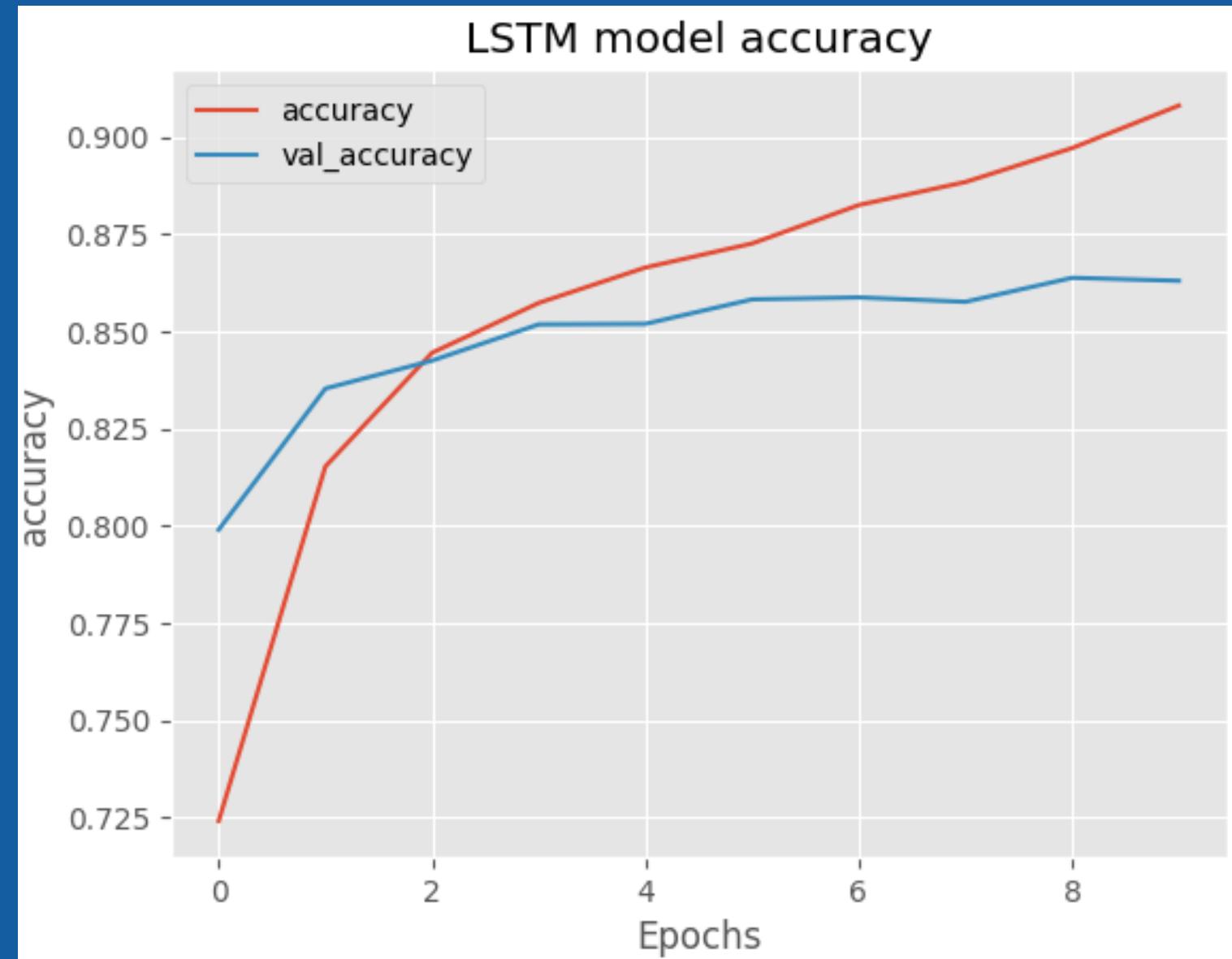
LSTM With Glove Embedding

Model: "sequential"		
Layer (type)	Output Shape	Param #
embedding (Embedding)	?	10,124,600
lstm (LSTM)	?	0 (unbuilt)
dense (Dense)	?	0 (unbuilt)

Total params: 10,124,600 (38.62 MB)
Trainable params: 0 (0.00 B)
Non-trainable params: 10,124,600 (38.62 MB)

Deep Learning Model

model accuracy

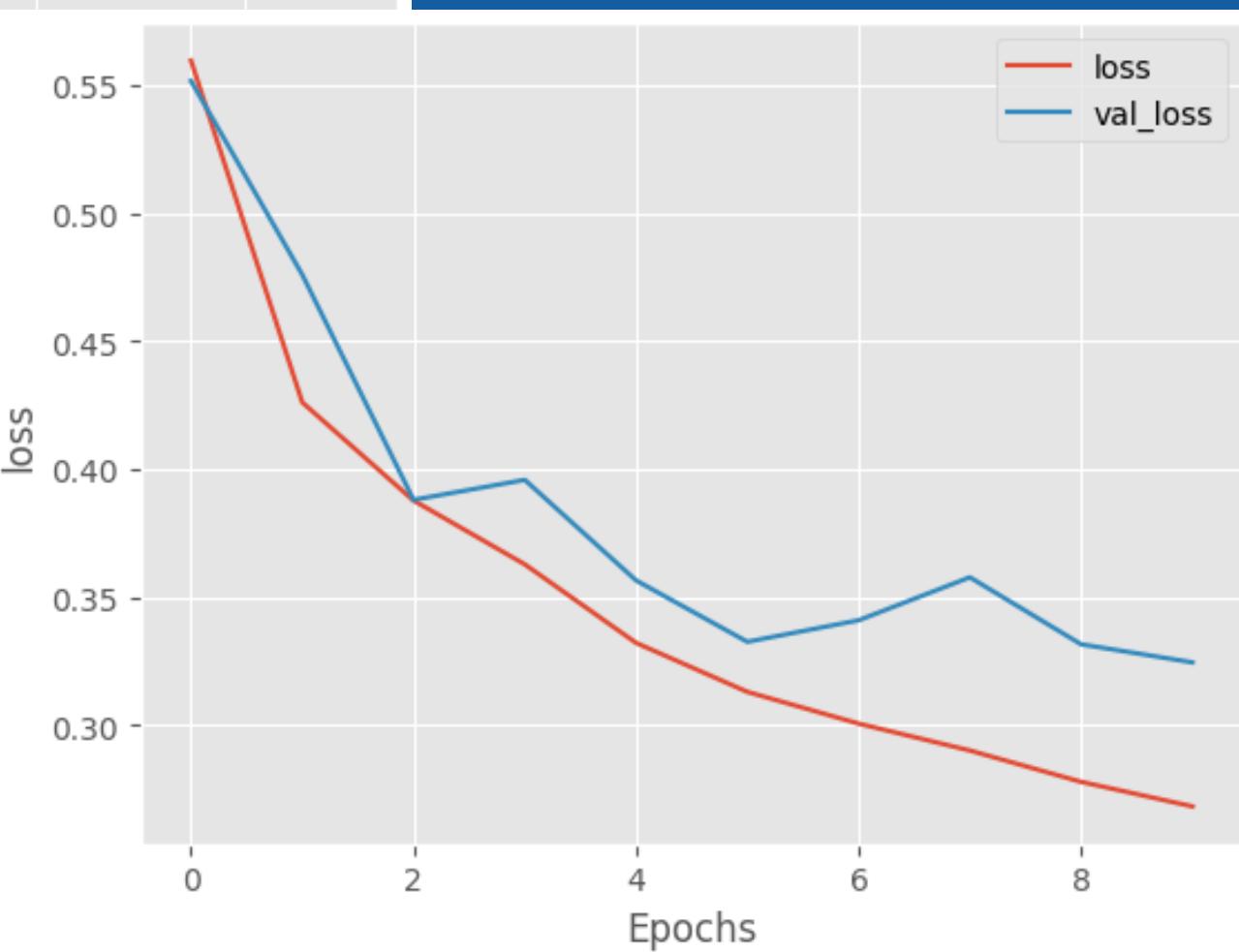
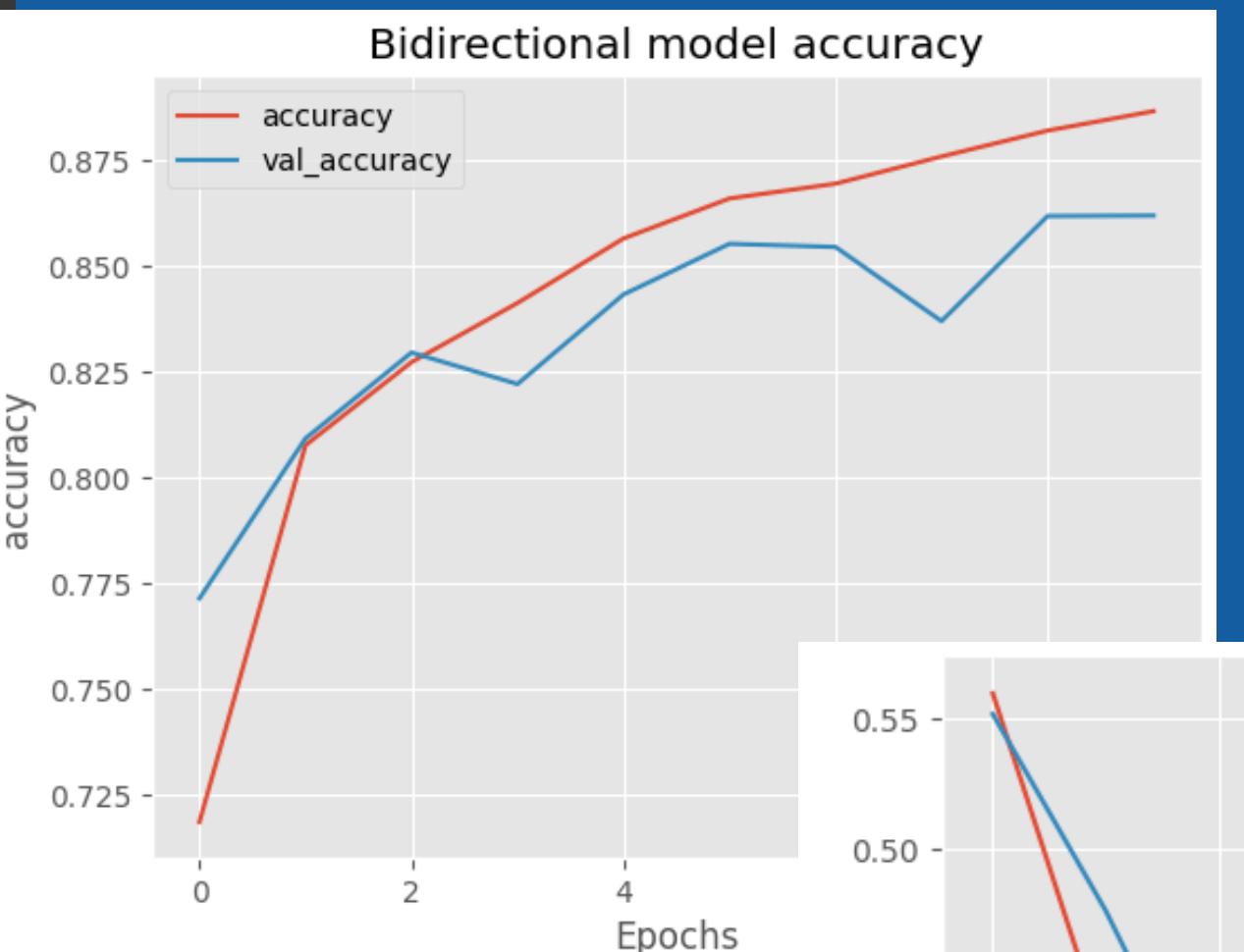


Bidirectional LSTM

Model: "sequential_10"

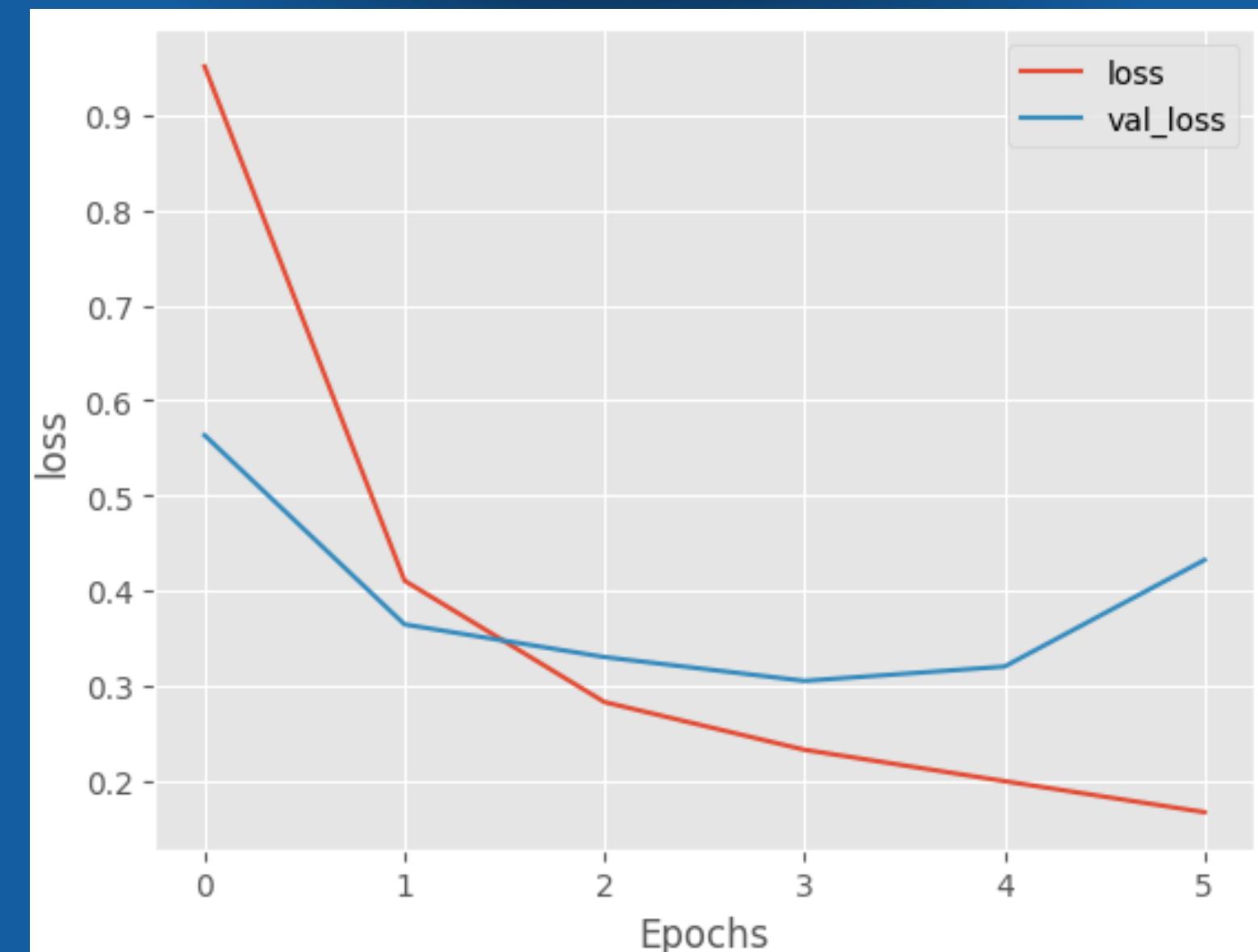
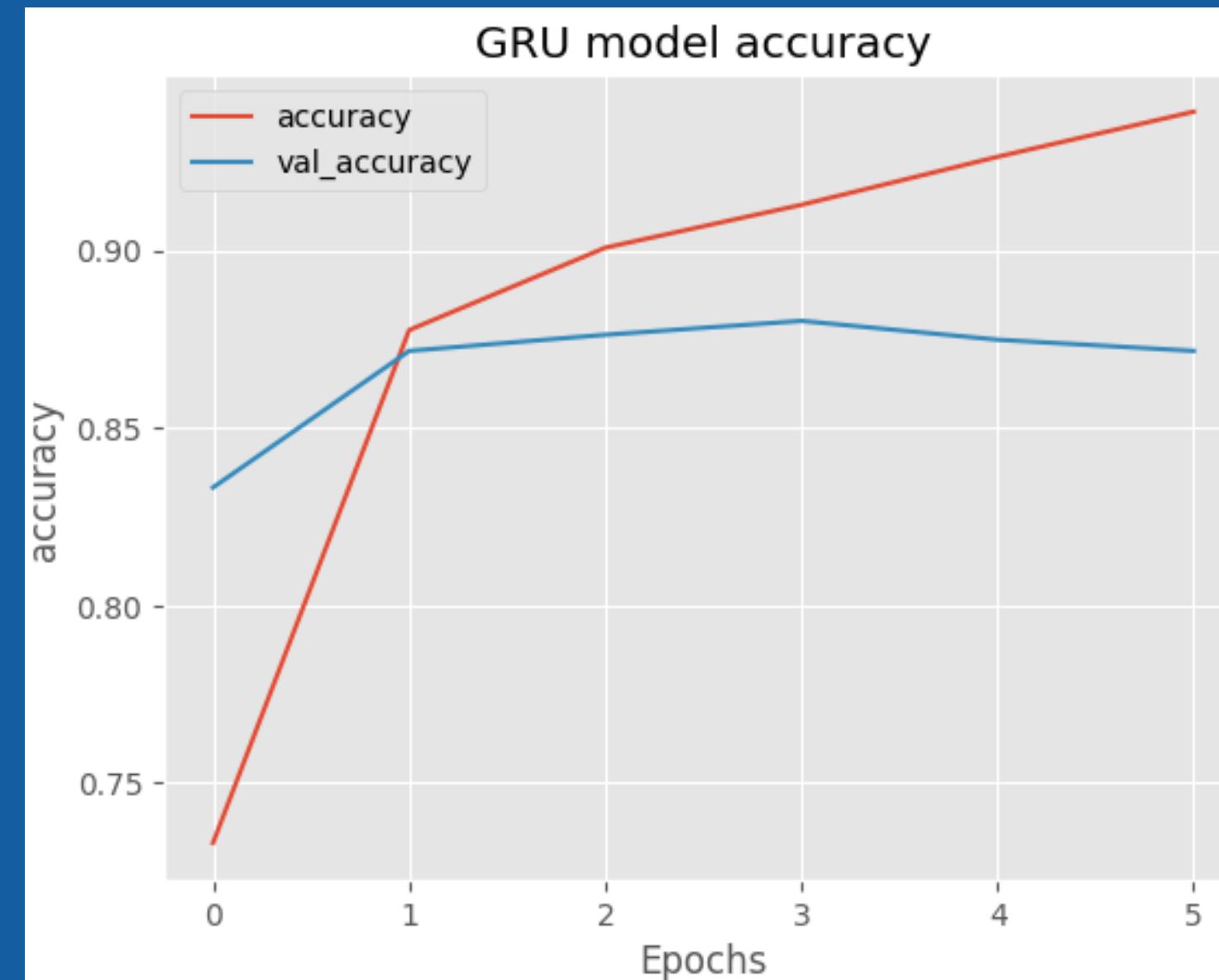
Layer (type)	Output Shape	Param #
embedding_8 (Embedding)	?	10,124,600
bidirectional_14 (Bidirectional)	?	0 (unbuilt)
dropout_20 (Dropout)	?	0 (unbuilt)
bidirectional_15 (Bidirectional)	?	0 (unbuilt)
dropout_21 (Dropout)	?	0 (unbuilt)
dense_23 (Dense)	?	0 (unbuilt)
batch_normalization_12 (BatchNormalization)	?	0 (unbuilt)
dense_24 (Dense)	?	0 (unbuilt)
Total params: 10,124,600 (38.62 MB)		
Trainable params: 0 (0.00 B)		
Non-trainable params: 10,124,600 (38.62 MB)		

model accuracy



Bidirectional GRU

model accuracy



Make Predictions

```
instance = df['review'][70]
print(instance)

caddyshack two good movie compared original cant stack robert stack horrible replaceme

# Convert the input text instance to a sequence of integers using the tokenizer
instance = tokenizer.texts_to_sequences(instance)

# Flatten the list of sequences into a single list
flat_list = []
for sublist in instance:
    for item in sublist:
        flat_list.append(item)

# Wrap the flat list in another list to create a 2D array (required for padding)
flat_list = [flat_list]

# Pad the sequences to ensure uniform input size for the model
instance = pad_sequences(flat_list, padding='post', maxlen=max_len)

# Use the model to predict the sentiment of the processed instance
model.predict(instance)

1/1 ━━━━━━ 0s 369ms/step
array([[0.90760624]], dtype=float32)
```

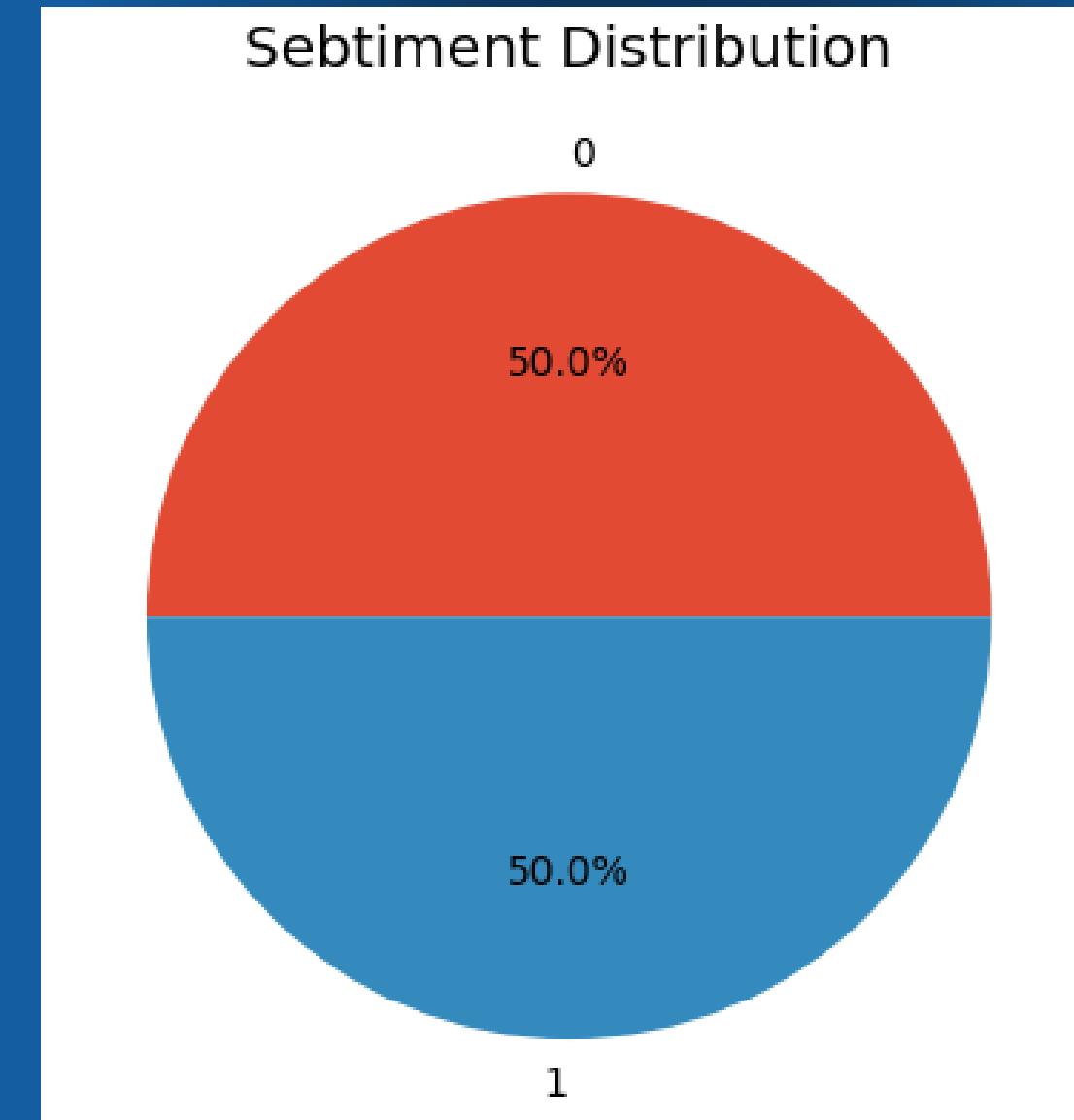
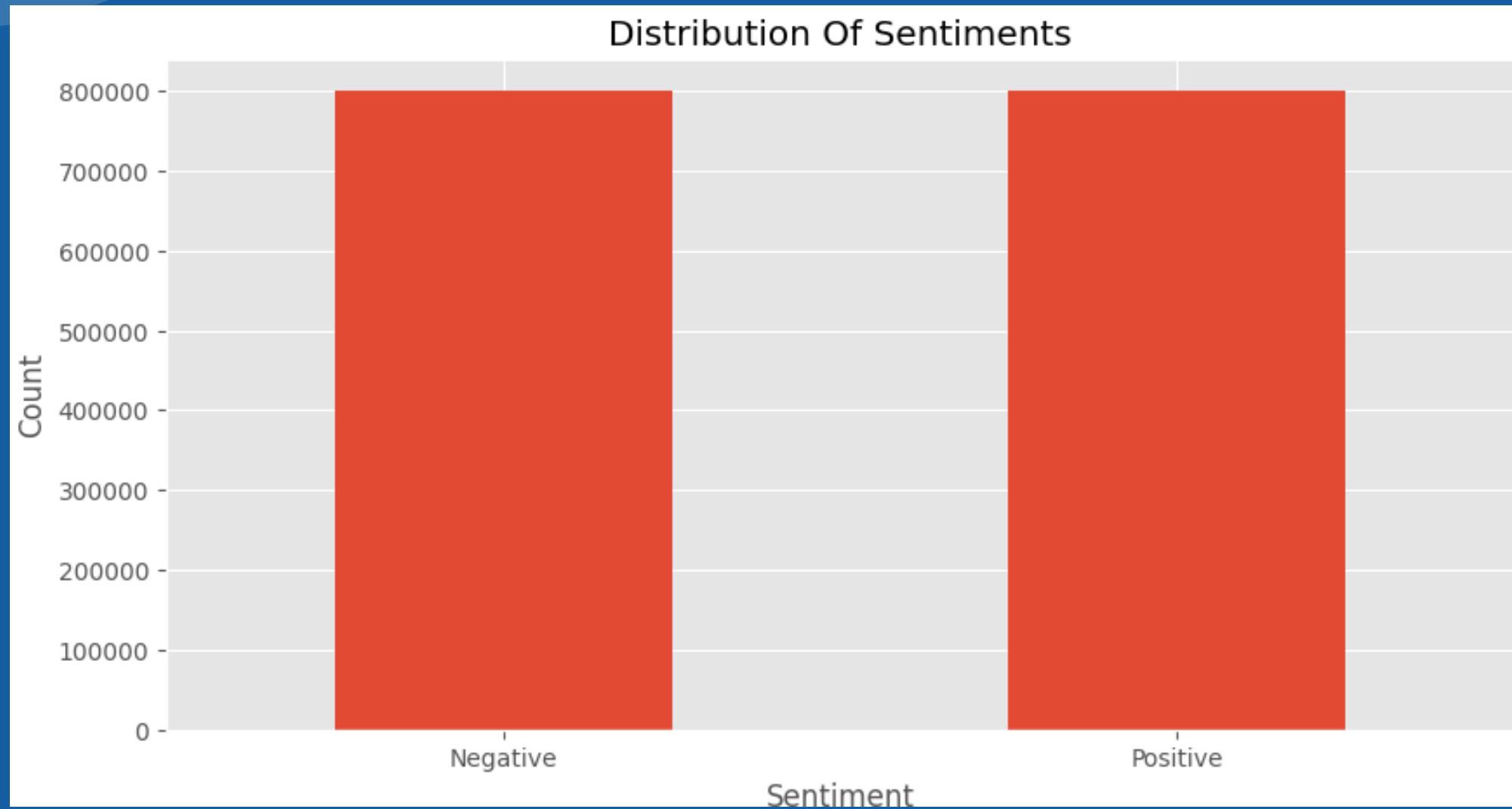
Twitter sentiment

This is the sentiment140 dataset. It contains 1,600,000 tweets extracted using the twitter api . The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment .

sentiment	Id	Date	Query	User_id	text
0	0 1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0 1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0 1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0 1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0 1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

Twitter sentiment

Plot the distribution of sentiments



Twitter sentiment

implement a balanced sampling strategy by randomly selecting 75 ,000 reviews from each Class . This approach ensures equal representation across all Classes , which facilitates more accurate sentiment . Additionally, by reducing the dataset size, we also decrease training time, making the model training process more efficient.

sentiment
0 75000
1 75000

Convert to Lowercase: Convert the entire text to lowercase, To standardize the text so that words are treated equally regardless of their case.

Remove Special Characters, Usernames, and Hyperlinks: To eliminate unwanted characters that do not contribute to the meaning of the text..

Remove Stopwords: Filter out common stopwords,that do not add significant meaning to the text.

Lemmatize the Text: Convert words to their base forms.

Joining Tokens: To reconstruct the processed text from the list of lemmatized tokens.

performs preprocessing on the string

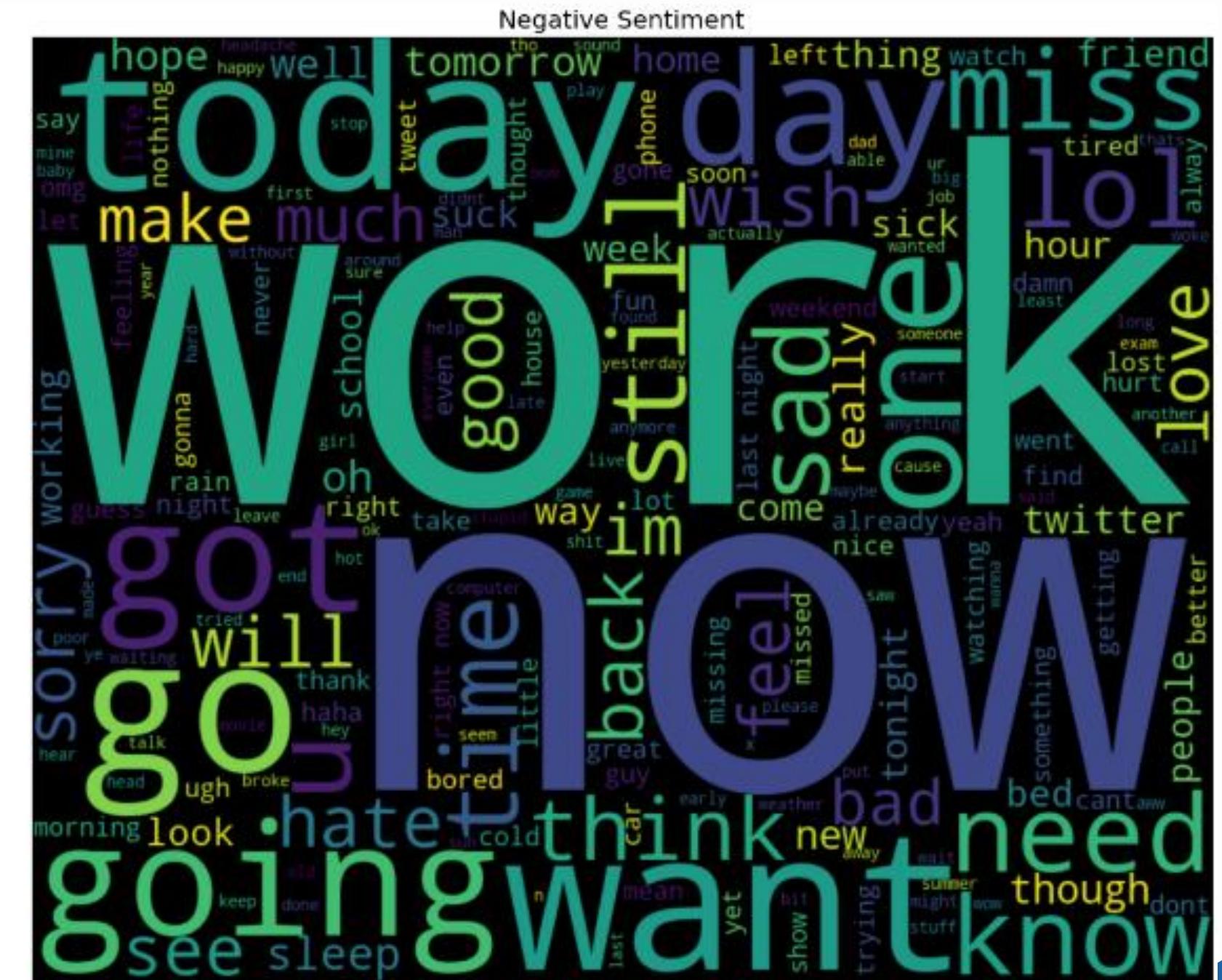
after perprocessing

Apply the preprocessing function to the DataFrame

```
lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))
def preprocess_tweet(tweet):
    url_pattern = re.compile(r"(?:\@|\https?://)\S+|[^\w\s#]")
    repeat_pattern = re.compile(r"(.)\1{2,}")
    tweet = BeautifulSoup(tweet, 'html.parser').get_text()
    tweet = tweet.lower()
    tweet = url_pattern.sub('', tweet)
    tokens = word_tokenize(tweet)
    expanded_words = [repeat_pattern.sub(r"\1\1", word) for word in tokens]
    expanded_words = [re.sub(r"^\w\s]", "", word) for
                      word in expanded_words if word.strip()]
    expanded_words = [word for word in expanded_words if not re.search(r'\d', word)]
    lemmatized_tokens = [lemmatizer.lemmatize(word) for word in
                         expanded_words if word not in stop_words and len(word) > 1]
    processed_tweet = ' '.join(lemmatized_tokens)
    return processed_tweet
```

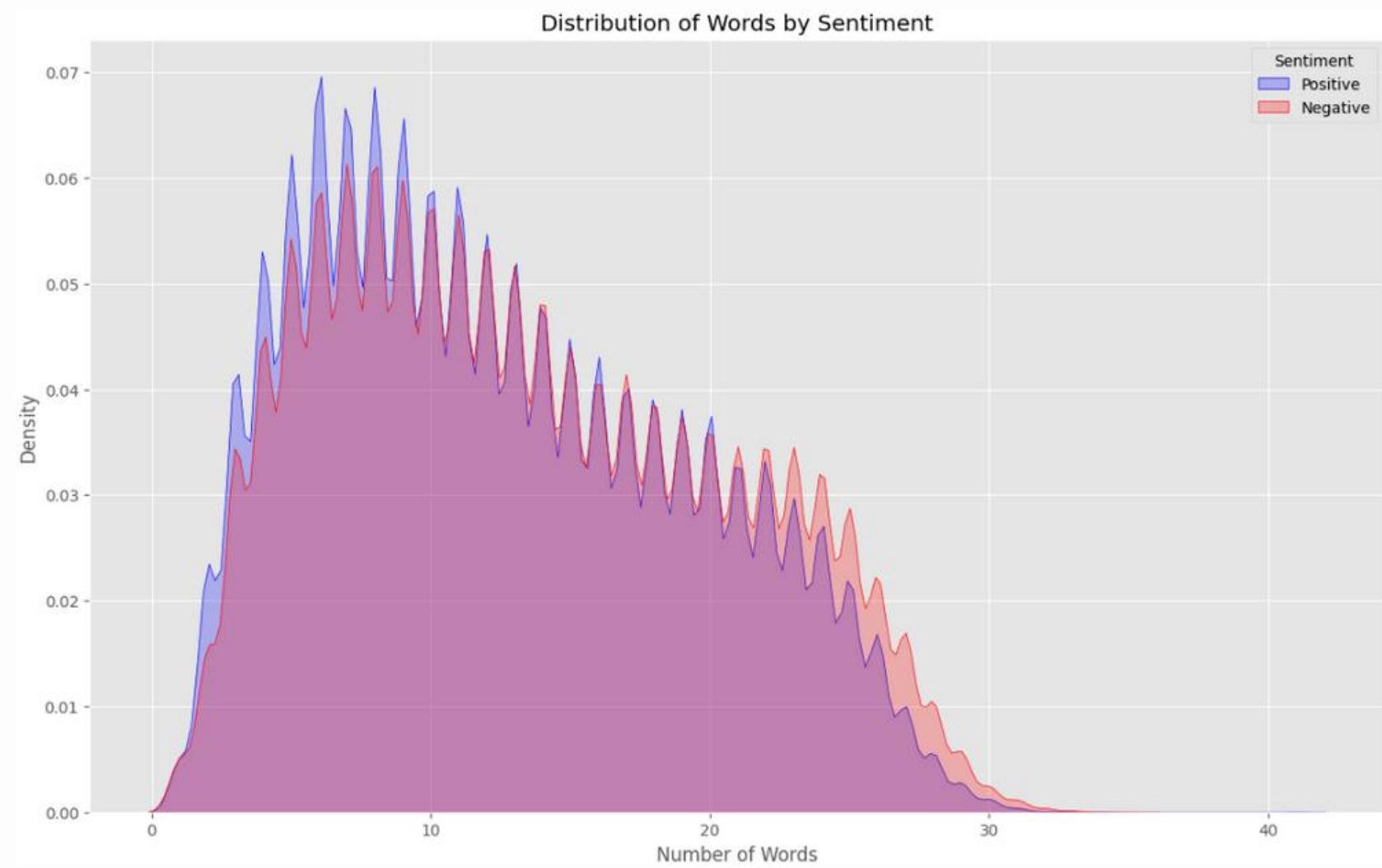
	sentiment	text
12934	0	OMG Transformers comes out tonite I'm tryin...
76182	1	Listening to Sia calming me down About to get ...
26739	0	Leanne0710 they horny kittys have been followi...
128613	1	got to level 10 on tetris it's school tomorrow...
73170	0	just for back from track practise and fricken ...
95052	1	dreamhampton Yes by far To this day before I g...
63971	0	I'm not sure if this is working told you I do...
45379	0	i miss youu tooo kblovesyouu you just called m...
97710	1	campusrock war cool, ja schon
57052	0	LouiseMayes Oh really, this nice weather is le...

word cloud for positive and negative sentiment

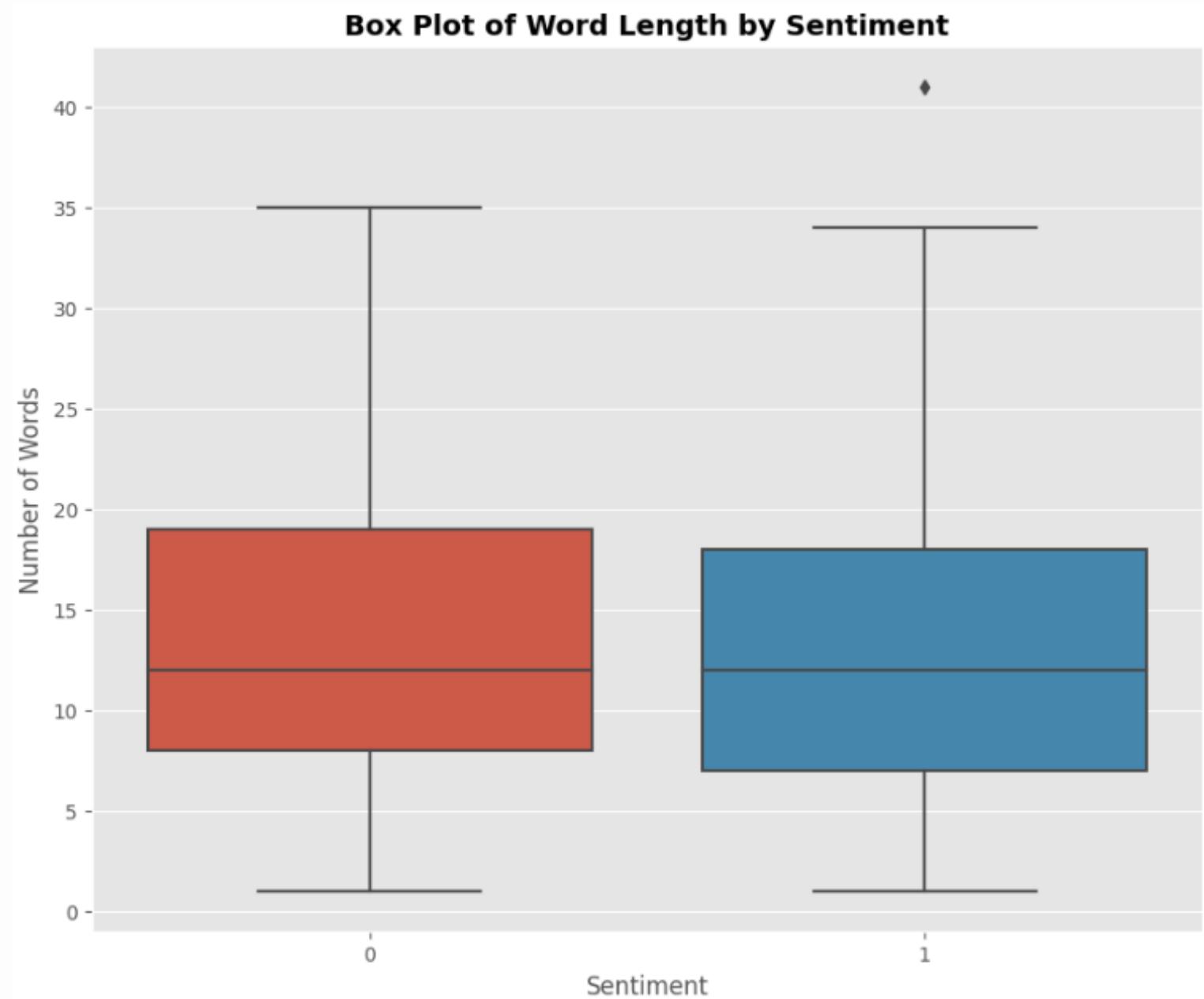


Visualization

KDE for Negative and positive sentiment



Create a box plot for word length distribution by sentiment



Twitter sentiment

- Preprocessing steps
- **Splitting the data into train, validation, and test sets**
- **One-Hot Encoding**
- **TF-IDF Vectorization**
- **Tokenization and Padding**

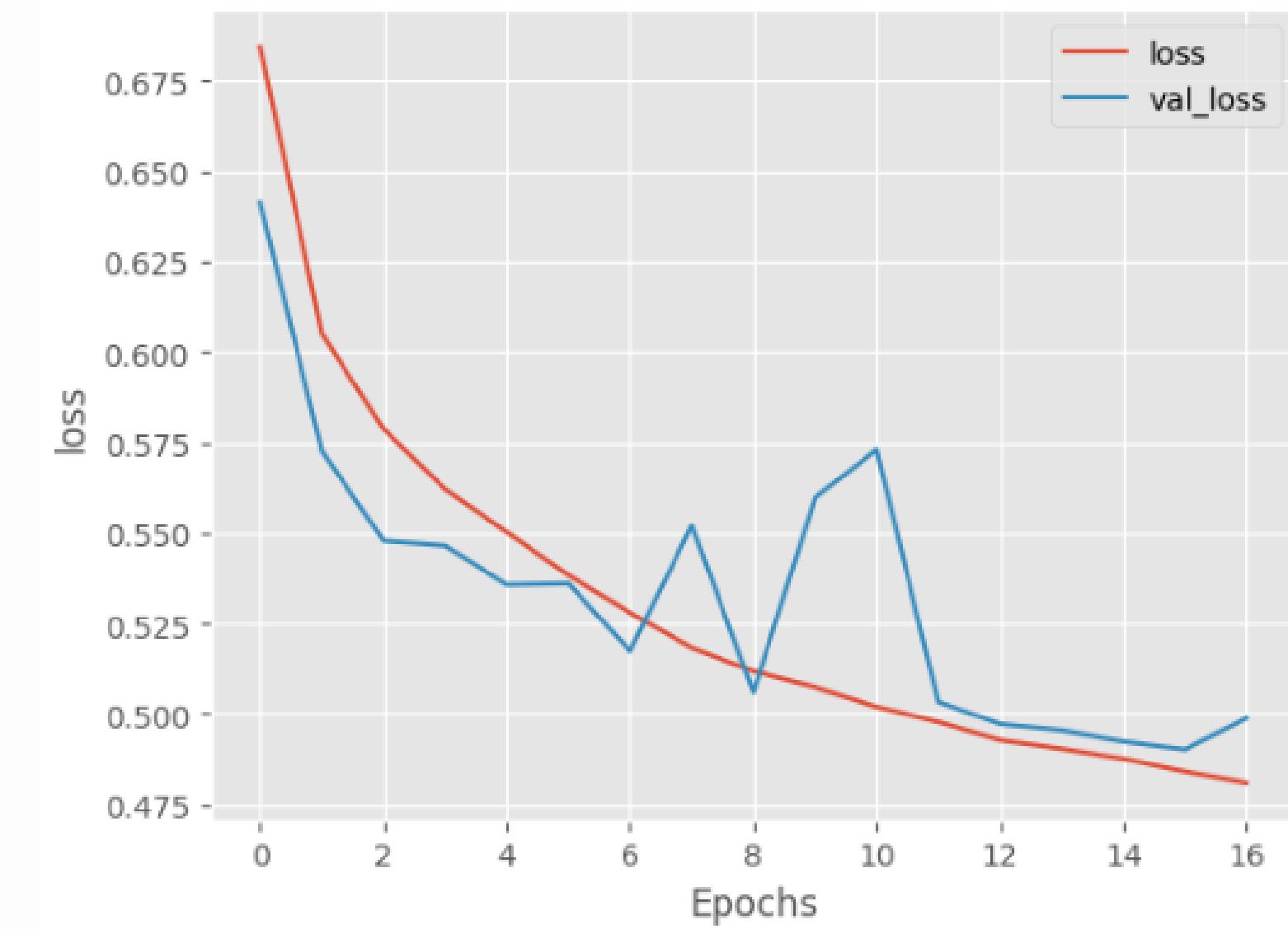
- Deep learning Model

- Load GloVe embeddings
- Create embedding matrix
- Initialize the model
- Embedding layer (converts text to dense vector representations)
- Bidirectional LSTM layer

Model: "sequential"		
Layer (type)	Output Shape	Param #
embedding (Embedding)	?	1,000,000
bidirectional (Bidirectional)	?	0 (unbuilt)
dropout (Dropout)	?	0 (unbuilt)
bidirectional_1 (Bidirectional)	?	0 (unbuilt)
dropout_1 (Dropout)	?	0 (unbuilt)
bidirectional_2 (Bidirectional)	?	0 (unbuilt)
dropout_2 (Dropout)	?	0 (unbuilt)
dense (Dense)	?	0 (unbuilt)
batch_normalization (BatchNormalization)	?	0 (unbuilt)
dense_1 (Dense)	?	0 (unbuilt)
Total params: 1,000,000 (3.81 MB)		
Trainable params: 0 (0.00 B)		
Non-trainable params: 1,000,000 (3.81 MB)		

Twitter sentiment

Plot training and validation accuracy



Twitter sentiment

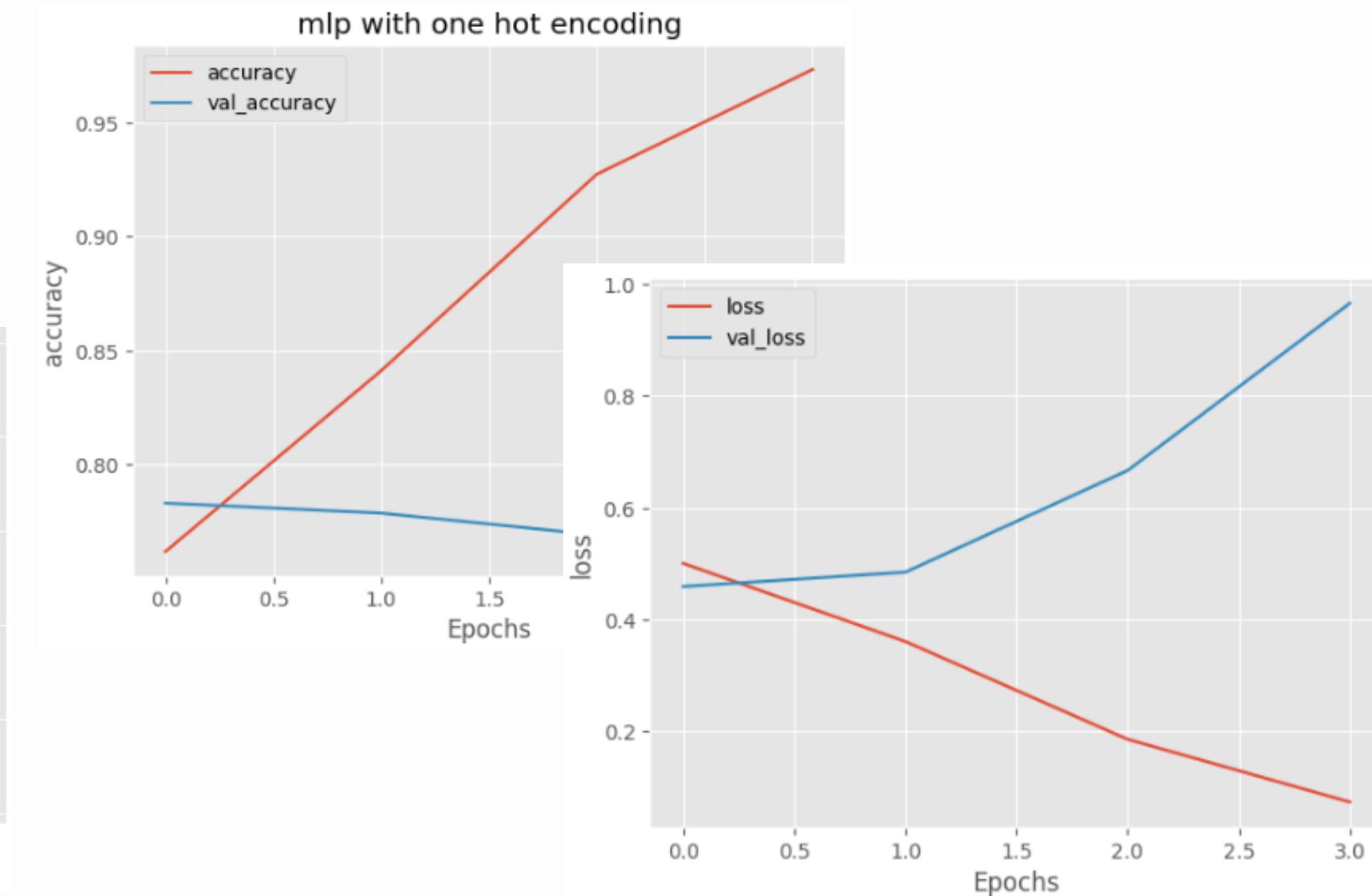
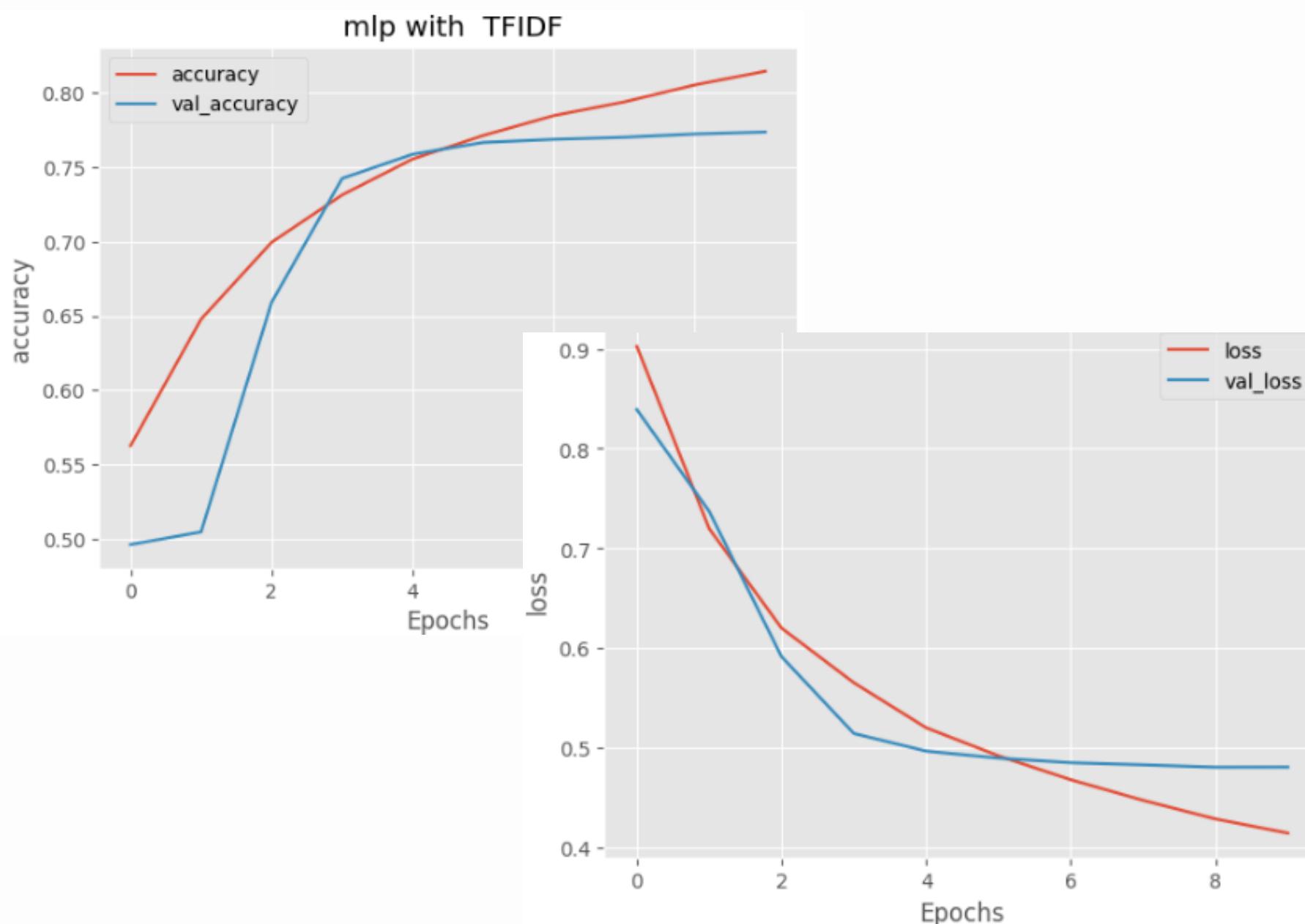
Define the MLP model with TF-IDF

```
Training Loss: 0.3477531969547272  
Training Accuracy: 0.8496285676956177  
Validation Loss: 0.48063763976097107  
Validation Accuracy: 0.7675555348396301
```

MLP model with one hot encoding

```
Training Loss: 0.3713001608848572  
Training Accuracy: 0.8438857197761536  
Validation Loss: 0.4552420675754547  
Validation Accuracy: 0.7828888893127441
```

Plot training and validation accuracy

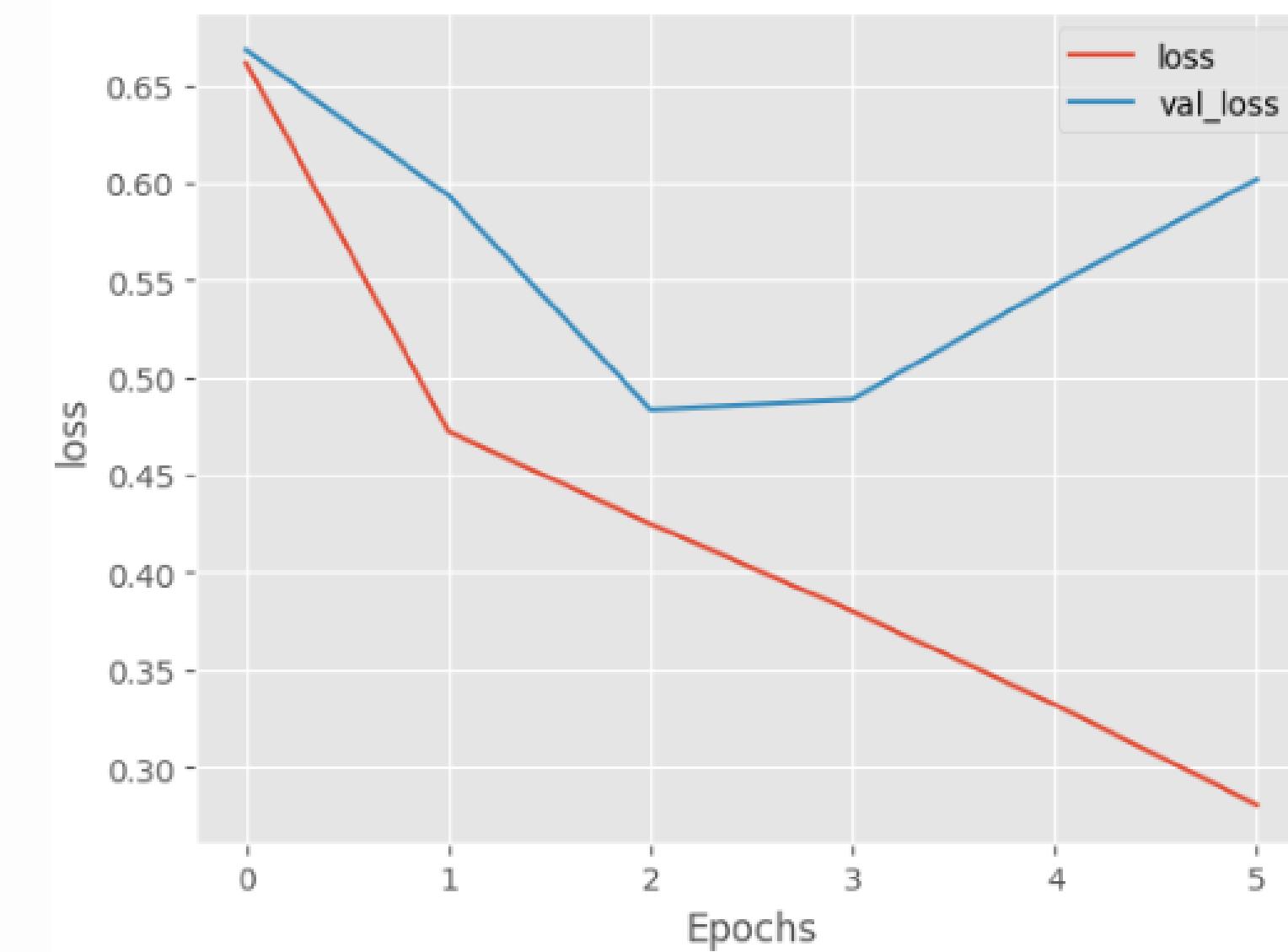
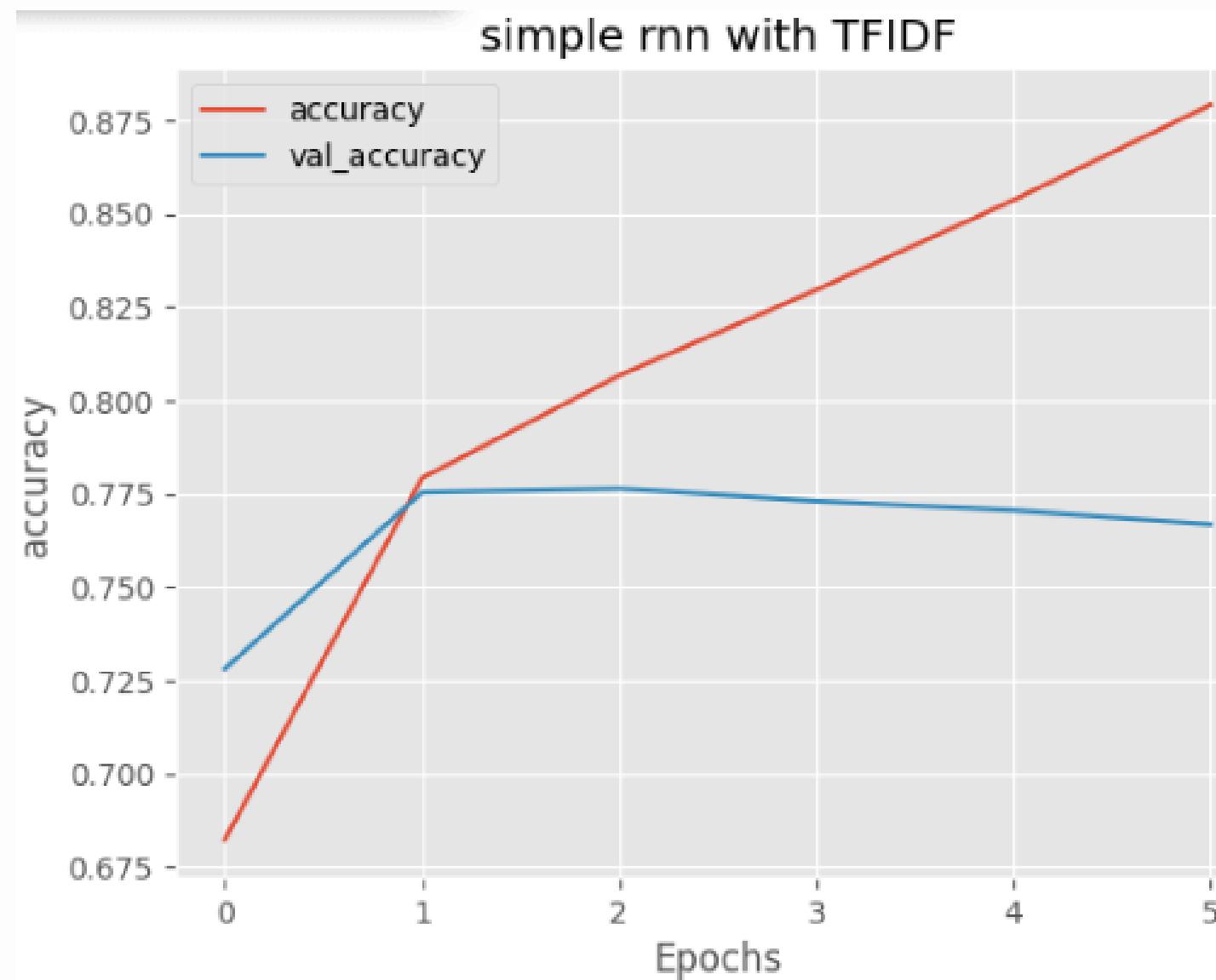


Twitter sentiment

simple RNN with TF-IDF

```
Training Loss: 0.3924139142036438
Training Accuracy: 0.8541333079338074
Validation Loss: 0.4823756814002991
Validation Accuracy: 0.7724888920783997
```

Plot training and validation accuracy



Twitter sentiment

DL Models

LSTM Model

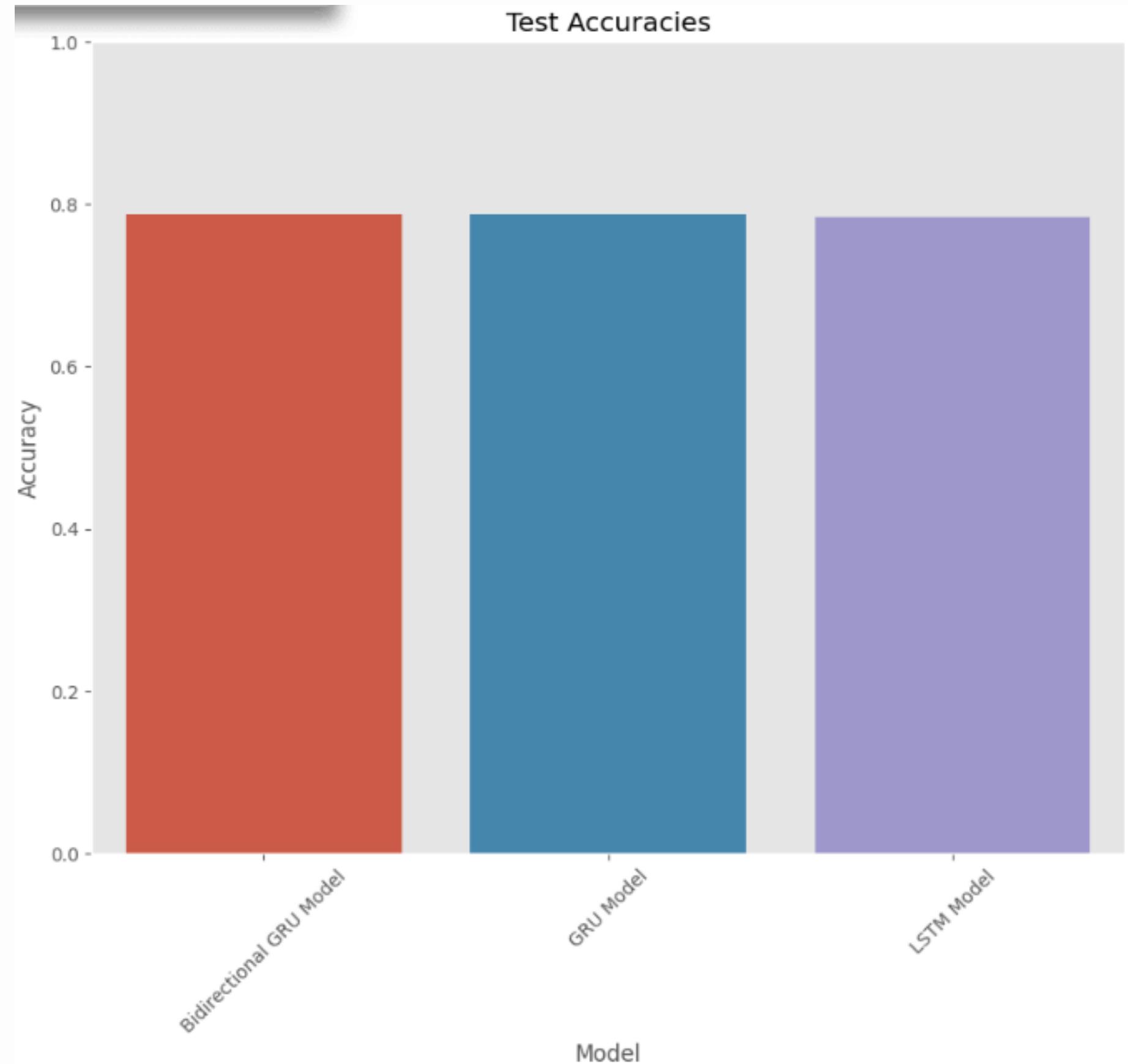
- Training Accuracy for LSTM Model: 0.8350
- Test Accuracy for LSTM Model: 0.7844

GRU Model

- Training Accuracy for GRU Model: 0.8302
- Test Accuracy for GRU Model: 0.7866

Bidirectional GRU Model

- Training Accuracy for Bidirectional GRU Model: 0.8495
- Test Accuracy for Bidirectional GRU Model: 0.7876



BERT Tokenization

BERT tokenization is used to convert the raw text into numerical inputs that can be fed into the BERT model. It tokenized the text and performs some preprocessing to prepare the text for the model's input format.

Batch Encoding: The `batch_encode_plus` function is used to tokenize and encode the training, validation, and test sets:

- 1- Padding: Ensures all sequences are the same length.
- 2- Truncation: Cuts off sequences that exceed the maximum length.
- 3- Tensor Conversion: Outputs are returned as TensorFlow tensors for compatibility with TensorFlow models.

Twitter sentiment using Bert

BERT

- Test Loss: 0.426116683483124
- Test Accuracy: 0.8358666896820068

DistilBERT

- Test Loss: 0.4014144241809845
- Test Accuracy: 0.829200029373169

RoBERTa

- Test Loss: 0.36609816551208496
- Test Accuracy: 0.8709333539009094

ALBERT

- Test Loss: 0.4147121012210846
- Test Accuracy: 0.833644449710846

```
Model: "tf_bert_for_sequence_classification_1"
```

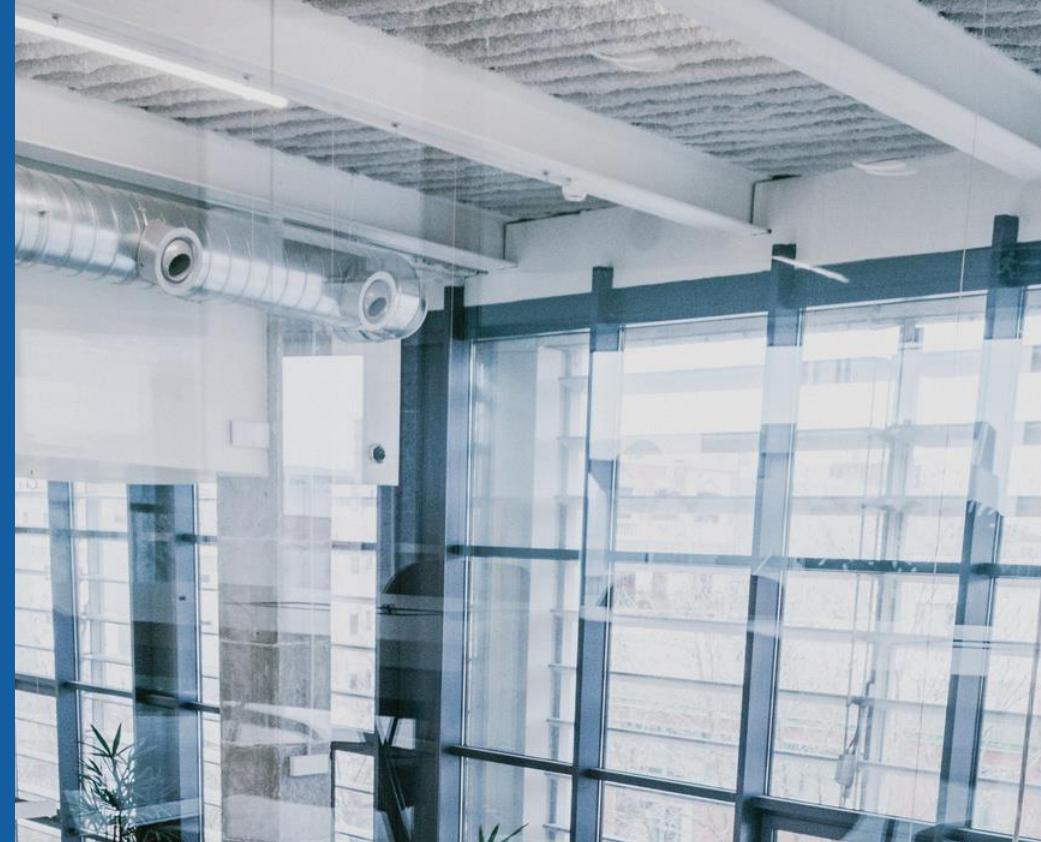
Layer (type)	Output Shape	Param #
<hr/>		
bert (TFBertMainLayer)	multiple	109482240
dropout_171 (Dropout)	multiple	0
classifier (Dense)	multiple	1538
<hr/>		
Total params:	109483778 (417.65 MB)	
Trainable params:	109483778 (417.65 MB)	
Non-trainable params:	0 (0.00 Byte)	

```
Model: "tf_distil_bert_for_sequence_classification_1"
```

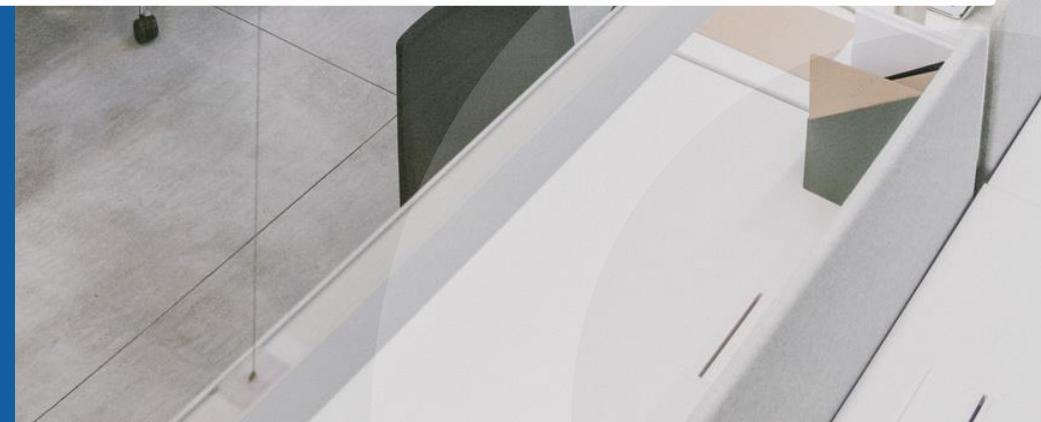
Layer (type)	Output Shape	Param #
<hr/>		
distilbert (TFDistilBertMainLayer)	multiple	66362880
pre_classifier (Dense)	multiple	590592
classifier (Dense)	multiple	1538
dropout_191 (Dropout)	multiple	0
<hr/>		
Total params:	66955010 (255.41 MB)	
Trainable params:	66955010 (255.41 MB)	
Non-trainable params:	0 (0.00 Byte)	

Amazon Product Reviews

This dataset contains more than 568k consumer reviews on different amazon products.

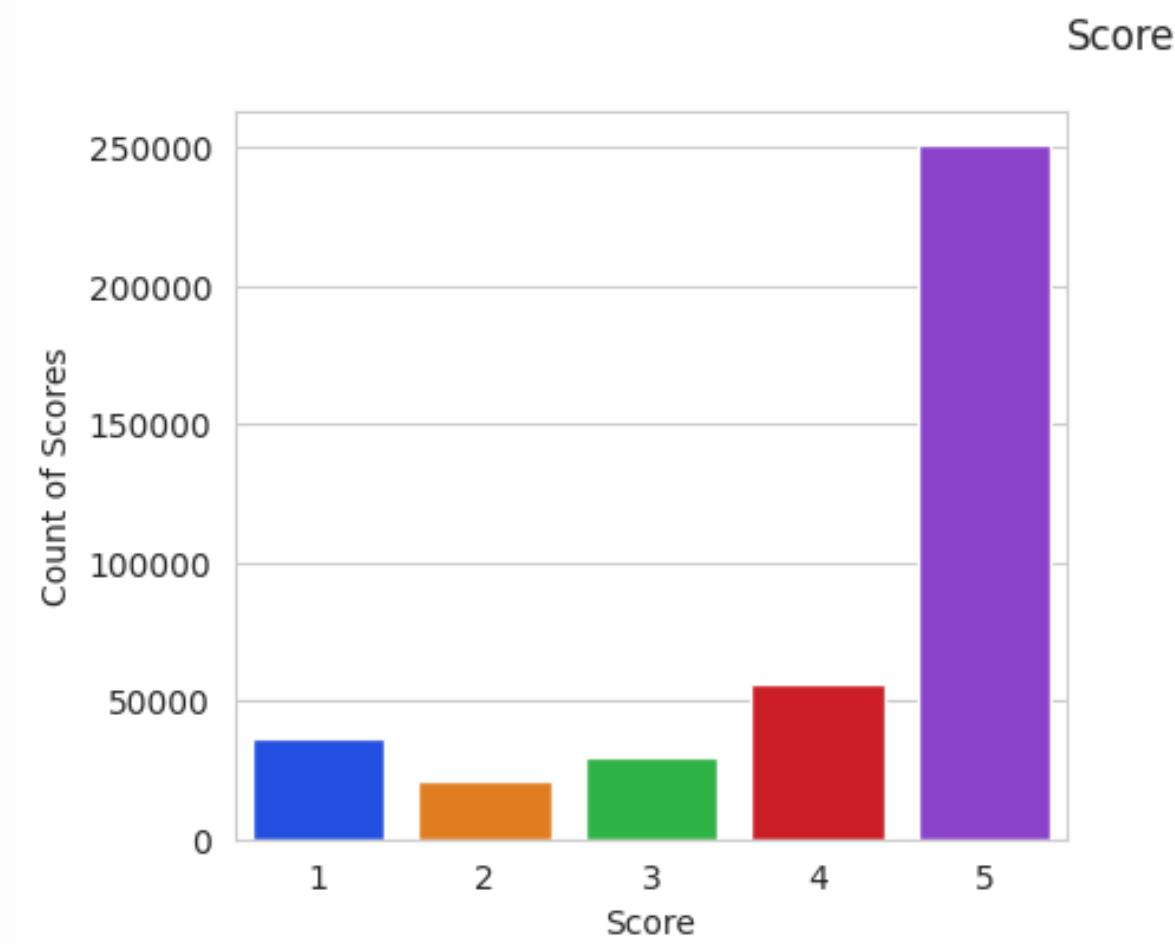


	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned dog food products and have found them all to be of ...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanuts...the peanuts were actually small sized unsalted...
2	3	B000LQOCHO	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a few centuries. It is a light, pillowy citrus gelati...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient in Robitussin I believe I have found it. I got thi...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wide assortment of yummy taffy. Delivery was very qu...

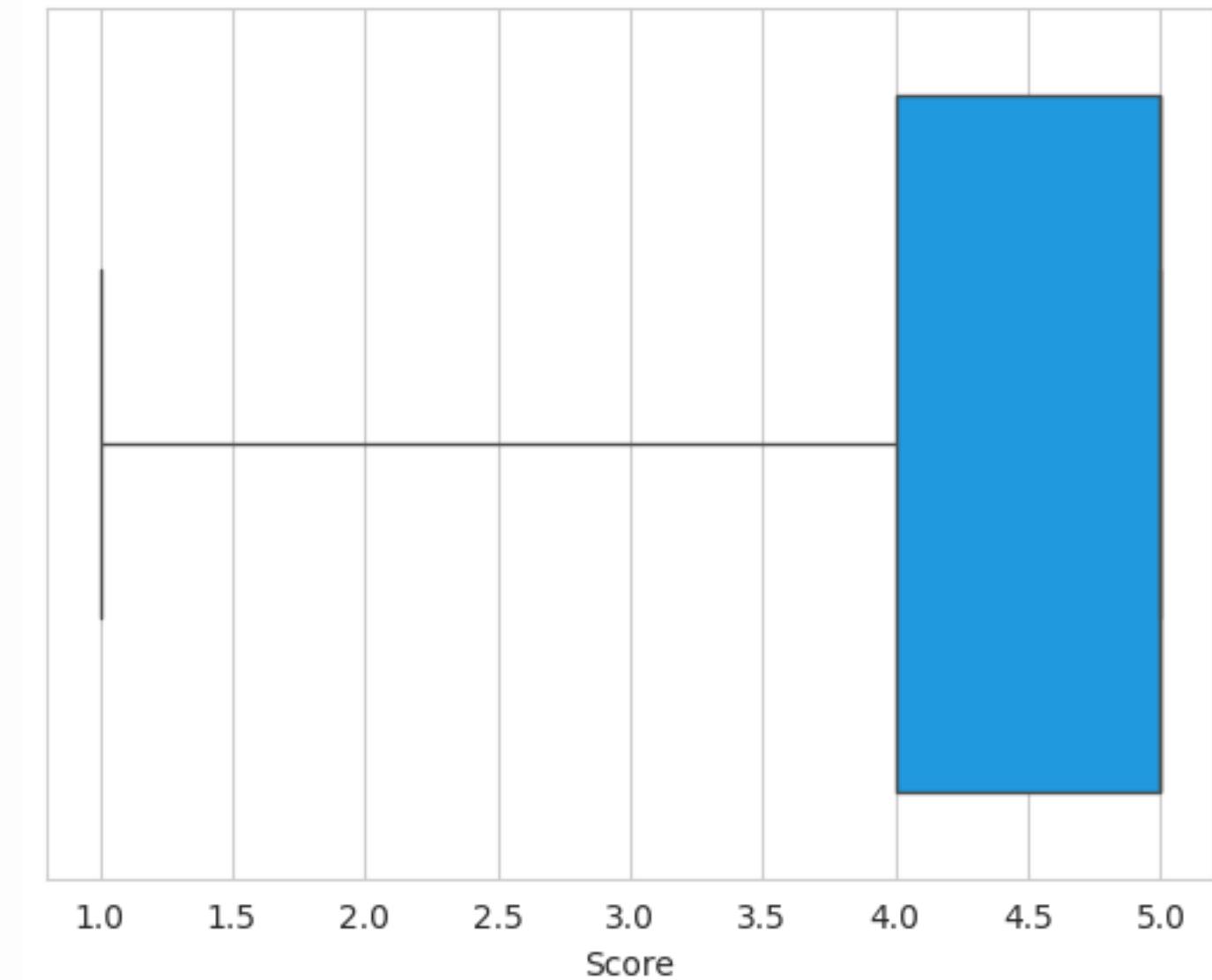


Data Exploration

These graph showing how many reviews get a specific score

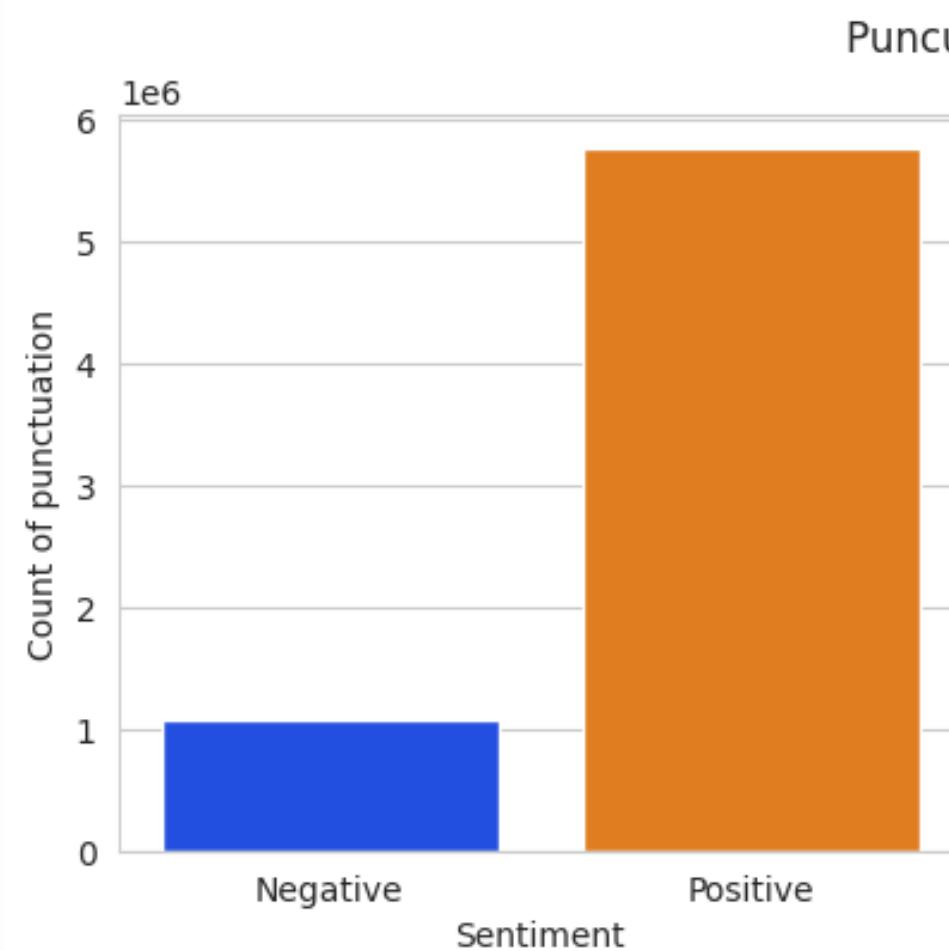


Box-Plot gives the concentration
data distribution

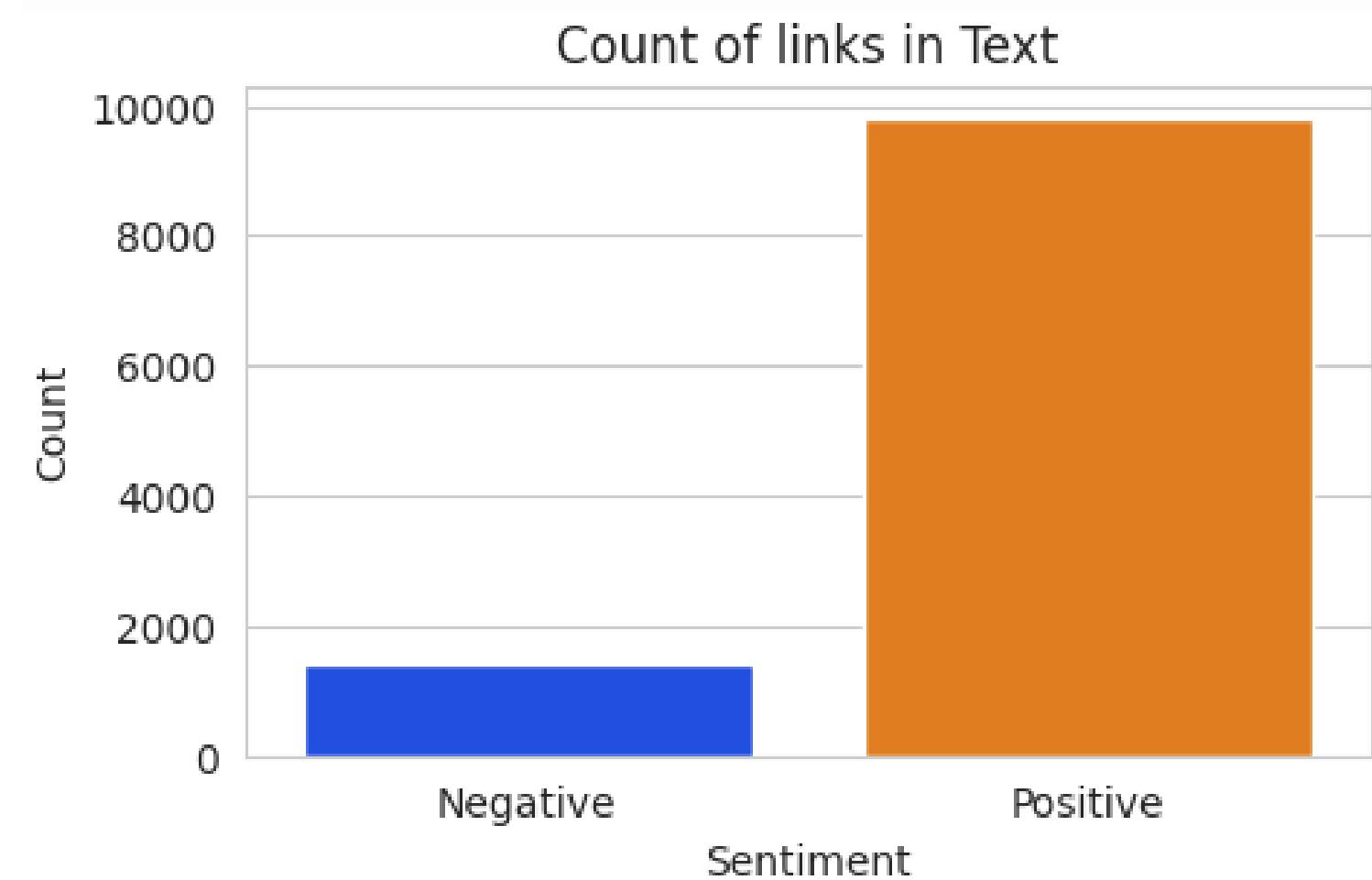


Data Exploration

Punctuation in text



Calculate count of links



positive
negative

15.72%
84.28%

Data Cleaning

Removing Duplicates of the Data

```
# Display duplicated rows and their counts
duplicated_data = data[data[["Text", "Score"]].duplicated(keep=False)] # Changed Score to Score
duplicated_counts = duplicated_data[["Text", "Score"]].value_counts() # Changed Score to Score
duplicated_counts

# Drop duplicates based on 'Text' and 'Score' columns
data = data.drop_duplicates(subset=["Text", "Score"])
```

Dropping null values

```
unique_products = len(data["ProductId"].unique())
print(f"Number of Unique Product IDs: {unique_products}")
```

Number of Unique Product IDs: 67563

Data Cleaning

Removing Unnecessary Columns and Data Sampling :

- to **reduce processing time and make it work for slow systems too. You can take all the 568k records if your system is efficient enough**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 568454 entries, 0 to 568453
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
---  --  
 0   Id               568454 non-null   int64  
 1   ProductId        568454 non-null   object  
 2   UserId            568454 non-null   object  
 3   ProfileName       568428 non-null   object  
 4   HelpfulnessNumerator  568454 non-null   int64  
 5   HelpfulnessDenominator 568454 non-null   int64  
 6   Score             568454 non-null   int64  
 7   Time              568454 non-null   int64  
 8   Summary            568427 non-null   object  
 9   Text               568454 non-null   object  
dtypes: int64(5), object(5)
memory usage: 43.4+ MB
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 568454 entries, 0 to 568453
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype  
---  --  
 0   ProductId        568454 non-null   object  
 1   Score             568454 non-null   int64  
 2   Time              568454 non-null   int64  
 3   Text               568454 non-null   object  
dtypes: int64(2), object(2)
memory usage: 17.3+ MB
```

```
Score
1   12500
2   12500
3   8500
4   8500
5   8500
```



```
count
rating
1    25500
0    25000
```

EDA

WordCloud of Positive and Negative Reviews



common words - Top 8

Positive :

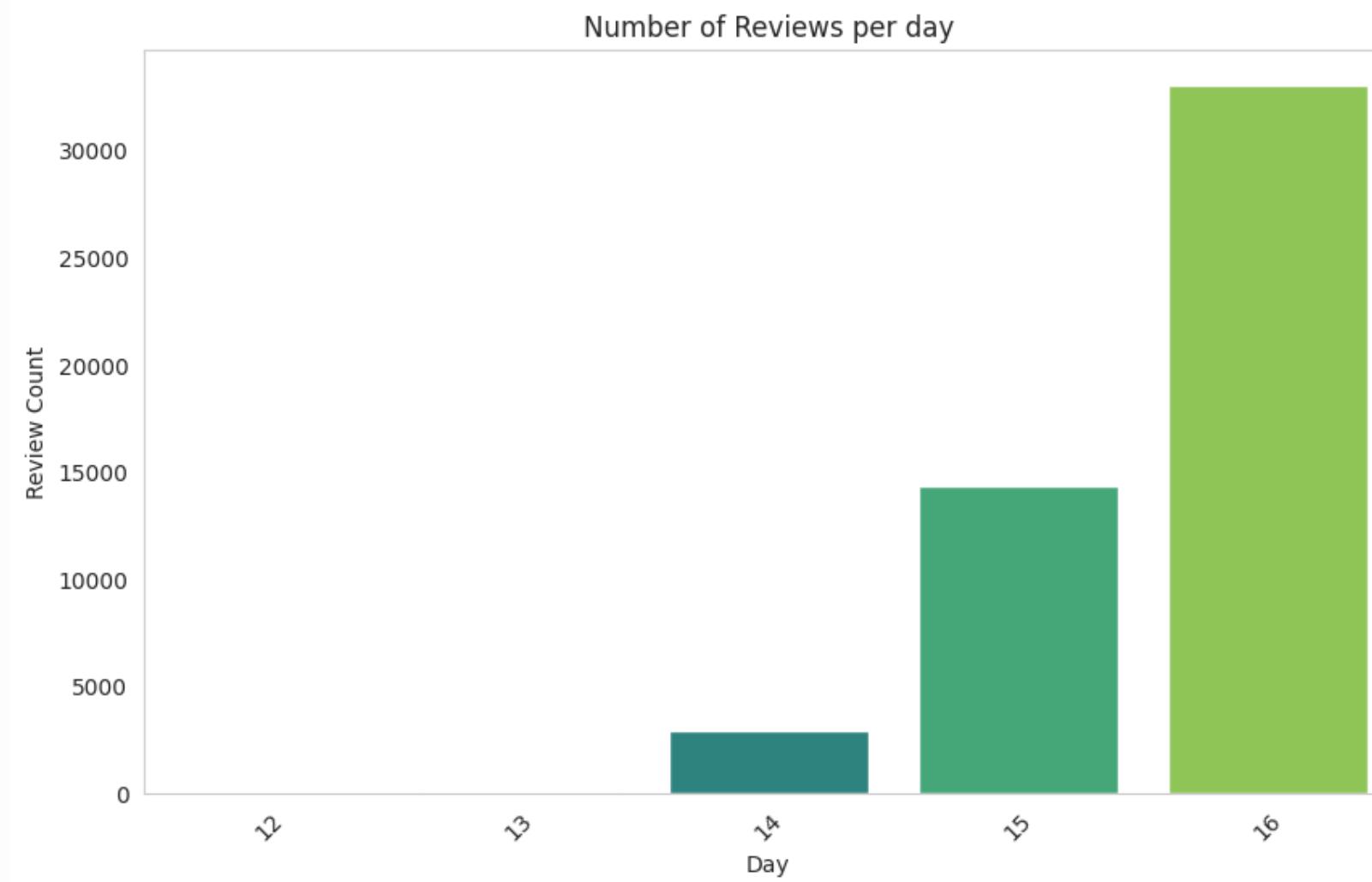
```
[('br', 375500), ('br br', 150429), ('like', 143241), ('good', 124033), ('great', 108981), ('just', 100612), ('taste', 97969),
```

Negative :

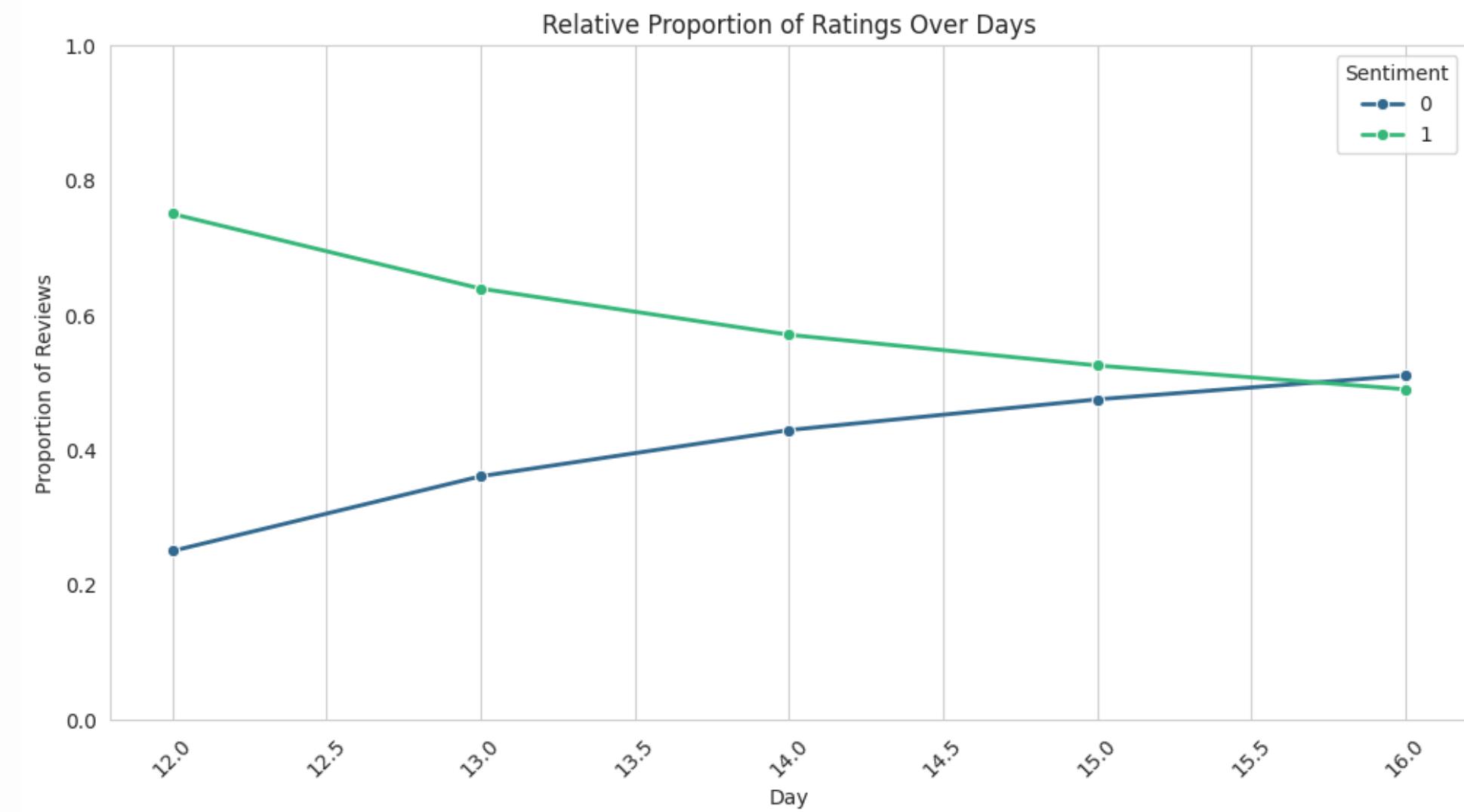
[('br', 69945), ('like', 30301), ('br br', 27916), ('product', 24984), ('taste', 22602), ('just', 18809), ('good', 14768), ('co

Reviews Trend Analysis

Number of reviews for each Day



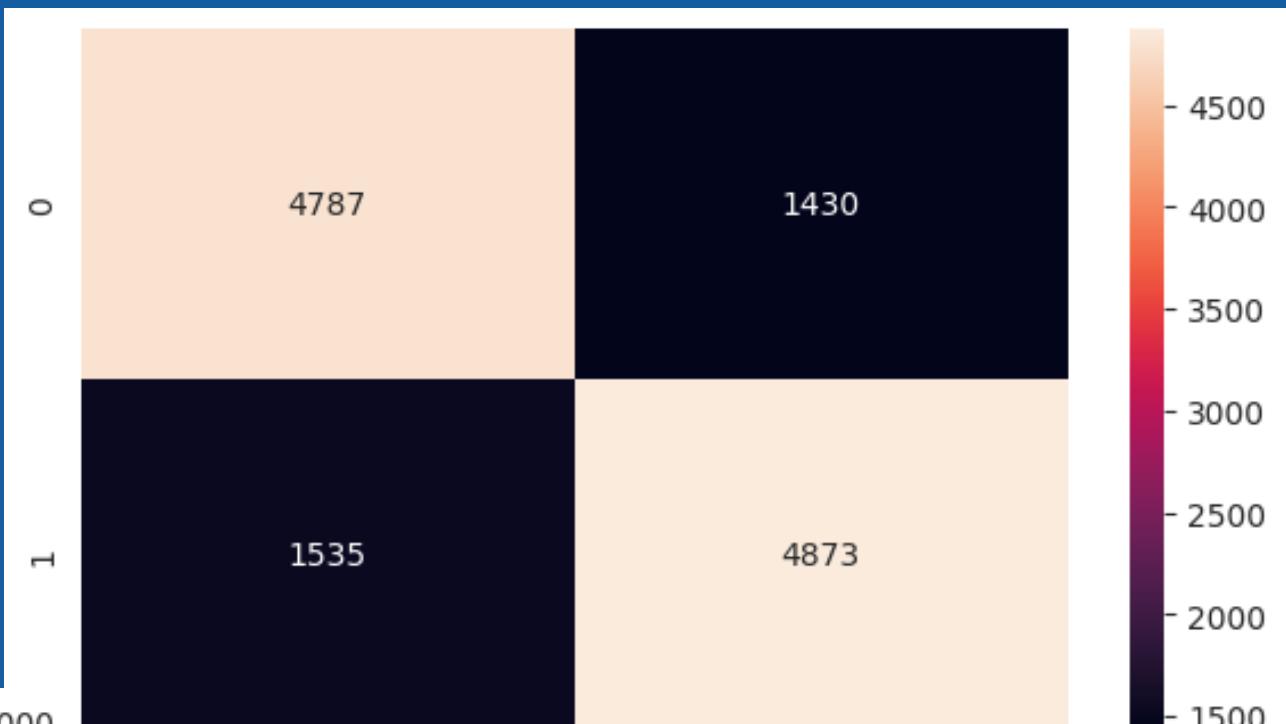
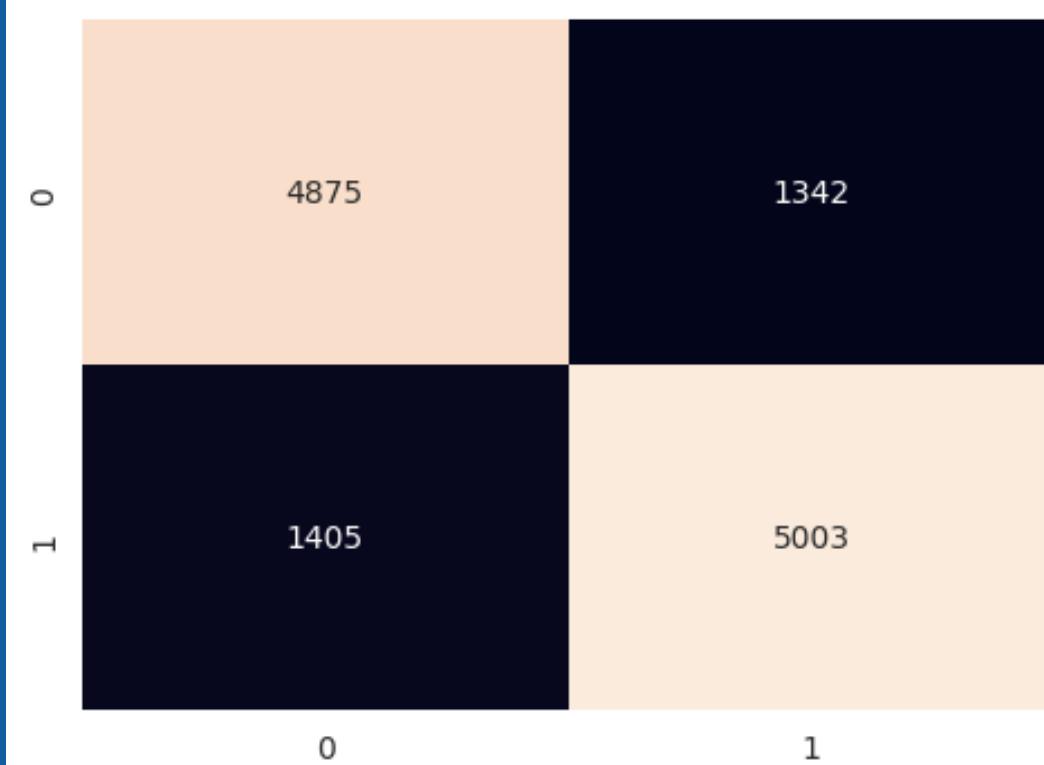
Line Chart



Models Training - TFIDF

Confusion matrix

Classifier: Logistic_Regression
Train Accuracy: 0.8374125412541255
Test Accuracy: 0.7824158415841584



Classifier: XGBoost
Train Accuracy: 0.8538613861386138
Test Accuracy: 0.7568316831683168

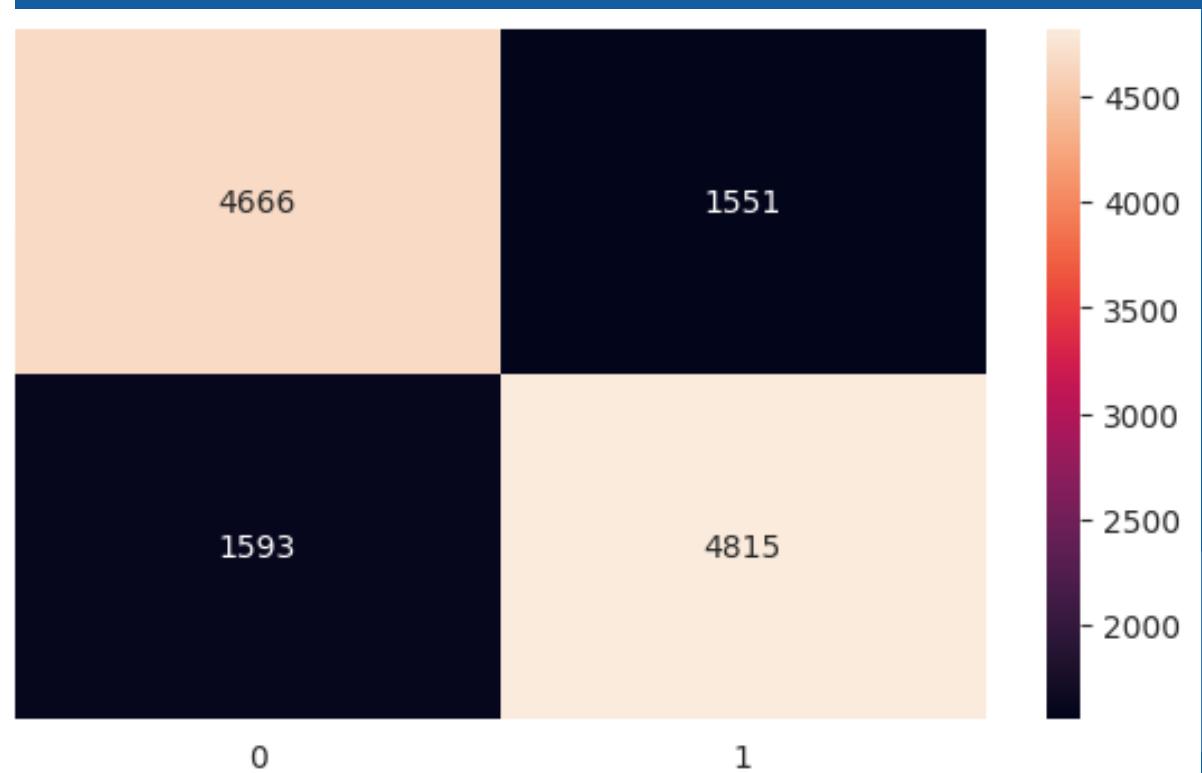
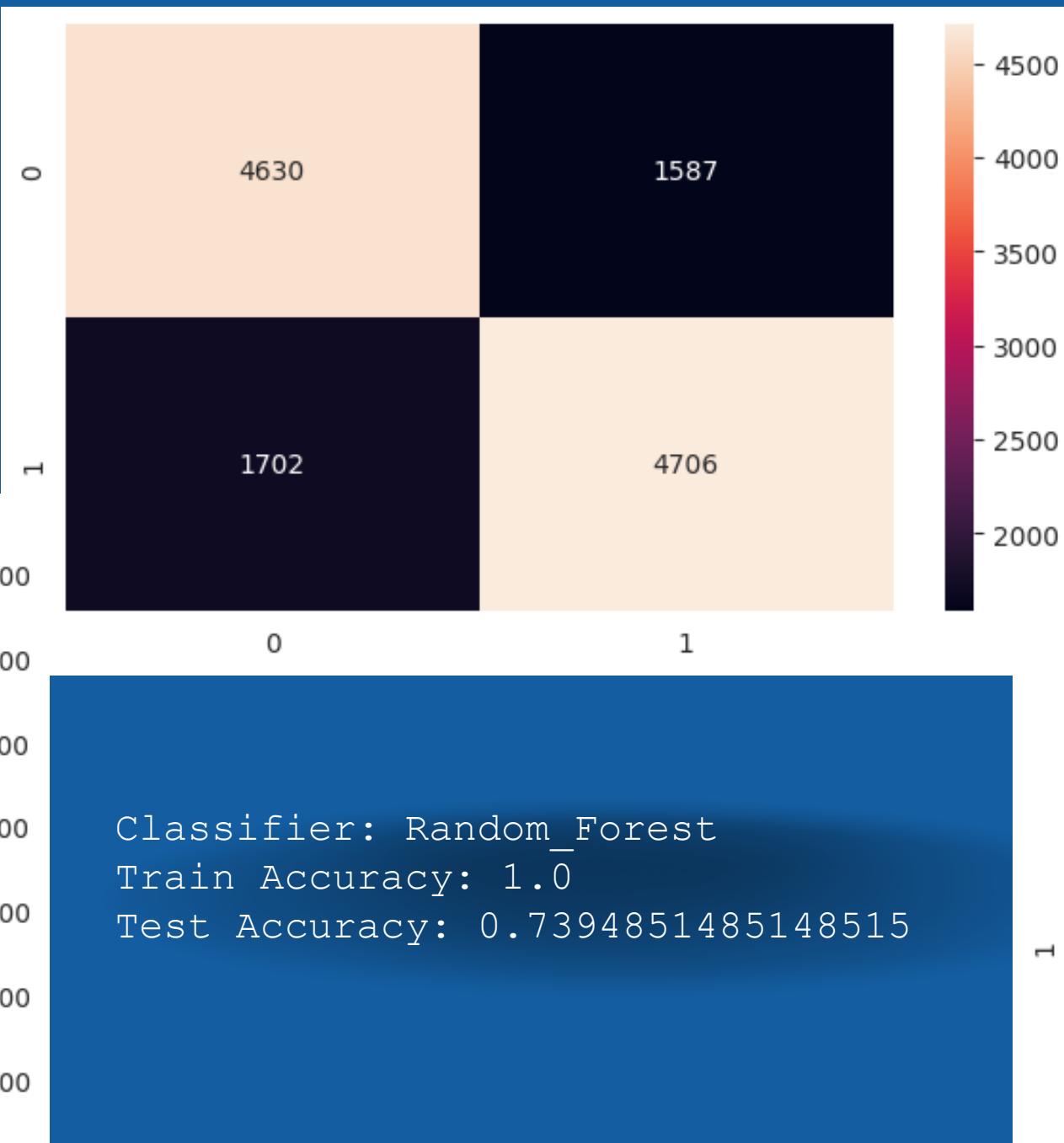
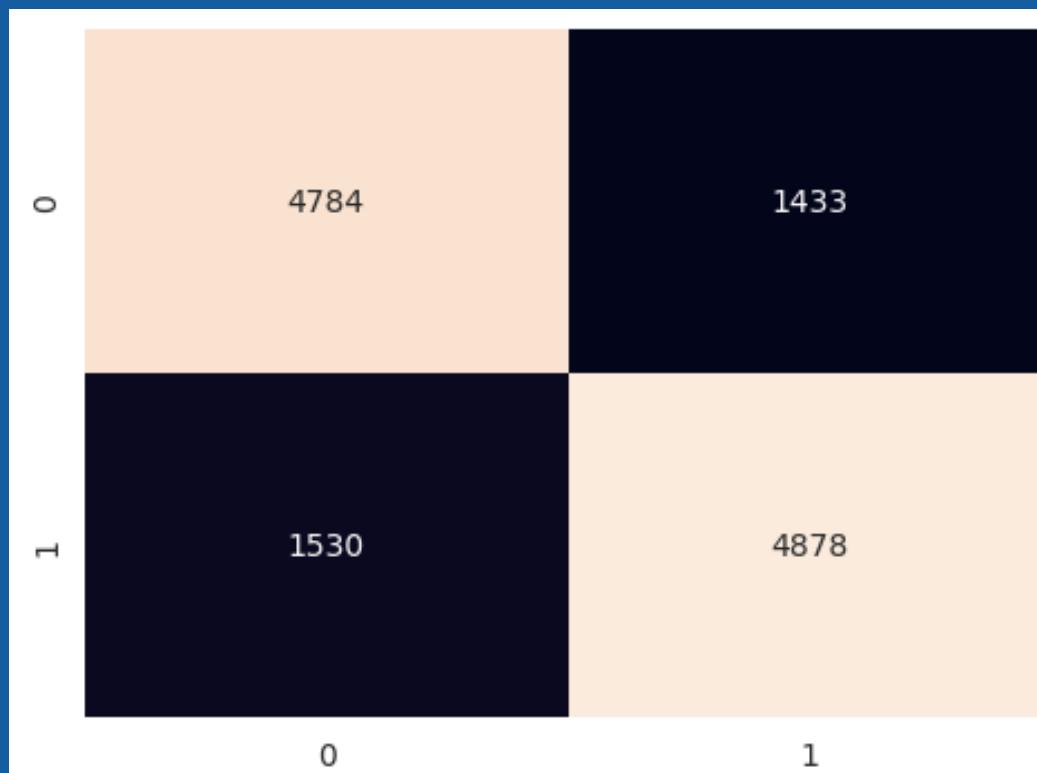
Classifier: Random_Forest
Train Accuracy: 1.0
Test Accuracy: 0.7651485148514852



Models Training - Word2Vec

Confusion matrix

Classifier: Logistic_Regression
Train Accuracy: 0.7661782178217822
Test Accuracy: 0.7653069306930693



Models Training - LSTM

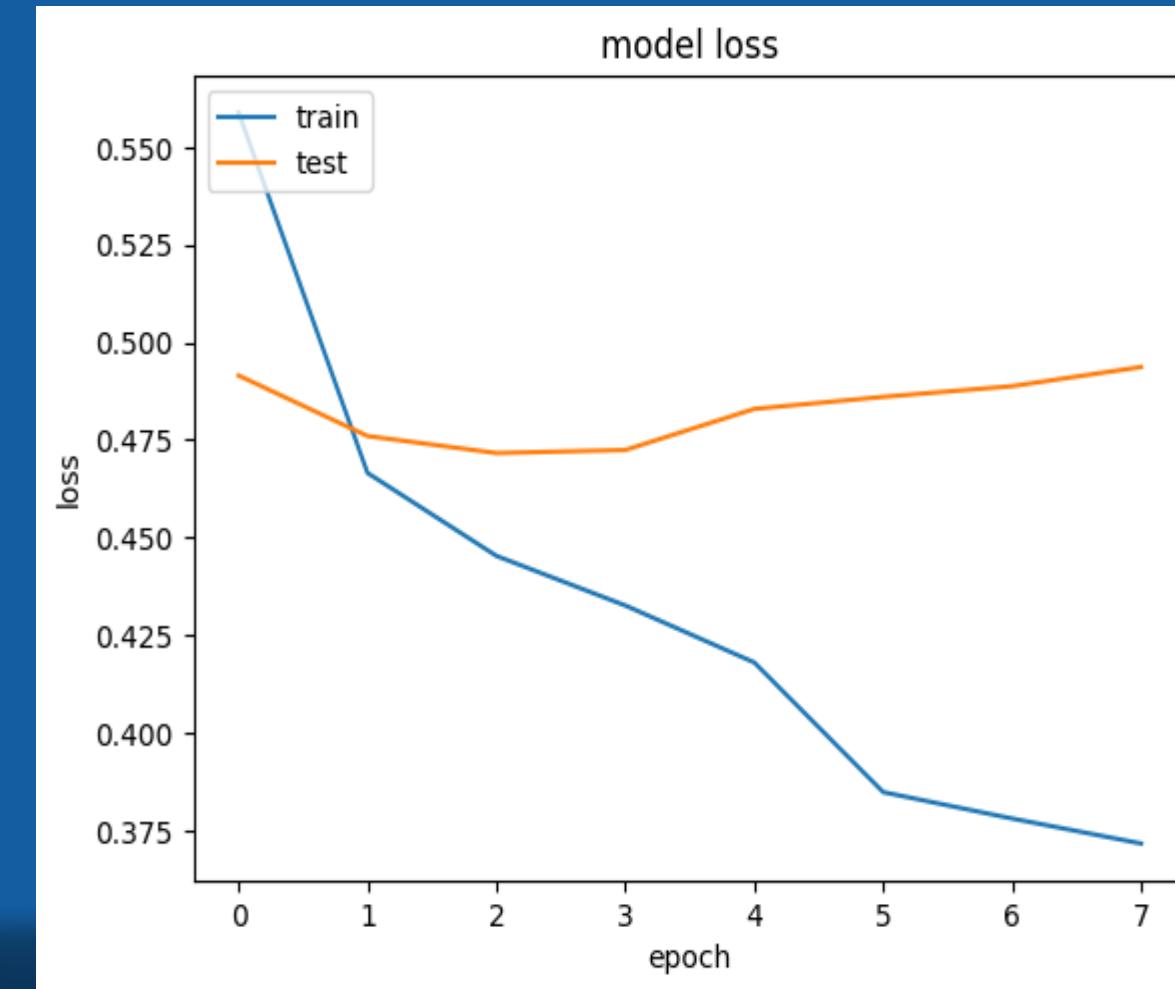
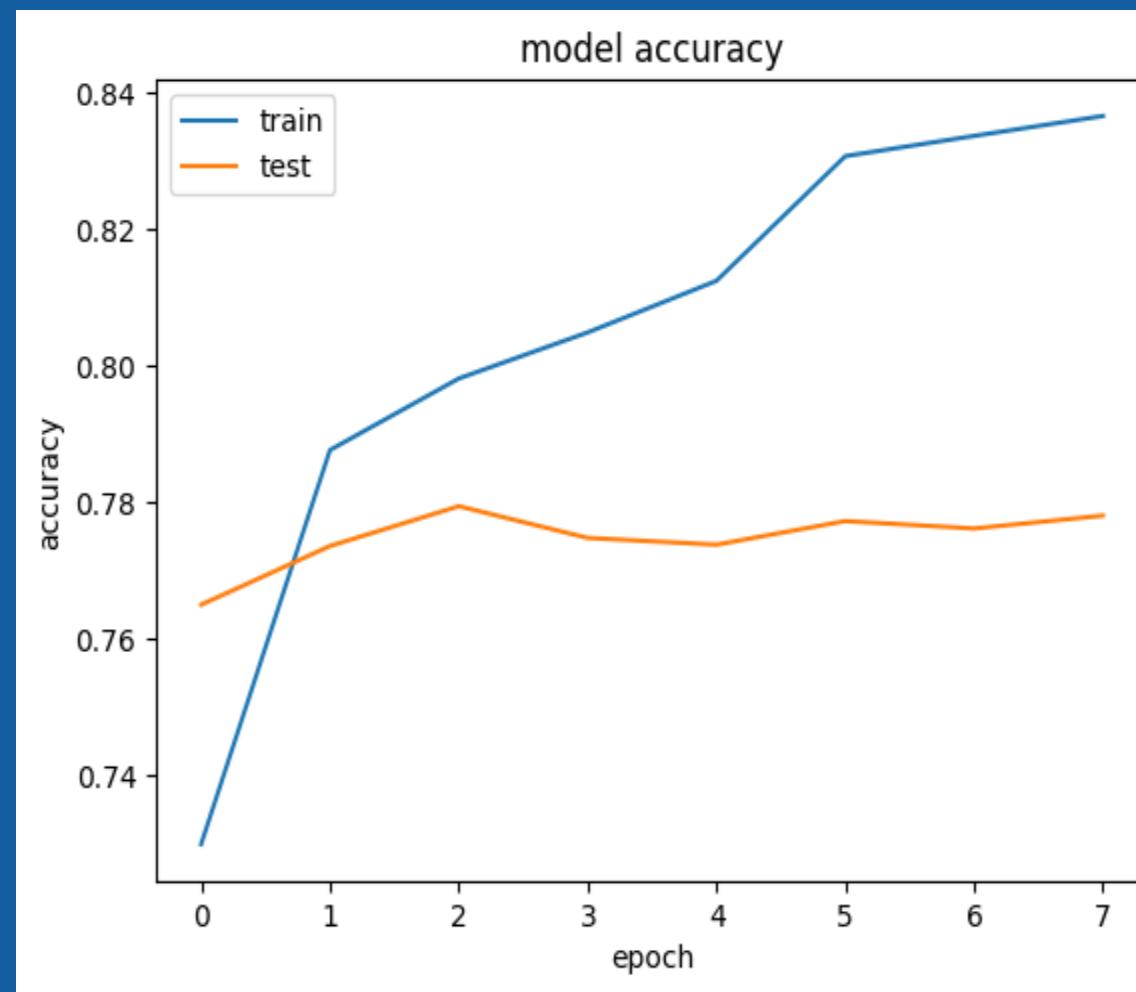
```
model=Sequential()  
model.add(Embedding(input_dim=voca_size,output_dim=embedding_size,input_length=max_len))  
model.add(LSTM(256,recurrent_dropout=0.3,dropout=0.3))  
model.add(Dropout(0.5))  
model.add(Dense(1,activation='sigmoid',kernel_regularizer=regularizers.l2(0.01)))
```

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
embedding (Embedding)	(None, 150, 200)	400000
lstm (LSTM)	(None, 256)	467968
dropout (Dropout)	(None, 256)	0
dense (Dense)	(None, 1)	257

dense (Dense) (None, 1) 257

Model Accuracy - LSTM



Final Test Accuracy: 77.94%

Final Train Accuracy: 83.69%

Models Training & Accuracy - DistilBert

[7575/7575 28:23, Epoch 3/3]

Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.064600	0.697429	0.870495	0.870500
2	0.033100	0.741822	0.871287	0.871284
3	0.031400	0.870741	0.874851	0.874857

```
8]: TrainOutput(global_step=7575, training_loss=0.04511223522349946, metrics={'train_runtime': 1703.9447, 'train_samples_per_second': 71.129, 'train_steps_per_second': 4.446, 'total_flos': 8027524358553600.0, 'train_loss': 0.04511223522349946, 'epoch': 3.0})
```

[+ Code](#) [+ Markdown](#)

```
3]:  
# Save the model  
model.save_pretrained('./DistilBert_Model')
```

MLOPS

IMDB MLflow

Logistic Regression EXPERIMENT

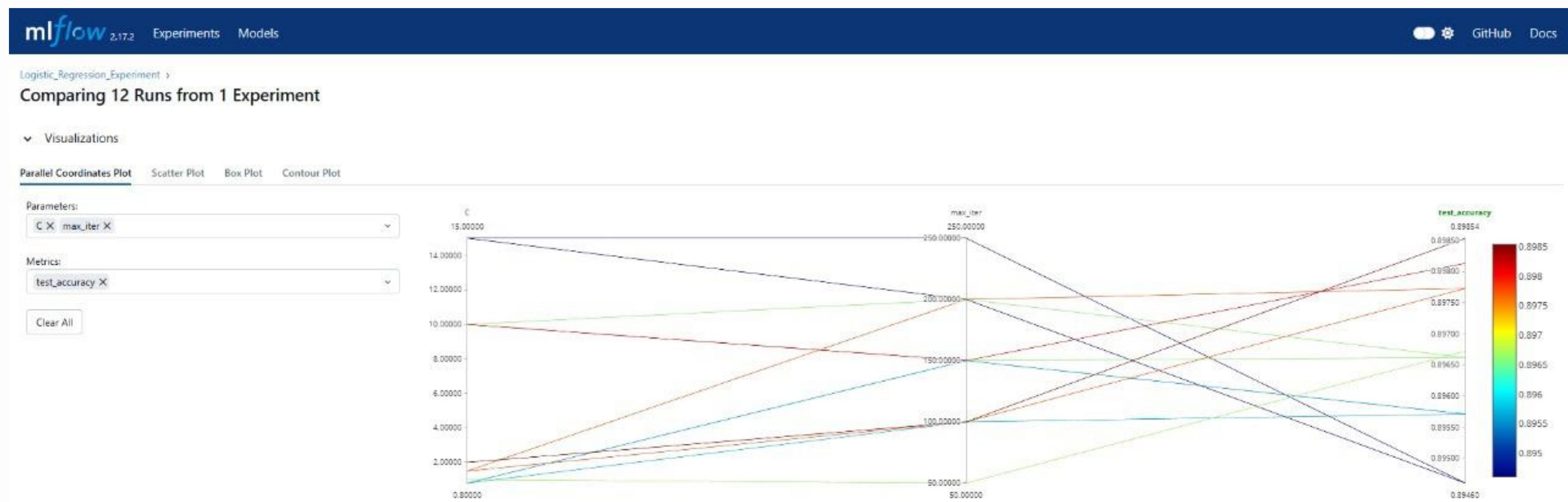
The screenshot shows the mlflow UI for a Logistic Regression Experiment. The top navigation bar includes links for mlflow 2.17.2, Experiments, Models, GitHub, and Docs. The main title is "Logistic_Regression_Experiment". Below the title, there are tabs for Runs, Evaluation, Experimental (which is selected), and Traces. A search bar at the top allows filtering by "metrics.rmse < 1 and params.model = 'tree'". The experimental table has columns for Run Name, Created (with a dropdown for sorting), Duration, Source, Models, and Metrics (auc, f1_score, test_accuracy, train_accuracy). It also includes a Parameters section with columns for C, data_source, max_iter, and solver. The table lists 12 matching runs, each with a unique color and timestamp.

Run Name	Created	Duration	Source	Models	Metrics	Parameters
Logistic_Regression	7 seconds ago	4.5s	c:\Users\...	-	auc: 0.962393456..., f1_score: 0.900607723..., test_accuracy: 0.897730711..., train_accuracy: 0.943822491..., C: 1.5	IMDB_DataSet, 200, lbfgs
Logistic_Regression	21 seconds ago	4.7s	c:\Users\...	-	auc: 0.960658743..., f1_score: 0.897054477..., test_accuracy: 0.894604135..., train_accuracy: 0.990695915..., C: 15	IMDB_DataSet, 250, lbfgs
Logistic_Regression	34 seconds ago	3.7s	c:\Users\...	-	auc: 0.961251280..., f1_score: 0.899745447..., test_accuracy: 0.896722138..., train_accuracy: 0.932602118..., C: 1	IMDB_DataSet, 50, lbfgs
Logistic_Regression	47 seconds ago	3.9s	c:\Users\...	-	auc: 0.961746179..., f1_score: 0.898925155..., test_accuracy: 0.896621280..., train_accuracy: 0.983055975..., C: 10	IMDB_DataSet, 200, lbfgs
Logistic_Regression	1 minute ago	4.0s	c:\Users\...	-	auc: 0.961746179..., f1_score: 0.898925155..., test_accuracy: 0.896621280..., train_accuracy: 0.983055975..., C: 10	IMDB_DataSet, 150, lbfgs
Logistic_Regression	1 minute ago	5.1s	c:\Users\...	-	auc: 0.961835880..., f1_score: 0.900551398..., test_accuracy: 0.898134140..., train_accuracy: 0.985930408..., C: 10	IMDB_DataSet, 150, liblinear
Logistic_Regression	5 minutes ago	3.8s	c:\Users\...	-	auc: 0.960547554..., f1_score: 0.898806028..., test_accuracy: 0.895713565..., train_accuracy: 0.929778113..., C: 0.8	IMDB_DataSet, 150, liblinear
Logistic_Regression	38 minutes ago	3.5s	c:\Users\...	-	auc: 0.960547554..., f1_score: 0.898806028..., test_accuracy: 0.895713565..., train_accuracy: 0.929778113..., C: 0.8	IMDB_DataSet, 150, liblinear
Logistic_Regression	38 minutes ago	3.8s	c:\Users\...	-	auc: 0.960547554..., f1_score: 0.898806028..., test_accuracy: 0.895713565..., train_accuracy: 0.929778113..., C: 0.8	IMDB_DataSet, 100, liblinear
Logistic_Regression	39 minutes ago	3.8s	c:\Users\...	-	auc: 0.962862349..., f1_score: 0.901081612..., test_accuracy: 0.898537569..., train_accuracy: 0.950680786..., C: 2	IMDB_DataSet, 100, lbfgs
Logistic_Regression	40 minutes ago	4.0s	c:\Users\...	-	auc: 0.962393456..., f1_score: 0.900607723..., test_accuracy: 0.897730711..., train_accuracy: 0.943822491..., C: 1.5	IMDB_DataSet, 100, lbfgs
Logistic_Regression	47 minutes ago	4.5s	c:\Users\...	-	auc: 0.960658743..., f1_score: 0.897054477..., test_accuracy: 0.894604135..., train_accuracy: 0.990695915..., C: 15	IMDB_DataSet, 200, lbfgs

MLOPS

IMDB MLflow

Logistic Regression Comparing Runs



MLOPS

IMDB MLflow

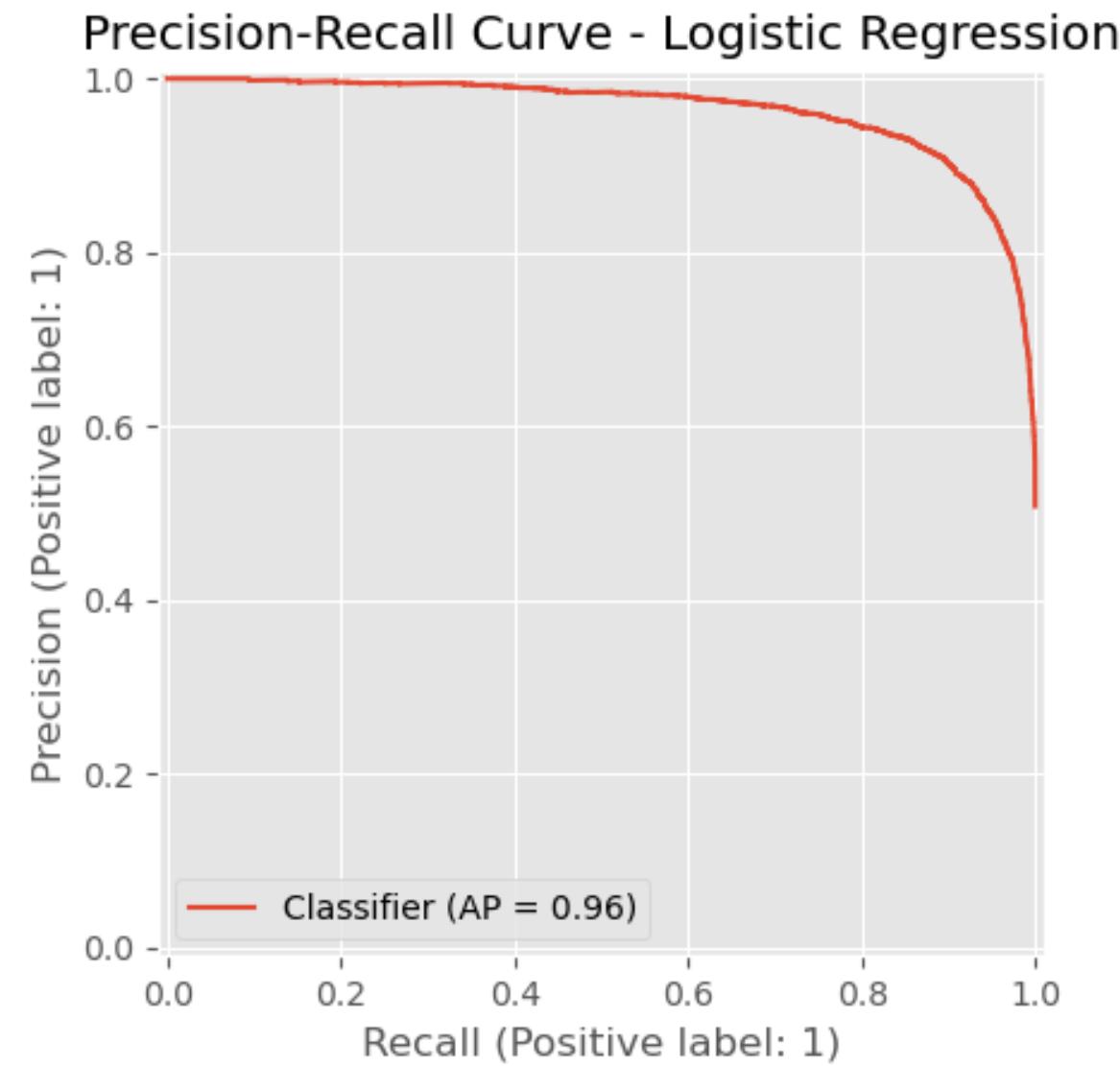
Logistic Regression Confusion Matrix



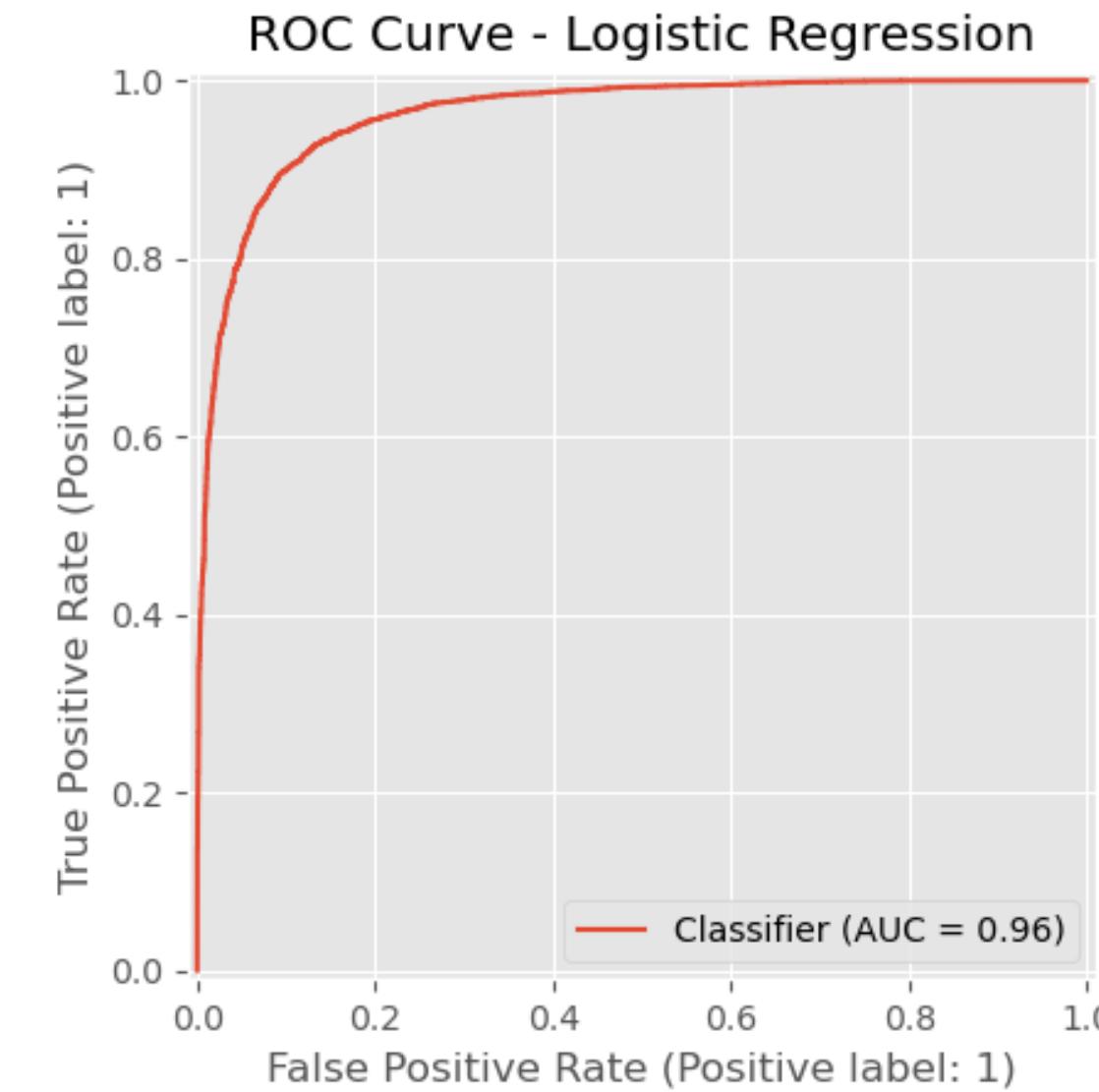
MLOPS

IMDB MLflow

Logistic Regression
Precision Recall Curve



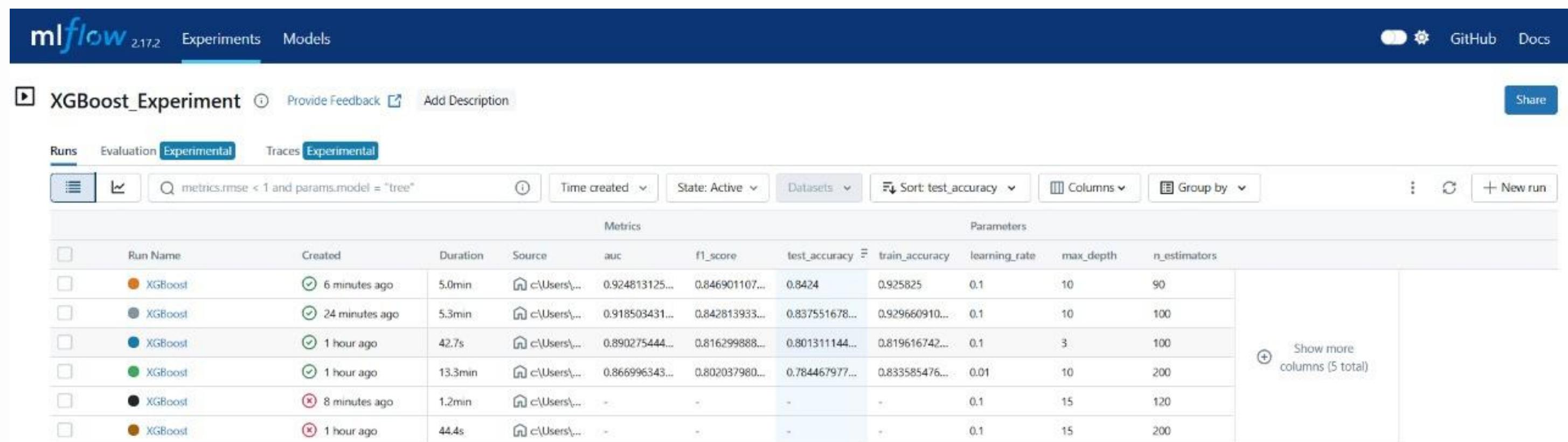
Logistic Regression
ROC Curve



MLOPS

IMDB MLflow

XGBoost Experiment



The screenshot shows the mlflow UI interface for an XGBoost experiment. The top navigation bar includes links for mlflow 2.17.2, Experiments, Models, GitHub, and Docs. The main title is "XGBoost_Experiment". Below the title, there are tabs for Runs, Evaluation, Experimental (which is selected), Traces, and Experimental. A search bar filters results by "metrics.rmse < 1 and params.model = 'tree'". The table displays experimental runs with columns for Run Name, Created, Duration, Source, and various metrics and parameters. The first six rows are visible, with a "Show more columns (5 total)" link.

	Metrics						Parameters				
	Run Name	Created	Duration	Source	auc	f1_score	test_accuracy	train_accuracy	learning_rate	max_depth	n_estimators
<input type="checkbox"/>	XGBoost	6 minutes ago	5.0min	c:\Users\...	0.924813125...	0.846901107...	0.8424	0.925825	0.1	10	90
<input type="checkbox"/>	XGBoost	24 minutes ago	5.3min	c:\Users\...	0.918503431...	0.842813933...	0.837551678...	0.929660910...	0.1	10	100
<input type="checkbox"/>	XGBoost	1 hour ago	42.7s	c:\Users\...	0.890275444...	0.816299888...	0.801311144...	0.819616742...	0.1	3	100
<input type="checkbox"/>	XGBoost	1 hour ago	13.3min	c:\Users\...	0.866996343...	0.802037980...	0.784467977...	0.833585476...	0.01	10	200
<input type="checkbox"/>	XGBoost	8 minutes ago	1.2min	c:\Users\...	-	-	-	-	0.1	15	120
<input type="checkbox"/>	XGBoost	1 hour ago	44.4s	c:\Users\...	-	-	-	-	0.1	15	200

MLOPS

IMDB MLflow

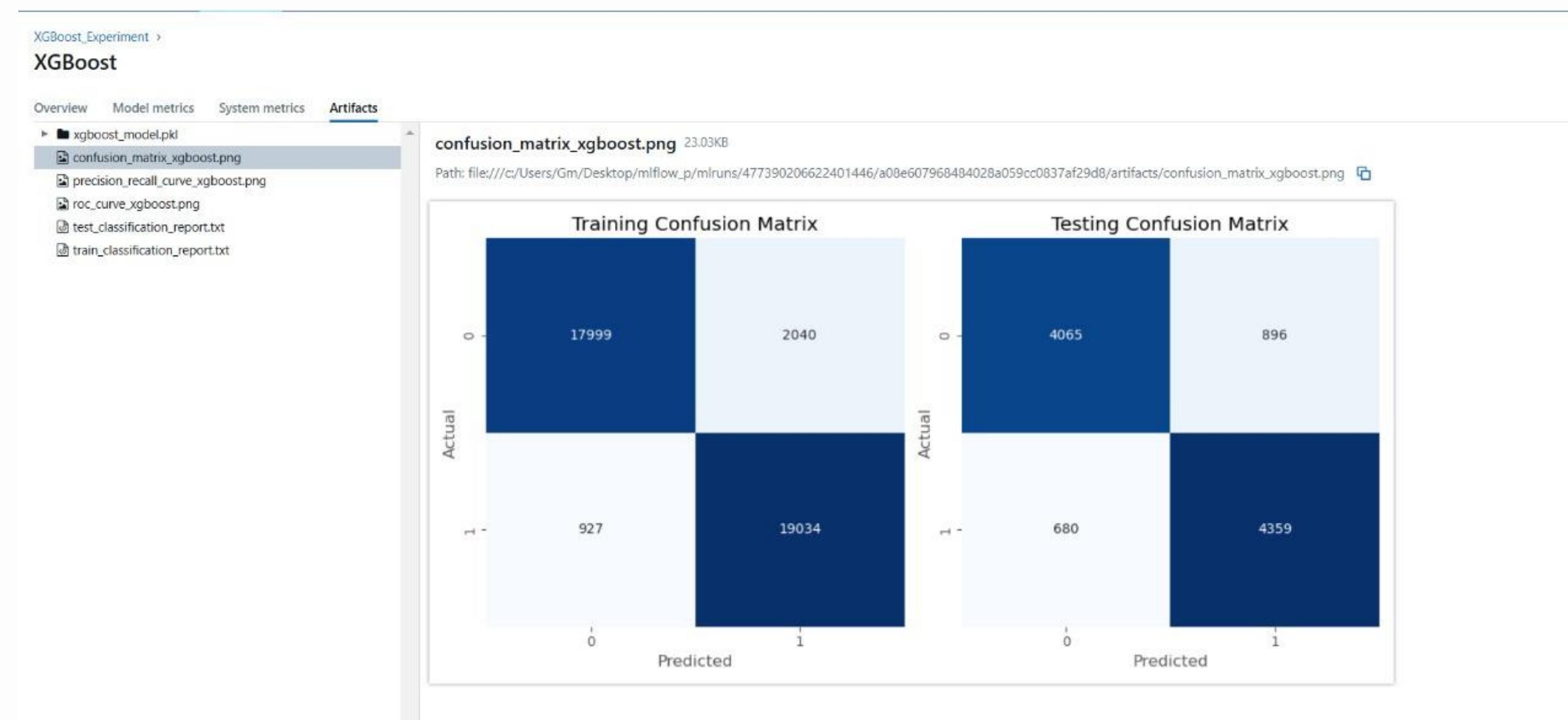
XGBoost Comparing runs



MLOPS

IMDB MLflow

XGBoost Confusion Matrix

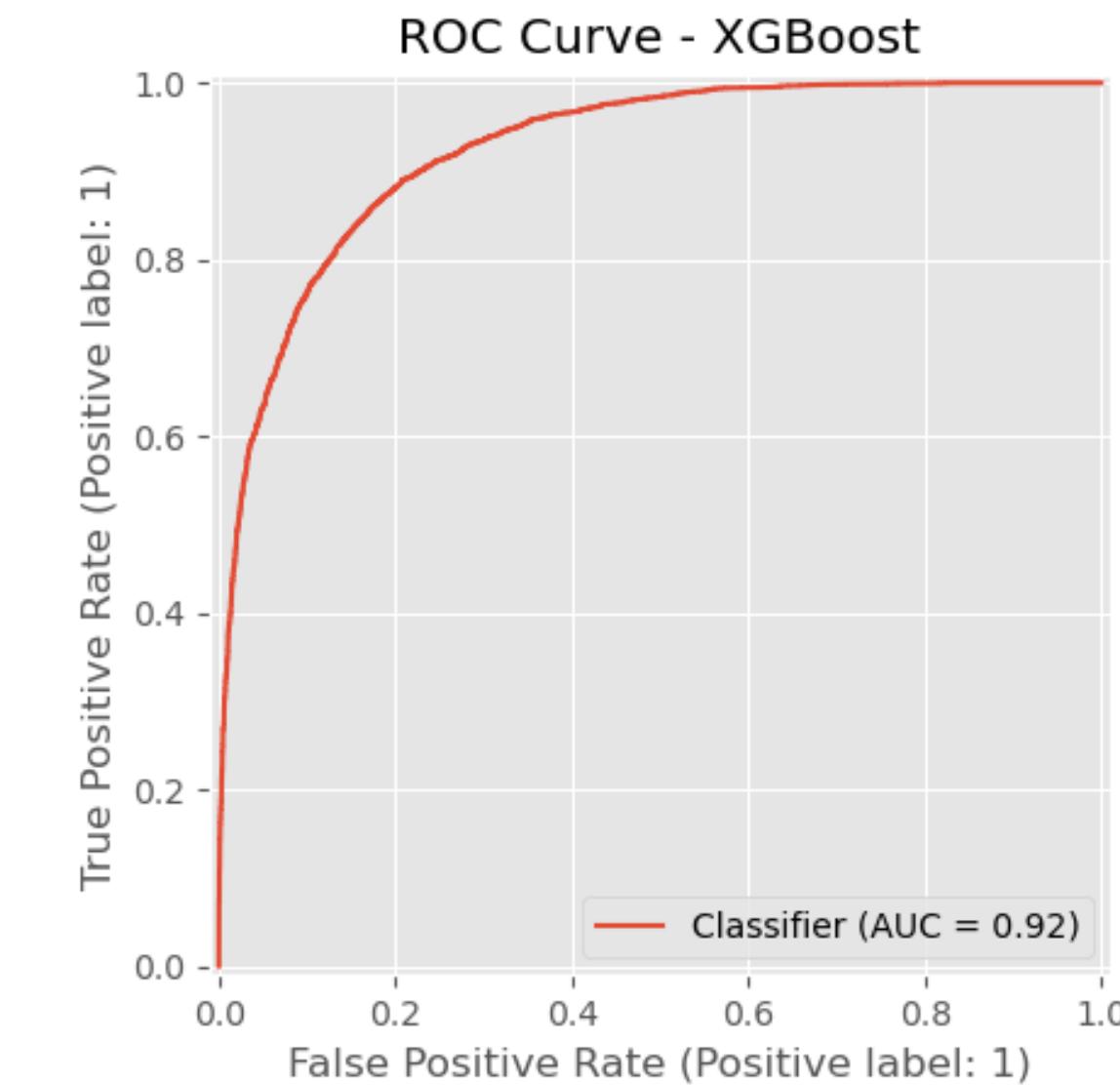
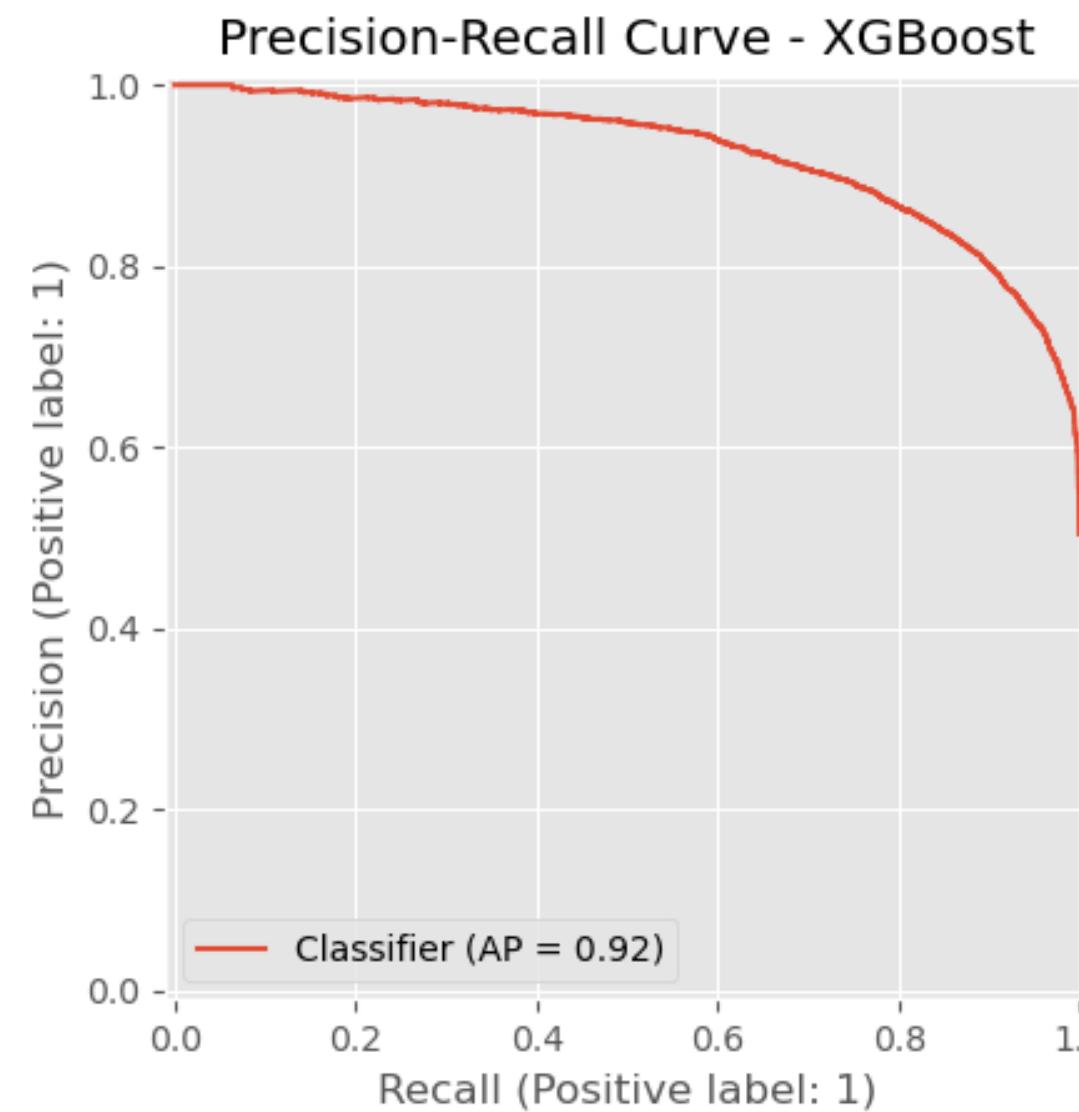


MLOPS

IMDB MLflow

XGBoost Precision Recall Curve

XGBoost ROC



MLOPS

IMDB MLflow

Random Forest Experiment

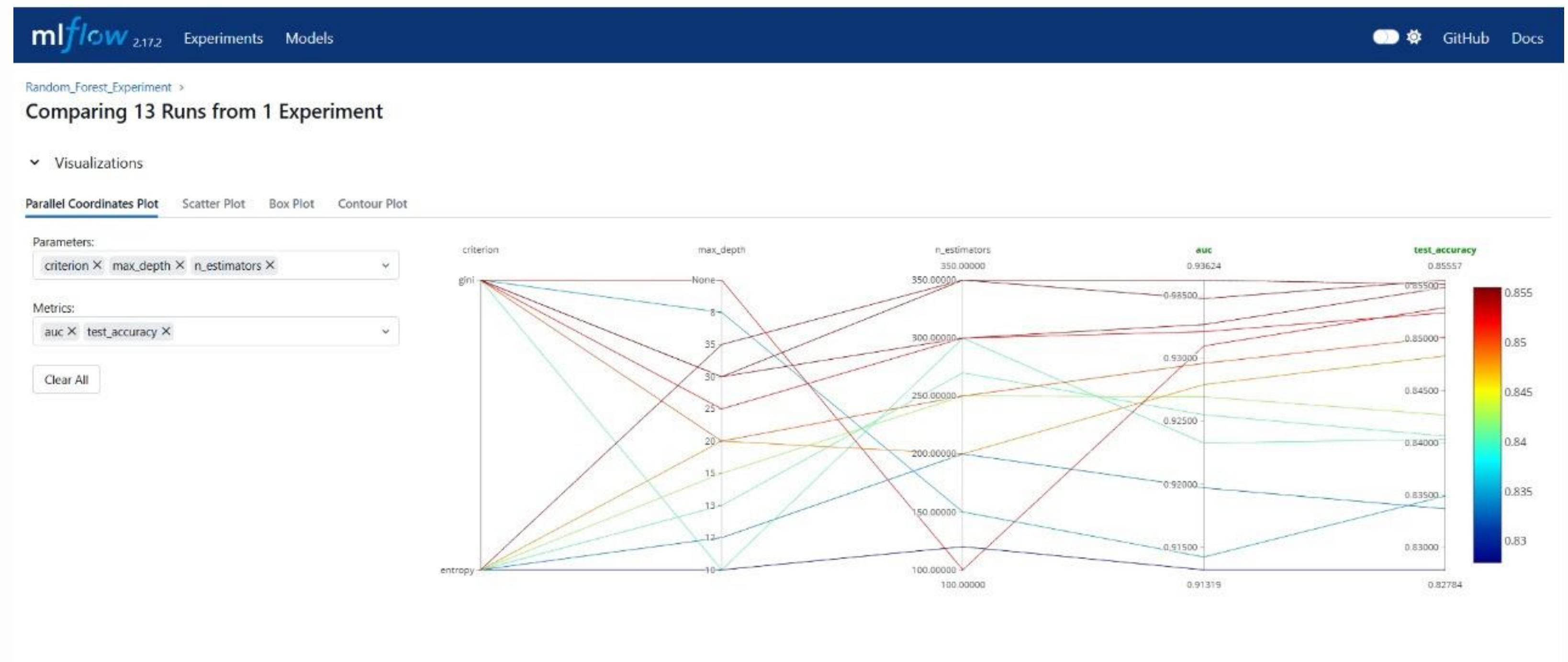
The screenshot shows the MLflow UI for the "Random_Forest_Experiment". The top navigation bar includes links for mlflow (2172), Experiments, Models, and GitHub/Docs. The main title is "Random_Forest_Experiment". Below the title, there are tabs for Runs, Evaluation, Experimental (which is selected), and Traces. A search bar filters results by "metrics.rmse < 1 and params.model = 'tree'". The table lists 15 runs, each with a checkbox, a colored dot, and a run name. The first run, "Random_Forest" (blue dot), was created 4 minutes ago and has the highest accuracy (0.936235818). The last run, "Random_Forest" (black dot), was created 1 hour ago.

Metrics												Parameters	
	Run Name	Created	Duration	Source	auc	f1_score	test_accuracy	train_accuracy	criterion	max_depth	n_estimators		
<input type="checkbox"/>	Random_Forest	4 minutes ago	3.4min	c:\Users\...\	0.936235818...	0.861470191...	0.855168935...	0.965204236...	entropy	35	350		
<input type="checkbox"/>	Random_Forest	8 minutes ago	2.9min	c:\Users\...\	0.934764062...	0.862148633...	0.855572365...	0.958673726...	gini	30	350		
<input type="checkbox"/>	Random_Forest	11 minutes ago	2.4min	c:\Users\...\	0.932705996...	0.861301204...	0.854866364...	0.956984367...	gini	30	300		
<input type="checkbox"/>	Random_Forest	14 minutes ago	1.9min	c:\Users\...\	0.932136984...	0.859637340...	0.852445789...	0.942208774...	gini	25	300		
<input type="checkbox"/>	Random_Forest	18 minutes ago	1.2min	c:\Users\...\	0.929620607...	0.856812487...	0.850126071...	0.925466464...	gini	20	250		
<input type="checkbox"/>	Random_Forest	19 minutes ago	17.3s	c:\Users\...\	0.914208190...	0.843274293...	0.834896621...	0.860463943...	gini	8	150		
<input type="checkbox"/>	Random_Forest	22 minutes ago	48.1s	c:\Users\...\	0.925525110...	0.850605143...	0.840645486...	0.887392839...	entropy	13	270		
<input type="checkbox"/>	Random_Forest	23 minutes ago	59.7s	c:\Users\...\	0.927928628...	0.856433753...	0.848310640...	0.916767523...	entropy	20	200		
<input type="checkbox"/>	Random_Forest	25 minutes ago	40.0s	c:\Users\...\	0.923273928...	0.849079988...	0.840342914...	0.877357539...	gini	10	300		
<input type="checkbox"/>	Random_Forest	26 minutes ago	51.5s	c:\Users\...\	0.926958609...	0.851711026...	0.842662632...	0.895890065...	entropy	15	250		
<input type="checkbox"/>	Random_Forest	28 minutes ago	32.6s	c:\Users\...\	0.919709145...	0.844301765...	0.833686333...	0.876550680...	entropy	12	200		
<input type="checkbox"/>	Random_Forest	28 minutes ago	17.4s	c:\Users\...\	0.913185629...	0.837598706...	0.827836611...	0.862531517...	entropy	10	120		
<input type="checkbox"/>	Random_Forest	1 hour ago	6.6min	c:\Users\...\	0.930999346...	0.854867608...	0.852950075...	1	gini	None	100		

MLOPS

IMDB MLflow

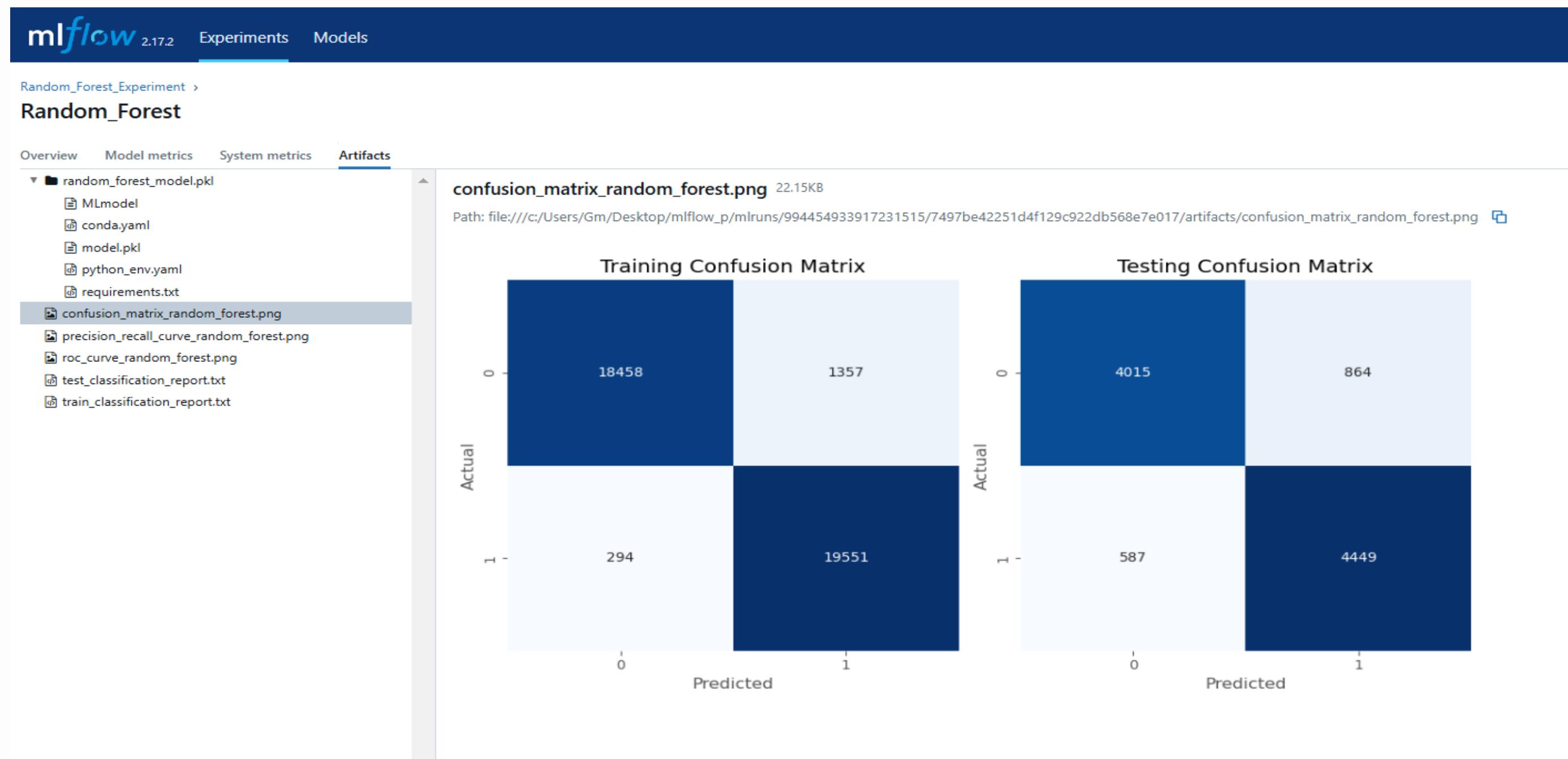
Random Forest Comparing Runs



MLOPS

IMDB MLflow

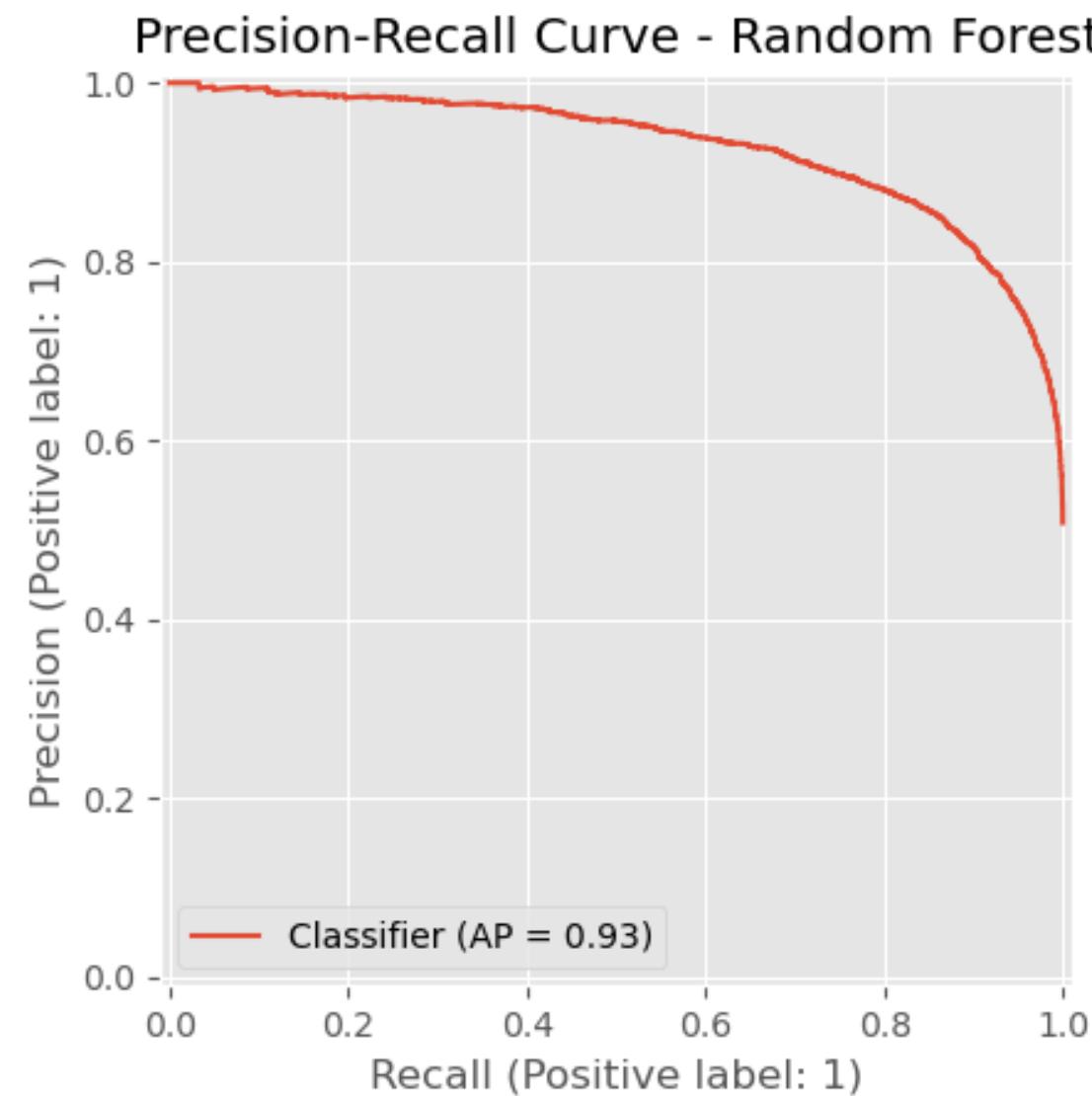
Random Forest Confusion matrix



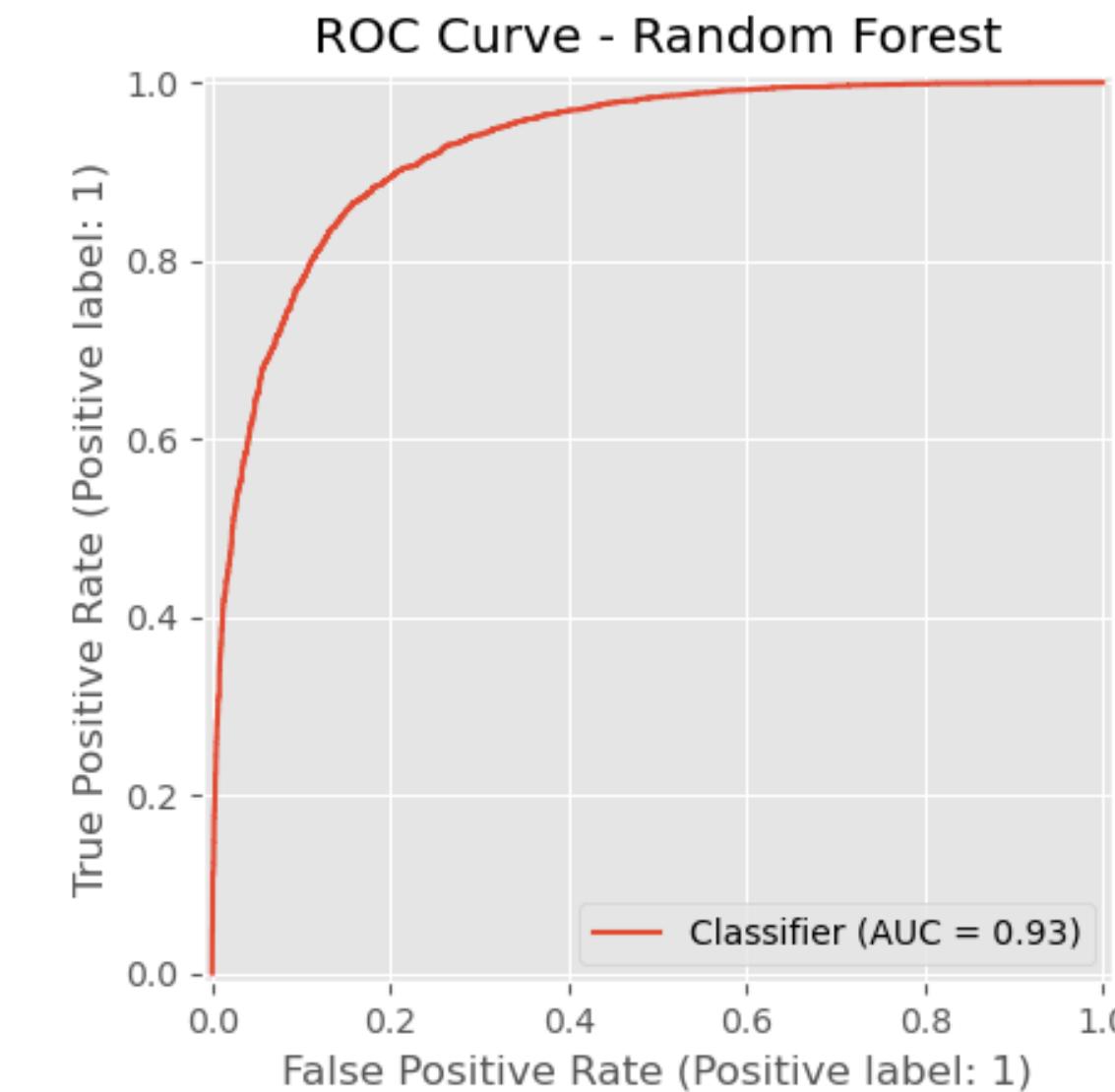
MLOPS

IMDB MLflow

Random Forest Precision Recall Curve



Random Forest Roc Curve



MLOPS

IMDB MLflow

LSTM EXPRESSION

The screenshot shows the MLflow UI interface for an experiment named "LSTM_MODEL_exp". The top navigation bar includes links for "mlflow" (version 2.17.2), "Experiments", "Models", and "GitHub Docs". The main content area displays the experimental runs for this experiment.

The experimental runs table has the following columns:

- Run Name
- Created (sorted by time)
- Duration
- Source
- train_accuracy
- train_loss
- val_accuracy
- val_loss
- batch_size
- epochs
- learning_rate

The table lists eight runs, each with a unique color-coded icon. The first run, "amusing-dog-899", is highlighted with a green dot and has a tooltip showing its creation time as "13 minutes ago". The last run, "youthful-hare-845", is highlighted with a red dot and has a tooltip showing its creation time as "21 hours ago".

At the bottom right of the table, there is a link "Show more columns (3 total)".

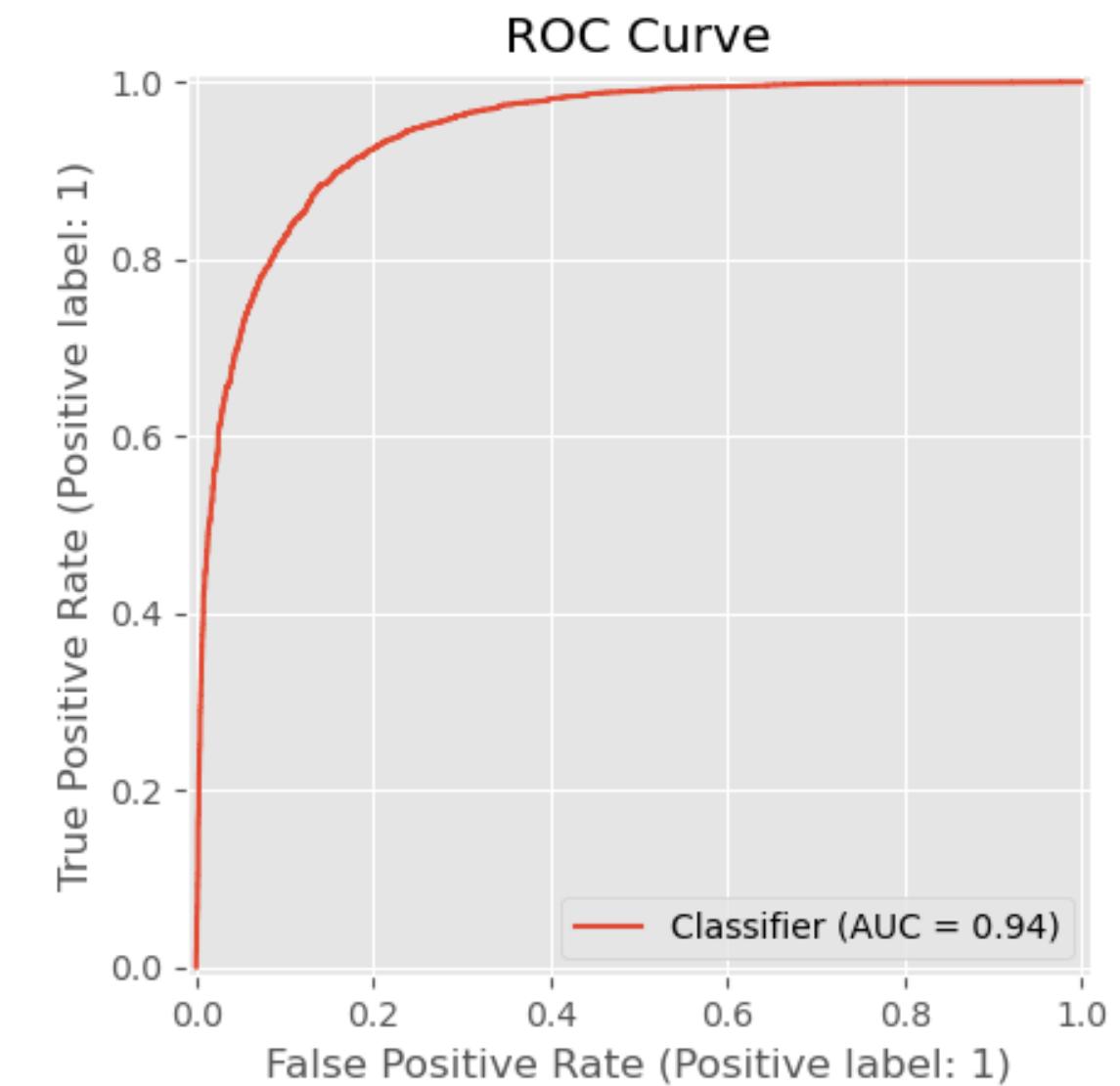
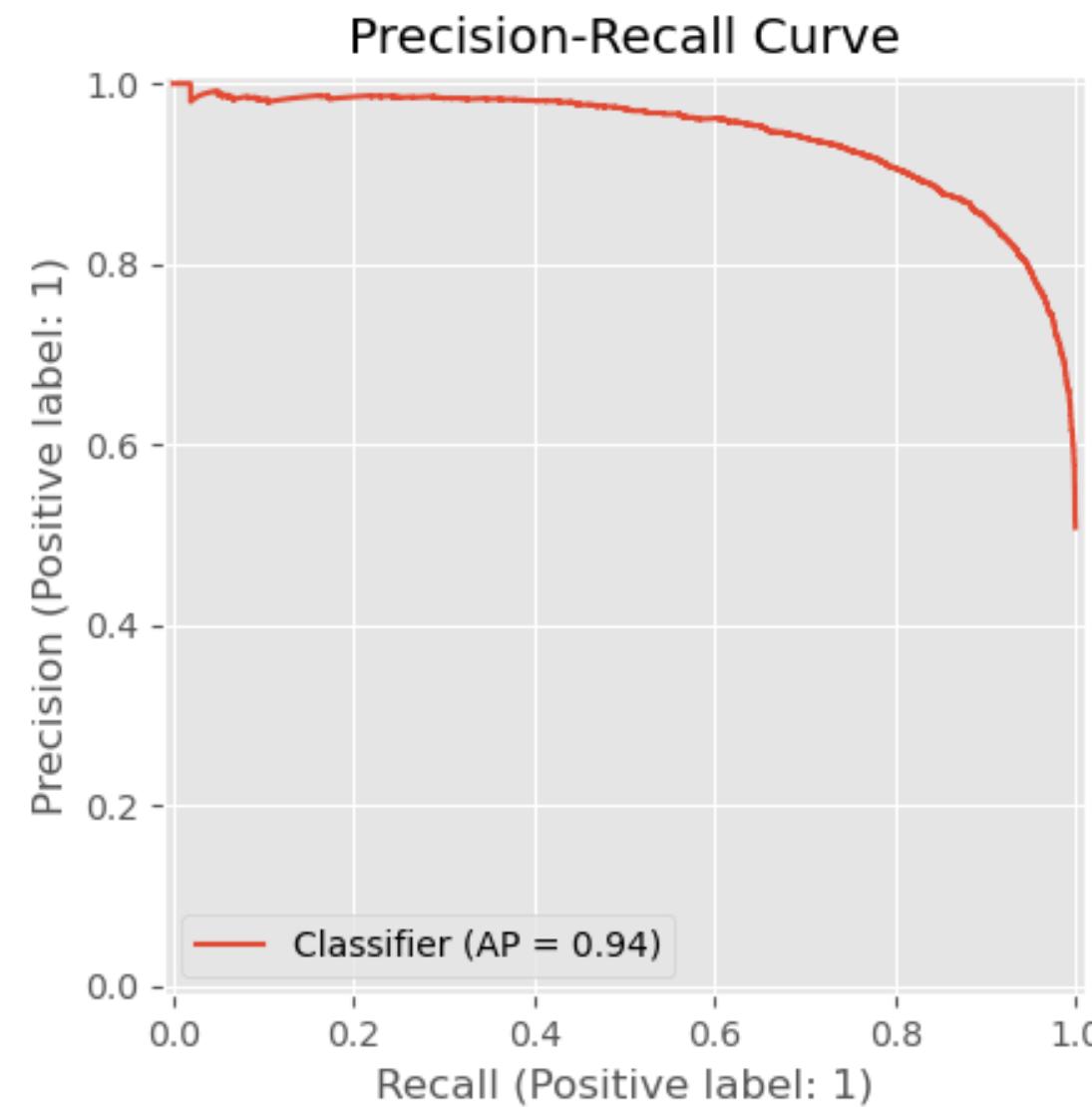
Metrics												Parameters		
Run Name	Created	Duration	Source	train_accuracy	train_loss	val_accuracy	val_loss	batch_size	epochs	learning_rate				
amusing-dog-899	13 minutes ago	10.7min	c:\Users\...	0.900718629...	0.246342316...	0.867750883...	0.320468753...	128	10	0.001				
logging_metrics	20 hours ago	4.5min	c:\Users\...	-	-	0.867750883102417	-	-	-	-				
flawless-newt-162	21 hours ago	2.4min	c:\Users\...	0.871958911...	0.300738602...	0.858943641...	0.328050017...	512	10	0.001				
persistent-snail-42	21 hours ago	2.3min	c:\Users\...	0.872935831...	0.300910830...	0.855161964...	0.340371012...	512	10	0.001				
spiffy-quail-373	21 hours ago	1.4min	c:\Users\...	0.860424816...	0.325690388...	0.824530422...	0.384829550...	512	6	0.002				
delicate-hare-296	21 hours ago	1.6min	c:\Users\...	0.855981349...	0.334912955...	0.824278354...	0.379365235...	512	7	0.001				
vaunted-mule-517	21 hours ago	6.9min	c:\Users\...	0.891466021...	0.261455446...	0.865498542...	0.325892895...	128	10	0.003				
youthful-hare-845	21 hours ago	6.4min	c:\Users\...	0.904197633...	0.240436553...	0.871549248...	0.346818745...	128	10	0.001				

MLOPS

IMDB MLflow

LSTM Precision recall curve

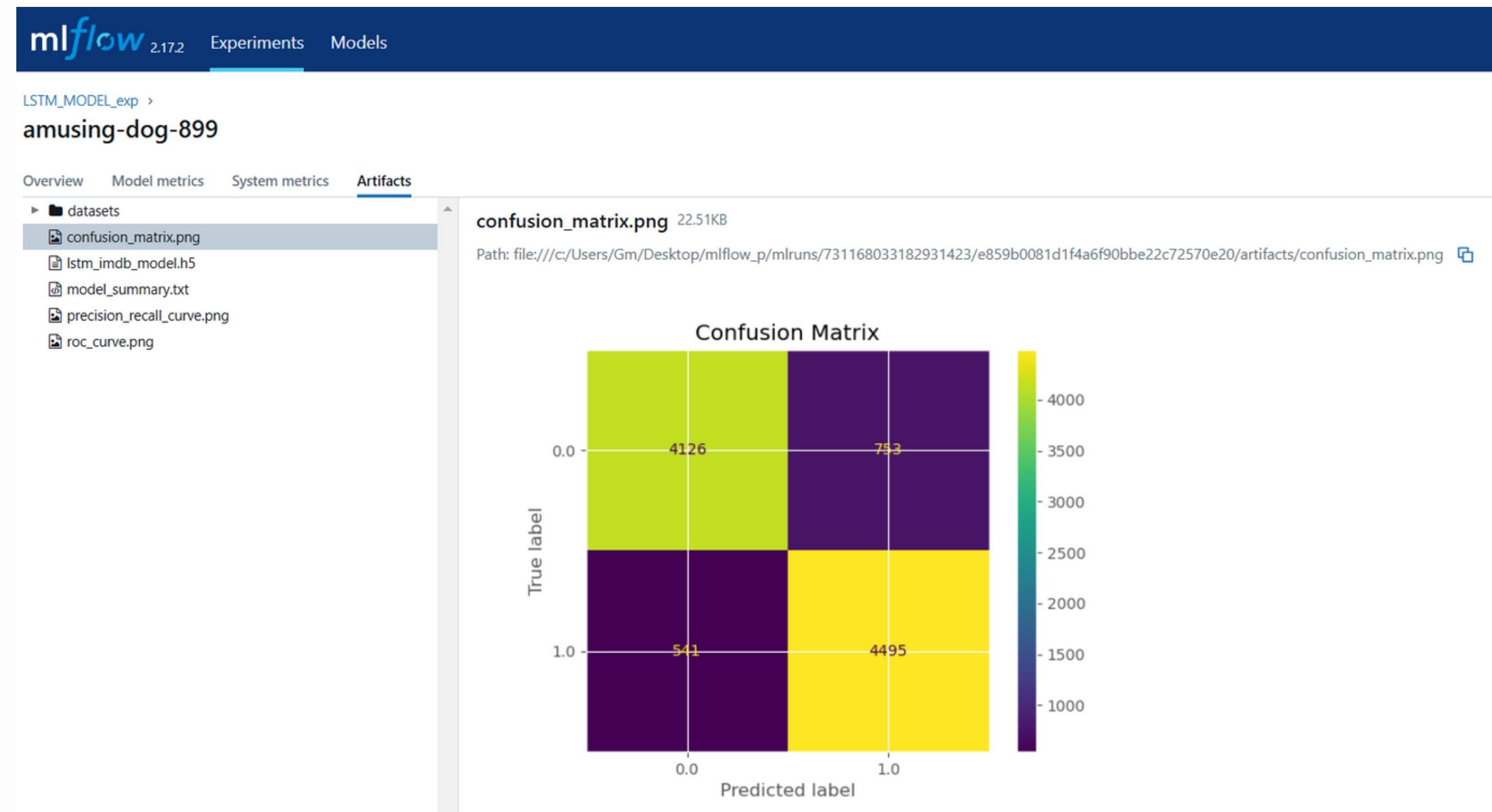
LSTM ROC Curve



MLOPS

IMDB MLflow

LSTM Confusion matrix



Model Deployment On Azure ML

After we finished training our models and achieved good results, we used multiple platforms like: Azure ML , streamlit , Github, MLFlow and DagsHub to version and deploy our model and automate lifecycle of our ML model and to make our system more reliable and scalable and maintainable. During MLOps to train and validate multiple versions of a model to ensure accurate tracking and comparison.

- Here are the steps that we have taken to deploy our model on Azure ML:

1. Create training and registering the model as MLFlow model.
2. Create Azure ML job.
3. Create a new Batch endpoint.
4. Deploy the model to the endpoint with the latest version of your model.



Training ML FLOW Model Results on Azure ML Job

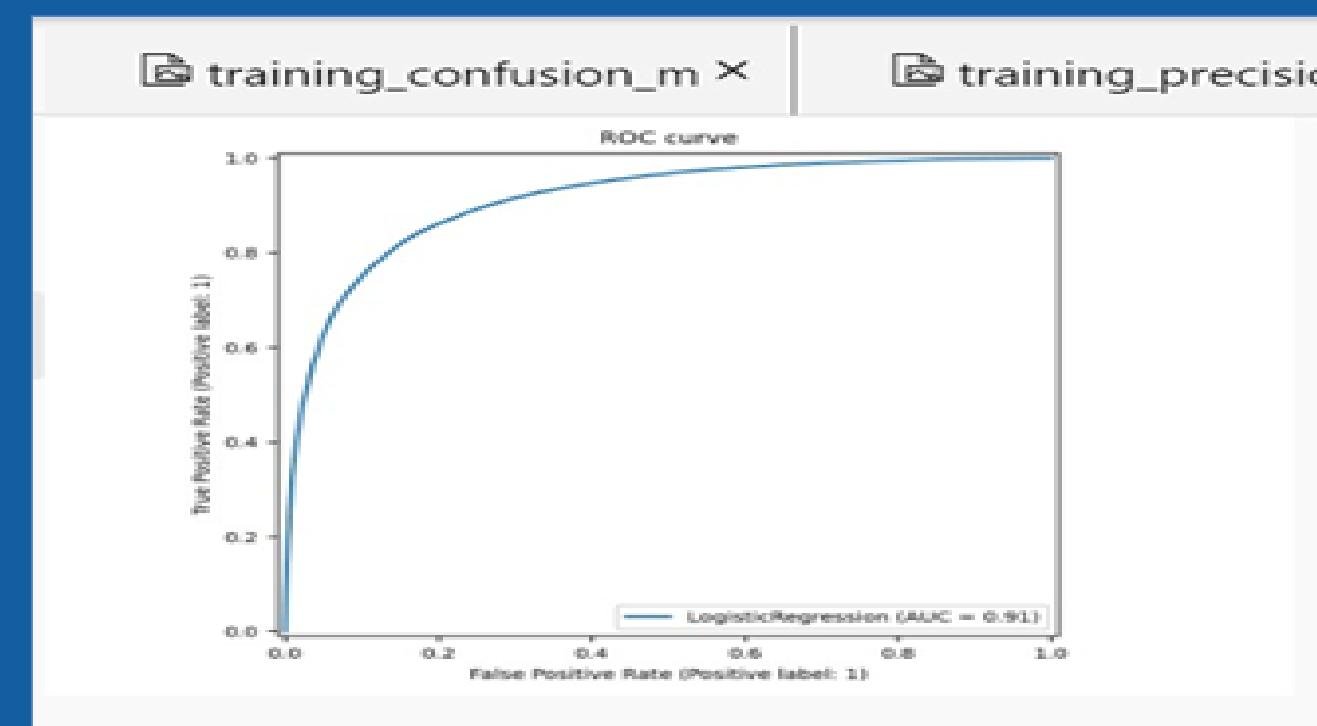
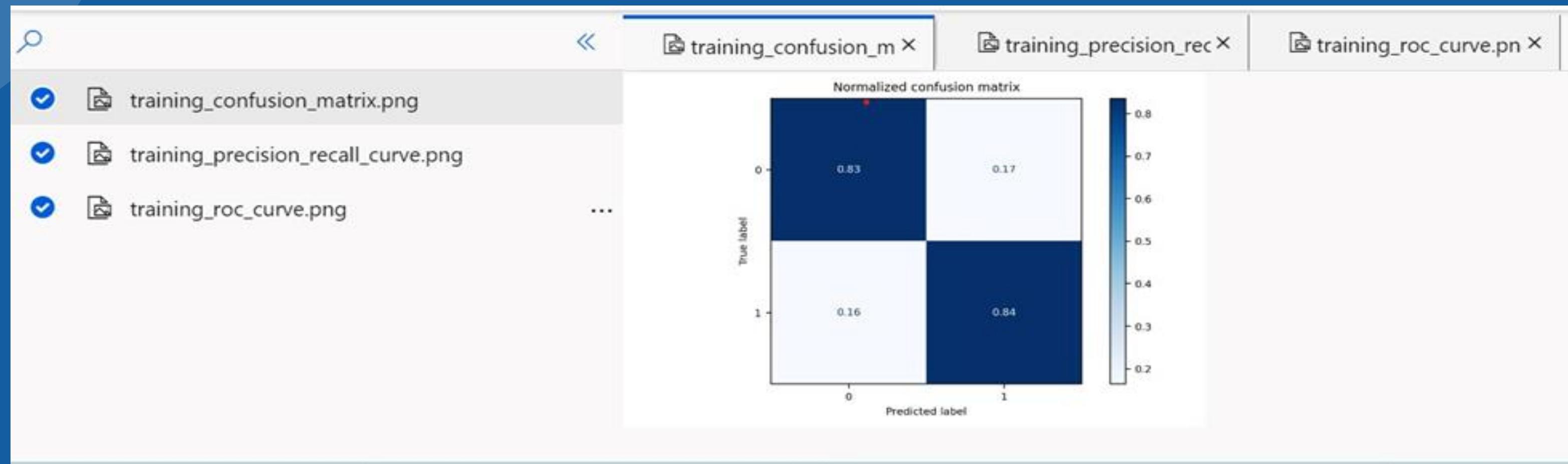
We have finished our model on Amazon products data and trained our Logistic Regression model and deployed our model after making sure that it achieves high accuracy on Azure and Streamlit and these some of the results we have achieved after making our customer sentiment and trend analysis machine learning model

- The following photo shows the model and its Accuracy metrics on Azure ML Job

The screenshot shows the Azure ML Studio interface for a completed job named "sentiment_analysis_logistic_regression6". The "Metrics" tab is selected. The page displays various training metrics in a grid format. The visible metrics and their values are:

metric	value	metric	value	metric	value
num_features	1	num_samples	50,500	training_accuracy_score	0.8343812
training_f1_score	0.8343817	training_log_loss	0.4002972		
training_precision_score	0.8343825	training_recall_score	0.8343812	training_roc_auc	0.9123898
training_score	0.8343812				

- Training confusion matrix and ROC curve of azure ML Job



Streamlit Deployment

- Also, we deployed our model on streamlit and Azure web Service and created Comprehensive Data Science Toolkit for EDA ,Sentiment Analysis and MLFlow Tracking.

Main Page

EDA

Tuning Random Forest

Tuning Logistic Regression

Tracking Logistic Regression

Model Deployment

About App

A Customer Sentiment analysis Project which collect data of reviews

Comprehensive Data Science Toolkit for Sentiment Analysis Tasks

Team Members

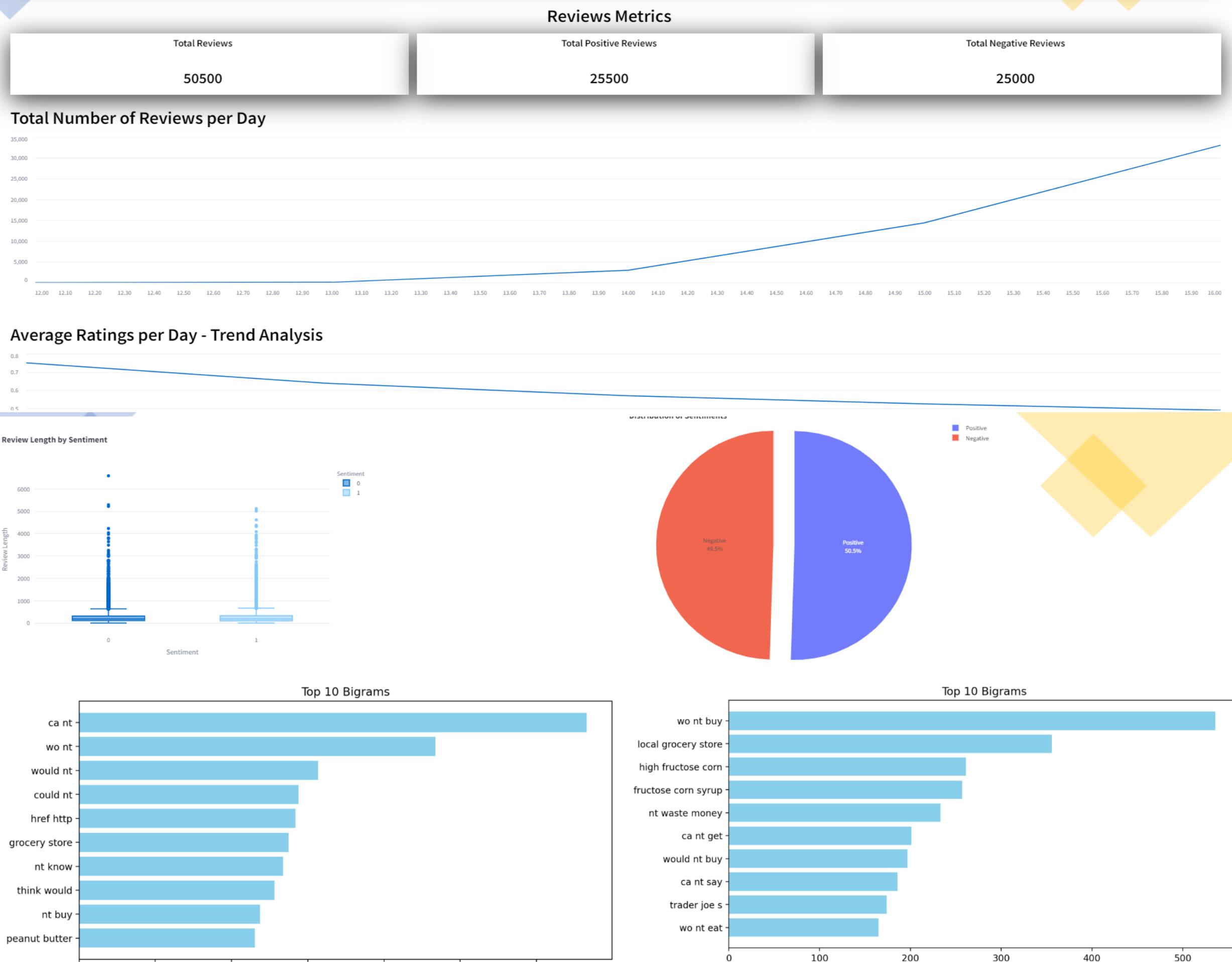


- Mohamed Talaat Abo Elftouh
- Mohamed Mostafa Abdelhamed
- Moaz Mohamed Tawfik
- Amr Khaled Mostafa
- Mahmoud Mohammed Abdelmawgoud
- Mohamed Alaa Elsayad

Customer Reviews EDA

Discover insights and trends from customer reviews

Streamlit EDA



Real-Time Sentiment Analysis Result

Sentiment Analysis APP 😊 😕

Choose Sentiment Analysis Model:

RandomForest XGBoost Logistic Regression LSTM

Enter text for sentiment analysis:

pretty good for the price.

Analyze Sentiment

Raw prediction output: [[1]]

Prediction: Positive ✓

Sentiment Analysis APP 😊 😕

Choose Sentiment Analysis Model:

RandomForest XGBoost Logistic Regression LSTM

Enter text for sentiment analysis:

Unfourtunatly this did not fit, returned

Analyze Sentiment

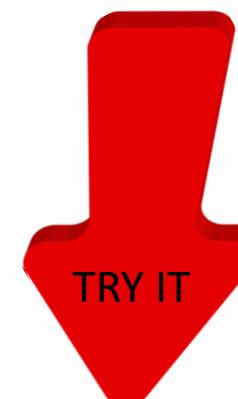
Raw prediction output: [[0]]

Prediction: Negative ✗

TRY IT

<https://depisentimentanalysisapp-11-6-2024.streamlit.app/>

Batch Sentiment Analysis Result



Batch Sentiment Analysis Section

Upload a CSV file for batch sentiment analysis

Drag and drop file here
Limit 200MB per file

Reviews_sample.csv 70.1KB

Data must have a text column with name is 'text'

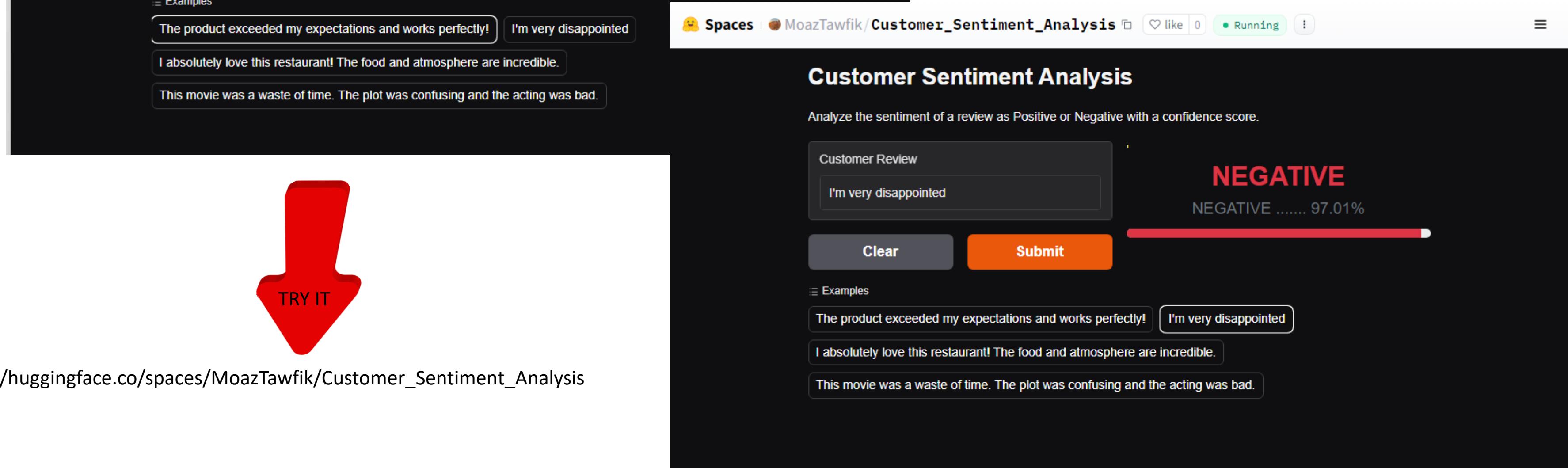
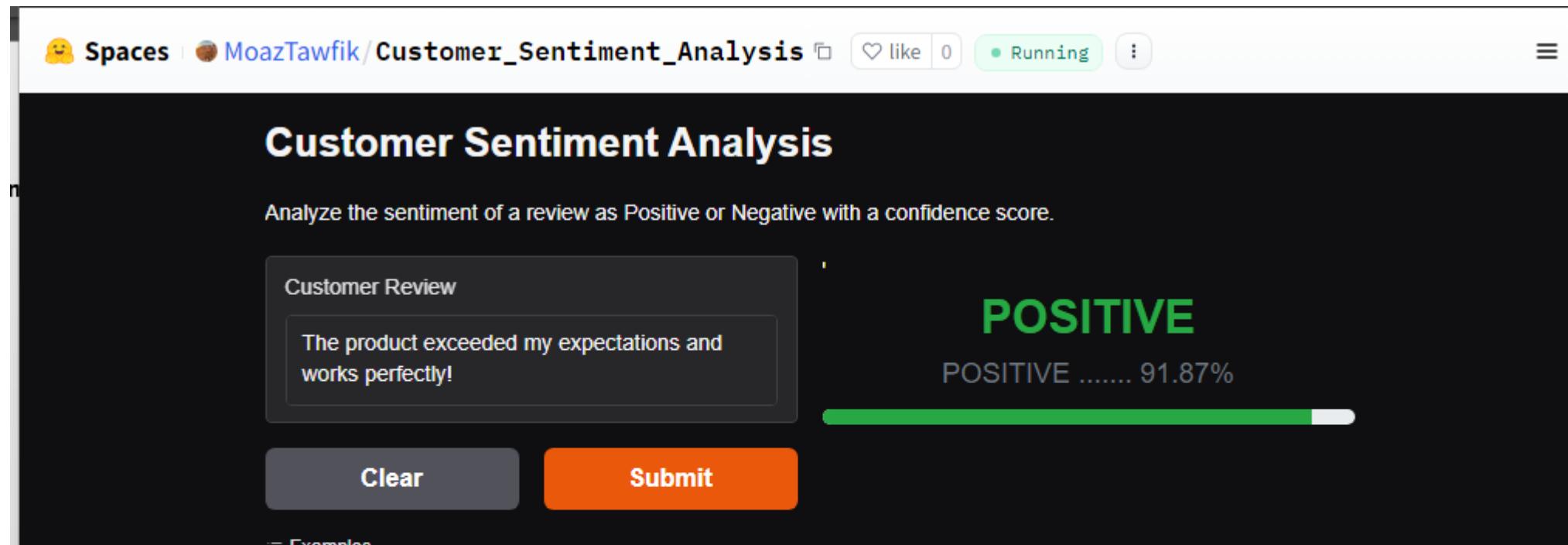
The uploaded file has the correct column!

Sentiment Analysis Results:

	text	clean_text	Sentiment
0	I have bought several of the Vitality canned dog food products and have found them	bought sever vital can dog food product found good qualiti product look like stew pr	Positive
1	Product arrived labeled as Jumbo Salted Peanuts...the peanuts were actually small s	product arriv label jumbo salt peanut ... peanut actual small size unsalt sure error ver	Positive
2	This is a confection that has been around a few centuries. It is a light, pillowy citrus g	confect around centuri light pillowi citru gelatin nut case filbert cut tini squar liber co	Negative
3	If you are looking for the secret ingredient in Robitussin I believe I have found it. I got	look secret ingredi robitussin believ found got addit root beer extract order good mac	Positive
4	Great taffy at a great price. There was a wide assortment of yummy taffy. Delivery w	great taffi great price wide assort yummi taffi deliveri quick taffi lover deal	Positive

<https://depisentimentanalysisapp-11-6-2024.streamlit.app/>

- Our sentiment analysis model on HuggingFace



https://huggingface.co/spaces/MoazTawfik/Customer_Sentiment_Analysis

TRY IT



Resource Page



Hugging Face



kaggle



THANK YOU!