In [ ]:
```
import pandas as pd
import numpy as np
import sklearn as sk
# !pip install translate
from translate import Translator
```

In [ ]:
```
# convert the _chat.txt file into a dataframe
df = pd.read_csv('./_chat.txt', sep='\t', header=None)
```

In [ ]:
```
df[0] = df[df[0].str.contains("görüntü") == False]
df[0] = df[df[0].str.contains("video") == False]
df[0] = df[df[0].str.contains("Çıkartma") == False]
df
```

Out[ ]:

	0
0	[12/9/22 AM 8:37:36] Person 1: Mesajlar ve ar...
1	[12/9/22 AM 8:37:36] Person 1: hey
2	[12/9/22 AM 8:49:53] Person 2: oh hey kiddo
3	[12/9/22 AM 8:50:28] Person 1: how u doing
4	[12/9/22 AM 8:50:38] Person 2: i'm alright
...	...
1232	[12/19/22 AM 11:07:33] Person 2: i'm working o...
1233	[12/19/22 AM 11:07:48] Person 2: i need some d...
1234	[12/19/22 AM 11:07:59] Person 1: ofc
1235	[12/19/22 AM 11:08:20] Person 2: if the result...
1236	[12/19/22 AM 11:08:50] Person 1: okay okay if ...

1237 rows × 1 columns

In [ ]:
```
# split the dataframe into 3 columns, one for each column in the _chat.txt file
df = df[0].str.split(' ', expand=True)
```

In [ ]:
```
df
```

Out[ ]:

	0	1	2	3	4	5	6	7	8	9	...	23
0	[12/9/22	AM	8:37:36]	Person	1:	Mesajlar	ve	aramalar	uçtan	uca	...	ve
1	[12/9/22	AM	8:37:36]	Person	1:	hey	None	None	None	None	...	None
2	[12/9/22	AM	8:49:53]	Person	2:	oh	hey	kiddo	None	None	...	None
3	[12/9/22	AM	8:50:28]	Person	1:	how	u	doing	None	None	...	None
4	[12/9/22	AM	8:50:38]	Person	2:	i'm	alright	None	None	None	...	None
...	...	...	...	...	...	...	...	...	...	...	...	...
1232	[12/19/22	AM	11:07:33]	Person	2:	i'm	working	on	a	project	...	None
1233	[12/19/22	AM	11:07:48]	Person	2:	i	need	some	data,	can	...	None
1234	[12/19/22	AM	11:07:59]	Person	1:	ofc	None	None	None	None	...	None
1235	[12/19/22	AM	11:08:20]	Person	2:	if	the	results	are	good	...	None
1236	[12/19/22	AM	11:08:50]	Person	1:	okay	okay	if	the	results	...	None

1237 rows × 33 columns

In [ ]:
```
# remove first three columns
df = df.drop(df.columns[[0, 1, 2]], axis=1)
# combine the first four columns into one column
df[3] = df[3].str.cat([df[4], df[5], df[6]], sep=" ")
# remove columns 4, 5, 6
df = df.drop([4,5,6], axis=1)
# remove the first row
df = df.drop(df.index[0])
```

In [ ]:
```
# combine the coluumns into one column
# replace none with empty string
df = df.replace(np.nan, '', regex=True)
```

In [ ]:
```
# reset column numbers
df = df.reset_index(drop=True)
df = df.rename(columns={3: "Who texted"})
```

In [ ]:
```
# combine the rest of the columns into one column
df['Message'] = df[df.columns[1:]].apply( lambda x: ' '.join(x.dropna().astype(
```

In [ ]:
```
# combine the message and who texted columns into a new dataframe
df_final = df[['Who texted', 'Message']]
df_final['Message'] =df_final['Message'].str.replace('[^\w\s]','')
```

```
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\3796077580.py:3: FutureWarni
ng: The default value of regex will change from True to False in a future vers
ion.
  df_final['Message'] =df_final['Message'].str.replace('[^\w\s]','')
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\3796077580.py:3: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_final['Message'] =df_final['Message'].str.replace('[^\w\s]','')
```

In [ ]:
```
df_final
```

Out[ ]:

	Who texted	Message
0		
1	Person 2: oh hey	kiddo
2	Person 1: how u	doing
3	Person 2: i'm alright	
4	Person 2: what about	you
...	...	...
1231	Person 2: i'm working	on a project to classify the content of the me...
1232	Person 2: i need	some data can i use our whatsapp conversations...
1233		
1234	Person 2: if the	results are good ill come back to you and get ...
1235	Person 1: okay okay	if the results are good you can share it you a...

1236 rows × 2 columns

In [ ]:
```
# convert all message to lowercase
df_final['Message'] = df_final['Message'].str.lower()
# convert all messages to english
translator= Translator(to_lang="en")
df_final['Message'] = df_final['Message'].apply(translator.translate)
```

```
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\1226664281.py:2: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_final['Message'] = df_final['Message'].str.lower()
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\1226664281.py:5: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_final['Message'] = df_final['Message'].apply(translator.translate)
```

In [ ]:
```
df_final
```

Out[ ]:

	Who texted	Message
0		
1	Person 2: oh hey	kiddo
2	Person 1: how u	doing
3	Person 2: i'm alright	
4	Person 2: what about	you
...	...	...
1231	Person 2: i'm working	on a project to classify the content of the me...
1232	Person 2: i need	some data can i use our whatsapp conversations...
1233		
1234	Person 2: if the	results are good ill come back to you and get ...
1235	Person 1: okay okay	if the results are good you can share it you a...

1236 rows × 2 columns

In [ ]:
```
#use nltk to tokenize the messages
import nltk
```

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

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\hcang\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[ ]:
```
True
```

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

df_final['Message'] = df_final['Message'].apply(word_tokenize)
```

```
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\3305707095.py:3: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_final['Message'] = df_final['Message'].apply(word_tokenize)
```

In [ ]:
```
df_final
```

Out[ ]:

	Who texted	Message
0		[]
1	Person 2: oh hey	[kiddo]
2	Person 1: how u	[doing]
3	Person 2: i'm alright	[]
4	Person 2: what about	[you]
...	...	...
1231	Person 2: i'm working	[on, a, project, to, classify, the, content, o...
1232	Person 2: i need	[some, data, can, i, use, our, whatsapp, conve...
1233		[]
1234	Person 2: if the	[results, are, good, ill, come, back, to, you,...
1235	Person 1: okay okay	[if, the, results, are, good, you, can, share,...

1236 rows × 2 columns

In [ ]:
```
# stem the words
from nltk.stem import PorterStemmer

ps = PorterStemmer()

df_final['Message'] = df_final['Message'].apply(lambda x: [ps.stem(y) for y in
df_final
```

```
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\1044180721.py:6: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_final['Message'] = df_final['Message'].apply(lambda x: [ps.stem(y) for y
in x])
```

Out[ ]:

	Who texted	Message
0		[]
1	Person 2: oh hey	[kiddo]
2	Person 1: how u	[do]
3	Person 2: i'm alright	[]
4	Person 2: what about	[you]
...	...	...
1231	Person 2: i'm working	[on, a, project, to, classifi, the, content, o...
1232	Person 2: i need	[some, data, can, i, use, our, whatsapp, conve...
1233		[]
1234	Person 2: if the	[result, are, good, ill, come, back, to, you, ...
1235	Person 1: okay okay	[if, the, result, are, good, you, can, share, ...

1236 rows × 2 columns

In [ ]:

```
# combine them back again into a string

df_final['Message'] = df_final['Message'].apply(lambda x: ' '.join(x))

df_final
```

```
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\1337946706.py:3: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_final['Message'] = df_final['Message'].apply(lambda x: ' '.join(x))
```

Out[ ]:

	Who texted	Message
0		
1	Person 2: oh hey	kiddo
2	Person 1: how u	do
3	Person 2: i'm alright	
4	Person 2: what about	you
...	...	...
1231	Person 2: i'm working	on a project to classifi the content of the me...
1232	Person 2: i need	some data can i use our whatsapp convers to te...
1233		
1234	Person 2: if the	result are good ill come back to you and get y...
1235	Person 1: okay okay	if the result are good you can share it you al...

1236 rows × 2 columns

In [ ]:
```
from transformers  import pipeline

# perform sentiment analysis on the messages, TODO: this only does it to the er

classifier = pipeline('sentiment-analysis')

# put the label and score into a new dataframe
df_final['Sentiment'] = df_final['Message'].apply(classifier)
```

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-en
glish and revision af0f99b (https://huggingface.co/distilbert-base-uncased-fin
etuned-sst-2-english).
Using a pipeline without specifying a model name and revision in production is
not recommended.
C:\Users\hcang\AppData\Local\Temp\ipykernel_11012\4216424547.py:8: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_final['Sentiment'] = df_final['Message'].apply(classifier)
```

In [ ]:
```
# df_copy = df_final.copy()
```

In [ ]:
```
df_final = df_copy.copy()
```

In [ ]:
```
df_final
```

Out[ ]:

	Who texted	Message	Sentiment
0	ZC:	hey	[{'label': 'POSITIVE', 'score': 0.997920930385...
1	JHCG:	oh hey kiddo	[{'label': 'POSITIVE', 'score': 0.998898744583...
2	ZC:	how u dooiiinnnn	[{'label': 'NEGATIVE', 'score': 0.982698261737...
3	JHCG:	im alriiight	[{'label': 'NEGATIVE', 'score': 0.947273492813...
4	JHCG:	what about you	[{'label': 'POSITIVE', 'score': 0.998605191707...
...	...	...	...
1212	JHCG:	im work on a project to classifi the content o...	[{'label': 'NEGATIVE', 'score': 0.997326970100...
1213	JHCG:	i need some data can i use our whatsapp conver...	[{'label': 'NEGATIVE', 'score': 0.999537467956...
1214	ZC:	ofc	[{'label': 'NEGATIVE', 'score': 0.844272613525...
1215	JHCG:	if the result are good ill come back to you an...	[{'label': 'NEGATIVE', 'score': 0.991953074932...
1216	ZC:	okay okay if the result are good you can share...	[{'label': 'POSITIVE', 'score': 0.894801676273...

1217 rows × 3 columns

In [ ]:
```
df_final.to_csv('output.csv')

# label goes into label column, score goes into score column
df_final['label'] = df_final['Sentiment'].apply(lambda x: x[0]['label'])
df_final['score'] = df_final['Sentiment'].apply(lambda x: x[0]['score'])
```

In [ ]:
```
# if the label is positive, then the score is positive, if the label is negativ

for index, row in df_final.iterrows():
    if row['label'] == 'NEGATIVE':
        df_final.at[index, 'score'] = df_final.at[index, 'score'] * -1

df_final
```

Out[ ]:

	Who texted	Message	Sentiment	label	score
0	ZC:	hey	[{'label': 'POSITIVE', 'score': 0.997920930385...	POSITIVE	0.997921
1	JHCG:	oh hey kiddo	[{'label': 'POSITIVE', 'score': 0.998898744583...	POSITIVE	0.998899
2	ZC:	how u dooiiinnnn	[{'label': 'NEGATIVE', 'score': 0.982698261737...	NEGATIVE	-0.982698
3	JHCG:	im alriiight	[{'label': 'NEGATIVE', 'score': 0.947273492813...	NEGATIVE	-0.947273
4	JHCG:	what about you	[{'label': 'POSITIVE', 'score': 0.998605191707...	POSITIVE	0.998605
...	...	...	...	...	...
1212	JHCG:	im work on a project to classifi the content o...	[{'label': 'NEGATIVE', 'score': 0.997326970100...	NEGATIVE	-0.997327
1213	JHCG:	i need some data can i use our whatsapp conver...	[{'label': 'NEGATIVE', 'score': 0.999537467956...	NEGATIVE	-0.999537
1214	ZC:	ofc	[{'label': 'NEGATIVE', 'score': 0.844272613525...	NEGATIVE	-0.844273
1215	JHCG:	if the result are good ill come back to you an...	[{'label': 'NEGATIVE', 'score': 0.991953074932...	NEGATIVE	-0.991953
1216	ZC:	okay okay if the result are good you can share...	[{'label': 'POSITIVE', 'score': 0.894801676273...	POSITIVE	0.894802

1217 rows × 5 columns

In [ ]:
```
df_final.drop(['Sentiment'], axis=1, inplace=True)
df_final.drop(['label'], axis=1, inplace=True)
```

In [ ]:
```
df_final
```

Out[ ]:

	Who texted	Message	score
0	ZC:	hey	0.997921
1	JHCG:	oh hey kiddo	0.998899
2	ZC:	how u dooiiinnnn	-0.982698
3	JHCG:	im alriiight	-0.947273
4	JHCG:	what about you	0.998605
...	...	...	...
1212	JHCG:	im work on a project to classifi the content o...	-0.997327
1213	JHCG:	i need some data can i use our whatsapp conver...	-0.999537
1214	ZC:	ofc	-0.844273
1215	JHCG:	if the result are good ill come back to you an...	-0.991953
1216	ZC:	okay okay if the result are good you can share...	0.894802

1217 rows × 3 columns

In [ ]:
```
df_final.describe()
```

Out[ ]:

	score
count	1217.000000
mean	-0.065373
std	0.932485
min	-0.999818
25%	-0.982902
50%	-0.673519
75%	0.982732
max	0.999875

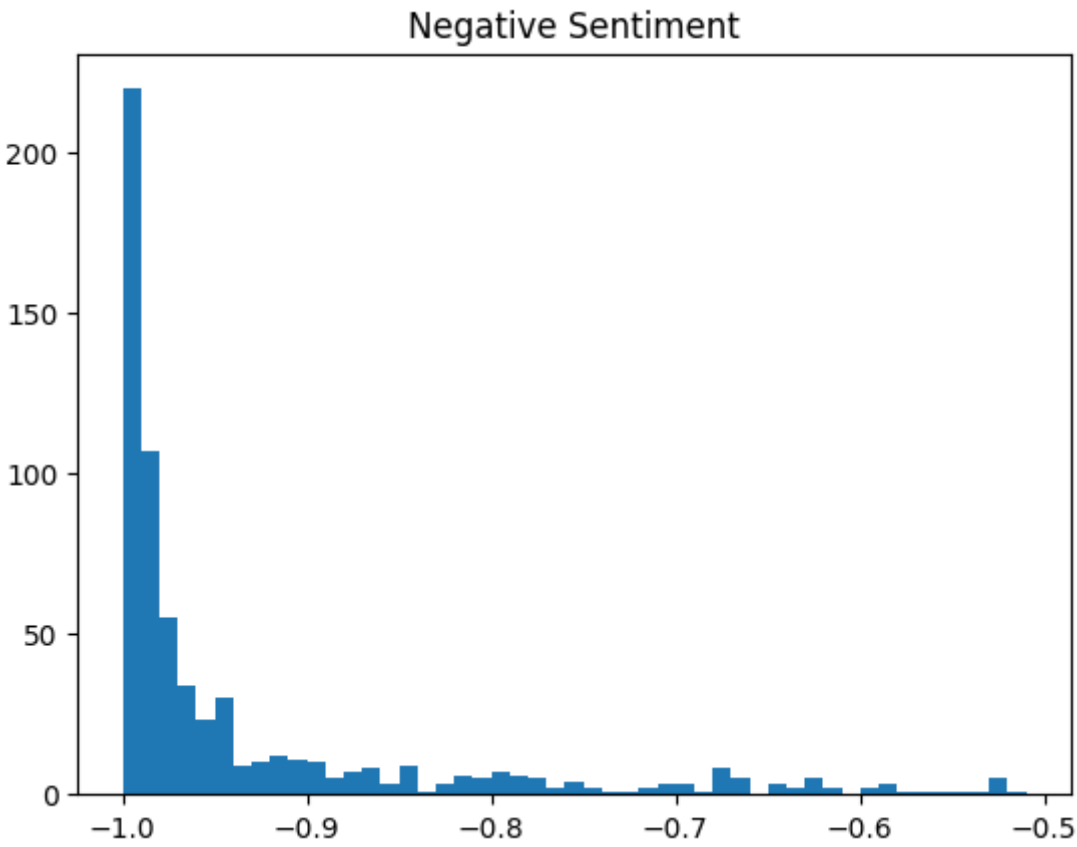
In [ ]:
```
# plot the score column, they go between -1 and -0.5 and 0.5 and 1, so we can s

import matplotlib.pyplot as plt

# plot from -1 to -0.5 intervals (represents negative sentiment)
plt.hist(df_final['score'], bins=np.arange(-1, -0.5, 0.01))
# add a title to the plot

plt.title("Negative Sentiment")
# make another plot from 0.5 to 1 intervals (represents positive sentiment)
```

Out[ ]:
```
Text(0.5, 1.0, 'Negative Sentiment')
```

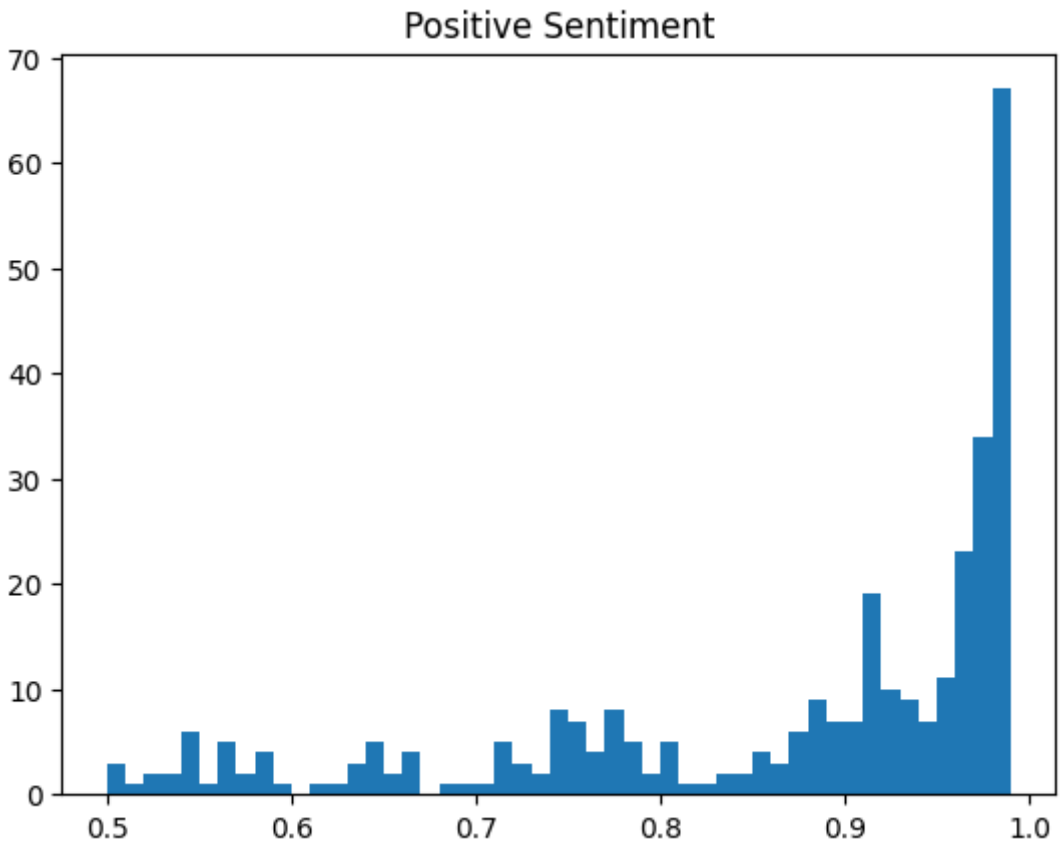


In [ ]:
```
# plot positive sentiment from 0.5 to 1

plt.hist(df_final['score'], bins=np.arange(0.5, 1, 0.01))

plt.title("Positive Sentiment")

plt.show()
```


Positive Sentiment
70
60
50
40
30
20
10
0
0.5
0.6
0.7
0.8
0.9
1.0