Programming Assignment 3c: Text classification (Part 3)

- Neha Devi Shakya
- Sarvesh Meenowa
- Khushi Chitra Uday

In [1]:

```
from google.colab import drive
drive.mount("/content/drive")
```

```
Mounted at /content/drive
```

Load required libraries

In [2]:

```
import re
from string import digits

!pip install altair
import altair as alt
import matplotlib.pyplot as plt
import nltk
import numpy as np
import pandas as pd
import seaborn as sns
import warnings
from sklearn import metrics
from sklearn.dummy import DummyClassifier
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.inspection import permutation_importance
from sklearn.metrics import (accuracy_score, confusion_matrix, ConfusionMatrixDisplay,
plot_confusion_matrix, roc_curve, roc_auc_score)
from sklearn.model_selection import cross_val_score, train_test_split, GridSearchCV
from sklearn.multiclass import OneVsRestClassifier
from sklearn.naive_bayes import BernoulliNB, MultinomialNB
from sklearn.svm import LinearSVC, SVC

!pip install stanza
import stanza
from nltk import word_tokenize
from nltk.stem import WordNetLemmatizer
from nltk.stem.porter import PorterStemmer
from sklearn.metrics import cohen_kappa_score
from sklearn.metrics.pairwise import linear_kernel
from sklearn.pipeline import make_pipeline

nltk.download("punkt")
nltk.download("wordnet")

warnings.filterwarnings("ignore")
```

```
Requirement already satisfied: altair in /usr/local/lib/python3.7/dist-pac
kages (4.2.0)
Requirement already satisfied: pandas>=0.18 in /usr/local/lib/python3.7/di
st-packages (from altair) (1.3.5)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-pack
ages (from altair) (1.21.6)
Requirement already satisfied: entrypoints in /usr/local/lib/python3.7/dis
t-packages (from altair) (0.4)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.7/dist-pac
kages (from altair) (2.11.3)
Requirement already satisfied: jsonschema>=3.0 in /usr/local/lib/python3.
7/dist-packages (from altair) (4.3.3)
Requirement already satisfied: toolz in /usr/local/lib/python3.7/dist-pack
ages (from altair) (0.11.2)
Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.7/d
ist-packages (from jsonschema>=3.0->altair) (21.4.0)
Requirement already satisfied: importlib-resources>=1.4.0 in /usr/local/li
b/python3.7/dist-packages (from jsonschema>=3.0->altair) (5.7.1)
Requirement already satisfied: typing-extensions in /usr/local/lib/python
3.7/dist-packages (from jsonschema>=3.0->altair) (4.2.0)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python
3.7/dist-packages (from jsonschema>=3.0->altair) (4.11.3)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.1
4.0 in /usr/local/lib/python3.7/dist-packages (from jsonschema>=3.0->altai
r) (0.18.1)
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.7/dis
t-packages (from importlib-resources>=1.4.0->jsonschema>=3.0->altair) (3.
8.0)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/py
thon3.7/dist-packages (from pandas>=0.18->altair) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/di
st-packages (from pandas>=0.18->altair) (2022.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-p
ackages (from python-dateutil>=2.7.3->pandas>=0.18->altair) (1.15.0)
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.
7/dist-packages (from jinja2->altair) (2.0.1)
Collecting stanza
  Downloading stanza-1.4.0-py3-none-any.whl (574 kB)
     |████████████████████████████████| 574 kB 8.4 MB/s
Requirement already satisfied: torch>=1.3.0 in /usr/local/lib/python3.7/di
st-packages (from stanza) (1.11.0+cu113)
Collecting emoji
  Downloading emoji-1.7.0.tar.gz (175 kB)
     |████████████████████████████████| 175 kB 20.2 MB/s
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-pack
ages (from stanza) (1.21.6)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packag
es (from stanza) (1.15.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packa
ges (from stanza) (4.64.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-p
ackages (from stanza) (2.23.0)
Requirement already satisfied: protobuf in /usr/local/lib/python3.7/dist-p
ackages (from stanza) (3.17.3)
Collecting transformers
  Downloading transformers-4.18.0-py3-none-any.whl (4.0 MB)
     |████████████████████████████████| 4.0 MB 42.4 MB/s
Requirement already satisfied: typing-extensions in /usr/local/lib/python
3.7/dist-packages (from torch>=1.3.0->stanza) (4.2.0)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/usr/local/lib/python3.7/dist-packages (from requests->stanza) (1.24.3)
```

```
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python
3.7/dist-packages (from requests->stanza) (2021.10.8)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/di
st-packages (from requests->stanza) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python
3.7/dist-packages (from requests->stanza) (3.0.4)
Collecting tokenizers!=0.11.3,<0.13,>=0.11.1
  Downloading tokenizers-0.12.1-cp37-cp37m-manylinux_2_12_x86_64.manylinux
2010_x86_64.whl (6.6 MB)
     |████████████████████████████████| 6.6 MB 34.2 MB/s
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python
3.7/dist-packages (from transformers->stanza) (2019.12.20)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-p
ackages (from transformers->stanza) (3.6.0)
Collecting sacremoses
  Downloading sacremoses-0.0.49-py3-none-any.whl (895 kB)
     |████████████████████████████████| 895 kB 44.6 MB/s
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.
7/dist-packages (from transformers->stanza) (21.3)
Collecting huggingface-hub<1.0,>=0.1.0
  Downloading huggingface_hub-0.5.1-py3-none-any.whl (77 kB)
     |████████████████████████████████| 77 kB 3.0 MB/s
Requirement already satisfied: importlib-metadata in /usr/local/lib/python
3.7/dist-packages (from transformers->stanza) (4.11.3)
Collecting pyyaml>=5.1
  Downloading PyYAML-6.0-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_6
4.manylinux_2_12_x86_64.manylinux2010_x86_64.whl (596 kB)
     |████████████████████████████████| 596 kB 11.4 MB/s
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/
python3.7/dist-packages (from packaging>=20.0->transformers->stanza) (3.0.
8)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-
packages (from importlib-metadata->transformers->stanza) (3.8.0)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-pack
ages (from sacremoses->transformers->stanza) (7.1.2)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-pac
kages (from sacremoses->transformers->stanza) (1.1.0)
Building wheels for collected packages: emoji
  Building wheel for emoji (setup.py) ... done
  Created wheel for emoji: filename=emoji-1.7.0-py3-none-any.whl size=1710
46 sha256=52f6e228ba7ce7a9ebc20b45387977b434b38a4929d7c901d47ce26715059220
  Stored in directory: /root/.cache/pip/wheels/8a/4e/b6/57b01db010d17ef6ea
9b40300af725ef3e210cb1acfb7ac8b6
Successfully built emoji
Installing collected packages: pyyaml, tokenizers, sacremoses, huggingface
-hub, transformers, emoji, stanza
  Attempting uninstall: pyyaml
    Found existing installation: PyYAML 3.13
    Uninstalling PyYAML-3.13:
      Successfully uninstalled PyYAML-3.13
Successfully installed emoji-1.7.0 huggingface-hub-0.5.1 pyyaml-6.0 sacrem
oses-0.0.49 stanza-1.4.0 tokenizers-0.12.1 transformers-4.18.0
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
```

Load train and test data

In [3]:

```
# load and check train data
train_df = pd.read_csv("/content/drive/Shareddrives/Applied Machine Learning/PA3c/PA3_train.tsv", names = ["Annotations", "Reviews"], sep = "\t")
train_df.head()
```

Out[3]:

	Annotations	Reviews
0	0/0	Ordered my food the hole meal looked dead. pla...
1	1/1	We stopped her whilst walking in the Haga area...
2	0/0	Bad experience, On 23/03/19 Myself and my part...
3	0/0	Extremely underwhelming experience here last n...
4	0/0	Waited 30 minutes to get a table…that was ok. ...

In [4]:

```
# load and check train data
test_df = pd.read_csv("/content/drive/Shareddrives/Applied Machine Learning/PA3c/PA3_test_clean.tsv",names = ["Annotations", "Reviews"], sep = "\t")
test_df.head()
```

Out[4]:

	Annotations	Reviews
0	0	Over all I felt a bit disappointing with abov...
1	1	A wonderful experience!
2	1	Always very delicious dishes and attentive ser...
3	1	Amazing as always
4	1	Amazing food, the aubergine mess and the Tunis...

In [5]:

```
train_df.shape[0] + test_df.shape[0]
```

Out[5]:

8769

Data Cleaning

Clean Reviews

Steps taken for cleaning the reviews:

1. Import libraries
    - re library for regular expressions
    - nltk library to work with language data
    - download and import english stopwords
2. Create review_cleaning function to
    - convert all words to lowercase
    - remove all emoticons
    - substitute multiple spaces with single space
3. Create LemmaTokenizer class to
    - lemmatize all words in a review that are longer than 1 character and are not punctuations

We decided not to use stemming as it did not improve the accuracy of the models (when used alone or along with lemmatization).

In [6]:

```
# function to clean reiews
def review_cleaning(review):

    # change to lower case
    review = review.lower()

    # deconcatinate words
    review = re.sub(r"n\'t", " not", review)
    review = re.sub(r"\'re", " are", review)
    review = re.sub(r"\'s", " is", review)
    review = re.sub(r"\'d", " would", review)
    review = re.sub(r"\'ll", " will", review)
    review = re.sub(r"\'ve", " have", review)
    review = re.sub(r"\'m", " am", review)

    # remove all emojis
    emoji_pattern = re.compile(
        "["
        u"\U0001F600-\U0001F64F"  # emoticons
        u"\U0001F300-\U0001F5FF"  # symbols & pictographs
        u"\U0001F680-\U0001F6FF"  # transport & map symbols
        u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
        u"\U000024C2-\U0001F251"  # chinese char
        u"\U00002702-\U000027B0"
        u"\U0001f926-\U0001f937"
        u"\U00010000-\U0010ffff"
        u"\u2600-\u2B55"
        u"\u200d"
        u"\u23cf"
        u"\u23e9"
        u"\u231a"
        u"\ufe0f"  # dingbats
        u"\u3030"
        "]+", flags = re.UNICODE
      )
    review = emoji_pattern.sub(r"", review)

    # substitute multiple spaces with single space
    review =  re.sub(r"\s+", " ", review, flags = re.I)

    # remove punctuations, URLs, and @:
    review = re.sub(r"(@\[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)|^rt|http.+?", "
", review)

    # remove digits
    review = review.translate(str.maketrans("", "", digits))

    # lemmatize words
    lemmatizer = WordNetLemmatizer()
    review = [lemmatizer.lemmatize(t) for t in word_tokenize(review) if len(t) > 1]
    review = ' '.join(txt for txt in review)

    return review
```

In [7]:

```
# get cleaned reviews for train data
train_df["Reviews_Cleaned"] = train_df["Reviews"].apply(lambda x: review_cleaning(x))
train_df
```

Out[7]:

	Annotations	Reviews	Reviews_Cleaned
0	0/0	Ordered my food the hole meal looked dead. pla...	ordered my food the hole meal looked dead plai...
1	1/1	We stopped her whilst walking in the Haga area...	we stopped her whilst walking in the haga area...
2	0/0	Bad experience, On 23/03/19 Myself and my part...	bad experience on myself and my partner arrive...
3	0/0	Extremely underwhelming experience here last n...	extremely underwhelming experience here last n...
4	0/0	Waited 30 minutes to get a table…that was ok. ...	waited minute to get table that wa ok sat at t...
...	...	...	...
7013	0/0	Bad service, stay away	bad service stay away
7014	0/0	Old school, but not always in a good way. Lots...	old school but not always in good way lot of f...
7015	1/1	Top 5 allergen free restaurant and the food is...	top allergen free restaurant and the food is g...
7016	-1/0	The ambiance is ok, the service is slightly sl...	the ambiance is ok the service is slightly slo...
7017	1/1	The food was yummy and the drinks excellent. W...	the food wa yummy and the drink excellent want...

7018 rows × 3 columns

In [8]:

```
# get cleaned reviews for test data
test_df["Reviews_Cleaned"] = test_df["Reviews"].apply(lambda x: review_cleaning(x))
```

Clean Annotations

In [9]:

```
# check unique annotation combinations
train_df["Annotations"].unique()
```

Out[9]:

```
array(['0/0', '1/1', '1/0', '-1/0', '-1/1', '0/1', '2/1', '2/0', '1/',
       '9/1'], dtype=object)
```

In [10]:

```
train_df["Annotations"].value_counts().to_dict()
```

Out[10]:

```
{'-1/0': 97,
 '-1/1': 28,
 '0/0': 3126,
 '0/1': 117,
 '1/': 1,
 '1/0': 148,
 '1/1': 3496,
 '2/0': 2,
 '2/1': 2,
 '9/1': 1}
```

In [11]:

```
test_df["Annotations"].value_counts().to_dict()
```

Out[11]:

```
{0: 865, 1: 886}
```

> The annotations that have "2/1", "2/0", "1/" and "9/1" are unclear thus we remove them.

In [12]:

```
# function to remove rows with certain values
def filter_rows_by_values(df, col: str, values):
    return df[~df[col].isin(values)]
```

In [13]:

```
# remove the unclear annotations
train_df_clean = filter_rows_by_values(train_df, "Annotations", ["2/1", "2/0", "1/","9/
1"])
```

In [14]:

```
# function to split annotations
def split_annotations(s):
  s = s.split("/")
  s1 = int(s[0])
  s2 = int(s[1])
  return pd.Series([s1, s2])
```

In [15]:

```
# split annotations
train_df_clean[["Annotator2", "Annotator1"]] = train_df_clean["Annotations"].apply(lamb
da x: split_annotations(x))
train_df_clean.head()
```

Out[15]:

	Annotations	Reviews	Reviews_Cleaned	Annotator2	Annotator1
0	0/0	Ordered my food the hole meal looked dead. pla...	ordered my food the hole meal looked dead plai...	0	0
1	1/1	We stopped her whilst walking in the Haga area...	we stopped her whilst walking in the haga area...	1	1
2	0/0	Bad experience, On 23/03/19 Myself and my part...	bad experience on myself and my partner arrive...	0	0
3	0/0	Extremely underwhelming experience here last n...	extremely underwhelming experience here last n...	0	0
4	0/0	Waited 30 minutes to get a table…that was ok. ...	waited minute to get table that wa ok sat at t...	0	0

In [16]:

```
# convert annotations 1 and 2 to list
annotator1 = train_df_clean["Annotator1"].tolist()
annotator2 = train_df_clean["Annotator2"].tolist()

# calculate cohen kappa score
cohen_kappa_score(annotator1, annotator2)
```

Out[16]:

```
0.8904734252430904
```

Dealing with unlear annotations

> We will now seperate all unclear annotations i.e. -1 and then perform a sentiment analysis using stanza to re-annotate the reviews. We then cross checked the results with the second annotator.

In [17]:

```
# # convert annotations from integer to string
# train_df_clean[["Annotator1","Annotator2"]] = train_df_clean[["Annotator1","Annotator
2"]].astype(str)
# train_df_clean
```

In [18]:

```
# seperate all -1 annotations
train_df_unclear = train_df_clean.loc[train_df_clean["Annotator1"].isin([-1]) | train_d
f_clean["Annotator2"].isin([-1])]
# train_df_unclear = train_df_clean.loc[train_df_clean["Annotations"].isin(["1/0", "-1/
0", "-1/1", "0/1"])]
```

In [19]:

```
# remove all uncertainties from train_df_clean and disagreements
train_df_clean = filter_rows_by_values(train_df_clean, "Annotations", ["1/0", "-1/0",
"-1/1", "0/1"])
# train_df_clean = filter_rows_by_values(train_df_clean, "Annotations", ["-1/0", "-1/
1"])
```

In [20]:

```
# reset index of dataframe
train_df_clean.reset_index(drop = True, inplace = True)
```

In [21]:

```
# set annotator 2 as final annotation
train_df_clean["Final_Annotation"] = train_df_clean["Annotator1"]
train_df_clean
```

Out[21]:

	Annotations	Reviews	Reviews_Cleaned	Annotator2	Annotator1	Final_Annotatior
0	0/0	Ordered my food the hole meal looked dead. pla...	ordered my food the hole meal looked dead plai...	0	0	(
1	1/1	We stopped her whilst walking in the Haga area...	we stopped her whilst walking in the haga area...	1	1	'
2	0/0	Bad experience, On 23/03/19 Myself and my part...	bad experience on myself and my partner arrive...	0	0	(
3	0/0	Extremely underwhelming experience here last n...	extremely underwhelming experience here last n...	0	0	(
4	0/0	Waited 30 minutes to get a table…that was ok. ...	waited minute to get table that wa ok sat at t...	0	0	(
...	...	...	...	...	...	..
6617	1/1	We recently dined at Ma Cuisine, and enjoyed e...	we recently dined at ma cuisine and enjoyed ev...	1	1	'
6618	0/0	Bad service, stay away	bad service stay away	0	0	(
6619	0/0	Old school, but not always in a good way. Lots...	old school but not always in good way lot of f...	0	0	(
6620	1/1	Top 5 allergen free restaurant and the food is...	top allergen free restaurant and the food is g...	1	1	'
6621	1/1	The food was yummy and the drinks excellent. W...	the food wa yummy and the drink excellent want...	1	1	'

6622 rows × 6 columns

In [22]:

```
# download stanza english model
print("Downloading English model...")
stanza.download("en")
```

```
Downloading English model...

2022-04-28 18:17:38 INFO: Downloading default packages for language: en (E
nglish)...

2022-04-28 18:18:03 INFO: Finished downloading models and saved to /root/s
tanza_resources.
```

In [23]:

```
# create a stanza pipeline
nlp = stanza.Pipeline(lang = "en", processors = "tokenize, sentiment")
```

```
2022-04-28 18:18:03 INFO: Loading these models for language: en (English):
========================
| Processor | Package  |
------------------------
| tokenize  | combined |
| sentiment | sstplus  |
========================

2022-04-28 18:18:03 INFO: Use device: cpu
2022-04-28 18:18:03 INFO: Loading: tokenize
2022-04-28 18:18:03 INFO: Loading: sentiment
2022-04-28 18:18:04 INFO: Done loading processors!
```

In [24]:

```
# function to perform sentiment analysis
# 0 = negative, 1 = neutral, 2 = positive
def get_sentiment(x):
    doc = nlp(x)
    for sentence in doc.sentences:
        return sentence.sentiment
```

In [25]:

```
# get the sentiment scores for the unclear review data
train_df_unclear["sent_score"] = train_df_unclear["Reviews"].apply(lambda x:get_sentime
nt(x))
train_df_unclear
```

Out[25]:

	Annotations	Reviews	Reviews_Cleaned	Annotator2	Annotator1	sent_score
19	-1/0	restaurant mon ami en las vegas , un verdadero...	restaurant mon ami en la vega un verdadero eng...	-1	0	0
45	-1/1	Love love love this place .. Its kind of small...	love love love this place it kind of small all...	-1	1	2
95	-1/0	The satay was ok, it was a little tough and sa...	the satay wa ok it wa little tough and sauce w...	-1	0	0
166	-1/1	I loved the ambience of this place. Food was g...	loved the ambience of this place food wa great...	-1	1	2
191	-1/0	The staff is a kind of rude but the foods and ...	the staff is kind of rude but the food and dri...	-1	0	2
...	...	...	...	...	...	...
6821	-1/1	The reason why Taipan Building is most attract...	the reason why taipan building is most attract...	-1	1	2
6915	-1/1	I had the most delicious fish for lunch. Inclu...	had the most delicious fish for lunch included...	-1	1	2
6921	-1/0	Graffiato is AWESOME. However, warning for th...	graffiato is awesome however warning for those...	-1	0	2
6994	-1/0	Some times this place is good today it wasn't ...	some time this place is good today it wa not n...	-1	0	0
7016	-1/0	The ambiance is ok, the service is slightly sl...	the ambiance is ok the service is slightly slo...	-1	0	0

125 rows × 6 columns

In [26]:

```
# remove all review with neutral sentiment i.e. 1
train_df_unclear = filter_rows_by_values(train_df_unclear,"sent_score",[1])
```

In [27]:

```
# set all positive sentiment score label from 2 to 1
train_df_unclear["sent_score"] = train_df_unclear["sent_score"].apply(lambda x: x if x
==  0 else 1)
```

In [28]:

```
# filter all reviews that have same sentiment score as second annotation
train_df_clear = train_df_unclear[train_df_unclear["Annotator1"] ==  train_df_unclear[
"sent_score"]]
```

In [29]:

```
# drop the sent_score column
train_df_clear.drop("sent_score", axis = 1, inplace = True)
```

In [30]:

```
# set first annotation as final annotation
train_df_clear["Final_Annotation"] = train_df_clear["Annotator1"]
train_df_clear
```

Out[30]:

	Annotations	Reviews	Reviews_Cleaned	Annotator2	Annotator1	Final_Annotatior
19	-1/0	restaurant mon ami en las vegas , un verdadero...	restaurant mon ami en la vega un verdadero eng...	-1	0	(
45	-1/1	Love love love this place .. Its kind of small...	love love love this place it kind of small all...	-1	1	'
95	-1/0	The satay was ok, it was a little tough and sa...	the satay wa ok it wa little tough and sauce w...	-1	0	(
166	-1/1	I loved the ambience of this place. Food was g...	loved the ambience of this place food wa great...	-1	1	'
335	-1/0	Sadly this place was a huge disappointment ser...	sadly this place wa huge disappointment servic...	-1	0	(
...	...	...	...	...	...	..
6459	-1/0	We went to the tango show and it was not good ...	we went to the tango show and it wa not good a...	-1	0	(
6821	-1/1	The reason why Taipan Building is most attract...	the reason why taipan building is most attract...	-1	1	'
6915	-1/1	I had the most delicious fish for lunch. Inclu...	had the most delicious fish for lunch included...	-1	1	'
6994	-1/0	Some times this place is good today it wasn't ...	some time this place is good today it wa not n...	-1	0	(
7016	-1/0	The ambiance is ok, the service is slightly sl...	the ambiance is ok the service is slightly slo...	-1	0	(

70 rows × 6 columns

In [31]:

```
# save the unclear review data with sentiment scores
train_df_clear.to_csv("/content/drive/Shareddrives/Applied Machine Learning/PA3c/unsure_reviews_fixed.csv", index = False)
```

In [32]:

```
# load the unclear review data with sentiment scores and get a peek
train_df_clear = pd.read_csv("/content/drive/Shareddrives/Applied Machine Learning/PA3c/unsure_reviews_fixed.csv")
train_df_clear.head()
```

Out[32]:

	Annotations	Reviews	Reviews_Cleaned	Annotator2	Annotator1	Final_Annotation
0	-1/0	restaurant mon ami en las vegas , un verdadero...	restaurant mon ami en la vega un verdadero eng...	-1	0	0
1	-1/1	Love love love this place .. Its kind of small...	love love love this place it kind of small all...	-1	1	1
2	-1/0	The satay was ok, it was a little tough and sa...	the satay wa ok it wa little tough and sauce w...	-1	0	0
3	-1/1	I loved the ambience of this place. Food was g...	loved the ambience of this place food wa great...	-1	1	1
4	-1/0	Sadly this place was a huge disappointment ser...	sadly this place wa huge disappointment servic...	-1	0	0

In [33]:

```
train_df_annot = pd.read_csv("/content/drive/Shareddrives/Applied Machine Learning/PA3c/PA3_train_annot.tsv", names = ["Annotations", "Reviews", "Final_Annotation"], sep = "\t")
train_df_annot
```

Out[33]:

	Annotations	Reviews	Final_Annotation
0	2-1	We were able to get a table immediately aroun...	1
1	2-1	Lovely staff so friendly all dishes are excell...	1
2	2/0	After a horrible experience a year ago we gave...	0
3	1/	We were very excited to try this restaurant, b...	1
4	2/0	All food tasted like the purchased it from Wal...	0

In [34]:

```
# get cleaned reviews for self annotated train data
train_df_annot["Reviews_Cleaned"] = train_df_annot["Reviews"].apply(lambda x: review_cl
eaning(x))
train_df_annot
```

Out[34]:

	Annotations	Reviews	Final_Annotation	Reviews_Cleaned
0	2-1	We were able to get a table immediately aroun...	1	we were able to get table immediately around p...
1	2-1	Lovely staff so friendly all dishes are excell...	1	lovely staff so friendly all dish are excellen...
2	2/0	After a horrible experience a year ago we gave...	0	after horrible experience year ago we gave thi...
3	1/	We were very excited to try this restaurant, b...	1	we were very excited to try this restaurant be...
4	2/0	All food tasted like the purchased it from Wal...	0	all food tasted like the purchased it from wal...

In [35]:

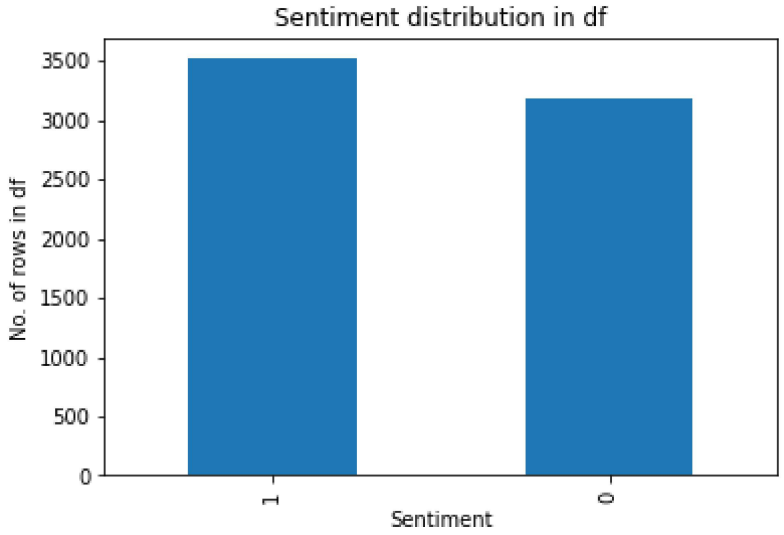
```
# merge both clean and now clear reviews and get a peek
train_df_final = pd.concat([train_df_clean[["Final_Annotation", "Reviews", "Reviews_Cle
aned"]],
                            train_df_clear[["Final_Annotation", "Reviews", "Reviews_Cle
aned"]],
                            train_df_annot[["Final_Annotation", "Reviews", "Reviews_Cle
aned"]]])
train_df_final.head()
```

Out[35]:

	Final_Annotation	Reviews	Reviews_Cleaned
0	0	Ordered my food the hole meal looked dead. pla...	ordered my food the hole meal looked dead plai...
1	1	We stopped her whilst walking in the Haga area...	we stopped her whilst walking in the haga area...
2	0	Bad experience, On 23/03/19 Myself and my part...	bad experience on myself and my partner arrive...
3	0	Extremely underwhelming experience here last n...	extremely underwhelming experience here last n...
4	0	Waited 30 minutes to get a table…that was ok. ...	waited minute to get table that wa ok sat at t...

In [36]:

```
# check the distribution of positive vs negative reviews
plt.figure()
pd.value_counts(train_df_final["Final_Annotation"]).plot.bar(title = "Sentiment distrib
ution in df")
plt.xlabel("Sentiment")
plt.ylabel("No. of rows in df")
plt.show()
```



Text Classification

In [37]:

```
# set X_train, Y_train, X_test and Y_test
X_train = train_df_final["Reviews_Cleaned"]
Y_train = train_df_final["Final_Annotation"]
X_test = test_df["Reviews_Cleaned"]
Y_test = test_df["Annotations"]
```

In [38]:

```
X_train.shape
```

Out[38]:

```
(6697,)
```

In [39]:

```
Y_train.shape
```

Out[39]:

```
(6697,)
```

TF-IDF Visualisation

In [40]:

```
tfidf_vectorizer = TfidfVectorizer()

tfidf_vector = tfidf_vectorizer.fit_transform(X_train)

tfidf_df = pd.DataFrame(tfidf_vector.toarray(), columns=tfidf_vectorizer.get_feature_na
mes_out())

tfidf_df.loc["00_Document Frequency"] = (tfidf_df > 0).sum()

tfidf_df = tfidf_df.drop("00_Document Frequency", errors="ignore")

tfidf_stacked_df = tfidf_df.stack().reset_index()

tfidf_stacked_df = tfidf_stacked_df.rename(columns={0:"tfidf", "level_0": "review","lev
el_1": "term", "level_2": "term"})

top_tfidf = tfidf_stacked_df.sort_values(by = ["review", "tfidf"], ascending = [True, F
alse]).groupby(["review"]).head(5)

top_tfidf = top_tfidf.tail(100)

top_tfidf
```

Out[40]:

	review	term	tfidf
67079394	6677	the	0.252863
67079999	6677	varied	0.237381
67076614	6677	opted	0.229657
67076567	6677	oily	0.225380
67074187	6677	generally	0.219876
...	...	...	...
67266766	6696	mart	0.375773
67270999	6696	wal	0.375773
67268283	6696	purchased	0.324135
67269219	6696	shift	0.324135
67266821	6696	mcdonald	0.285205

100 rows × 3 columns

In [41]:

```
tfidf_df_few =  tfidf_df[['aback', 'abandoned', 'abdul', 'abe', 'abhimanyu', 'ability'
]]
tfidf_df_few
```

Out[41]:

	aback	abandoned	abdul	abe	abhimanyu	ability
0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...
6692	0.0	0.0	0.0	0.0	0.0	0.0
6693	0.0	0.0	0.0	0.0	0.0	0.0
6694	0.0	0.0	0.0	0.0	0.0	0.0
6695	0.0	0.0	0.0	0.0	0.0	0.0
6696	0.0	0.0	0.0	0.0	0.0	0.0

6697 rows × 6 columns

In [42]:

```
tfidf_df_few.drop_duplicates(subset = ['aback', 'abandoned', 'abdul', 'abe', 'abhimanyu', 'ability'],
                     keep = "first", inplace = True)
tfidf_df_few
```

Out[42]:

	aback	abandoned	abdul	abe	abhimanyu	ability
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1017	0.050291	0.000000	0.000000	0.000000	0.000000	0.000000
1349	0.000000	0.000000	0.000000	0.000000	0.000000	0.136796
1509	0.000000	0.000000	0.397527	0.000000	0.000000	0.000000
1565	0.000000	0.000000	0.391277	0.000000	0.000000	0.000000
3214	0.000000	0.325214	0.000000	0.000000	0.000000	0.000000
3390	0.000000	0.000000	0.000000	0.000000	0.000000	0.183183
4563	0.000000	0.000000	0.459192	0.000000	0.000000	0.000000
4712	0.093027	0.000000	0.000000	0.000000	0.000000	0.000000
6067	0.000000	0.000000	0.000000	0.000000	0.305106	0.000000
6458	0.000000	0.000000	0.000000	0.134947	0.000000	0.000000
6493	0.000000	0.000000	0.000000	0.091275	0.000000	0.000000

In [43]:

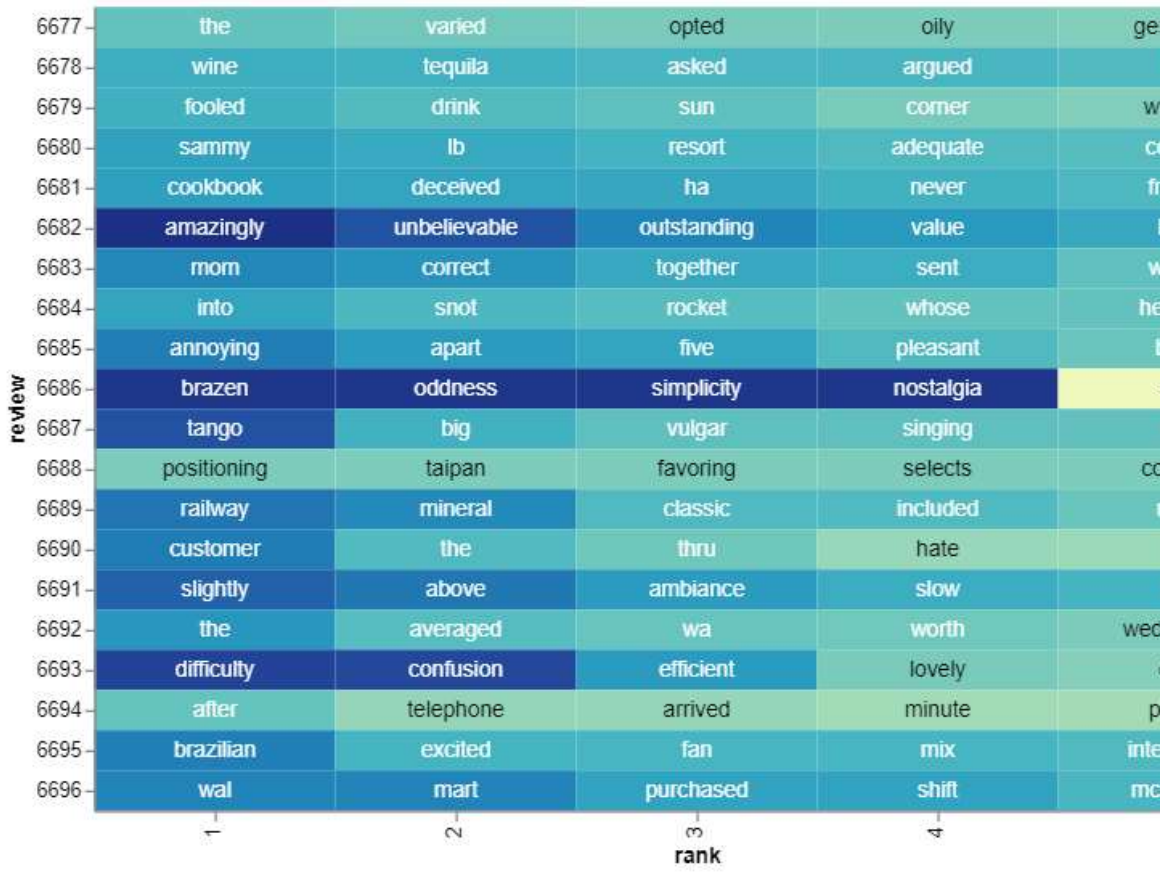
```
print(tfidf_df_few.to_latex())
```

```
\begin{tabular}{lrrrrrr}
\toprule
{} &     aback &  abandoned &     abdul &       abe &  abhimanyu &   abili
ty \\
\midrule
0    &  0.000000 &   0.000000 &  0.000000 &  0.000000 &   0.000000 &  0.00
0000 \\
1017 &  0.050291 &   0.000000 &  0.000000 &  0.000000 &   0.000000 &  0.00
0000 \\
1349 &  0.000000 &   0.000000 &  0.000000 &  0.000000 &   0.000000 &  0.13
6796 \\
1509 &  0.000000 &   0.000000 &  0.397527 &  0.000000 &   0.000000 &  0.00
0000 \\
1565 &  0.000000 &   0.000000 &  0.391277 &  0.000000 &   0.000000 &  0.00
0000 \\
3214 &  0.000000 &   0.325214 &  0.000000 &  0.000000 &   0.000000 &  0.00
0000 \\
3390 &  0.000000 &   0.000000 &  0.000000 &  0.000000 &   0.000000 &  0.18
3183 \\
4563 &  0.000000 &   0.000000 &  0.459192 &  0.000000 &   0.000000 &  0.00
0000 \\
4712 &  0.093027 &   0.000000 &  0.000000 &  0.000000 &   0.000000 &  0.00
0000 \\
6067 &  0.000000 &   0.000000 &  0.000000 &  0.000000 &   0.305106 &  0.00
0000 \\
6458 &  0.000000 &   0.000000 &  0.000000 &  0.134947 &   0.000000 &  0.00
0000 \\
6493 &  0.000000 &   0.000000 &  0.000000 &  0.091275 &   0.000000 &  0.00
0000 \\
\bottomrule
\end{tabular}
```

In [44]:

```
alt.data_transformers.disable_max_rows()


# adding a little randomness to break ties in term ranking
top_tfidf_plusRand = top_tfidf.copy()
top_tfidf_plusRand['tfidf'] = top_tfidf_plusRand['tfidf'] + np.random.rand(top_tfidf.sh
ape[0])*0.0001

# base for all visualizations, with rank calculation
base = alt.Chart(top_tfidf_plusRand).encode(
    x = 'rank:O',
    y = 'review:N'
).transform_window(
    rank = "rank()",
    sort = [alt.SortField("tfidf", order="descending")],
    groupby = ["review"],
)

# heatmap specification
heatmap = base.mark_rect().encode(
    color = 'tfidf:Q'
)

# text labels, white for darker heatmap colors
text = base.mark_text(baseline='middle').encode(
    text = 'term:N',
    color = alt.condition(alt.datum.tfidf >= 0.23, alt.value('white'), alt.value('blac
k'))
)

# display the three superimposed visualizations
(heatmap + text).properties(width = 600)
```

Out[44]:



review	1	2	3	4	
6677	the	varied	opted	oily	ge
6678	wine	tequila	asked	argued	
6679	fooled	drink	sun	corner	w
6680	sammy	lb	resort	adequate	c
6681	cookbook	deceived	ha	never	fr
6682	amazingly	unbelievable	outstanding	value	
6683	mom	correct	together	sent	w
6684	into	snot	rocket	whose	he
6685	annoying	apart	five	pleasant	t
6686	brazen	oddness	simplicity	nostalgia	
6687	tango	big	vulgar	singing	
6688	positioning	taipan	favoring	selects	co
6689	railway	mineral	classic	included	t
6690	customer	the	thru	hate	
6691	slightly	above	ambiance	slow	
6692	the	averaged	wa	worth	wed
6693	difficulty	confusion	efficient	lovely	
6694	after	telephone	arrived	minute	p
6695	brazilian	excited	fan	mix	inte
6696	wal	mart	purchased	shift	mo

rank

TF-IDF with Dummy Classifier

In [45]:

```
# create tf-idf with dummy classifier
pipeline_dclf = make_pipeline(TfidfVectorizer(), DummyClassifier())

# fit the pipeline
pipeline_dclf.fit(X_train, Y_train)

# get the prediction
Y_pred = pipeline_dclf.predict(X_test)

# view the classification report
print(metrics.classification_report(Y_test, Y_pred, target_names = ["Positive", "Negati
ve"]))
```

```
              precision    recall  f1-score   support

    Positive       0.00      0.00      0.00       865
    Negative       0.51      1.00      0.67       886

    accuracy                           0.51      1751
   macro avg       0.25      0.50      0.34      1751
weighted avg       0.26      0.51      0.34      1751
```

TF-IDF with Bernoulli Naive Bayes

In [46]:

```
# create tf-idf with multinomial naive bayes
pipeline_bnb = make_pipeline(TfidfVectorizer(binary = True), BernoulliNB())

# fit the pipeline
pipeline_bnb.fit(X_train, Y_train)

# get the prediction
Y_pred = pipeline_bnb.predict(X_test)

# view the classification report
print(metrics.classification_report(Y_test, Y_pred, target_names = ["Positive", "Negati
ve"]))
```

```
              precision    recall  f1-score   support

    Positive       0.92      0.75      0.83       865
    Negative       0.80      0.93      0.86       886

    accuracy                           0.84      1751
   macro avg       0.86      0.84      0.84      1751
weighted avg       0.86      0.84      0.84      1751
```

TF-IDF with Multinomial Naive Bayes

In [47]:

```
# create tf-idf with multinomial naive bayes
pipeline_mnb = make_pipeline(TfidfVectorizer(), MultinomialNB())

# fit the pipeline
pipeline_mnb.fit(X_train, Y_train)

# get the prediction
Y_pred = pipeline_mnb.predict(X_test)

# view the classification report
print(metrics.classification_report(Y_test, Y_pred, target_names = ["Positive", "Negati
ve"]))
```

```
              precision    recall  f1-score   support

    Positive       0.94      0.95      0.95       865
    Negative       0.96      0.94      0.95       886

    accuracy                           0.95      1751
   macro avg       0.95      0.95      0.95      1751
weighted avg       0.95      0.95      0.95      1751
```

TF-IDF with Linear SVC

In [48]:

```
# create tf-idf with linear svc
pipeline_lsvc = make_pipeline(TfidfVectorizer(), LinearSVC(random_state = 123))

# fit the pipeline
pipeline_lsvc.fit(X_train, Y_train)

# get the prediction
Y_pred = pipeline_lsvc.predict(X_test)

# view the classification report
print(metrics.classification_report(Y_test, Y_pred, target_names = ["Positive", "Negati
ve"]))
```

```
              precision    recall  f1-score   support

    Positive       0.97      0.97      0.97       865
    Negative       0.97      0.97      0.97       886

    accuracy                           0.97      1751
   macro avg       0.97      0.97      0.97      1751
weighted avg       0.97      0.97      0.97      1751
```

Hyperparameter Tuning

In [49]:

```
# create list of pipelines
pipelines =  [pipeline_bnb, pipeline_mnb, pipeline_lsvc]

# select parameters for each one
param_bnb = {
    "bernoullinb__alpha": [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1],
    "bernoullinb__fit_prior": [True, False]
}
param_mnb = {
    "multinomialnb__alpha": [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1],
    "multinomialnb__fit_prior": [True, False]
}
param_lsvc = {
    "linearsvc__C": [1, 10, 100, 1000],
    "linearsvc__dual": [True, False],
    "linearsvc__random_state": [123]
}

# create list of params
params_list = [param_bnb, param_mnb, param_lsvc]
```

In [50]:

```
# Run gridsearch and use zip to attach each models to their parameters
best_clf = ""
best_score = -1
best_parameters = {}
for pipe, params in zip(pipelines, params_list):
    print("Testing parameters for {}".format(pipe))
    grid = GridSearchCV(estimator = pipe, param_grid = params, verbose = 0, cv = 5, n_j
obs = -1, scoring = "roc_auc").fit(X_train, Y_train)
    if grid.best_score_ > best_score :
        best_clf = grid.best_estimator_
        best_score = grid.best_score_
        best_parameters = grid.best_params_
        best_keys = list(grid.best_estimator_.named_steps.keys())
        best_coefs = grid.best_estimator_.named_steps[best_keys[1]].coef_[0]
        best_features = grid.best_estimator_.named_steps[best_keys[0]].get_feature_name
s_out()
    print("Best parameters set:")
    print(grid.best_estimator_.steps)
    print("\n")
    print(grid.best_score_)
    print("\n")
```

```
Testing parameters for Pipeline(steps=[('tfidfvectorizer', TfidfVectorizer
(binary=True)),
                ('bernoullinb', BernoulliNB())])
Best parameters set:
[('tfidfvectorizer', TfidfVectorizer(binary=True)), ('bernoullinb', Bernou
lliNB(alpha=0.1))]


0.958910677431035


Testing parameters for Pipeline(steps=[('tfidfvectorizer', TfidfVectorizer
()),
                ('multinomialnb', MultinomialNB())])
Best parameters set:
[('tfidfvectorizer', TfidfVectorizer()), ('multinomialnb', MultinomialNB(a
lpha=0.5))]


0.9913781096076457


Testing parameters for Pipeline(steps=[('tfidfvectorizer', TfidfVectorizer
()),
                ('linearsvc', LinearSVC(random_state=123))])
Best parameters set:
[('tfidfvectorizer', TfidfVectorizer()), ('linearsvc', LinearSVC(C=1, dual
=False, random_state=123))]


0.993795342596591
```

In [51]:

```
# create tf-idf with multinomial naive bayes
pipeline_bnb_best = make_pipeline(TfidfVectorizer(binary = True), BernoulliNB(alpha =
0.1))

# fit the pipeline
pipeline_bnb_best.fit(X_train, Y_train)

# get the prediction
Y_pred = pipeline_bnb_best.predict(X_test)

# view the classification report
print(metrics.classification_report(Y_test, Y_pred, target_names = ["Positive", "Negati
ve"]))
```

```
              precision    recall  f1-score   support

    Positive       0.92      0.80      0.85       865
    Negative       0.82      0.93      0.87       886

    accuracy                           0.86      1751
   macro avg       0.87      0.86      0.86      1751
weighted avg       0.87      0.86      0.86      1751
```

In [52]:

```
# create tf-idf with multinomial naive bayes
pipeline_mnb_best = make_pipeline(TfidfVectorizer(), MultinomialNB(alpha = 0.5))

# fit the pipeline
pipeline_mnb_best.fit(X_train, Y_train)

# get the prediction
Y_pred = pipeline_mnb_best.predict(X_test)

# view the classification report
print(metrics.classification_report(Y_test, Y_pred, target_names = ["Positive", "Negati
ve"]))
```

```
              precision    recall  f1-score   support

    Positive       0.94      0.95      0.95       865
    Negative       0.95      0.94      0.95       886

    accuracy                           0.95      1751
   macro avg       0.95      0.95      0.95      1751
weighted avg       0.95      0.95      0.95      1751
```

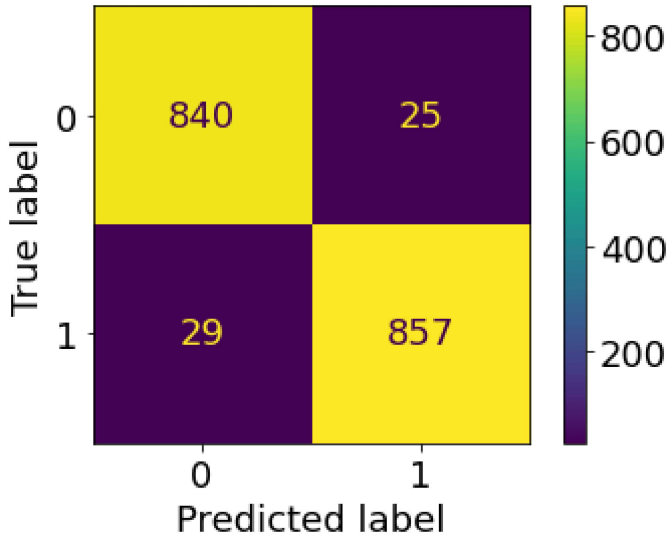
In [53]:

```
# create tf-idf with linear svc
pipeline_lsvc_best = make_pipeline(TfidfVectorizer(), LinearSVC(C = 1, dual = False, ra
ndom_state = 123))

# fit the pipeline
pipeline_lsvc_best.fit(X_train, Y_train)

# get the prediction
Y_pred = pipeline_lsvc_best.predict(X_test)

# view the classification report
print(metrics.classification_report(Y_test, Y_pred, target_names = ["Positive", "Negati
ve"]))
```

```
              precision    recall  f1-score   support

    Positive       0.97      0.97      0.97       865
    Negative       0.97      0.97      0.97       886

    accuracy                           0.97      1751
   macro avg       0.97      0.97      0.97      1751
weighted avg       0.97      0.97      0.97      1751
```

In [54]:

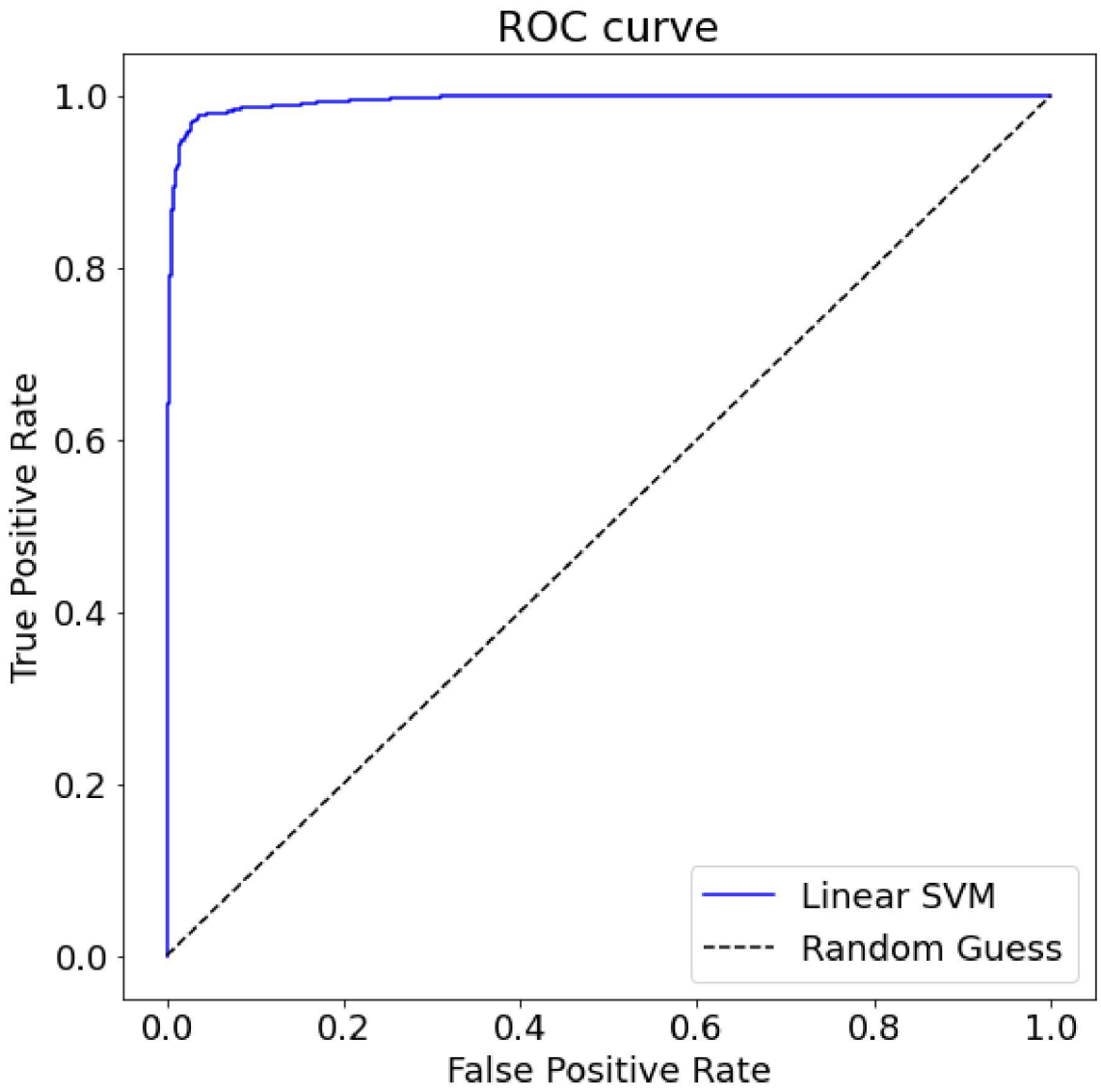
```
cm = confusion_matrix(Y_test, Y_pred, labels = pipeline_lsvc_best.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix = cm, display_labels = pipeline_lsvc_bes
t.classes_)
plt.rc('font', size = 18)
disp.plot()
plt.show()
```



In [55]:

```
decision_scores = pipeline_lsvc_best.decision_function(X_test)
fpr, tpr, thres = roc_curve(Y_test, decision_scores)
print("AUC: {:.3f}".format(roc_auc_score(Y_test, decision_scores)))

# roc curve
plt.figure(figsize = (9, 9))
plt.plot(fpr, tpr, "b", label = "Linear SVM")
plt.plot([0,1],[0,1], "k--", label = "Random Guess")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.legend(loc = "best")
plt.title("ROC curve")
plt.show()
```

```
AUC: 0.994
```



In [56]:

```
# get a missclassified review
misclassified = (Y_pred != Y_test)
X_test[misclassified]
```

Out[56]:

```
43                    no reasonably priced set lunch menu
83      they don have chef and they just warm the food...
178     we ate here last night after film the food wa ...
196     while visiting london and staying in brick lan...
205     if you have vegan in your group would recommen...
258     my favourite place in hong kong for peking duc...
281          nicest chinese restaurant have been in while
287     vinegarry wine bored looking staff and all the...
300     atmosphere is good however the food wa not ver...
325     this wa my first time and ca not wait until th...
386     could not recommend this restaurant highly enough
461     whilst good these are not much different from ...
483     the staff is bit untrained and difficult to ge...
532     certainly not for asian food gourmet tasteless...
536     one nice thing wa that they added gratuity on ...
545     traveling for work and my option were limited ...
549     friendly expected tagine to be piece rather th...
576     came here with large group for drink after bru...
580     this is easily one of if not the best meal we ...
633     went here because it wa close to the eiffel to...
659     our lunch wa wonderful and not rushed we start...
663     steak wa cook just the way requested medium ra...
677     whenever we order schnitzel we tend to overest...
725                           definitely le than expected
727     the breakfast sandwich were incredible and no ...
729     cosy cafe on popular street of haga they offer...
763     great food and atmosphere the bofsandwich just...
771     this used to be priority go to even though the...
795     being living in shanghai have been in several ...
842     this is the only pub around the area so worth ...
892     long wait but good service for the moment mcdo...
922                    really nice food for alright price
980     the flavor is unbelievably wrong have metal fl...
1039    very professional and kind service we took tes...
1087    family of for dinner and service wa very good ...
1093    it is really cute place with two floor then fe...
1109    they were able to provide an pa friendly meal ...
1125    the course or even more wa just mind blowing c...
1128    just hec of lot of food with little flavor blo...
1142    we went in unplanned based on high percentage ...
1177    we the worst experience ever my family came to...
1217    we had my birthday breakfast at mon ami gabi t...
1258    we had jolly evening last sunday having tried ...
1278    the restuarant ha made it way to world is top ...
1370    this is the grossest restaurant have had the m...
1439    their service is good but the food is not a go...
1442     this is my second time and it did not dissapoint
1476    good service and nice people but the food wa u...
1501    the restaurant is cosy and the food is good ye...
1572    our visit to the nd floor wa an absolute highl...
1588    our waitress wa kind and asked about allergy b...
1595    dropped in to the hyatt to meet someone pot of...
1718    this iconic restaurant ha fallen beyond recogn...
1728    had the lunch menu and it wa clearly not great...
Name: Reviews_Cleaned, dtype: object
```

In [57]:

```
test_df.iloc[83]
```

Out[57]:

```
Annotations                                                       0
Reviews            They don’t have a chef and they just warm the ...
Reviews_Cleaned    they don have chef and they just warm the food...
Name: 83, dtype: object
```

In [58]:

```
print(test_df["Reviews"][83])
```

```
They don’t have a chef and they just warm the foods because they’re alread
y cooked it and it comes in packaged when you order they just warm the foo
d and topped with oil to look nice
```

In [59]:

```
Y_pred[83]
```

Out[59]:

```
1
```

In [60]:

```
decision_scores = pipeline_dclf.predict_proba(X_test)[:, 1]
# fpr, tpr, thres = roc_curve(Y_test, decision_scores)
print("AUC: {:.2f}".format(roc_auc_score(Y_test, decision_scores)))
```

```
AUC: 0.50
```

In [61]:

```
decision_scores = pipeline_bnb_best.predict_proba(X_test)[:, 1]
fpr, tpr, thres = roc_curve(Y_test, decision_scores)
print("AUC: {:.2f}".format(roc_auc_score(Y_test, decision_scores)))
```

```
AUC: 0.96
```

In [62]:

```
decision_scores = pipeline_mnb_best.predict_proba(X_test)[:, 1]
# fpr, tpr, thres = roc_curve(Y_test, decision_scores)
print("AUC: {:.2f}".format(roc_auc_score(Y_test, decision_scores)))
```

```
AUC: 0.99
```

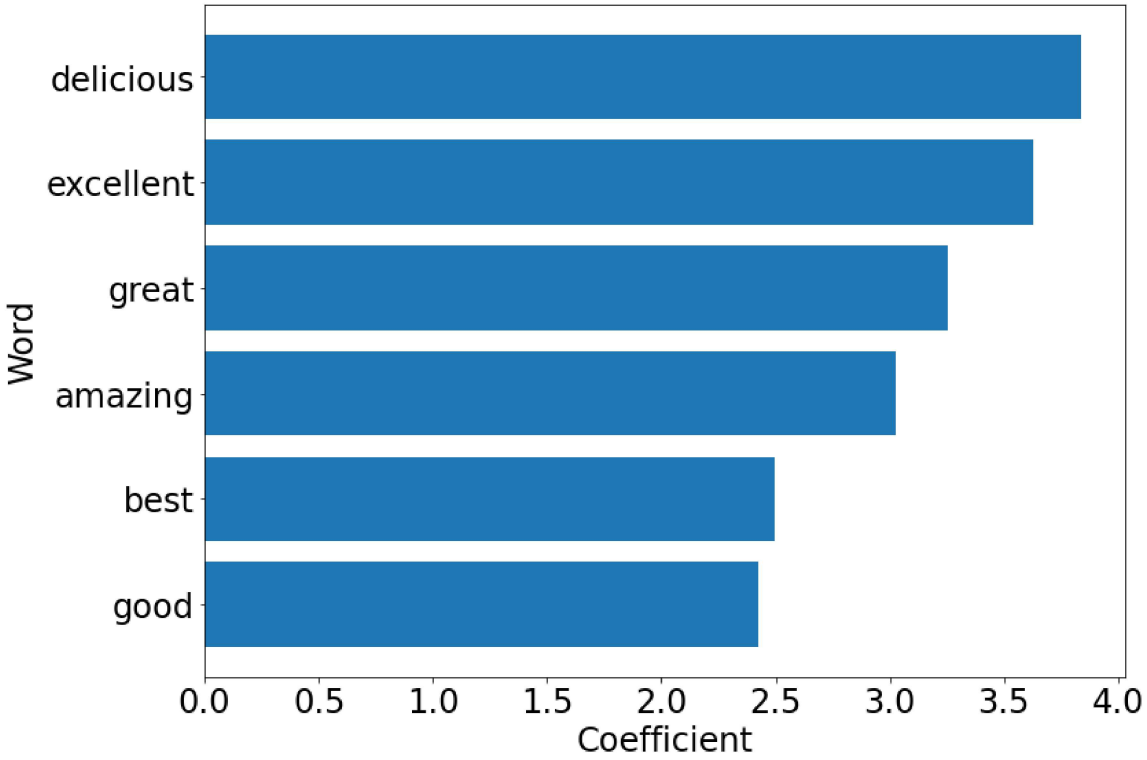
Feature Importance

In [63]:

```
def plot_important_words(coef, names, label, n = 10):
    coef_argsorted = np.argsort(coef)
    feature_names_array = np.array(names)
    words = []
    values = []

    if label == "Positive":
        for i in range(-n,0):
            words.append(names[coef_argsorted[i]])
            values.append(coef[coef_argsorted[i]])
    else:
        for i in range(n, 0, -1):
            words.append(names[coef_argsorted[i]])
            values.append(coef[coef_argsorted[i]])

    y = np.arange(n)
    plt.rc('font', size = 24)
    plt.barh(y, values)
    plt.yticks(y, words)
    plt.xlabel("Coefficient")
    plt.ylabel("Word")
    # plt.title(f"Feature Importance for {label} reviews")
    plt.show()
```

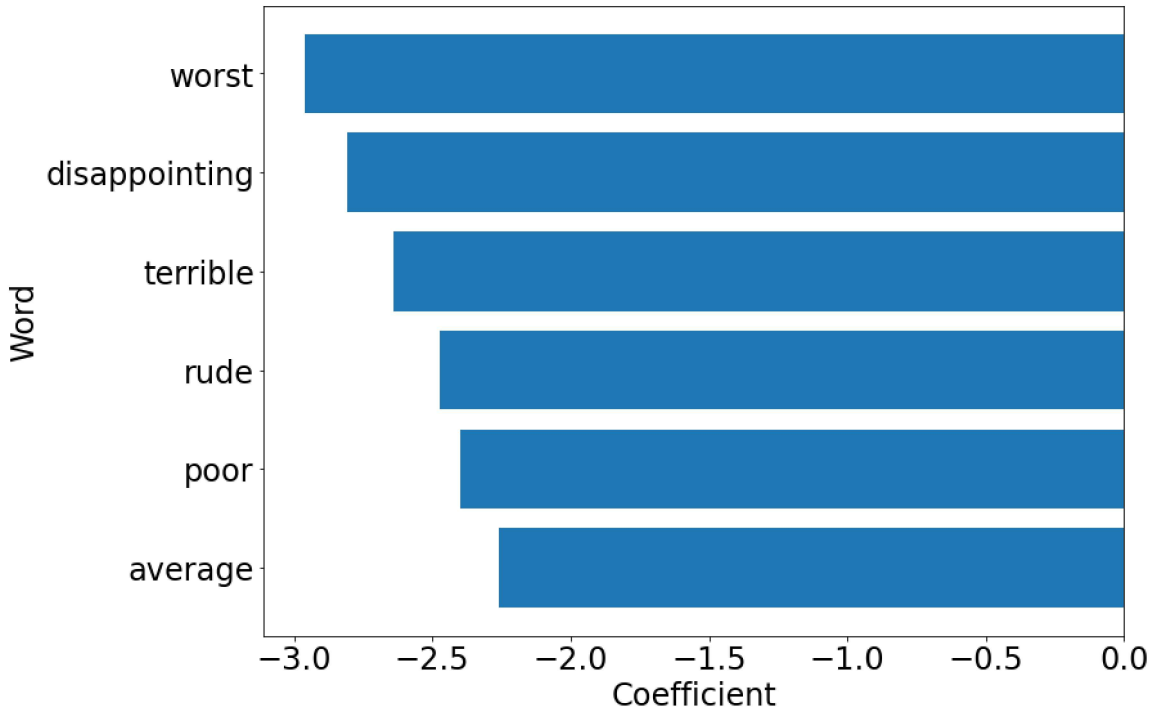
In [64]:

```
plt.figure(figsize = (12,9))
plot_important_words(best_coefs, best_features, "Positive", 6)
```



In [65]:

```
plt.figure(figsize = (12,9))
plot_important_words(best_coefs, best_features, "Negative", 6)
```



In [65]:

```
```