CSCI544: Homework Assignment 1

1. Dataset Preparation

The data was read with '/t' as the separator and there were a few rows in the dataset which were throwing errors so "error_bad_lines" was set to False and "quoting" was set to csv.QUOTE_NONE which means that do not quote anything on output.

Three sample reviews from the dataset along with the ratings

	star_rating	review_body
0	5	Beautiful. Looks great on counter.
1	5	I personally have 5 days sets and have also bought 2 sets for other people in my home. The two other sets I have decided to keep for myself. the purpose of keeping them for myself is to use them for other other than salt and pepper. they stay perfect, I use them constantly! I have a couple of people here that use them, say that there's just awesome. I did have a salt shaker that had a little problem converting from sea salt to Himalaya salt. did not fall out correctly so what I did was I just took a top of part which is really simple it's like five pieces total and just kind of cleaned around the Teflon and you know that the salt buildup from there was caused by humidity, cleaned out, my gosh it's unbelievable how much better, its almost better than new Thank You Bavaria these are top of the line in my book I hope in the near future you have some more come out. I sure would like to buy some more and some for my kids and family for Christmas otherwise I'm keeping what I have. I think I deserve it. thank you.... don't know if this makes much sense doesn't need to, self-ness !!!
2	5	Fabulous and worth every penny. Used for cleaning corn from the cob in seconds :) Would recommend its purchase

Statistics on Ratings:

Number of Reviews with Ratings: {5: 3128564, 1: 427306, 2: 242196, 3: 349929, 4: 732471}

Sentiment of Reviews: Positive: 3861035, Negative: 669502, Neutral: 349929

2. Data Cleaning

While cleaning the data, all non-alphabetical characters were removed, and only alphabets and the apostrophe was kept and this was to make sure the contraction function worked later in the pipeline. Contractions were done using the contraction library. URL's and HTML was removed using the Beautiful Soup Library. Removing extra spaces and non-alphabetical characters were done using the regex library.

Three Ratings before Data Cleaning and Preprocessing

	star_rating	review_body	Sentiment
0	5	Beautiful. Looks great on counter.	1
1	5	I personally have 5 days sets and have also bought 2 sets for other people in my home. The two other sets I have decided to keep for myself. the purpose of keeping them for myself is to use them for other other than salt and pepper. they stay perfect, I use them constantly! I have a couple of people here that use them, say that there's just awesome. I did have a salt shaker that had a little problem converting from sea salt to Himalaya salt. did not fall out correctly so what I did was I just took a top of part which is really simple it's like five pieces total and just kind of cleaned around the Teflon and you know that the salt buildup from there was caused by humidity, cleaned out, my gosh it's unbelievable how much better, its almost better than new Thank You Bavaria these are top of the line in my book I hope in the near future you have some more come out. I sure would like to buy some more and some for my kids and family for Christmas otherwise I'm keeping what I have. I think I deserve it. thank you.... don't know if this makes much sense doesn't need to, self-ness !!!	1
2	5	Fabulous and worth every penny. Used for cleaning corn from the cob in seconds :) Would recommend its purchase	1

Average Length Before Data Cleaning: 212.75001625381964

Average Length After Data Cleaning: 204.840155

3. Data Preprocessing

Lemmatization and removing stop words was done using the NLTK library

Average Length Before Pre Processing: 204.840155

Average Length After Pre Processing: 126.252915

Three Ratings After Data Cleaning and Preprocessing

	star_rating	review_body
0	5	beautiful look great counter
1	5	personally day set also bought set people home two set decided keep purpose keeping use salt pepper stay perfect use constantly couple people use say awesome salt shaker little problem converting sea salt himalaya salt fall correctly took top part really simple like five piece total kind cleaned around teflon know salt buildup caused humidity cleaned gosh unbelievable much better almost better new thank bavaria top line book hope near future come sure would like buy kid family christmas otherwise I keeping think deserve thank know make much sense need selfness
2	5	fabulous worth every penny used cleaning corn cob second would recommend purchase

4. Feature Extraction

Feature extraction was done after splitting the dataset into test and training sets in the ratio (20:80) and limited to 5000 maximum features. The tf-idf vectorizer from the sklearn library was used to fit and transform the training set and just transform the test set (used exclusively on the review_body column).

5. Perceptron

Used the sklearn.linear_model Perceptron library as the model for this task

Accuracy 0.8612 Precision 0.8418 Recall 0.8933 and f1-score 0.8668 for Perceptron on training data
Accuracy 0.8481 Precision 0.8279 Recall 0.8806 and f1-score 0.8534 for Perceptron on test data

6. SVM

Used the sklearn.svm LinearSVC library as the model for this task

Accuracy 0.9111 Precision 0.9147 Recall 0.9068 and f1-score 0.9108 for SVM on training data
Accuracy 0.8965 Precision 0.9011 Recall 0.8897 and f1-score 0.8954 for SVM on test data

7. Logistic Regression

Used the sklearn.linear_model LogisticRegression library as the model for this task

Accuracy 0.9082 Precision 0.9142 Recall 0.9012 and f1-score 0.9077 for Logistic Regression on training data
Accuracy 0.8985 Precision 0.9049 Recall 0.8896 and f1-score 0.8972 for Logistic Regression on test data

8. Naive Bayes

Used the sklearn.naive_bayes MultinomialNB library as the model for this task

Accuracy 0.8827 Precision 0.8810 Recall 0.8853 and f1-score 0.8831 for Naive Bayes on training data
Accuracy 0.8763 Precision 0.8729 Recall 0.8797 and f1-score 0.8763 for Naive Bayes on test data

In [1]:
```
import pandas as pd
import numpy as np
import nltk
nltk.download('wordnet')
import re
from bs4 import BeautifulSoup
import csv
import contractions
pd.set_option('display.max_colwidth', -1)
from IPython.display import display
```

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

In [2]:
```
#! pip install bs4 # in case you don't have it installed

# Dataset: https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Kit
```

Read Data

In [3]:
```
df = pd.read_csv("amazon_reviews_us_Kitchen_v1_00.tsv", sep='\t', error_bad_lin
```

Keep Reviews and Ratings

In [4]:
```
df = df[['star_rating','review_body']]
df = df.dropna(subset=['star_rating'])
df['star_rating'] = df['star_rating'].astype(int)
display(df.head(3))
```

	star_rating	review_body
0	5	Beautiful. Looks great on counter.
1	5	I personally have 5 days sets and have also bought 2 sets for other people in my home. The two other sets I have decided to keep for myself. the purpose of keeping them for myself is to use them for other other than salt and pepper. they stay perfect, I use them constantly! I have a couple of people here that use them, say that there's just awesome. I did have a salt shaker that had a little problem converting from sea salt to Himalaya salt. did not fall out correctly so what I did was I just took a top of part which is really simple it's like five pieces total and just kind of cleaned around the Teflon and you know that the salt buildup from there was caused by humidity, cleaned out, my gosh it's unbelievable how much better, its almost better than new Thank You Bavaria these are top of the line in my book I hope in the near future you have some more come out. I sure would like to buy some more and some for my kids and family for Christmas otherwise I'm keeping what I have. I think I deserve it. thank you.... don't know if this makes much sense doesn't need to, self-ness !!!
2	5	Fabulous and worth every penny. Used for cleaning corn from the cob in seconds :) Would recommend its purchase

In [5]:
```
ratings = {5:0, 1:0, 2:0, 3:0, 4:0}
sentiment = {'Postive':0, 'Negative':0, 'Neutral':0}
def count_rating(x):
    ratings[int(x)]+=1
    if int(x)>3:
        sentiment['Postive']+=1
    elif int(x)<3:
        sentiment['Negative']+=1
    else:
        sentiment['Neutral']+=1
result = [count_rating(x) for x in df['star_rating']]
print("Ratings: " + str(ratings))
print("Positive: " + str(sentiment['Postive']) + ", Negative: " + str(sentiment[
```

```
Ratings: {5: 3128564, 1: 427306, 2: 242196, 3: 349929, 4: 732471}
Positive: 3861035, Negative: 669502, Neutral: 349929
```

Labelling Reviews:

The reviews with rating 4,5 are labelled to be 1 and 1,2 are labelled as 0. Discard the reviews with rating 3'

In [6]:
```
df = df[df['star_rating'] != 3]
df['Sentiment'] = [1 if x > 3 else 0 for x in df['star_rating']]
```

We select 200000 reviews randomly with 100,000 positive and 100,000 negative reviews.

In [7]:
```
df_postive = df[(df.Sentiment == 1)].head(100000)
df_negative = df[(df.Sentiment == 0)].head(100000)
frames = [df_postive, df_negative]
df = pd.concat(frames)
```

Data Cleaning

Convert the all reviews into the lower case.

In [8]:
```
print("3 Ratings before Data Cleaning and Preprocessing ")
display(df.head(3))
print("Average Length Before Data Cleaning: " + str(df['review_body'].str.len().
```

```
3 Ratings before Data Cleaning and Preprocessing
```

	star_rating	review_body	Sentiment
0	5	Beautiful. Looks great on counter.	1
1	5	I personally have 5 days sets and have also bought 2 sets for other people in my home. The two other sets I have decided to keep for myself. the purpose of keeping them for myself is to use them for other other than salt and pepper. they stay perfect, I use them constantly! I have a couple of people here that use them, say that there's just awesome. I did have a salt shaker that had a little problem converting from sea salt to Himalaya salt. did not fall out correctly so what I did was I just took a top of part which is really simple it's like five pieces total and just kind of cleaned around the Teflon and you know that the salt buildup from there was caused by humidity, cleaned out, my gosh it's unbelievable how much better, its almost better than new Thank You Bavaria these are top of the line in my book I hope in the near future you have some more come out. I sure would like to buy some more and some for my kids and family for Christmas otherwise I'm keeping what I have. I think I deserve it. thank you.... don't know if this makes much sense doesn't need to, self-ness !!!	1
2	5	Fabulous and worth every penny. Used for cleaning corn from the cob in seconds :) Would recommend its purchase	1

```
Average Length Before Data Cleaning: 212.75001625381964
```

In [9]:
```
df['review_body'] = df['review_body'].str.lower()
```

remove the HTML and URLs from the reviews

In [10]:
```
def remove_HTML(review):
    cleanText = BeautifulSoup(review, "html.parser").text
    return cleanText

df['review_body'] = df['review_body'].apply(lambda review : remove_HTML(str(revi
```

```
C:\Users\acer\Anaconda3\lib\site-packages\bs4\__init__.py:336: UserWarning: "ht
tp://www.amazon.com/gp/product/b00mg6zezm?psc=1&redirect=true&ref_=od_aui_detai
lpages00" looks like a URL. Beautiful Soup is not an HTTP client. You should pr
obably use an HTTP client like requests to get the document behind the URL, and
feed that document to Beautiful Soup.
  ' that document to Beautiful Soup.' % decoded_markup
C:\Users\acer\Anaconda3\lib\site-packages\bs4\__init__.py:336: UserWarning: "ht
tp://www.amazon.com/review/create-review/ref=cm_cr_dp_wrt_summary?ie=utf8&asin=
b00xp0d9p0" looks like a URL. Beautiful Soup is not an HTTP client. You should
probably use an HTTP client like requests to get the document behind the URL, a
nd feed that document to Beautiful Soup.
  ' that document to Beautiful Soup.' % decoded_markup
C:\Users\acer\Anaconda3\lib\site-packages\bs4\__init__.py:336: UserWarning: "ht
tp://www.amazon.com/gp/product/b00rc1xfkm?redirect=true&ref_=cm_cr_ryp_prd_ttl_
sol_0" looks like a URL. Beautiful Soup is not an HTTP client. You should proba
bly use an HTTP client like requests to get the document behind the URL, and fe
ed that document to Beautiful Soup.
  ' that document to Beautiful Soup.' % decoded_markup
C:\Users\acer\Anaconda3\lib\site-packages\bs4\__init__.py:336: UserWarning: "ht
tp://www.asbestosnation.org/facts/tests-find-asbestos-in-kids-crayons-crime-sce
ne-kits/" looks like a URL. Beautiful Soup is not an HTTP client. You should pr
obably use an HTTP client like requests to get the document behind the URL, and
feed that document to Beautiful Soup.
  ' that document to Beautiful Soup.' % decoded_markup
C:\Users\acer\Anaconda3\lib\site-packages\bs4\__init__.py:336: UserWarning: "ht
tp://www.consumerreports.org/cro/magazine-archive/2011/november/appliances/can-
you-stop-stirring/overview/index.htm" looks like a URL. Beautiful Soup is not a
n HTTP client. You should probably use an HTTP client like requests to get the
document behind the URL, and feed that document to Beautiful Soup.
  ' that document to Beautiful Soup.' % decoded_markup
```

remove non-alphabetical characters

In [11]:
```
regex = re.compile('[^a-zA-Z \']')
df['review_body'] = df['review_body'].apply(lambda review : regex.sub('', review
```

Remove the extra spaces between the words

In [12]:
```
regex = re.compile('[ +]')
df['review_body'] = df['review_body'].apply(lambda review : regex.sub(' ', revie
```

perform contractions on the reviews.

In [13]:
```
def contractionfunction(s):
    return contractions.fix(s)
df['review_body'] = df['review_body'].apply(lambda review :contractionfunction(r
```

In [14]:
```
print("Average Length After Data Cleaning: " + str(df['review_body'].str.len().me
```

```
Average Length After Data Cleaning: 204.840155
```

Pre-processing

remove the stop words

In [15]:
```
print("Average Length Before Pre Processing: " + str(df['review_body'].str.len()
```

```
Average Length Before Pre Processing: 204.840155
```

In [16]:
```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

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

def remove_stopWords(review):
    tokens = word_tokenize(review)
    filtered_words = [word for word in tokens if word not in stop_words]
    return " ".join(filtered_words)

df['review_body'] = df['review_body'].apply(lambda review : remove_stopWords(rev
```

perform lemmatization

In [17]:
```
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import WhitespaceTokenizer

torkenizer = WhitespaceTokenizer()
lemmatizer = WordNetLemmatizer()

def lemmatize_text(review):
    lemmatized_words = [lemmatizer.lemmatize(w) for w in torkenizer.tokenize(rev
    return " ".join(lemmatized_words)

df['review_body'] = df['review_body'].apply(lambda review : lemmatize_text(revie
```

In [18]:
```
print("Average Length After Pre Processing: " + str(df['review_body'].str.len().
print("3 Ratings After Data Cleaning and Preprocessing")
display(df.head(3))
```

```
Average Length After Pre Processing: 126.252915
3 Ratings After Data Cleaning and Preprocessing
```

	star_rating	review_body	Sentiment
0	5	beautiful look great counter	1
1	5	personally day set also bought set people home two set decided keep purpose keeping use salt pepper stay perfect use constantly couple people use say awesome salt shaker little problem converting sea salt himalaya salt fall correctly took top part really simple like five piece total kind cleaned around teflon know salt buildup caused humidity cleaned gosh unbelievable much better almost better new thank bavaria top line book hope near future come sure would like buy kid family christmas otherwise I keeping think deserve thank know make much sense need selfness	1
2	5	fabulous worth every penny used cleaning corn cob second would recommend purchase	1

TF-IDF Feature Extraction

In [19]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_score, recall_score, f1_score, accuracy_sc

X_train, X_test, y_train, y_test = train_test_split(df['review_body'],df['Sentim
```

In [20]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=5000)
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
```

Perceptron

In [25]:
```
from sklearn.linear_model import Perceptron

clf = Perceptron(random_state=0)

clf.fit(X_train, y_train)

y_pred_train = clf.predict(X_train)
accuracy_train = accuracy_score(y_train, y_pred_train)
precision_score_train = precision_score(y_train, y_pred_train)
recall_score_train = recall_score(y_train, y_pred_train)
f1_score_train = f1_score(y_train, y_pred_train)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for Percept

y_pred_test= clf.predict(X_test)
accuracy_test = accuracy_score(y_test, y_pred_test)
precision_score_test = precision_score(y_test, y_pred_test)
recall_score_test = recall_score(y_test, y_pred_test)
f1_score_test = f1_score(y_test, y_pred_test)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for Percept
```

```
C:\Users\acer\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic_gradi
ent.py:166: FutureWarning: max_iter and tol parameters have been added in Perce
ptron in 0.19. If both are left unset, they default to max_iter=5 and tol=None.
If tol is not None, max_iter defaults to max_iter=1000. From 0.21, default max_
iter will be 1000, and default tol will be 1e-3.
  FutureWarning)

Accuracy 0.8612 Precision 0.8684 Recall 0.8518 and f1-score 0.8601 for Perceptr
on on training data
Accuracy 0.8481 Precision 0.8565 Recall 0.8347 and f1-score 0.8455 for Perceptr
on on test data
```

SVM

In [22]:
```
from sklearn.svm import LinearSVC

clf = LinearSVC()

clf.fit(X_train, y_train)

y_pred_train = clf.predict(X_train)
accuracy_train = accuracy_score(y_train, y_pred_train)
precision_score_train = precision_score(y_train, y_pred_train)
recall_score_train = recall_score(y_train, y_pred_train)
f1_score_train = f1_score(y_train, y_pred_train)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for SVM on

y_pred_test= clf.predict(X_test)
accuracy_test = accuracy_score(y_test, y_pred_test)
precision_score_test = precision_score(y_test, y_pred_test)
recall_score_test = recall_score(y_test, y_pred_test)
f1_score_test = f1_score(y_test, y_pred_test)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for SVM on
```

```
Accuracy 0.9111 Precision 0.9147 Recall 0.9068 and f1-score 0.9108 for SVM on t
raining data
Accuracy 0.8965 Precision 0.9011 Recall 0.8897 and f1-score 0.8954 for SVM on t
est data
```

Logistic Regression

In [23]:
```
from sklearn.linear_model import LogisticRegression


clf = LogisticRegression(random_state=0)

clf.fit(X_train, y_train)

y_pred_train = clf.predict(X_train)
accuracy_train = accuracy_score(y_train, y_pred_train)
precision_score_train = precision_score(y_train, y_pred_train)
recall_score_train = recall_score(y_train, y_pred_train)
f1_score_train = f1_score(y_train, y_pred_train)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for Logisti

y_pred_test= clf.predict(X_test)
accuracy_test = accuracy_score(y_test, y_pred_test)
precision_score_test = precision_score(y_test, y_pred_test)
recall_score_test = recall_score(y_test, y_pred_test)
f1_score_test = f1_score(y_test, y_pred_test)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for Logisti
```

```
C:\Users\acer\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433:
FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a sol
ver to silence this warning.
  FutureWarning)

Accuracy 0.9082 Precision 0.9142 Recall 0.9012 and f1-score 0.9077 for Logistic
Regression on training data
Accuracy 0.8985 Precision 0.9049 Recall 0.8896 and f1-score 0.8972 for Logistic
Regression on test data
```

Naive Bayes

In [24]:
```
from sklearn.naive_bayes import MultinomialNB


clf =  MultinomialNB()

clf.fit(X_train, y_train)

y_pred_train = clf.predict(X_train)
accuracy_train = accuracy_score(y_train, y_pred_train)
precision_score_train = precision_score(y_train, y_pred_train)
recall_score_train = recall_score(y_train, y_pred_train)
f1_score_train = f1_score(y_train, y_pred_train)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for Naive B

y_pred_test= clf.predict(X_test)
accuracy_test = accuracy_score(y_test, y_pred_test)
precision_score_test = precision_score(y_test, y_pred_test)
recall_score_test = recall_score(y_test, y_pred_test)
f1_score_test = f1_score(y_test, y_pred_test)

print("Accuracy %2.4f Precision %2.4f Recall %2.4f and f1-score %2.4f for Naive B
```

```
Accuracy 0.8827 Precision 0.8810 Recall 0.8853 and f1-score 0.8831 for Naive Ba
yes on training data
Accuracy 0.8763 Precision 0.8729 Recall 0.8797 and f1-score 0.8763 for Naive Ba
yes on test data
```