Configuration

```
# Parameters
PROJECT_NAME = 'ML1010-Group-Project'
ENABLE_COLAB = True

#Root Machine Learning Directory. Projects appear underneath
GOOGLE_DRIVE_MOUNT = '/content/gdrive'
COLAB_ROOT_DIR = GOOGLE_DRIVE_MOUNT + '/MyDrive/Colab Notebooks'
COLAB_INIT_DIR = COLAB_ROOT_DIR + '/utility_files'

LOCAL_ROOT_DIR = '/home/magni/Documents/ML_Projects'
LOCAL_INIT_DIR = LOCAL_ROOT_DIR + '/utility_files'
```

Bootstrap Environment

```
#add in support for utility file directory and importing
import sys
import os

if ENABLE_COLAB:
  #Need access to drive
  from google.colab import drive
  drive.mount(GOOGLE_DRIVE_MOUNT, force_remount=True)

  #add in utility directory to syspath to import
  INIT_DIR = COLAB_INIT_DIR
  sys.path.append(os.path.abspath(INIT_DIR))

  #Config environment variables
  ROOT_DIR = COLAB_ROOT_DIR

else:
  #add in utility directory to syspath to import
  INIT_DIR = LOCAL_INIT_DIR
  sys.path.append(os.path.abspath(INIT_DIR))

  #Config environment variables
  ROOT_DIR = LOCAL_ROOT_DIR

#Import Utility Support
from jarvis import Jarvis
jarvis = Jarvis(ROOT_DIR, PROJECT_NAME)
```

```
import mv_python_utils as mvutils
```

```
Mounted at /content/gdrive
Wha...where am I?
I am awake now.

I have set your current working directory to /content/gdrive/MyDrive/Colab Notebooks/ML
The current time is 18:57
Hello sir. Reminder, no more coffee.
```

Setup Runtime Environment

```
if ENABLE_COLAB:
  #!pip install scipy -q
  #!pip install scikit-learn -q
  #!pip install pycaret -q
  #!pip install matplotlib -q
  #!pip install joblib -q
  #!pip install pandasql -q
  !pip install umap_learn -q
  !pip install sentence_transformers -q
  !pip install spacytextblob -q
  !pip install flair -q
  display('Google Colab enabled')
else:
  display('Google Colab not enabled')

#Common imports
import json
import pandas as pd
import numpy as np
import matplotlib
import re
import nltk
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn import metrics
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split as tts
#from yellowbrick.classifier import ConfusionMatrix
#from sklearn.linear_model import LogisticRegression
from yellowbrick.target import ClassBalance
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
```

```
nltk.download('stopwords')
%matplotlib inline
```

```
     |████████████████████████████████| 86 kB 4.1 MB/s
     |████████████████████████████████| 1.1 MB 31.9 MB/s
  Building wheel for umap-learn (setup.py) ... done
  Building wheel for pynndescent (setup.py) ... done
     |████████████████████████████████| 78 kB 4.1 MB/s
     |████████████████████████████████| 3.1 MB 14.1 MB/s
     |████████████████████████████████| 3.3 MB 56.7 MB/s
     |████████████████████████████████| 1.2 MB 56.5 MB/s
     |████████████████████████████████| 61 kB 599 kB/s
     |████████████████████████████████| 895 kB 69.0 MB/s
     |████████████████████████████████| 596 kB 60.0 MB/s
  Building wheel for sentence-transformers (setup.py) ... done
     |████████████████████████████████| 6.0 MB 6.0 MB/s
     |████████████████████████████████| 42 kB 1.6 MB/s
     |████████████████████████████████| 451 kB 74.2 MB/s
     |████████████████████████████████| 628 kB 78.0 MB/s
     |████████████████████████████████| 10.1 MB 52.5 MB/s
     |████████████████████████████████| 181 kB 74.4 MB/s
     |████████████████████████████████| 322 kB 7.3 MB/s
     |████████████████████████████████| 48 kB 5.7 MB/s
     |████████████████████████████████| 19.7 MB 1.5 MB/s
     |████████████████████████████████| 981 kB 68.3 MB/s
     |████████████████████████████████| 788 kB 51.3 MB/s
     |████████████████████████████████| 64 kB 3.2 MB/s
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
    Preparing wheel metadata ... done
     |████████████████████████████████| 1.2 MB 16.9 MB/s
     |████████████████████████████████| 62 kB 977 kB/s
  Building wheel for gdown (PEP 517) ... done
  Building wheel for mpld3 (setup.py) ... done
  Building wheel for overrides (setup.py) ... done
  Building wheel for segtok (setup.py) ... done
  Building wheel for sqlitedict (setup.py) ... done
  Building wheel for ftfy (setup.py) ... done
  Building wheel for langdetect (setup.py) ... done
  Building wheel for wikipedia-api (setup.py) ... done
ERROR: pip's dependency resolver does not currently take into account all the packages
markdown 3.3.6 requires importlib-metadata>=4.4; python_version < "3.10", but you have
google-colab 1.0.0 requires requests~=2.23.0, but you have requests 2.26.0 which is inc
datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is incompati
'Google Colab enabled'
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
import cw_df_metric_utils as cwutils
```

```
axis_labels5=[1,2,3,4,5]
axis_labels2=[0,1]
```

Load Data

```
jarvis.getPackageVersion('pandas')
!python -V
jarvis.showProjectDataFiles()
```

```
     pandas version: pandas 1.1.5
     Python 3.7.12
     Here are all your project data files
     [D] /content/gdrive/MyDrive/Colab Notebooks/data/ML1010-Group-Project [Empty directory]

     [D] /content/gdrive/MyDrive/Colab Notebooks/data/ML1010-Group-Project/01_original
     ---[   gz][ json]--> Cell_Phones_and_Accessories_5.json.gz (161.24 MB)
     ---[   gz][ json]--> meta_Cell_Phones_and_Accessories.json.gz (343.33 MB)

     [D] /content/gdrive/MyDrive/Colab Notebooks/data/ML1010-Group-Project/02_working
     [*][  pkl]---------> 01_Cellphone_small.pkl (45.46 MB)
     ---[   gz][  pkl]--> 01_NLP_ReviewText_Narrow_1.pkl.gz (6.88 MB)
     ---[   gz][  pkl]--> 01_NLP_ReviewText_Narrow_2.pkl.gz (170.55 MB)
     ---[   gz][  pkl]--> 01_NLP_ReviewText_Narrow_3.pkl.gz (295.59 MB)
     [*][  pkl]---------> 01_NLP_ReviewText_small.pkl (28.94 MB)
     [*][  pkl]---------> 01_NLP_Summary_small.pkl (3.82 MB)
     [*][  pkl]---------> 01_NLP_Title_small.pkl (2.73 MB)
     ---[   gz][  pkl]--> 01_NL_ReviewText_All(new).pkl.gz (593.23 MB)
     ---[   gz][  pkl]--> 01_NL_ReviewText_All.pkl.gz (592.92 MB)
     ---[   gz][  pkl]--> 01_NL_ReviewText_textSplit.pkl.gz (15.78 MB)
     [*][  pkl]---------> 02_Cellphone.pkl (46.32 MB)
     [*][  pkl]---------> 02_NLP_ReviewTextData.pkl (87.00 MB)
     [*][  pkl]---------> 02_NLP_SummaryData.pkl (8.32 MB)
     [*][  pkl]---------> 02_NLP_TitleData.pkl (16.71 MB)
     [*][  pkl]---------> 03_Cellphone.pkl (46.31 MB)
     [*][  pkl]---------> 03_NLP_ReviewTextData.pkl (28.94 MB)
     [*][  pkl]---------> 03_NLP_ReviewText_Narrow.pkl (17.13 MB)
     [*][  pkl]---------> 03_NLP_SummaryData.pkl (3.82 MB)
     [*][  pkl]---------> 03_NLP_TitleData.pkl (2.73 MB)
     [*][  pkl]---------> 04_NLP_ReviewText_Narrow.pkl (16.95 MB)
     [*][  pkl]---------> 05_NLP_ReviewText_Narrow.pkl (66.15 MB)
     [*][  pkl]---------> 05_NLP_ReviewText_Narrow_full.pkl (207.91 MB)

     [D] /content/gdrive/MyDrive/Colab Notebooks/data/ML1010-Group-Project/03_train [Empty d

     [D] /content/gdrive/MyDrive/Colab Notebooks/data/ML1010-Group-Project/04_test [Empty di
```

```
dataSrc = pd.read_pickle(jarvis.DATA_DIR_WORK + "/01_NL_ReviewText_All(new).pkl.gz")
mvutils.exploreDataframe (dataSrc, numRecords=1)
```

```
 15  reviewText_adjectives_tb_subj           50732 non-null  float64
 16  reviewText_adjectives_tb_tokens         50732 non-null  float64
 17  reviewText_adjectives_tb_length         50732 non-null  float64
 18  reviewText_adjectives_bert              63413 non-null  object
 19  reviewText_adjectives_flairSent         50732 non-null  float64
 20  reviewText_verbs_tb_pol                 43234 non-null  float64
 21  reviewText_verbs_tb_subj                43234 non-null  float64
 22  reviewText_verbs_tb_tokens              43234 non-null  float64
 23  reviewText_verbs_tb_length              43234 non-null  float64
 24  reviewText_verbs_bert                   63413 non-null  object
 25  reviewText_verbs_flairSent              43234 non-null  float64
 26  reviewText_nav_tb_pol                   62332 non-null  float64
 27  reviewText_nav_tb_subj                  62332 non-null  float64
 28  reviewText_nav_tb_tokens                62332 non-null  float64
 29  reviewText_nav_tb_length                62332 non-null  float64
 30  reviewText_nav_bert                     63413 non-null  object
 31  reviewText_nav_flairSent                62332 non-null  float64
 32  overall_posneg                          63413 non-null  int64
 33  reviewText_lemma_flairSent_norm         63310 non-null  float64
 34  reviewText_lemma_flairSent_posneg       63310 non-null  float64
 35  reviewText_adjectives_flairSent_norm    50732 non-null  float64
 36  reviewText_adjectives_flairSent_posneg  50732 non-null  float64
 37  reviewText_verbs_flairSent_norm         43234 non-null  float64
 38  reviewText_verbs_flairSent_posneg       43234 non-null  float64
 39  reviewText_nav_flairSent_norm           62332 non-null  float64
 40  reviewText_nav_flairSent_posneg         62332 non-null  float64
 41  reviewText_lemma_tb_pol_norm            63310 non-null  float64
 42  reviewText_lemma_tb_pol_posneg          63310 non-null  float64
 43  reviewText_adjectives_tb_pol_norm       50732 non-null  float64
 44  reviewText_adjectives_tb_pol_posneg     50732 non-null  float64
 45  reviewText_verbs_tb_pol_norm            43234 non-null  float64
 46  reviewText_verbs_tb_pol_posneg          43234 non-null  float64
 47  reviewText_nav_tb_pol_norm              62332 non-null  float64
 48  reviewText_nav_tb_pol_posneg            62332 non-null  float64
dtypes: float64(37), int64(1), object(11)
memory usage: 23.7+ MB
None
```

Top 1 in dataframe

	uuid	reviewText	overall	reviewText_lemma	reviewText_nouns	reviewText_adj
0	e5322688-1105-401b-be69-888bc1d89bcf	This phone is ugly and heavy and has a terribl...	1.0	phone ugly heavy terrible user interface techi...	phone user interface techie call Manhattan Mot...	ugly heavy terr well well we

Bottom 1 in dataframe

	uuid	reviewText	overall	reviewText_lemma	reviewText_nouns	reviewTex
63412	8f71ec3b-73e0-408e-b7e0-acd146f697e7	This is a great Smartphone for someone who's n...	4.0	great smartphone entirely sure want smartphone...	smartphone smartphone cost Samsung thing quali...	great sure low ab

Lemma - dropNA and Balance (2 class and 5 class)

```
dataCoreLemma = dataSrc[['uuid',
                         'overall',
                         'overall_posneg',
                         'reviewText_lemma',
                         'reviewText_lemma_tb_pol',
                         'reviewText_lemma_tb_subj',
                         'reviewText_lemma_tb_tokens',
                         'reviewText_lemma_tb_length',
                         'reviewText_lemma_bert',
                         'reviewText_lemma_flairSent',
                         'reviewText_lemma_flairSent_norm',
                         'reviewText_lemma_flairSent_posneg',
                         'reviewText_lemma_tb_pol_norm',
                         'reviewText_lemma_tb_pol_posneg'
                        ]].copy()
```

```
dataCoreLemma.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 63413 entries, 0 to 63412
Data columns (total 14 columns):
```

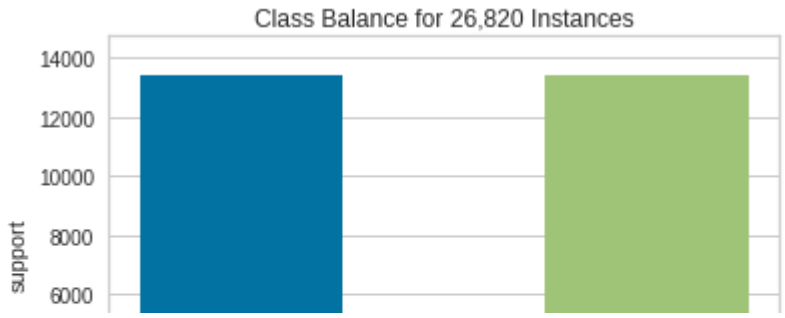
```
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   uuid                            63413 non-null  object 
 1   overall                         63413 non-null  float64
 2   overall_posneg                  63413 non-null  int64  
 3   reviewText_lemma                63413 non-null  object 
 4   reviewText_lemma_tb_pol         63310 non-null  float64
 5   reviewText_lemma_tb_subj        63310 non-null  float64
 6   reviewText_lemma_tb_tokens      63310 non-null  float64
 7   reviewText_lemma_tb_length      63310 non-null  float64
 8   reviewText_lemma_bert           63413 non-null  object 
 9   reviewText_lemma_flairSent      63310 non-null  float64
 10  reviewText_lemma_flairSent_norm 63310 non-null  float64
 11  reviewText_lemma_flairSent_posneg 63310 non-null  float64
 12  reviewText_lemma_tb_pol_norm    63310 non-null  float64
 13  reviewText_lemma_tb_pol_posneg  63310 non-null  float64
dtypes: float64(10), int64(1), object(3)
memory usage: 6.8+ MB
```

```
#Drop null values from flairSent
dataCoreLemma.dropna(subset=['reviewText_lemma_flairSent'], inplace=True)
dataCoreLemma.reset_index(drop=True, inplace=True)
dataCoreLemma.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 63310 entries, 0 to 63309
Data columns (total 14 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   uuid                            63310 non-null  object 
 1   overall                         63310 non-null  float64
 2   overall_posneg                  63310 non-null  int64  
 3   reviewText_lemma                63310 non-null  object 
 4   reviewText_lemma_tb_pol         63310 non-null  float64
 5   reviewText_lemma_tb_subj        63310 non-null  float64
 6   reviewText_lemma_tb_tokens      63310 non-null  float64
 7   reviewText_lemma_tb_length      63310 non-null  float64
 8   reviewText_lemma_bert           63310 non-null  object 
 9   reviewText_lemma_flairSent      63310 non-null  float64
 10  reviewText_lemma_flairSent_norm 63310 non-null  float64
 11  reviewText_lemma_flairSent_posneg 63310 non-null  float64
 12  reviewText_lemma_tb_pol_norm    63310 non-null  float64
 13  reviewText_lemma_tb_pol_posneg  63310 non-null  float64
dtypes: float64(10), int64(1), object(3)
memory usage: 6.8+ MB
```

```
dataCoreLemmaBal2 = mvutils.classBalanceUndersample(dataCoreLemma, 'overall_posneg')
```


```
Undersampling data to match min class: 0 of size: 13410
```



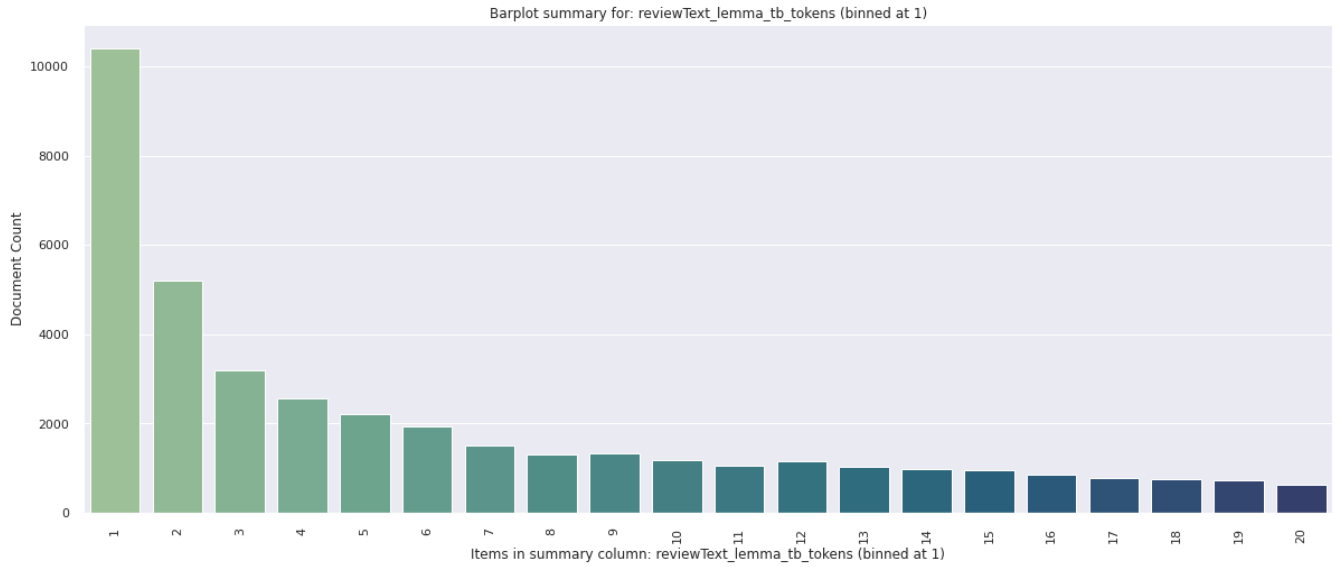

```
dataCoreLemmaBal5 = mvutils.classBalanceUndersample(dataCoreLemma, 'overall')
```

Lemma - Prune data and Balance (2 class and 5 class)

```
dataCoreLemmaPrune = dataCoreLemma.copy()
dataCoreLemmaPrune.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 63310 entries, 0 to 63309
Data columns (total 14 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   uuid                            63310 non-null  object 
 1   overall                         63310 non-null  float64
 2   overall_posneg                  63310 non-null  int64  
 3   reviewText_lemma                63310 non-null  object 
 4   reviewText_lemma_tb_pol         63310 non-null  float64
 5   reviewText_lemma_tb_subj        63310 non-null  float64
 6   reviewText_lemma_tb_tokens      63310 non-null  float64
 7   reviewText_lemma_tb_length      63310 non-null  float64
 8   reviewText_lemma_bert           63310 non-null  object 
 9   reviewText_lemma_flairSent      63310 non-null  float64
 10  reviewText_lemma_flairSent_norm 63310 non-null  float64
 11  reviewText_lemma_flairSent_posneg 63310 non-null  float64
 12  reviewText_lemma_tb_pol_norm    63310 non-null  float64
 13  reviewText_lemma_tb_pol_posneg  63310 non-null  float64
dtypes: float64(10), int64(1), object(3)
memory usage: 6.8+ MB
```

```
mvutils.examineColumnNumeric(dataCoreLemmaPrune,
                             'reviewText_lemma_tb_tokens',
                             binsize=1,
                             zoom=True,
                             minZoomLevel=0,
                             maxZoomLevel=20,
                             plotsize=5)
```




```
mvutils.examineColumnNumeric(dataCoreLemmaPrune,
                             'reviewText_lemma_tb_tokens',
                             binsize=100,
                             zoom=True,
                             minZoomLevel=1500,
                             maxZoomLevel=5000,
                             plotsize=5)
```

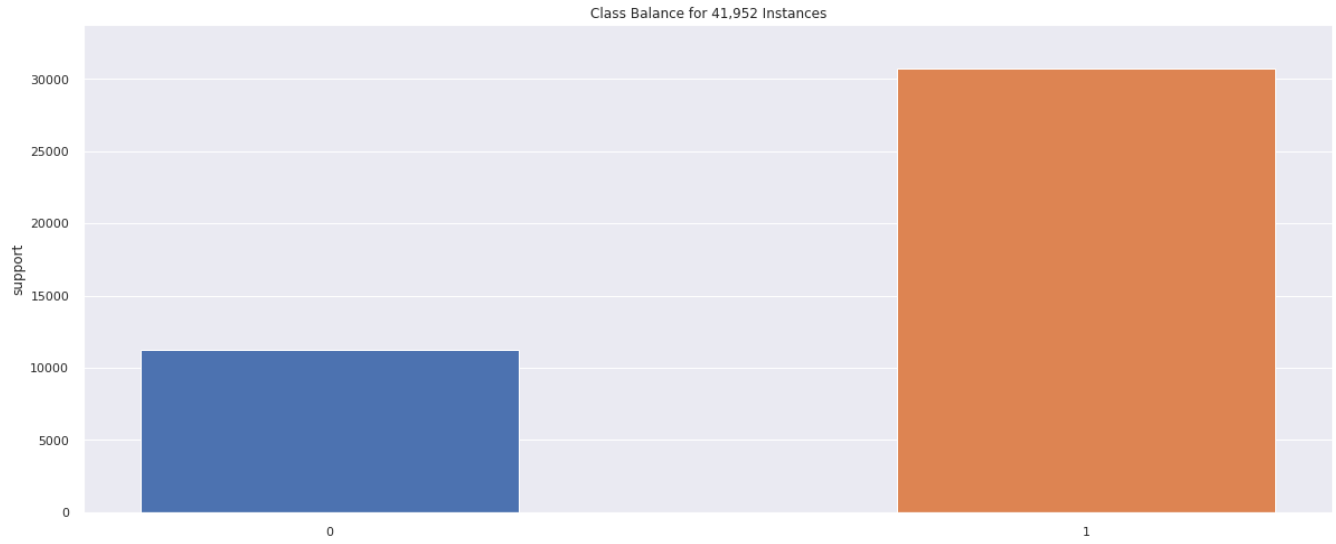

```
#import numpy as np
idx = np.where((dataCoreLemmaPrune['reviewText_lemma_tb_tokens']>=5) &
               (dataCoreLemmaPrune['reviewText_lemma_tb_tokens']<= 2100))
```


```
dataCoreLemmaPrune = dataCoreLemmaPrune.loc[idx].copy()
dataCoreLemmaPrune.reset_index(inplace=True, drop=True)
```

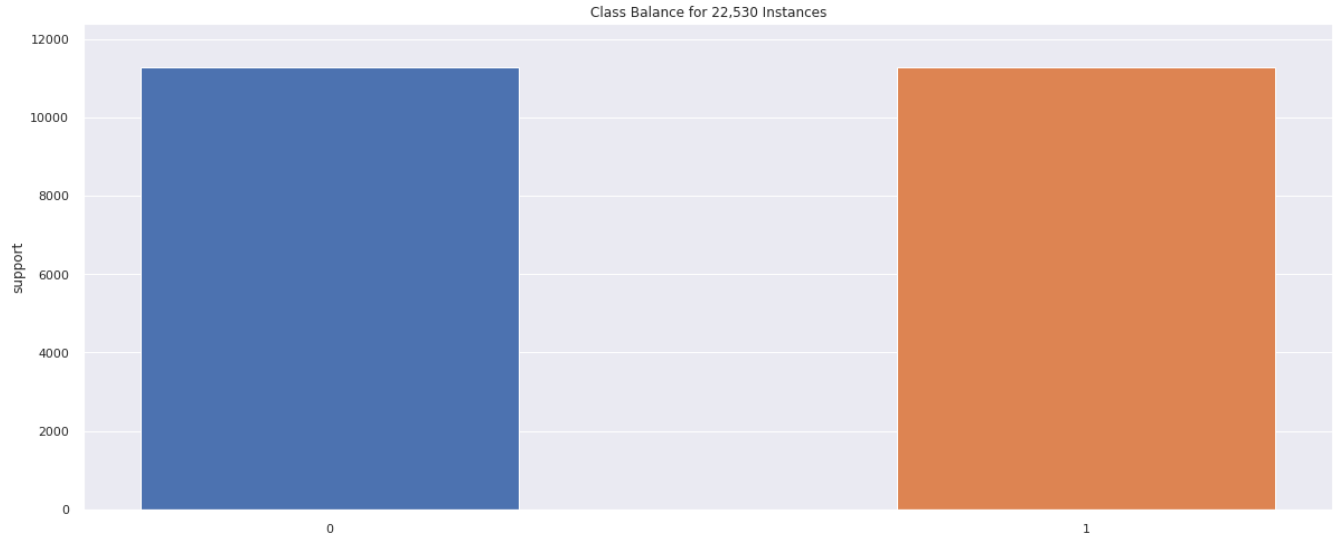
```
dataCoreLemmaPrune.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41952 entries, 0 to 41951
Data columns (total 14 columns):
 #   Column                            Non-Null Count  Dtype  
---  ------                            --------------  -----  
 0   uuid                              41952 non-null  object 
 1   overall                           41952 non-null  float64
 2   overall_posneg                    41952 non-null  int64  
 3   reviewText_lemma                  41952 non-null  object 
 4   reviewText_lemma_tb_pol           41952 non-null  float64
 5   reviewText_lemma_tb_subj          41952 non-null  float64
 6   reviewText_lemma_tb_tokens        41952 non-null  float64
 7   reviewText_lemma_tb_length        41952 non-null  float64
 8   reviewText_lemma_bert             41952 non-null  object 
 9   reviewText_lemma_flairSent        41952 non-null  float64
 10  reviewText_lemma_flairSent_norm   41952 non-null  float64
 11  reviewText_lemma_flairSent_posneg 41952 non-null  float64
 12  reviewText_lemma_tb_pol_norm      41952 non-null  float64
 13  reviewText_lemma_tb_pol_posneg    41952 non-null  float64
dtypes: float64(10), int64(1), object(3)
memory usage: 4.5+ MB
```

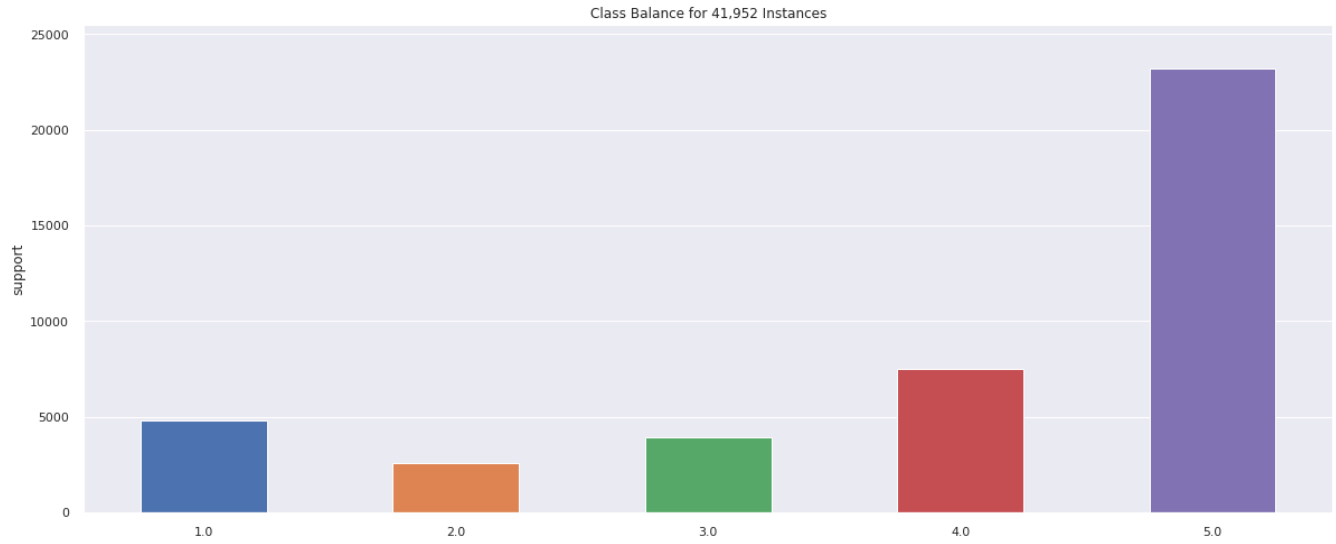
```
dataCoreLemmaPruneBal2 = mvutils.classBalanceUndersample(dataCoreLemmaPrune, 'overall_posneg'
```



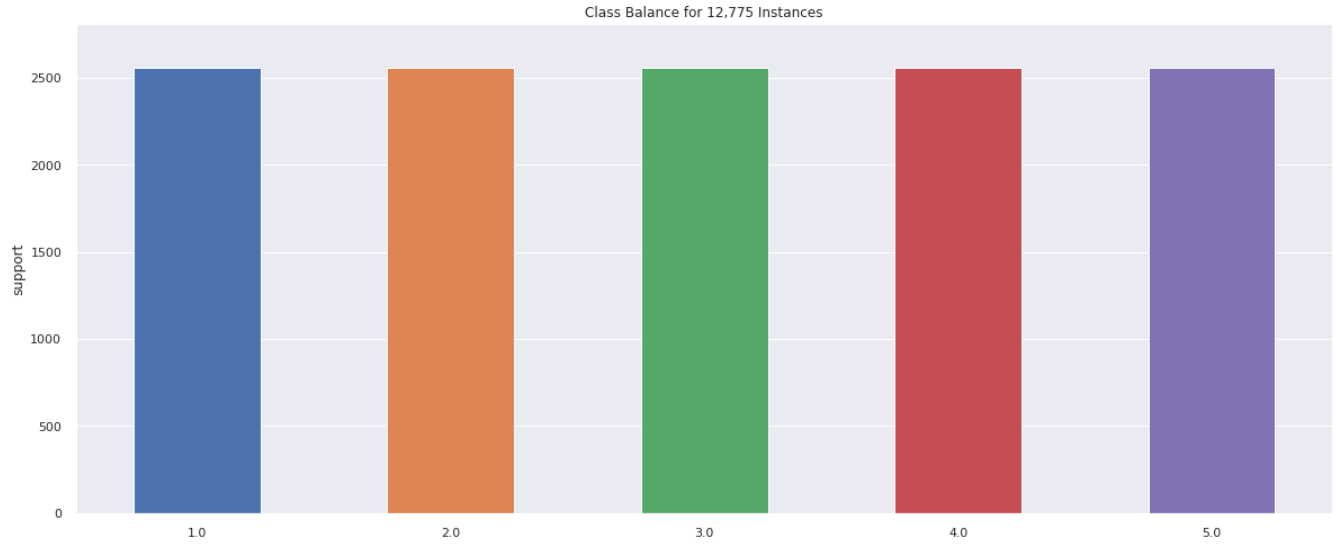

```
Undersampling data to match min class: 0 of size: 11265
```




```
dataCoreLemmaPruneBal5 = mvutils.classBalanceUndersample(dataCoreLemmaPrune, 'overall')
```




```
Undersampling data to match min class: 2.0 of size: 2555
```

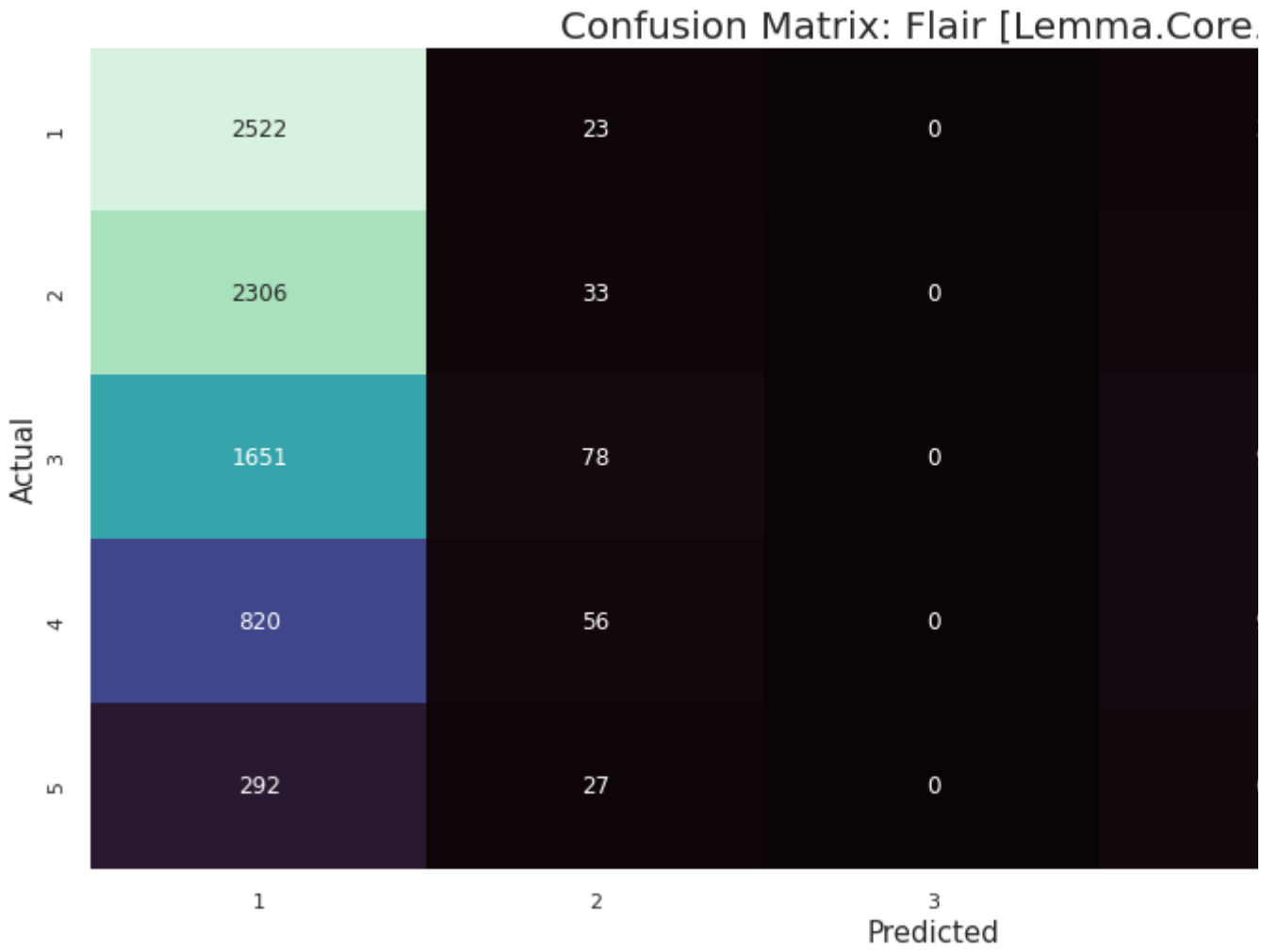


Report Scotty! [Lemma.Core]

Reporting [Lemma.Core.Flair.5 Star]

```
cwutils.showTestReport(df=dataCoreLemmaBal5,
                       colNameActual='overall',
                       colNamePredict='reviewText_lemma_flairSent_norm',
                       axisLabels=axis_labels5,
                       chartTitle='Flair [Lemma.Core.5]')
```

```
                  precision    recall  f1-score   support

             1.0       0.33      0.86      0.48      2939
             2.0       0.15      0.01      0.02      2939
             3.0       0.00      0.00      0.00      2939
             4.0       0.27      0.03      0.06      2939
             5.0       0.39      0.87      0.54      2939

        accuracy                           0.35     14695
       macro avg       0.23      0.35      0.22     14695
    weighted avg       0.23      0.35      0.22     14695
```

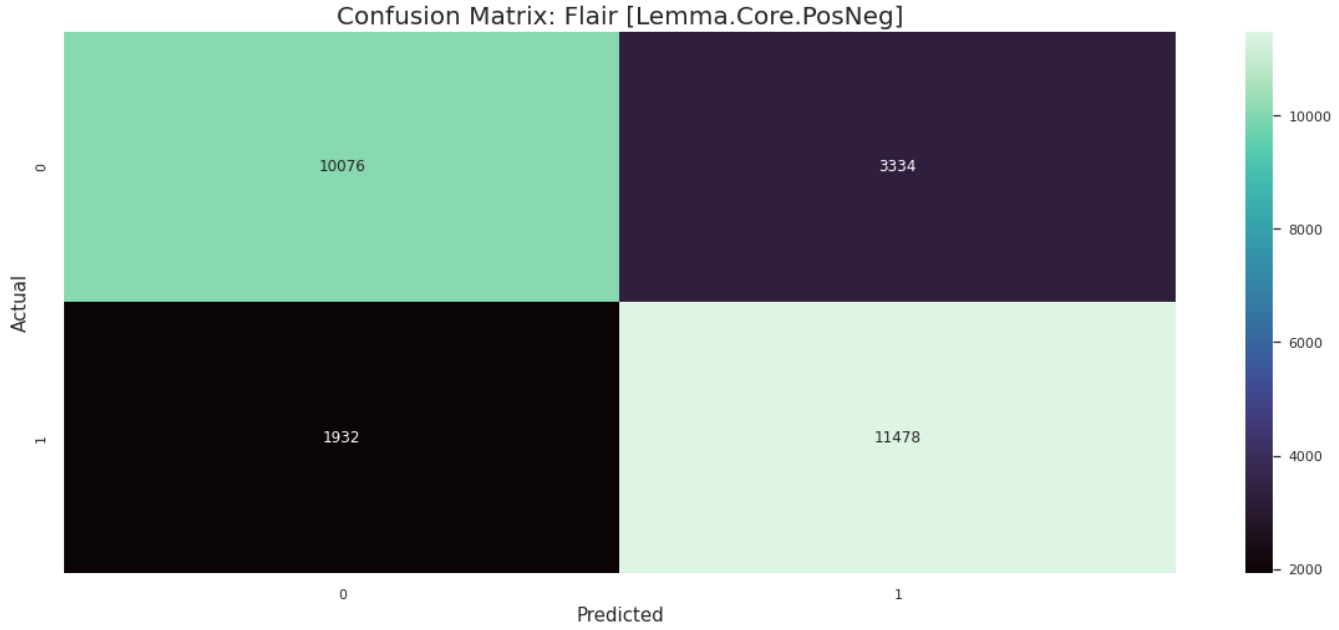


Reporting [Lemma.Core.Flair.Pos/Neg]

```
cwutils.showTestReport(df=dataCoreLemmaBal2,
                       colNameActual='overall_posneg',
                       colNamePredict='reviewText_lemma_flairSent_posneg',
```

```
                axisLabels=axis_labels2,
                chartTitle='Flair [Lemma.Core.PosNeg]')
```

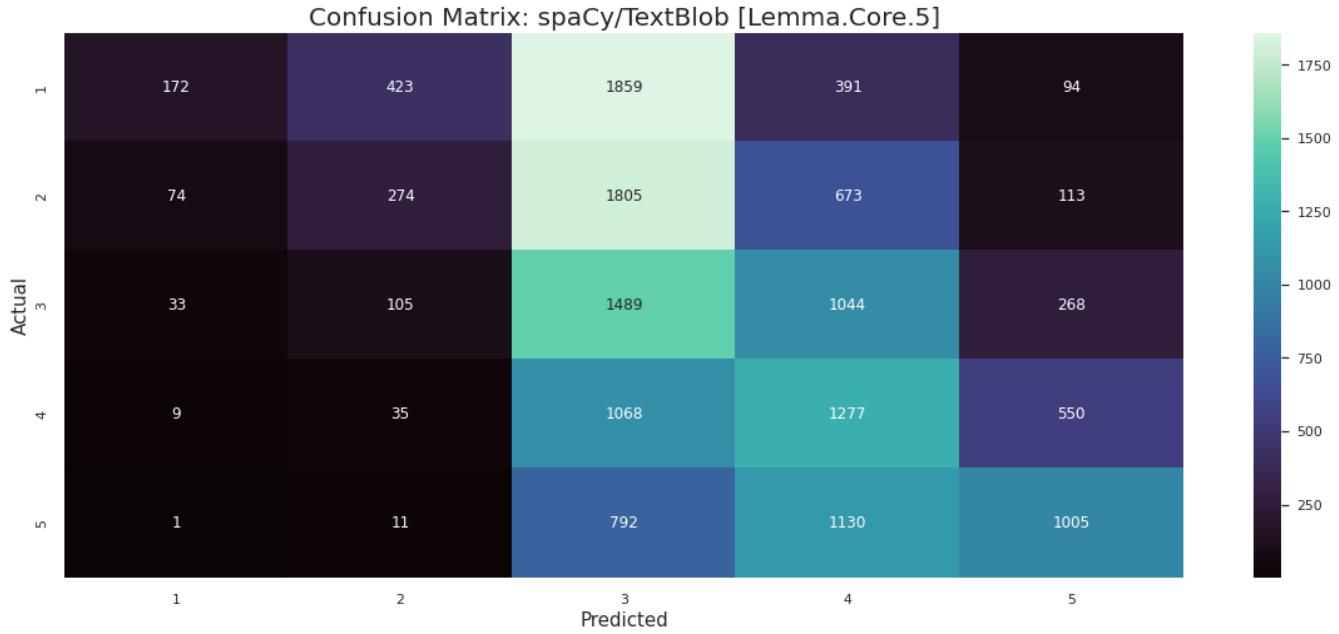
```
                precision    recall  f1-score   support

             0       0.84      0.75      0.79     13410
             1       0.77      0.86      0.81     13410

      accuracy                           0.80     26820
     macro avg       0.81      0.80      0.80     26820
  weighted avg       0.81      0.80      0.80     26820
```



Reporting [Lemma.Core.spaCyTextBlob.5 Star]

```
cwutils.showTestReport(df=dataCoreLemmaBal5,
                colNameActual='overall',
                colNamePredict='reviewText_lemma_tb_pol_norm',
                axisLabels=axis_labels5,
                chartTitle='spaCy/TextBlob [Lemma.Core.5]')
```

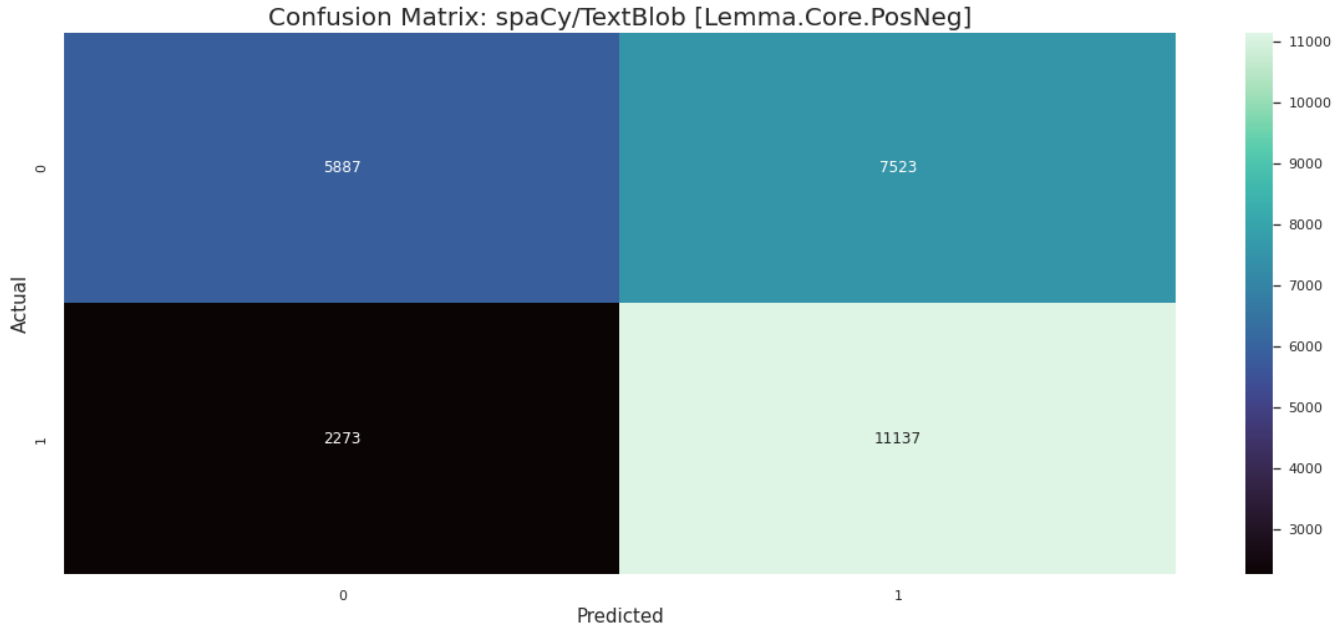
	precision	recall	f1-score	support
1.0	0.60	0.06	0.11	2939
2.0	0.32	0.09	0.14	2939
3.0	0.21	0.51	0.30	2939
4.0	0.28	0.43	0.34	2939
5.0	0.50	0.34	0.40	2939
accuracy			0.29	14695
macro avg	0.38	0.29	0.26	14695
weighted avg	0.38	0.29	0.26	14695



Reporting [Lemma.Core.SpacyTextBlob.Pos/Neg]

```
cwutils.showTestReport(df=dataCoreLemmaBal2,
                       colNameActual='overall_posneg',
                       colNamePredict='reviewText_lemma_tb_pol_posneg',
                       axisLabels=axis_labels2,
                       chartTitle='spaCy/TextBlob [Lemma.Core.PosNeg]')
```

```
                precision    recall  f1-score   support

             0       0.72      0.44      0.55     13410
             1       0.60      0.83      0.69     13410

      accuracy                           0.63     26820
     macro avg       0.66      0.63      0.62     26820
  weighted avg       0.66      0.63      0.62     26820
```



BERT - [Lemma.Core]

```
import importlib
importlib.reload(cwutils)
```

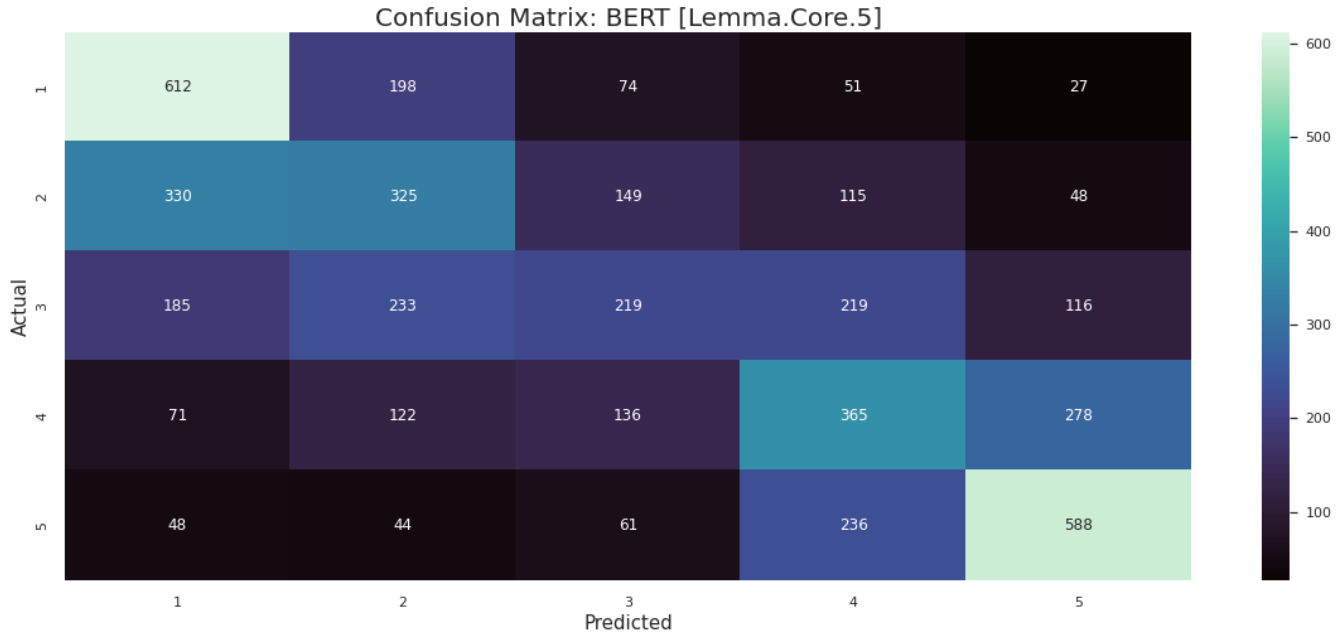
```
<module 'cw_df_metric_utils' from '/content/gdrive/MyDrive/Colab Notebooks/utility_file
```

BERT - [Lemma.Core.5]

```
modelBertCoreLemma5, dfmodelBertCoreLemma5 = cwutils.createBertModel(df=dataCoreLemmaBal5,
                bertColumn='reviewText_lemma_bert',
                uniqueColumn='uuid',
                targetColumn='overall'
                )

cwutils.showTestReport(df=dfmodelBertCoreLemma5,
```

```
                colNameActual='y_test',
                colNamePredict='y_pred',
                axisLabels=axis_labels5,
                chartTitle='BERT [Lemma.Core.5]')
```

```
              precision    recall  f1-score   support

         1.0       0.49      0.64      0.55       962
         2.0       0.35      0.34      0.34       967
         3.0       0.34      0.23      0.27       972
         4.0       0.37      0.38      0.37       972
         5.0       0.56      0.60      0.58       977

    accuracy                           0.43      4850
   macro avg       0.42      0.43      0.42      4850
weighted avg       0.42      0.43      0.42      4850
```

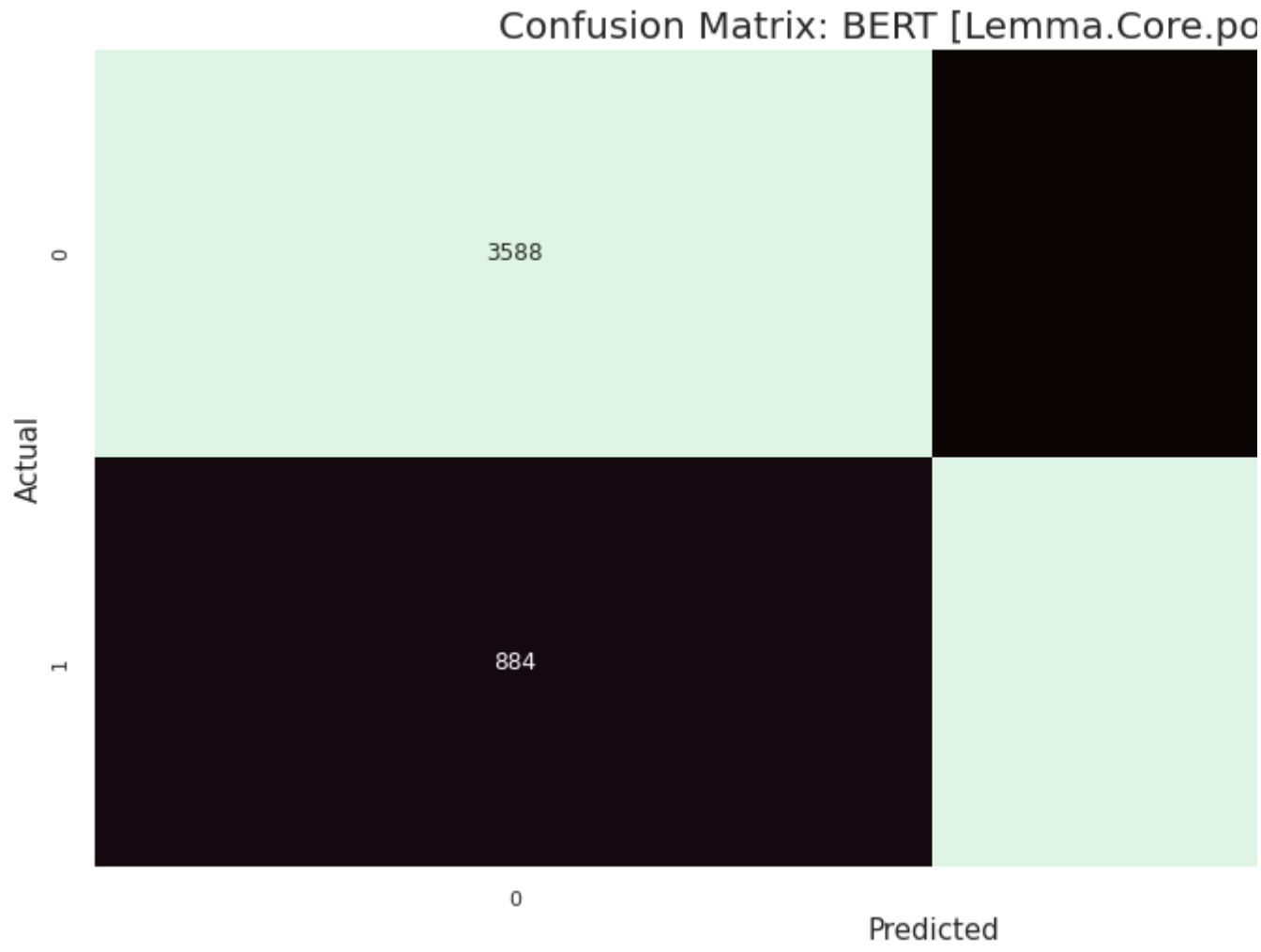


BERT - [Lemma.Core.posneg]

```
modelBertCoreLemma5, dfmodelBertCoreLemma5 = cwutils.createBertModel(df=dataCoreLemmaBal2,
                bertColumn='reviewText_lemma_bert',
                uniqueColumn='uuid',
                targetColumn='overall_posneg'
```

```
                )

cwutils.showTestReport(df=dfmodelBertCoreLemma5,
              colNameActual='y_test',
              colNamePredict='y_pred',
              axisLabels=axis_labels2,
              chartTitle='BERT [Lemma.Core.posneg]')
```

```
                precision    recall  f1-score   support

             0       0.80      0.82      0.81      4386
             1       0.82      0.80      0.81      4465

      accuracy                           0.81      8851
     macro avg       0.81      0.81      0.81      8851
  weighted avg       0.81      0.81      0.81      8851
```

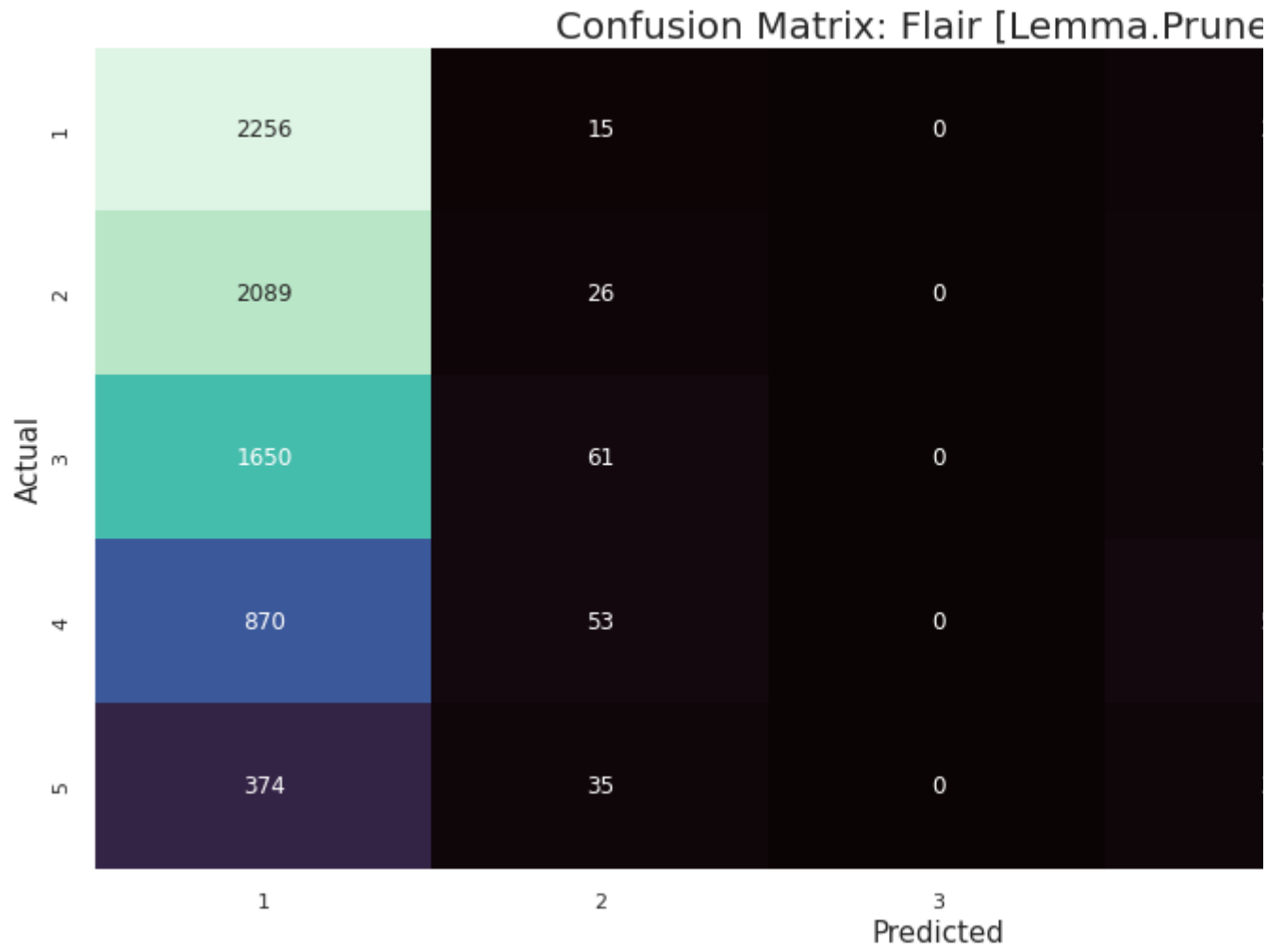


Report Scotty! [Lemma.Prune]

Reporting [Lemma.Prune.Flair.5 Star]

```
cwutils.showTestReport(df=dataCoreLemmaPruneBal5,
                       colNameActual='overall',
                       colNamePredict='reviewText_lemma_flairSent_norm',
                       axisLabels=axis_labels5,
                       chartTitle='Flair [Lemma.Prune.5]')
```

```
                  precision    recall  f1-score   support

             1.0       0.31      0.88      0.46      2555
             2.0       0.14      0.01      0.02      2555
             3.0       0.00      0.00      0.00      2555
             4.0       0.31      0.02      0.04      2555
             5.0       0.41      0.83      0.55      2555

        accuracy                           0.35     12775
       macro avg       0.23      0.35      0.21     12775
    weighted avg       0.23      0.35      0.21     12775
```

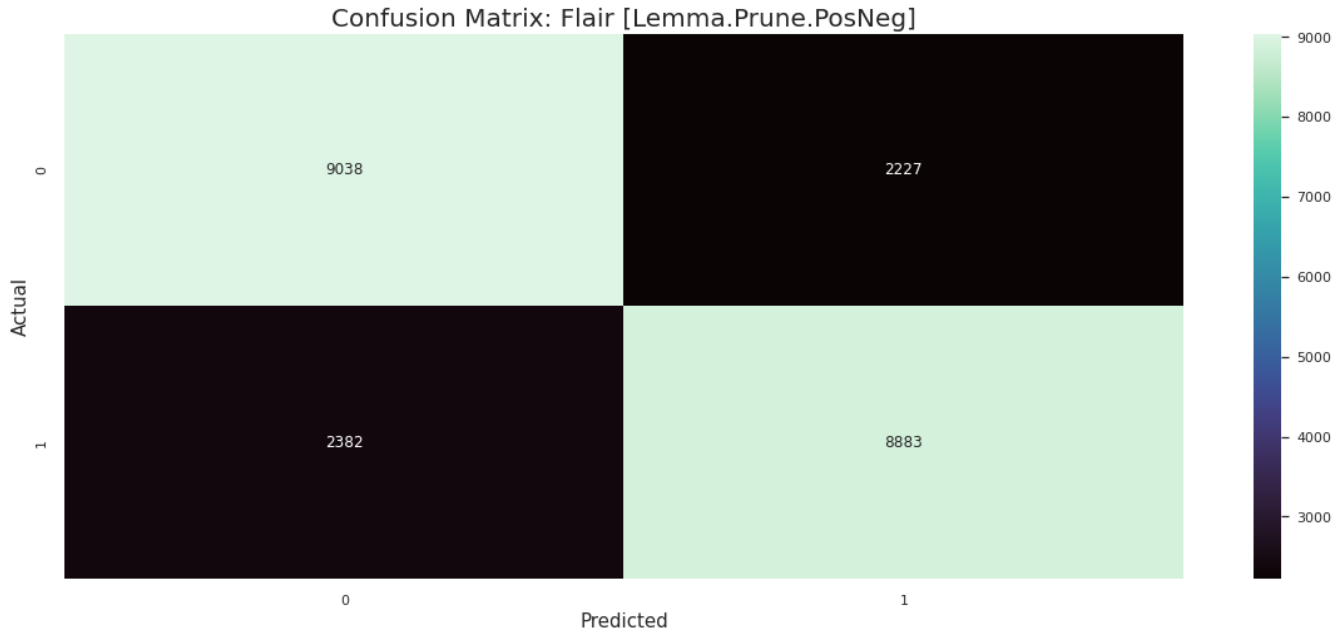


Reporting [Lemma.Prune.Flair.Pos/Neg]

```
cwutils.showTestReport(df=dataCoreLemmaPruneBal2,
                       colNameActual='overall_posneg',
                       colNamePredict='reviewText_lemma_flairSent_posneg',
```

```
                axisLabels=axis_labels2,
                chartTitle='Flair [Lemma.Prune.PosNeg]')
```

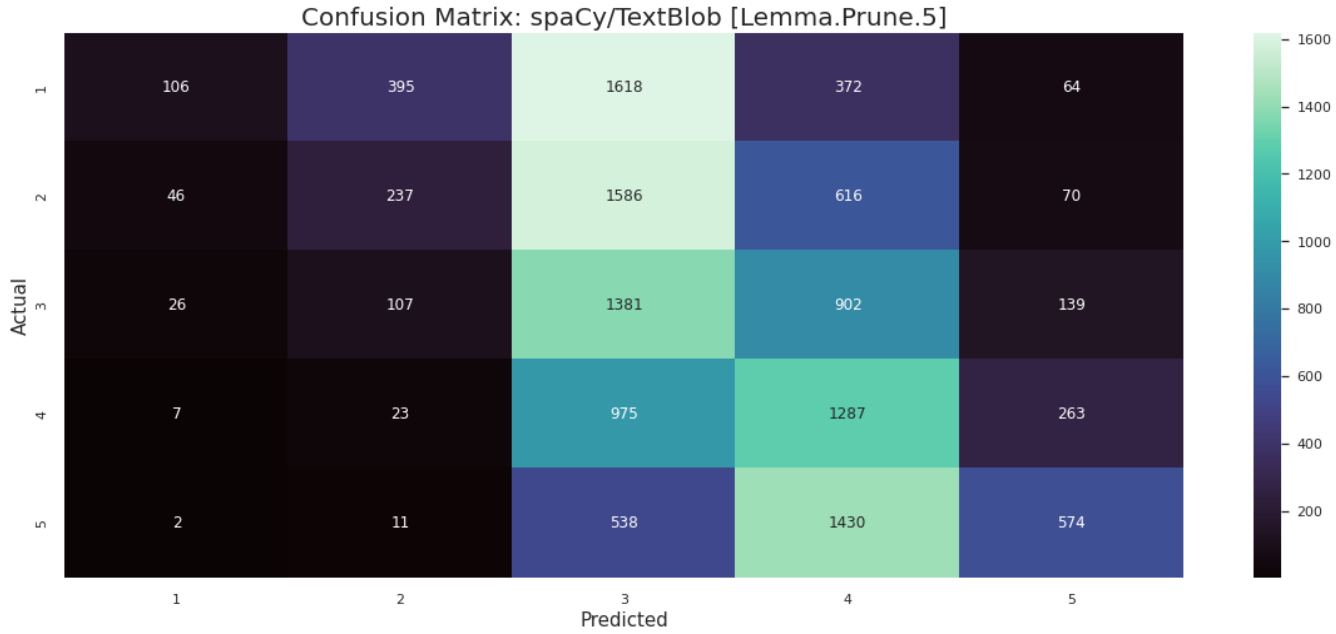
```
                precision    recall  f1-score   support

             0       0.79      0.80      0.80     11265
             1       0.80      0.79      0.79     11265

      accuracy                           0.80     22530
     macro avg       0.80      0.80      0.80     22530
  weighted avg       0.80      0.80      0.80     22530
```



Reporting [Lemma.Prune.spaCyTextBlob.5 Star]

```
cwutils.showTestReport(df=dataCoreLemmaPruneBal5,
                colNameActual='overall',
                colNamePredict='reviewText_lemma_tb_pol_norm',
                axisLabels=axis_labels5,
                chartTitle='spaCy/TextBlob [Lemma.Prune.5]')
```

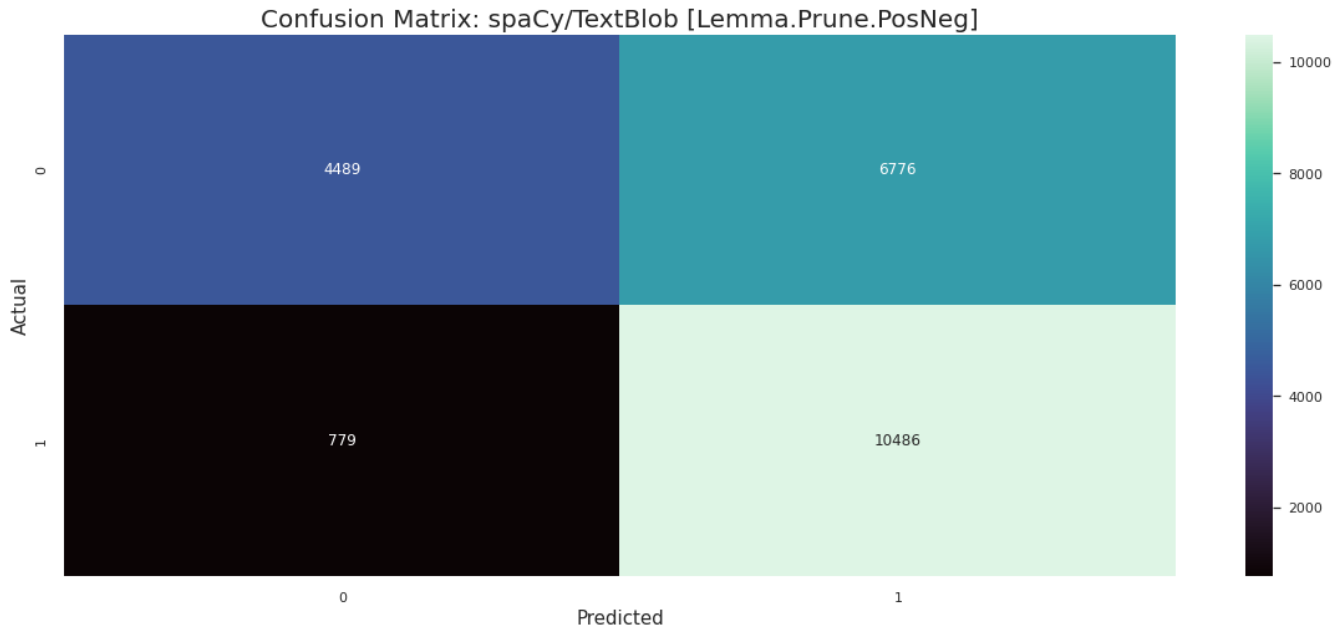
	precision	recall	f1-score	support
1.0	0.57	0.04	0.08	2555
2.0	0.31	0.09	0.14	2555
3.0	0.23	0.54	0.32	2555
4.0	0.28	0.50	0.36	2555
5.0	0.52	0.22	0.31	2555
accuracy			0.28	12775
macro avg	0.38	0.28	0.24	12775
weighted avg	0.38	0.28	0.24	12775



Reporting [Lemma.Prune.SpacyTextBlob.Pos/Neg]

```
cwutils.showTestReport(df=dataCoreLemmaPruneBal2,
                       colNameActual='overall_posneg',
                       colNamePredict='reviewText_lemma_tb_pol_posneg',
                       axisLabels=axis_labels2,
                       chartTitle='spaCy/TextBlob [Lemma.Prune.PosNeg]')
```

```
                precision    recall  f1-score   support

             0       0.85      0.40      0.54     11265
             1       0.61      0.93      0.74     11265

      accuracy                           0.66     22530
     macro avg       0.73      0.66      0.64     22530
  weighted avg       0.73      0.66      0.64     22530
```

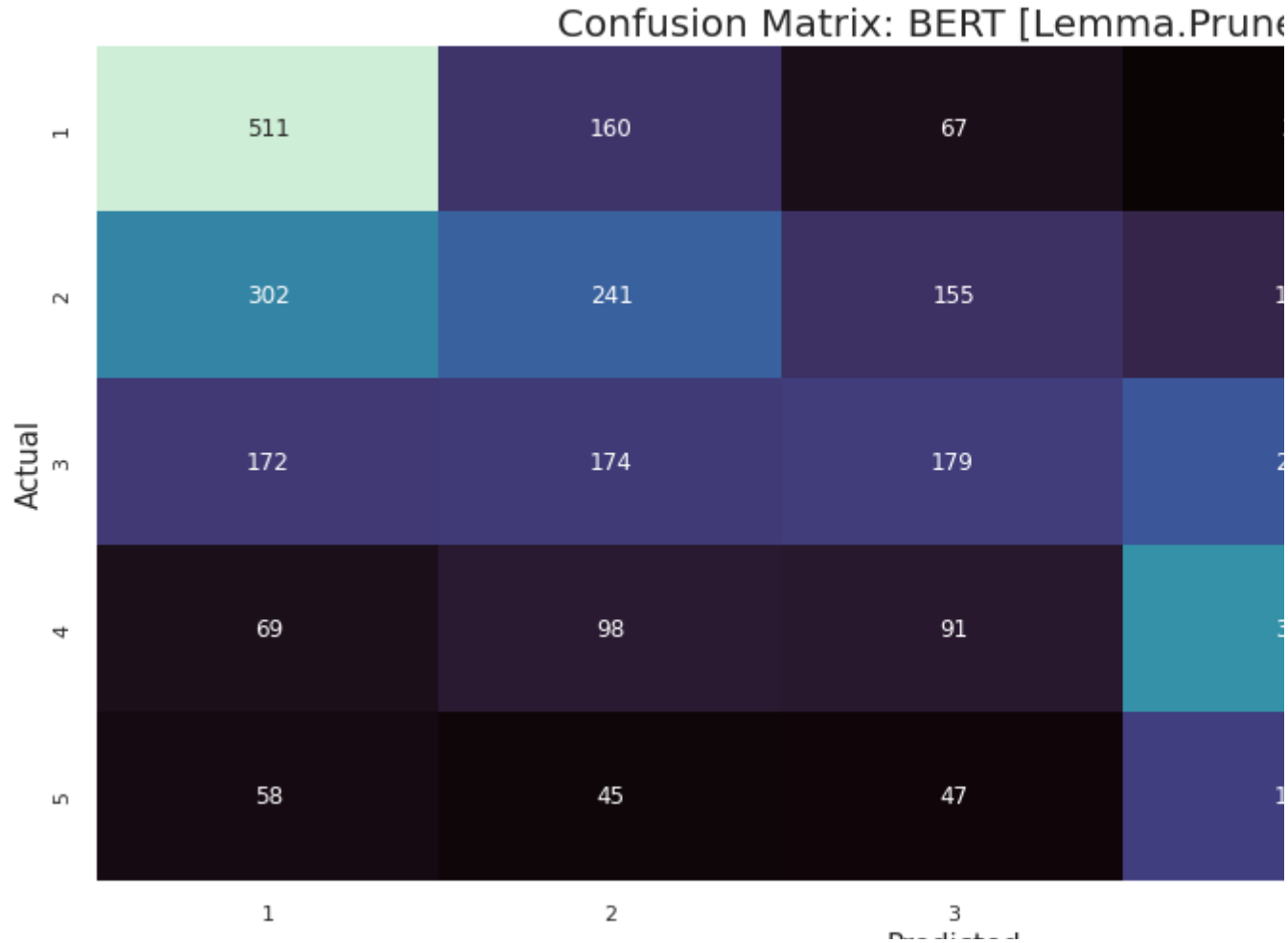


BERT - [Lemma.Prune]

BERT - [Lemma.Prune.5]

```
modelBertPruneLemma5, dfmodelBertPruneLemma5 = cwutils.createBertModel(df=dataCoreLemmaPruneE
                bertColumn='reviewText_lemma_bert',
                uniqueColumn='uuid',
                targetColumn='overall'
                )

cwutils.showTestReport(df=dfmodelBertPruneLemma5,
               colNameActual='y_test',
               colNamePredict='y_pred',
               axisLabels=axis_labels5,
               chartTitle='BERT [Lemma.Prune.5]')
```

	precision	recall	f1-score	support
1.0	0.46	0.62	0.53	824
2.0	0.34	0.28	0.30	869
3.0	0.33	0.21	0.26	854
4.0	0.36	0.40	0.38	809
5.0	0.55	0.61	0.58	860
accuracy			0.42	4216
macro avg	0.41	0.42	0.41	4216
weighted avg	0.41	0.42	0.41	4216

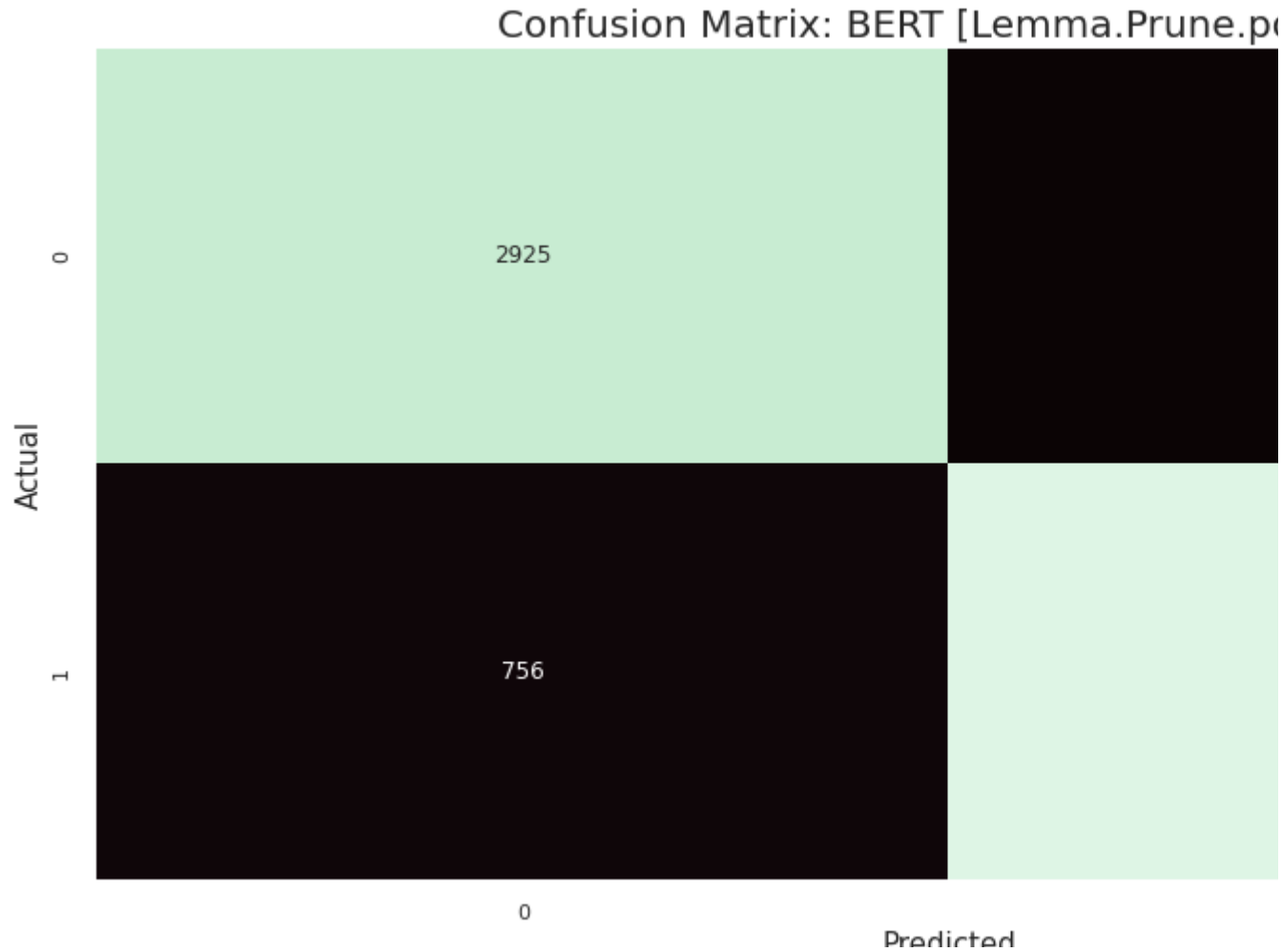


BERT - [Lemma.Prune.posneg]

```
modelBertPruneLemma2, dfmodelBertPruneLemma2 = cwutils.createBertModel(df=dataCoreLemmaPruneL
                bertColumn='reviewText_lemma_bert',
                uniqueColumn='uuid',
                targetColumn='overall_posneg'
                )

cwutils.showTestReport(df=dfmodelBertPruneLemma2,
               colNameActual='y_test',
               colNamePredict='y_pred',
```

```
                axisLabels=axis_labels2,
                chartTitle='BERT [Lemma.Prune.posneg]')
```

```
                  precision    recall  f1-score   support

               0       0.79      0.80      0.80      3653
               1       0.81      0.80      0.80      3782

        accuracy                           0.80      7435
       macro avg       0.80      0.80      0.80      7435
    weighted avg       0.80      0.80      0.80      7435
```



BERT Model - (adj) 5

```
tDfBert = data_bal.copy()
```

```
tDfBert.info()
```

```
dfBert = cwutils.getBertEncodeFrame(df=tDfBert,
                            bertColumn='reviewText_adjectives_bert',
                            uniqueColumn='uuid',
```

```
                                otherColumns=['overall']
                                )

#Get X Value from dataframe
dfX = dfBert.copy()
dfX.drop(['uuid', 'overall'], axis=1, inplace=True)

X = dfX.to_numpy()
Y = np.array(dfBert['overall'])

#X = dataset[:,0:8]
#Y = dataset[:,8]

# split data into train and test sets
seed = 7
test_size = 0.33
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size, random_state=s
# fit model no training data
modelBert5 = XGBClassifier()
modelBert5.fit(X_train, y_train)
# make predictions for test data
y_pred = modelBert5.predict(X_test)
#predictions = [round(value) for value in y_pred]
```

Report Scotty! Bert (adj) 5

```
# evaluate predictions
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
results = metrics.classification_report(y_test,
                                        y_pred,
                                        zero_division=0)
print(results)


cm = confusion_matrix(y_test, y_pred)
cwutils.plotConfusionMatrix(cm, axis_labels5, 'BERT (adjectives) 5')
```

BERT Model - (adj) 2

```
tDfBert = data_bal_posneg.copy()


tDfBert.info()
```

```
dfBert = cwutils.getBertEncodeFrame(df=tDfBert,
                            bertColumn='reviewText_adjectives_bert',
                            uniqueColumn='uuid',
                            otherColumns=['overall_posneg']
                            )


#Get X Value from dataframe
dfX = dfBert.copy()
dfX.drop(['uuid', 'overall_posneg'], axis=1, inplace=True)

X = dfX.to_numpy()
Y = np.array(dfBert['overall_posneg'])

#X = dataset[:,0:8]
#Y = dataset[:,8]

# split data into train and test sets
seed = 7
test_size = 0.33
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size, random_state=s
# fit model no training data
modelBert5 = XGBClassifier()
modelBert5.fit(X_train, y_train)
# make predictions for test data
y_pred = modelBert5.predict(X_test)
#predictions = [round(value) for value in y_pred]
```

Report Scotty! Bert (adj) 2

```
# evaluate predictions
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
results = metrics.classification_report(y_test,
                                        y_pred,
                                        zero_division=0)
print(results)


cm = confusion_matrix(y_test, y_pred)
cwutils.plotConfusionMatrix(cm, axis_labels2, 'BERT (adjectives) (positive-negative')
```

BERT Model - (verb) 5

```
tDfBert = data_bal.copy()


tDfBert.info()


dfBert = cwutils.getBertEncodeFrame(df=tDfBert,
                            bertColumn='reviewText_verbs_bert',
                            uniqueColumn='uuid',
                            otherColumns=['overall']
                            )


#Get X Value from dataframe
dfX = dfBert.copy()
dfX.drop(['uuid', 'overall'], axis=1, inplace=True)

X = dfX.to_numpy()
Y = np.array(dfBert['overall'])

#X = dataset[:,0:8]
#Y = dataset[:,8]

# split data into train and test sets
seed = 7
test_size = 0.33
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size, random_state=s
# fit model no training data
modelBert5 = XGBClassifier()
modelBert5.fit(X_train, y_train)
# make predictions for test data
y_pred = modelBert5.predict(X_test)
#predictions = [round(value) for value in y_pred]
```

Report Scotty! Bert (verb) 5

```
# evaluate predictions
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
results = metrics.classification_report(y_test,
                                        y_pred,
                                        zero_division=0)
print(results)


cm = confusion_matrix(y_test, y_pred)
cwutils.plotConfusionMatrix(cm, axis_labels5, 'BERT (verbs) 5')
```

BERT Model - (verb) 2

```
tDfBert = data_bal_posneg.copy()
```

```
tDfBert.info()
```

```
dfBert = cwutils.getBertEncodeFrame(df=tDfBert,
                            bertColumn='reviewText_verbs_bert',
                            uniqueColumn='uuid',
                            otherColumns=['overall_posneg']
                            )
```

```
#Get X Value from dataframe
dfX = dfBert.copy()
dfX.drop(['uuid', 'overall_posneg'], axis=1, inplace=True)

X = dfX.to_numpy()
Y = np.array(dfBert['overall_posneg'])

#X = dataset[:,0:8]
#Y = dataset[:,8]

# split data into train and test sets
seed = 7
test_size = 0.33
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size, random_state=s
# fit model no training data
modelBert5 = XGBClassifier()
modelBert5.fit(X_train, y_train)
# make predictions for test data
y_pred = modelBert5.predict(X_test)
#predictions = [round(value) for value in y_pred]
```

Report Scotty! Bert (verb) 2

```
# evaluate predictions
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
results = metrics.classification_report(y_test,
                                        y_pred,
```

```
                                        zero_division=0)
print(results)
cm = confusion_matrix(y_test, y_pred)
cwutils.plotConfusionMatrix(cm, axis_labels2, 'BERT (verbs) (positive-negative')
```

✓ 0s completed at 9:36 PM