```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input direct
ory

# import os
# for dirname, _, filenames in os.walk('/kaggle/input'):
#     for filename in filenames:
#         print(os.path.join(dirname, filename))

# Any results you write to the current directory are saved as output.
```

In [2]:

```
import numpy as np
import os
import pandas as pd
import cv2
from PIL import Image
import scipy

import tensorflow as tf
from tensorflow.keras.applications import *
from tensorflow.keras.optimizers import *
from tensorflow.keras.losses import *
from tensorflow.keras.layers import *
from tensorflow.keras.models import *
from tensorflow.keras.callbacks import *
from tensorflow.keras.preprocessing.image import *
from tensorflow.keras.utils import *
from sklearn.neural_network import MLPClassifier
# import pydot
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import *
from sklearn.model_selection import *
import tensorflow.keras.backend as K

from tqdm import tqdm, tqdm_notebook
from colorama import Fore
import json
import matplotlib.pyplot as plt
import seaborn as sns
from glob import glob
from skimage.io import *
%config Completer.use_jedi = False
import time
from sklearn.decomposition import PCA
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import lightgbm as lgb
from xgboost import XGBClassifier
from sklearn.ensemble import AdaBoostClassifier,RandomForestClassifier

from sklearn.metrics import confusion_matrix

import numpy as np
import pandas as pd
from pathlib import Path
import os.path
import matplotlib.pyplot as plt
from IPython.display import Image, display, Markdown
import matplotlib.cm as cm
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import tensorflow as tf
from time import perf_counter
import seaborn as sns
```

```
def printmd(string):
    # Print with Markdowns
    display(Markdown(string))

print("All modules have been imported")
```

```
All modules have been imported
```

In [3]:

```
image_dir = Path('../input/diabetic-retinopathy-224x224-2019-data/colored_images')

# Get filepaths and labels
filepaths = list(image_dir.glob(r'**/*.png'))
labels = list(map(lambda x: os.path.split(os.path.split(x)[0])[1], filepaths))
```

In [6]:

```
df = pd.read_csv("/kaggle/input/diabetic-retinopathy-224x224-2019-data/train.csv")
df
```

Out[6]:

	id_code	diagnosis
0	000c1434d8d7	2
1	001639a390f0	4
2	0024cdab0c1e	1
3	002c21358ce6	0
4	005b95c28852	0
...	...	...
3657	ffa47f6a7bf4	2
3658	ffc04fed30e6	0
3659	ffcf7b45f213	2
3660	ffd97f8cd5aa	0
3661	ffec9a18a3ce	2

3662 rows × 2 columns

In [11]:

```
filepaths
```

Out[11]:

```
[PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f481f76a6b75.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/abdb365cacbc.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/33ffddea8c6e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/059bc89df7f4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/278aa860dffd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d66b6f333dc7.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a8c950a99107.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cab3dfa7962d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/63363410389a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4da2961e62fe.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/22098b1fe461.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e9286ddf6ffe.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4c129470cec4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6b07971c3bf6.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b1197f2cc9b3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/90c982cc2d96.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/58184d6fd087.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2a8a9e957a6c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/76be29bb30b2.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0a61bddab956.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b9b99dad668d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/849a91e9ab28.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ea15a290eb96.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8f2996b8d855.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4ef7144e24ff.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/101b9ebfc720.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b640e3bdff75.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0d310aba6373.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7a3ea1779b13.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fdd18ccbbdc5.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5efa24b03d5e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/513b0a4651fa.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c1ebe785503a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1b329a127307.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4dd5d5ccddcf.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4f7755e74a9e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4d7d6928534a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/99ecdb41d5e7.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c56e65f74187.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1d674e2e32e0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fd62bd0db4f1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/545df1bbcd61.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bb11db08584a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/35aa7f5c2ec0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3044022c6969.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/194814669fee.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a75bab2463d4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b6a0e348a01e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/55eac26bd383.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6d292ca4c9ad.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d0d59ed675b5.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1c3a6b4449e9.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/eed4afc8ec83.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/db4ed1e07aa3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b07bc463b718.png'),
```

PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/382752f6694a.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6c6efb6b1358.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/944a233fbf8e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/49386d603494.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/53704c80f0d8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e96bd80a8a53.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/49c5e7f6b8d2.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c8fc0df22999.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fea14b3d44b0.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/63a03880939c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fe674c2f73f5.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/9eaf735cf01f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d801c0a66738.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/45e4b7eada54.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a2696f444ecb.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6ea07d19b4ce.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4c3c1ed09771.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3de8ad4151e1.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d85588ff2ebd.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/30941b65348b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/66cd9c28e636.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e2a233493b90.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4ecd1fdd1435.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4e82c3c8d31f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f762c272c522.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/655cafb4c932.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4e0656629d02.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/33b893e18eb3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0ad7f631dedb.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/200d947f75db.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a3ad6c2db6f1.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/95a4cc805c7b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8bbd7835e9aa.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3dbc90c7ee7d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2d7666b8884f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/086d41d17da8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2994f17f58a5.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bb783d8e496f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/89ed6a0dd53f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b94c58d063bf.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/01b3aed3ed4c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/582115961a3d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e580676516b0.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a8c54e2a4b79.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/79ce83c07588.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e26d8718ca58.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/19722bff5a09.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1e8a1fdee5b9.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8a01daa423f7.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/358d2224de73.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5347b4c8e9b3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d567a1a22d33.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cae51154e1ce.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fecf4c5ae84b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/248139c423c4.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/595446774178.png'),

```
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cfd1bd0fcbb4.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ababe19ed448.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e07045d7c5f7.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/38b9bb961847.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/107aea0d9289.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4393c5bc576a.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0fb1053285cf.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b576c5269ad1.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a28bfb772f50.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/61bbe8db6f3a.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/22a6da005395.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/17eff993386f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7427dedafccf.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5a091e8cd95c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/27e4c800a449.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cd45bfa07d41.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6377e23928f6.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1a03a7970337.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7ae69d22075a.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3c726de3ee90.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2d9d97a6e713.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a8582e346df0.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/29b52f64d2db.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ef8109305128.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/756b0d6488bb.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/78bcdffb8785.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/52ae917fcea4.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6b00cb764237.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/77a9538b8362.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5879285f9d8d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/59e5212f7139.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b56340f472d2.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c0e15e8e2b46.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f47a2a4a0411.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ead23cc922ed.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/531b39880c32.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ca63fe4f4b52.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1c0cf251b426.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ca6842bfcbc9.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/89d9c071a56f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/36041171f441.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5cab3ef4b31c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d4f32b9c07df.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/58ccba7eec9c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5671eb95512b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/365f8c01d994.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/68332fdcaa70.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7f60f2a083d3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/dd19428c3d29.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c96f743915b5.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c9485c38fdd5.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5548a7961a3e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/19e350c7c83c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0a3202889f4d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d85d052900b4.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7347f5133a6a.png'),
```

```
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4dd9d29eae5d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ca1036496659.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/821789e9053f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b8ebedd382de.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/587146a55885.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/64eb5a79dfdd.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/9a3109657ac1.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c739ff9580d3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ae8472f8d310.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c0f15fe3b4b7.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/73881f55a3ec.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cc12453ea915.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5b068765e846.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/13ab8db8c700.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/111898ab463d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1d11794057ff.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b7278b4f2448.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a32886cb31ab.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5090917a2676.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/64678182d8a8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/82ac8463fadd.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/613bacb35c05.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8714d17bb6da.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/977e1ca77653.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a5a2a7003d60.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bd269a1f0e4d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bacfb1029f6b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5b72ff04333d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c027e5482e8c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7ea756985353.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bb9a3d835a94.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cc9270f06b65.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d06ccd0cf4b8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8ff863f8874f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/12ce6a1a1f31.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/80a02014b418.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/35d6c4c50072.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5777ef74c9ec.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/76e589911303.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f55e1d2a19e4.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ad3fc5076852.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/40e9b5630438.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cf0575534cec.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c102db7634d8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3ee17aa12e46.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3f73c91b7e32.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/12e3f5f2cb17.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8a25a080f28f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e25ccfe38e44.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/05a5183c92d0.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a3d2a0c4cd17.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/578109578b46.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8c2f0f04e1ed.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7116128c65ab.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2f4e81787d9b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3461dc601cc2.png'),
```

```
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c40976189f22.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ca25745942b0.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cd5714db652d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/18b06f56ab27.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a7ec056502e7.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a01024054596.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/06b71823f9cd.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f0c13be90519.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1df3e03a8f5f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d5b4705ac2ee.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/36677b70b1ef.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a19ecd0a706e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4e6071b73120.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/99132193eaa0.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/47d1603a555b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/494fc9c745a3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7ccb267fd394.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/96a9706b8534.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/9ed666e982cd.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e38f3a65b02b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ca7140ecf389.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/eba3acc42197.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bebb3f167654.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fb6b8200b7f8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ec4649213ccf.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a77dbec966d4.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/172df1330a60.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0eb52045349f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d1cad012a254.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/94111ed3d276.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ad1f7445b1a8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b50b30aa6e6c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/24b943fe725e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0124dffecf29.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cb2f3c5d71a7.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/07a1c7073982.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c4a8f2fcf6e8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0369f3efe69b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4cae247d9909.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3b73a3a4a734.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7ef5ff774a48.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c1e6fa1ad314.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ca30a97e9d13.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/04ac765f91a1.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0684311afdfc.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/da1fb35f5df9.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8dc22e65c06f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8bc6716c2238.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a95858e052d6.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/28f98cfe3858.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4fa26d065ad3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/93be637084a2.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fca931da5c5e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/53f6c1c65c04.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a47432cd41e7.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6028a575dc27.png'),
```

```
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/65e51e18242b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8273fdb4405e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5712e2aa73a2.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7b20210d9120.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5b0e53f53ef3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/67f5d89da548.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/30cab14951ac.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/259d30f693b6.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/38e0e28d35d3.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b71428739d4e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/af133a85ea0c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a00b4cb250a7.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0024cdab0c1e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/677f087cd697.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c0968d41eb93.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4661006f3ba6.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6cdd0f985270.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bb45257258cc.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bf18ff30a8f6.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/09935d72892b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8fc09fecd22f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0dc031c94225.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8114d6a160df.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/03e25101e8e8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/523b3f0fc646.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/3c72f580d4ba.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/384631079d1e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/665ce639a331.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4aa07d720638.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0f495d87656a.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2f2e1949ad56.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/83e529e95b0e.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b22354b5f94b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8ef2eb8c51c4.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7005be54cab1.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7335a2d43ada.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/71c1a3cdbe47.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/98fbe56dcc2c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/cf603a9ef2d5.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d3de0d313d61.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2ecbc2e3f239.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/a443c4fd489c.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/495255c7492f.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6762b2b48ea5.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f72ef9ceeaa8.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e55188915f9d.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/33105f9b3a04.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/94372043d55b.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/9782c0489eca.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4029d70e9d8a.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6165081b9021.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d6e26fe51dce.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f5650eb52640.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/10f36b0239fb.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/0dce95217626.png'),
PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b4f41b5bf0ef.png'),
```

```
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5b804948e35f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/22325552a4e3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/31cb39681f6a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/4a213b405ee4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/25e9fd872182.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1bb0ddfe753a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/2a08ed6bbcbc.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/6298468d7d75.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/07929d32b5b3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/71f6a6e4620a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f7fec8935126.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/bc73ce76ec43.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/eeb231c3ef1f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/5633ced07d8e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/fe2df69676cf.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/94b1d8ad35ec.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/7bf981d9c7fe.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/15cc2aef772a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/d66ccb75ada1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/84a72e15b23c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/c7b622ec8104.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/8676427e4625.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/1116271db4ea.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/00cb6555d108.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/e9ff9352ccb3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/b17f0b81dab3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/f6f7dba7104d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/50840c36f0b4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/ee78ce914066.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/51131b48f9d4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/db690e2d02f8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/274f5029189b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/dbd062558b81.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/80d24897669f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Mild/92d9e9f08709.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b49b2fac2
514.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0243404e8
a00.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0083ee805
4ee.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/576e189d2
3d4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/15e96e848
b46.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8fd7ad26e
691.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/65c958379
680.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1a7e3356b
39c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/cb547e723
a16.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/03a7f4a57
86f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/64bad93fd
```

e3f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/cf1b9d26d38d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f0098e9d4aee.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/87774aafe068.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a3fcf42ff56d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/97da093947e8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4a3da369b227.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ba4d2c4b3039.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/5c7ab966a3ee.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4dc2211a1c31.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/fb696a8e055a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7a238a1d3cf3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2f284b6a1940.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/cc3d2e961768.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7c90ab025331.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/65e120143825.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d51b3fe0fa1b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e62490b7d0e9.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6c250a30593b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/59fee5bc3479.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f850cb51fdba.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/499c8df39222.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ba08cee68c71.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2bbcfdc477db.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f6f3ea0d2693.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/9fab29e69a6b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/de16416220de.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e4f12411fd85.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f69835dc7

```
c50.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3f752fccc
ec0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bc34ed91c
9bc.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c6a145742
708.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0ada12c0e
78f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6f4719c6b
b4b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d659d7fd5
ccf.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d0b132d2c
7ec.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/5dd2e26fc
244.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/84c663f39
632.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/710b05a96
e0f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/262ad7043
19c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/211518c46
162.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7bc4dd99e
ee5.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6cd606dc5
2e9.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6c3745a22
2da.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/374535e0a
db8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ed246ae1e
d08.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/25d069089
c5e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/02685f13c
efd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6630f8675
a97.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/22895c897
92f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3ccf96c1d
d6d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ff8a0b45c
789.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b70cb31b9
abb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/5e7db41b3
bee.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/55eb405ec
71e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d29011440
70c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2f7789c1e
```

046.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c62585bd6
8fb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4ccfa0b4e
96c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6d259b5b4
c76.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e4730ddde
408.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/247ac63e5
510.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/9c72ed6be
fa0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a0cd7bffd
aa0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bba38f229
4a3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/51a1d162e
223.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1e9224ccc
a95.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6e73acb2c
f60.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3ca637fdd
d56.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d1a24527a
15d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/16ce55574
8d8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/51af8c112
682.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3b185ac44
5d0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/254052cf3
e48.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8eeac97f0
2f0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e019b3e0f
33d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4abca30b6
76b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f549294e1
2e1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c8d2d32f7
f29.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d85ea1220
a03.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4bd941611
343.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/46cdc8b68
5bd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1fd5d860d
4d7.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/875a2fc5f
e23.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/381004009

6cb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8f318a978
844.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a32b5ce3d
48a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8596a24a1
4bd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/cd54d022e
37d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a5c9a8c72
6b2.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/37c4dfe03
aba.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/44ecf3f4e
fa5.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e9ab8413e
771.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/64fedbf97
473.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/9cc6b1f9b
cbd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d9ba04467
1e1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1efa5d443
707.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b37aae3c8
fe1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a81b06f50
612.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3796af4d9
87a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2fe06bedb
2c4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/27e2be850
a99.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/187f6ccda
87a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2df07eb57
79f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/19ef4d292
196.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/dad71ba27
a9b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e821c1b64
17a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/408ea9d5e
082.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/26463a5fb
949.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/aeed1f251
ceb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2017cd92c
63d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3c78bfca2
47b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/82bb8a019

35f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/905cc86bf
100.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7e0598cc8
8a0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4dd71fc7f
22b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a11bf2edd
470.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/5bda2ed09
e62.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6fe67482b
fae.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0bf37ca31
56a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3b232b394
e4f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8bed09514
c3b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7efc91af4
ae6.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ed3a0fc5b
546.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/eaa0dfbd5
024.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b96b51859
6b3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7d626a7ff
e76.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2fde69f20
585.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bfefa7344
e7d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8785b7123
8d8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/fce93caa4
758.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/001639a39
0f0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b87f9c597
48b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7cc4b7aab
e04.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/5b76117c4
bcb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0318598cf
d16.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/753b14c27
c83.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bd9904495
ccd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4a7dc013e
802.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6d7d26025
122.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bcd503c72

6ba.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b90bc89ce
8d8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f1dc26c4b
fa3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b3819a805
dca.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f58d37d48
e42.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/df84e7113
003.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/21d18b022
429.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/df4aec4a0
eaf.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e3ec668f6
fad.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ab1c20a94
f3f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1c4f3aa4d
f06.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c5e238aa1
8be.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/20d5fdd45
0ae.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4462fba1d
2a1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e7d2c2c3b
30f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/61f403fdb
434.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/cd972e563
9e0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/664b1f9a2
087.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/11242a671
22d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7c629b491
d1a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/62318d514
160.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/58b866484
a05.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/838c87c63
422.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e39b627cf
648.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6daef3e5c
a22.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6cfb7b44e
f6f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/222f3ee3a
1e8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7e4019ac7
f5a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1638404f3

```
85c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f2d2a0c92
034.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/cbd0870aa
933.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c3d12a23f
451.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b99afe713
7fb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a6d45de20
e4d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0981195eb
9fb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/60f15dd68
d30.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f025f33b2
c9b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1bf30c84b
bad.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0e82bcacc
475.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/66a0bf258
013.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ba735b286
d62.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e23add229
074.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e03764324
4b7.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1b32e1d77
5ea.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/857230f64
a2e.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/702de9dcd
e32.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/79d44db3d
a2d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/5d74f98d6
2be.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6f923b609
34b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8d8aca52c
07b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c01eae4b4
939.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e3ab63dc9
a60.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/eb1d37b71
fd1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8ae049175
db6.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8ac0c44bb
f24.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8c7c26c52
a6c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d271d3a2b
```

552.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/2628305cb
b29.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ebe0175e5
30c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e740af6ac
6ea.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/94ef1d145
97f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/785777558
f05.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/9a496b1e2
0f9.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3a1ecf5e2
839.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a2ddabee1
4e9.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/72d981886
48f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/61bbc11fe
503.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/08a387506
3c3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/df4913ca3
712.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/165cd2070
ebd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/29bc0e721
cfe.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/080ee76c9
58c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0e0fc1d98
10c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bdff5d8bd
df8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/e32dc722e
ca5.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3ac3fbfca
7d4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/eadc57064
154.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bfdee9be1
f1d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/281d7b7c7
676.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/6194e0fff
071.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c1896142a
20a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/cd93a472e
5cd.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ef2662512
1b3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7b211d8bd
249.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f0f89314e

860.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3b4a5fcbe
5e0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d48178e4a
49b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/bd5013540
a13.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b1b3e7d0a
5f3.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8db2ce991
101.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f5e6226bd
2e0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8bad12d70
368.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/69fff98cb
32a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c1799a6f5
c65.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3ee484193
6ef.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/ee1ec90b9
80f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/02dda30d3
acf.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/887c26fc0
e1f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/63b4d030b
016.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3a6e9730b
298.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/0ceb222f6
629.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/5b5b80a3e
dee.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/95e732e04
3a1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/612f2df37
a1d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/247e98aba
610.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d803598da
bda.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/080f66eed
fb9.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/78b3f819d
cc5.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d2c2f02bb
313.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/46d3316c4
857.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/c0e509786
f7f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/85cbb84ac
8e0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/1a90fad9f

fa2.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f03d3c4ce7fb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/9859e2a6cc24.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/10fca1abf338.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/fdd534271f3d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/7270367410a1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/aa6673241154.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f8cf7ed8ef00.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b77b8a1f09f1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/034cb07a550f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/07122e268a1d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/b55d2ddb3e75.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/873dcc0b468f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/80964d8e0863.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a76b69e443ce.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/789434d095d1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/8af6a4e5396f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/613028ede6a0.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/66375b3c64db.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/70d657f8f503.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/d10ef306996b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/fa59221cf464.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4a693dd3921a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/9e3510963315.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/312694ea8e6a.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/378963f9df22.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/518e880613de.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/a182b5b191de.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/4689b739d

240.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/61e301bd3c25.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/3206171db5be.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/21abd36095a1.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Proliferate_DR/f72adcac5638.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6dcde47060f9.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/be68322c7223.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a688f20f8895.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d4be0403e6ab.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/388f12e8df0b.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d83c3efade75.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1e4650743fa2.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0ac436400db4.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0d0a21fd354f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d15ca3469b87.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/28f93cad89c5.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/932181b93b2f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c68dfa021d62.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/7d1b40fdbd86.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/04d029cfb612.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c252da9b41d8.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/82910bba4753.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/08752092140d.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b83a6eca125f.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6666c4f18396.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6363b360aefb.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/812d5adafaf2.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ca891d37a43c.png'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/437cbec4a3f8.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6b869f37cdf3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e229aca862c7.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3a122851e526.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2e26762daed5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9a28d4e8aef0.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/530d78467615.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3ddb86eb530e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6c2555a9cae4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/da44f80b422b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/aef9016557ca.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/8a759f94613a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c4e8b1ec8893.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2ba0b0d9bda2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/31b5d6fb0256.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9faad91b6578.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b91ef82e723a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/71e4130bf5c8.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ea1d045f9fea.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/12e6e66c80a7.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a3706ce27869.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/fd48cf452e9d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/367c7049929c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b0d35981708b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9eaac43744f5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/51030843fde2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/62ecdc90dd42.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a7b7dc8788b9.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0eff8eacb2f7.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/8421107255ae.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1ade1e949383.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/42b08dca9b2f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9d9bfefa809c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/302bcdb635ff.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/290ecdba359f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/4489d421e5aa.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/8000a6b97a84.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/873fe0404d6e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1c578b72d7b3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/8d3d67661620.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9a78c6a7b1c2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/064af6592ba6.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/97a5ad7548b7.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e6a6acf7fca1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/44e0d56e9d42.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0a9ec1e99ce4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/f9aa35187bf3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d3d578fe433f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/913b1890ed1e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/246e4506824a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/650fbed3fdca.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9b7b6e4db1d5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e499434242cc.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/bda8c973b09d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/df8365d6ac33.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c3acf47700ea.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a125377fb985.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/db49cdf1ea64.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/fa3e544a7401.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6d454444f17c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/4f20f9a9a65b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/524f240e0c90.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/42a850acd2ac.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6889bc64ab09.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6b7cf869622a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3402124408ea.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/921433215353.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/46923eea9a4e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/59ee65760535.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d91273efb92a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/417f408ee8e0.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/4ee1ad981a6d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/547b37da9223.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d968a983d4d2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3dbfbc11e105.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/57760be09c03.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/eb32a815f78c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/308f7fce6f0d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ab50123abadb.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/17eb5d4ad740.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1541226c5d72.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1dd9adcbfff4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/032d7b0b4bf6.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e0863b353093.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/01c7808d901d.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a1b12fdce6c3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a49b0b4484ea.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/52dbec057cc8.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/48c72dec46e5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/441affbe99aa.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/43fb6eda9b97.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a56729de89e9.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3aa2b1ce6700.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/fba493e17448.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/af8aa32beee4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ca360bec5851.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/49d69c4c6290.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/bf7047dc683c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/52230bbef30e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2131aa3a1e6f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1943983492e5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/18af532e7e1e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/4a558a1cd243.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/f9ecf1795804.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a664d2055886.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b9127e38d9b9.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/460893cd86e3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1124ffcd76c2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d95959798b57.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2e79041ef722.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/4b422b48d0d4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/51405d042000.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/261c6bd63bff.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/59f3f70abddd.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/7ec1ffe8220b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9dab2e6ba44b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/eadfc8809ec8.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5265dc9acdf8.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/10ecc5292ab1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/abf0f56c6f12.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5c8482926a08.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d94e10f42861.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3f3de2a6b0f5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/76e6a9238570.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/43ddd0ab0cc4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3b9c1f42c2f2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/857002ed4e49.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/98f48850ebce.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a88f68b0b114.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/236f56771ec6.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/709784f7fcc2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ee2c2a5f7d0e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/7102f29e052e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ceb601fe8dba.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2f9b66784109.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/33e8e26a75d4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/86b3a7929bec.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e10190a9d52f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/cc1eebed9276.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/60e269e3e188.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b8fb9f55cd6d.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/188a9323be03.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/15cd5f52d300.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/00e4ddff966a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/bdb98063fe84.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0afdfe5f422c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/23d7ca170bdb.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/40140a925c43.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/91a88d3b0358.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/453a1e2754b2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/da8900ac7f29.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/61c2fbd16e38.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/48afe8c47454.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/07d8db76b301.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a7b0d0c51731.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1bea04b2bb2d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ed88faaa325a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a9c7b83caf81.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/62cc7ddb53b6.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9d74428188bb.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0ca0aee4d57e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b9b6ee2b9453.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1f63d44d9e3c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/8446826853d0.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/523d0c2cb4d6.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ff77e8e5b5f3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/207a580de0ea.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/10eefba568dd.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/18323d8f2470.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e7a372a1c3a4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1d55e689cf84.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d8cdb7d7283a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ba2624883599.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/349f3c0ac83e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/657859f893d9.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/8dfff47b06b7.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/85cc6d636898.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/bf1b7e21e774.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/8cb6b5b2f19c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b927a9238434.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c8a3eb9a5b52.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a0d04a19cf40.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/47e51065b819.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b376def52ccc.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/624fb7317106.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e724866f5084.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6cbc3dad809c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/441117562359.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/aca88f566228.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/00b74780d31d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ed2c06fcc573.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/86baef833ae0.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d6283ded6aea.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9da74370835a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/92f313287a29.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/57f933d3d7c7.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9f436886e056.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2bd4d4fbed5c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a87f53bc984a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/144b01e7b993.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/66bae1ba227f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ad570b850a4f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/354b8911d6ed.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c7e827fc7f41.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e2a47a74e6e1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1cb814ed6332.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/760b6f4c6d82.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1c4d87baaffc.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2cef97083e6f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/25002fe43f92.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/7bda86d95c5b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/7455e2b5fc57.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0519b934f6b1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/12025b34deb8.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/55034b1dbff2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d29096bd94aa.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/144a1a426137.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2b07790a2422.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1f0e223b8055.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2c1d5be654dd.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d1f1ea894da1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/203275daf46d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9f4132bd6ed6.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/239f2c348ea4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1438288bb2e1.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/89ee1fa16f90.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/75a7bc945b7d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b468ebf5cb11.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5b644a403e1f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e663c6627a95.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/55092c0071eb.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/97bf61736b86.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5327f88a1919.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ca7570c5925c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9cedf5c7016b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/cc839823755b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9b418ce42c13.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/23fca0693e2a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b498b84d383f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/f3a268d2726d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a45d77edf8d9.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b402daa0864c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d18b1d8ac4de.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2967e578939f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e1e490773462.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/f0c0f7b5e820.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/907aaff827e5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9df31421cdd2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d990a3f0cbdb.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5b301a6d1ac7.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d3be5346684b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/89fc080f7e83.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e7fc93ac5b6d.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d144144a2f3f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ff344e5c9341.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/975252e325e3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2af1bf226f51.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/f4d3777f2710.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/83b61051737f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6810410187a0.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a804cef3e51f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e7a7187066ad.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5a36cea278ae.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1c6d119c3d70.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/4189d4e631ec.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1968183f0e61.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/ef7a4ed8d5d1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/435d900fa7b2.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/855f0a5442b6.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/878a3a097436.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1dbdc32c17db.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9904939ab83d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/bfda2fd0533a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b2748ac28fc1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1594ca6c30d3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9ad92f1c1542.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c80f79579fed.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c568e5245ea5.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b0cc9f8d06e4.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/caec68f11c86.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2a3a1ed1c285.pn

g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a04fb36db784.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a8652b2de23f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e811f39a1243.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d10d315f123f.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/c8823cdaf7fa.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/3d663a6a50a3.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1632c4311fc9.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/7d1da90d3ca9.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/80feb1f7ca5e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/fcc55ae641ae.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/115e42dd6a81.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/a64273801bde.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/2735be026d44.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/b574d229ec4c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5ed6dc419e4d.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/e540d2e35d15.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/9c5dd3612f0c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/4f60129e9a5b.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1c5e6cdc7ee1.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5905a9b06a73.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0fd16b64697e.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/f1a761c68559.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/6531070bf03c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0fffa73e2402.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/1f4bf8e28b41.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5e18af29d812.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/be21d8b60e2a.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/d6228d951958.pn

```
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/0dbaa09a458c.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/06024377d573.pn
g'),
 PosixPath('../input/diabetic-retinopathy-224x224-2019-data/colored_images/Moderate/5e52c9fe676f.pn
g'),
 ...]
```

In [12]:

```
filepaths = pd.Series(filepaths, name='Filepath').astype(str)
labels = pd.Series(labels, name='Label')

# Concatenate filepaths and labels
image_df = pd.concat([filepaths, labels], axis=1)

# Shuffle the DataFrame and reset index
image_df = image_df.sample(frac=1).reset_index(drop = True)

# Show the result
image_df.head()
```

Out[12]:

	Filepath	Label
0	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate
1	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR
2	../input/diabetic-retinopathy-224x224-2019-dat...	Proliferate_DR
3	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate
4	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate

In [13]:

```
image_df
```

Out[13]:

	Filepath	Label
0	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate
1	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR
2	../input/diabetic-retinopathy-224x224-2019-dat...	Proliferate_DR
3	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate
4	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate
...	...	...
3657	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR
3658	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR
3659	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR
3660	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR
3661	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR

3662 rows × 2 columns

In [14]:

```
level = []
for i in image_df['Label']:
    if i=='No_DR':
        level.append(0)
    elif i=='Mild':
        level.append(1)
    elif i=='Moderate':
        level.append(2)
    elif i=='Severe':
        level.append(3)
    else:
        level.append(4)
```

In [15]:

```
level
```

Out[15]:

```
[2,
 0,
 4,
 2,
 2,
 2,
 0,
 2,
 2,
 0,
 4,
 0,
 2,
 3,
 0,
 0,
 0,
 0,
 4,
 0,
 2,
 2,
 0,
 2,
 0,
 0,
 1,
 0,
 2,
 0,
 2,
 4,
 0,
 4,
 4,
 0,
 0,
 4,
 2,
 0,
 0,
 2,
 0,
 1,
 0,
 1,
 2,
 0,
 0,
 0,
 0,
 2,
 0,
 0,
 1,
```

0,
0,
2,
0,
2,
2,
4,
0,
0,
2,
2,
0,
0,
0,
1,
0,
3,
0,
3,
0,
0,
0,
4,
0,
1,
3,
2,
2,
0,
0,
4,
2,
0,
2,
0,
0,
0,
2,
0,
0,
0,
2,
1,
0,
2,
4,
2,
0,
1,
0,
2,
0,
0,
1,
2,
4,

2,
0,
0,
3,
3,
0,
4,
0,
0,
4,
2,
1,
3,
2,
0,
4,
3,
2,
2,
1,
1,
0,
2,
0,
2,
1,
1,
0,
0,
2,
0,
3,
2,
3,
1,
4,
0,
0,
2,
4,
0,
4,
0,
0,
4,
0,
2,
0,
1,
0,
4,
4,
0,
2,
2,
2,

0,
0,
4,
0,
2,
0,
0,
0,
0,
1,
0,
0,
2,
0,
4,
0,
2,
3,
1,
1,
3,
1,
0,
2,
2,
3,
2,
0,
0,
1,
0,
2,
2,
2,
4,
0,
2,
2,
1,
4,
2,
4,
2,
0,
0,
0,
2,
1,
0,
0,
0,
0,
2,
0,
1,
2,

0,
2,
0,
2,
4,
0,
0,
1,
0,
0,
3,
0,
0,
0,
4,
2,
0,
0,
1,
0,
1,
2,
2,
0,
2,
0,
2,
0,
0,
0,
4,
0,
2,
0,
1,
0,
0,
0,
0,
0,
0,
2,
0,
0,
1,
1,
4,
3,
2,
2,
1,
0,
0,
2,
2,
0,
0,

4,
2,
0,
0,
2,
0,
0,
0,
0,
3,
0,
1,
0,
0,
0,
3,
0,
2,
0,
3,
4,
0,
0,
0,
1,
0,
0,
1,
0,
1,
0,
0,
4,
0,
1,
0,
0,
0,
3,
0,
0,
0,
1,
0,
0,
1,
0,
2,
0,
0,
3,
2,
0,
0,
2,
0,

0,
0,
2,
2,
2,
0,
2,
0,
0,
2,
0,
2,
0,
0,
0,
0,
2,
2,
0,
4,
4,
1,
4,
0,
0,
0,
0,
0,
2,
0,
0,
4,
0,
2,
2,
2,
2,
4,
2,
0,
0,
0,
0,
1,
2,
0,
2,
1,
0,
0,
2,
2,
0,
0,
2,
0,

0,
0,
3,
2,
2,
0,
0,
0,
2,
0,
0,
2,
2,
0,
0,
4,
3,
0,
2,
2,
0,
0,
2,
4,
1,
2,
2,
0,
0,
4,
1,
0,
0,
0,
2,
2,
2,
2,
2,
2,
0,
0,
0,
3,
0,
0,
4,
4,
4,
1,
0,
2,
0,
3,
2,
0,
2,

0,
0,
0,
0,
0,
0,
2,
2,
0,
0,
0,
0,
3,
2,
0,
0,
2,
3,
2,
0,
0,
1,
2,
0,
2,
2,
2,
0,
0,
0,
0,
0,
2,
2,
0,
1,
0,
4,
0,
4,
4,
0,
3,
4,
0,
2,
2,
0,
0,
1,
2,
4,
0,
3,
3,
2,

0,
2,
2,
3,
1,
2,
3,
2,
2,
0,
0,
1,
3,
0,
2,
2,
1,
0,
2,
0,
1,
0,
2,
0,
2,
3,
0,
2,
1,
1,
0,
0,
0,
2,
0,
4,
2,
1,
1,
0,
2,
2,
0,
4,
0,
0,
3,
0,
2,
2,
0,
1,
0,
0,
1,
0,

0,
0,
1,
0,
0,
1,
1,
2,
0,
0,
4,
0,
0,
2,
0,
2,
3,
0,
0,
2,
3,
0,
0,
1,
2,
0,
0,
0,
0,
0,
0,
2,
0,
0,
0,
0,
4,
2,
0,
0,
2,
0,
2,
0,
2,
1,
0,
0,
0,
0,
0,
0,
0,
3,
0,
2,
0,

0,
1,
0,
1,
2,
4,
0,
3,
2,
1,
2,
4,
2,
0,
0,
0,
2,
2,
0,
2,
0,
0,
1,
0,
2,
3,
2,
4,
0,
1,
1,
2,
0,
2,
0,
4,
2,
2,
4,
0,
0,
0,
2,
4,
2,
2,
0,
0,
0,
3,
0,
2,
0,
0,
2,
0,

0,
2,
1,
0,
4,
4,
0,
0,
0,
0,
2,
4,
2,
0,
0,
2,
2,
0,
2,
0,
0,
0,
2,
0,
0,
0,
0,
0,
0,
1,
0,
2,
3,
2,
0,
0,
0,
0,
1,
0,
0,
0,
0,
0,
0,
0,
3,
4,
0,
2,
0,
2,
0,
2,
1,
0,
0,

2,
0,
0,
1,
4,
0,
4,
2,
1,
2,
0,
0,
0,
0,
0,
0,
0,
2,
2,
2,
0,
0,
1,
0,
2,
0,
0,
2,
2,
1,
0,
0,
0,
0,
1,
3,
0,
0,
0,
2,
0,
0,
0,
0,
0,
0,
2,
0,
0,
2,
3,
2,
0,
1,
0,
2,
2,
0,

1,
2,
0,
1,
0,
0,
0,
1,
3,
2,
0,
0,
0,
0,
2,
2,
2,
0,
2,
1,
0,
0,
4,
0,
0,
0,
1,
4,
3,
0,
1,
4,
2,
0,
4,
0,
2,
0,
0,
0,
0,
0,
4,
0,
0,
0,
0,
2,
0,
2,
0,
2,
2,
0,
0,
2,

2,
4,
0,
0,
0,
0,
0,
2,
4,
0,
0,
0,
2,
0,
2,
0,
1,
0,
3,
0,
0,
1,
0,
2,
0,
3,
2,
0,
1,
4,
0,
0,
0,
3,
3,
2,
3,
2,
0,
0,
2,
0,
2,
0,
0,
3,
0,
2,
0,
0,
1,
0,
2,
2,
2,
0,

1,
0,
0,
0,
1,
2,
0,
0,
0,
2,
0,
0,
0,
4,
0,
3,
0,
1,
0,
4,
0,
2,
2,
0,
0,
1,
3,
1,
2,
4,
1,
0,
0,
0,
1,
0,
0,
0,
2,
0,
0,
0,
2,
0,
2,
2,
1,
2,
0,
4,
2,
2,
0,
0,
0,
0,

```
3,
0,
4,
0,
1,
0,
4,
0,
0,
0,
0,
2,
2,
0,
2,
0,
2,
0,
1,
2,
0,
0,
0,
0,
0,
0,
0,
0,
2,
0,
0,
0,
0,
2,
4,
3,
0,
0,
2,
2,
2,
4,
4,
0,
0,
4,
1,
4,
1,
2,
...]
```

In [16]:

```
image_df['Level'] = level
image_df.head()
```

Out[16]:

	Filepath	Label	Level
0	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate	2
1	../input/diabetic-retinopathy-224x224-2019-dat...	No_DR	0
2	../input/diabetic-retinopathy-224x224-2019-dat...	Proliferate_DR	4
3	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate	2
4	../input/diabetic-retinopathy-224x224-2019-dat...	Moderate	2

Different image labels taken into different folders with their label name, and a dataframe was created with file path, label name, level (this is just integer representation)..

In [17]:
```
X = []
for i in image_df['Filepath']:
    image = cv2.imread(i)
    X.append(image)

X = np.asarray(X)
y = image_df['Level']
Y = np.asarray(y)
```

In [23]:
```
image.shape   ## Creates the RGB component of image
```

Out[23]:
```
(224, 224, 3)
```

In [21]:
```
X.shape   ## append RGB component of image to itself
```

Out[21]:
```
(3662, 224, 224, 3)
```

In [24]:
```
y
```

Out[24]:
```
0       2
1       0
2       4
3       2
4       2
       ..
3657    0
3658    0
3659    0
3660    0
3661    0
Name: Level, Length: 3662, dtype: int64
```

The images were taken and found their RGB component, then that further appended into the each images creating an array of (3662, 224, 224, 3). The level column also converted into array format.

In [25]:
```
# Y=to_categorical(Y,5)
x_train, x_test1, y_train, y_test1 = train_test_split(X, Y, test_size=0.3, random_state=42)
x_val, x_test, y_val, y_test = train_test_split(x_test1, y_test1, test_size=0.3, random_state=42)
print(len(x_train),len(x_val),len(x_test))
```

```
2563 769 330
```

In [26]:
```
print(x_train.shape)
print(x_val.shape)
print(x_test.shape)
```

```
(2563, 224, 224, 3)
(769, 224, 224, 3)
(330, 224, 224, 3)
```

Train, Test and evaluation set prepared.

DNN Model

In [29]:
```
# Defining our DNN Model

dnn_model=Sequential()
dnn_model.add(Dense(32, input_dim=5, kernel_initializer = 'uniform', activation = 'relu'))
# dnn_model.add(BatchNormalization())
# dnn_model.add(Dropout(0.2))
dnn_model.add(Dense(64, kernel_initializer = 'HeUniform', activation = 'relu' ))
# dnn_model.add(BatchNormalization())
# dnn_model.add(Dropout(0.2))
dnn_model.add(Dense(128, kernel_initializer = 'uniform', activation = 'relu' ))
# dnn_model.add(BatchNormalization())
# dnn_model.add(Dropout(0.2))
dnn_model.add(Dense(256, kernel_initializer = 'uniform', activation = 'relu' ))
# dnn_model.add(BatchNormalization())
# dnn_model.add(Dropout(0.2))
dnn_model.add(Dense(128, kernel_initializer = 'uniform', activation = 'relu' ))
# dnn_model.add(BatchNormalization())
# dnn_model.add(Dropout(0.2))

dnn_model.add(Dense(5,activation='softmax'))
dnn_model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_12 (Dense)             (None, 32)                192
_________________________________________________________________
dense_13 (Dense)             (None, 64)                2112
_________________________________________________________________
dense_14 (Dense)             (None, 128)               8320
_________________________________________________________________
dense_15 (Dense)             (None, 256)               33024
_________________________________________________________________
dense_16 (Dense)             (None, 128)               32896
_________________________________________________________________
dense_17 (Dense)             (None, 5)                 645
=================================================================
Total params: 77,189
Trainable params: 77,189
Non-trainable params: 0
_________________________________________________________________
```

In [30]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
```

In [31]:

```
zipped_clf
```

Out[31]:

```
<zip at 0x7e42ac41f500>
```

In [32]:

```
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    y_pred_train = [1 if x>0.5 else 0 for x in y_pred_train]
    y_pred_val = [1 if x>0.5 else 0 for x in y_pred_val]
    y_pred_test = [1 if x>0.5 else 0 for x in y_pred_test]

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))

    print('----------------------- Test Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()
```

In [33]:

```
def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data-------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

ResNet50

In [35]:

```
base_model= ResNet50(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_
weights_tf_dim_ordering_tf_kernels_notop.h5
94773248/94765736 [==============================] - 3s 0us/step
```

In [36]:

```
from sklearn.pipeline import make_pipeline
from sklearn import pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
```

```
    print('------------------------ Test Set Metrics-----------------------')
    print("Accuracy score : {}%".format(test_accuracy))

    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [37]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

------------------------ Train Set Metrics-----------------------

Accuracy core : 60.01%
------------------------ Validation Set Metrics-----------------------

Accuracy score : 43.95%
------------------------ Test Set Metrics-----------------------
Accuracy score : 39.39%

Accuracy score : 39.39%
F1_score : 0.34
Kappa Score : -0.06
Recall score: 0.39
Precision score : 0.31
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
----------------------- Train Set Metrics-----------------------

Accuracy core : 48.22%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.410000000000004%
----------------------- Test Set Metrics-----------------------
Accuracy score : 50.3%

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------

----------------------- Train Set Metrics-----------------------

Accuracy core : 71.44%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.019999999999996%
----------------------- Test Set Metrics-----------------------
Accuracy score : 53.64%

Accuracy score : 53.64%
F1_score : 0.47
Kappa Score : 0.16
Recall score: 0.54
Precision score : 0.47
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data---------------------------
----

----------------------- Train Set Metrics-----------------------

Accuracy core : 51.23%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.54%
----------------------- Test Set Metrics-----------------------
Accuracy score : 49.7%

Accuracy score : 49.7%
F1_score : 0.43
Kappa Score : 0.09
Recall score: 0.5
Precision score : 0.38
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
```

```
[16:40:24] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


------------------------ Train Set Metrics-----------------------

Accuracy core : 97.54%
------------------------ Validation Set Metrics-----------------------

Accuracy score : 49.41%
------------------------ Test Set Metrics-----------------------
Accuracy score : 50.61%

Accuracy score : 50.61%
F1_score : 0.47
Kappa Score : 0.16
Recall score: 0.51
Precision score : 0.45
--------------------------------------------------------------------------------
```

In [38]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 6ms/step - loss: 1.4441 - accuracy: 0.4611 - val_loss:
1.2739 - val_accuracy: 0.5241
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2840 - accuracy: 0.4943 - val_loss:
1.2685 - val_accuracy: 0.5241
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3229 - accuracy: 0.4724 - val_loss:
1.2823 - val_accuracy: 0.5241
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3057 - accuracy: 0.4717 - val_loss:
1.2747 - val_accuracy: 0.5241
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2678 - accuracy: 0.5024 - val_loss:
1.2695 - val_accuracy: 0.5241
Epoch 6/10
81/81 [==============================] - 0s 4ms/step - loss: 1.2830 - accuracy: 0.4879 - val_loss:
1.2646 - val_accuracy: 0.5241
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2749 - accuracy: 0.4940 - val_loss:
1.2609 - val_accuracy: 0.5241
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2910 - accuracy: 0.4946 - val_loss:
1.2682 - val_accuracy: 0.5241
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2729 - accuracy: 0.5027 - val_loss:
1.2555 - val_accuracy: 0.5241
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3026 - accuracy: 0.4821 - val_loss:
1.2510 - val_accuracy: 0.5241
81/81 [==============================] - 0s 2ms/step - loss: 1.2814 - accuracy: 0.4822
Train_accuracy is:0.48224735260009766
25/25 [==============================] - 0s 2ms/step - loss: 1.2510 - accuracy: 0.5241
Validation_accuracy is := 0.5240572094917297
11/11 [==============================] - 0s 2ms/step - loss: 1.2436 - accuracy: 0.5030
test_accuracy is : = 0.5030303001403809
```

In [39]:

```
print("Performance Report:")
y_pred10=dnn_model.predict_classes(test_features)
y_test10=[np.argmax(x) for x in test_y]
y_pred_prb10=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test10, y_pred10),4))
print('Precision score is :', np.round(metrics.precision_score(y_test10, y_pred10, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test10,y_pred10, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test10, y_pred10, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test10, y_pred10),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test10, y_pred10,target_names=tar
get))
```

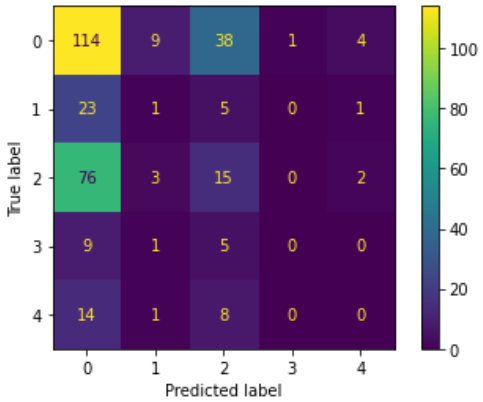
```
Performance Report:
Accuracy score is : 0.503
Precision score is : 0.253
Recall score is : 0.503
F1 Score is : 0.3367
Cohen Kappa Score: 0.0
                Classification Report:
               precision    recall  f1-score   support

           0       0.50      1.00      0.67       166
           1       0.00      0.00      0.00        30
           2       0.00      0.00      0.00        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.50       330
   macro avg       0.10      0.20      0.13       330
weighted avg       0.25      0.50      0.34       330



/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [40]:

```python
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[40]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4258284450>
```

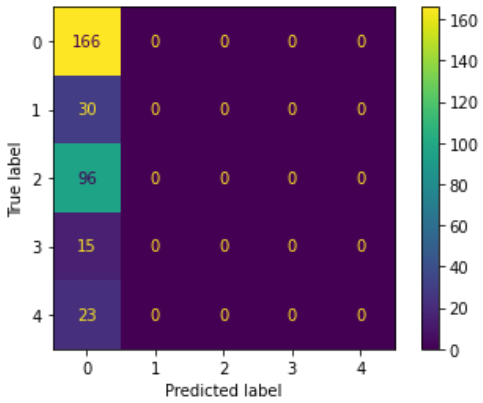


In [42]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[42]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e425803da90>
```

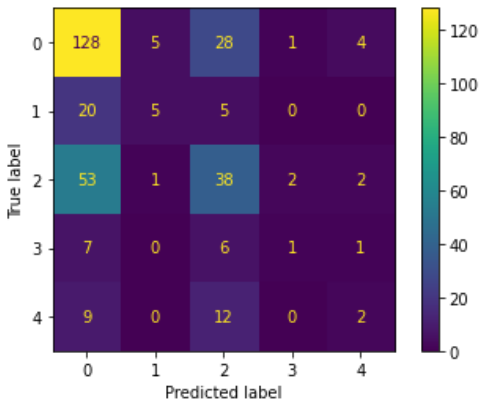


In [43]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[43]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42346cc0d0>
```

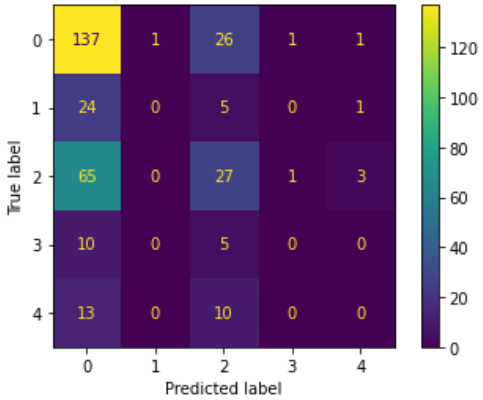


In [44]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[44]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42344deb10>
```



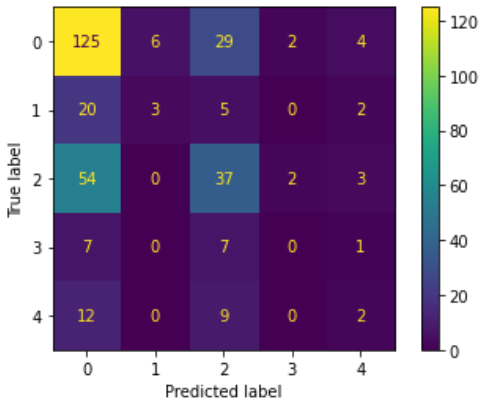
In [45]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:41:02] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[45]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42346f9310>
```



VGG-16

In [46]:

```
base_model= VGG16(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weig
hts_tf_dim_ordering_tf_kernels_notop.h5
58892288/58889256 [==============================] - 2s 0us/step
```

In [47]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [48]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

----------------------- Train Set Metrics-----------------------

Accuracy core : 60.629999999999995%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 47.33%
----------------------- Test Set Metrics-----------------------

Accuracy score : 49.09%
F1_score : 0.44
Kappa Score : 0.12
Recall score: 0.49
Precision score : 0.41
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 48.22%
------------------------ Validation Set Metrics------------------------

Accuracy score : 52.410000000000004%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 70.23%
------------------------ Validation Set Metrics------------------------

Accuracy score : 53.449999999999996%
------------------------ Test Set Metrics------------------------

Accuracy score : 56.36%
F1_score : 0.5
Kappa Score : 0.24
Recall score: 0.56
Precision score : 0.45
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data----------------------------
----
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


----------------------- Train Set Metrics-----------------------

Accuracy core : 53.300000000000004%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 50.33%
----------------------- Test Set Metrics-----------------------

Accuracy score : 54.55%
F1_score : 0.49
Kappa Score : 0.23
Recall score: 0.55
Precision score : 0.45
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:41:31] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

----------------------- Train Set Metrics-----------------------

Accuracy core : 98.48%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 53.190000000000005%
----------------------- Test Set Metrics-----------------------

Accuracy score : 55.45%
F1_score : 0.52
Kappa Score : 0.24
Recall score: 0.55
Precision score : 0.51
--------------------------------------------------------------------------------
```

In [49]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='sgd',loss='categorical_crossentropy', metrics=['accuracy'],)
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 5ms/step - loss: 1.3992 - accuracy: 0.4892 - val_loss:
1.2882 - val_accuracy: 0.5241
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2922 - accuracy: 0.4933 - val_loss:
1.2623 - val_accuracy: 0.5241
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2735 - accuracy: 0.4915 - val_loss:
1.2568 - val_accuracy: 0.5241
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2860 - accuracy: 0.4894 - val_loss:
1.2471 - val_accuracy: 0.5241
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2965 - accuracy: 0.4700 - val_loss:
1.2445 - val_accuracy: 0.5241
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2922 - accuracy: 0.4800 - val_loss:
1.2378 - val_accuracy: 0.5241
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2802 - accuracy: 0.4829 - val_loss:
1.2375 - val_accuracy: 0.5241
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2667 - accuracy: 0.4926 - val_loss:
1.2364 - val_accuracy: 0.5241
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2743 - accuracy: 0.4737 - val_loss:
1.2362 - val_accuracy: 0.5241
Epoch 10/10
81/81 [==============================] - 0s 5ms/step - loss: 1.2723 - accuracy: 0.4863 - val_loss:
1.2319 - val_accuracy: 0.5241
81/81 [==============================] - 0s 2ms/step - loss: 1.2741 - accuracy: 0.4822
Train_accuracy is:0.48224735260009766
25/25 [==============================] - 0s 3ms/step - loss: 1.2319 - accuracy: 0.5241
Validation_accuracy is := 0.5240572094917297
11/11 [==============================] - 0s 3ms/step - loss: 1.2218 - accuracy: 0.5030
test_accuracy is : = 0.5030303001403809
```

In [50]:

```
print("Performance Report:")
y_pred2=dnn_model.predict_classes(test_features)
y_test2=[np.argmax(x) for x in test_y]
y_pred_prb2=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test2, y_pred2),4))
print('Precision score is :', np.round(metrics.precision_score(y_test2, y_pred2, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test2,y_pred2, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test2, y_pred2, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test2, y_pred2),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test2, y_pred2,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.503
Precision score is : 0.253
Recall score is : 0.503
F1 Score is : 0.3367
Cohen Kappa Score: 0.0
                Classification Report:
               precision    recall  f1-score   support

           0       0.50      1.00      0.67       166
           1       0.00      0.00      0.00        30
           2       0.00      0.00      0.00        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.50       330
   macro avg       0.10      0.20      0.13       330
weighted avg       0.25      0.50      0.34       330
```

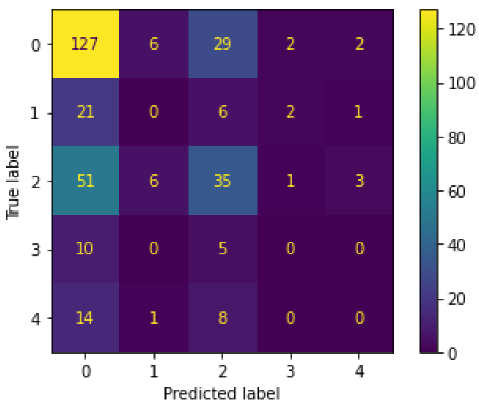
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [51]:

```python
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[51]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4230447390>
```

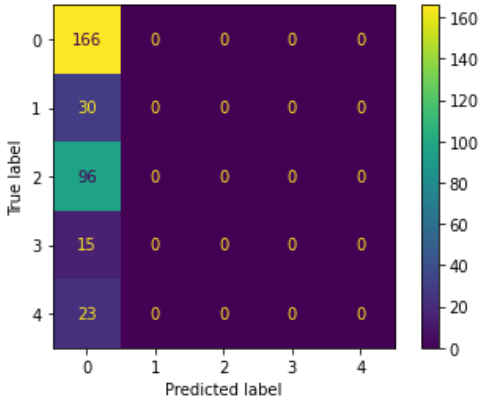


In [52]:

```python
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[52]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42303c63d0>
```

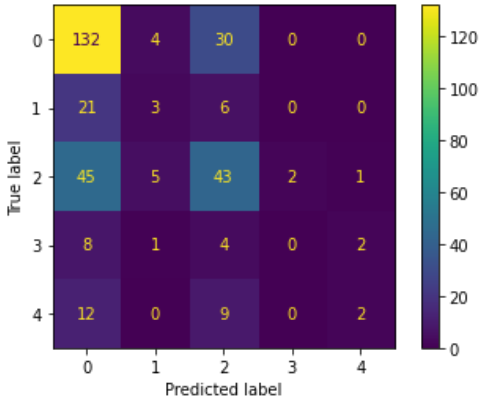


In [53]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[53]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42346ed210>
```

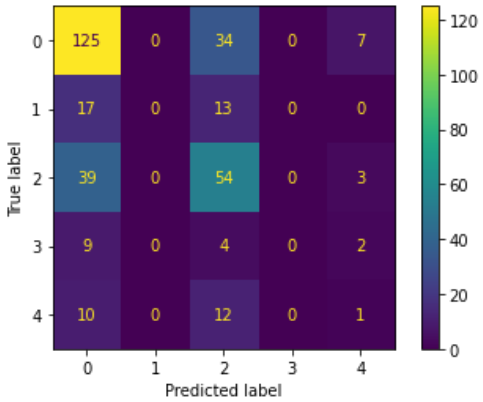


In [54]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[54]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4258061e90>
```



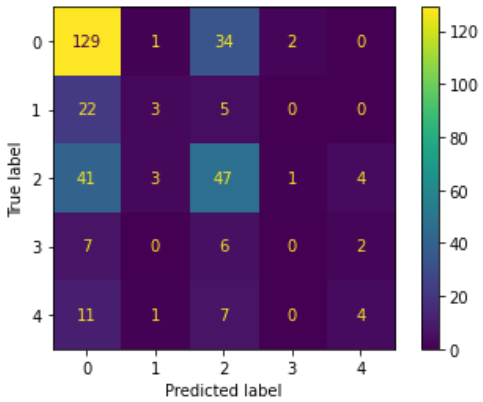
In [55]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:41:52] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[55]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42582b0b10>
```



VGG-19

In [56]:

```
base_model= VGG19(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg19/vgg19_weig
hts_tf_dim_ordering_tf_kernels_notop.h5
80142336/80134624 [==============================] - 2s 0us/step
```

In [57]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [58]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

----------------------- Train Set Metrics-----------------------

Accuracy core : 57.67%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 47.46%
----------------------- Test Set Metrics-----------------------

Accuracy score : 43.94%
F1_score : 0.39
Kappa Score : 0.02
Recall score: 0.44
Precision score : 0.37
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 48.22%
------------------------ Validation Set Metrics------------------------

Accuracy score : 52.410000000000004%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------

/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
------------------------ Train Set Metrics------------------------

Accuracy core : 60.550000000000004%
------------------------ Validation Set Metrics------------------------

Accuracy score : 49.93%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.39
Kappa Score : 0.04
Recall score: 0.5
Precision score : 0.37
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data-----------------------------
----

/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
----------------------- Train Set Metrics-----------------------

Accuracy core : 48.38%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 49.93%
----------------------- Test Set Metrics-----------------------

Accuracy score : 48.79%
F1_score : 0.35
Kappa Score : -0.0
Recall score: 0.49
Precision score : 0.32
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:42:07] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

----------------------- Train Set Metrics-----------------------

Accuracy core : 97.74000000000001%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 45.25%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.91%
F1_score : 0.47
Kappa Score : 0.15
Recall score: 0.51
Precision score : 0.46
--------------------------------------------------------------------------------
```

In [59]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 8ms/step - loss: 1.3118 - accuracy: 0.4817 - val_loss:
1.2729 - val_accuracy: 0.5241
Epoch 2/10
81/81 [==============================] - 0s 5ms/step - loss: 1.3045 - accuracy: 0.4872 - val_loss:
1.2767 - val_accuracy: 0.5241
Epoch 3/10
81/81 [==============================] - 0s 4ms/step - loss: 1.3037 - accuracy: 0.4719 - val_loss:
1.2747 - val_accuracy: 0.5241
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2932 - accuracy: 0.4863 - val_loss:
1.2738 - val_accuracy: 0.5241
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2928 - accuracy: 0.4858 - val_loss:
1.2818 - val_accuracy: 0.5241
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3017 - accuracy: 0.4818 - val_loss:
1.2787 - val_accuracy: 0.5241
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2932 - accuracy: 0.5001 - val_loss:
1.2828 - val_accuracy: 0.5241
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3107 - accuracy: 0.4699 - val_loss:
1.2724 - val_accuracy: 0.5241
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2973 - accuracy: 0.4850 - val_loss:
1.2736 - val_accuracy: 0.5241
Epoch 10/10
81/81 [==============================] - 0s 4ms/step - loss: 1.3049 - accuracy: 0.4858 - val_loss:
1.2835 - val_accuracy: 0.5241
81/81 [==============================] - 0s 2ms/step - loss: 1.3063 - accuracy: 0.4822
Train_accuracy is:0.48224735260009766
25/25 [==============================] - 0s 2ms/step - loss: 1.2835 - accuracy: 0.5241
Validation_accuracy is := 0.5240572094917297
11/11 [==============================] - 0s 2ms/step - loss: 1.2595 - accuracy: 0.5030
test_accuracy is : = 0.5030303001403809
```

In [60]:

```
print("Performance Report:")
y_pred1=dnn_model.predict_classes(test_features)
y_test1=[np.argmax(x) for x in test_y]
y_pred_prb1=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test1, y_pred1),4))
print('Precision score is :', np.round(metrics.precision_score(y_test1, y_pred1, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test1,y_pred1, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test1, y_pred1, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test1, y_pred1),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test1, y_pred1,target_names=targe
t))
```

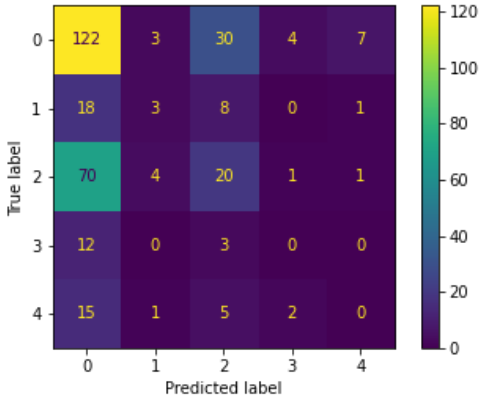
```
Performance Report:
Accuracy score is : 0.503
Precision score is : 0.253
Recall score is : 0.503
F1 Score is : 0.3367
Cohen Kappa Score: 0.0
                Classification Report:
               precision    recall  f1-score   support

           0       0.50      1.00      0.67       166
           1       0.00      0.00      0.00        30
           2       0.00      0.00      0.00        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.50       330
   macro avg       0.10      0.20      0.13       330
weighted avg       0.25      0.50      0.34       330



/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [61]:

```
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[61]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42304fc350>
```

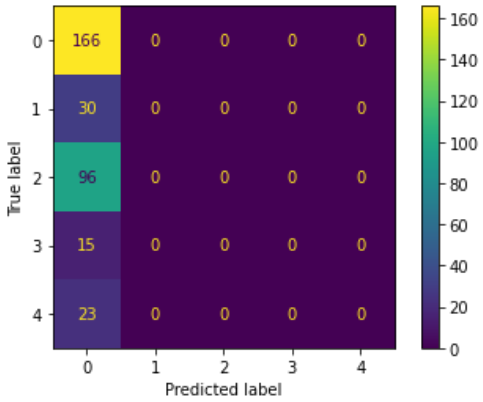


In [62]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[62]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4230193710>
```

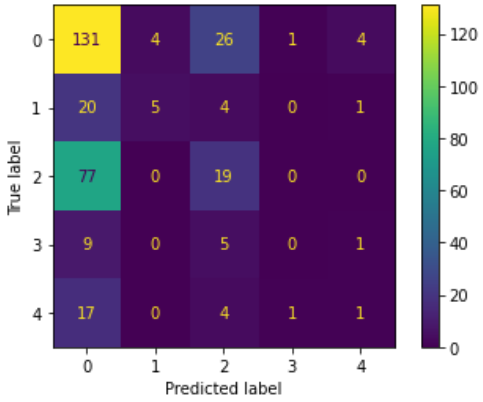


In [63]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

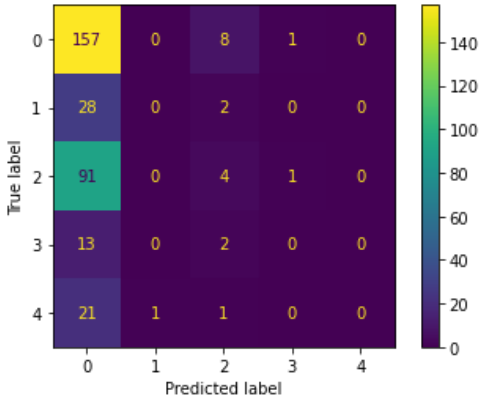
Out[63]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42345e8850>
```



In [64]:
```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

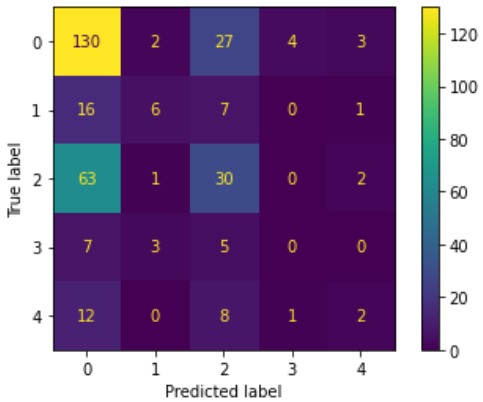
Out[64]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e423020f650>
```



In [65]:
```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:42:15] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[65]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42346f9c10>
```



ResNet101

In [66]:

```
base_model= ResNet101(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet101
_weights_tf_dim_ordering_tf_kernels_notop.h5
171450368/171446536 [==============================] - 6s 0us/step
```

In [67]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [68]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

----------------------- Train Set Metrics-----------------------

Accuracy core : 62.86000000000001%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 54.36%
----------------------- Test Set Metrics-----------------------

Accuracy score : 55.45%
F1_score : 0.51
Kappa Score : 0.23
Recall score: 0.55
Precision score : 0.52
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 54.31%
------------------------ Validation Set Metrics------------------------

Accuracy score : 57.220000000000006%
------------------------ Test Set Metrics------------------------

Accuracy score : 57.58%
F1_score : 0.5
Kappa Score : 0.24
Recall score: 0.58
Precision score : 0.45
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 75.69%
------------------------ Validation Set Metrics------------------------

Accuracy score : 60.209999999999994%
------------------------ Test Set Metrics------------------------

Accuracy score : 60.0%
F1_score : 0.54
Kappa Score : 0.31
Recall score: 0.6
Precision score : 0.54
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data----------------------------
----
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


------------------------ Train Set Metrics------------------------

Accuracy core : 58.64%
------------------------ Validation Set Metrics------------------------

Accuracy score : 57.74%
------------------------ Test Set Metrics------------------------

Accuracy score : 59.089999999999996%
F1_score : 0.53
Kappa Score : 0.28
Recall score: 0.59
Precision score : 0.49
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:42:39] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

------------------------ Train Set Metrics------------------------

Accuracy core : 98.28%
------------------------ Validation Set Metrics------------------------

Accuracy score : 58.91%
------------------------ Test Set Metrics------------------------

Accuracy score : 62.12%
F1_score : 0.59
Kappa Score : 0.37
Recall score: 0.62
Precision score : 0.58
--------------------------------------------------------------------------------
```

In [69]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 8ms/step - loss: 1.2824 - accuracy: 0.4823 - val_loss:
1.1654 - val_accuracy: 0.5605
Epoch 2/10
81/81 [==============================] - 0s 5ms/step - loss: 1.2109 - accuracy: 0.5415 - val_loss:
1.1424 - val_accuracy: 0.5735
Epoch 3/10
81/81 [==============================] - 0s 5ms/step - loss: 1.2270 - accuracy: 0.5273 - val_loss:
1.1708 - val_accuracy: 0.5553
Epoch 4/10
81/81 [==============================] - 0s 4ms/step - loss: 1.2201 - accuracy: 0.5250 - val_loss:
1.1288 - val_accuracy: 0.5813
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2174 - accuracy: 0.5332 - val_loss:
1.1647 - val_accuracy: 0.5657
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2030 - accuracy: 0.5377 - val_loss:
1.1208 - val_accuracy: 0.5865
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1868 - accuracy: 0.5525 - val_loss:
1.1184 - val_accuracy: 0.5774
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1764 - accuracy: 0.5598 - val_loss:
1.1302 - val_accuracy: 0.5813
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1783 - accuracy: 0.5632 - val_loss:
1.1232 - val_accuracy: 0.5839
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1786 - accuracy: 0.5477 - val_loss:
1.1316 - val_accuracy: 0.5930
81/81 [==============================] - 0s 2ms/step - loss: 1.1705 - accuracy: 0.5587
Train_accuracy is:0.5587202310562134
25/25 [==============================] - 0s 2ms/step - loss: 1.1316 - accuracy: 0.5930
Validation_accuracy is := 0.5929778814315796
11/11 [==============================] - 0s 2ms/step - loss: 1.1335 - accuracy: 0.5758
test_accuracy is : = 0.5757575631141663
```

In [70]:

```
print("Performance Report:")
y_pred3=dnn_model.predict_classes(test_features)
y_test3=[np.argmax(x) for x in test_y]
y_pred_prb3=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test3, y_pred3),4))
print('Precision score is :', np.round(metrics.precision_score(y_test3, y_pred3, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test3,y_pred3, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test3, y_pred3, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test3, y_pred3),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test3, y_pred3,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.5758
Precision score is : 0.4655
Recall score is : 0.5758
F1 Score is : 0.514
Cohen Kappa Score: 0.2732
                Classification Report:
               precision    recall  f1-score   support

           0       0.68      0.80      0.73       166
           1       0.00      0.00      0.00        30
           2       0.43      0.60      0.50        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.58       330
   macro avg       0.22      0.28      0.25       330
weighted avg       0.47      0.58      0.51       330
```

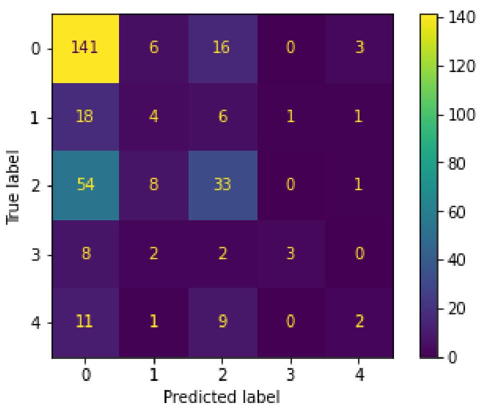
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [71]:

```
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[71]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4278245a50>
```

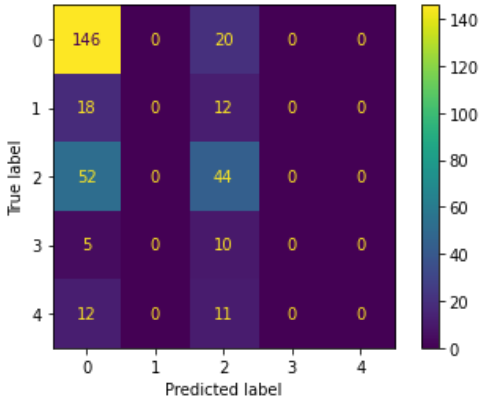


In [72]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[72]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e54f18890>
```

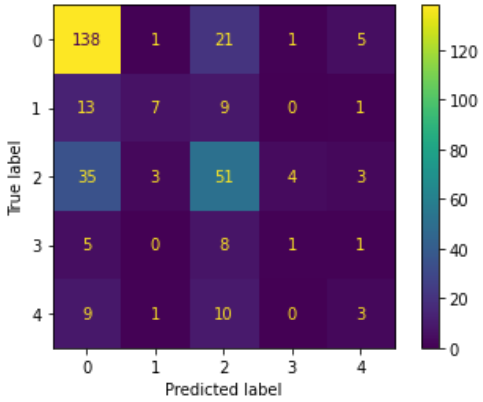


In [73]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[73]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e423045db50>
```

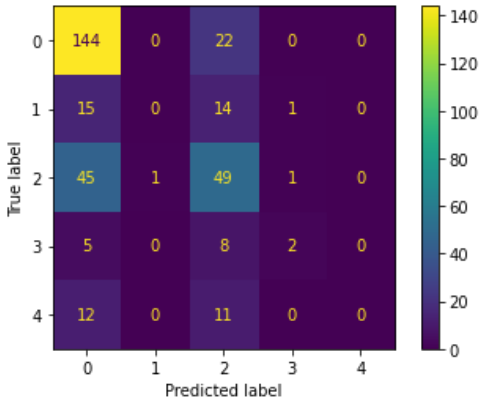


In [74]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[74]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42207b1cd0>
```



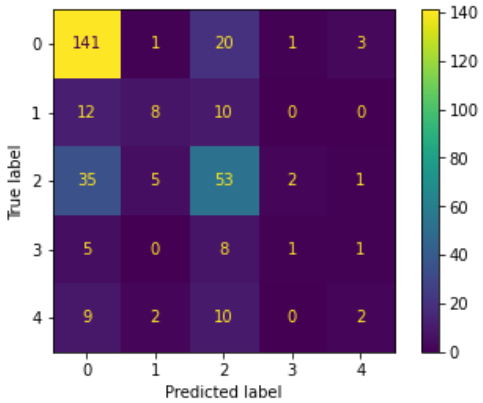
In [75]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:42:47] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[75]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4220642410>
```



MobileNetV2

In [76]:

```
base_model= MobileNetV2(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet_v2/mob
ilenet_v2_weights_tf_dim_ordering_tf_kernels_1.0_224_no_top.h5
9412608/9406464 [==============================] - 1s 0us/step
```

In [77]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("-----------------------------Fitting {} on input_data-------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [78]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

----------------------- Train Set Metrics-----------------------

Accuracy core : 65.47%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.54%
----------------------- Test Set Metrics-----------------------

Accuracy score : 55.15%
F1_score : 0.51
Kappa Score : 0.25
Recall score: 0.55
Precision score : 0.48
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 57.38999999999999%
------------------------ Validation Set Metrics------------------------

Accuracy score : 57.48%
------------------------ Test Set Metrics------------------------

Accuracy score : 57.269999999999996%
F1_score : 0.52
Kappa Score : 0.29
Recall score: 0.57
Precision score : 0.49
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
------------------------ Train Set Metrics------------------------

Accuracy core : 71.56%
------------------------ Validation Set Metrics------------------------

Accuracy score : 56.57%
------------------------ Test Set Metrics------------------------

Accuracy score : 59.089999999999996%
F1_score : 0.54
Kappa Score : 0.32
Recall score: 0.59
Precision score : 0.56
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data-----------------------------
----


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
----------------------- Train Set Metrics-----------------------

Accuracy core : 54.86%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 53.449999999999996%
----------------------- Test Set Metrics-----------------------

Accuracy score : 51.82%
F1_score : 0.47
Kappa Score : 0.19
Recall score: 0.52
Precision score : 0.46
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:42:58] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

----------------------- Train Set Metrics-----------------------

Accuracy core : 98.91%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 54.49%
----------------------- Test Set Metrics-----------------------

Accuracy score : 54.55%
F1_score : 0.52
Kappa Score : 0.26
Recall score: 0.55
Precision score : 0.53
--------------------------------------------------------------------------------
```

In [79]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 5ms/step - loss: 1.2965 - accuracy: 0.4853 - val_loss:
1.1739 - val_accuracy: 0.5566
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1809 - accuracy: 0.5383 - val_loss:
1.1747 - val_accuracy: 0.5501
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1844 - accuracy: 0.5241 - val_loss:
1.1403 - val_accuracy: 0.5618
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1708 - accuracy: 0.5418 - val_loss:
1.1337 - val_accuracy: 0.5605
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1564 - accuracy: 0.5283 - val_loss:
1.1402 - val_accuracy: 0.5540
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1335 - accuracy: 0.5526 - val_loss:
1.1244 - val_accuracy: 0.5605
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1147 - accuracy: 0.5710 - val_loss:
1.1262 - val_accuracy: 0.5657
Epoch 8/10
81/81 [==============================] - 0s 4ms/step - loss: 1.1221 - accuracy: 0.5594 - val_loss:
1.1335 - val_accuracy: 0.5618
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1196 - accuracy: 0.5535 - val_loss:
1.1260 - val_accuracy: 0.5605
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1122 - accuracy: 0.5717 - val_loss:
1.1282 - val_accuracy: 0.5605
81/81 [==============================] - 0s 2ms/step - loss: 1.1269 - accuracy: 0.5568
Train_accuracy is:0.5567694306373596
25/25 [==============================] - 0s 2ms/step - loss: 1.1282 - accuracy: 0.5605
Validation_accuracy is := 0.5604681372642517
11/11 [==============================] - 0s 2ms/step - loss: 1.1036 - accuracy: 0.5667
test_accuracy is : = 0.5666666626930237
```

In [80]:

```
print("Performance Report:")
y_pred4=dnn_model.predict_classes(test_features)
y_test4=[np.argmax(x) for x in test_y]
y_pred_prb4=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test4, y_pred4),4))
print('Precision score is :', np.round(metrics.precision_score(y_test4, y_pred4, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test4,y_pred4, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test4, y_pred4, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test4, y_pred4),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test4, y_pred4,target_names=targe
t))
```

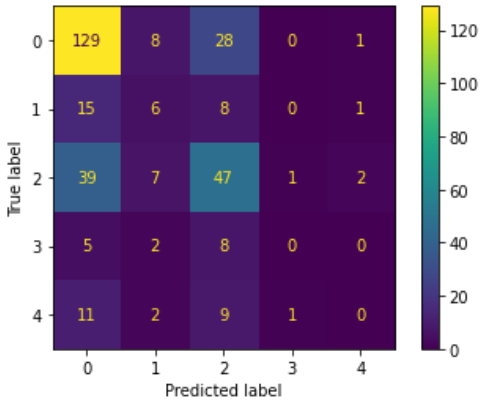
```
Performance Report:
Accuracy score is : 0.5667
Precision score is : 0.4988
Recall score is : 0.5667
F1 Score is : 0.5147
Cohen Kappa Score: 0.2942
                Classification Report:
               precision    recall  f1-score   support

           0       0.75      0.67      0.71       166
           1       0.00      0.00      0.00        30
           2       0.42      0.79      0.55        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.57       330
   macro avg       0.23      0.29      0.25       330
weighted avg       0.50      0.57      0.51       330



/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [81]:

```
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[81]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e55398690>
```

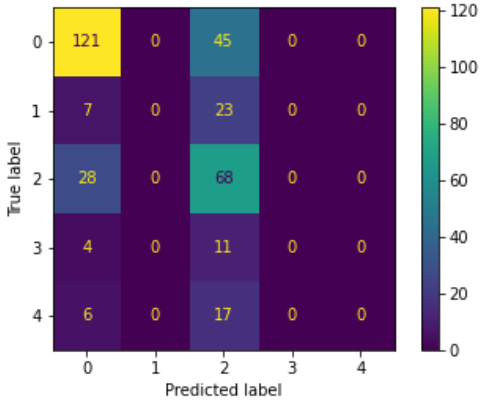


In [82]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[82]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4230534390>
```

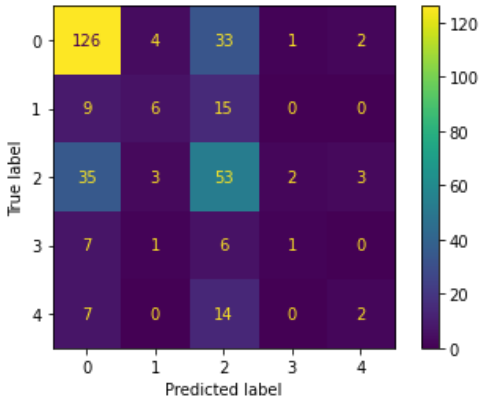


In [83]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[83]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42ac1c6c50>
```

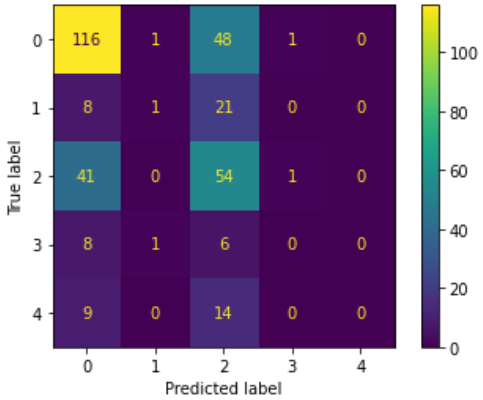


In [84]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[84]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42ac053710>
```



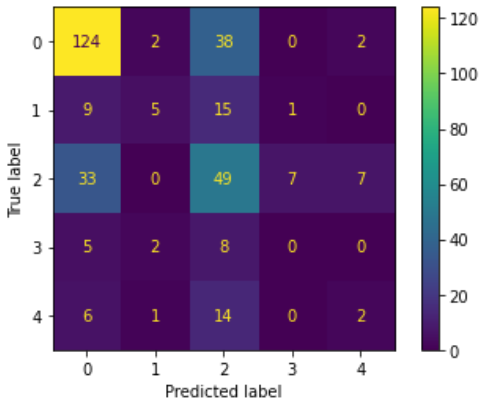
In [85]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:43:06] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[85]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42ac1b2710>
```



MobileNet

In [86]:

```
base_model= MobileNet(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet/mobile
net_1_0_224_tf_no_top.h5
17227776/17225924 [==============================] - 1s 0us/step
```

In [87]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [88]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

----------------------- Train Set Metrics-----------------------

Accuracy core : 63.29%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 50.72%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.91%
F1_score : 0.47
Kappa Score : 0.18
Recall score: 0.51
Precision score : 0.46
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 49.2%
------------------------ Validation Set Metrics------------------------

Accuracy score : 53.32%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.61%
F1_score : 0.36
Kappa Score : 0.02
Recall score: 0.51
Precision score : 0.35
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 69.41000000000001%
------------------------ Validation Set Metrics------------------------

Accuracy score : 54.620000000000005%
------------------------ Test Set Metrics------------------------

Accuracy score : 53.03%
F1_score : 0.47
Kappa Score : 0.17
Recall score: 0.53
Precision score : 0.42
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data----------------------------
----
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


------------------------ Train Set Metrics------------------------

Accuracy core : 51.78%
------------------------ Validation Set Metrics------------------------

Accuracy score : 51.11%
------------------------ Test Set Metrics------------------------

Accuracy score : 51.519999999999996%
F1_score : 0.47
Kappa Score : 0.16
Recall score: 0.52
Precision score : 0.44
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:43:15] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

------------------------ Train Set Metrics------------------------

Accuracy core : 98.79%
------------------------ Validation Set Metrics------------------------

Accuracy score : 54.1%
------------------------ Test Set Metrics------------------------

Accuracy score : 55.15%
F1_score : 0.53
Kappa Score : 0.26
Recall score: 0.55
Precision score : 0.51
--------------------------------------------------------------------------------
```

In [89]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 5ms/step - loss: 1.7029 - accuracy: 0.4594 - val_loss:
1.2200 - val_accuracy: 0.4707
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2127 - accuracy: 0.4825 - val_loss:
1.1848 - val_accuracy: 0.5241
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1953 - accuracy: 0.5211 - val_loss:
1.1800 - val_accuracy: 0.5358
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2127 - accuracy: 0.4949 - val_loss:
1.2342 - val_accuracy: 0.4551
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2027 - accuracy: 0.4995 - val_loss:
1.1722 - val_accuracy: 0.5449
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1837 - accuracy: 0.5111 - val_loss:
1.1715 - val_accuracy: 0.5488
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1899 - accuracy: 0.5120 - val_loss:
1.1788 - val_accuracy: 0.5319
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1965 - accuracy: 0.5131 - val_loss:
1.1722 - val_accuracy: 0.5462
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1848 - accuracy: 0.5181 - val_loss:
1.1740 - val_accuracy: 0.5371
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1802 - accuracy: 0.5265 - val_loss:
1.1893 - val_accuracy: 0.5410
81/81 [==============================] - 0s 2ms/step - loss: 1.2102 - accuracy: 0.4990
Train_accuracy is:0.49902456998825073
25/25 [==============================] - 0s 2ms/step - loss: 1.1893 - accuracy: 0.5410
Validation_accuracy is := 0.540962278842926
11/11 [==============================] - 0s 2ms/step - loss: 1.1868 - accuracy: 0.5212
test_accuracy is : = 0.521212100982666
```

In [90]:

```
print("Performance Report:")
y_pred5=dnn_model.predict_classes(test_features)
y_test5=[np.argmax(x) for x in test_y]
y_pred_prb5=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test5, y_pred5),4))
print('Precision score is :', np.round(metrics.precision_score(y_test5, y_pred5, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test5,y_pred5, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test5, y_pred5, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test5, y_pred5),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test5, y_pred5,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.5212
Precision score is : 0.4023
Recall score is : 0.5212
F1 Score is : 0.3928
Cohen Kappa Score: 0.0621
                Classification Report:
               precision    recall  f1-score   support

           0       0.52      0.98      0.68       166
           1       0.00      0.00      0.00        30
           2       0.48      0.10      0.17        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.52       330
   macro avg       0.20      0.22      0.17       330
weighted avg       0.40      0.52      0.39       330
```

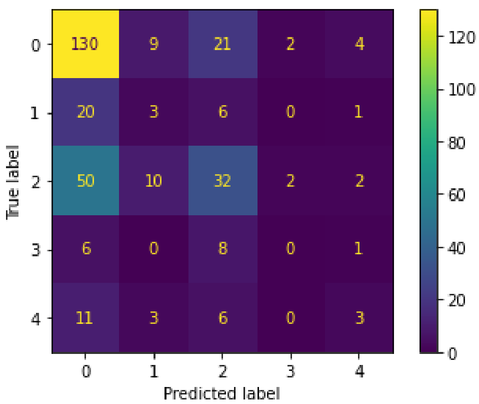
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [91]:

```python
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[91]:

```
&lt;sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e405af110&gt;
```

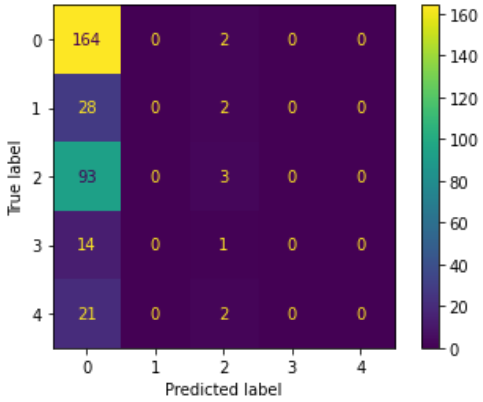


In [92]:

```python
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[92]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e403adbd0>
```

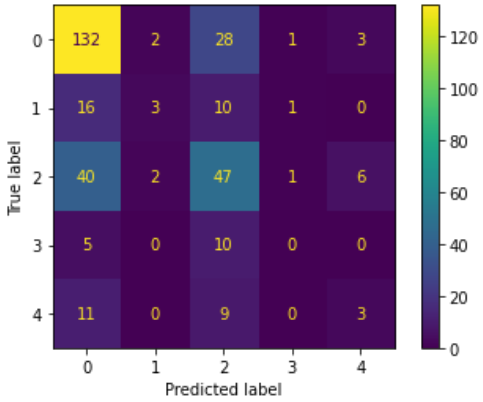


In [93]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[93]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e54436690>
```

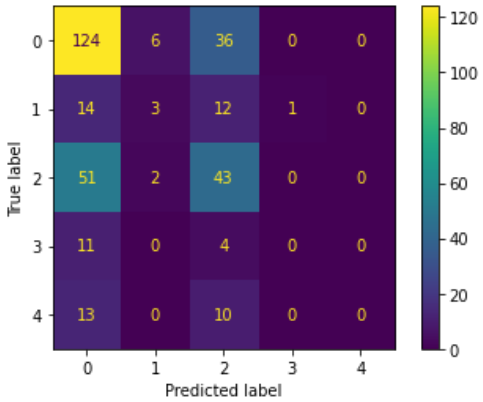


In [94]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[94]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42ac187050>
```



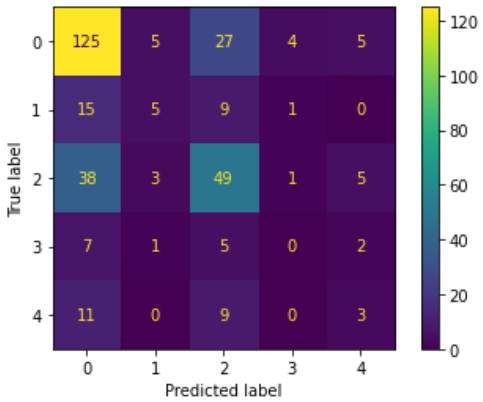
In [95]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:43:23] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[95]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4230175f10>
```



InceptionV3

In [96]:

```
base_model= MobileNet(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

In [97]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [98]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

----------------------- Train Set Metrics-----------------------

Accuracy core : 66.84%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 56.830000000000005%
----------------------- Test Set Metrics-----------------------

Accuracy score : 56.06%
F1_score : 0.54
Kappa Score : 0.3
Recall score: 0.56
Precision score : 0.53
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))

------------------------ Train Set Metrics------------------------

Accuracy core : 59.19%
------------------------ Validation Set Metrics------------------------

Accuracy score : 62.68%
------------------------ Test Set Metrics------------------------

Accuracy score : 61.519999999999996%
F1_score : 0.54
Kappa Score : 0.32
Recall score: 0.62
Precision score : 0.48
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------

/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))

------------------------ Train Set Metrics------------------------

Accuracy core : 78.66%
------------------------ Validation Set Metrics------------------------

Accuracy score : 63.849999999999994%
------------------------ Test Set Metrics------------------------

Accuracy score : 63.94%
F1_score : 0.59
Kappa Score : 0.39
Recall score: 0.64
Precision score : 0.56
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data----------------------------
----
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


----------------------- Train Set Metrics-----------------------

Accuracy core : 60.709999999999994%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 62.029999999999994%
----------------------- Test Set Metrics-----------------------

Accuracy score : 60.91%
F1_score : 0.56
Kappa Score : 0.34
Recall score: 0.61
Precision score : 0.54
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:43:31] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

----------------------- Train Set Metrics-----------------------

Accuracy core : 99.1%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 60.6%
----------------------- Test Set Metrics-----------------------

Accuracy score : 64.85%
F1_score : 0.62
Kappa Score : 0.44
Recall score: 0.65
Precision score : 0.61
--------------------------------------------------------------------------------
```

In [99]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 5ms/step - loss: 1.6981 - accuracy: 0.4296 - val_loss:
1.2808 - val_accuracy: 0.5228
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3100 - accuracy: 0.4583 - val_loss:
1.2538 - val_accuracy: 0.5254
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2740 - accuracy: 0.4976 - val_loss:
1.2633 - val_accuracy: 0.5176
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2551 - accuracy: 0.4871 - val_loss:
1.1861 - val_accuracy: 0.5319
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2296 - accuracy: 0.5175 - val_loss:
1.1251 - val_accuracy: 0.5982
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1756 - accuracy: 0.5623 - val_loss:
1.1728 - val_accuracy: 0.5722
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1705 - accuracy: 0.5549 - val_loss:
1.0796 - val_accuracy: 0.6112
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1303 - accuracy: 0.5923 - val_loss:
1.1234 - val_accuracy: 0.5995
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.1224 - accuracy: 0.5989 - val_loss:
1.0728 - val_accuracy: 0.6203
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.0962 - accuracy: 0.6033 - val_loss:
1.1013 - val_accuracy: 0.6073
81/81 [==============================] - 0s 2ms/step - loss: 1.1197 - accuracy: 0.6009
Train_accuracy is:0.6008583903312683
25/25 [==============================] - 0s 2ms/step - loss: 1.1013 - accuracy: 0.6073
Validation_accuracy is := 0.6072821617126465
11/11 [==============================] - 0s 2ms/step - loss: 1.0718 - accuracy: 0.6000
test_accuracy is : = 0.6000000238418579
```

In [100]:

```
print("Performance Report:")
y_pred6=dnn_model.predict_classes(test_features)
y_test6=[np.argmax(x) for x in test_y]
y_pred_prb6=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test6, y_pred6),4))
print('Precision score is :', np.round(metrics.precision_score(y_test6, y_pred6, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test6,y_pred6, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test6, y_pred6, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test6, y_pred6),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test6, y_pred6,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.6
Precision score is : 0.5179
Recall score is : 0.6
F1 Score is : 0.5454
Cohen Kappa Score: 0.3402
                Classification Report:
               precision    recall  f1-score   support

           0       0.78      0.75      0.77       166
           1       0.00      0.00      0.00        30
           2       0.43      0.76      0.55        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.60       330
   macro avg       0.24      0.30      0.26       330
weighted avg       0.52      0.60      0.55       330
```

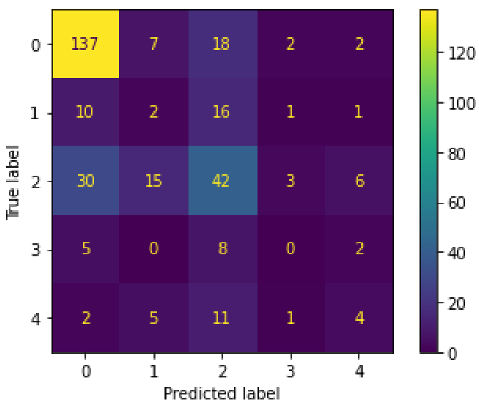
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [101]:

```
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[101]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e5541b950>
```

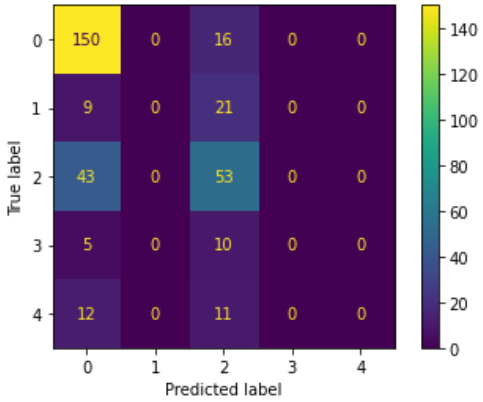


In [102]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[102]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e55467490>
```

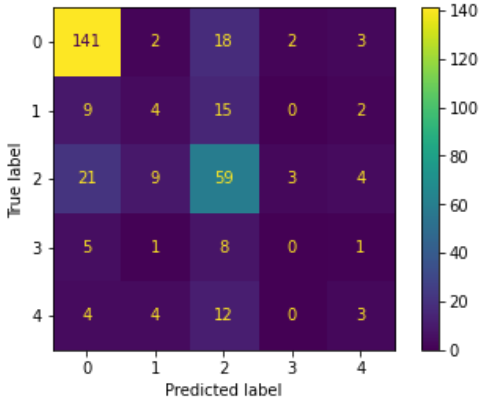


In [103]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[103]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e4021ef90>
```

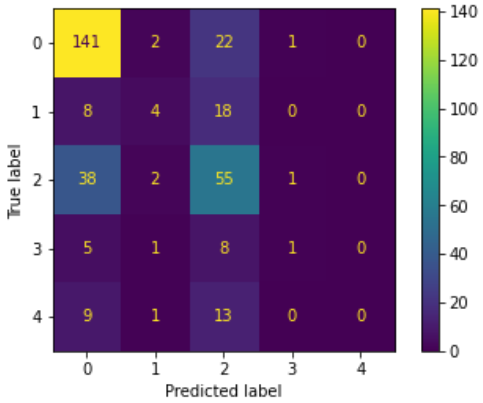


In [104]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[104]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e401011d0>
```



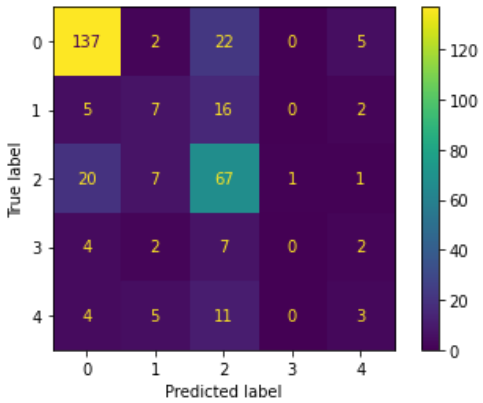
In [105]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:43:38] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[105]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e4006f250>
```



InceptionResNetV2

In [106]:

```
base_model= InceptionResNetV2(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_resnet
_v2/inception_resnet_v2_weights_tf_dim_ordering_tf_kernels_notop.h5
219062272/219055592 [==============================] - 7s 0us/step
```

In [107]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("-----------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [108]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
------------------------ Train Set Metrics------------------------

Accuracy core : 30.080000000000002%
------------------------ Validation Set Metrics------------------------

Accuracy score : 26.009999999999998%
------------------------ Test Set Metrics------------------------

Accuracy score : 32.12%
F1_score : 0.2
Kappa Score : 0.03
Recall score: 0.32
Precision score : 0.46
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 48.22%
------------------------ Validation Set Metrics------------------------

Accuracy score : 52.410000000000004%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
------------------------ Train Set Metrics------------------------

Accuracy core : 48.26%
------------------------ Validation Set Metrics------------------------

Accuracy score : 52.410000000000004%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data-----------------------------
----

------------------------ Train Set Metrics------------------------

Accuracy core : 48.26%
------------------------ Validation Set Metrics------------------------

Accuracy score : 52.410000000000004%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:44:08] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
------------------------ Train Set Metrics-----------------------

Accuracy core : 49.120000000000005%
------------------------ Validation Set Metrics-----------------------

Accuracy score : 52.15%
------------------------ Test Set Metrics-----------------------

Accuracy score : 50.0%
F1_score : 0.34
Kappa Score : -0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [109]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 5ms/step - loss: 2.2411 - accuracy: 0.4642 - val_loss:
1.2745 - val_accuracy: 0.5241
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3123 - accuracy: 0.4696 - val_loss:
1.2716 - val_accuracy: 0.5241
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2951 - accuracy: 0.4856 - val_loss:
1.2733 - val_accuracy: 0.5241
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3062 - accuracy: 0.4782 - val_loss:
1.2732 - val_accuracy: 0.5241
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2940 - accuracy: 0.4983 - val_loss:
1.2783 - val_accuracy: 0.5241
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2973 - accuracy: 0.4932 - val_loss:
1.2725 - val_accuracy: 0.5241
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3314 - accuracy: 0.4664 - val_loss:
1.2851 - val_accuracy: 0.5241
Epoch 8/10
81/81 [==============================] - 0s 4ms/step - loss: 1.3453 - accuracy: 0.4593 - val_loss:
1.2717 - val_accuracy: 0.5241
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3317 - accuracy: 0.4756 - val_loss:
1.2817 - val_accuracy: 0.5241
Epoch 10/10
81/81 [==============================] - 0s 4ms/step - loss: 1.2941 - accuracy: 0.4909 - val_loss:
1.2845 - val_accuracy: 0.5241
81/81 [==============================] - 0s 2ms/step - loss: 1.3095 - accuracy: 0.4822
Train_accuracy is:0.48224735260009766
25/25 [==============================] - 0s 2ms/step - loss: 1.2845 - accuracy: 0.5241
Validation_accuracy is := 0.5240572094917297
11/11 [==============================] - 0s 2ms/step - loss: 1.2644 - accuracy: 0.5030
test_accuracy is : = 0.5030303001403809
```

In [110]:

```
print("Performance Report:")
y_pred7=dnn_model.predict_classes(test_features)
y_test7=[np.argmax(x) for x in test_y]
y_pred_prb7=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test7, y_pred7),4))
print('Precision score is :', np.round(metrics.precision_score(y_test7, y_pred7, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test7,y_pred7, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test7, y_pred7, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test7, y_pred7),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test7, y_pred7,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.503
Precision score is : 0.253
Recall score is : 0.503
F1 Score is : 0.3367
Cohen Kappa Score: 0.0
                Classification Report:
               precision    recall  f1-score   support

           0       0.50      1.00      0.67       166
           1       0.00      0.00      0.00        30
           2       0.00      0.00      0.00        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.50       330
   macro avg       0.10      0.20      0.13       330
weighted avg       0.25      0.50      0.34       330
```

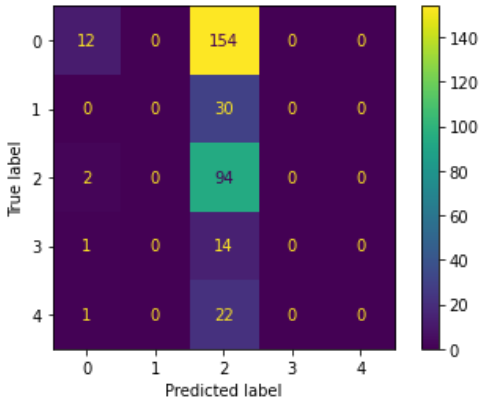
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [111]:

```
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[111]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e318f44d0>
```

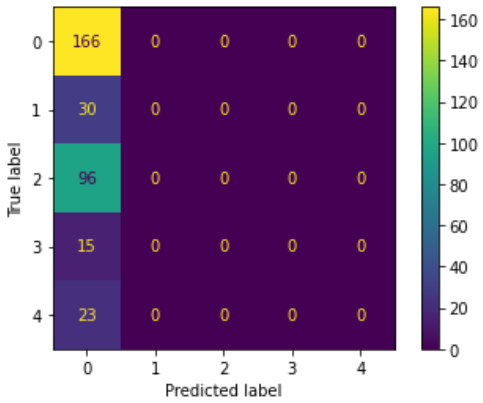


In [112]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[112]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e380f12d0>
```

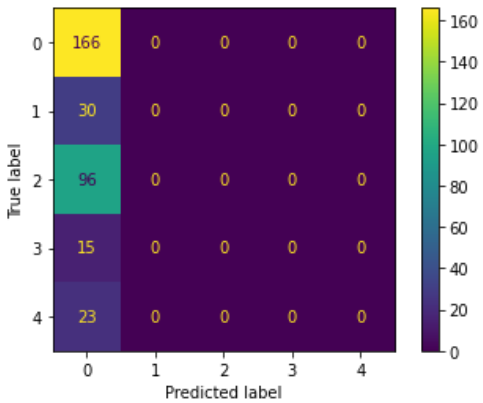


In [113]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

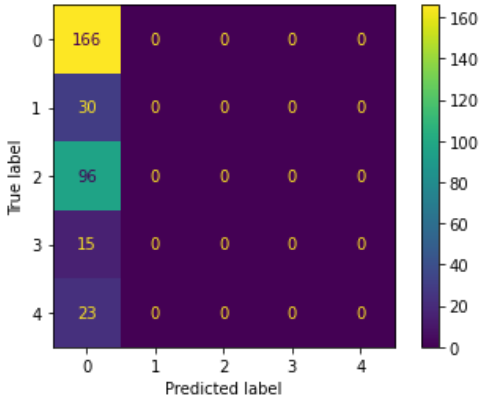
Out[113]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e5c2bb250>
```



In [114]:
```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

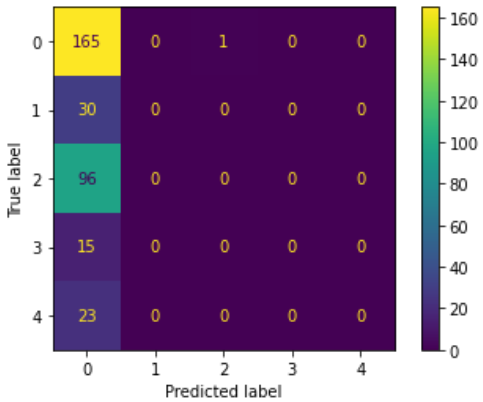
Out[114]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e42ac03f410>
```



In [115]:
```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:44:15] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[115]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e5c394cd0>
```



DenseNet169

In [116]:

```
base_model= InceptionResNetV2(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

In [117]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [118]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------

----------------------- Train Set Metrics-----------------------

Accuracy core : 53.769999999999996%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 54.230000000000004%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.91%
F1_score : 0.4
Kappa Score : 0.06
Recall score: 0.51
Precision score : 0.38
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 50.449999999999996%
------------------------ Validation Set Metrics------------------------

Accuracy score : 52.93%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.0%
F1_score : 0.4
Kappa Score : 0.05
Recall score: 0.5
Precision score : 0.36
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
------------------------ Train Set Metrics------------------------

Accuracy core : 53.1%
------------------------ Validation Set Metrics------------------------

Accuracy score : 53.059999999999995%
------------------------ Test Set Metrics------------------------

Accuracy score : 48.79%
F1_score : 0.39
Kappa Score : 0.03
Recall score: 0.49
Precision score : 0.36
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data-----------------------------
----

------------------------ Train Set Metrics------------------------

Accuracy core : 50.6%
------------------------ Validation Set Metrics------------------------

Accuracy score : 53.32%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.4
Kappa Score : 0.05
Recall score: 0.5
Precision score : 0.37
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:44:37] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

------------------------ Train Set Metrics------------------------

Accuracy core : 63.83%
------------------------ Validation Set Metrics------------------------

Accuracy score : 54.230000000000004%
------------------------ Test Set Metrics------------------------

Accuracy score : 49.39%
F1_score : 0.39
Kappa Score : 0.06
Recall score: 0.49
Precision score : 0.4
--------------------------------------------------------------------------------
```

In [119]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 5ms/step - loss: 1.3826 - accuracy: 0.4501 - val_loss:
1.2641 - val_accuracy: 0.5306
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2850 - accuracy: 0.5154 - val_loss:
1.2536 - val_accuracy: 0.5306
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2975 - accuracy: 0.4912 - val_loss:
1.2541 - val_accuracy: 0.5306
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3044 - accuracy: 0.4878 - val_loss:
1.2556 - val_accuracy: 0.5319
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2878 - accuracy: 0.5027 - val_loss:
1.2561 - val_accuracy: 0.5306
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2460 - accuracy: 0.5200 - val_loss:
1.2569 - val_accuracy: 0.5319
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2790 - accuracy: 0.5074 - val_loss:
1.2618 - val_accuracy: 0.5306
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3013 - accuracy: 0.5020 - val_loss:
1.2563 - val_accuracy: 0.5306
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3080 - accuracy: 0.4917 - val_loss:
1.2635 - val_accuracy: 0.5319
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2584 - accuracy: 0.5224 - val_loss:
1.2690 - val_accuracy: 0.5306
81/81 [==============================] - 0s 2ms/step - loss: 1.2811 - accuracy: 0.5045: 0s - loss:
1.2729 - accuracy: 0.
Train_accuracy is:0.5044869184494019
25/25 [==============================] - 0s 2ms/step - loss: 1.2690 - accuracy: 0.5306
Validation_accuracy is := 0.5305591821670532
11/11 [==============================] - 0s 2ms/step - loss: 1.2463 - accuracy: 0.5030
test_accuracy is : = 0.5030303001403809
```

In [120]:

```
print("Performance Report:")
y_pred8=dnn_model.predict_classes(test_features)
y_test8=[np.argmax(x) for x in test_y]
y_pred_prb8=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test8, y_pred8),4))
print('Precision score is :', np.round(metrics.precision_score(y_test8, y_pred8, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test8,y_pred8, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test8, y_pred8, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test8, y_pred8),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test8, y_pred8,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.503
Precision score is : 0.3682
Recall score is : 0.503
F1 Score is : 0.4006
Cohen Kappa Score: 0.055
                Classification Report:
               precision    recall  f1-score   support

           0       0.53      0.90      0.67       166
           1       0.00      0.00      0.00        30
           2       0.36      0.17      0.23        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.50       330
   macro avg       0.18      0.21      0.18       330
weighted avg       0.37      0.50      0.40       330
```

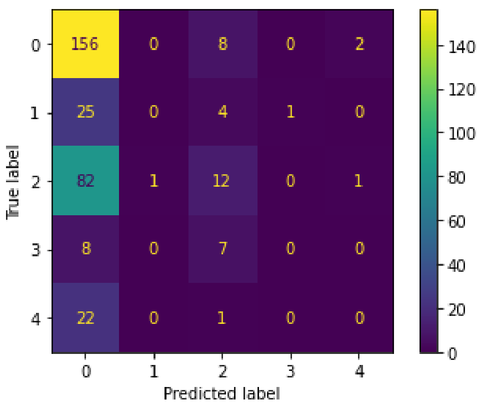
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [121]:

```python
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[121]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e2e658b50>
```

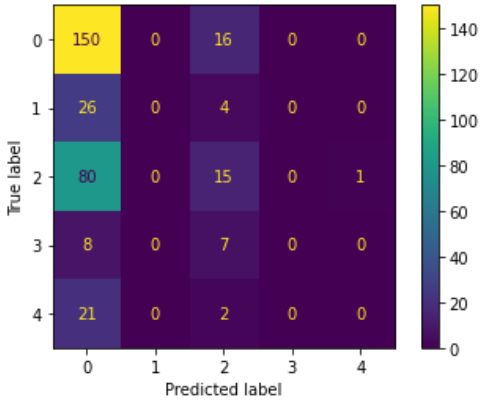


In [122]:

```python
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[122]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e2f7321d0>
```

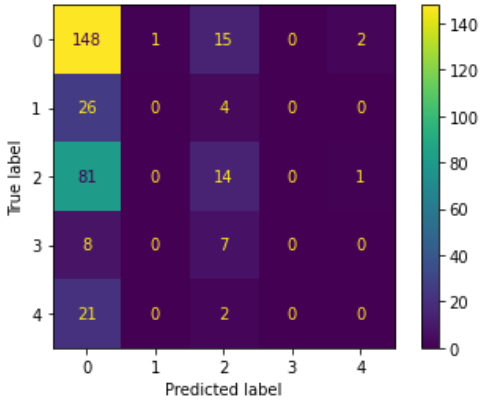


In [123]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[123]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e5554d710>
```

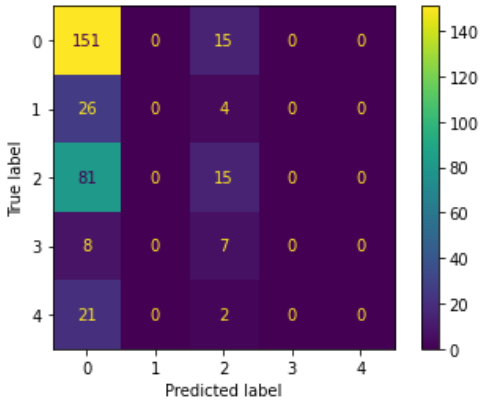


In [124]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[124]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e4053da50>
```



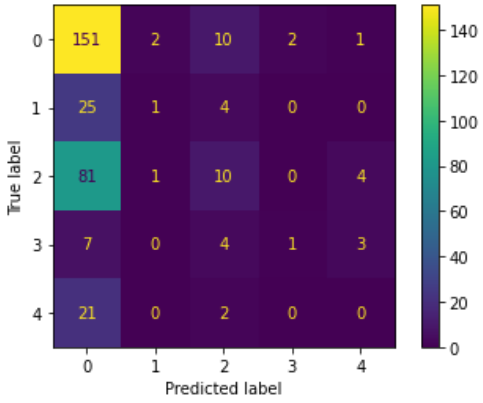
In [125]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:44:43] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[125]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e3872bdd0>
```



DenseNet121

In [126]:

```
base_model= InceptionResNetV2(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

In [127]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```python
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [128]:

```python
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
------------------------ Train Set Metrics-----------------------

Accuracy core : 10.03%
------------------------ Validation Set Metrics-----------------------

Accuracy score : 10.92%
------------------------ Test Set Metrics-----------------------

Accuracy score : 9.09%
F1_score : 0.02
Kappa Score : 0.0
Recall score: 0.09
Precision score : 0.01
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics-----------------------

Accuracy core : 48.22%
------------------------ Validation Set Metrics-----------------------

Accuracy score : 52.410000000000004%
------------------------ Test Set Metrics-----------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
----------------------- Train Set Metrics-----------------------

Accuracy core : 48.22%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.410000000000004%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data-----------------------------
----

----------------------- Train Set Metrics-----------------------

Accuracy core : 48.22%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.410000000000004%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:45:06] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

----------------------- Train Set Metrics-----------------------

Accuracy core : 48.22%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.410000000000004%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [129]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 1s 5ms/step - loss: 1.3070 - accuracy: 0.4735 - val_loss:
1.2758 - val_accuracy: 0.5241
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2820 - accuracy: 0.4983 - val_loss:
1.2738 - val_accuracy: 0.5241
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2973 - accuracy: 0.4872 - val_loss:
1.2806 - val_accuracy: 0.5241
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3155 - accuracy: 0.4774 - val_loss:
1.2764 - val_accuracy: 0.5241
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3118 - accuracy: 0.4840 - val_loss:
1.2810 - val_accuracy: 0.5241
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3100 - accuracy: 0.4866 - val_loss:
1.2744 - val_accuracy: 0.5241
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2993 - accuracy: 0.4793 - val_loss:
1.2754 - val_accuracy: 0.5241
Epoch 8/10
81/81 [==============================] - 0s 4ms/step - loss: 1.2982 - accuracy: 0.4863 - val_loss:
1.2785 - val_accuracy: 0.5241
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2983 - accuracy: 0.5008 - val_loss:
1.2838 - val_accuracy: 0.5241
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3144 - accuracy: 0.4892 - val_loss:
1.2754 - val_accuracy: 0.5241
81/81 [==============================] - 0s 2ms/step - loss: 1.3048 - accuracy: 0.4822
Train_accuracy is:0.48224735260009766
25/25 [==============================] - 0s 2ms/step - loss: 1.2754 - accuracy: 0.5241
Validation_accuracy is := 0.5240572094917297
11/11 [==============================] - 0s 2ms/step - loss: 1.2501 - accuracy: 0.5030
test_accuracy is : = 0.5030303001403809
```

In [130]:

```
print("Performance Report:")
y_pred9=dnn_model.predict_classes(test_features)
y_test9=[np.argmax(x) for x in test_y]
y_pred_prb9=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test9, y_pred9),4))
print('Precision score is :', np.round(metrics.precision_score(y_test9, y_pred9, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test9,y_pred9, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test9, y_pred9, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test9, y_pred9),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test9, y_pred9,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.503
Precision score is : 0.253
Recall score is : 0.503
F1 Score is : 0.3367
Cohen Kappa Score: 0.0
                Classification Report:
               precision    recall  f1-score   support

           0       0.50      1.00      0.67       166
           1       0.00      0.00      0.00        30
           2       0.00      0.00      0.00        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.50       330
   macro avg       0.10      0.20      0.13       330
weighted avg       0.25      0.50      0.34       330
```

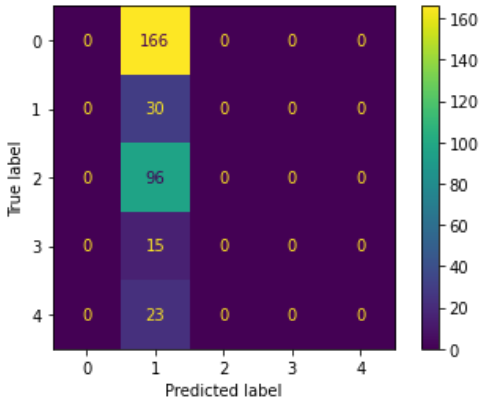
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [131]:

```
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[131]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e552c4710>
```

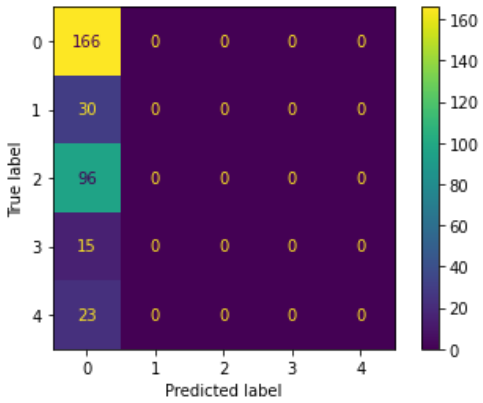


In [132]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[132]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e5403e310>
```

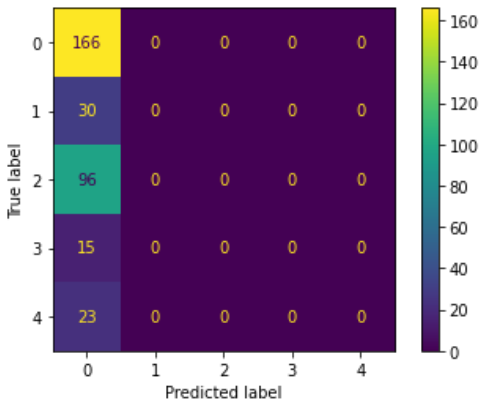


In [133]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

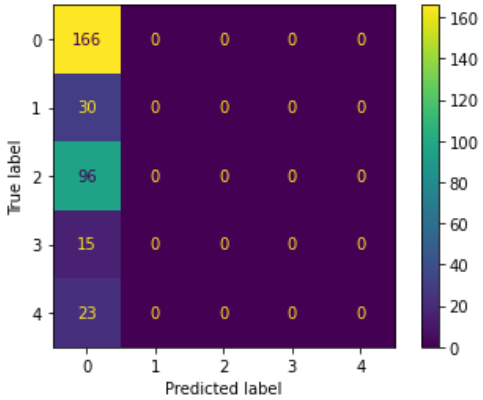
Out[133]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e4051c910>
```



In [134]:
```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

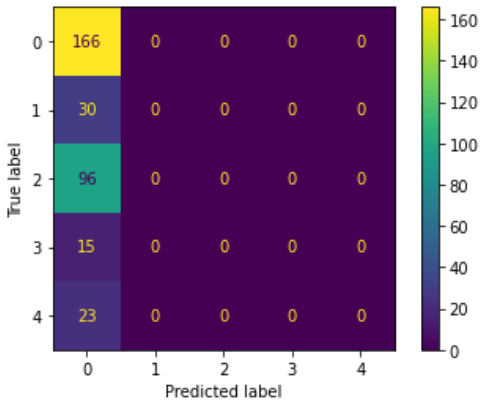
Out[134]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e2e43c910>
```



In [135]:
```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:45:12] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[135]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e2e446550>
```



XceptionNet

In [136]:

```
base_model= InceptionResNetV2(input_shape=(224,224,3), weights='imagenet', include_top=False)
x = base_model.output
# x = Dropout(0.5)(x)
x = Flatten()(x)
# x = BatchNormalization()(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(256,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(128,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
x = Dense(64,kernel_initializer='he_uniform')(x)
# x = BatchNormalization()(x)
x = Activation('relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(5, activation='softmax')(x)

model_feat = Model(inputs=base_model.input,outputs=predictions)

train_features = model_feat.predict(x_train)
val_features=model_feat.predict(x_val)
test_features=model_feat.predict(x_test)
```

In [137]:

```
from sklearn.pipeline import make_pipeline
from sklearn.pipeline import Pipeline
names = [
        "K Nearest Neighbour Classifier",
        'SVM',
        "Random Forest Classifier",
        "AdaBoost Classifier",
        "XGB Classifier"
         ]
classifiers = [
    KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30),
    SVC(),
    RandomForestClassifier(max_depth=9,criterion = 'entropy'),
    AdaBoostClassifier(),
    XGBClassifier()
        ]
zipped_clf = zip(names,classifiers)
def classifier_summary(pipeline, X_train, y_train, X_val, y_val,X_test,y_test):
    sentiment_fit = pipeline.fit(X_train, y_train)

    y_pred_train= sentiment_fit.predict(X_train)
    y_pred_val = sentiment_fit.predict(X_val)
    y_pred_test = sentiment_fit.predict(X_test)

    train_accuracy = np.round(accuracy_score(y_train, y_pred_train),4)*100
    train_precision = np.round(precision_score(y_train, y_pred_train, average='weighted'),4)
    train_recall = np.round(recall_score(y_train, y_pred_train, average='weighted'),4)
    train_F1 = np.round(f1_score(y_train, y_pred_train, average='weighted'),4)
    train_kappa =  np.round(cohen_kappa_score(y_train, y_pred_train),4)


    val_accuracy = np.round(accuracy_score(y_val, y_pred_val),4)*100
    val_precision = np.round(precision_score(y_val, y_pred_val, average='weighted'),4)
    val_recall = np.round(recall_score(y_val, y_pred_val, average='weighted'),4)
    val_F1 = np.round(f1_score(y_val, y_pred_val, average='weighted'),4)
    val_kappa =  np.round(cohen_kappa_score(y_val, y_pred_val),4)


    test_accuracy = np.round(accuracy_score(y_test, y_pred_test),4)*100
    test_precision = np.round(precision_score(y_test, y_pred_test, average='weighted'),2)
    test_recall = np.round(recall_score(y_test, y_pred_test, average='weighted'),2)
    test_F1 = np.round(f1_score(y_test, y_pred_test, average='weighted'),2)
    test_kappa =  np.round(cohen_kappa_score(y_test, y_pred_test),2)



    print()
    print('----------------------- Train Set Metrics-----------------------')
    print()
    print("Accuracy core : {}%".format(train_accuracy))

    print('----------------------- Validation Set Metrics-----------------------')
    print()
    print("Accuracy score : {}%".format(val_accuracy))
    print('----------------------- Test Set Metrics-----------------------')
```

```
    print()
    print("Accuracy score : {}%".format(test_accuracy))
    print("F1_score : {}".format(test_F1))
    print("Kappa Score : {} ".format(test_kappa))
    print("Recall score: {}".format(test_recall))
    print("Precision score : {}".format(test_precision))

    print("-"*80)
    print()

def classifier_comparator(X_train,y_train,X_val,y_val,X_test,y_test,classifier=zipped_clf):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([('Classifier', c)])
        print("------------------------------Fitting {} on input_data--------------------------------
".format(n))
        #print(c)
        classifier_summary(checker_pipeline,X_train, y_train, X_val, y_val,X_test,y_test)
```

In [138]:

```
classifier_comparator(train_features,y_train,val_features,y_val,test_features,y_test,classifier=zipped_
clf)
```

```
------------------------------Fitting K Nearest Neighbour Classifier on input_data-----------------
---------------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
------------------------ Train Set Metrics------------------------

Accuracy core : 49.32%
------------------------ Validation Set Metrics------------------------

Accuracy score : 50.33%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.38
Kappa Score : 0.03
Recall score: 0.5
Precision score : 0.36
--------------------------------------------------------------------------------

------------------------------Fitting SVM on input_data--------------------------------


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))


------------------------ Train Set Metrics------------------------

Accuracy core : 48.3%
------------------------ Validation Set Metrics------------------------

Accuracy score : 52.410000000000004%
------------------------ Test Set Metrics------------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting Random Forest Classifier on input_data-----------------------
---------
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
----------------------- Train Set Metrics-----------------------

Accuracy core : 49.24%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.15%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.0%
F1_score : 0.34
Kappa Score : -0.0
Recall score: 0.5
Precision score : 0.25
--------------------------------------------------------------------------------

------------------------------Fitting AdaBoost Classifier on input_data-----------------------------
----

----------------------- Train Set Metrics-----------------------

Accuracy core : 48.46%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 52.28%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.3%
F1_score : 0.34
Kappa Score : 0.0
Recall score: 0.5
Precision score : 0.35
--------------------------------------------------------------------------------

------------------------------Fitting XGB Classifier on input_data--------------------------------
[16:45:33] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.

----------------------- Train Set Metrics-----------------------

Accuracy core : 53.61%
----------------------- Validation Set Metrics-----------------------

Accuracy score : 51.629999999999995%
----------------------- Test Set Metrics-----------------------

Accuracy score : 50.0%
F1_score : 0.36
Kappa Score : 0.02
Recall score: 0.5
Precision score : 0.39
--------------------------------------------------------------------------------
```

In [139]:

```
train_y=to_categorical(y_train,5)
val_y=to_categorical(y_val,5)
test_y=to_categorical(y_test,5)
dnn_model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
history = dnn_model.fit(train_features, train_y,validation_data=(val_features,val_y), epochs=10)
loss_value , accuracy = dnn_model.evaluate(train_features, train_y)
print('Train_accuracy is:' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(val_features, val_y)
print('Validation_accuracy is := ' + str(accuracy))
loss_value , accuracy = dnn_model.evaluate(test_features, test_y)
print('test_accuracy is : = ' + str(accuracy))
```

```
Epoch 1/10
81/81 [==============================] - 2s 5ms/step - loss: 1.3382 - accuracy: 0.4413 - val_loss:
1.2708 - val_accuracy: 0.5254
Epoch 2/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3139 - accuracy: 0.4691 - val_loss:
1.2673 - val_accuracy: 0.5254
Epoch 3/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3009 - accuracy: 0.4863 - val_loss:
1.2724 - val_accuracy: 0.5254
Epoch 4/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3139 - accuracy: 0.4785 - val_loss:
1.2740 - val_accuracy: 0.5254
Epoch 5/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3075 - accuracy: 0.4780 - val_loss:
1.2740 - val_accuracy: 0.5254
Epoch 6/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2935 - accuracy: 0.4963 - val_loss:
1.2792 - val_accuracy: 0.5241
Epoch 7/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2876 - accuracy: 0.4880 - val_loss:
1.2761 - val_accuracy: 0.5241
Epoch 8/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3115 - accuracy: 0.4801 - val_loss:
1.2664 - val_accuracy: 0.5241
Epoch 9/10
81/81 [==============================] - 0s 3ms/step - loss: 1.2890 - accuracy: 0.4827 - val_loss:
1.2732 - val_accuracy: 0.5241
Epoch 10/10
81/81 [==============================] - 0s 3ms/step - loss: 1.3060 - accuracy: 0.4870 - val_loss:
1.2767 - val_accuracy: 0.5241
81/81 [==============================] - 0s 2ms/step - loss: 1.2988 - accuracy: 0.4830
Train_accuracy is:0.48302769660949707
25/25 [==============================] - 0s 2ms/step - loss: 1.2767 - accuracy: 0.5241
Validation_accuracy is := 0.5240572094917297
11/11 [==============================] - 0s 2ms/step - loss: 1.2552 - accuracy: 0.5030
test_accuracy is : = 0.5030303001403809
```

In [140]:

```
print("Performance Report:")
y_pred9=dnn_model.predict_classes(test_features)
y_test9=[np.argmax(x) for x in test_y]
y_pred_prb9=dnn_model.predict_proba(test_features)
target=['0','1','2','3','4']
from sklearn import metrics
print('Accuracy score is :', np.round(metrics.accuracy_score(y_test9, y_pred9),4))
print('Precision score is :', np.round(metrics.precision_score(y_test9, y_pred9, average='weighted'),
4))
print('Recall score is :', np.round(metrics.recall_score(y_test9,y_pred9, average='weighted'),4))
print('F1 Score is :', np.round(metrics.f1_score(y_test9, y_pred9, average='weighted'),4))
print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(y_test9, y_pred9),4))
print('\t\tClassification Report:\n', metrics.classification_report(y_test9, y_pred9,target_names=targe
t))
```

```
Performance Report:
Accuracy score is : 0.503
Precision score is : 0.253
Recall score is : 0.503
F1 Score is : 0.3367
Cohen Kappa Score: 0.0
                Classification Report:
               precision    recall  f1-score   support

           0       0.50      1.00      0.67       166
           1       0.00      0.00      0.00        30
           2       0.00      0.00      0.00        96
           3       0.00      0.00      0.00        15
           4       0.00      0.00      0.00        23

    accuracy                           0.50       330
   macro avg       0.10      0.20      0.13       330
weighted avg       0.25      0.50      0.34       330
```

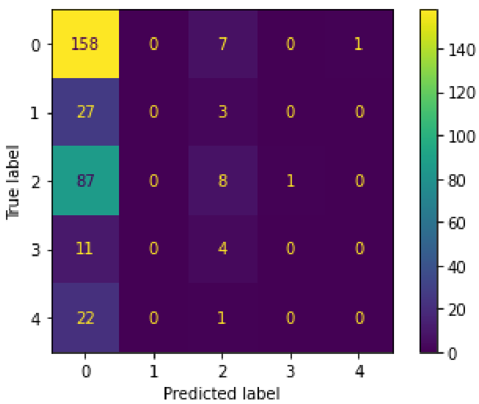
```
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarnin
g: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instea
d:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g.
if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWarnin
g: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.pr
edict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_d
ivision` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1245: UndefinedMetricWarn
ing: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted sample
s. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [141]:

```
knn = KNeighborsClassifier(n_neighbors = 5, algorithm='ball_tree', leaf_size=30)
knn.fit(train_features, y_train)
plot_confusion_matrix(knn, test_features, y_test)
```

Out[141]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e385bbd10>
```

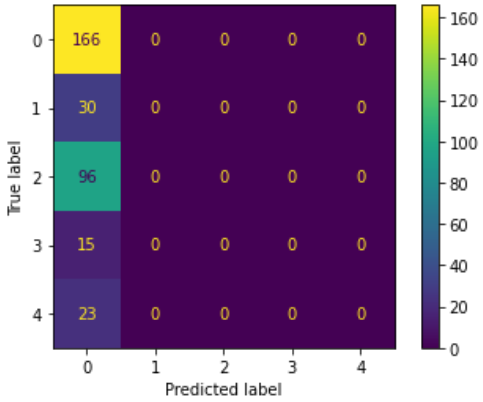


In [142]:

```
svc = SVC()
svc.fit(train_features, y_train)
plot_confusion_matrix(svc, test_features, y_test)
```

Out[142]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e4220482490>
```

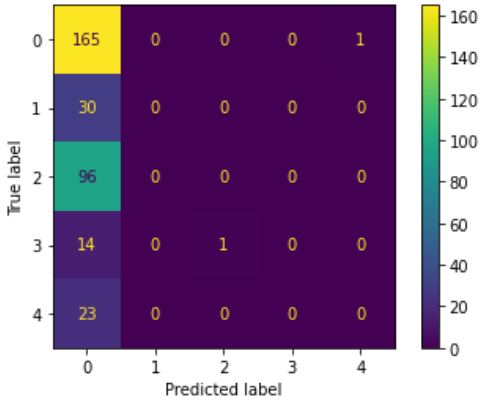


In [143]:

```
rf = RandomForestClassifier()
rf.fit(train_features, y_train)
plot_confusion_matrix(rf, test_features, y_test)
```

Out[143]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e32c0cb50>
```

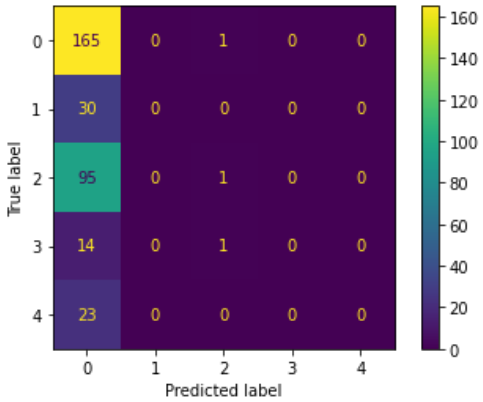


In [144]:

```
ada = AdaBoostClassifier()
ada.fit(train_features, y_train)
plot_confusion_matrix(ada, test_features, y_test)
```

Out[144]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e32bb6250>
```



In [145]:

```
xgbc = XGBClassifier()
xgbc.fit(train_features, y_train)
plot_confusion_matrix(xgbc, test_features, y_test)
```

```
/opt/conda/lib/python3.7/site-packages/xgboost/sklearn.py:888: UserWarning: The use of label encode
r in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, d
o the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[16:45:40] WARNING: ../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metri
c used with the objective 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set
eval_metric if you'd like to restore the old behavior.
```

Out[145]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7e3e2e391b90>
```