PyCaret Using Google Drive

```
# installations
!pip install -U tensorflow-gpu==2.0.0 grpcio
!pip install pycaret
!pip install -U -q PyDrive
```

```
# imports
import numpy as np
import pandas as pd
from pycaret.classification import *

# Code to read csv file into Colaboratory:
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials
```

```
# Authenticate and create the PyDrive client.
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)
```

Generate Data

```
# Generating Dataframe for taxonomic level MANUAL
link = "https://drive.google.com/file/d/1fD6TGo_j29WKz6PI8PV4kCbjzT-MDMcS/view?usp=s

# to get the id part of the file
id = link.split("/")[-2]

downloaded = drive.CreateFile({'id':id})
downloaded.GetContentFile("training.csv")

training_df = pd.read_csv('training.csv')
#df = df.drop(columns = 'Unnamed: 0')
print(training_df)
```

```
# Generating Dataframe for COVID-19 Sequences
testing_link = "https://drive.google.com/file/d/1_SxcTlA9dDIergs__seb-DbnifluBQF6/vi
```

```
sublevel = input("Sublevel of Testing Data: ")
# to get the id part of the file
id = testing_link.split("/")[-2]

downloaded = drive.CreateFile({'id':id})
downloaded.GetContentFile('testing.csv')

testing_df = pd.read_csv('testing.csv')
testing_df = testing_df.drop(columns = 'Unnamed: 0')
testing_df = testing_df[testing_df['Sublevel Name'] == sublevel]
print(testing_df)
```

Magtropy

```
magtropy_df = training_df.drop(columns = ["pp_avg_magnitude", "entropy"])
print(magtropy_df)
```

```
experiment = setup(data=magtropy_df, target='Sublevel Name')
# if the error states target is not defined, change from Sublevel_Name to Sublevel N
# label encodings alphabetical
```

	Description	Value
0	session_id	7766
1	Target	Sublevel Name
2	Target Type	Multiclass
3	Label Encoded	Duplodnaviria: 0, Monodnaviria: 1, Riboviria: ...
4	Original Data	(400, 3)
5	Missing Values	False
6	Numeric Features	2
7	Categorical Features	0
8	Ordinal Features	False
9	High Cardinality Features	False
10	High Cardinality Method	None
11	Transformed Train Set	(279, 1)
12	Transformed Test Set	(121, 1)
13	Shuffle Train-Test	True
14	Stratify Train-Test	False
15	Fold Generator	StratifiedKFold
16	Fold Number	10
17	CPU Jobs	-1
18	Use GPU	False
19	Log Experiment	False
20	Experiment Name	clf-default-name
21	USI	c866
22	Imputation Type	simple
23	Iterative Imputation Iteration	None
24	Numeric Imputer	mean
25	Iterative Imputation Numeric Model	None
26	Categorical Imputer	constant
27	Iterative Imputation Categorical Model	None
28	Unknown Categoricals Handling	least_frequent
29	Normalize	False
30	Normalize Method	None

30	Normalize Method	None
31	Transformation	False
32	Transformation Method	None
33	PCA	False
34	PCA Method	None
35	PCA Components	None
36	Ignore Low Variance	False
37	Combine Rare Levels	False
38	Rare Level Threshold	None
39	Numeric Binning	False
40	Remove Outliers	False
41	Outliers Threshold	None

```
compare_models()
```

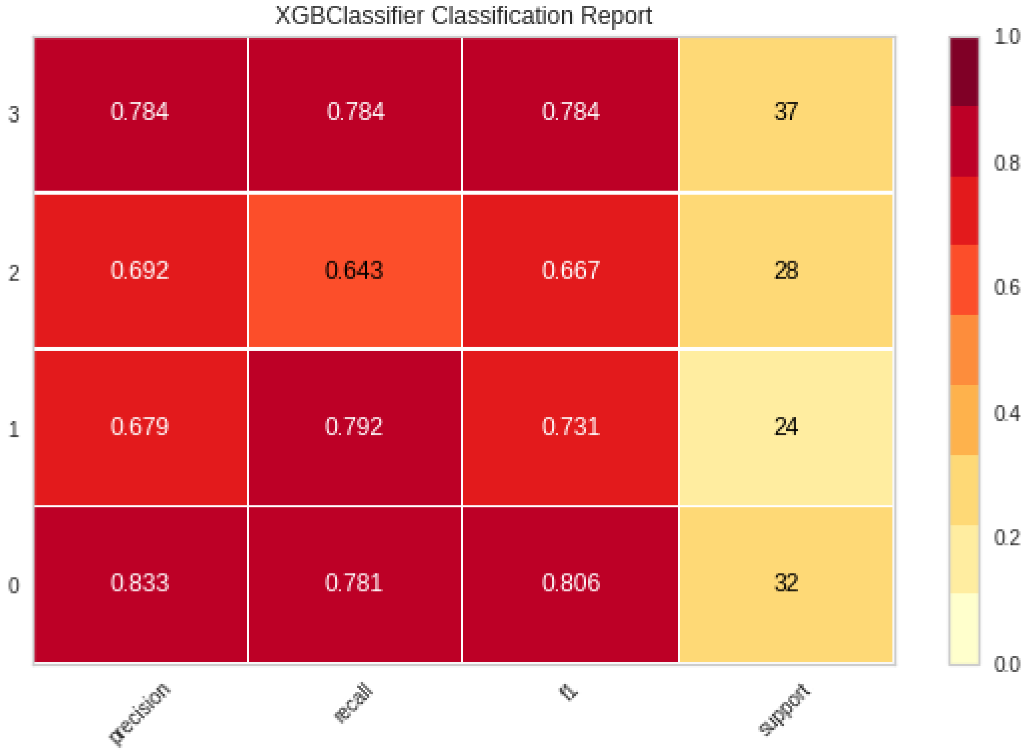
	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
xgboost	Extreme Gradient Boosting	0.7955	0.9565	0.7909	0.8168	0.7941	0.7267	0.7331	0.642

```
estimator = create_model('xgboost')
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.8571	0.9766	0.8512	0.8696	0.8554	0.8095	0.8151
1	0.8214	0.9617	0.8095	0.8254	0.8211	0.7607	0.7620
2	0.7143	0.9728	0.7143	0.7758	0.6986	0.6190	0.6412
3	0.7500	0.9661	0.7366	0.7579	0.7437	0.6638	0.6696
4	0.7857	0.9499	0.7723	0.7893	0.7835	0.7133	0.7158
5	0.8214	0.9368	0.8155	0.8631	0.8292	0.7623	0.7715
6	0.7500	0.9186	0.7485	0.7847	0.7487	0.6661	0.6742
7	0.7857	0.9569	0.7887	0.7991	0.7903	0.7133	0.7145
8	0.9286	0.9862	0.9286	0.9376	0.9250	0.9044	0.9091
9	0.7407	0.9392	0.7440	0.7652	0.7450	0.6545	0.6581
Mean	0.7955	0.9565	0.7909	0.8168	0.7941	0.7267	0.7331
SD	0.0606	0.0196	0.0606	0.0543	0.0625	0.0810	0.0789

```
plot_model(estimator, 'confusion_matrix')
```

```
plot_model(estimator, 'class_report')
```



```
magtropy_testing_df = testing_df.drop(columns = ["pp_avg_magnitude", "entropy"])
print(magtropy_testing_df)
```

```
      Sublevel Name  pp_magtropy
112    Embecovirus   114.269624
113    Embecovirus   114.111031
114    Embecovirus   114.987320
115    Embecovirus   114.226726
116    Embecovirus   114.320187
..             ...          ...
207    Embecovirus   112.497193
208    Embecovirus   114.288491
209    Embecovirus   114.870606
210    Embecovirus   115.440977
211    Embecovirus   114.422743

[100 rows x 2 columns]
```

```
X_test = magtropy_testing_df.drop(columns = ["Sublevel Name"])
predict = estimator.predict(X_test)
print(predict)
print(len(predict))
```

```
[3 3 0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 0 3 0 3 3 3 3 3 3 3 3 3
 0 3 3 0 0 3 0 3 3 3 3 3 3 3 3 3 3 0 3 3 3 3 3 0 0 3 0 3 0 3 0 3 3 0 3 3 3
 3 3 3 0 0 3 3 3 0 0 3 3 3 3 0 0 0 3 3 3 3 0 3 0 0 3]
100
```

```
unique_elements, count_elements = np.unique(predict, return_counts = "True")
results = np.asarray((unique_elements, count_elements))
print(results)
```

```
[[ 0  3]
 [24 76]]
```

Magnitude avg

```
avg_magnitude_df = training_df.drop(columns = ["pp_magtropy", "entropy"])
print(avg_magnitude_df)
```

```
     Unnamed: 0  Sublevel Name  pp_avg_magnitude
0             0  Duplodnaviria        151.202449
1             1  Duplodnaviria        357.998334
2             2  Duplodnaviria        168.981876
3             3  Duplodnaviria        170.966669
4             4  Duplodnaviria        177.257002
..          ...            ...               ...
395         395   Varidnaviria        356.886186
396         396   Varidnaviria        322.779124
397         397   Varidnaviria        165.974924
398         398   Varidnaviria        164.620626
399         399   Varidnaviria        378.774070

[400 rows x 3 columns]
```

```
experiment = setup(data=avg_magnitude_df, target='Sublevel Name')
```

	Description	Value
0	session_id	734
1	Target	Sublevel Name
2	Target Type	Multiclass
3	Label Encoded	Duplodnaviria: 0, Monodnaviria: 1, Riboviria: ...
4	Original Data	(400, 3)
5	Missing Values	False
6	Numeric Features	2
7	Categorical Features	0
8	Ordinal Features	False
9	High Cardinality Features	False
10	High Cardinality Method	None
11	Transformed Train Set	(279, 1)
12	Transformed Test Set	(121, 1)
13	Shuffle Train-Test	True
14	Stratify Train-Test	False
15	Fold Generator	StratifiedKFold
16	Fold Number	10
17	CPU Jobs	-1
18	Use GPU	False
19	Log Experiment	False
20	Experiment Name	clf-default-name
21	USI	9127
22	Imputation Type	simple
23	Iterative Imputation Iteration	None
24	Numeric Imputer	mean
25	Iterative Imputation Numeric Model	None
26	Categorical Imputer	constant
27	Iterative Imputation Categorical Model	None
28	Unknown Categoricals Handling	least_frequent
29	Normalize	False
30	Normalize Method	None

30	Normalize Method	None
31	Transformation	False
32	Transformation Method	None
33	PCA	False
34	PCA Method	None
35	PCA Components	None
36	Ignore Low Variance	False
37	Combine Rare Levels	False
38	Rare Level Threshold	None
39	Numeric Binning	False
40	Remove Outliers	False
41	Outliers Threshold	None
42	Remove Multicollinearity	False
43	Multicollinearity Threshold	None
44	Clustering	False
45	Clustering Iteration	None
46	Polynomial Features	False
47	Polynomial Degree	None
48	Trignometry Features	False

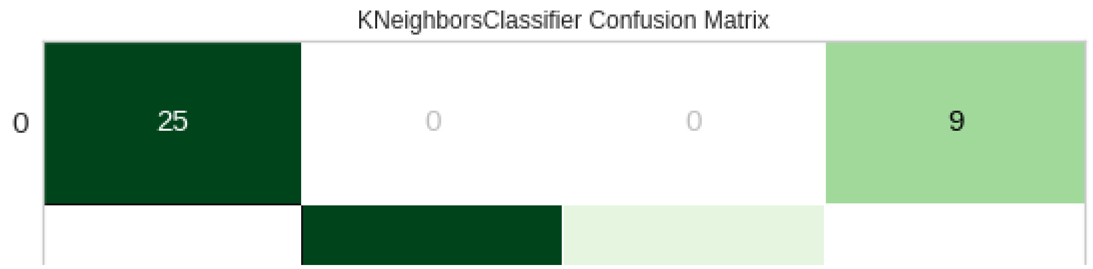
```
compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
knn	K Neighbors Classifier	0.7598	0.9311	0.7613	0.7740	0.7539	0.6792	0.6866	0.122
catboost	CatBoost Classifier	0.7382	0.9389	0.7399	0.7646	0.7283	0.6506	0.6628	1.136
et	Extra Trees Classifier	0.7238	0.8794	0.7251	0.7530	0.7139	0.6313	0.6438	0.470

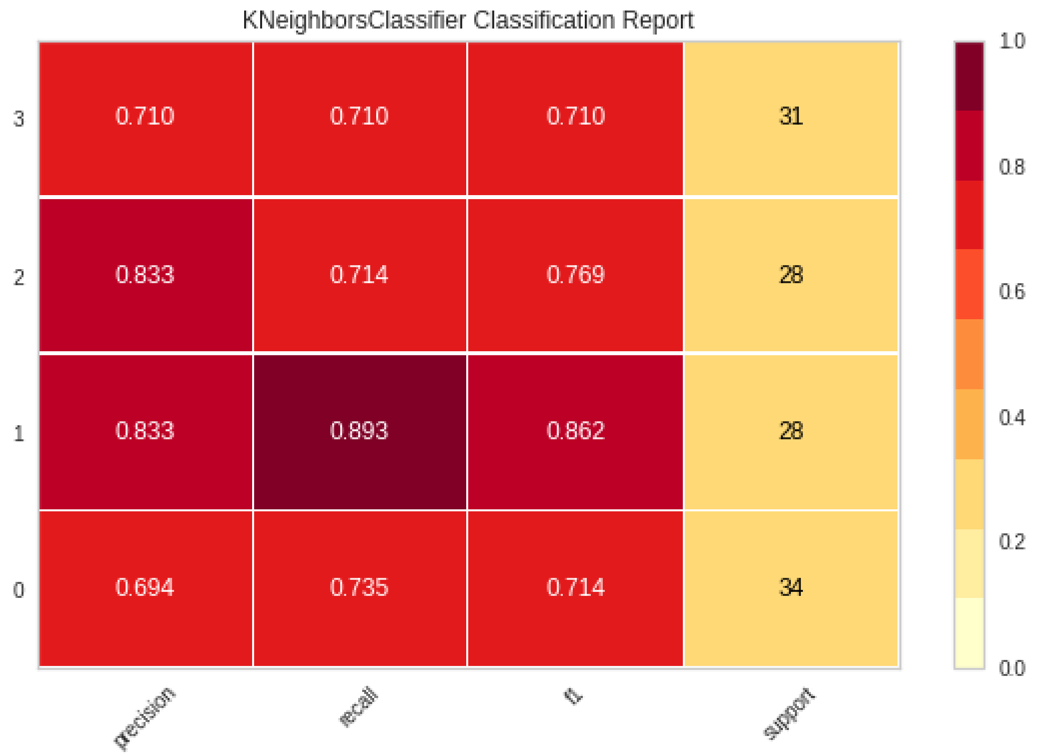
```
estimator = create_model('knn')
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7857	0.9559	0.7991	0.7836	0.7806	0.7143	0.7167
1	0.7500	0.9532	0.7500	0.7864	0.7407	0.6632	0.6811
2	0.8571	0.9462	0.8557	0.8661	0.8591	0.8085	0.8099
3	0.6071	0.9022	0.6071	0.6417	0.5902	0.4762	0.4924
4	0.8929	0.9813	0.8929	0.8958	0.8923	0.8571	0.8586
5	0.6786	0.9379	0.6786	0.6896	0.6747	0.5714	0.5764
6	0.7143	0.8776	0.7143	0.7202	0.7024	0.6190	0.6266
7	0.8571	0.9082	0.8571	0.8562	0.8500	0.8095	0.8137
8	0.7143	0.9337	0.7143	0.7333	0.7051	0.6190	0.6299
9	0.7407	0.9149	0.7440	0.7667	0.7440	0.6532	0.6605
Mean	0.7598	0.9311	0.7613	0.7740	0.7539	0.6792	0.6866
SD	0.0848	0.0290	0.0850	0.0772	0.0886	0.1130	0.1092

```
plot_model(estimator, 'confusion_matrix')
```



```
plot_model(estimator, 'class_report')
```



```
magnitude_avg_testing_df = testing_df.drop(columns = ["pp_magtropy", "entropy"])
print(magnitude_avg_testing_df )
```

```
    Sublevel Name  pp_avg_magnitude
112   Embecovirus        153.103733
113   Embecovirus        155.141480
114   Embecovirus        153.815693
115   Embecovirus        153.062393
116   Embecovirus        153.136267
..            ...               ...
207   Embecovirus        153.807531
208   Embecovirus        153.117355
209   Embecovirus        153.996769
210   Embecovirus        150.518479
211   Embecovirus        153.317131

[100 rows x 2 columns]
```

```
X_test = magnitude_avg_testing_df.drop(columns = ["Sublevel Name"])
predict = estimator.predict(X_test)
print(predict)
print(predict)
print(len(predict))
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
100
```

```
unique_elements, count_elements = np.unique(predict, return_counts = "True")
results = np.asarray((unique_elements, count_elements))
print(results)
```

```
[[  0]
 [100]]
```

Entropy

```
entropy_df = training_df.drop(columns = ["pp_magtropy","pp_avg_magnitude"])
print(entropy_df)
```

```
     Unnamed: 0  Sublevel Name   entropy
0             0  Duplodnaviria  1.345202
1             1  Duplodnaviria  1.335980
2             2  Duplodnaviria  1.345792
3             3  Duplodnaviria  1.362089
4             4  Duplodnaviria  1.369623
..          ...            ...       ...
395         395   Varidnaviria  1.336851
396         396   Varidnaviria  1.343654
397         397   Varidnaviria  1.385762
398         398   Varidnaviria  1.381696
399         399   Varidnaviria  1.380763

[400 rows x 3 columns]
```

```
experiment = setup(data=entropy_df, target='Sublevel Name')
```

	Description	Value
0	session_id	2580
1	Target	Sublevel Name
2	Target Type	Multiclass
3	Label Encoded	Duplodnaviria: 0, Monodnaviria: 1, Riboviria: ...
4	Original Data	(400, 3)
5	Missing Values	False
6	Numeric Features	2
7	Categorical Features	0
8	Ordinal Features	False
9	High Cardinality Features	False
10	High Cardinality Method	None
11	Transformed Train Set	(279, 1)
12	Transformed Test Set	(121, 1)
13	Shuffle Train-Test	True
14	Stratify Train-Test	False
15	Fold Generator	StratifiedKFold
16	Fold Number	10
17	CPU Jobs	-1
18	Use GPU	False
19	Log Experiment	False
20	Experiment Name	clf-default-name
21	USI	09f1
22	Imputation Type	simple
23	Iterative Imputation Iteration	None
24	Numeric Imputer	mean
25	Iterative Imputation Numeric Model	None
26	Categorical Imputer	constant
27	Iterative Imputation Categorical Model	None
28	Unknown Categoricals Handling	least_frequent
29	Normalize	False
30	Normalize Method	None

30	Normalize Method	None
31	Transformation	False
32	Transformation Method	None
33	PCA	False
34	PCA Method	None
35	PCA Components	None
36	Ignore Low Variance	False
37	Combine Rare Levels	False
38	Rare Level Threshold	None
39	Numeric Binning	False
40	Remove Outliers	False
41	Outliers Threshold	None
42	Remove Multicollinearity	False
43	Multicollinearity Threshold	None
44	Clustering	False
45	Clustering Iteration	None
46	Polynomial Features	False
47	Polynomial Degree	None
48	Trignometry Features	False
49	Polynomial Threshold	None

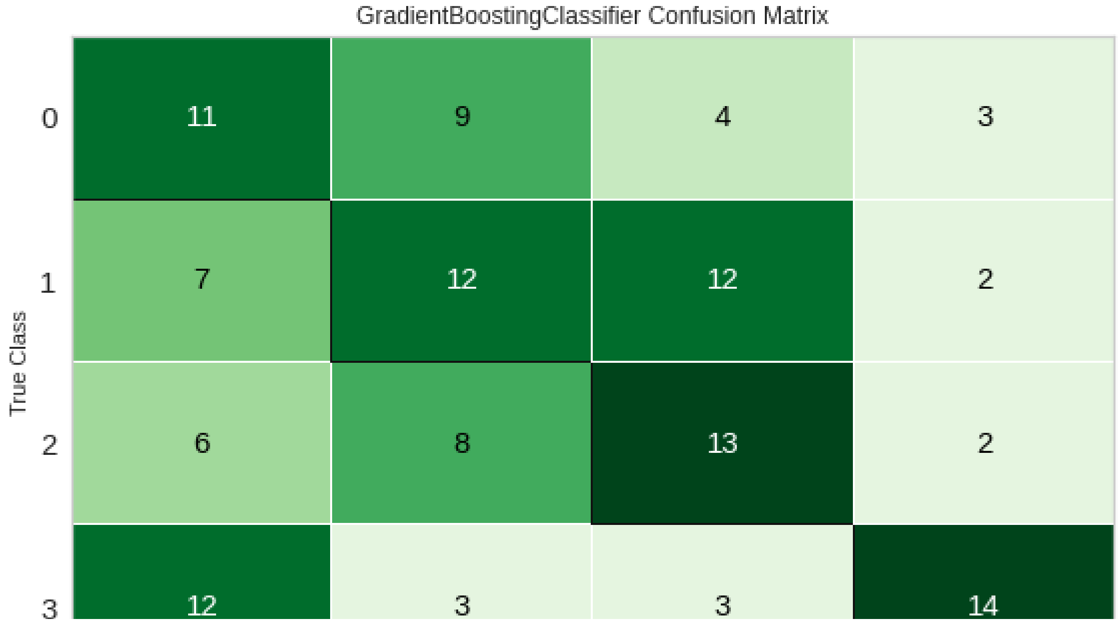
```
compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
gbc	Gradient Boosting Classifier	0.4226	0.6785	0.4193	0.4545	0.4214	0.2303	0.2340	0.300
dt	Decision Tree Classifier	0.4155	0.6103	0.4132	0.4352	0.4106	0.2204	0.2247	0.023
rf	Random Forest Classifier	0.4155	0.6513	0.4132	0.4352	0.4106	0.2204	0.2247	0.525
et	Extra Trees Classifier	0.4048	0.6301	0.4034	0.4250	0.4007	0.2064	0.2104	0.478
xgboost	Extreme Gradient Boosting	0.3975	0.6534	0.3948	0.4179	0.3935	0.1963	0.2003	1.189
catboost	CatBoost Classifier	0.3762	0.6749	0.3735	0.4088	0.3732	0.1679	0.1709	1.153

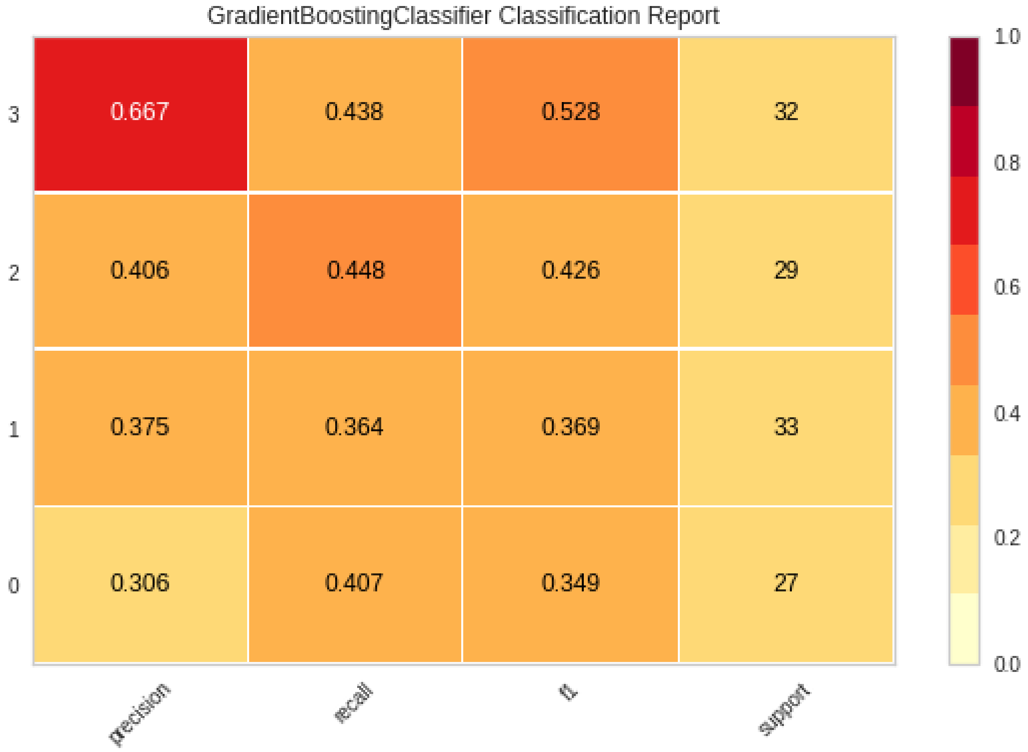
```
estimator = create_model('gbc')
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.5714	0.7693	0.5521	0.5882	0.5663	0.4167	0.4235
1	0.1429	0.4894	0.1399	0.2347	0.1577	-0.1237	-0.1346
2	0.3214	0.6383	0.3051	0.3108	0.3141	0.0922	0.0926
3	0.5357	0.7738	0.5357	0.5325	0.5298	0.3810	0.3836
4	0.5000	0.7245	0.5000	0.5387	0.5063	0.3333	0.3374
5	0.3571	0.6446	0.3571	0.3786	0.3577	0.1429	0.1451
6	0.4643	0.7500	0.4643	0.5119	0.4754	0.2857	0.2892
7	0.5000	0.6939	0.5000	0.6100	0.4833	0.3333	0.3565
8	0.5000	0.7211	0.5000	0.4988	0.4914	0.3333	0.3374
9	0.3333	0.5807	0.3393	0.3407	0.3324	0.1083	0.1095
Mean	0.4226	0.6785	0.4193	0.4545	0.4214	0.2303	0.2340
SD	0.1247	0.0864	0.1242	0.1217	0.1203	0.1610	0.1664

```
plot_model(estimator, 'confusion_matrix')
```



```
plot_model(estimator, 'class_report')
```



```
entropy_testing_df = testing_df.drop(columns = ["pp_avg_magnitude", "pp_magtropy"])
print(entropy_testing_df)
```

```
     Sublevel Name   entropy
112    Embecovirus  1.339846
113    Embecovirus  1.359566
114    Embecovirus  1.337675
115    Embecovirus  1.339988
116    Embecovirus  1.339538
..             ...       ...
207    Embecovirus  1.367212
```

```
   208   Embecovirus  1.339744
   209   Embecovirus  1.340611
   210   Embecovirus  1.303857
   211   Embecovirus  1.339918

   [100 rows x 2 columns]
```

```
X_test =entropy_testing_df.drop(columns = ["Sublevel Name"])
predict = estimator.predict(X_test)
print(predict)
print(len(predict))
```

```
   [0 2 3 0 0 0 0 0 0 0 0 0 3 0 0 0 0 1 0 3 0 0 0 0 0 3 0 0 2 2 0 0 0 0 0 0 0
    0 0 0 2 0 0 3 2 0 0 0 1 0 0 0 0 1 0 1 1 0 1 3 3 0 0 1 0 2 0 0 0 0 0 0 0 0
    3 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 1 0 0 3 0]
   100
```

```
unique_elements, count_elements = np.unique(predict, return_counts = "True")
results = np.asarray((unique_elements, count_elements))
print(results)
```

```
   [[ 0  1  2  3]
    [75  8  7 10]]
```