Prashanth Manja

USC ID: 3073150764

ML- Assignment -4

Attribute Information:

1. variance of Wavelet Transformed image (continuous)
2. skewness of Wavelet Transformed image (continuous)
3. curtosis of Wavelet Transformed image (continuous)
4. entropy of image (continuous)
5. class (integer)

Check for missing data:

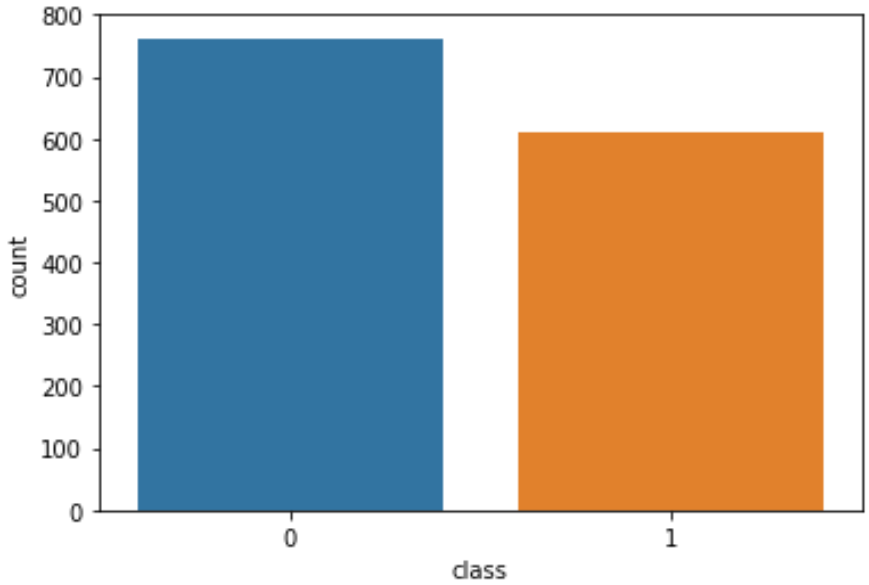
RangeIndex: 1372 entries, 0 to 1371
Data columns (total 5 columns):
variance 1372 non-null float64
skewness 1372 non-null float64
curtosis 1372 non-null float64
entropy 1372 non-null float64
class 1372 non-null int64

So, the dataset has a total of 1372 data points and no missing data points.

Number of features for each class in the training set:

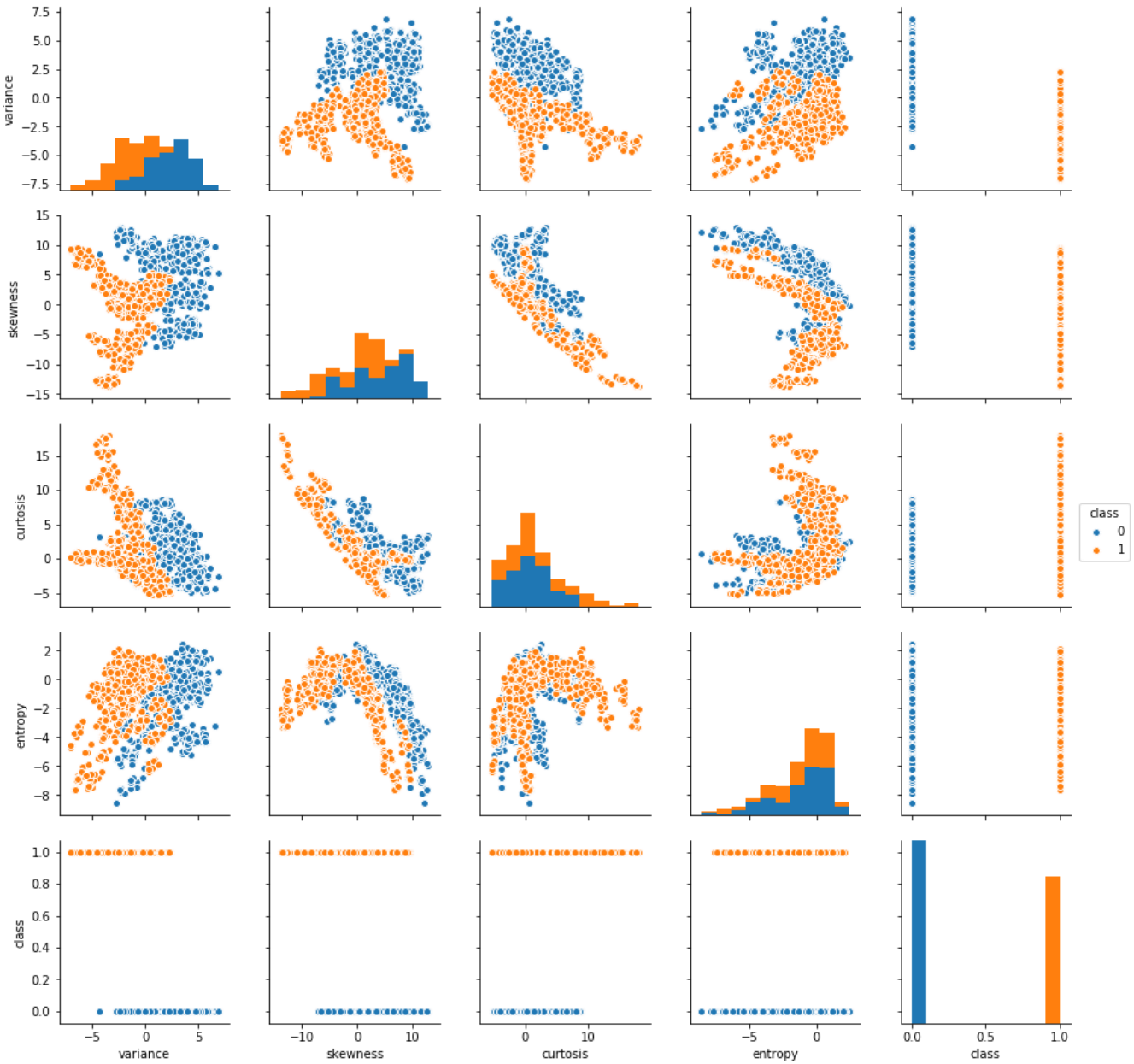
```
0    762
1    610
```

('Number of Original Bank Notes: ' **610**)
('Number of Fake Bank Notes: ' **762**)
('Total number of Notes: ' **1372**)

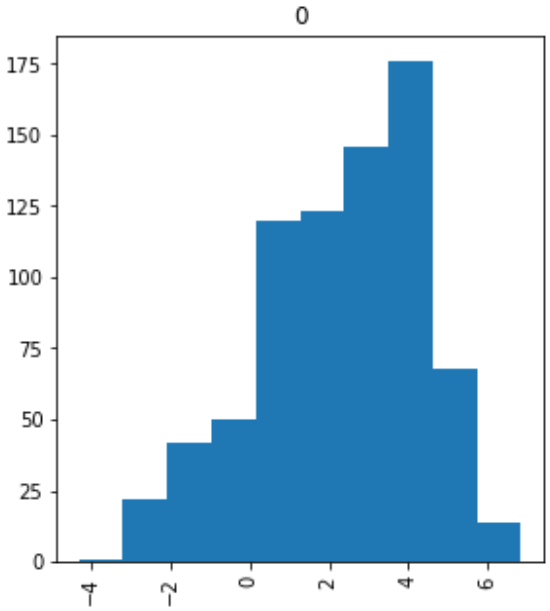


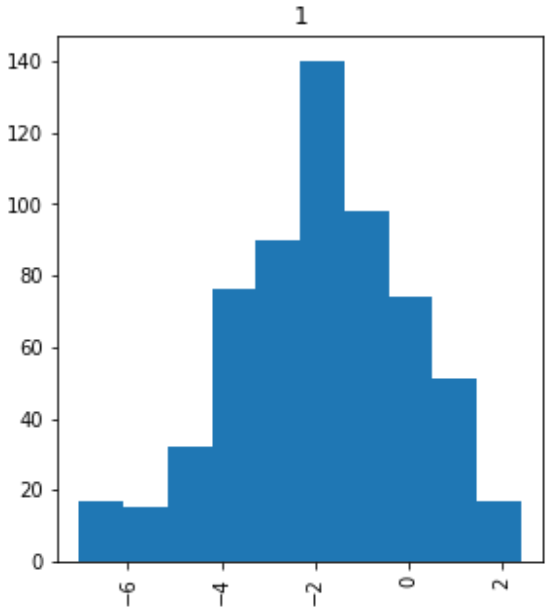
We have an almost equal split of the classes in the training set which is ok for the predicti on task

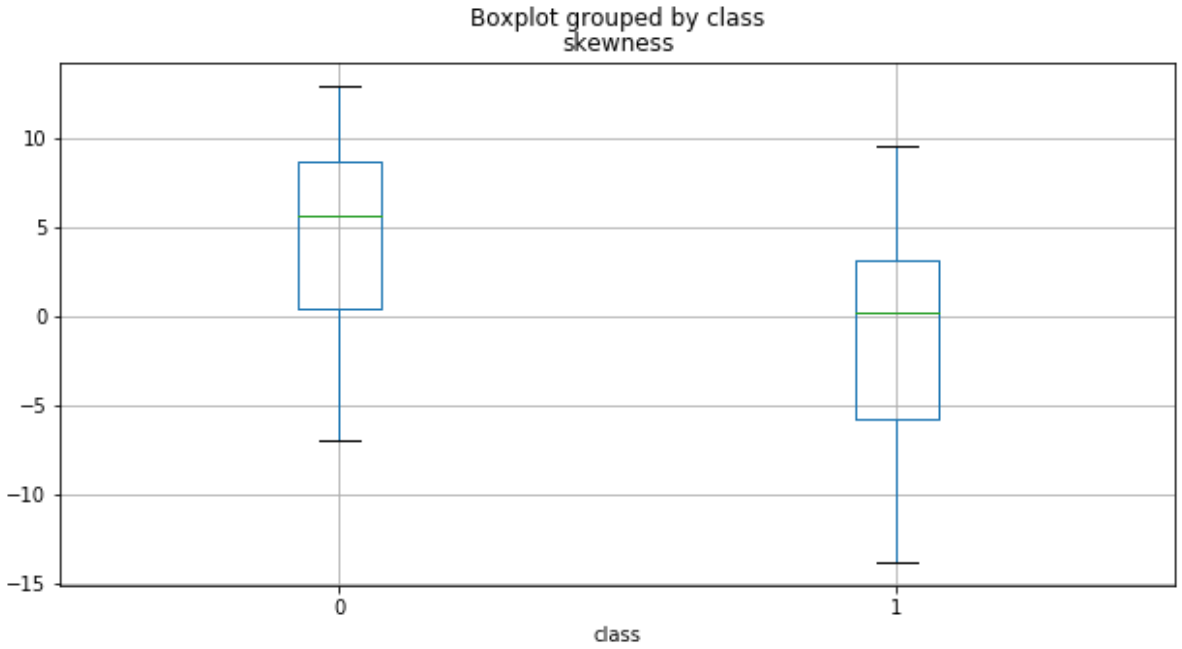
Pairplot :



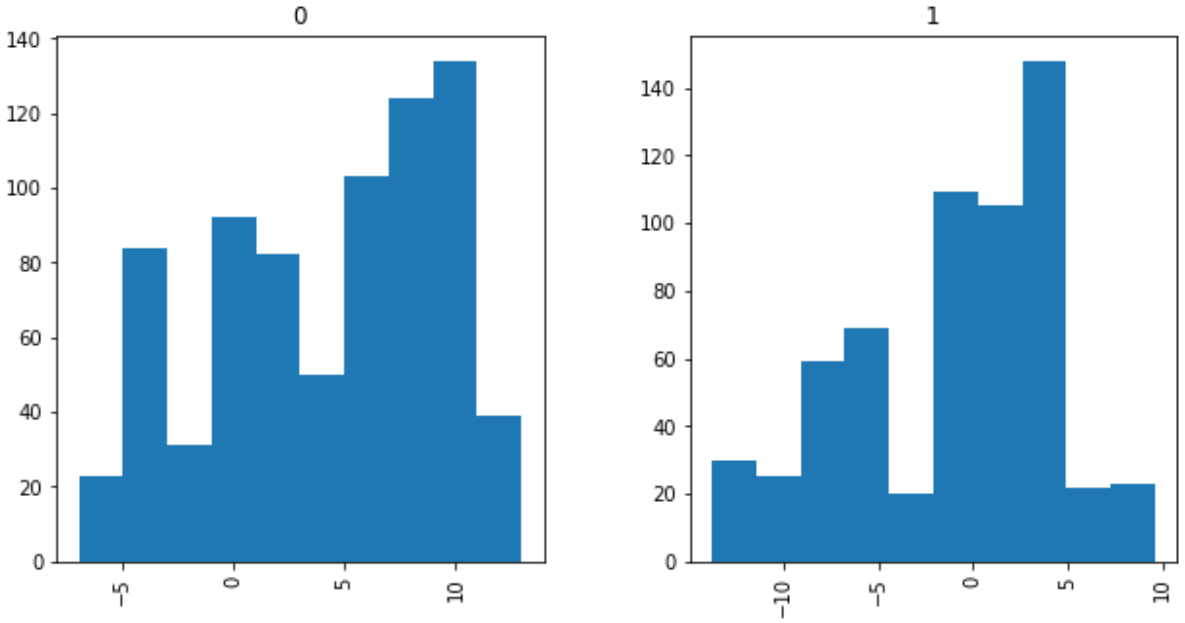
Variance vs Class:

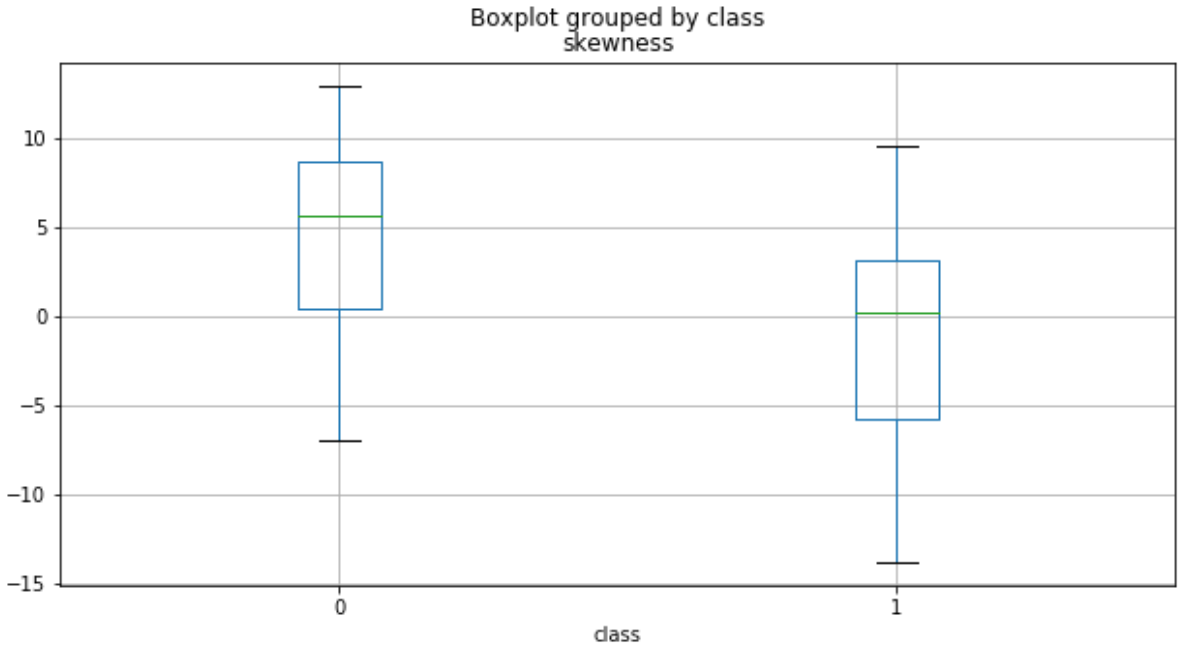




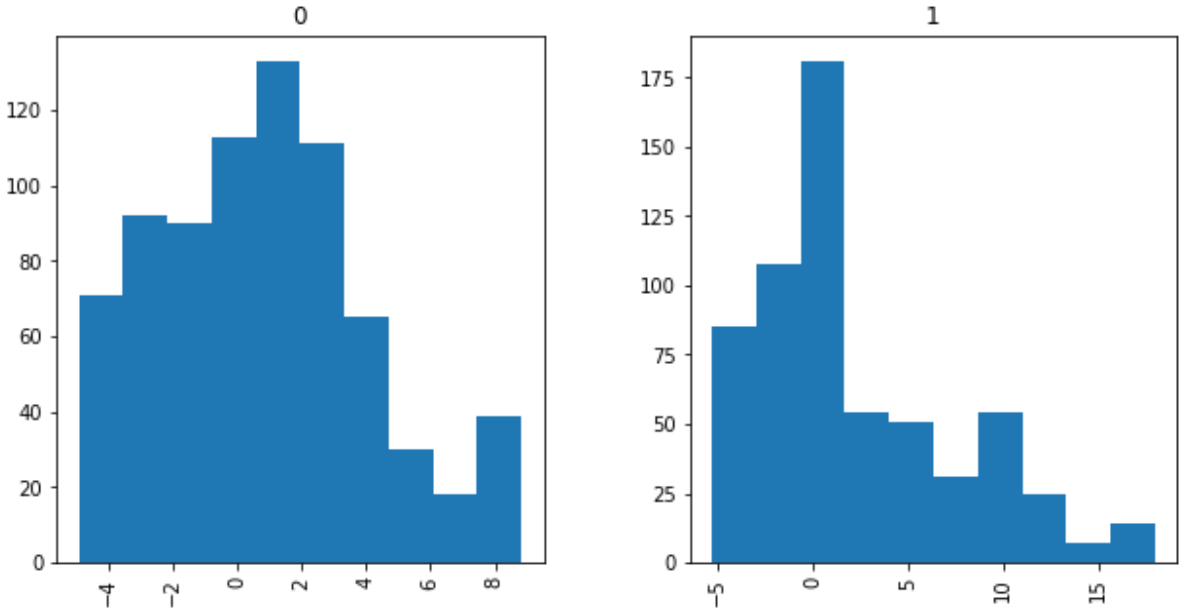


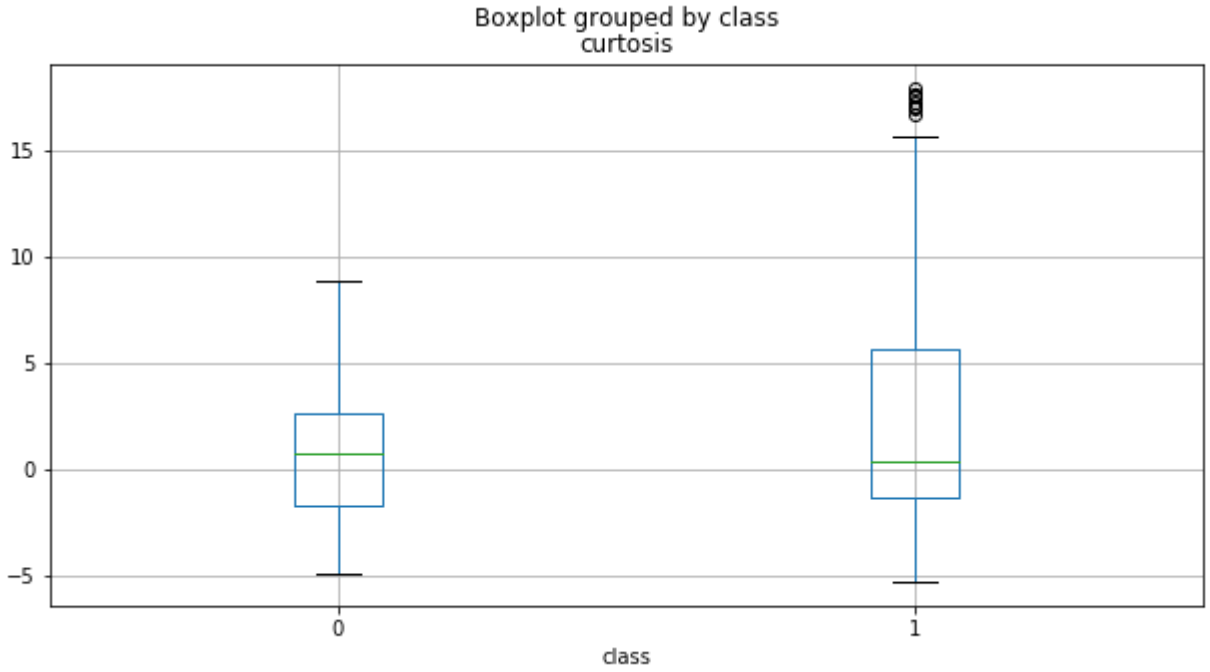
Skewness vs Class:



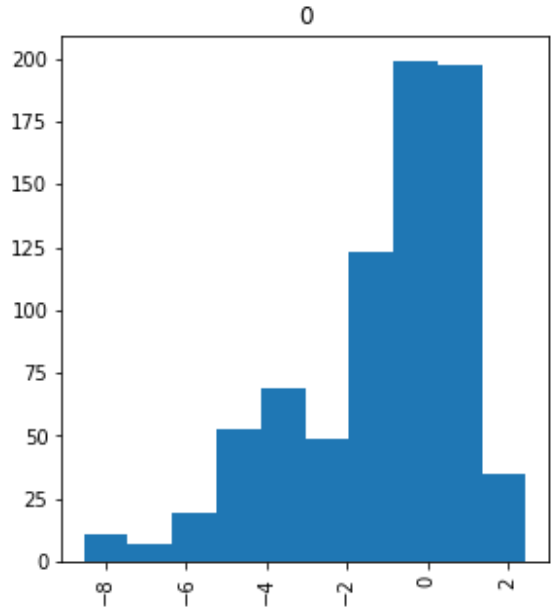


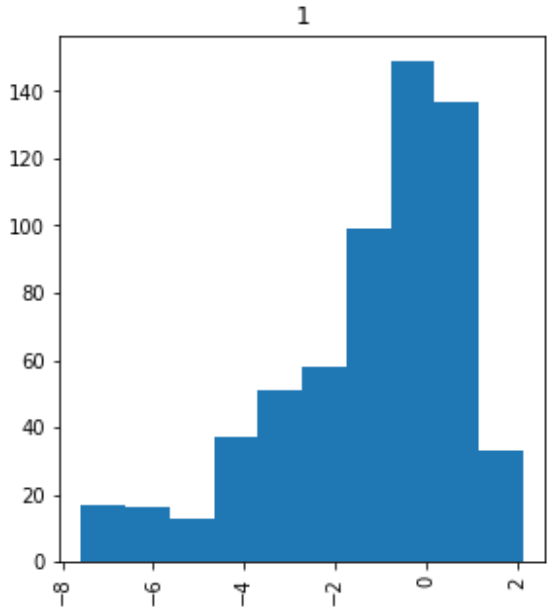
Curtosis vs Class

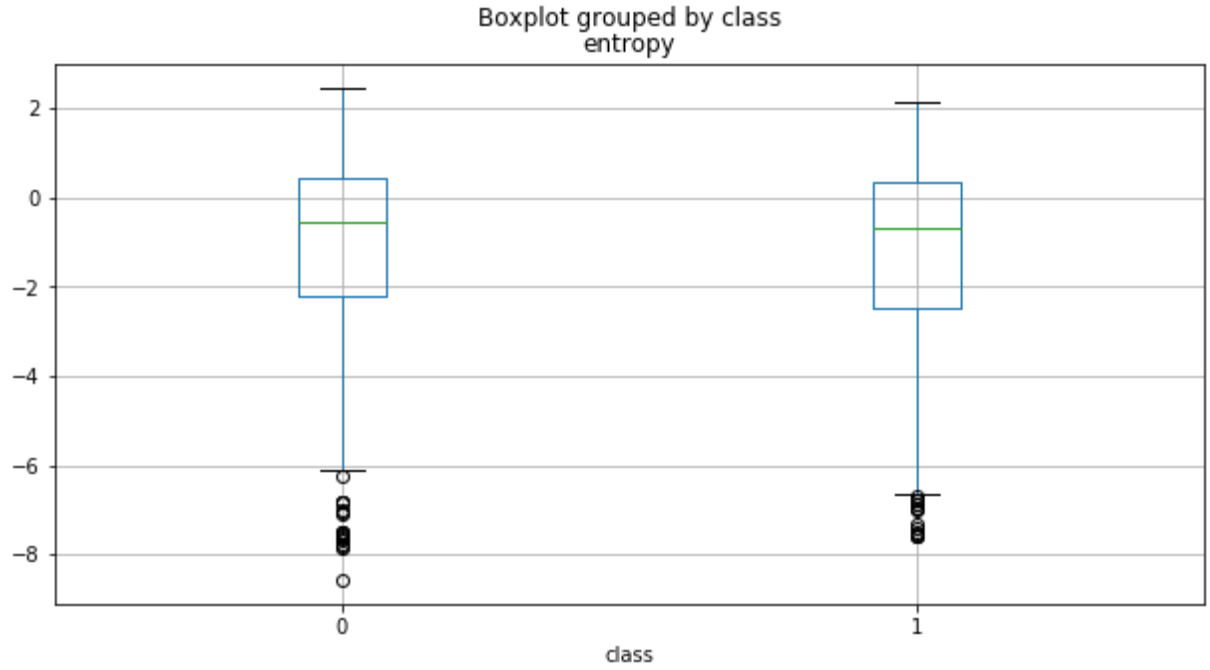




Entropy vs Class:



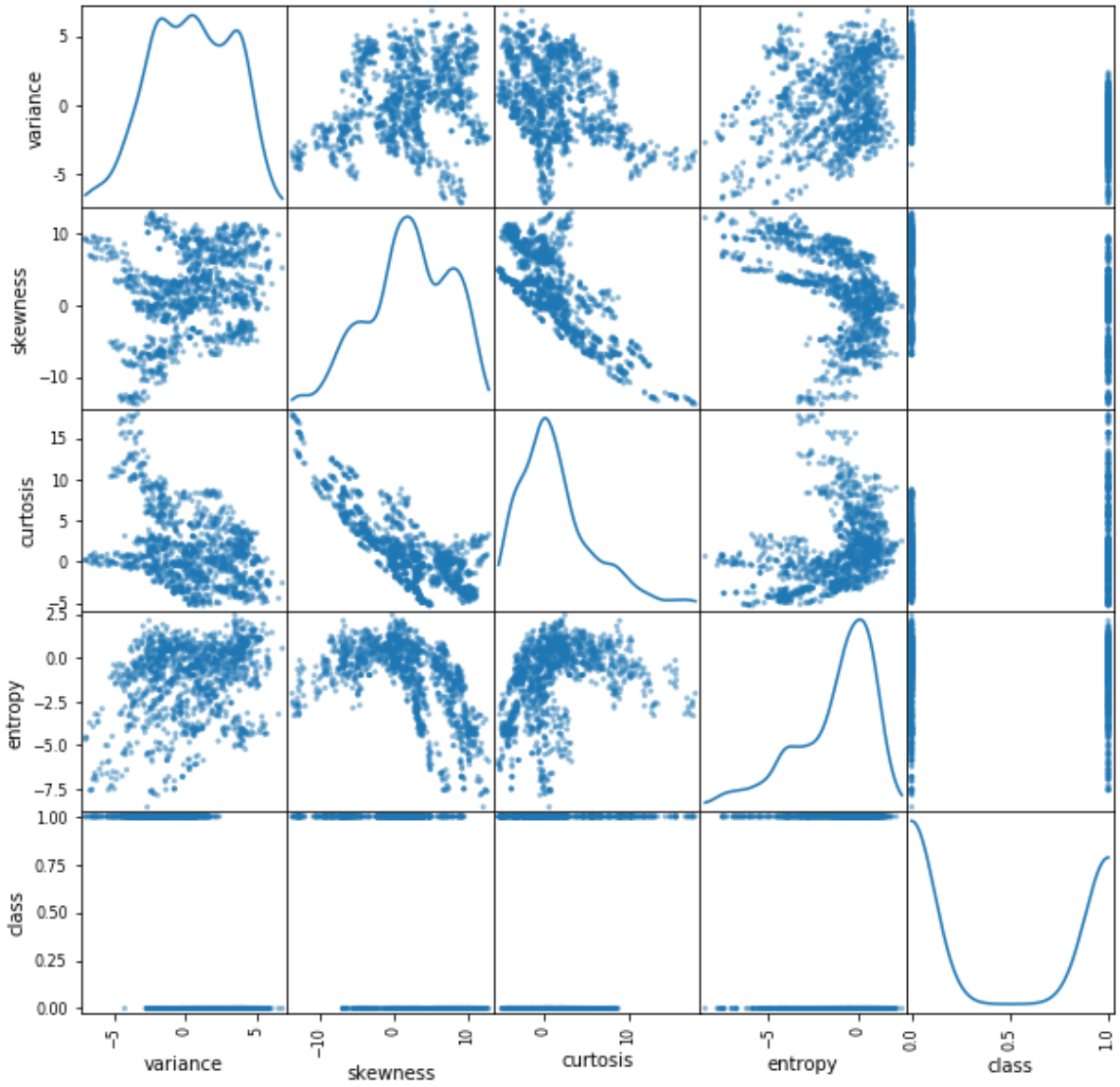




The entropy feature has no differences between then two classes. It has the same mean and median, the same variance and the same distribution. In will not be taken into account for the classification task.

Check the correlation between the features and the label:

	variance	skewness	curtosis	entropy	class
variance	1.000000	0.264026	-0.380850	0.276817	-0.724843
skewness	0.264026	1.000000	-0.786895	-0.526321	-0.444688
curtosis	-0.380850	-0.786895	1.000000	0.318841	0.155883
entropy	0.276817	-0.526321	0.318841	1.000000	-0.023424
class	-0.724843	-0.444688	0.155883	-0.023424	1.000000



Choosing 472 data points randomly as the test set :

	variance	skewness	curtosis	entropy	class
201	4.00260	-3.5943	3.5573	0.26809	0
116	3.89690	7.4163	-1.8245	0.14007	0
1308	-4.63380	-12.7509	16.7166	-3.21680	1
1168	0.74067	1.7299	-3.1963	-0.14570	1
2	3.86600	-2.6383	1.9242	0.10645	0

```
(472, 5)
```

Training Dataset has 900 records

	variance	skewness	curtosis	entropy	class
0	3.62160	8.6661	-2.8073	-0.44699	0
1	4.54590	8.1674	-2.4586	-1.46210	0
3	3.45660	9.5228	-4.0112	-3.59440	0
4	0.32924	-4.4552	4.5718	-0.98880	0
5	4.36840	9.6718	-3.9606	-3.16250	0

```
(900, 5)
```

50 errors for 90 SVMs:

Code:

```
from sklearn.model_selection import KFold
bn_train_new=pd.DataFrame()
test_error_total_passive = []
for i in range(0,50):#50 times
    test_error_passive = []
    best_penality_parameter_passive = []
    for j in range(0,90):#90 times

        bn_train_extracted = bn_train.sample(n=10,replace=False)
        bn_train_new =pd.concat([bn_train_new, bn_train_extracted], axis=0)

        X_train = bn_train_new.drop('class', axis=1)
        Y_train1 = bn_train_new['class']
        Y_train = Y_train1.to_frame();

        x_test = bn_test.drop('class', axis=1)
        y_test1 = bn_test['class']
        y_test = y_test1.to_frame();

        C_range = np.logspace(-2,10,13)
        parameters_grid = dict(C=C_range)
        svc = svm.LinearSVC(penalty='l1',dual=False)
        cv=KFold(10)
        clf = GridSearchCV(svc,param_grid=parameters_grid,cv=cv)
        clf.fit(X_train, Y_train)
        y_pred = clf.predict(x_test)
        score = round(clf.score(x_test, y_test),3)

        best_penality_parameter_passive.append(clf.best_params_)
        test_error_passive.append(score)
    print test_error_passive

    test_error_total_passive.append(test_error_passive)
    #print test_error_a
    best_penality_parameter_passive
print test_error_total_passive
print "BEST PARAMETS ARE: "
print best_penality_parameter_passive
```

Output:

```
print "Passive Leaning 90 test-errors"
print test_error_total_passive

print "BEST PARAMETS ARE: "
print best_penality_parameter_passive
```

```
Passive Leaning 90 test-errors
[[0.883, 0.873, 0.917, 0.949, 0.97, 0.968, 0.962, 0.972, 0.97], [0.97, 0.972, 0.972, 0.972, 0.972, 0.983, 0.983, 0.983, 0.977],
[0.977, 0.985, 0.981, 0.979, 0.979, 0.979, 0.981, 0.981, 0.981], [0.981, 0.981, 0.979, 0.979, 0.979, 0.981, 0.981, 0.979, 0.98
1], [0.979, 0.981, 0.979, 0.983, 0.983, 0.983, 0.983, 0.983, 0.985]]
BEST PARAMETS ARE:
[{'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}]
```

Note : This output is for when the outer loop is 5 and inner loop is 9 (since it is computationally expensive to do it for 90SVMs 50 times)

Explanation:

Here we see '**total_error_total_passive'** is a list of list consists of the test_errors of SVMs run by taking 10,20,30 etc ., Number of samples in each inner list and the number of inner lists counts to 50 i.e for 50runs .

total_error_total_passive = [[test_errors of SVMs run by taking 10][test_errors of SVMs run by taking 20]….[test_errors of SVMs run by taking 900]**]**

&

len(total_error_total_passive) = 50 .

Preparing train data based on the distance from margin:

Here we sort the training data based on the distance from the margin. We then take 10 points from this pool once we fit SVM using 10 randomly chosen points for the first time.

	variance	skewness	curtosis	entropy	class	distance
731	2.01770	1.79820	-2.9581	0.20990	1	0.031
621	-2.90200	-7.65630	11.8318	-0.84268	1	0.207
3	0.32924	-4.45520	4.5718	-0.98880	0	0.275
17	0.32920	-4.45520	4.5718	-0.98880	0	0.275
606	1.56310	0.89599	-1.9702	0.65472	1	0.450

```
(900, 6)
```

Fit SVM using 10 Closest datata points to the Margin:

Code:

```
from sklearn.model_selection import KFold
bn_train_new=pd.DataFrame()
test_error_total_active = []
bn_train_extracted = bn_train_10
for i in range(0,50):#50 times
    test_error_active = []
    best_penality_parameter_active = []
    for j in range(1,90):#90 times

        bn_train_new =pd.concat([bn_train_new, bn_train_extracted], axis=0)

        X_train = bn_train_new.drop('class', axis=1)
        Y_train1 = bn_train_new['class']
        Y_train = Y_train1.to_frame();

        x_test = bn_test.drop('class', axis=1)
        y_test1 = bn_test['class']
        y_test = y_test1.to_frame();

        C_range = np.logspace(-2,10,13)
        parameters_grid = dict(C=C_range)
        svc = svm.LinearSVC(penalty='l1',dual=False)
        cv=KFold(10)
        clf = GridSearchCV(svc,param_grid=parameters_grid,cv=cv)
        clf.fit(X_train, Y_train)
        y_pred = clf.predict(x_test)
        score = round(clf.score(x_test, y_test),3)

        best_penality_parameter_active.append(clf.best_params_)
        test_error_active.append(score)

        bn_train_extracted = bn_train_rem.iloc[:10,:]
        bn_train_rem=bn_train_rem.drop(bn_train_rem.index[:10])

    #print test_error_active
    test_error_total_active.append(test_error_active)
    #print test_error_a

    #print test_error_total_active
print test_error_total_active
print "BEST PARAMETS ARE: "
```

Output:

```
print "Active Leaning 90 test-errors"
print test_error_total_active

print "BEST PARAMETS ARE: "
print best_penality_parameter_active
```

```
Active Leaning 90 test-errors
[[0.867, 0.975, 0.972, 0.979, 0.977, 0.977, 0.979, 0.977, 0.979], [0.979, 0.979, 0.983, 0.979, 0.979, 0.979, 0.979, 0.979, 0.98
3], [0.979, 0.979, 0.983, 0.979, 0.979, 0.979, 0.983, 0.983, 0.983], [0.979, 0.979, 0.979, 0.979, 0.979, 0.979, 0.977, 0.979,
0.979], [0.977, 0.977, 0.979, 0.979, 0.979, 0.979, 0.979, 0.979, 0.979]]
BEST PARAMETS ARE:
[{'C': 10.0}, {'C': 10.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}, {'C': 1.0}]
```

Note : This output is for when the outer loop is 5 and inner loop is 9 (since it is computationally expensive to do it for 90SVMs 50 times)

Explanation:

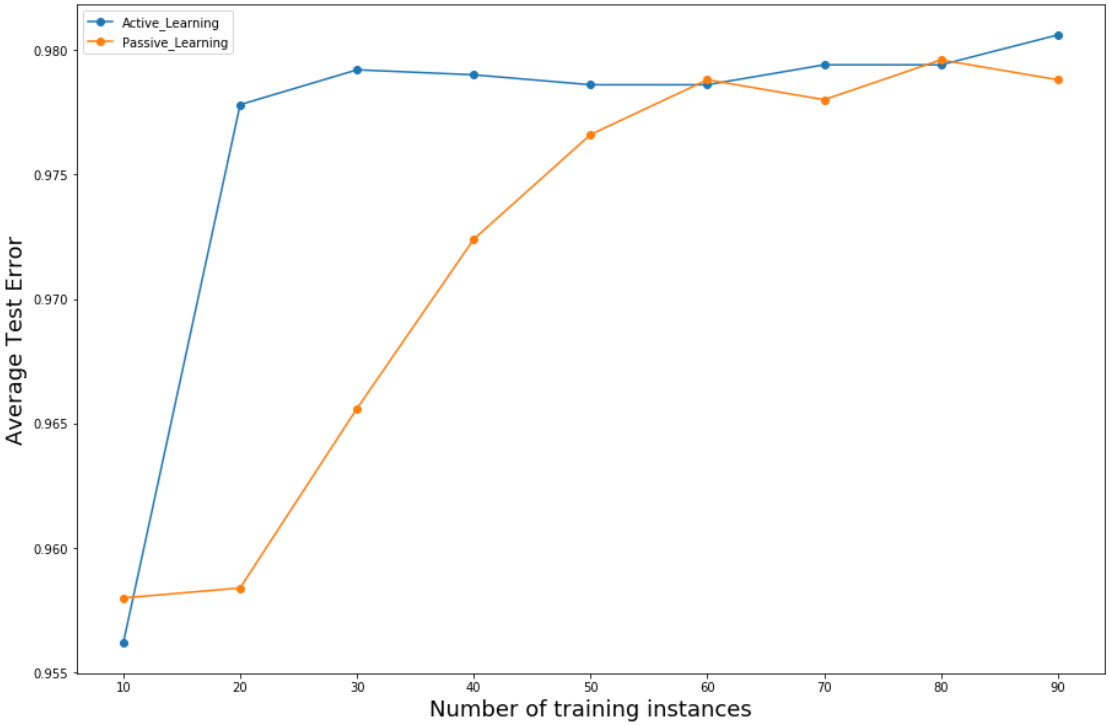
Here we see '**total_error_total_active'** is a list of list consists of the test_errors of SVMs run by taking 10,20,30 etc ., Number of samples in each inner list and the number of inner lists counts to 50 i.e for 50runs .

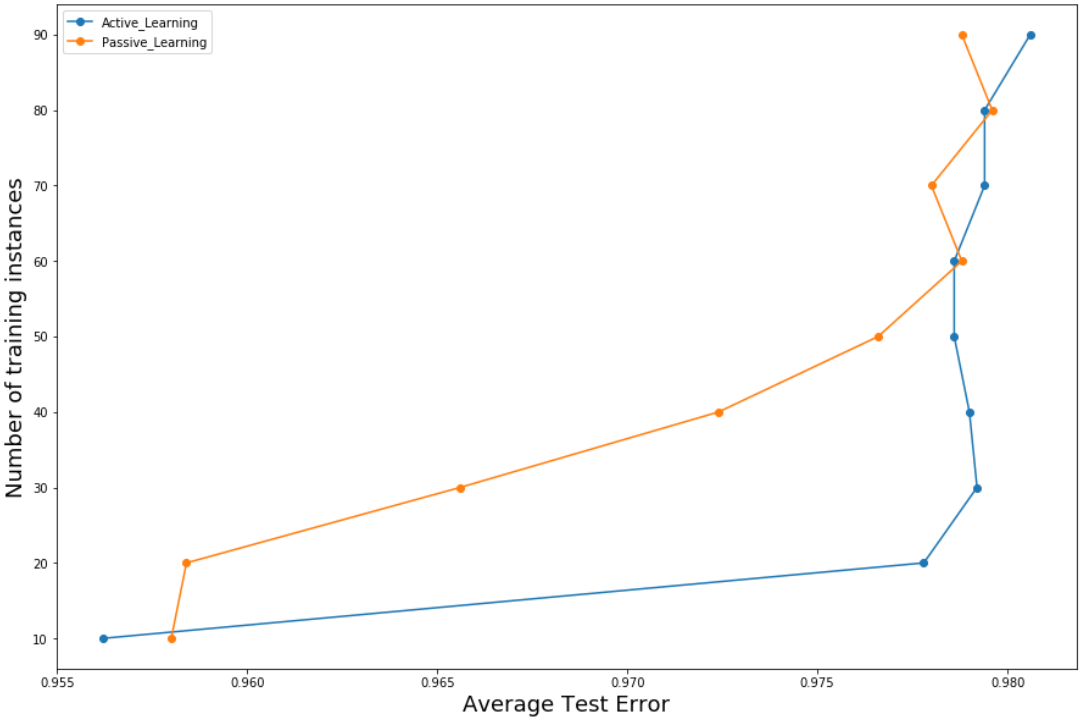
total_error_total_active= [[test_errors of SVMs run by taking 10][test_errors of SVMs run by taking 20]….[test_errors of SVMs run by taking 900]**]**

&

len(total_error_total_active) = 50 .

Monte Carlo Simulation: (results of SVMs taking 10,20…90 instanes)





(same plot as above with x and y axis switched)

As we can see that when the number of training instances are small the Average Test Error for Active Learning is more when compared to that of Passive Learning and as we increase the number of the samples in the training set pool Average test errors for both Active and Passive Learning falls in to approximately same range.

Multi-class Classification Using Support Vector Machines

Anuran Calls (MFCCs) Data Set

This dataset was used in several classifications tasks related to the challenge of anuran species recognition through their calls.

It is a multilabel dataset with three columns of labels. This dataset was created segmenting 60 audio records belonging to 4 different families, 8 genus, and 10 species.

Each audio corresponds to one specimen (an individual frog), the record ID is also included as an extra column.

The amount of instances per class are:

Families:
Bufonidae 68
Dendrobatidae 542
Hylidae 2165
Leptodactylidae 4420

Genus:
Adenomera 4150
Ameerega 542
Dendropsophus 310
Hypsiboas 1593
Leptodactylus 270
Osteocephalus 114
Rhinella 68
Scinax 148

Species:
AdenomeraAndre 672
AdenomeraHylaedactâ€¦ 3478
Ameeregatrivittata 542
HylaMinuta 310
HypsiboasCinerascens 472
HypsiboasCordobae 1121
LeptodactylusFuscus 270
OsteocephalusOophaâ€¦ 114
Rhinellagranulosa 68
ScinaxRuber 148

Each instance has three labels: Families, Genus, and Species. Each of the labels has multiple classes. We wish to solve a multi-class and multi-label problem.

Data Pre-processing:

Checking for missing Data:

1. **X_train:**

```
MFCCs_ 1    5037 non-null float64
MFCCs_ 2    5037 non-null float64
MFCCs_ 3    5037 non-null float64
MFCCs_ 4    5037 non-null float64
MFCCs_ 5    5037 non-null float64
MFCCs_ 6    5037 non-null float64
MFCCs_ 7    5037 non-null float64
MFCCs_ 8    5037 non-null float64
MFCCs_ 9    5037 non-null float64
MFCCs_10    5037 non-null float64
MFCCs_11    5037 non-null float64
MFCCs_12    5037 non-null float64
MFCCs_13    5037 non-null float64
MFCCs_14    5037 non-null float64
MFCCs_15    5037 non-null float64
MFCCs_16    5037 non-null float64
MFCCs_17    5037 non-null float64
MFCCs_18    5037 non-null float64
MFCCs_19    5037 non-null float64
MFCCs_20    5037 non-null float64
MFCCs_21    5037 non-null float64
MFCCs_22    5037 non-null float64
```

2. **Y_train (Family-Label)**

```
Family    5037 non-null object
```

3. **Y_train (Genus-Label)**

```
Genus    5037 non-null object
```

4. **Y_train (Species-Label)**

```
Species    5037 non-null object
```

Count of Each Class in Each of the Label:

1. Family -Label

```
Leptodactylidae    3075
Hylidae            1518
Dendrobatidae       394
Bufonidae            50
```

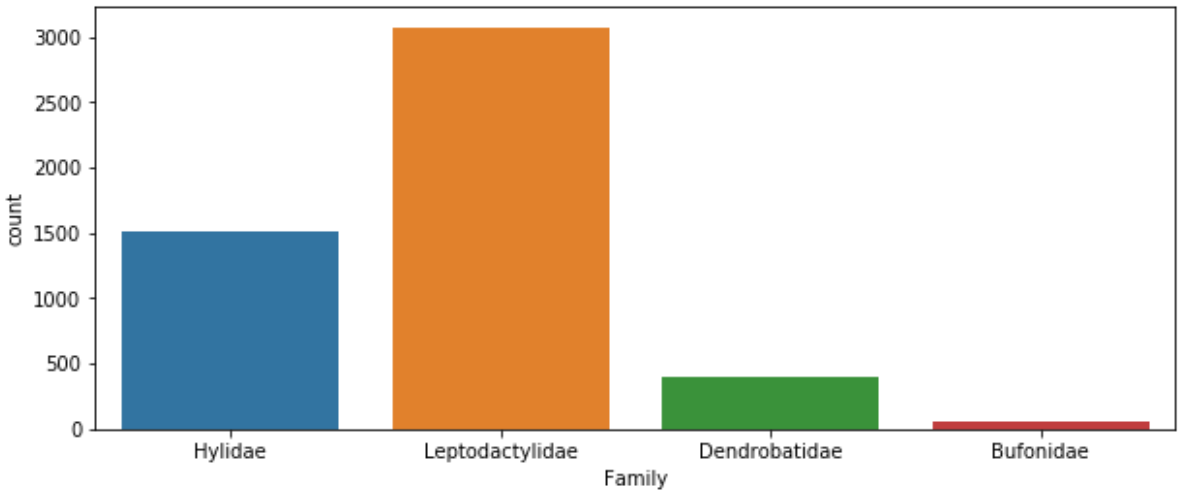
2. Genus-Label

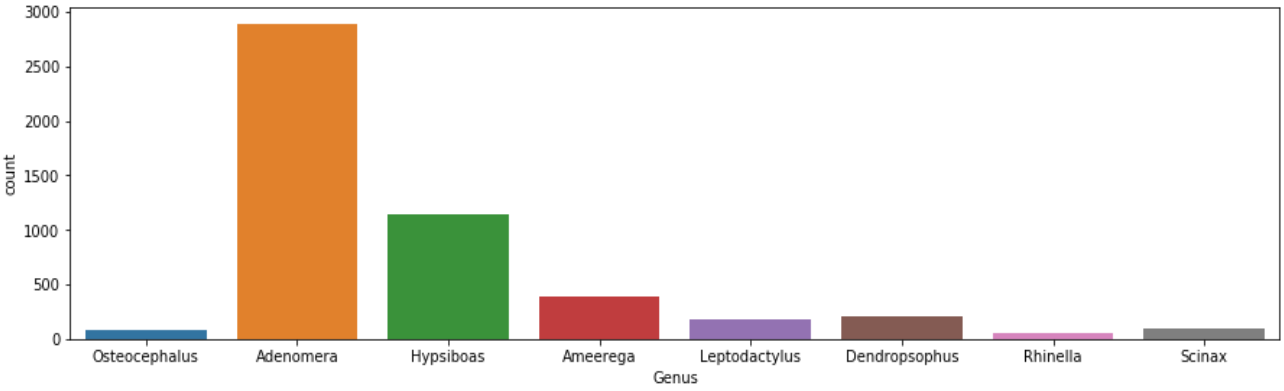
```
Adenomera         2891
Hypsiboas         1138
Ameerega           394
Dendropsophus      208
Leptodactylus      184
Scinax              90
Osteocephalus       82
Rhinella            50
```

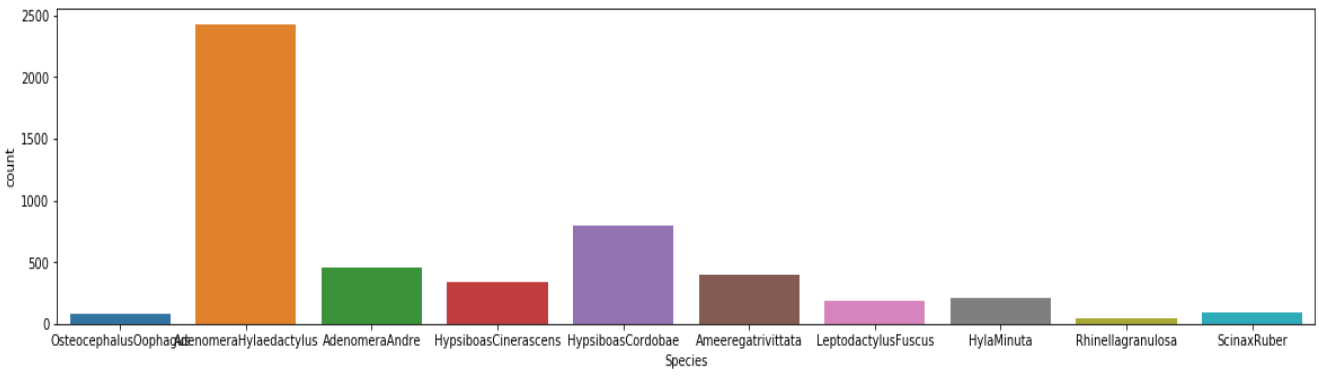
3. Species-Label

```
AdenomeraHylaedactylus    2430
HypsiboasCordobae          801
AdenomeraAndre             461
Ameeregatrivittata         394
HypsiboasCinerascens       337
HylaMinuta                 208
LeptodactylusFuscus        184
ScinaxRuber                 90
OsteocephalusOophagus       82
Rhinellagranulosa           50
```

Understanding Data Further







Hamming loss

The Hamming loss is the fraction of labels that are incorrectly predicted.

In multiclass classification, the Hamming loss correspond to the Hamming distance between **y_true** and **y_pred** which is equivalent to the subset **zero_one_loss** function.

The Hamming loss is upperbounded by the subset zero-one loss. When normalized over samples, the Hamming loss is always between 0 and 1.

In multilabel classification, the Hamming loss is different from the subset zero-one loss. The zero-one loss considers the entire set of labels for a given sample incorrect if it does entirely match the true set of labels. Hamming loss is more forgiving in that it penalizes the individual labels.

Exact match:

In multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must *exactly* match the corresponding set of labels in y_true.

1. For first Lable- Family

A.Hamming Score - 0.06255792400370713

B. Confusion Matrix

```
array([[   0,    0,   16,    4],
       [   0,  147,    9,    3],
       [   0,    8,  553,   64],
       [   0,   12,   19, 1323]], dtype=int64)
```

C. **matthews_corrcoef - 0.8777121723357324**

D. jaccard_similarity_score - 0.9374420759962928

E. Exact Match Score - 0.9374420759962928

2 . For second Lable- Genus

A.Hamming Score - 0.04865616311399444

B. Confusion Matrix

```
array([[1255,   11,    0,    3,    0,    2,    0,    1],
       [   0,  159,    0,    0,    0,    0,    0,    0],
       [  26,    6,   62,    0,    0,    0,    0,    0],
       [  19,    0,    0,  423,    5,    0,    0,    2],
       [   2,    0,    0,    6,   74,    0,    0,    0],
       [   3,    0,    0,   13,    0,   17,    0,    0],
       [   3,    0,    0,    0,    2,    0,   15,    0],
       [   0,    0,    0,    1,    0,    0,    0,   48]],
```

C. **matthews_corrcoef - 0.9181733978444799**

D. jaccard_similarity_score - 0.9513438368860055

E. Exact Match Score - **0.9513438368860055**

3. For third Lable- Species

A.Hamming Score - 0.04309545875810936

B. Confusion Matrix

```
array([[ 203,    0,    8,    0,    0,    2,    0,    2,    0,    0],
       [   0, 1055,    0,    0,    0,    0,    0,    0,    0,    2],
       [   1,    0,  155,    3,    0,    0,    0,    0,    0,    0],
       [  14,    9,    5,   64,    1,    1,    0,    0,    0,    0],
       [   5,    0,    0,    0,  127,    1,    0,    1,    0,    0],
       [   1,    4,    0,    0,    4,  297,    8,    1,    0,    0],
       [   1,    0,    0,    0,    0,    2,   79,    0,    0,    0],
       [   1,    0,    0,    0,    8,    2,    0,   21,    1,    0],
       [   1,    0,    0,    0,    0,    0,    4,    0,   15,    0],
       [   0,    0,    0,    0,    0,    0,    0,    0,    0,   49]],
```

C. **matthews_corrcoef - 0.9397568410736615**

D. jaccard_similarity_score - 0.9569045412418906

E. Exact Match Score **0.9569045412418906**

Train a SVM for each of the labels: Gaussian kernels and one versus all classifiers:

With Raw Attributes (Before Normalization):

1. **SVM for First Classifier with Label-Family:**

```
GridSearchCV(cv=10, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params=None, iid=True, n_jobs=1,
       param_grid={'C': [1, 10, 100, 1000], 'gamma': [0.01, 0.1, 0.2, 0.5]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
```

```
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
()
Grid scores on training set:
()
0.792 (+/-0.046) for {'C': 1, 'gamma': 0.01}
0.931 (+/-0.021) for {'C': 1, 'gamma': 0.1}
0.941 (+/-0.014) for {'C': 1, 'gamma': 0.2}
0.962 (+/-0.014) for {'C': 1, 'gamma': 0.5}
0.925 (+/-0.021) for {'C': 10, 'gamma': 0.01}
0.958 (+/-0.016) for {'C': 10, 'gamma': 0.1}
0.973 (+/-0.017) for {'C': 10, 'gamma': 0.2}
0.984 (+/-0.010) for {'C': 10, 'gamma': 0.5}
0.944 (+/-0.022) for {'C': 100, 'gamma': 0.01}
0.979 (+/-0.012) for {'C': 100, 'gamma': 0.1}
0.983 (+/-0.012) for {'C': 100, 'gamma': 0.2}
0.988 (+/-0.009) for {'C': 100, 'gamma': 0.5}
0.962 (+/-0.016) for {'C': 1000, 'gamma': 0.01}
0.985 (+/-0.013) for {'C': 1000, 'gamma': 0.1}
0.985 (+/-0.009) for {'C': 1000, 'gamma': 0.2}
0.988 (+/-0.008) for {'C': 1000, 'gamma': 0.5}
```

```
Detailed classification report:
The model is trained on the full development set.
The scores are computed on the full evaluation set.
                precision    recall  f1-score   support

      Bufonidae       0.88      0.83      0.86        18
  Dendrobatidae       0.98      1.00      0.99       148
        Hylidae       0.98      0.99      0.98       647
Leptodactylidae       0.99      0.99      0.99      1345

    avg / total       0.99      0.99      0.99      2158
```

2. SVM for Second Classifier with Label-Genus:

```
GridSearchCV(cv=10, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params=None, iid=True, n_jobs=1,
       param_grid={'C': [1, 10, 100, 1000], 'gamma': [0.01, 0.1, 0.2, 0.5]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
```

```
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
()
Grid scores on training set:
()
0.791 (+/-0.008) for {'C': 1, 'gamma': 0.01}
0.918 (+/-0.012) for {'C': 1, 'gamma': 0.1}
0.936 (+/-0.012) for {'C': 1, 'gamma': 0.2}
0.966 (+/-0.014) for {'C': 1, 'gamma': 0.5}
0.917 (+/-0.010) for {'C': 10, 'gamma': 0.01}
0.965 (+/-0.016) for {'C': 10, 'gamma': 0.1}
0.974 (+/-0.015) for {'C': 10, 'gamma': 0.2}
0.985 (+/-0.009) for {'C': 10, 'gamma': 0.5}
0.963 (+/-0.014) for {'C': 100, 'gamma': 0.01}
0.982 (+/-0.011) for {'C': 100, 'gamma': 0.1}
0.986 (+/-0.009) for {'C': 100, 'gamma': 0.2}
0.987 (+/-0.009) for {'C': 100, 'gamma': 0.5}
0.975 (+/-0.014) for {'C': 1000, 'gamma': 0.01}
0.984 (+/-0.005) for {'C': 1000, 'gamma': 0.1}
0.985 (+/-0.008) for {'C': 1000, 'gamma': 0.2}
0.986 (+/-0.009) for {'C': 1000, 'gamma': 0.5}
```

```
Detailed classification report:
The model is trained on the full development set.
The scores are computed on the full evaluation set.
```

	precision	recall	f1-score	support
Adenomera	1.00	1.00	1.00	1259
Ameerega	0.98	1.00	0.99	148
Dendropsophus	0.98	0.94	0.96	102
Hypsiboas	0.99	1.00	0.99	455
Leptodactylus	0.97	0.98	0.97	86
Osteocephalus	0.97	0.94	0.95	32
Rhinella	0.88	0.83	0.86	18
Scinax	1.00	0.91	0.95	58
avg / total	0.99	0.99	0.99	2158

3. SVM for Third Classifier with Species:

```
GridSearchCV(cv=10, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params=None, iid=True, n_jobs=1,
       param_grid={'C': [1, 10, 100, 1000], 'gamma': [0.01, 0.1, 0.2, 0.5]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.2}
```

```
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.2}
()
Grid scores on training set:
()
0.806 (+/-0.024) for {'C': 1, 'gamma': 0.01}
0.924 (+/-0.016) for {'C': 1, 'gamma': 0.1}
0.943 (+/-0.013) for {'C': 1, 'gamma': 0.2}
0.965 (+/-0.017) for {'C': 1, 'gamma': 0.5}
0.924 (+/-0.017) for {'C': 10, 'gamma': 0.01}
0.970 (+/-0.015) for {'C': 10, 'gamma': 0.1}
0.976 (+/-0.014) for {'C': 10, 'gamma': 0.2}
0.986 (+/-0.013) for {'C': 10, 'gamma': 0.5}
0.969 (+/-0.014) for {'C': 100, 'gamma': 0.01}
0.984 (+/-0.012) for {'C': 100, 'gamma': 0.1}
0.987 (+/-0.012) for {'C': 100, 'gamma': 0.2}
0.987 (+/-0.009) for {'C': 100, 'gamma': 0.5}
0.979 (+/-0.016) for {'C': 1000, 'gamma': 0.01}
0.984 (+/-0.010) for {'C': 1000, 'gamma': 0.1}
0.985 (+/-0.010) for {'C': 1000, 'gamma': 0.2}
0.986 (+/-0.007) for {'C': 1000, 'gamma': 0.5}
```

Detailed classification report:
The model is trained on the full development set.
The scores are computed on the full evaluation set.

	precision	recall	f1-score	support
AdenomeraAndre	0.97	0.99	0.98	211
AdenomeraHylaedactylus	1.00	1.00	1.00	1048
Ameeregatrivittata	0.99	1.00	1.00	148
HylaMinuta	0.96	0.94	0.95	102
HypsiboasCinerascens	0.96	0.99	0.97	135
HypsiboasCordobae	0.99	0.99	0.99	320
LeptodactylusFuscus	0.98	0.97	0.97	86
OsteocephalusOophagus	0.89	0.97	0.93	32
Rhinellagranulosa	0.89	0.89	0.89	18
ScinaxRuber	1.00	0.93	0.96	58
avg / total	0.99	0.99	0.99	2158

Repeating the procedure After Normalization (Gausian Kernel and one verses other classifiers):

1. SVM for First Classifier with Label-Family:

```
GridSearchCV(cv=10, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params=None, iid=True, n_jobs=1,
       param_grid={'C': [1, 10, 100, 1000], 'gamma': [0.01, 0.1, 0.2, 0.5]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
```

```
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
()
Grid scores on training set:
()
0.792 (+/-0.046) for {'C': 1, 'gamma': 0.01}
0.931 (+/-0.021) for {'C': 1, 'gamma': 0.1}
0.941 (+/-0.014) for {'C': 1, 'gamma': 0.2}
0.962 (+/-0.014) for {'C': 1, 'gamma': 0.5}
0.925 (+/-0.021) for {'C': 10, 'gamma': 0.01}
0.958 (+/-0.016) for {'C': 10, 'gamma': 0.1}
0.973 (+/-0.017) for {'C': 10, 'gamma': 0.2}
0.984 (+/-0.010) for {'C': 10, 'gamma': 0.5}
0.944 (+/-0.022) for {'C': 100, 'gamma': 0.01}
0.979 (+/-0.012) for {'C': 100, 'gamma': 0.1}
0.983 (+/-0.012) for {'C': 100, 'gamma': 0.2}
0.988 (+/-0.009) for {'C': 100, 'gamma': 0.5}
0.962 (+/-0.016) for {'C': 1000, 'gamma': 0.01}
0.985 (+/-0.013) for {'C': 1000, 'gamma': 0.1}
0.985 (+/-0.009) for {'C': 1000, 'gamma': 0.2}
0.988 (+/-0.008) for {'C': 1000, 'gamma': 0.5}
```

```
Detailed classification report:
The model is trained on the full development set.
The scores are computed on the full evaluation set.
                precision    recall  f1-score   support

      Bufonidae       0.88      0.83      0.86        18
  Dendrobatidae       0.98      1.00      0.99       148
        Hylidae       0.98      0.99      0.98       647
Leptodactylidae       0.99      0.99      0.99      1345

    avg / total       0.99      0.99      0.99      2158
```

2. SVM for Second Classifier with Label-Genus:

```
GridSearchCV(cv=10, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params=None, iid=True, n_jobs=1,
       param_grid={'C': [1, 10, 100, 1000], 'gamma': [0.01, 0.1, 0.2, 0.5]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
```

```
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.5}
()
Grid scores on training set:
()
0.791 (+/-0.008) for {'C': 1, 'gamma': 0.01}
0.918 (+/-0.012) for {'C': 1, 'gamma': 0.1}
0.936 (+/-0.012) for {'C': 1, 'gamma': 0.2}
0.966 (+/-0.014) for {'C': 1, 'gamma': 0.5}
0.917 (+/-0.010) for {'C': 10, 'gamma': 0.01}
0.965 (+/-0.016) for {'C': 10, 'gamma': 0.1}
0.974 (+/-0.015) for {'C': 10, 'gamma': 0.2}
0.985 (+/-0.009) for {'C': 10, 'gamma': 0.5}
0.963 (+/-0.014) for {'C': 100, 'gamma': 0.01}
0.982 (+/-0.011) for {'C': 100, 'gamma': 0.1}
0.986 (+/-0.009) for {'C': 100, 'gamma': 0.2}
0.987 (+/-0.009) for {'C': 100, 'gamma': 0.5}
0.975 (+/-0.014) for {'C': 1000, 'gamma': 0.01}
0.984 (+/-0.005) for {'C': 1000, 'gamma': 0.1}
0.985 (+/-0.008) for {'C': 1000, 'gamma': 0.2}
0.986 (+/-0.009) for {'C': 1000, 'gamma': 0.5}
```

```
Detailed classification report:
The model is trained on the full development set.
The scores are computed on the full evaluation set.
```

	precision	recall	f1-score	support
Adenomera	1.00	1.00	1.00	1259
Ameerega	0.98	1.00	0.99	148
Dendropsophus	0.98	0.94	0.96	102
Hypsiboas	0.99	1.00	0.99	455
Leptodactylus	0.97	0.98	0.97	86
Osteocephalus	0.97	0.94	0.95	32
Rhinella	0.88	0.83	0.86	18
Scinax	1.00	0.91	0.95	58
avg / total	0.99	0.99	0.99	2158

3. SVM for Second Classifier with Label-Species:

```
GridSearchCV(cv=10, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params=None, iid=True, n_jobs=1,
       param_grid={'C': [1, 10, 100, 1000], 'gamma': [0.01, 0.1, 0.2, 0.5]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.2}
```

```
Best parameters set found on development set:
()
{'C': 100, 'gamma': 0.2}
()
Grid scores on training set:
()
0.806 (+/-0.024) for {'C': 1, 'gamma': 0.01}
0.924 (+/-0.016) for {'C': 1, 'gamma': 0.1}
0.943 (+/-0.013) for {'C': 1, 'gamma': 0.2}
0.965 (+/-0.017) for {'C': 1, 'gamma': 0.5}
0.924 (+/-0.017) for {'C': 10, 'gamma': 0.01}
0.970 (+/-0.015) for {'C': 10, 'gamma': 0.1}
0.976 (+/-0.014) for {'C': 10, 'gamma': 0.2}
0.986 (+/-0.013) for {'C': 10, 'gamma': 0.5}
0.969 (+/-0.014) for {'C': 100, 'gamma': 0.01}
0.984 (+/-0.012) for {'C': 100, 'gamma': 0.1}
0.987 (+/-0.012) for {'C': 100, 'gamma': 0.2}
0.987 (+/-0.009) for {'C': 100, 'gamma': 0.5}
0.979 (+/-0.016) for {'C': 1000, 'gamma': 0.01}
0.984 (+/-0.010) for {'C': 1000, 'gamma': 0.1}
0.985 (+/-0.010) for {'C': 1000, 'gamma': 0.2}
0.986 (+/-0.007) for {'C': 1000, 'gamma': 0.5}
```

Detailed classification report:
The model is trained on the full development set.
The scores are computed on the full evaluation set.

	precision	recall	f1-score	support
AdenomeraAndre	0.97	0.99	0.98	211
AdenomeraHylaedactylus	1.00	1.00	1.00	1048
Ameeregatrivittata	0.99	1.00	1.00	148
HylaMinuta	0.96	0.94	0.95	102
HypsiboasCinerascens	0.96	0.99	0.97	135
HypsiboasCordobae	0.99	0.99	0.99	320
LeptodactylusFuscus	0.98	0.97	0.97	86
OsteocephalusOophagus	0.89	0.97	0.93	32
Rhinellagranulosa	0.89	0.89	0.89	18
ScinaxRuber	1.00	0.93	0.96	58
avg / total	0.99	0.99	0.99	2158

L1 Penalised SVMs for 'Classifier 1-for Label-Family'

clf.score(x_test, y_test_family) - **0.9360518999073216**

L1 Penalised SVMs for 'Classifier 1-for Label-Genus'

clf.score(x_test, y_test_genus) - **0.9518072289156626**

L1 Penalised SVMs for 'Classifier 1-for Label-Species'

clf.score(x_test, y_test_species) - **0.9582947173308619**

SMOTE:

1. For Label –Family

```
Resampled dataset shape Counter({'Bufonidae': 3075, 'Hylidae': 30
75, 'Dendrobatidae': 3075, 'Leptodactylidae': 3075})
```

clf.score(x_test_fam, y_test_fam)= **0.9064881948317531**

2. For Label –Genus

```
Resampled dataset shape Counter({'Adenomera': 2891, 'Scinax': 289
1, 'Dendropsophus': 2891, 'Osteocephalus': 2890, 'Ameerega': 2890
, 'Rhinella': 2890, 'Hypsiboas': 2890, 'Leptodactylus': 2890})
```

clf.score(x_test_gen, y_test_gen) = **0.9285998013902681**

3. For Label –Species

```
Resampled dataset shape Counter({'AdenomeraHylaedactylus': 1048, 'Ameer
egatrivittata': 1048, 'HylaMinuta': 1047, 'Rhinellagranulosa': 1047, 'H
ypsiboasCinerascens': 1047, 'OsteocephalusOophagus': 1047, 'HypsiboasCo
rdobae': 1047, 'LeptodactylusFuscus': 1047, 'AdenomeraAndre': 1047, 'Sc
inaxRuber': 1047})
```

clf.score(x_test_sep, y_test_sep) = **0.887318563789152**