1. Full Screenshot
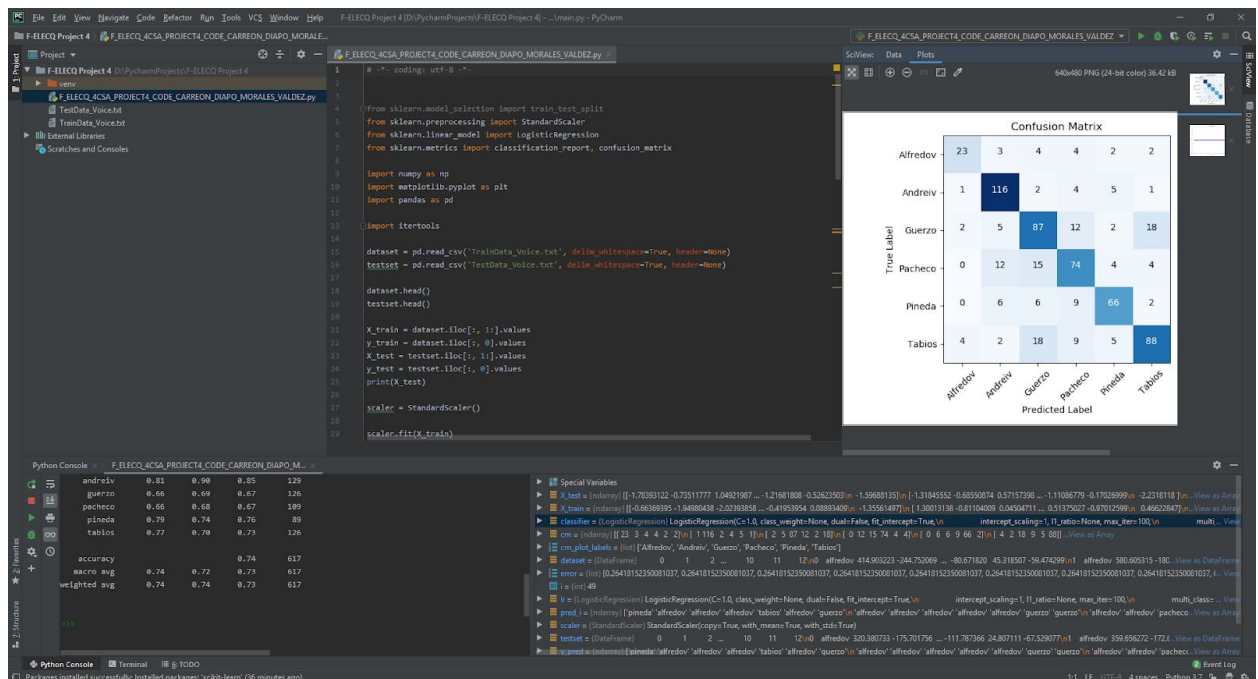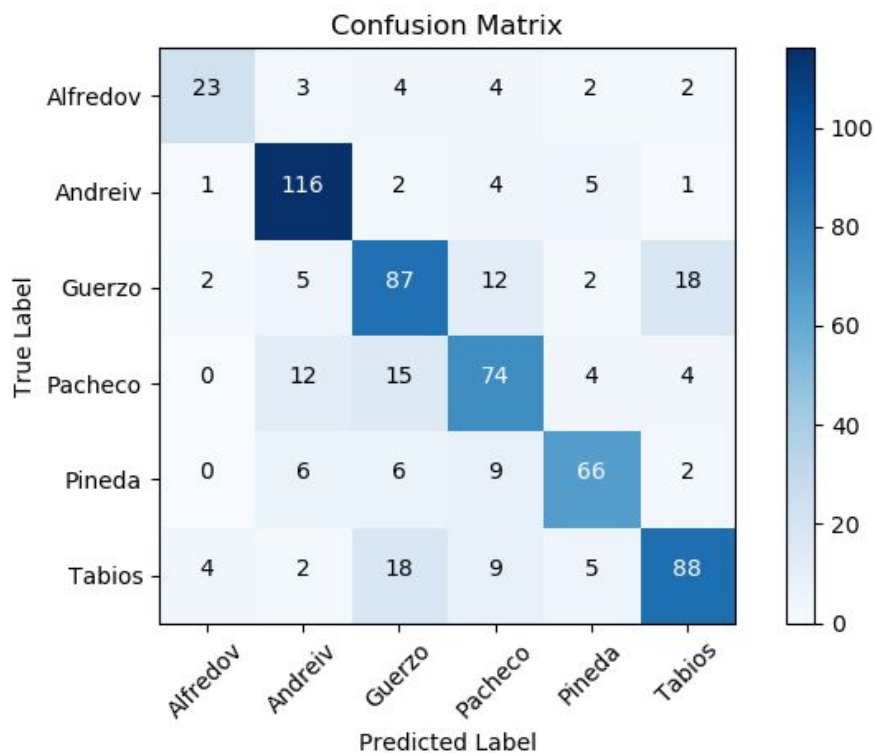

2. Confusion Matrix

3. Accuracy & Averages

```
"C:\Users\Ranier Diapo\AppData\Local\Programs\Python\Python37\pyth

import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['D:\\PycharmProjects\\F-ELECQ Project 4', 'D:/Pyc

PyDev console: starting.

Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC
>>> runfile('D:/PycharmProjects/F-ELECQ Project 4/F_ELECQ_4CSA_PRC
[[ 320.38073329 -175.70175624  225.51563906 ... -111.78736598
    24.80711095  -67.5290773 ]
 [ 359.65627159 -172.8816713   202.85449563 ... -107.65242324
    36.67594448  -88.72649449]
 [ 370.71302412 -142.50388841  199.19264872 ...  -99.3881671
    47.70283434  -93.24951035]
 ...
 [ 554.61856386 -150.90633669  182.86465688 ...  -47.8750573
    16.2055759    13.14674688]
 [ 562.41722594 -190.09543442  168.83165168 ...  -43.81135196
    34.71100928   30.83215834]
 [ 564.68830796 -151.05504246  177.02137135 ...  -31.81653032
    24.30429612   -1.44737245]]
              precision    recall  f1-score   support

    alfredov       0.77      0.61      0.68        38
     andreiv       0.81      0.90      0.85       129
      guerzo       0.66      0.69      0.67       126
     pacheco       0.66      0.68      0.67       109
      pineda       0.79      0.74      0.76        89
      tabios       0.77      0.70      0.73       126

    accuracy                          0.74       617
   macro avg       0.74      0.72      0.73       617
weighted avg       0.74      0.74      0.73       617


>>>
```

4. Special Variables

```
▶  ▦ Special Variables
▶  ▤ X_test = {ndarray} [[-1.78393122 -0.73511777  1.04921987 ...  -1.21681808 -0.52623503]\n  -1.59688135]\n [-1.31845552 -0.68550874  0.57157398 ...  -1.11086779 -0.1702
▶  ▤ X_train = {ndarray} [[-0.66369395 -1.94980438 -2.02393858 ...  -0.41953954  0.08893409\n  -1.35561497]\n [ 1.30013138 -0.81104009  0.04504711 ...  0.51375027 -0.9701
▶  ▤ classifier = {LogisticRegression} LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,\n              intercept_scaling=1, l1_ratio=None, m
▶  ▤ cm = {ndarray} [[ 23   3   4   4   2   2]\n [  1 116   2   4   5   1]\n [  2   5  87  12   2  18]\n [  0  12  15  74   4   4]\n [  0   6   6   9  66   2]\n [  4   2  18   9   5  88]]...View as Arra
▶  ▤≡ cm_plot_labels = {list} ['Alfredov', 'Andreiv', 'Guerzo', 'Pacheco', 'Pineda', 'Tabios']
▶  ▤ dataset = {DataFrame}        0         1         2  ...       10         11         12\n0  alfredov  414.903223 -244.752069  ...  -80.671820  45.318507 -59.474299\n1  alfredov
▶  ▤≡ error = {list} [0.26418152350081037, 0.26418152350081037, 0.26418152350081037, 0.26418152350081037, 0.26418152350081037, 0.26418152350081037, 0.26418152350
   ▣ i = {int} 49
▶  ▤ lr = {LogisticRegression} LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,\n              intercept_scaling=1, l1_ratio=None, max_iter=
▶  ▤ pred_i = {ndarray} ['pineda' 'alfredov' 'alfredov' 'alfredov' 'tabios' 'alfredov' 'guerzo'\n 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'guerzo' 'guerzo'\n 'alfred
▶  ▤ scaler = {StandardScaler} StandardScaler(copy=True, with_mean=True, with_std=True)
▶  ▤ testset = {DataFrame}        0         1         2  ...       10         11         12\n0  alfredov  320.380733 -175.701756  ...  -111.787366  24.807111 -67.529077\n1  alfredov
▶  ▤ y_pred = {ndarray} ['pineda' 'alfredov' 'alfredov' 'alfredov' 'tabios' 'alfredov' 'guerzo'\n 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'guerzo' 'guerzo'\n 'alfre
▶  ▤ y_test = {ndarray} ['alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov'\n 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov'\n 'alfredov' 'alfredov
▶  ▤ y_train = {ndarray} ['alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov'\n 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov' 'alfredov'\n 'alfredov' 'alfredo
```