# Clasificacion_arboles

October 16, 2020

Clasificación supervisada Usando scikit-learn

Predicción humedad a las 3 pm a partír de los features calculados entre las 8:55 am y 9:04 am por método de árboles de decisión

Importamos Librerías

[1]:
```
#!pip install sklearn
#!pip install graphviz
#!pip install pydotplus
#!conda install -c anaconda graphviz
#!pip install graphviz
#!pip install nbconvert
```

[2]:
```
import pandas as pd
from sklearn.metrics import accuracy_score
#from sklearn.externals.six import StringIO
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import export_graphviz
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import Image
import pydotplus
```

Cargamos los datos

[3]:
```
data = pd.read_csv('meteo/diario.csv')
```

[4]:
```
# Numero de datos
data.shape
```

[4]:
```
(1095, 11)
```

Caracteristicas a analizar

[5]:
```
#Visualiza las features
data.columns
```

[5]:
```
Index(['number', 'air_pressure_9am', 'air_temp_9am', 'avg_wind_direction_9am',
       'avg_wind_speed_9am', 'max_wind_direction_9am', 'max_wind_speed_9am',
       'rain_accumulation_9am', 'rain_duration_9am', 'relative_humidity_9am',
       'relative_humidity_3pm'],
      dtype='object')
```

Las primeras 10 filas del dataset

[50]:
```
#Muestra las 10 primeras filas
data.head(50)
```

[50]:

| | air_pressure_9am | air_temp_9am | avg_wind_direction_9am | \ |
|---|---|---|---|---|
| 0 | 918.060000 | 74.822000 | 271.100000 | |
| 1 | 917.347688 | 71.403843 | 101.935179 | |
| 2 | 923.040000 | 60.638000 | 51.000000 | |
| 3 | 920.502751 | 70.138895 | 198.832133 | |
| 4 | 921.160000 | 44.294000 | 277.800000 | |
| 5 | 915.300000 | 78.404000 | 182.800000 | |
| 6 | 915.598868 | 70.043304 | 177.875407 | |
| 7 | 918.070000 | 51.710000 | 242.400000 | |
| 8 | 920.080000 | 80.582000 | 40.700000 | |
| 9 | 915.010000 | 47.498000 | 163.100000 | |
| 10 | 919.650000 | 77.036000 | 70.600000 | |
| 11 | 915.640000 | 45.716000 | 241.600000 | |
| 12 | 917.390000 | 49.784000 | 204.100000 | |
| 13 | 920.820000 | 62.438000 | 213.600000 | |
| 14 | 911.000000 | 86.432000 | 202.900000 | |
| 15 | 922.383131 | 70.865263 | 36.174175 | |
| 17 | 916.915255 | 77.018961 | 234.539345 | |
| 18 | 918.800000 | 67.082000 | 176.100000 | |
| 19 | 922.040000 | 68.576000 | 58.300000 | |
| 20 | 919.992262 | 62.964383 | 54.799094 | |
| 21 | 917.230000 | 67.676000 | 177.800000 | |
| 22 | 921.125626 | 68.818772 | 71.799092 | |
| 23 | 920.350000 | 47.570000 | 192.100000 | |
| 24 | 921.788229 | 71.659572 | 217.405520 | |
| 25 | 918.030000 | 50.666000 | 128.900000 | |
| 26 | 914.490000 | 49.892000 | 163.000000 | |
| 27 | 914.900000 | 78.620000 | 203.300000 | |
| 28 | 915.840000 | 40.118000 | 171.900000 | |
| 29 | 916.310000 | 45.428000 | 183.100000 | |
| 30 | 919.350000 | 91.112000 | 96.400000 | |
| 31 | 916.280000 | 55.544000 | 174.600000 | |
| 32 | 918.700000 | 76.226000 | 184.600000 | |
| 33 | 918.370000 | 63.914000 | 53.700000 | |
| 34 | 923.600000 | 59.882000 | 192.800000 | |
| 35 | 917.300000 | 85.082000 | 149.700000 | |
| 36 | 912.640000 | 53.996000 | 177.700000 | |

```
37        914.670000     64.166000              137.800000
38        914.660000     50.360000              177.600000
39        921.969951     60.955342              164.621486
40        919.300000     64.220000              182.200000
41        921.090000     68.666000               50.700000
42        921.500000     49.550000               90.700000
43        914.900000     64.688000              174.600000
44        924.200000     68.882000               56.300000
45        914.540000     67.658000              170.200000
46        917.900000     65.660000              183.700000
47        916.050000     87.188000              210.000000
48        920.230000     67.316000              153.200000
49        923.663677     69.031070               64.689908
50        921.900000     65.876000              199.000000

    avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
0             2.080354              295.400000            2.863283
1             2.443009              140.471548            3.533324
2            17.067852               63.700000           22.100967
3             4.337363              211.203341            5.190045
4             1.856660              136.500000            2.863283
5             9.932014              189.000000           10.983375
6             3.745587              186.606696            4.589632
7             2.527742              271.600000            3.646212
8             4.518619               63.000000            5.883152
9             4.943637              195.900000            6.576604
10            3.825167               85.500000            4.764682
11            5.860783              265.800000            8.030615
12            1.275056              211.800000            2.013246
13            2.617220              165.700000            3.310671
14            1.207948              162.900000            1.677705
15            1.847278               58.428632            2.529142
17            2.274725              229.474199            2.906513
18            4.876529              183.400000            5.569981
19            9.551734               81.900000           12.571603
20           12.680436               74.254223           15.452306
21            2.460634               93.200000            3.288302
22            2.576538               95.472334            3.487444
23            6.263432              205.700000            7.605596
24            1.946447              253.758003            2.719712
25            2.527742              117.400000            4.004123
26            4.854160              189.600000            6.733189
27            1.811921              240.000000            2.751436
28           10.535987              188.000000           12.929513
29            8.343786              194.600000           10.088599
30            2.527742               96.300000            3.377779
31            2.594850              191.700000            3.377779
```

| | | | |
|---|---|---|---|
| 32 | 4.093600 | 194.900000 | 4.742313 |
| 33 | 14.450632 | 72.100000 | 17.045483 |
| 34 | 4.294925 | 203.500000 | 5.189701 |
| 35 | 2.013246 | 71.000000 | 2.863283 |
| 36 | 6.822667 | 190.700000 | 8.634588 |
| 37 | 2.125093 | 177.000000 | 2.729067 |
| 38 | 8.522741 | 186.300000 | 10.021491 |
| 39 | 1.664442 | 195.779968 | 2.367181 |
| 40 | 10.759681 | 186.500000 | 11.788674 |
| 41 | 14.204569 | 62.500000 | 17.045483 |
| 42 | 1.744813 | 112.300000 | 2.684328 |
| 43 | 5.659458 | 184.500000 | 6.867406 |
| 44 | 9.887275 | 69.000000 | 11.878151 |
| 45 | 8.254309 | 184.300000 | 10.849159 |
| 46 | 6.062107 | 188.500000 | 6.509495 |
| 47 | 1.565858 | 109.300000 | 2.594850 |
| 48 | 1.230317 | 136.900000 | 2.080354 |
| 49 | 7.557775 | 73.049232 | 8.819354 |
| 50 | 4.496249 | 207.200000 | 5.323917 |

| | rain_accumulation_9am | rain_duration_9am | relative_humidity_9am | \ |
|---|---|---|---|---|
| 0 | 0.000 | 0.0 | 42.420000 | |
| 1 | 0.000 | 0.0 | 24.328697 | |
| 2 | 0.000 | 20.0 | 8.900000 | |
| 3 | 0.000 | 0.0 | 12.189102 | |
| 4 | 8.900 | 14730.0 | 92.410000 | |
| 5 | 0.020 | 170.0 | 35.130000 | |
| 6 | 0.000 | 0.0 | 10.657422 | |
| 7 | 0.000 | 0.0 | 80.470000 | |
| 8 | 0.000 | 0.0 | 29.580000 | |
| 9 | 0.000 | 0.0 | 88.600000 | |
| 10 | 0.000 | 0.0 | 22.070000 | |
| 11 | 0.550 | 1770.0 | 90.560000 | |
| 12 | 0.000 | 0.0 | 73.150000 | |
| 13 | 0.000 | 0.0 | 43.640000 | |
| 14 | 0.000 | 0.0 | 15.190000 | |
| 15 | 0.000 | 0.0 | 12.110889 | |
| 17 | 0.000 | 0.0 | 21.031462 | |
| 18 | 0.000 | 0.0 | 18.900000 | |
| 19 | 0.000 | 0.0 | 7.540000 | |
| 20 | 0.000 | 0.0 | 18.809518 | |
| 21 | 0.000 | 0.0 | 40.640000 | |
| 22 | 0.000 | 0.0 | 11.742201 | |
| 23 | 0.000 | 0.0 | 54.550000 | |
| 24 | 0.000 | 0.0 | 11.194250 | |
| 25 | 0.000 | 0.0 | 76.880000 | |
| 26 | 0.000 | 0.0 | 92.100000 | |

| | | | |
|---|---|---|---|
| 27 | 0.000 | 220.0 | 47.890000 |
| 28 | 0.000 | 0.0 | 86.430000 |
| 29 | 0.000 | 0.0 | 86.150000 |
| 30 | 0.000 | 0.0 | 25.620000 |
| 31 | 0.000 | 0.0 | 79.920000 |
| 32 | 0.000 | 0.0 | 19.170000 |
| 33 | 0.000 | 0.0 | 19.680000 |
| 34 | 0.000 | 0.0 | 22.470000 |
| 35 | 0.000 | 0.0 | 32.020000 |
| 36 | 0.000 | 0.0 | 89.770000 |
| 37 | 0.000 | 0.0 | 58.570000 |
| 38 | 0.000 | 20.0 | 91.220000 |
| 39 | 0.000 | 0.0 | 12.109629 |
| 40 | 0.000 | 0.0 | 34.630000 |
| 41 | 0.000 | 0.0 | 17.580000 |
| 42 | 1.530 | 5420.0 | 34.350000 |
| 43 | 0.000 | 0.0 | 43.370000 |
| 44 | 0.000 | 0.0 | 14.230000 |
| 45 | 0.000 | 0.0 | 35.050000 |
| 46 | 0.021 | 319.0 | 74.220000 |
| 47 | 0.000 | 0.0 | 12.880000 |
| 48 | 0.000 | 0.0 | 13.890000 |
| 49 | 0.000 | 0.0 | 14.692501 |
| 50 | 0.000 | 0.0 | 10.350000 |

| | relative_humidity_3pm |
|---|---|
| 0 | 36.160000 |
| 1 | 19.426597 |
| 2 | 14.460000 |
| 3 | 12.742547 |
| 4 | 76.740000 |
| 5 | 33.930000 |
| 6 | 21.385657 |
| 7 | 74.920000 |
| 8 | 24.030000 |
| 9 | 68.050000 |
| 10 | 32.130000 |
| 11 | 79.090000 |
| 12 | 58.430000 |
| 13 | 27.990000 |
| 14 | 24.370000 |
| 15 | 14.801706 |
| 17 | 20.755683 |
| 18 | 45.870000 |
| 19 | 7.740000 |
| 20 | 14.649909 |
| 21 | 41.340000 |

```
22              17.281161
23              66.000000
24              16.331716
25              47.030000
26              90.990000
27              43.900000
28              84.390000
29              90.580000
30              35.690000
31              61.840000
32              31.970000
33              14.540000
34              14.260000
35              29.840000
36              55.570000
37              39.960000
38              88.350000
39              15.849564
40              61.320000
41              14.240000
42              49.500000
43              49.390000
44              13.580000
45              48.760000
46              69.190000
47              14.830000
48              20.520000
49              17.887744
50              15.370000
```

Las últimas 10 filas del dataset

[7]:
```
data.tail(10)
```

[7]:
```
      number  air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
1085    1085            914.84        47.354                   190.9
1086    1086            921.26        52.646                   261.9
1087    1087            914.00        66.650                   173.8
1088    1088            912.90        71.870                   129.2
1089    1089            915.00        55.040                   191.8
1090    1090            918.90        63.104                   192.9
1091    1091            918.71        49.568                   241.6
1092    1092            916.60        71.096                   189.3
1093    1093            912.60        58.406                   172.7
1094    1094            921.53        77.702                    97.1

      avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
1085            3.713320                   204.4            4.652835
```

```
1086            2.035615                   260.5            3.042238
1087            8.366156                   181.0            9.439887
1088            1.431642                   160.0            2.057985
1089            5.368656                   220.9            7.068730
1090            3.869906                   207.3            5.212070
1091            1.811921                   227.4            2.371156
1092            3.064608                   200.8            3.892276
1093            3.825167                   189.1            4.764682
1094            3.265932                   125.9            4.451511

      rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
1085                    0.0                0.0                  92.30
1086                    0.0                0.0                  91.11
1087                    0.0                0.0                  30.92
1088                    0.0                0.0                  51.84
1089                    0.0                0.0                  73.55
1090                    0.0                0.0                  26.02
1091                    0.0                0.0                  90.35
1092                    0.0                0.0                  45.59
1093                    0.0                0.0                  64.84
1094                    0.0                0.0                  14.56

      relative_humidity_3pm
1085                  88.16
1086                  81.89
1087                  47.34
1088                  55.49
1089                  69.67
1090                  38.18
1091                  73.34
1092                  52.31
1093                  58.28
1094                  15.10
```

Limpieza y depuración de la información

Comenzamos a revisar la información y a limpiar información inservible, también revisamos que variables nos interesan

[8]:
```
data[data.isnull().any(axis=1)] #selecciono filas  (f i l a s por eso uso
↪axis=1) que tengan valores null
```

[8]:
```
      number  air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
16        16         917.890000           NaN              169.200000
111      111         915.290000     58.820000              182.600000
177      177         915.900000           NaN              183.300000
262      262         923.596607     58.380598               47.737753
277      277         920.480000     62.600000              194.400000
```

| | | | | |
|---|---|---|---|---|
| 334 | 334 | 916.230000 | 75.740000 | 149.100000 |
| 358 | 358 | 917.440000 | 58.514000 | 55.100000 |
| 361 | 361 | 920.444946 | 65.801845 | 49.823346 |
| 381 | 381 | 918.480000 | 66.542000 | 90.900000 |
| 409 | 409 | NaN | 67.853833 | 65.880616 |
| 517 | 517 | 920.570000 | 53.600000 | 100.100000 |
| 519 | 519 | 916.250000 | 55.670000 | 176.400000 |
| 546 | 546 | NaN | 42.746000 | 251.100000 |
| 620 | 620 | 921.200000 | 56.786000 | 192.300000 |
| 625 | 625 | 912.400000 | 50.774000 | 171.600000 |
| 656 | 656 | 920.830000 | 66.344000 | NaN |
| 670 | 670 | 910.920000 | 48.362000 | 156.500000 |
| 672 | 672 | 922.448945 | 72.863773 | NaN |
| 705 | 705 | 911.900000 | 59.072000 | 199.800000 |
| 731 | 731 | 922.970166 | 51.391847 | 33.810942 |
| 737 | 737 | 917.895130 | 76.804690 | 104.771020 |
| 788 | 788 | 917.923442 | 73.249717 | 42.101739 |
| 840 | 840 | 918.043767 | NaN | 181.774042 |
| 848 | 848 | 915.250000 | 37.562000 | 246.500000 |
| 861 | 861 | 919.065408 | NaN | 172.303728 |
| 869 | 869 | NaN | 45.104000 | 259.000000 |
| 998 | 998 | 914.140000 | 71.240000 | NaN |
| 1031 | 1031 | 922.669195 | NaN | 47.946284 |
| 1035 | 1035 | 919.670000 | 77.576000 | 171.800000 |
| 1063 | 1063 | 917.300185 | 65.790001 | NaN |
| 1066 | 1066 | 919.564869 | 73.726732 | 68.704694 |

| | avg_wind_speed_9am | max_wind_direction_9am | max_wind_speed_9am | \ |
|---|---|---|---|---|
| 16 | 2.192201 | 196.800000 | 2.930391 | |
| 111 | 15.613841 | 189.000000 | NaN | |
| 177 | 4.719943 | 189.900000 | 5.346287 | |
| 262 | 10.636273 | 67.145843 | 13.671423 | |
| 277 | 2.751436 | NaN | 3.869906 | |
| 334 | 2.751436 | 187.500000 | 4.183078 | |
| 358 | 10.021491 | NaN | 12.705819 | |
| 361 | 21.520177 | 61.886944 | 25.549112 | |
| 381 | 3.467257 | 89.400000 | 4.406772 | |
| 409 | 4.328594 | 78.570923 | 5.216734 | |
| 517 | 4.697574 | NaN | 6.285801 | |
| 519 | 6.666081 | 188.200000 | NaN | |
| 546 | 12.929513 | 274.400000 | 17.604718 | |
| 620 | 9.551734 | 201.400000 | 11.005745 | |
| 625 | NaN | 181.400000 | 4.831790 | |
| 656 | 15.457255 | 189.400000 | 16.486248 | |
| 670 | NaN | 177.500000 | 16.128337 | |
| 672 | 3.682370 | 214.196160 | 4.849450 | |
| 705 | 1.275056 | 239.500000 | 1.834291 | |

| | | | |
|---|---|---|---|
| 731 | NaN | 59.290089 | 11.111555 |
| 737 | 1.632705 | 97.178763 | NaN |
| 788 | 4.132698 | 64.284969 | 5.345258 |
| 840 | 0.964376 | 185.618601 | 1.570007 |
| 848 | 11.587349 | 258.700000 | NaN |
| 861 | 2.639600 | 193.058141 | 3.326949 |
| 869 | 3.265932 | 275.000000 | 4.026492 |
| 998 | 1.722444 | 232.900000 | 2.326418 |
| 1031 | 7.969686 | 65.770066 | 10.262337 |
| 1035 | 6.554234 | 191.000000 | 8.164831 |
| 1063 | 1.879553 | 222.498226 | 2.692862 |
| 1066 | 3.551777 | 102.571616 | 4.861315 |

| | rain_accumulation_9am | rain_duration_9am | relative_humidity_9am | \ |
|---|---|---|---|---|
| 16 | 0.000 | 0.000000 | 48.990000 | |
| 111 | 0.000 | 0.000000 | 21.500000 | |
| 177 | 0.000 | 0.000000 | 29.260000 | |
| 262 | 0.000 | NaN | 17.990876 | |
| 277 | 0.000 | 0.000000 | 52.580000 | |
| 334 | NaN | 1480.000000 | 31.880000 | |
| 358 | 0.000 | 0.000000 | 13.880000 | |
| 361 | NaN | 40.364018 | 12.278715 | |
| 381 | NaN | 0.000000 | 20.640000 | |
| 409 | 0.000 | 0.000000 | 18.487385 | |
| 517 | 4.712 | 14842.000000 | 79.880000 | |
| 519 | 0.000 | 0.000000 | 72.550000 | |
| 546 | 14.627 | 7825.000000 | 87.870000 | |
| 620 | NaN | 0.000000 | 59.790000 | |
| 625 | 0.000 | 0.000000 | 86.840000 | |
| 656 | 0.000 | 0.000000 | 23.770000 | |
| 670 | 4.970 | 10560.000000 | 80.560000 | |
| 672 | 0.000 | 0.000000 | 16.753670 | |
| 705 | NaN | 0.000000 | 77.630000 | |
| 731 | 0.000 | 4.735034 | 34.807753 | |
| 737 | 0.000 | 0.000000 | 13.771311 | |
| 788 | 0.000 | NaN | 6.939692 | |
| 840 | 0.000 | 0.000000 | 11.911222 | |
| 848 | 3.171 | 2891.000000 | 91.000000 | |
| 861 | 0.000 | 0.000000 | 12.497839 | |
| 869 | 0.000 | 80.000000 | 85.270000 | |
| 998 | 0.000 | 0.000000 | 24.200000 | |
| 1031 | 0.000 | 0.000000 | 18.920805 | |
| 1035 | 0.000 | NaN | 56.860000 | |
| 1063 | 0.000 | 0.000000 | 14.972668 | |
| 1066 | NaN | 0.000000 | 11.657314 | |

| | relative_humidity_3pm |
|---|---|

```
16                51.190000
111               29.690000
177               46.500000
262               16.461685
277               54.030000
334               32.900000
358               25.930000
361                7.618649
381               14.350000
409               20.356594
517               84.530000
519               74.390000
546               70.770000
620               77.750000
625               64.740000
656               51.630000
670               88.220000
672               17.804720
705               59.130000
731               18.418179
737               16.792455
788               18.793825
840               18.154358
848               90.780000
861               13.438518
869               90.260000
998               41.380000
1031              19.641841
1035              50.650000
1063              20.966267
1066              17.331823
```

Comenzamos borrando los valores nulos y eliminamos la columna number que no nos serviría para el ejercicio...

[9]:
```
data = data.dropna() #Elimina los valores nulos
del data['number'] #borro la columna number ya que no la necesito
```

[10]:
```
data.head(10)
```

[10]:

| | air_pressure_9am | air_temp_9am | avg_wind_direction_9am | avg_wind_speed_9am | \ |
|---|---|---|---|---|---|
| 0 | 918.060000 | 74.822000 | 271.100000 | 2.080354 | |
| 1 | 917.347688 | 71.403843 | 101.935179 | 2.443009 | |
| 2 | 923.040000 | 60.638000 | 51.000000 | 17.067852 | |
| 3 | 920.502751 | 70.138895 | 198.832133 | 4.337363 | |
| 4 | 921.160000 | 44.294000 | 277.800000 | 1.856660 | |
| 5 | 915.300000 | 78.404000 | 182.800000 | 9.932014 | |
| 6 | 915.598868 | 70.043304 | 177.875407 | 3.745587 | |

```
7        918.070000     51.710000              242.400000            2.527742
8        920.080000     80.582000               40.700000            4.518619
9        915.010000     47.498000              163.100000            4.943637

   max_wind_direction_9am  max_wind_speed_9am  rain_accumulation_9am  \
0              295.400000            2.863283                   0.00
1              140.471548            3.533324                   0.00
2               63.700000           22.100967                   0.00
3              211.203341            5.190045                   0.00
4              136.500000            2.863283                   8.90
5              189.000000           10.983375                   0.02
6              186.606696            4.589632                   0.00
7              271.600000            3.646212                   0.00
8               63.000000            5.883152                   0.00
9              195.900000            6.576604                   0.00

   rain_duration_9am  relative_humidity_9am  relative_humidity_3pm
0                0.0              42.420000              36.160000
1                0.0              24.328697              19.426597
2               20.0               8.900000              14.460000
3                0.0              12.189102              12.742547
4            14730.0              92.410000              76.740000
5              170.0              35.130000              33.930000
6                0.0              10.657422              21.385657
7                0.0              80.470000              74.920000
8                0.0              29.580000              24.030000
9                0.0              88.600000              68.050000
```

[11]:
```
data.shape
```

[11]:
```
(1064, 10)
```

Metodo de Clasificación

Binarizamos la variable objetivo (target) relative_humidity_3pm a 0 o 1 para predecir si hay alta o baja humedad, se considera alta los datos > a 24.99 y baja los menores o iguales.

[12]:
```
#copiamos los datos depurados.
clean_data = data.copy()
#Creamos una nueva columna Humedad_alta_baja con informacion del
#dato objetivo pero binario para datos >24.99 = 1 y los demas 0
clean_data['Humedad_alta_baja'] = (clean_data['relative_humidity_3pm'] > 24.
 ↪99)*1
print(clean_data['Humedad_alta_baja'])
```

```
0       1
1       0
2       0
3       0
```

```
4       1
       ..
1090    1
1091    1
1092    1
1093    1
1094    0
Name: Humedad_alta_baja, Length: 1064, dtype: int32
```

El objetivo lo guardamos en 'y'.

[13]:
```
y=clean_data[['Humedad_alta_baja']].copy()
y
```

[13]:
```
      Humedad_alta_baja
0                     1
1                     0
2                     0
3                     0
4                     1
…                   …
1090                  1
1091                  1
1092                  1
1093                  1
1094                  0

[1064 rows x 1 columns]
```

[14]:
```
clean_data['relative_humidity_3pm']
```

[14]:
```
0       36.160000
1       19.426597
2       14.460000
3       12.742547
4       76.740000
          …
1090    38.180000
1091    73.340000
1092    52.310000
1093    58.280000
1094    15.100000
Name: relative_humidity_3pm, Length: 1064, dtype: float64
```

[15]:
```
y.head(10)
```

[15]:
```
   Humedad_alta_baja
0                  1
```

```
1                0
2                0
3                0
4                1
5                1
6                0
7                1
8                0
9                1
```

Ahora tomamos las caracteristicas de los sensores tomadas a las 9 am para predecir la humedad a las 3 pm

[16]:
```
morning_features =␣
↪['air_pressure_9am','air_temp_9am','avg_wind_direction_9am','avg_wind_speed_9am',
      ␣
↪'max_wind_direction_9am','max_wind_speed_9am','rain_accumulation_9am','rain_duration_9am'
       ]
```

Guardamos las caracteristicas en 'X' que son las features con que se entrena el modelo.

[17]:
```
X = clean_data[morning_features].copy()
```

[18]:
```
X.columns
```

[18]:
```
Index(['air_pressure_9am', 'air_temp_9am', 'avg_wind_direction_9am',
       'avg_wind_speed_9am', 'max_wind_direction_9am', 'max_wind_speed_9am',
       'rain_accumulation_9am', 'rain_duration_9am'],
      dtype='object')
```

[19]:
```
y.columns
```

[19]:
```
Index(['Humedad_alta_baja'], dtype='object')
```

Realizamos la partición de nuestro set de datos en set de entrenamiento y prueba

Separamos los datos de entrenamiento y de prueba, manejando una relación de 70% de entrenamiento y 30% para prueba

[20]:
```
# Se esta reservando 30% para pruebas
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=324)
```

[21]:
```
#type(X_train)
#type(X_test)
#type(y_train)
#type(y_test)
#X_train.head()
#y_train.describe()
```

Comenzamos con el ajuste y entrenamiento del modelo, al cual le pasamos los datos de entrenamiento…

[22]:
```
#creo el clasificador
clasificador_humedad = DecisionTreeClassifier(max_leaf_nodes=10, random_state=0)
#Ajustamos el modelo con el metodo fit, las caracteristicas (X) y el objetivo␣
↪(y)
#X_train Estos son los datos
#y_train Esto es lo que deseo obtener
clasificador_humedad.fi t(X_train, y_train)
```

[22]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                       max_features=None, max_leaf_nodes=10,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=0, splitter='best')
```

[23]:
```
type(clasificador_humedad)
```

[23]:
```
sklearn.tree.tree.DecisionTreeClassifier
```

PREDICCIONES

Ahora le pasamos todas las variables de prueba (X_test) a predictions y por medio de la funcion de clasificación, se crea el arbol binario

[24]:
```
predictions = clasificador_humedad.predict(X_test)
X_test
```

[24]:

| | air_pressure_9am | air_temp_9am | avg_wind_direction_9am | \ |
|---|---|---|---|---|
| 456 | 918.800000 | 80.384000 | 189.600000 | |
| 845 | 921.613373 | 68.658973 | 70.703555 | |
| 693 | 917.900000 | 76.802000 | 154.100000 | |
| 259 | 914.830000 | 74.570000 | 175.500000 | |
| 723 | 917.010000 | 51.836000 | 130.200000 | |
| .. | … | … | … | |
| 853 | 917.300000 | 73.184000 | 178.600000 | |
| 883 | 918.600000 | 64.382000 | 177.100000 | |
| 503 | 919.738161 | 67.432744 | 147.470882 | |
| 776 | 920.897662 | 66.528786 | 198.122180 | |
| 391 | 920.000000 | 68.054000 | 197.100000 | |

| | avg_wind_speed_9am | max_wind_direction_9am | max_wind_speed_9am | \ |
|---|---|---|---|---|
| 456 | 1.767183 | 80.300000 | 2.773806 | |
| 845 | 2.248932 | 96.844701 | 3.043049 | |
| 693 | 2.550112 | 199.400000 | 3.400149 | |
| 259 | 1.409272 | 153.800000 | 2.236940 | |
| 723 | 1.610597 | 159.900000 | 2.348787 | |

```
..                 …                   …                 …
853            4.675205          183.700000          5.256809
883            2.460634          190.700000          3.064608
503            1.157775          129.981819          1.899000
776            4.079293          210.715249          4.842644
391            4.675205          209.600000          6.151585

     rain_accumulation_9am  rain_duration_9am
456                    0.0                0.0
845                    0.0                0.0
693                    0.0                0.0
259                    0.0                0.0
723                    0.0                0.0
..                     …                  …
853                    0.0                0.0
883                    0.0                0.0
503                    0.0                0.0
776                    0.0                0.0
391                    0.0                0.0

[320 rows x 8 columns]
```

[36]:
```
predictions
```

[36]:
```
array([0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
       1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1,
       0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0,
       0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1,
       0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0])
```

RESULTADOS

[71]:
```
y_test['Humedad_alta_baja']
```

[71]:
```
456    0
845    0
693    1
259    1
```

```
723    1
       ..
853    1
883    1
503    0
776    0
391    0
Name: Humedad_alta_baja, Length: 320, dtype: int32
```

[33]:
```
comp = pd.DataFrame({'real': y_test, 'predicciones' : predictions})
```

```
␣
↪---------------------------------------------------------------------------

       ValueError                                Traceback (most recent call␣
↪last)

       ~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in␣
↪create_block_manager_from_arrays(arrays, names, axes)
      1694         blocks = form_blocks(arrays, names, axes)
   -> 1695         mgr = BlockManager(blocks, axes)
      1696         mgr._consolidate_inplace()


       ~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in␣
↪__init__(self, blocks, axes, do_integrity_check)
       142         if do_integrity_check:
   --> 143             self._verify_integrity()
       144


       ~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in␣
↪_verify_integrity(self)
       344             if block._verify_integrity and block.shape[1:] !=␣
↪mgr_shape[1:]:
   --> 345                 construction_error(tot_items, block.shape[1:], self.
↪axes)
       346         if len(self.items) != tot_items:


       ~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in␣
↪construction_error(tot_items, block_shape, axes, e)
      1718     raise ValueError(
   -> 1719         "Shape of passed values is {0}, indices imply {1}".
↪format(passed, implied)
      1720     )
```

```
       ValueError: Shape of passed values is (1, 2), indices imply (320, 2)


   During handling of the above exception, another exception occurred:


       ValueError                                Traceback (most recent call␣
↪last)

       <ipython-input-33-2378bc7dccfc> in <module>
   ----> 1 comp = pd.DataFrame({'real': y_test, 'predicciones' : predictions})


       ~\Anaconda3\lib\site-packages\pandas\core\frame.py in __init__(self,␣
↪data, index, columns, dtype, copy)
       409             )
       410         elif isinstance(data, dict):
   --> 411             mgr = init_dict(data, index, columns, dtype=dtype)
       412         elif isinstance(data, ma.MaskedArray):
       413             import numpy.ma.mrecords as mrecords


       ~\Anaconda3\lib\site-packages\pandas\core\internals\construction.py in␣
↪init_dict(data, index, columns, dtype)
       255             arr if not is_datetime64tz_dtype(arr) else arr.copy()␣
↪for arr in arrays
       256         ]
   --> 257     return arrays_to_mgr(arrays, data_names, index, columns,␣
↪dtype=dtype)
       258
       259


       ~\Anaconda3\lib\site-packages\pandas\core\internals\construction.py in␣
↪arrays_to_mgr(arrays, arr_names, index, columns, dtype)
        85     axes = [ensure_index(columns), index]
        86
   ---> 87     return create_block_manager_from_arrays(arrays, arr_names, axes)
        88
        89


       ~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in␣
↪create_block_manager_from_arrays(arrays, names, axes)
      1697         return mgr
```

```
   1698     except ValueError as e:
-> 1699         construction_error(len(arrays), arrays[0].shape, axes, e)
   1700
   1701


~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in␣
↪construction_error(tot_items, block_shape, axes, e)
   1717         raise ValueError("Empty data passed with indices specified.")
   1718     raise ValueError(
-> 1719         "Shape of passed values is {0}, indices imply {1}".
↪format(passed, implied)
   1720     )
   1721


ValueError: Shape of passed values is (1, 2), indices imply (320, 2)
```

Medimos la precisión

Si comparamos los resultados y las predicciones uno a uno podemos notar que los resultados son similares solo cambia en algunos casos pero para ser más precisos se puede medir precisión por medio de la siguiente función.

[31]:
```
#podemos medir que tan bien o mal esta funcionando el modelo
accuracy_score(y_test, predictions)
```

[31]:
```
0.825
```

El modelo tiene una presicion de 82.5% se puede buscar mejorar este porcentaje modificando el numero de nodos cuando se realiza en ajuste y entrenamiento del modelo DecisionTreeClassifier. (El valor de presicion dado es para un arbol de 10 nodos). Tambien se pueden revisar columnas que estan en ceros y se descartan. En estse parte se pueden realizar diferentes pruebas para ver si se puede mejorar la precisión.

[28]:
```
y_test.shape
```

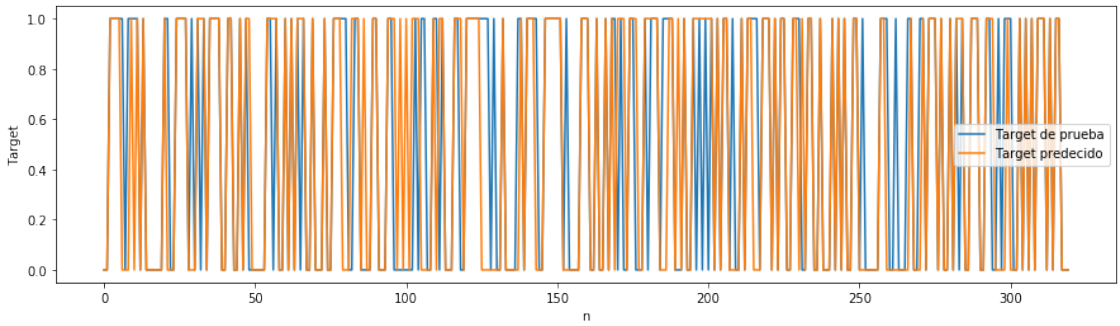
[28]:
```
(320, 1)
```

[29]:
```
predictions.shape
```

[29]:
```
(320,)
```

[30]:
```
import numpy as np
n = len(y_test)
t = np.array(range(n))
plt.figure(figsize=(15, 4))
plt.plot(t, y_test, label="Target de prueba")
```

```
plt.plot(t, predictions, label="Target predecido")
plt.legend()
plt.xlabel("n")
plt.ylabel("Target")
plt.show()
```




### 0.1 BIBLIOGRAFIA

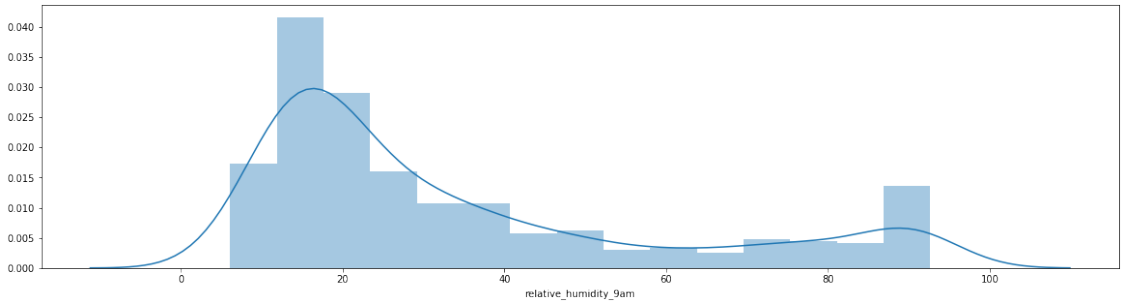
https://www.udemy.com/course/mlmasterclass/learn/lecture/14956362#overview
https://sitiobigdata.com/2019/12/14/arboles-de-decision-en-machine-learning-parte-2/

# 1 GRAFICO DE HUMEDAD RELATIVA A LAS 9:00 a.m

Este grafico muestra como se distriuyen los datos de humedad relativa a las 9:00 am a lo largo de los dias evaluados

[51]:
```
fig, ax=plt.subplots(figsize=(20,5))
sns.distplot(data.iloc[:,8])
```

[51]: <matplotlib.axes._subplots.AxesSubplot at 0xd640f60348>

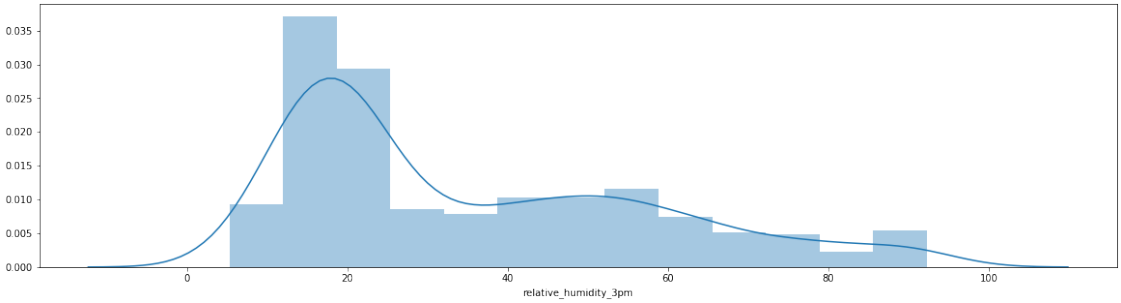


# 2 GRAFICO DE HUMEDAD RELATIVA A LAS 3:00 p.m

Este grafico muestra como se distriuyen los datos de la humedad relativa a las 3:00pm a lo largo de los dias evaluados

[53]:
```
fig, ax=plt.subplots(figsize=(20,5))
sns.distplot(data.iloc[:,9])
```

[53]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xd640b8df88>
```




[61]:
```
y_test
```

[61]:
```
     Humedad_alta_baja
456                  0
845                  0
693                  1
259                  1
723                  1
..                   …
853                  1
883                  1
503                  0
776                  0
391                  0

[320 rows x 1 columns]
```

## 2.1 Numero de días con humedad relativa alta y baja (Datos de entrada)

Para los datos de entrada se identifica que el 50,28% de los días son de humedad baja y el 49,72% son de humedad alta. Es decir que estan en una proporcion equivalente.

[99]:
```
total = (y[y==0].count())+(y[y==1].count())
ceros = y[y==0].count()
unos = y[y==1].count()
```

[100]:
```
#Numero de días con humedad relativa baja datos originales
print(total)
```

```
Humedad_alta_baja    1064
dtype: int64
```

[95]:
```
#Numero de días con humedad relativa baja datos originales
print(ceros)
```

```
Humedad_alta_baja    535
dtype: int64
```

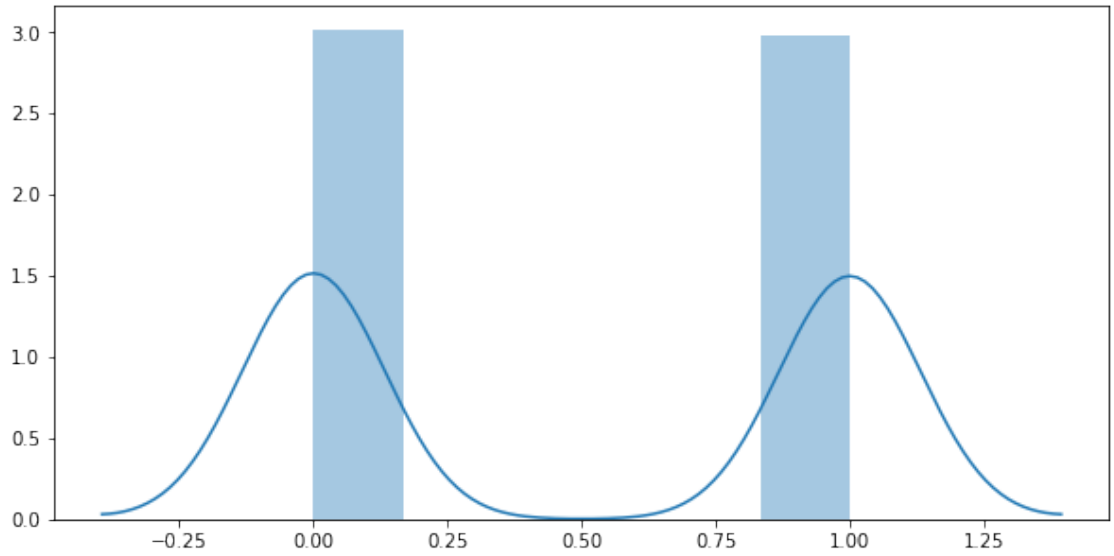
[94]:
```
#Numero de días con humedad relativa alta datos originales
print(unos)
```

```
Humedad_alta_baja    529
dtype: int64
```

### 2.2 Gráfico de número de días con humedad relativa alta y baja (datos originales)

[76]:
```
fig, ax=plt.subplots(figsize=(10,5))
sns.distplot(y)
```

[76]: <matplotlib.axes._subplots.AxesSubplot at 0xd640e3ba48>




### 2.3 Numero de días con humedad relativa alta y baja (Datos Pronosticados)

Para los datos de entrada se identifica que el 49,68% de los días son de humedad baja y el 50,31% son de humedad alta. Es decir que estan en una proporcion equivalente.

[101]:
```
total = (y_test[y==0].count())+(y_test[y_test==1].count())
ceros = y_test[y_test==0].count()
unos = y_test[y_test==1].count()
```

[102]:
```
#Numero de días con humedad relativa baja datos originales
print(total)
```

```
Humedad_alta_baja    320
dtype: int64
```

[103]:
```
#Numero de días con humedad relativa baja datos originales
print(ceros)
```

```
Humedad_alta_baja    159
dtype: int64
```

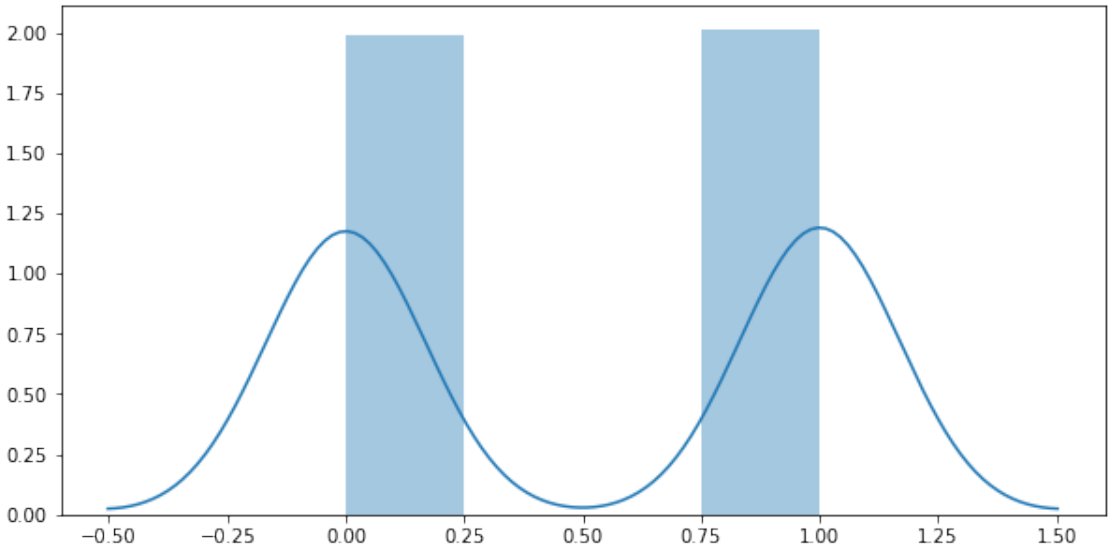
[104]:
```
#Numero de días con humedad relativa alta datos originales
print(unos)
```

```
Humedad_alta_baja    161
dtype: int64
```

### 2.4 Gráfico de número de días con humedad relativa alta y baja (datos originales)

[106]:
```
fig, ax=plt.subplots(figsize=(10,5))
sns.distplot(y_test)
```

[106]: <matplotlib.axes._subplots.AxesSubplot at 0xd640ed6ec8>



[ ]: