

# P3\_Tree\_Reg\_Matriculas

April 27, 2022

```
[1]: import tabula
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import matplotlib.colors as mcolors
from matplotlib.colors import ListedColormap

import sklearn.tree
import sklearn.metrics as metrics
from sklearn.model_selection import TimeSeriesSplit
from sklearn.linear_model import LinearRegression
from sklearn.neural_network import MLPRegressor
from sklearn.datasets import make_regression
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer
from sklearn import datasets, ensemble
from sklearn.inspection import permutation_importance
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error

[2]: def Matriculas(Dat, year):
    Aux = Dat.loc[Dat.index==year] # Selects only data for given
    Matricula = Aux.loc[Aux['Programa'] != 'Total'].fillna(0).reset_index()
    return Matricula

[3]: ##### Número de inscritos, admitidos y estudiantes_
    ↪ #####
```

```

##### matriculados por programa de pregrado
↳#####

# Take index_col=2 since col 2 has the sample periods and thus is more easy
↳data filtering.
Data = pd.read_csv('T15_Sup_Est_2021-I.csv', index_col=2, encoding='latin-1') #
↳Table 15 is in pages 32 - 36.

Matricula_2017_1 = Matriculas(Data, '2017-1')
Matricula_2017_2 = Matriculas(Data, '2017-2')
Matricula_2018_1 = Matriculas(Data, '2018-1')
Matricula_2018_2 = Matriculas(Data, '2018-2')
Matricula_2019_1 = Matriculas(Data, '2019-1')
Matricula_2019_2 = Matriculas(Data, '2019-2')
Matricula_2020_1 = Matriculas(Data, '2020-1')
Matricula_2020_2 = Matriculas(Data, '2020-2')
Matricula_2021_1 = Matriculas(Data, '2021-1')

```

```

[4]: data = Matricula_2017_2
year = '2017-2'
depmax = 3
reg_type = 'absolute_error'
cv_splits = 10 # Must be less than sample number
test = Matricula_2021_1

```

```

[5]: plt.rcParams["figure.figsize"] = [12, 8]

# setup the normalization and the colormap
deps = np.arange(0, len(data['Programa']))
normalize = mcolors.Normalize(vmin = deps.min(), vmax = deps.max())
colormap = cm.get_cmap('turbo', deps.max()+1)
# setup the colorbar
scalarmappable = cm.ScalarMappable(norm=normalize, cmap=colormap)
scalarmappable.set_array(deps)

fig = plt.figure()

fig.suptitle('Número de Estudiantes Matriculados por Programa '+str(year),
↳size=22)
plt.xlabel('No. Estudiantes Inscritos/Admitidos', fontsize=16)
plt.ylabel('No. Estudiantes Matriculados', fontsize=16)

plt.scatter(data['Inscritos (1ra Opción)'], data['Matriculados'],
↳label='Inscritos',
alpha=0.75, s=100, c=range(0, len(data)), cmap='turbo')

```

```

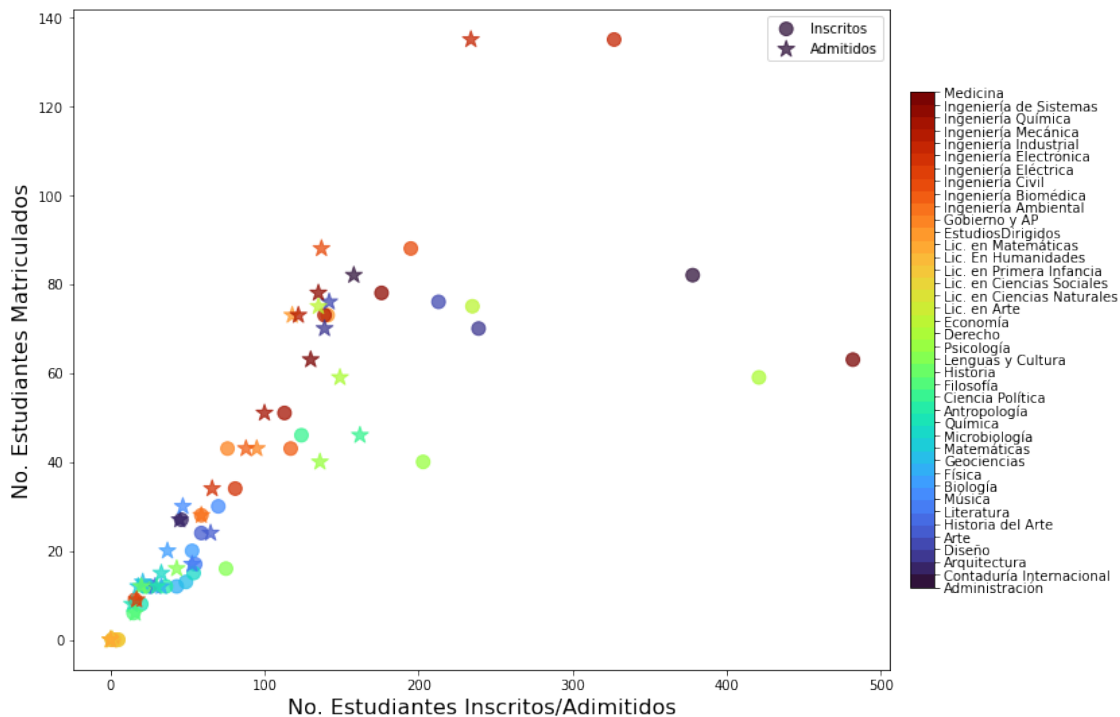
plt.scatter(data['Admitidos (1ra y 2da Opción)'], data['Matriculados'],
            label='Admitidos',
            alpha=0.75, marker='*', s=170, c=range(0, len(data)), cmap='turbo')
plt.legend()

cbar = plt.colorbar(scalarmappable, orientation='vertical', pad=0.02,
                    ticks=range(0, deps.max()+1, 1), shrink=0.75)
cbar.ax.set_yticklabels(data['Programa'])

plt.subplots_adjust(left=0.075, bottom=0.075, right=0.92, top=0.925)
plt.show()
fig.savefig('Matriculados_'+str(year)+'.pdf', dpi=500)
plt.close(fig)

```

Número de Estudiantes Matriculados por Programa 2017-2



```

[6]: Scores = []
     Deeps = []

     for deeps in range(2, 15):
         regs = DecisionTreeRegressor(max_depth=deeps, criterion=reg_type)
         scs = -cross_val_score(regs,
                                data[['Inscritos (1ra Opción)', 'Admitidos (1ra y 2da Opción)']],

```

```

        data['Matriculados'],
        scoring='neg_mean_absolute_error',
        cv=cv_splits)

    Scores.append(scs)
    Deeps.append(deeps)
    print(scs.mean(), scs.std())

fig = plt.figure()
plt.boxplot(Scores, labels=Deeps)
plt.title('Scores vs Depth Tree Using Cross-Validation with '+str(cv_splits)+'_
↳ splits - '+str(year), size=22)
plt.xlabel('Depth', fontsize=16)
plt.ylabel('Mean Absolute Error', fontsize=16)
plt.show()
fig.savefig('Scores_Splits_'+str(cv_splits)+'_'+str(year)+'.pdf', dpi=500)
plt.close(fig)

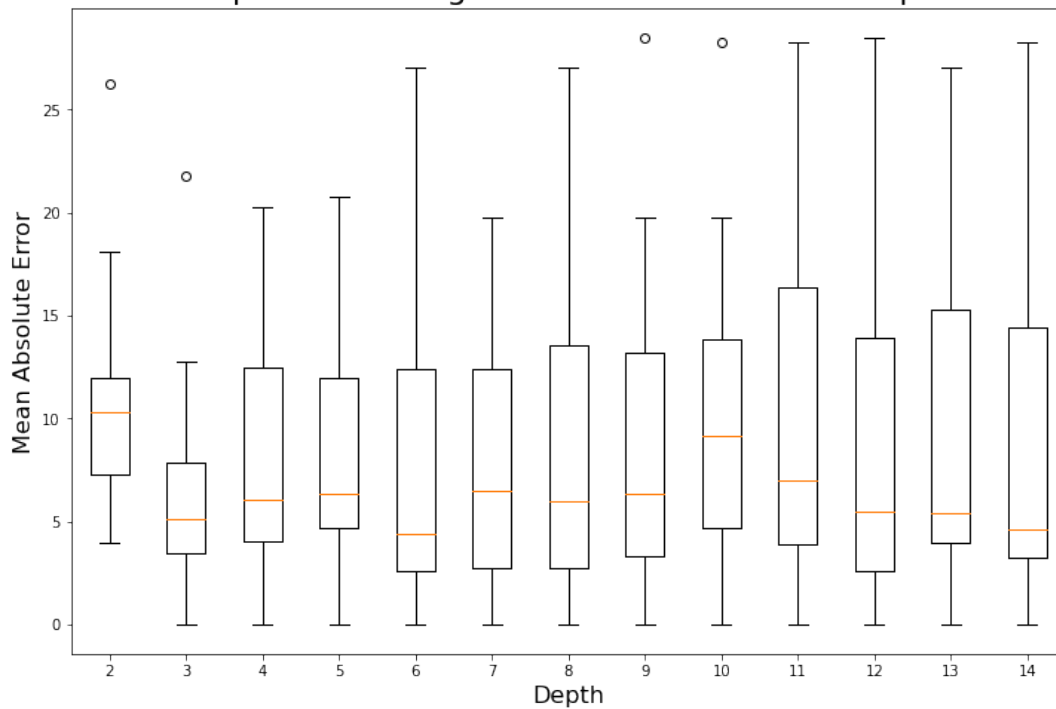
```

```

11.45 6.160915110598425
6.975 5.958659664723267
8.1375 5.975326873904055
8.8875 6.4847055638633275
8.4625 8.372583606629439
8.05 6.6208005558240455
8.975 8.419805520319336
9.65 8.429264499349868
10.475 8.24655230990503
10.375 8.673414840764853
9.025 8.852577308332304
9.6 8.40699113833243
8.925 8.92331916945707

```

Scores vs Depth Tree Using Cross-Validation with 10 splits - 2017-2



```
[7]: regressor = DecisionTreeRegressor(max_depth=depmax, criterion=reg_type)

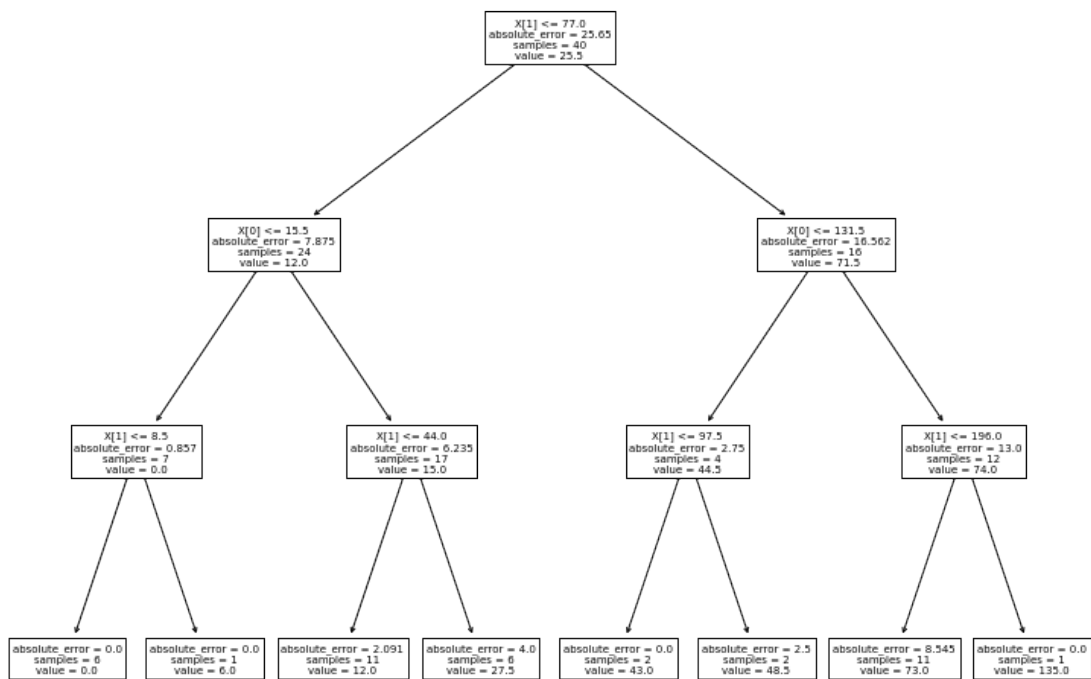
regressor.fit(data[['Inscritos (1ra Opción)', 'Admitidos (1ra y 2da Opción)']],
              data['Matriculados'])

#DecisionTreeRegressor(criterion='mse',
#                        #
#                        #      max_depth=2,
#                        #      max_features=None,
#                        #      max_leaf_nodes=None,
#                        #      min_samples_leaf=1,
#                        #      min_samples_split=2,
#                        #      min_weight_fraction_leaf=0.0,
#                        #      presort=False,
#                        #      random_state=None,
#                        #      splitter='best')

scores = -cross_val_score(regressor,
                           data[['Inscritos (1ra Opción)', 'Admitidos (1ra y 2da Opción)']],
                           data['Matriculados'],
                           scoring='neg_mean_absolute_error', cv=cv_splits)
```

```
[8]: fig = plt.figure(figsize=(13, 10))
fig.suptitle('Árbol de Decisión '+str(year)+' (Deep Tree: '+str(depmax)+')',
            size=22)
_ = sklearn.tree.plot_tree(regressor)
plt.show()
fig.savefig('Tree_'+str(year)+'.pdf', dpi=500)
plt.close(fig)
```

Árbol de Decisión 2017-2 (Deep Tree: 3)



```
[9]: Ins_max = max(data['Inscritos (1ra Opción)'])
Adm_max = max(data['Admitidos (1ra y 2da Opción)'])
Ins_min = min(data['Inscritos (1ra Opción)'])
Adm_min = min(data['Admitidos (1ra y 2da Opción)'])

N = len(data)

Ins_Test = np.linspace(Ins_min, Ins_max, N)
Adm_Test = np.linspace(Adm_min, Adm_max, N)
```

```
xx, yy = np.meshgrid(Ins_Test, Adm_Test)

Mat_Predtc = regressor.predict(np.array([xx.flatten(), yy.flatten()]).T)

zz = np.reshape(Mat_Predtc, (N, N))
```

/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:  
X does not have valid feature names, but DecisionTreeRegressor was fitted with  
feature names

```
warnings.warn(
```

```
[10]: plt.rcParams["figure.figsize"] = [15, 8]

cmap = plt.cm.ocean#Reds#RdBu
my_cmap = cmap(np.arange(cmap.N))
my_cmap[:, -1] = np.linspace(0, 1, cmap.N)
my_cmap = ListedColormap(my_cmap)

fig = plt.figure()

fig.suptitle('Número de Estudiantes Matriculados por Programa '+str(year)+'  
↳(Deep Tree: '+str(depmax)+' )',
            size=22)
plt.xlabel('Inscritos', fontsize=16)
plt.ylabel('Admitidos', fontsize=16)

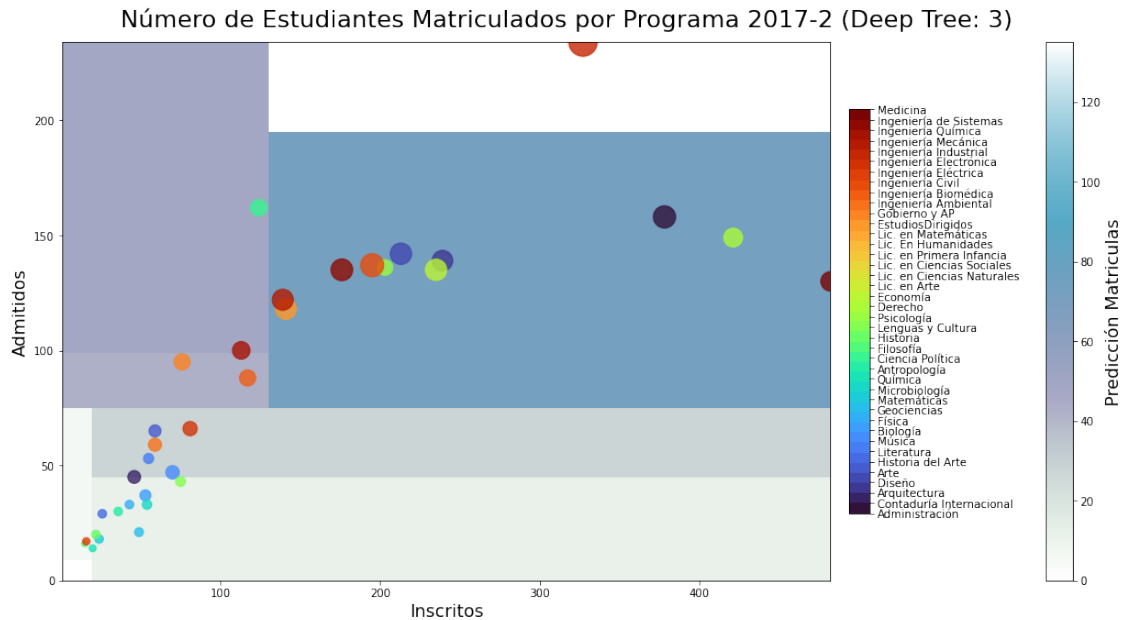
# setting the limit for each axis
plt.xlim(Ins_min, Ins_max)
plt.ylim(Adm_min, Adm_max)

# plotting the predictions
plt.pcolormesh(Ins_Test, Adm_Test, zz, cmap=my_cmap)
cb = plt.colorbar(label='Predicción Matriculas') # add a colorbar on the right
cb.set_label('Predicción Matriculas', size=16,)
# plotting also the observations
plt.scatter(data['Inscritos (1ra Opción)'], data['Admitidos (1ra y 2da  
↳Opción)'],
            alpha=0.8, s=5*data['Matriculados'],
            c=range(0, len(data)), cmap='turbo')

cbar = plt.colorbar(scalarmappable, orientation='vertical', pad=0.02,
                    ticks=range(0, deps.max()+1, 1), shrink=0.75)
cbar.ax.set_yticklabels(data['Programa'])

plt.subplots_adjust(left=0.075, bottom=0.075, right=1.05, top=0.925)
plt.show()
```

```
fig.savefig('Predicciones_'+str(year)+'.pdf', dpi=500)
plt.close(fig)
```

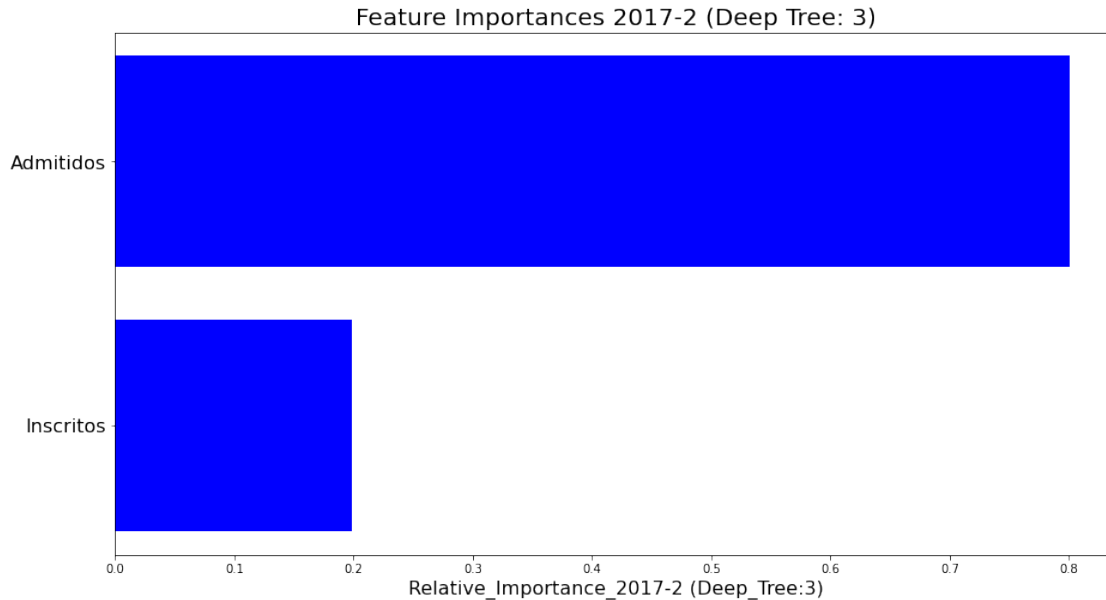


```
[11]: Importances = regressor.feature_importances_

Features = ['Inscritos', 'Admitidos']
Indices = np.argsort(Importances)

fig = plt.figure()
plt.title('Feature Importances '+str(year)+' (Deep Tree: '+str(depmax)+')',
         ↪fontsize=20)
plt.barh(range(len(Indices)), Importances[Indices], color='b', align='center')
plt.yticks(range(len(Indices)), [Features[i] for i in Indices], fontsize=16)
plt.xlabel('Relative_Importance_'+str(year)+' (Deep_Tree:'+str(depmax)+')',
         ↪fontsize=16)
plt.show()
fig.savefig('Importances_'+str(year)+'.pdf', dpi=500)
plt.close(fig)
```





```
[12]: X1_Test = np.array(test['Inscritos (1ra Opción)'])
      X2_Test = np.array(test['Admitidos (1ra y 2da Opción)'])

      Y_real = np.array(test['Matriculados'])
      Y_Pred = []

      for i in range(len(test)):
          y0 = regressor.predict(np.array([[X1_Test[i], X2_Test[i]]]))
          Y_Pred.append(y0[0])

      Y_diff = np.abs(Y_real - Y_Pred) #/np.array(Y_real)
```

```
/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
  warnings.warn(
/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
  warnings.warn(
/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
  warnings.warn(
/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
  warnings.warn(
```







```

/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
    warnings.warn(
/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
    warnings.warn(
/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
    warnings.warn(
/home/john/.local/lib/python3.8/site-packages/sklearn/base.py:450: UserWarning:
X does not have valid feature names, but DecisionTreeRegressor was fitted with
feature names
    warnings.warn(

```

```

[13]: plt.rcParams["figure.figsize"] = [12, 8]

# setup the normalization and the colormap
deps = np.arange(0, len(test['Programa']))
normalize = mcolors.Normalize(vmin = deps.min(), vmax = deps.max())
colormap = cm.get_cmap('turbo', deps.max()+1)
# setup the colorbar
scalarmappable = cm.ScalarMappable(norm=normalize, cmap=colormap)
scalarmappable.set_array(deps)

fig = plt.figure()

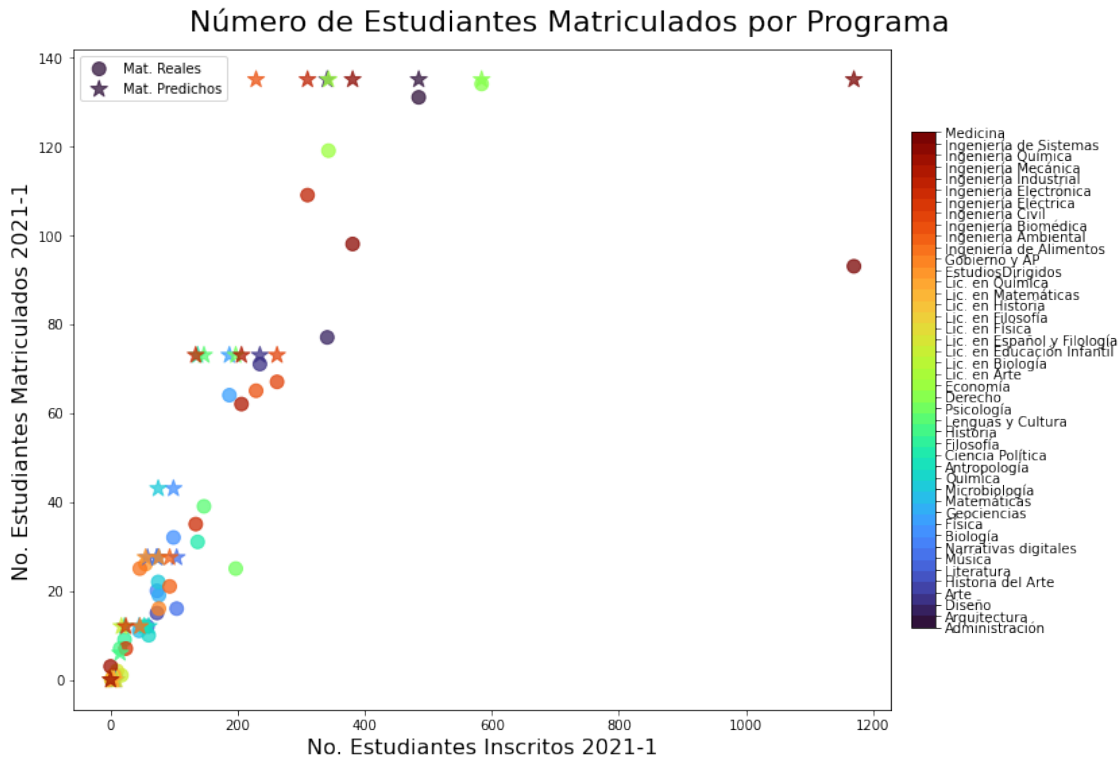
fig.suptitle('Número de Estudiantes Matriculados por Programa', size=22)
plt.xlabel('No. Estudiantes Inscritos 2021-1', fontsize=16)
plt.ylabel('No. Estudiantes Matriculados 2021-1', fontsize=16)

plt.scatter(X1_Test, Y_real, label='Mat. Reales',
            alpha=0.75, s=100, c=range(0, len(test)), cmap='turbo')
plt.scatter(X1_Test, Y_Pred, label='Mat. Predichos',
            alpha=0.75, marker='*', s=170, c=range(0, len(test)), cmap='turbo')
plt.legend()

cbar = plt.colorbar(scalarmappable, orientation='vertical', pad=0.02,
                    ticks=range(0, deps.max()+1, 1), shrink=0.75)
cbar.ax.set_yticklabels(test['Programa'])

plt.subplots_adjust(left=0.075, bottom=0.075, right=0.92, top=0.925)
plt.show()
fig.savefig('2021-1_Matriculados_por_Ins_'+str(year)+'.pdf', dpi=500)
plt.close(fig)

```



```
[14]: plt.rcParams["figure.figsize"] = [12, 8]

# setup the normalization and the colormap
deps = np.arange(0, len(test['Programa']))
normalize = mcolors.Normalize(vmin = deps.min(), vmax = deps.max())
colormap = cm.get_cmap('turbo', deps.max()+1)
# setup the colorbar
scalarmappable = cm.ScalarMappable(norm=normalize, cmap=colormap)
scalarmappable.set_array(deps)

fig = plt.figure()

fig.suptitle('Número de Estudiantes Matriculados por Programa', size=22)
plt.xlabel('No. Estudiantes Inscritos 2021-1', fontsize=16)
plt.ylabel('No. Estudiantes Matriculados 2021-1', fontsize=16)

plt.scatter(X2_Test, Y_real, label='Mat. Reales',
            alpha=0.75, s=100, c=range(0, len(test)), cmap='turbo')
plt.scatter(X2_Test, Y_Pred, label='Mat. Predichos',
            alpha=0.75, marker='*', s=170, c=range(0, len(test)), cmap='turbo')
plt.legend()

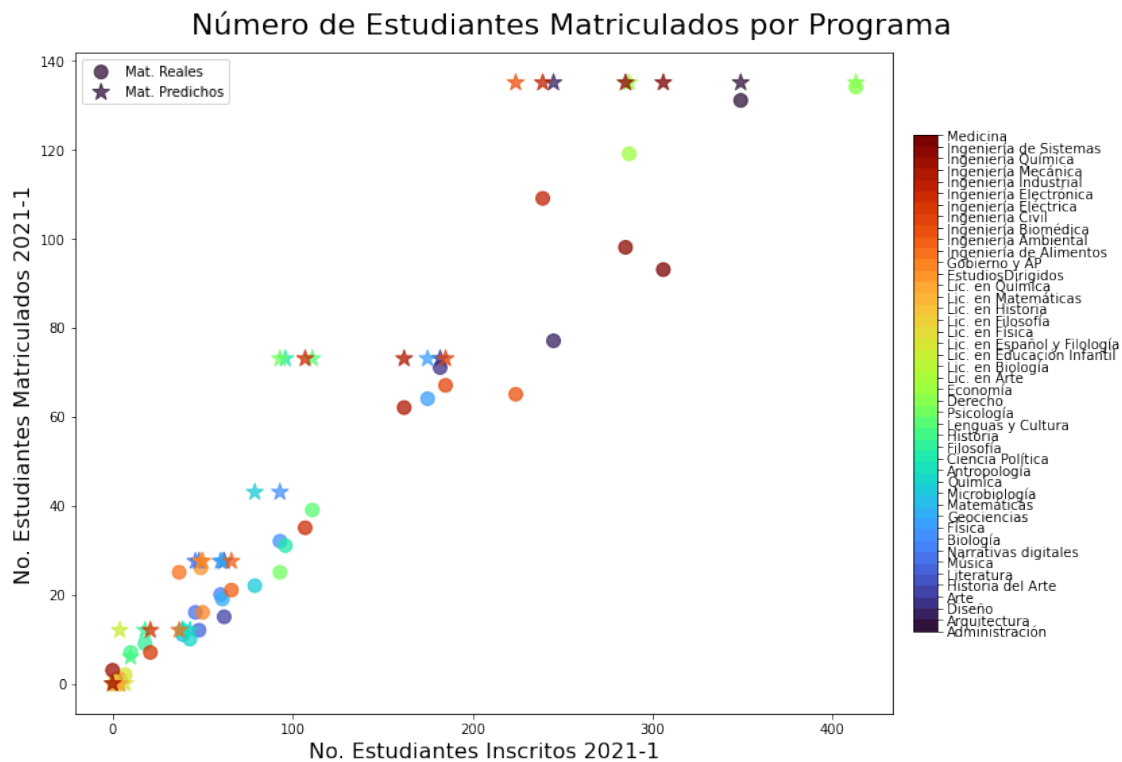
cbar = plt.colorbar(scalarmappable, orientation='vertical', pad=0.02,
```

```

        ticks=range(0, deps.max()+1, 1), shrink=0.75)
cbar.ax.set_yticklabels(test['Programa'])

plt.subplots_adjust(left=0.075, bottom=0.075, right=0.92, top=0.925)
plt.show()
fig.savefig('2021-1_Matriculados_por_Adm_'+str(year)+'.pdf', dpi=500)
plt.close(fig)

```



```

[15]: Comparacion = test.filter(items=['Programa',
                                     'Inscritos (1ra Opción)',
                                     'Admitidos (1ra y 2da Opción)',
                                     'Matriculados'])

Y_Pred = pd.DataFrame(Y_Pred, columns=['Mat. Predichos'])
Y_diff = pd.DataFrame(Y_diff, columns=['Diferencia'])

Comparacion = pd.concat([Comparacion, Y_Pred], axis=1)

Comparacion = pd.concat([Comparacion, Y_diff], axis=1)

Comparacion

```

[15]:	Programa	Inscritos (1ra Opción)	\
0	Administración	485.0	
1	Arquitectura	341.0	
2	Diseño	235.0	
3	Arte	73.0	
4	Historia del Arte	8.0	
5	Literatura	58.0	
6	Música	104.0	
7	Narrativas digitales	73.0	
8	Biología	99.0	
9	Física	187.0	
10	Geociencias	76.0	
11	Matemáticas	45.0	
12	Microbiología	75.0	
13	Química	60.0	
14	Antropología	53.0	
15	Ciencia Política	137.0	
16	Filosofía	22.0	
17	Historia	15.0	
18	Lenguas y Cultura	147.0	
19	Psicología	197.0	
20	Derecho	584.0	
21	Economía	343.0	
22	Lic. en Arte	6.0	
23	Lic. en Biología	2.0	
24	Lic. en Educación Infantil	17.0	
25	Lic. en Español y Filología	10.0	
26	Lic. en Física	2.0	
27	Lic. en Filosofía	1.0	
28	Lic. en Historia	3.0	
29	Lic. en Matemáticas	5.0	
30	Lic. en Química	5.0	
31	EstudiosDirigidos	55.0	
32	Gobierno y AP	76.0	
33	Ingeniería de Alimentos	46.0	
34	Ingeniería Ambiental	93.0	
35	Ingeniería Biomédica	229.0	
36	Ingeniería Civil	262.0	
37	Ingeniería Eléctrica	24.0	
38	Ingeniería Electrónica	134.0	
39	Ingeniería Industrial	310.0	
40	Ingeniería Mecánica	206.0	
41	Ingeniería Química	0.0	
42	Ingeniería de Sistemas	381.0	
43	Medicina	1170.0	

Admitidos (1ra y 2da Opción)    Matriculados    Mat. Predichos    Diferencia



0	349.0	131.0	135.0	4.0
1	245.0	77.0	135.0	58.0
2	182.0	71.0	73.0	2.0
3	62.0	15.0	27.5	12.5
4	4.0	1.0	0.0	1.0
5	48.0	12.0	27.5	15.5
6	46.0	16.0	27.5	11.5
7	60.0	20.0	27.5	7.5
8	93.0	32.0	43.0	11.0
9	175.0	64.0	73.0	9.0
10	61.0	19.0	27.5	8.5
11	39.0	11.0	12.0	1.0
12	79.0	22.0	43.0	21.0
13	43.0	10.0	12.0	2.0
14	39.0	12.0	12.0	0.0
15	96.0	31.0	73.0	42.0
16	18.0	9.0	12.0	3.0
17	10.0	7.0	6.0	1.0
18	111.0	39.0	73.0	34.0
19	93.0	25.0	73.0	48.0
20	413.0	134.0	135.0	1.0
21	287.0	119.0	135.0	16.0
22	3.0	1.0	0.0	1.0
23	1.0	0.0	0.0	0.0
24	4.0	1.0	12.0	11.0
25	7.0	2.0	0.0	2.0
26	2.0	0.0	0.0	0.0
27	1.0	0.0	0.0	0.0
28	2.0	0.0	0.0	0.0
29	2.0	1.0	0.0	1.0
30	4.0	1.0	0.0	1.0
31	49.0	26.0	27.5	1.5
32	50.0	16.0	27.5	11.5
33	37.0	25.0	12.0	13.0
34	66.0	21.0	27.5	6.5
35	224.0	65.0	135.0	70.0
36	185.0	67.0	73.0	6.0
37	21.0	7.0	12.0	5.0
38	107.0	35.0	73.0	38.0
39	239.0	109.0	135.0	26.0
40	162.0	62.0	73.0	11.0
41	0.0	3.0	0.0	3.0
42	285.0	98.0	135.0	37.0
43	306.0	93.0	135.0	42.0

```
[16]: Comparacion.sort_values(by=['Diferencia'])
```

[16]:	Programa	Inscritos (1ra Opción)	\
26	Lic. en Física	2.0	
14	Antropología	53.0	
23	Lic. en Biología	2.0	
28	Lic. en Historia	3.0	
27	Lic. en Filosofía	1.0	
20	Derecho	584.0	
22	Lic. en Arte	6.0	
11	Matemáticas	45.0	
17	Historia	15.0	
29	Lic. en Matemáticas	5.0	
4	Historia del Arte	8.0	
30	Lic. en Química	5.0	
31	EstudiosDirigidos	55.0	
25	Lic. en Español y Filología	10.0	
13	Química	60.0	
2	Diseño	235.0	
41	Ingeniería Química	0.0	
16	Filosofía	22.0	
0	Administración	485.0	
37	Ingeniería Eléctrica	24.0	
36	Ingeniería Civil	262.0	
34	Ingeniería Ambiental	93.0	
7	Narrativas digitales	73.0	
10	Geociencias	76.0	
9	Física	187.0	
40	Ingeniería Mecánica	206.0	
24	Lic. en Educación Infantil	17.0	
8	Biología	99.0	
6	Música	104.0	
32	Gobierno y AP	76.0	
3	Arte	73.0	
33	Ingeniería de Alimentos	46.0	
5	Literatura	58.0	
21	Economía	343.0	
12	Microbiología	75.0	
39	Ingeniería Industrial	310.0	
18	Lenguas y Cultura	147.0	
42	Ingeniería de Sistemas	381.0	
38	Ingeniería Electrónica	134.0	
15	Ciencia Política	137.0	
43	Medicina	1170.0	
19	Psicología	197.0	
1	Arquitectura	341.0	
35	Ingeniería Biomédica	229.0	

Admitidos (1ra y 2da Opción)    Matriculados    Mat. Predichos    Diferencia

26	2.0	0.0	0.0	0.0
14	39.0	12.0	12.0	0.0
23	1.0	0.0	0.0	0.0
28	2.0	0.0	0.0	0.0
27	1.0	0.0	0.0	0.0
20	413.0	134.0	135.0	1.0
22	3.0	1.0	0.0	1.0
11	39.0	11.0	12.0	1.0
17	10.0	7.0	6.0	1.0
29	2.0	1.0	0.0	1.0
4	4.0	1.0	0.0	1.0
30	4.0	1.0	0.0	1.0
31	49.0	26.0	27.5	1.5
25	7.0	2.0	0.0	2.0
13	43.0	10.0	12.0	2.0
2	182.0	71.0	73.0	2.0
41	0.0	3.0	0.0	3.0
16	18.0	9.0	12.0	3.0
0	349.0	131.0	135.0	4.0
37	21.0	7.0	12.0	5.0
36	185.0	67.0	73.0	6.0
34	66.0	21.0	27.5	6.5
7	60.0	20.0	27.5	7.5
10	61.0	19.0	27.5	8.5
9	175.0	64.0	73.0	9.0
40	162.0	62.0	73.0	11.0
24	4.0	1.0	12.0	11.0
8	93.0	32.0	43.0	11.0
6	46.0	16.0	27.5	11.5
32	50.0	16.0	27.5	11.5
3	62.0	15.0	27.5	12.5
33	37.0	25.0	12.0	13.0
5	48.0	12.0	27.5	15.5
21	287.0	119.0	135.0	16.0
12	79.0	22.0	43.0	21.0
39	239.0	109.0	135.0	26.0
18	111.0	39.0	73.0	34.0
42	285.0	98.0	135.0	37.0
38	107.0	35.0	73.0	38.0
15	96.0	31.0	73.0	42.0
43	306.0	93.0	135.0	42.0
19	93.0	25.0	73.0	48.0
1	245.0	77.0	135.0	58.0
35	224.0	65.0	135.0	70.0

```
[17]: x = Comparacion.index
      y = Comparacion['Mat. Predichos']
```

```

dy = Comparacion['Diferencia']

plt.rcParams["figure.figsize"] = [12, 8]

# setup the normalization and the colormap
deps = np.arange(0, len(test['Programa']))
normalize = mcolors.Normalize(vmin = deps.min(), vmax = deps.max())
colormap = cm.get_cmap('turbo', deps.max()+1)
# setup the colorbar
scalarmappable = cm.ScalarMappable(norm=normalize, cmap=colormap)
scalarmappable.set_array(deps)

fig = plt.figure()

plt.xticks([])

fig.suptitle('Número de Estudiantes Matriculados por Programa', size=22)
plt.xlabel('Programa Ofertado', fontsize=16)
plt.ylabel('No. Estudiantes Matriculados 2021-1', fontsize=16)

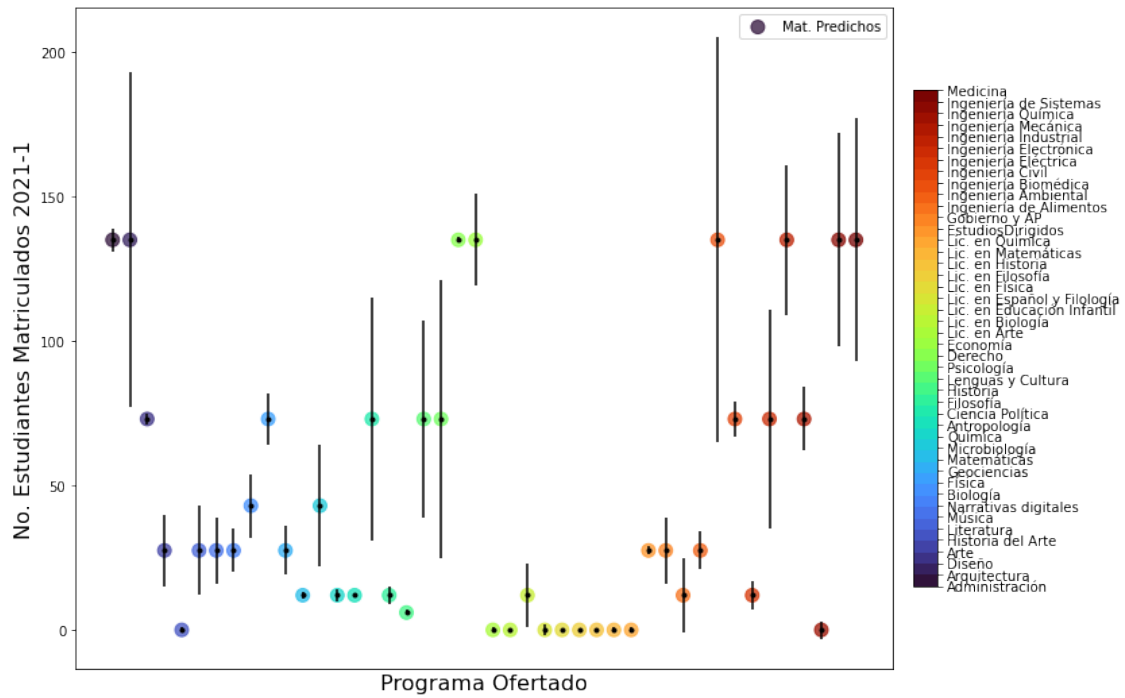
plt.errorbar(x, y, yerr=dy, fmt='.k')
plt.scatter(x, y, label='Mat. Predichos',
            alpha=0.75, s=100, c=range(0, len(test)), cmap='turbo')
plt.legend()

cbar = plt.colorbar(scalarmappable, orientation='vertical', pad=0.02,
                    ticks=range(0, deps.max()+1, 1), shrink=0.75)
cbar.ax.set_yticklabels(test['Programa'])

plt.subplots_adjust(left=0.075, bottom=0.075, right=0.92, top=0.925)
plt.show()
fig.savefig('2021-1_Matriculados_Diff_'+str(year)+'.pdf', dpi=500)
plt.close(fig)

```

## Número de Estudiantes Matriculados por Programa



[ ]:

[ ]: