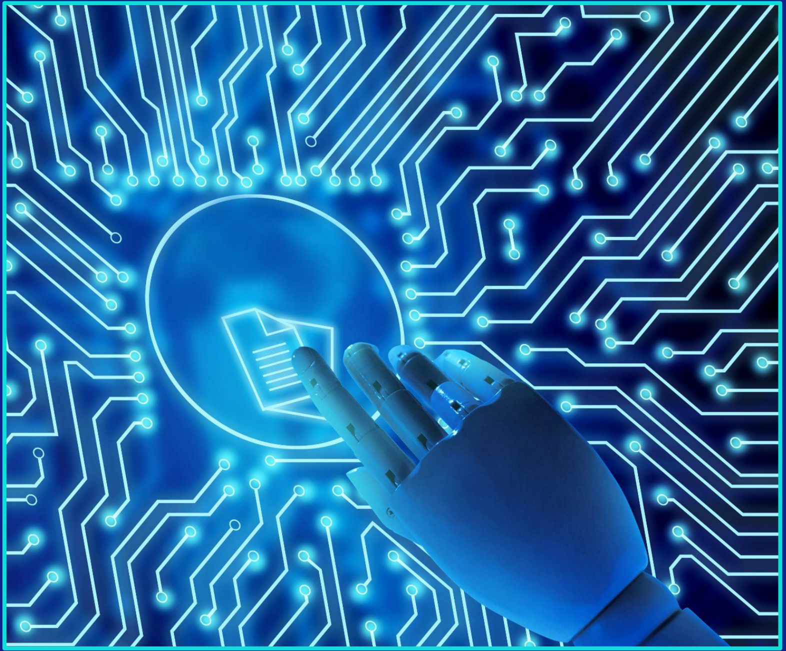
Machine Learning

Squad Rapidminer

Francisco Ávila

Rubén Tenreiro

Marco Hernani



UCI

Machine Learning Repository

Center for Machine Learning and Intelligent Systems

INDICE

01 Introducción

02 EDA

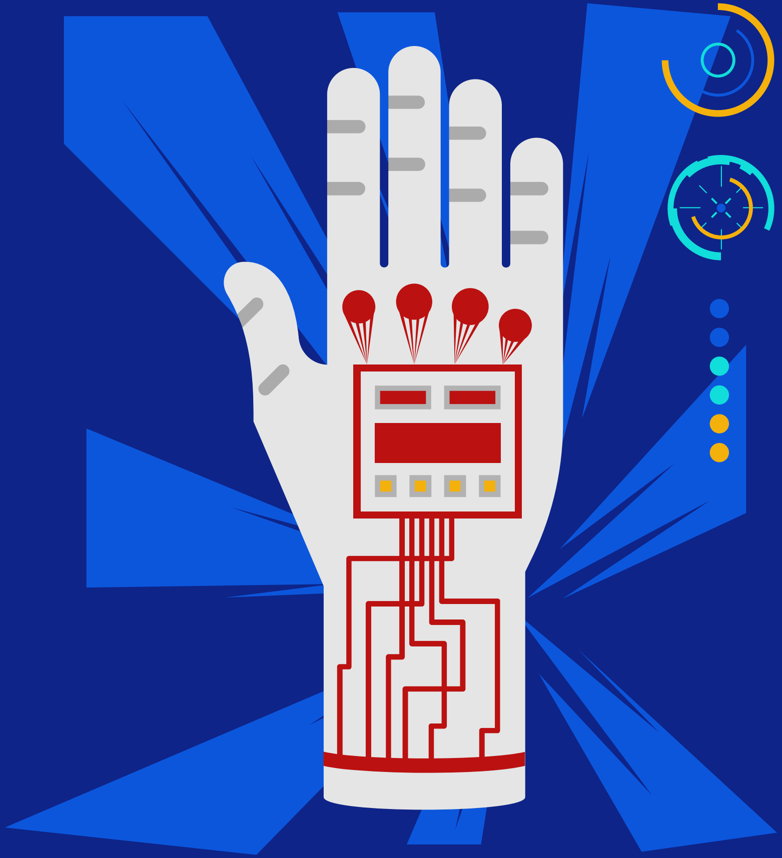
03 Preprocesado

04 Modelos

https://archive.ics.uci.edu/ml/datasets/Census+Income

01

Introducción



Variables cuantitativas

age	Edad
fnlwgt	Número que el censo cree que la entrada representa
education_num	Un número relacionado con el nivel de educación (sin un orden específico)
capital_gain	Capital ganado con inversiones
capital_loss	Capital perdido en inversiones
hours_per_week	Horas trabajadas por semanas

Variables cualitativas

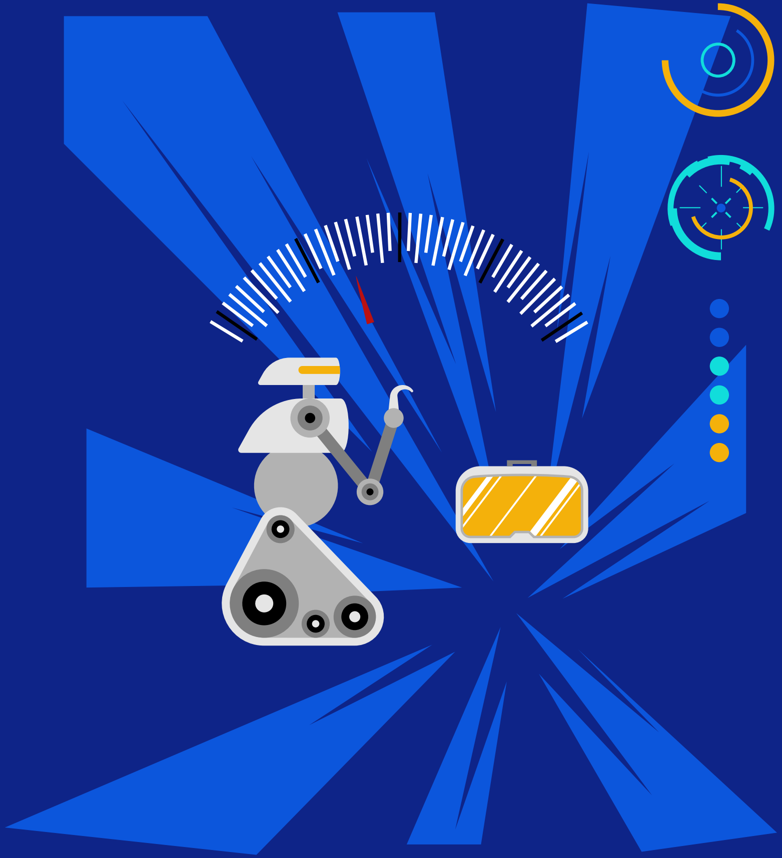
work_class	Clase en la que se categoriza el trabajo de la persona
education	Nivel más alto de educación de la persona
marital_status	Estado civil de la persona
occupation	Trabajo de la persona
relationship	Tipo de relación que tiene esa persona
race	Raza de la persona
native_country	País de nacimiento de la persona
sex	Sexo de la persona

Variable objetivo

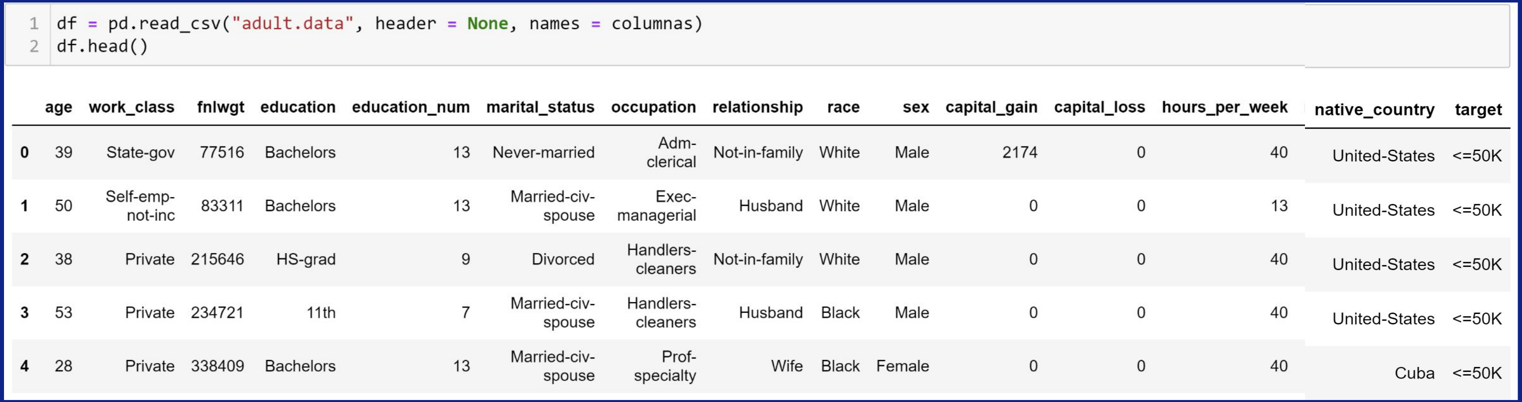
target	Variable objetivo que contiene un valor binario en función de si una persona gana o no más de 50K

02

EDA



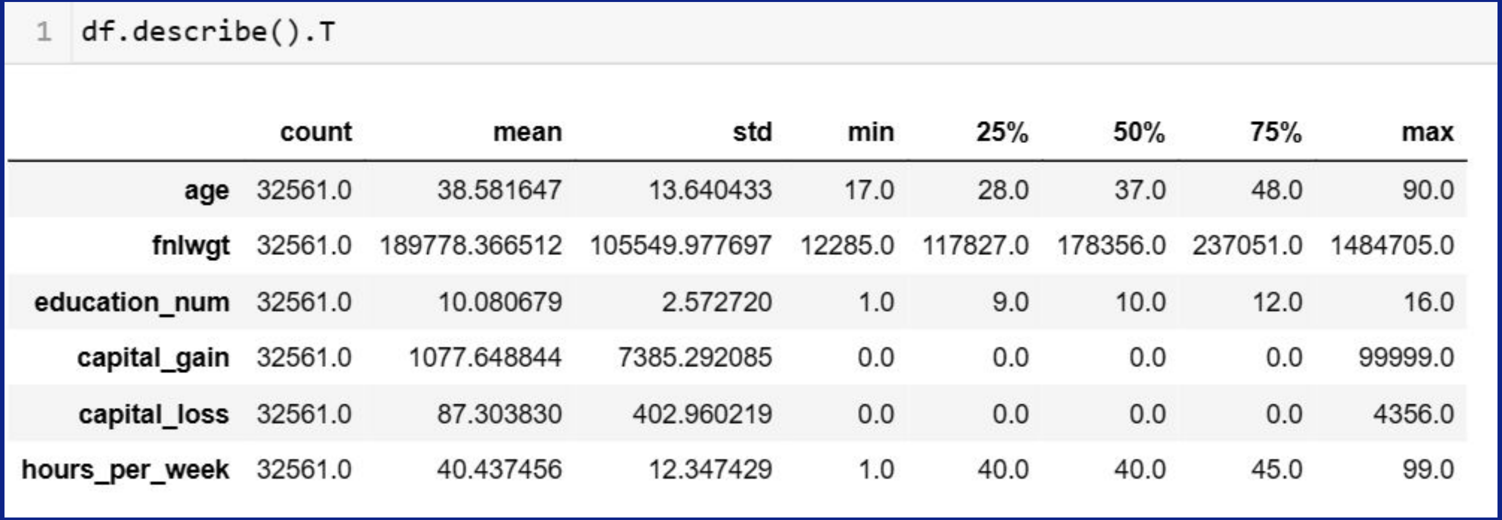
Dataframe



```python
df = pd.read_csv("adult.data", header = None, names = columnas)
df.head()
```

	age	work_class	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country	target
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K

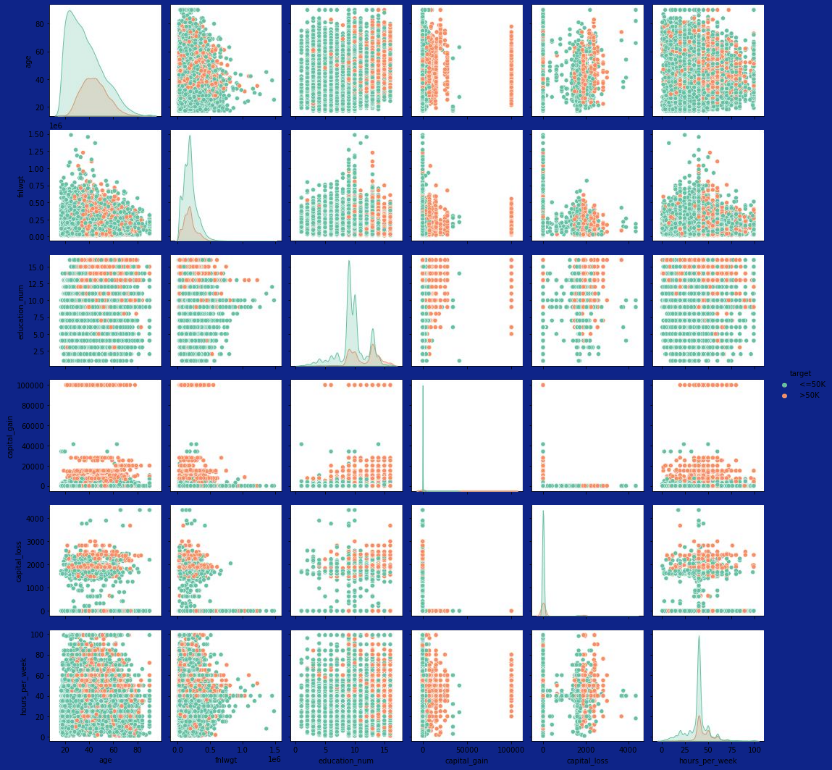
Resumen variables numéricas

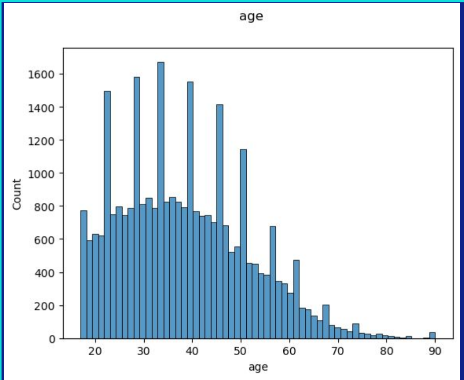


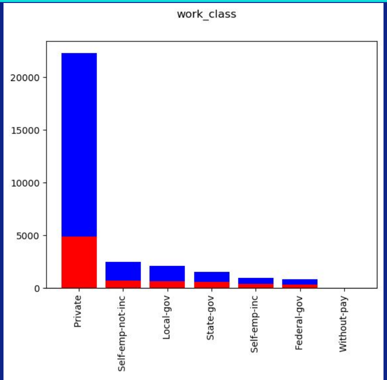
```python
df.describe().T
```

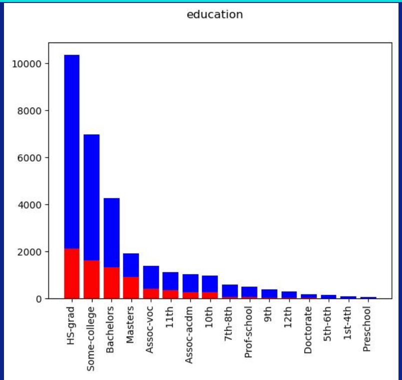
	count	mean	std	min	25%	50%	75%	max
age	32561.0	38.581647	13.640433	17.0	28.0	37.0	48.0	90.0
fnlwgt	32561.0	189778.366512	105549.977697	12285.0	117827.0	178356.0	237051.0	1484705.0
education_num	32561.0	10.080679	2.572720	1.0	9.0	10.0	12.0	16.0
capital_gain	32561.0	1077.648844	7385.292085	0.0	0.0	0.0	0.0	99999.0
capital_loss	32561.0	87.303830	402.960219	0.0	0.0	0.0	0.0	4356.0
hours_per_week	32561.0	40.437456	12.347429	1.0	40.0	40.0	45.0	99.0

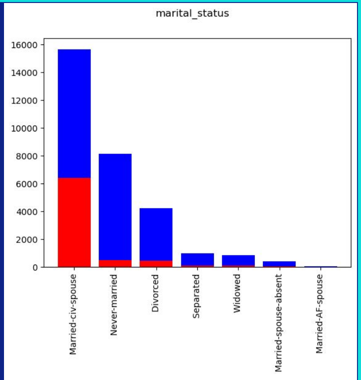
Scatter Matrix

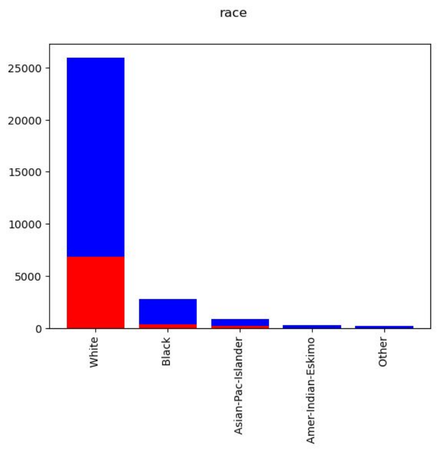


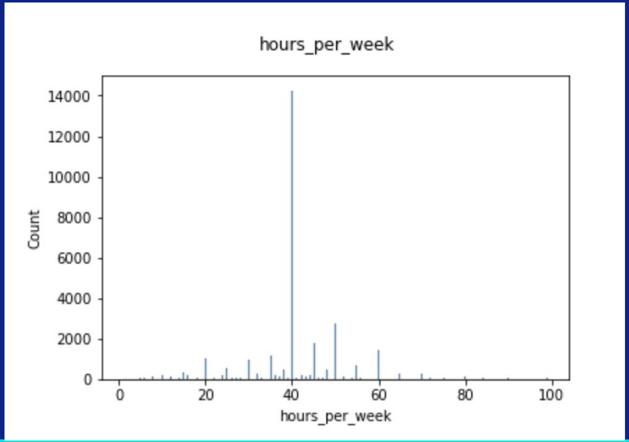

age

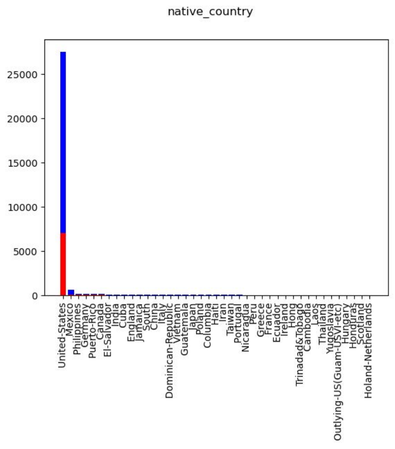

work_class


education


marital_status


race


hours_per_week


native_country

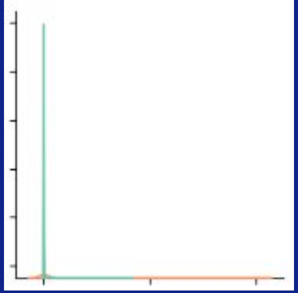
EDA 02

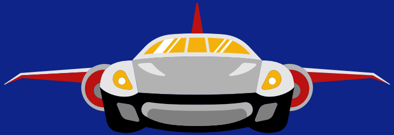
03

Preprocesado

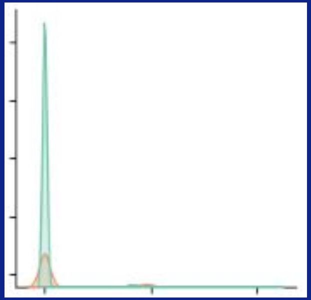
Eliminamos

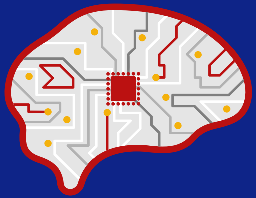
capital_gain



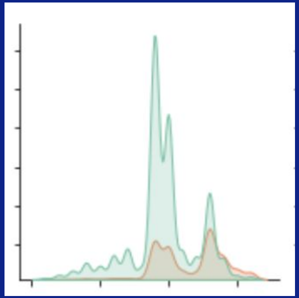


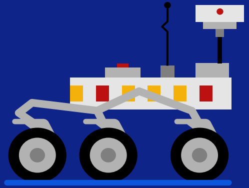
capital_loss



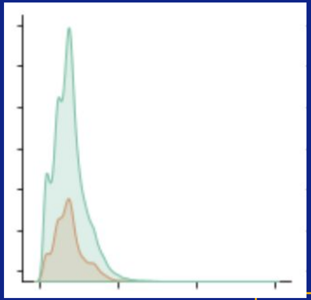


education_num



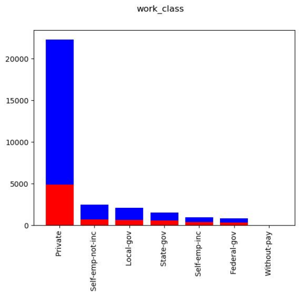


fnlwgt



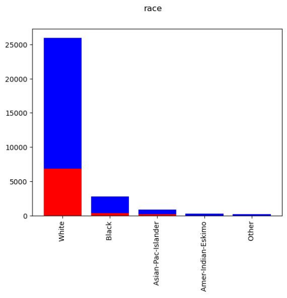
Agrupamos

work_class



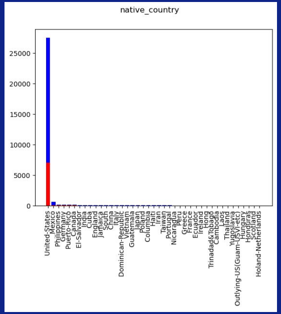
"Private", "Other"

race



"White", "Other"

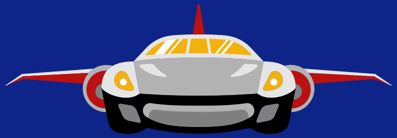
native_country



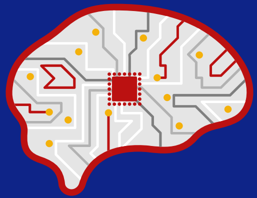
"United-States", "Other"

OneHotEncodeamos

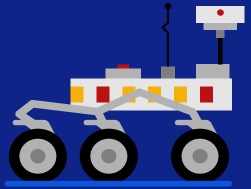
education



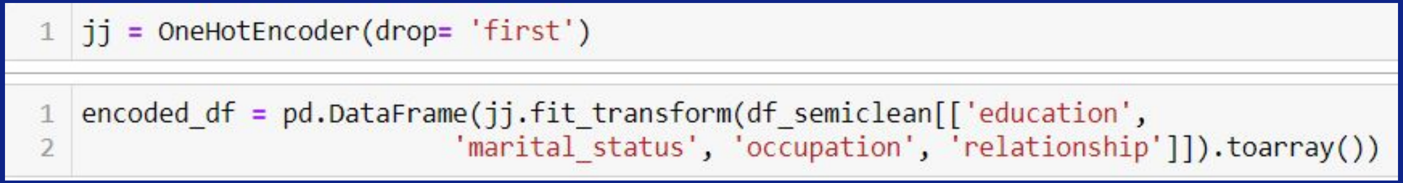
marital_status



occupation



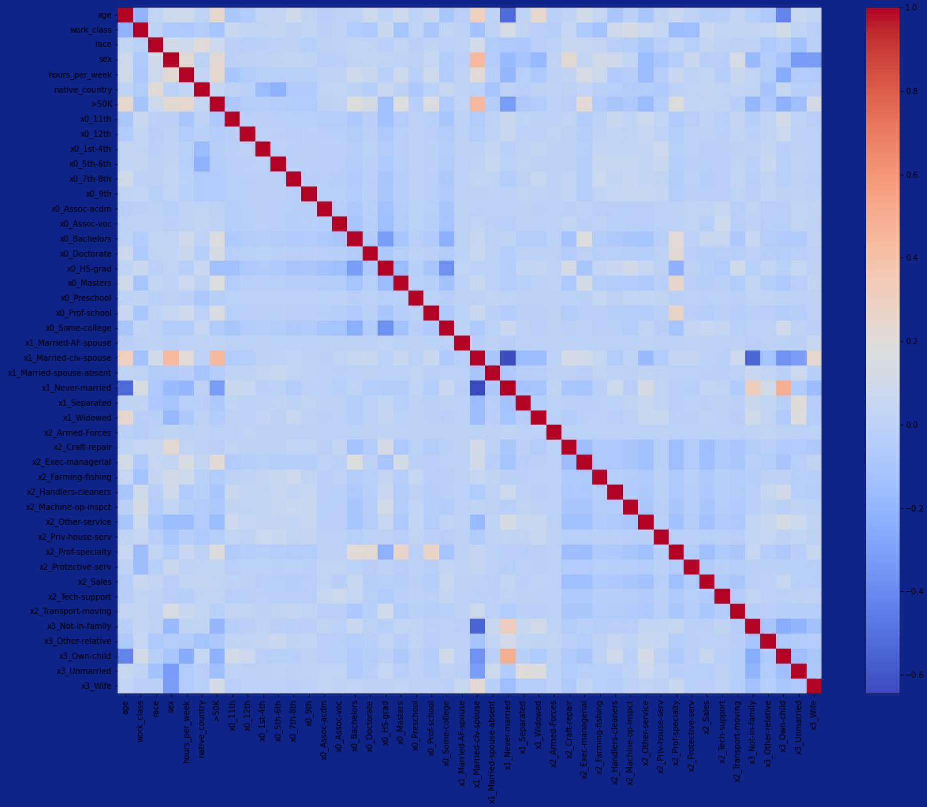
relationship



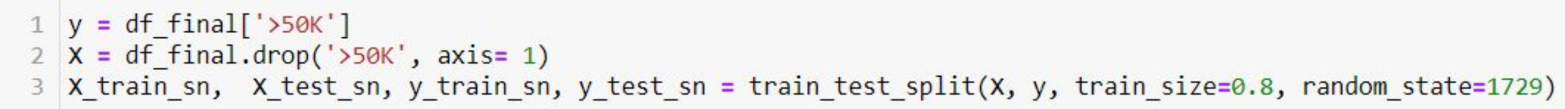
```
jj = OneHotEncoder(drop= 'first')
```

```
encoded_df = pd.DataFrame(jj.fit_transform(df_semiclean[['education',
                                 'marital_status', 'occupation', 'relationship']]).toarray())
```

Matriz de correlación

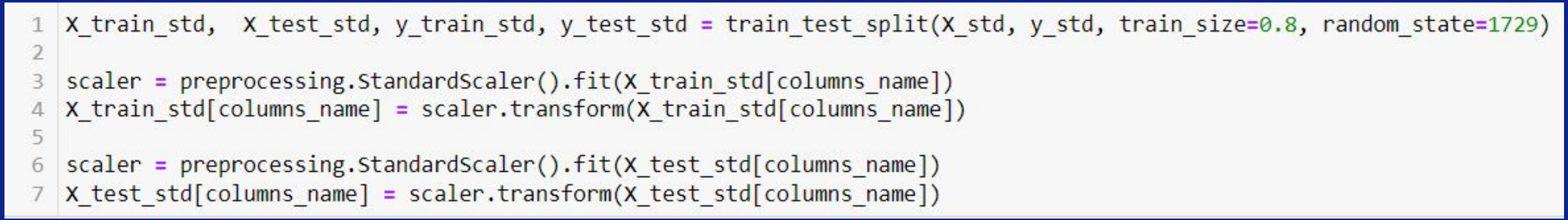


Train y Test split



```
y = df_final['>50K']
X = df_final.drop('>50K', axis= 1)
X_train_sn,  X_test_sn, y_train_sn, y_test_sn = train_test_split(X, y, train_size=0.8, random_state=1729)
```

StandardScaler()



```
X_train_std,  X_test_std, y_train_std, y_test_std = train_test_split(X_std, y_std, train_size=0.8, random_state=1729)

scaler = preprocessing.StandardScaler().fit(X_train_std[columns_name])
X_train_std[columns_name] = scaler.transform(X_train_std[columns_name])

scaler = preprocessing.StandardScaler().fit(X_test_std[columns_name])
X_test_std[columns_name] = scaler.transform(X_test_std[columns_name])
```

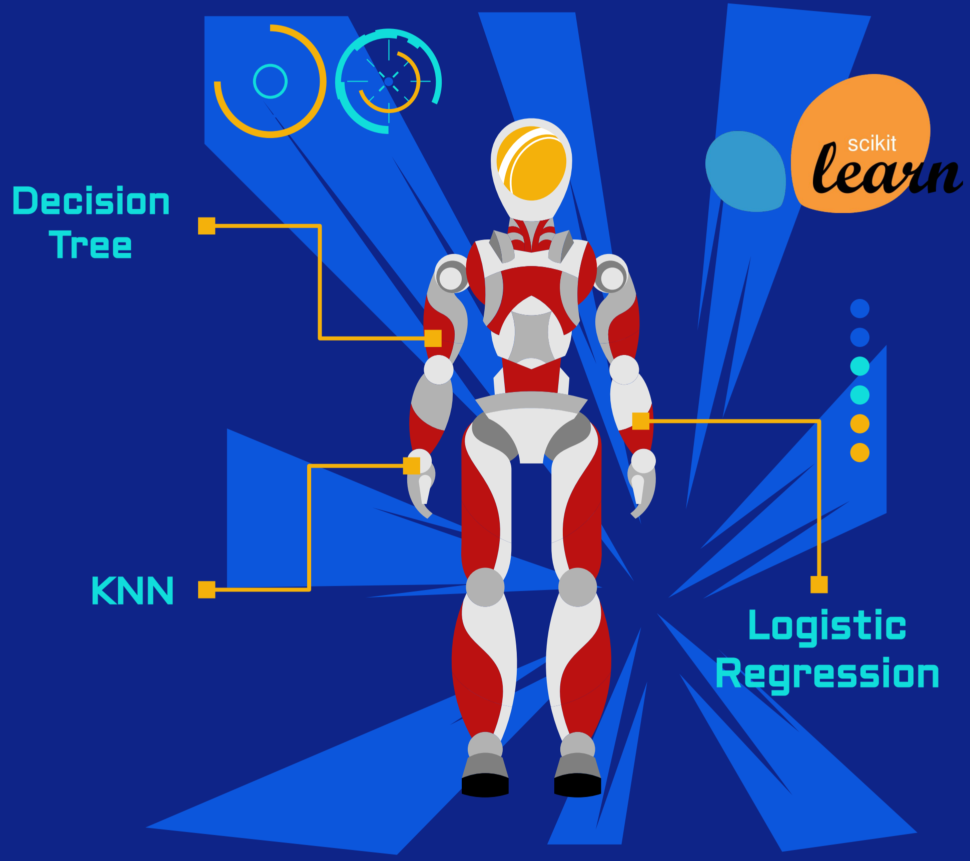
Dos datasets

Sin estandarizar

Estandarizado

04

Modelos



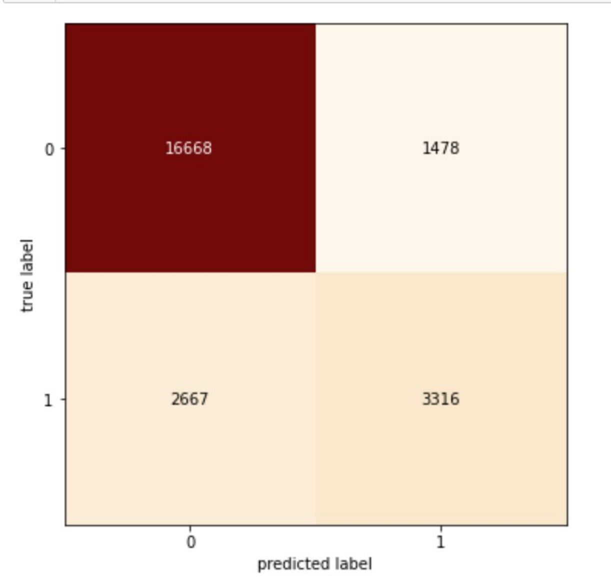
Decision Tree

KNN

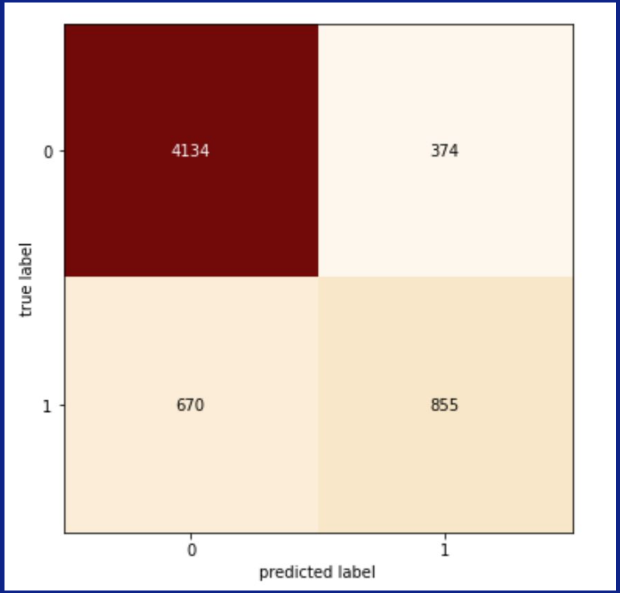
Logistic Regression

Regresión Logística

Regresión logística SN train

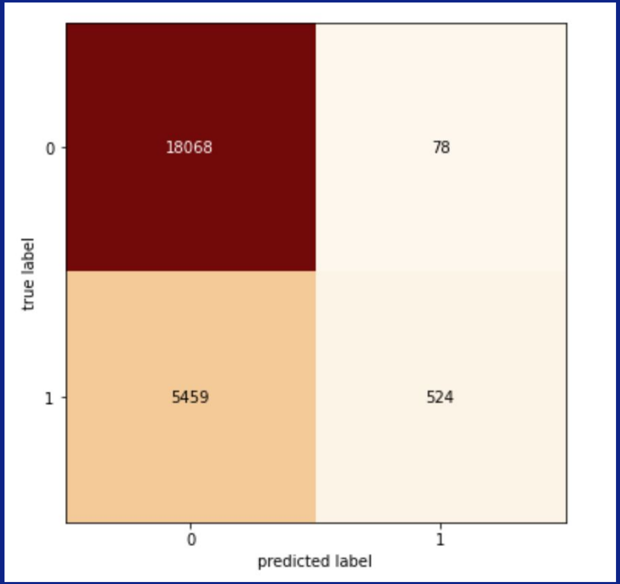


Regresión logística SN test

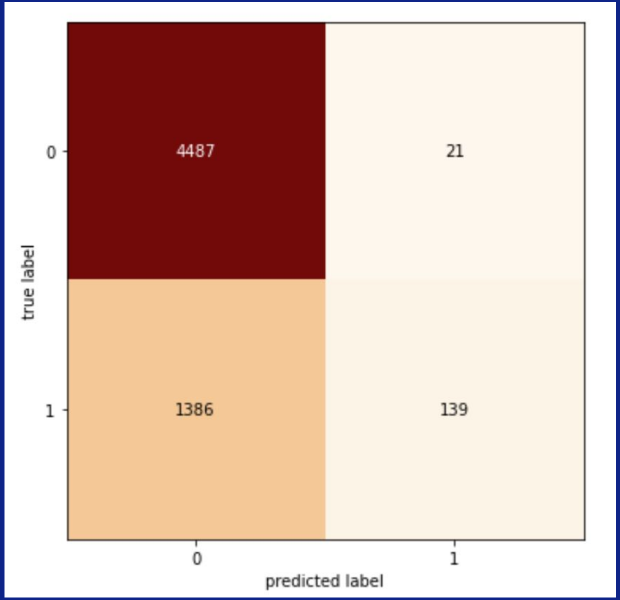


Regresión Logística

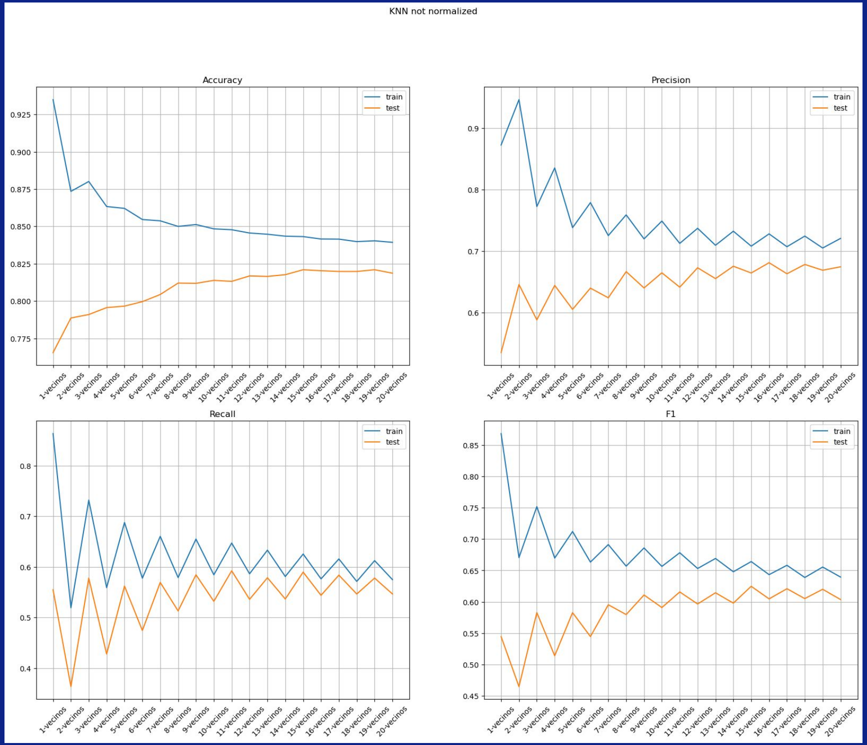
Regresión logística STD train



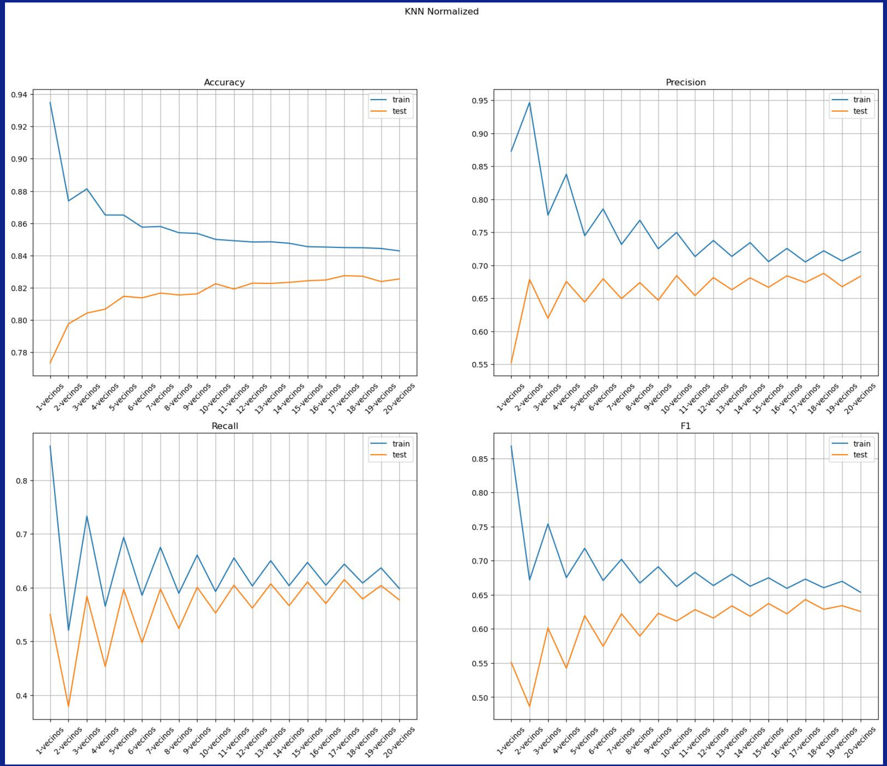
Regresión logística STD test



KNN not normalized



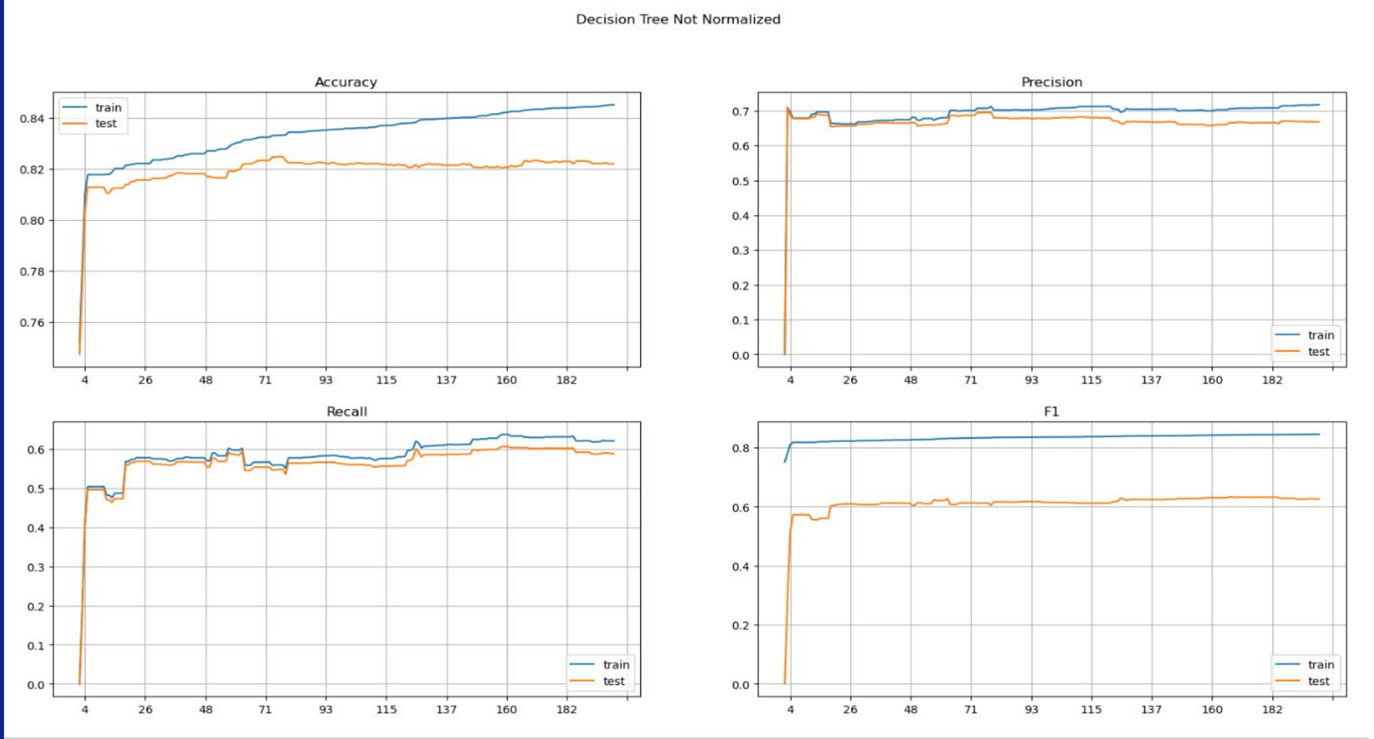
KNN Normalized



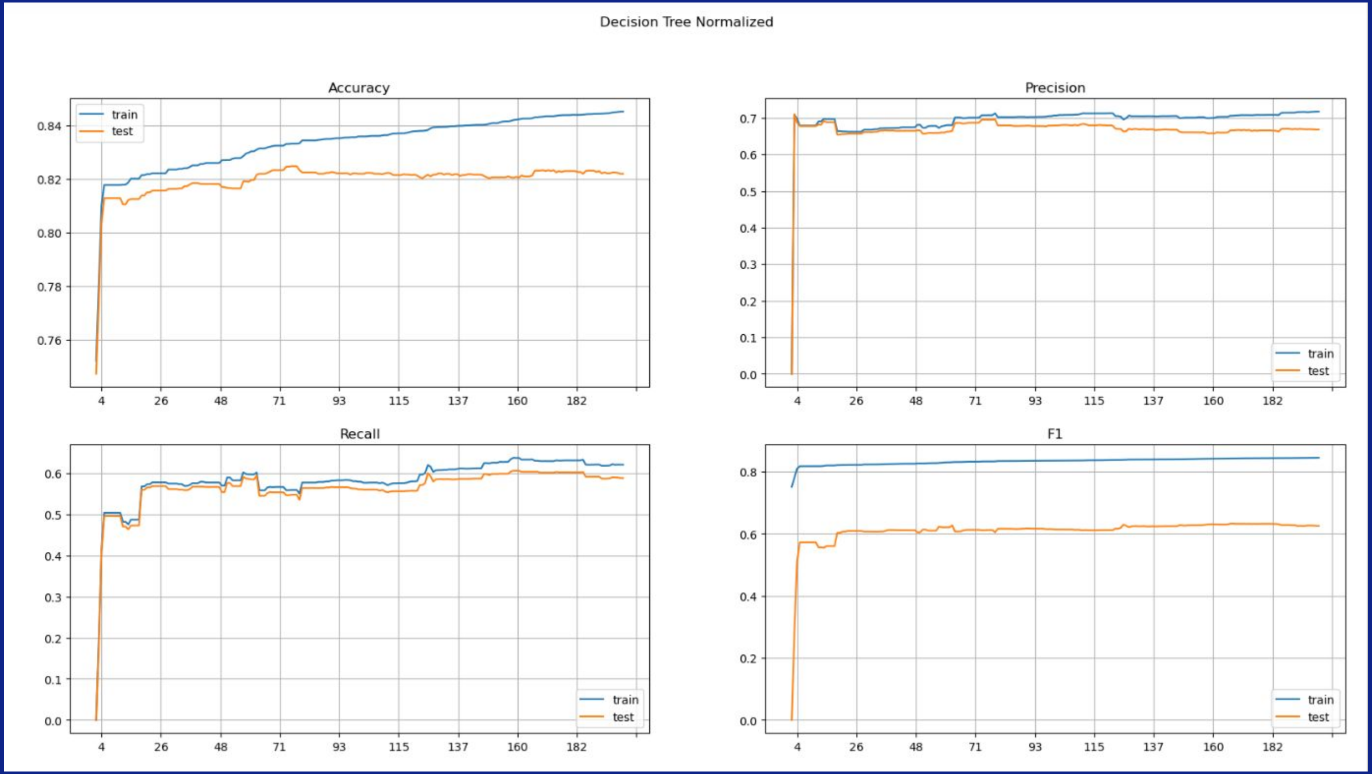
Modelos

04

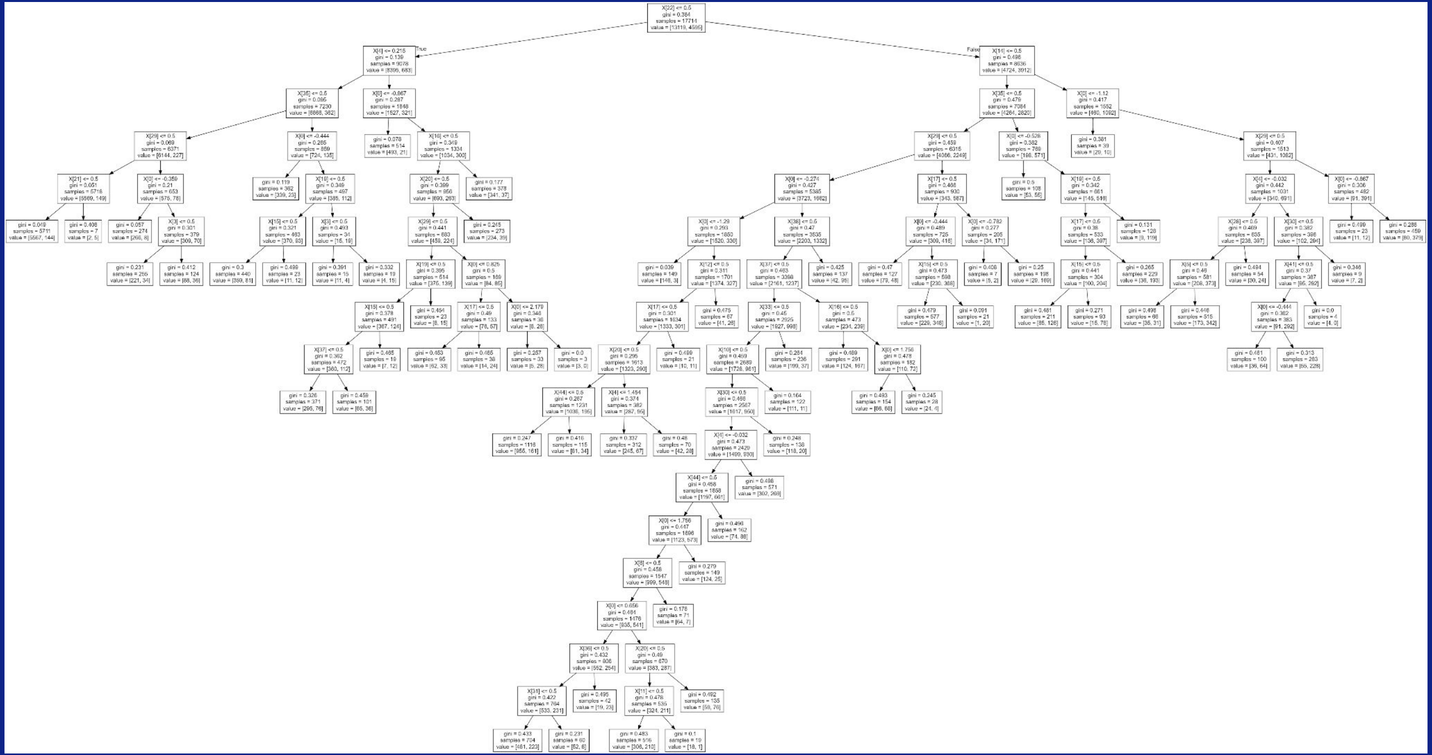
Decision Tree not normalized



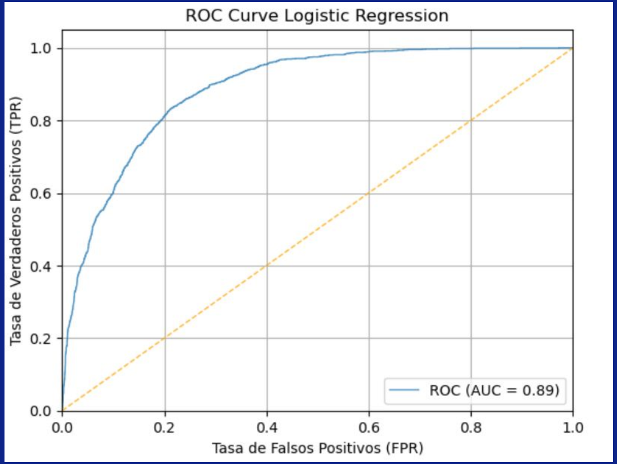
Decision Tree normalized

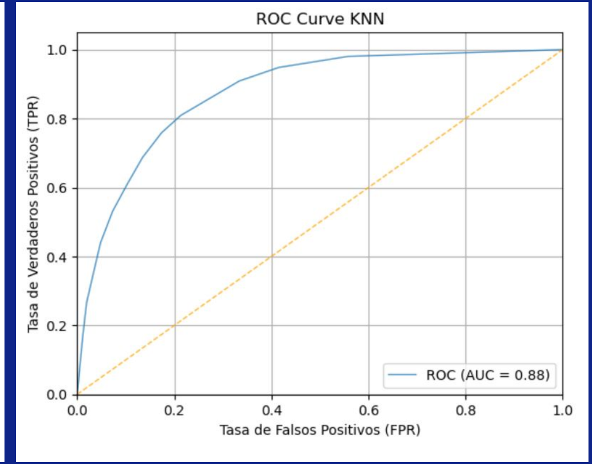


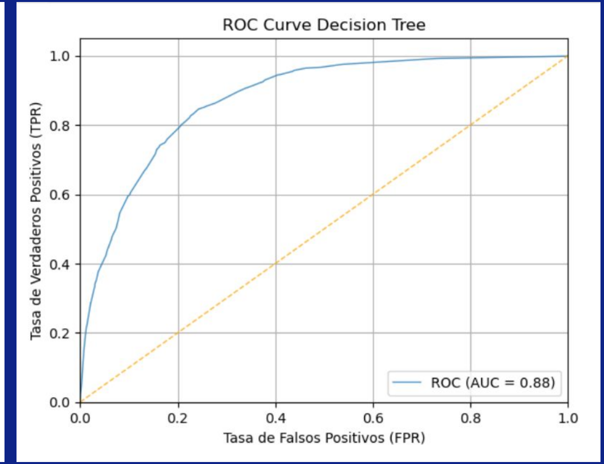
Decision Tree


X[22] <= 0.5 gini = 0.384 samples = 17714 value = [13119, 4595]
True
False
X[4] <= 0.215 gini = 0.139 samples = 9078 value = [8395, 683]
X[14] <= 0.5 gini = 0.496 samples = 8636 value = [4724, 3912]
X[35] <= 0.5 gini = 0.086 samples = 7230 value = [6868, 362]
X[0] <= -0.867 gini = 0.287 samples = 1848 value = [1527, 321]
X[35] <= 0.5 gini = 0.479 samples = 7084 value = [4264, 2820]
X[0] <= -1.12 gini = 0.417 samples = 1552 value = [460, 1092]
X[29] <= 0.5 gini = 0.069 samples = 6371 value = [6144, 227]
X[0] <= -0.444 gini = 0.265 samples = 859 value = [724, 135]
gini = 0.078 samples = 514 value = [493, 21]
X[18] <= 0.5 gini = 0.349 samples = 1334 value = [1034, 300]
X[29] <= 0.5 gini = 0.459 samples = 6315 value = [4066, 2249]
X[0] <= -0.528 gini = 0.382 samples = 769 value = [198, 571]
gini = 0.381 samples = 39 value = [29, 10]
X[29] <= 0.5 gini = 0.407 samples = 1513 value = [431, 1082]
X[21] <= 0.5 gini = 0.051 samples = 5718 value = [5569, 149]
X[0] <= -0.359 gini = 0.21 samples = 653 value = [575, 78]
gini = 0.119 samples = 362 value = [339, 23]
X[19] <= 0.5 gini = 0.349 samples = 497 value = [385, 112]
X[20] <= 0.5 gini = 0.399 samples = 956 value = [693, 263]
gini = 0.177 samples = 378 value = [341, 37]
X[0] <= -0.274 gini = 0.427 samples = 5385 value = [3723, 1662]
X[17] <= 0.5 gini = 0.468 samples = 930 value = [343, 587]
gini = 0.5 samples = 108 value = [53, 55]
X[19] <= 0.5 gini = 0.342 samples = 661 value = [145, 516]
X[4] <= -0.032 gini = 0.442 samples = 1031 value = [340, 691]
X[0] <= -0.867 gini = 0.306 samples = 482 value = [91, 391]
gini = 0.049 samples = 5711 value = [5567, 144]
gini = 0.408 samples = 7 value = [2, 5]
gini = 0.057 samples = 274 value = [266, 8]
X[3] <= 0.5 gini = 0.301 samples = 379 value = [309, 70]
X[15] <= 0.5 gini = 0.321 samples = 463 value = [370, 93]
X[3] <= 0.5 gini = 0.493 samples = 34 value = [15, 19]
X[29] <= 0.5 gini = 0.441 samples = 683 value = [459, 224]
gini = 0.245 samples = 273 value = [234, 39]
X[0] <= -1.29 gini = 0.293 samples = 1850 value = [1520, 330]
X[38] <= 0.5 gini = 0.47 samples = 3535 value = [2203, 1332]
X[0] <= -0.444 gini = 0.489 samples = 725 value = [309, 416]
X[0] <= -0.782 gini = 0.277 samples = 205 value = [34, 171]
X[17] <= 0.5 gini = 0.38 samples = 533 value = [136, 397]
gini = 0.131 samples = 128 value = [9, 119]
X[28] <= 0.5 gini = 0.469 samples = 635 value = [238, 397]
X[30] <= 0.5 gini = 0.382 samples = 396 value = [102, 294]
gini = 0.499 samples = 23 value = [11, 12]
gini = 0.288 samples = 459 value = [80, 379]
gini = 0.231 samples = 255 value = [221, 34]
gini = 0.412 samples = 124 value = [88, 36]
gini = 0.3 samples = 440 value = [359, 81]
gini = 0.499 samples = 23 value = [11, 12]
gini = 0.391 samples = 15 value = [11, 4]
gini = 0.332 samples = 19 value = [4, 15]
X[19] <= 0.5 gini = 0.395 samples = 514 value = [375, 139]
X[0] <= 0.825 gini = 0.5 samples = 169 value = [84, 85]
gini = 0.039 samples = 149 value = [146, 3]
X[12] <= 0.5 gini = 0.311 samples = 1701 value = [1374, 327]
X[37] <= 0.5 gini = 0.463 samples = 3398 value = [2161, 1237]
gini = 0.425 samples = 137 value = [42, 95]
gini = 0.47 samples = 127 value = [79, 48]
X[15] <= 0.5 gini = 0.473 samples = 598 value = [230, 368]
gini = 0.408 samples = 7 value = [5, 2]
gini = 0.25 samples = 198 value = [29, 169]
X[15] <= 0.5 gini = 0.441 samples = 304 value = [100, 204]
gini = 0.265 samples = 229 value = [36, 193]
X[5] <= 0.5 gini = 0.46 samples = 581 value = [208, 373]
gini = 0.494 samples = 54 value = [30, 24]
X[41] <= 0.5 gini = 0.37 samples = 387 value = [95, 292]
gini = 0.346 samples = 9 value = [7, 2]
X[15] <= 0.5 gini = 0.378 samples = 491 value = [367, 124]
gini = 0.454 samples = 23 value = [8, 15]
X[17] <= 0.5 gini = 0.49 samples = 133 value = [76, 57]
X[0] <= 2.179 gini = 0.346 samples = 36 value = [8, 28]
X[17] <= 0.5 gini = 0.301 samples = 1634 value = [1333, 301]
gini = 0.475 samples = 67 value = [41, 26]
X[33] <= 0.5 gini = 0.45 samples = 2925 value = [1927, 998]
X[16] <= 0.5 gini = 0.5 samples = 473 value = [234, 239]
gini = 0.479 samples = 577 value = [229, 348]
gini = 0.091 samples = 21 value = [1, 20]
gini = 0.481 samples = 211 value = [85, 126]
gini = 0.271 samples = 93 value = [15, 78]
gini = 0.498 samples = 66 value = [35, 31]
gini = 0.446 samples = 515 value = [173, 342]
X[0] <= -0.444 gini = 0.382 samples = 383 value = [91, 292]
gini = 0.0 samples = 4 value = [4, 0]
X[37] <= 0.5 gini = 0.362 samples = 472 value = [360, 112]
gini = 0.465 samples = 19 value = [7, 12]
gini = 0.453 samples = 95 value = [62, 33]
gini = 0.465 samples = 38 value = [14, 24]
gini = 0.257 samples = 33 value = [5, 28]
gini = 0.0 samples = 3 value = [3, 0]
X[20] <= 0.5 gini = 0.295 samples = 1613 value = [1323, 290]
gini = 0.499 samples = 21 value = [10, 11]
X[10] <= 0.5 gini = 0.459 samples = 2689 value = [1728, 961]
gini = 0.264 samples = 236 value = [199, 37]
gini = 0.489 samples = 291 value = [124, 167]
X[0] <= 1.756 gini = 0.478 samples = 182 value = [110, 72]
gini = 0.481 samples = 100 value = [36, 64]
gini = 0.313 samples = 283 value = [55, 228]
gini = 0.326 samples = 371 value = [295, 76]
gini = 0.459 samples = 101 value = [65, 36]
X[44] <= 0.5 gini = 0.267 samples = 1231 value = [1036, 195]
X[4] <= 1.454 gini = 0.374 samples = 382 value = [287, 95]
X[30] <= 0.5 gini = 0.466 samples = 2567 value = [1617, 950]
gini = 0.164 samples = 122 value = [111, 11]
gini = 0.493 samples = 154 value = [86, 68]
gini = 0.245 samples = 28 value = [24, 4]
gini = 0.247 samples = 1116 value = [955, 161]
gini = 0.416 samples = 115 value = [81, 34]
gini = 0.337 samples = 312 value = [245, 67]
gini = 0.48 samples = 70 value = [42, 28]
X[4] <= -0.032 gini = 0.473 samples = 2429 value = [1499, 930]
gini = 0.248 samples = 138 value = [118, 20]
X[44] <= 0.5 gini = 0.458 samples = 1858 value = [1197, 661]
gini = 0.498 samples = 571 value = [302, 269]
X[0] <= 1.756 gini = 0.447 samples = 1696 value = [1123, 573]
gini = 0.496 samples = 162 value = [74, 88]
X[8] <= 0.5 gini = 0.458 samples = 1547 value = [999, 548]
gini = 0.279 samples = 149 value = [124, 25]
X[0] <= 0.656 gini = 0.464 samples = 1476 value = [935, 541]
gini = 0.178 samples = 71 value = [64, 7]
X[36] <= 0.5 gini = 0.432 samples = 806 value = [552, 254]
X[20] <= 0.5 gini = 0.49 samples = 670 value = [383, 287]
X[31] <= 0.5 gini = 0.422 samples = 764 value = [533, 231]
gini = 0.495 samples = 42 value = [19, 23]
X[11] <= 0.5 gini = 0.478 samples = 535 value = [324, 211]
gini = 0.492 samples = 135 value = [59, 76]
gini = 0.433 samples = 704 value = [481, 223]
gini = 0.231 samples = 60 value = [52, 8]
gini = 0.483 samples = 516 value = [306, 210]
gini = 0.1 samples = 19 value = [18, 1]

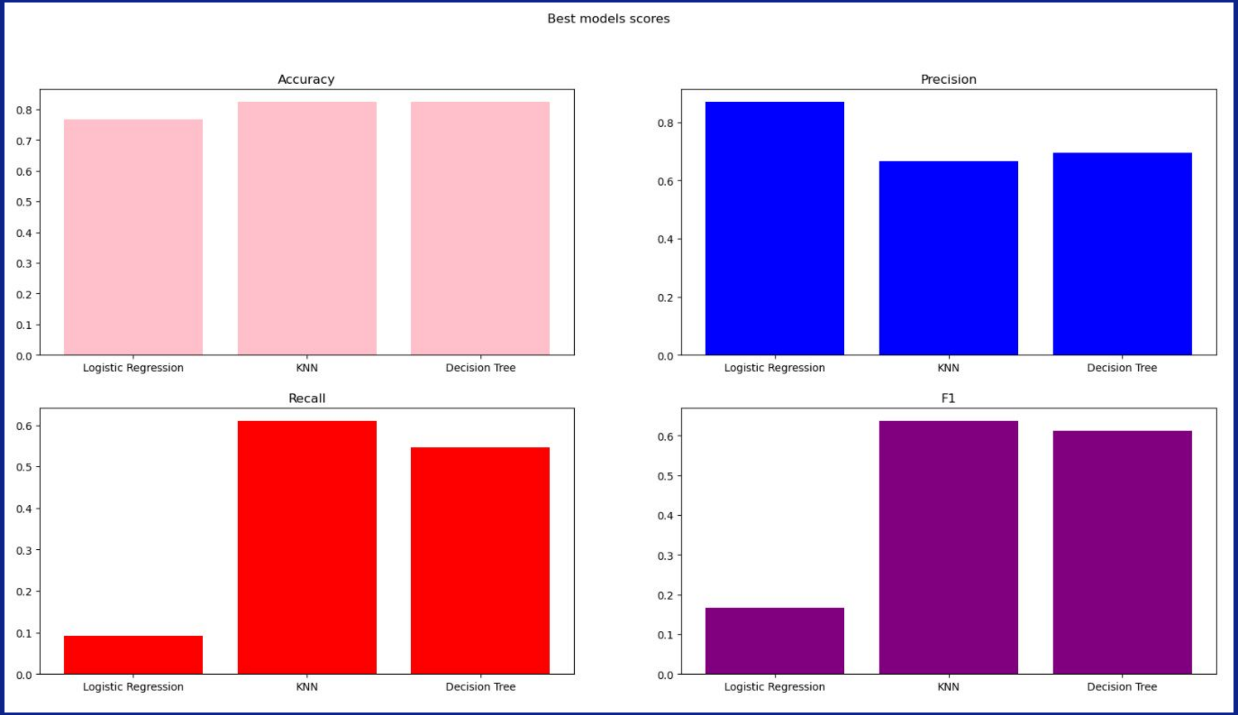
Curvas ROC


ROC Curve Logistic Regression


ROC Curve KNN


ROC Curve Decision Tree

Comparación de Resultados



Preguntas