▼ Variable "parroquia_urbana"

Cargamos las librerias necesarias para la elaboracion y desarrollo de la minería de datos

```
!pip install plotly
import numpy as np #Operaciones matemáticas rápidas sobre matrices
import pandas as pd #biblioteca de análisis y manipulación de datos para Python
import plotly.express as px
import matplotlib.pyplot as plt #Proporciona una forma de trazado similar a MATLAB. pyplot está diseñado principalmente para gráficos in
import seaborn as sns #permite generar fácilmente elegantes gráficos, proporciona una interfaz de alto nivel que es realmente sencilla d
import statsmodels.api as sm
# Preprocesado y modelado
from sklearn.model_selection import train_test_split
from \ sklearn.tree \ import \ Decision Tree Classifier
from sklearn.tree import plot_tree
from sklearn.tree import export graphviz
from sklearn.tree import export_text
from sklearn.model_selection import GridSearchCV
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import accuracy score
from sklearn.metrics import confusion_matrix
# Configuración warnings
import warnings
warnings.filterwarnings('once')
df= pd.read_csv('AT2021_NBD.csv')
```

Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages (5.13.1)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly)

	fecha	dia	hora	latitud	longitud	tipologia	direccion	zona	parroquia_urbana	cau
0	01-01	viernes	h11	-3.991993	-79.201155	estrellamiento	orillas del zamora y jose felix de valdivieso	urbana	el valle	conduci estadi embriaç
1	01-05	martes	h17	-4.020370	-79.217962	choque lateral perpendicular	benjamin carrion y gustavo serrano	urbana	punzara	conduci estadi embriaç
2	01-05	martes	h12	-3.987230	-79.202984	choque por alcance	nueva loja y guaranda	urbana	sucre	no mante la dista reglament
3	01-08	viernes	h10	-3.989410	-79.236506	atropello	angel felicisimo rojas	urbana	sucre	imprude del pea
4	01-09	sabado	h06	-3.979784	-79.218689	choque lateral angular	isidro ayora y habana	urbana	sucre	conduci estadı embriaç



Graficamos el mapa de calor de accidentabilidad dentro del cantón Loja

fig = px.density_mapbox(df,lat='latitud', lon='longitud',radius=3,center=dict(lat=-3.99313,lon=-79.20422),zoom=10.5,mapbox_style="open-s
fig.show()

nro_fallecidos dtype: int64

df.sample(10)

```
modelo Parroquia Urbana Weka python Calles accidentes 2021.ipynb - Colaboratory
     /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:
     `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1
                                                                                   0.8
                                                                                   0.6
                                                                                   0.4
                                                                                   0.2
#verifico datos nulos
df.isnull().sum()
     /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:
     `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer
     fecha
                               0
     dia
                               0
     hora
                               0
     latitud
                               1
     longitud
                               1
     tipologia
                               0
     direccion
                               0
     zona
                               0
     parroquia_urbana
                               0
     causas
                               0
     gravedad
                               0
     nro_heridos
                               0
     nro_fallecidos
                               0
     vehiculos retenidos
                               0
     senalizacion existente
                               0
     condicion_calzada
                               0
     condicion_atmosferica
                               0
     dtype: int64
     4
df=df.loc[:,df.columns!="fecha"]
df=df.loc[:,df.columns!="zona"]
df=df.loc[:,df.columns!="latitud"]
df=df.loc[:,df.columns!="longitud"]
df=df.loc[:,df.columns!="direccion"]
df=df.loc[:,df.columns!="gravedad"]
df=df.loc[:,df.columns!="vehiculos_retenidos"]
df=df.loc[:,df.columns!="senalizacion_existente"]
df=df.loc[:,df.columns!="condicion_calzada"]
df=df.loc[:,df.columns!="condicion_atmosferica"]
     /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:
     `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer
    4
df.isnull().sum()
     dia
                         0
     hora
                         0
     tipologia
                         0
     parroquia_urbana
                         0
                         0
     causas
     nro_heridos
                         0
```

https://colab.research.google.com/drive/1KdGC IK0x4HOD9J aMGScrndqfVRBIc-?authuser=1#scrollTo=Sl7sjJ1t8uod&printMode=true

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1

	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos
289	viernes	h10	choque lateral angular	sucre	imprudencia del conductor	0	0
29	jueves	h16	perdida de pista	sucre	fallas mecanicas no previsibles	2	0
225	sabado	h22	estrellamiento	sucre	conducir en estado de embriaguez	0	0
349	martes	h07	choque lateral perpendicular	el sagrario	imprudencia del conductor	2	0
269	domingo	h11	choque lateral perpendicular	san sebastian	no respetar las senales de transito	0	0
188	sabado	h00	estrellamiento	punzara	conducir en estado de embriaguez	0	0
207	viernes	h02	estrellamiento	carigan	conducir en exceso de velocidad	0	0
<pre>df.head()</pre>							

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should run async` will not call `transform cell` automatically in the future. Please pass the result 1

	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos
0	viernes	h11	estrellamiento	el valle	conducir en estado de embriaguez	1	0
1	martes	h17	choque lateral perpendicular	punzara	conducir en estado de embriaguez	1	0
2	martes	h12	choque por alcance	sucre	no mantener la distancia reglamentaria	0	0
3	viernes	h10	atropello	sucre	imprudencia del peaton	1	0

```
ds=pd.DataFrame(df)
#Presenta el numero de filas
print("El numero de filas(observaciones) es: ",ds.shape[0])
#Presenta el numero de columnas
print("El numero de columnas(variables) es: ",len(ds.columns))
     El numero de filas(observaciones) es: 370
     El numero de columnas(variables) es: 7
     /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:
     `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer
```

tipos de la variables ds.dtypes

> dia object hora object tipologia object parroquia_urbana object causas obiect nro_heridos int64 nro fallecidos int64 dtype: object

df.shape[0]

370

df['causas'].value_counts()

```
imprudencia del conductor
                                          161
conducir en estado de embriaguez
                                           88
conducir en exceso de velocidad
                                           75
imprudencia del peaton
                                           15
no respetar las senales de transito
                                           15
fallas mecanicas no previsibles
                                            6
no mantener la distancia reglamentaria
                                            2
```

```
condiciones climaticas desfavorables
     no ceder el derecho de via
                                                   2
     impericia del conductor
                                                   2
     cruce de animales en la via
     Name: causas, dtype: int64
print(df['causas'].unique())# datos en texto
     ['conducir en estado de embriaguez'
       'no mantener la distancia reglamentaria' 'imprudencia del peaton'
      'conducir en exceso de velocidad' 'no respetar las senales de transito'
      'condiciones climaticas desfavorables' 'no ceder el derecho de via'
      'impericia del conductor' 'imprudencia del conductor'
      'fallas mecanicas no previsibles' 'cruce de animales en la via']
     /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:
     `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer
    4
print(df['causas'].unique())# datos en texto
     ['conducir en estado de embriaguez'
       'no mantener la distancia reglamentaria' 'imprudencia del peaton'
      'conducir en exceso de velocidad' 'no respetar las senales de transito'
      'condiciones climaticas desfavorables' 'no ceder el derecho de via'
      'impericia del conductor' 'imprudencia del conductor'
      'fallas mecanicas no previsibles' 'cruce de animales en la via']
Transformamos los datos a numéricos
df['causas']= df['causas'].apply(lambda x:
                                      1 if x == 'conducir en estado de embriaguez' else
                                      2 if x == 'imprudencia del conductor' else
                                      3 if x == 'no ceder el derecho de via' else
                                      4 if x == 'conducir en exceso de velocidad' else
                                      5 if x == 'cambio brusco e indebido de carril' else
                                      6 if x == 'fallas mecanicas no previsibles' else
                                      7 if x == 'no respetar las senales de transito' else
                                      8 if x == 'invadir carril de circulacion' else
                                      9 if x == 'imprudencia del peaton' else
                                      10 if x == 'no mantener la distancia reglamentaria' else
                                      11 if x == 'razones desconocidas' else
                                     12 if x == 'condiciones climaticas desfavorables' else
                                      13 if x == 'negligencia del conductor' else
                                      14 if x == 'no respetar las ordenes del agente de transito' else
                                      15 if x == 'impericia del conductor' else
                                      16)
print(df['parroquia_urbana'].unique()) # datos convertidos a numeros enteros
     ['el valle' 'punzara' 'sucre' 'san sebastian' 'carigan' 'el sagrario']
df['parroquia_urbana']= df['parroquia_urbana'].apply(lambda x:
                                     1 if x == 'el sagrario' else
                                      2 if x == 'san sebastian' else
                                      3 if x == 'el valle' else
                                      4 if x == 'sucre' else
                                      5 if x == 'punzara' else
                                      6)
print(df['parroquia_urbana'].unique()) # datos convertidos a numeros enteros
     [3 5 4 2 6 1]
print(df['tipologia'].unique()) # datos convertidos a numeros enteros
     ['estrellamiento' 'choque lateral perpendicular' 'choque por alcance' 'atropello' 'choque lateral angular' 'roce negativo' 'perdida de carril'
      'perdida de pista' 'choque frontal excentrico' 'colision' 'atipico' 'arrollamiento' 'volcamiento' 'roce positivo' 'rozamiento'
      'caida de pasajero']
df['tipologia']= df['tipologia'].apply(lambda x:
                                     1 if x == 'arrollamiento' else
                                      2 if x == 'atipico' else
                                      3 if x == 'atropello' else
```

```
4 if x == 'caida de pasajero' else
                                    5 if x == 'choque frontal' else
                                    6 if x == 'choque frontal excentrico' else
                                    7 if x == 'choque frontal longitudinal' else
                                    8 if x == 'choque lateral angular' else
                                    9 if x == 'choque lateral perpendicular' else
                                    10 if x == 'choque por alcance' else
                                    11 if x == 'colision' else
                                   12 if x == 'encunetamiento' else
                                   13 if x == 'estrellamiento' else
                                   14 if x == 'perdida de carril' else
                                   15 if x == 'perdida de pista' else
                                   16 if x == 'roce negativo' else
                                    17 if x == 'roce positivo' else
                                   18 if x == 'rozamiento' else
                                    19 if x == 'volcamiento' else
                                    20 if x == 'volcamiento lateral' else
                                    21)
print(df['tipologia'].unique()) # datos convertidos a numeros enteros
     [13 9 10 3 8 16 14 15 6 11 2 1 19 17 18 4]
df['dia']= df['dia'].apply(lambda x:
                                   1 if x == 'lunes' else
                                    2 if x == 'martes' else
                                    3 if x == 'miercoles' else
                                    4 if x == 'jueves' else
                                    5 if x == 'viernes' else
                                    6 if x == 'sabado' else
                                    7)
print(df['dia'].unique()) # datos convertidos a numeros enteros
     [5 2 6 7 1 4 3]
df= df[df['parroquia_urbana'] <= 6]</pre>
df['hora']= df['hora'].apply(lambda x:
                                   0 if x == 'h00' else
                                    1 if x == 'h01' else
                                   2 if x == 'h02' else
                                   3 if x == 'h03' else
                                   4 if x == 'h04' else
                                    5 if x == 'h05' else
                                   6 if x == 'h06' else
                                    7 if x == 'h07' else
                                   8 if x == 'h08' else
                                    9 if x == 'h09' else
                                   10 if x == 'h10' else
                                    11 if x == 'h11' else
                                   12 if x == 'h12' else
                                   13 if x == 'h13' else
                                   14 if x == 'h14' else
                                   15 if x == 'h15' else
                                   16 if x == 'h16' else
                                   17 if x == 'h17' else
                                   18 if x == 'h18' else
                                   19 if x == 'h19' else
                                    20 if x == 'h20' else
                                    21 if x == 'h21' else
                                    22 if x == 'h22' else
                                    23)
df.head()
```

	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos	1	th
0	5	11	13	3	1	1	0		
1	2	17	9	5	1	1	0		
2	2	12	10	4	10	0	0		
3	5	10	3	4	9	1	0		
4	6	6	8	4	1	0	0		

datos aleatorios (muestra de 10 elementos)

df.sample(10)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1

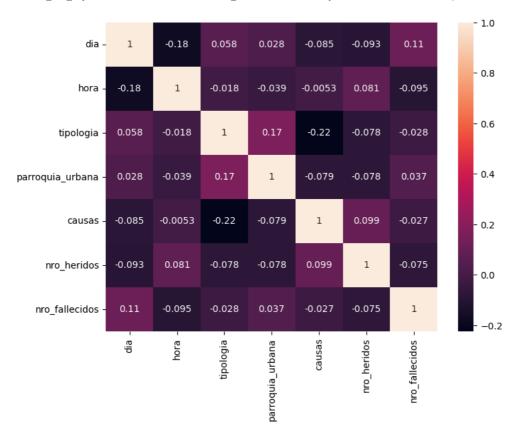
	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos	/
270	4	15	8	1	2	1	0	
73	1	21	13	6	2	1	0	
255	7	3	14	5	1	0	0	
330	6	20	13	4	2	0	1	
28	6	4	9	1	1	0	0	
205	5	13	9	5	2	1	0	
21	7	22	10	5	15	1	0	
296	3	18	9	5	2	1	0	
33	2	19	8	4	2	0	0	
361	6	21	10	1	1	0	0	

Verificamos la correlaxion de los datos

```
#corelacion
corr_df = df.corr(method='pearson')
plt.figure(figsize=(8, 6))
sns.heatmap(corr_df, annot=True)
plt.show()
```

 $/usr/local/lib/python 3.10/dist-packages/ipykernel/ipkernel.py: 283: \ Deprecation Warning: 1.00/dist-packages/ipykernel.py: 283: \ Deprecation Warning: 283: \ Deprecat$

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1



X = df.iloc[:, [0,1,2,4,5,6]] # atributos de entrada seran las primeras columnas <math>Y = df.iloc[:,[3]] # atributos de destino

#presentacion de los atributos de entrada
X.head()

	dia	hora	tipologia	causas	nro_heridos	nro_fallecidos	1	ıl.
0	5	11	13	1	1	0		
1	2	17	9	1	1	0		
2	2	12	10	10	0	0		
3	5	10	3	9	1	0		
4	6	6	8	1	0	0		

feature names = X.columns.tolist()

Imprimir los nombres de las características
print(feature_names)

```
['dia', 'hora', 'tipologia', 'causas', 'nro_heridos', 'nro_fallecidos'] /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:
```

→

#presentacion de los atributos de destino
Y.head()

	parroquia_urbana	1	ıl.
0	3		
1	5		
2	4		
3	4		
4	4		

Cargamos el modelo

```
from joblib import load
arbol_modelo = load('modelo_parroquia_urbana_python.joblib')
y_pred = arbol_modelo.predict(X)
```

 $/usr/local/lib/python 3.10/dist-packages/ipykernel/ipkernel.py: 283: \ Deprecation Warning: 283. \ Deprecation W$

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:318: UserWarning:

Trying to unpickle estimator DecisionTreeClassifier from version 1.0.2 when using version 1.2.2. This might lead to breaking code c https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations

```
←
```

print(y pred)

 $/usr/local/lib/python 3.10/dist-packages/ipykernel/ipkernel.py: 283: \ Deprecation Warning: 1.00/dist-packages/ipykernel.py: 283: \ Deprecation Warning: 283: \ Deprecat$

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` arguments

```
# Resumen de las predicciones hechas por el clasificador
from sklearn import metrics
reporte = metrics.classification_report(Y, y_pred,output_dict=True)
pre = pd.DataFrame(reporte).transpose()
```

[`]should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer

```
print(pre)
pre.to_excel("resumen_precision_parroquia_urbana_2021.xlsx")
          /usr/local/lib/python 3.10/dist-packages/sklearn/metrics/\_classification.py: 1344: \ Undefined Metric Warning: like the property of the prop
          Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cc
          /usr/local/lib/python3.10/dist-packages/sklearn/metrics/ classification.py:1344: UndefinedMetricWarning:
          Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cc
          /usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning:
          Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cc
          <frozen importlib._bootstrap>:914: ImportWarning:
          APICoreClientInfoImportHook.find spec() not found; falling back to find module()
          <frozen importlib. bootstrap>:914: ImportWarning:
          _PyDriveImportHook.find_spec() not found; falling back to find_module()
          <frozen importlib._bootstrap>:914: ImportWarning:
          _OpenCVImportHook.find_spec() not found; falling back to find_module()
          <frozen importlib. bootstrap>:914: ImportWarning:
          _BokehImportHook.find_spec() not found; falling back to find_module()
          <frozen importlib. bootstrap>:914: ImportWarning:
          _AltairImportHook.find_spec() not found; falling back to find_module()
                                                                                                              support
                                       precision
                                                                  recall f1-score
                                          0.333333
                                                            0.444444
                                                                                  0.380952
                                                                                                          63.000000
          2
                                         0.000000
                                                            0.000000 0.000000
                                                                                                         55.000000
          3
                                         0.000000
                                                             0.000000
                                                                                  0.000000
                                                                                                         33.000000
          4
                                                             0.860870
                                                                                  0.500000 115.000000
                                         0.352313
                                                                                  0.029851
                                                                                                         62,000000
          5
                                         0.200000
                                                             0.016129
          6
                                         0.000000
                                                             0.000000
                                                                                  0.000000
                                                                                                         42,000000
          accuracy
                                         0.345946
                                                             0.345946
                                                                                  0.345946
                                                                                                           0.345946
          macro avg
                                         0.147608 0.220241 0.151801
                                                                                                       370.000000
          weighted avg
                                         0.199773 0.345946 0.225272 370.000000
```

Pedicciones

```
#Predicciones
pred = pd.DataFrame(y_pred)
pred.head()
```

 $/usr/local/lib/python 3.10/dist-packages/ipykernel/ipkernel.py: 283: \ Deprecation Warning: 200. \ Deprecation W$

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1



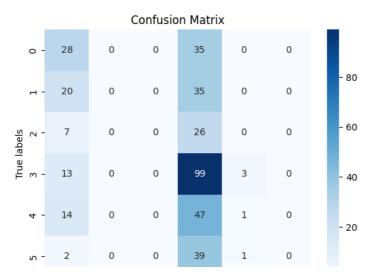
- 0 4
- **1** 1
- 2 4
- 3 4
- 4 4

Generamos la matriz de confusión

```
#Matriz de confusion
matriz=confusion_matrix(Y, y_pred)
ax= plt.subplot()
sns.heatmap(matriz, annot=True, cmap="Blues",fmt='g');
ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
```

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1



#Para concatenar variables
import pandas as pd

```
prediccion = pd.DataFrame(y_pred,columns=['parroquia_urbana_prediccion'])
original = Y
original.reset_index(drop=True, inplace=True)
df_combined = pd.concat([prediccion,original], axis=1)
df combined.head(10)
```

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1

	parroquia_urbana_prediccion	parroquia_urbana	1	ıl.
0	4	3		
1	1	5		
2	4	4		
3	4	4		
4	4	4		
5	4	4		
6	4	2		
7	4	4		
8	4	2		
9	1	2		

Visualizamos la exactitud del modelo

```
# Precisión

from sklearn.metrics import accuracy_score, recall_score, precision_score

print('Exactitud árboles de decisión: ',accuracy_score(pred,Y)*100)

print('Exhaustividad árboles de decisión: ', recall_score(pred,Y,average='micro')*100)

print('Precision árboles de decisión: ',accuracy_score(pred,Y)*100)

Exactitud árboles de decisión: 34.5945945946

Exhaustividad árboles de decisión: 34.5945945945946

Precision árboles de decisión: 34.5945945946

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer
```

Transformamos las variables

```
#transformar datos para visualización de variables predictoras y originales df_combined['parroquia_urbana_prediccion']= df_combined['parroquia_urbana_prediccion'].apply(lambda x:
```

```
'el sagrario' if x == 1 else
'san sebastian' if x == 2 else
'el valle' if x == 3 else
'sucre' if x == 4 else
'punzara' if x == 5 else
'carigan')

df_combined['parroquia_urbana'] = df_combined['parroquia_urbana'].apply(lambda x:
'el sagrario' if x == 1 else
'san sebastian' if x == 2 else
'el valle' if x == 3 else
'sucre' if x == 4 else
```

 $/usr/local/lib/python 3.10/dist-packages/ipykernel/ipkernel.py: 283: \ Deprecation Warning: \\$

'punzara' if x == 5 else

'carigan')

ha='center', va='bottom') # set the alignment of the text

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer

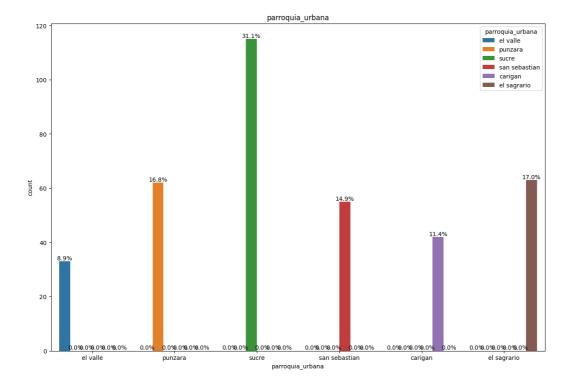
←

Graficamos las estadisticas originales de los datos

```
import seaborn as sns #permite generar fácilmente elegantes gráficos, proporciona una interfaz de alto nivel que es realmente sencilla d
ax = plt.subplots(figsize = (15,10))
ncount=len(df_combined)
sns.countplot(x='parroquia_urbana',hue = 'parroquia_urbana', data = df_combined, ax = ax[1]) #Muestre el conteo de observaciones en cada
ax[1].set_title('parroquia_urbana')

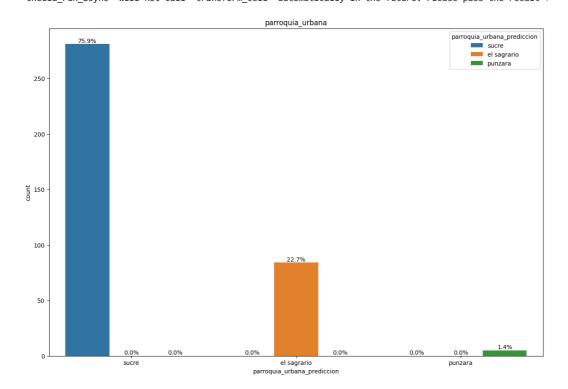
for p in ax[1].patches:
    x=p.get_bbox().get_points()[:,0]
    y=p.get_bbox().get_points()[1,1]
    ax[1].annotate('{:.1f}%'.format(100.*y/ncount), (x.mean(), y),
```

#Guardamos la figura de barras original ax[0].savefig("barras_original_parroquia_urbana_2021.png")



Graficamos las estadisticas predecidas de los datos

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result t



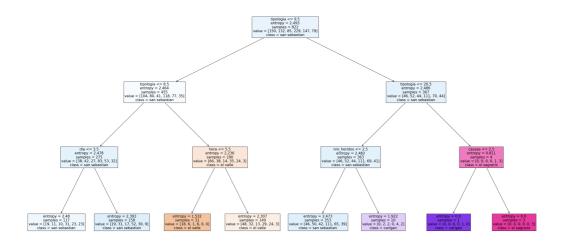
Crear el arbol de desición actual con el modelo con datos predichos del 2021

 $/usr/local/lib/python 3.10/dist-packages/ipykernel/ipkernel.py: 283: \ Deprecation Warning: \\$

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer

Creamos la figura del arbol

figura_arbol = plt.figure(figsize=(40,20)) # Le indicamos las dimensiones que queremos que tenga
plot_tree(arbol_modelo,feature_names=feature_names,filled=True, class_names=target_names_str, fontsize=15)
plt.show()



figura_arbol.savefig("arbol_colab_parroquia_urbana_2021.png")

X.head()

	dia	hora	tipologia	causas	nro_heridos	nro_fallecidos	1	ıl.
C	5	11	13	1	1	0		
1	2	17	9	1	1	0		
2	2	12	10	10	0	0		
3	5	10	3	9	1	0		
4	6	6	8	1	0	0		

Verificamos la probabilidad de los accidentes de tránsito a través de la predicción del modelo

18.41 14.16 14.16 14.16 18.41 18.41 18.41 14.16 18.41 18.41 18.41 18.41

```
#Verificamos la probabilidad de todas las variables
y proba = arbol modelo.predict proba(X)
probabilidad_acierto = np.round(y_proba[0][y_pred] * 100, 2)
print("Probabilidad de Acierto: " + str(probabilidad_acierto) + "%")
     Probabilidad de Acierto: [18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41 18.41 14.16 18.41 18.41
      18.41 18.41 11.05 18.41 18.41 14.16 18.41 14.16 14.16 18.41 11.05 18.41
      18.41 14.16 14.16 14.16 14.16 18.41 14.16 14.16 14.16 18.41 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 14.16
      18.41 \ 18.41 \ 18.41 \ 14.16 \ 18.41 \ 18.41 \ 11.05 \ 18.41 \ 18.41 \ 14.16 \ 14.16 \ 14.16
      14.16 18.41 14.16 14.16 18.41 18.41 18.41 14.16 18.41 18.41 18.41 14.16
      18.41 18.41 18.41 18.41 18.41 14.16 18.41 18.41 14.16 18.41 18.41 18.41
      14.16 18.41 18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41 18.41 14.16
      18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 14.16
      14.16 18.41 18.41 14.16 18.41 18.41 18.41 18.41 18.41 14.16 14.16 18.41
      18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 14.16 18.41 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 14.16 14.16 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 14.16 18.41 18.41 18.41 14.16 14.16 18.41
      18.41 18.41 18.41 14.16 18.41 18.41 18.41 18.41 14.16 18.41 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41
```

```
18.41\ 14.16\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 14.16
      14.16 18.41 18.41 14.16 14.16 18.41 18.41 14.16 14.16 18.41 14.16 18.41
      18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41
      14.16 14.16 18.41 18.41 14.16 18.41 18.41 18.41 18.41 14.16 18.41 18.41
      18.41 18.41 18.41 18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41
      18.41 18.41 14.16 18.41 18.41 14.16 14.16 18.41 14.16 18.41 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 18.41 14.16 18.41 14.16 18.41 11.05 18.41
      18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 18.41\ 14.16\ 18.41\ 14.16
      18.41 18.41 18.41 18.41 14.16 14.16 18.41 14.16 11.05 14.16 18.41 18.41
      18.41 18.41 18.41 18.41 18.41 18.41 18.41 18.41 14.16 18.41 14.16 18.41
      18.41 18.41 14.16 18.41 18.41 18.41 18.41 18.41 18.41 14.16 18.41 18.41
      18.41 14.16 18.41 18.41 18.41 14.16 18.41 18.41 14.16 18.41 18.41 18.41
      18.41 18.41 18.41 14.16 14.16 18.41 18.41 18.41 14.16 18.41]%
     /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:
     `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer
print(type(probabilidad_acierto))
columna_probabilidades = pd.DataFrame()
columna_probabilidades['probabilidad'] = pd.DataFrame(probabilidad_acierto)
print(columna_probabilidades)
     <class 'numpy.ndarray'>
          probabilidad
     0
                 18.41
     1
                 14.16
     2
                 18.41
     3
                 18.41
     4
                 18.41
```

[370 rows x 1 columns]

18.41

18.41

18.41

14.16 18.41

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer

365 366

367

368

Presentamos la probabilidad de los datos

```
y_prediccion = pd.DataFrame()
y_prediccion['predicciones'] = pd.DataFrame(y_pred)
probabilidades_2021_df = pd.concat([df,columna_probabilidades, y_prediccion], axis=1)
probabilidades_2021_df.sample(10)
```

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1

	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos	probabilidad	predicc
249	3	22	9	5	7	1	0	14.16	
133	6	19	10	5	1	0	0	18.41	
101	2	21	8	1	4	1	0	18.41	
135	6	15	13	5	4	1	0	18.41	
178	5	20	13	5	2	1	0	18.41	
238	1	15	3	4	4	1	0	18.41	
295	4	13	4	2	2	1	0	18.41	
200	1	19	13	5	1	0	0	18.41	
207	5	2	13	6	4	0	0	18.41	
8	1	2	13	2	4	1	0	18.41	
4.4	_								

```
probabilidades_2021_df['predicciones']= probabilidades_2021_df['predicciones'].apply(lambda x:
                                    'h00-h05' if x == 0 else
                                    'h06-h011' if x == 1 else
```

^{&#}x27;h012-h17' if x == 2 else

```
probabilidades_2021_df['dia']= probabilidades_2021_df['dia'].apply(lambda x:
                                                                'lunes' if x == 1 else
                                                                'martes' if x == 2 else
                                                                'miercoles' if x == 3 else
                                                                'jueves' if x == 4 else
                                                                 'viernes' if x == 5 else
                                                                'sabado' if x == 6 else
                                                                'domingo' )
probabilidades_2021_df['hora'] = probabilidades_2021_df['hora'].apply(lambda x:
                                                                'h00' if x == 0 else
                                                                'h01' if x == 1 else
                                                                'h02' if x == 2 else
                                                                'h03' if x == 3 else
                                                                'h04' if x == 4 else
                                                                'h05' if x == 5 else
                                                                h06' if x == 6 else
                                                                'h07' if x == 7 else
                                                                'h08' if x == 8 else
                                                                'h09' if x == 9 else
                                                                'h10' if x == 10 else
                                                                'h11' if x == 11 else
                                                                'h12' if x == 12 else
                                                                'h13' if x == 13 else
                                                                'h14' if x == 14 else
                                                                'h15' if x == 15 else
                                                                'h16' if x == 16 else
                                                               'h17' if x == 17 else
                                                                'h18' if x == 18 else
                                                                'h19' if x == 19 else
                                                                'h20' if x == 20 else
                                                                'h21' if x == 21 else
                                                                'h22' if x == 22 else
                                                                'h23')
probabilidades\_2021\_df['parroquia\_urbana'] = probabilidades\_2021\_df['parroquia\_urbana']. apply(lambda \ x: lambda \ x: lambd
                                                                'el sagrario' if x == 1 else
                                                                'san sebastian' if x == 2 else
                                                                'el valle' if x == 3 else
                                                                 'sucre' if x == 4 else
                                                                'punzara' if x == 5 else
                                                                'carigan')
probabilidades_2021_df['tipologia'] = probabilidades_2021_df['tipologia'].apply(lambda x:
                                                                 'arrollamiento' if x == 1 else
                                                                'atipico' if x == 2 else
                                                                'atropello' if x == 3 else
                                                                'caida pasajero' if x == 4 else
                                                                'choque frontal' if x == 5 else
                                                                'choque frontal excentrico' if x == 6 else
                                                                'choque frontal longitudinal' if x == 7 else
                                                                'choque lateral angular' if x == 8 else
                                                                'choque lateral perpendicular' if x == 9 else
                                                                'choque alcance' if x == 10 else
                                                                'colision' if x == 11 else
                                                                'encunetamiento' if x == 12 else
                                                                'estrellamiento' if x == 13 else
                                                                'perdida carril' if x == 14 else
                                                                'perdida pista' if x == 15 else
                                                                'roce negativo' if x == 16 else
                                                                'roce posositivo' if x == 17 else
                                                                'rozamiento' if x == 18 else
                                                                'volcamiento' if x == 19 else
                                                                'volcamiento lateral' if x == 20 else
                                                                'volcamiento longitudinal')
probabilidades_2021_df['causas'] = probabilidades_2021_df['causas'].apply(lambda x:
                                                                'conducir en estado de embriaguez' if x == 1 else
                                                                'imprudencia del conductor' if x == 2 else
                                                                'no ceder el derecho de via' if x == 3 else
                                                                'conducir en exceso de velocidad' if x == 4 else
                                                                'cambio brusco e indebido de carril' if x == 5 else
                                                                'fallas mecanicas no previsibles' if x == 6 else
                                                                'no respetar las senales de transito' if x == 7 else
                                                                'invadir carril de circulacion' if x == 8 else
                                                                'imprudencia del peaton' if x == 9 else
                                                                'no mantener la distancia reglamentaria' if x == 10 else
                                                                'razones desconocidas' if x == 11 else
                                                                'condiciones climaticas desfavorables' if x == 12 else
                                                                'negligencia del conductor' if x == 13 else
                                                                'no respetar las ordenes del agente de transito' if x == 14 else
```

```
'impericia del conductor' if x == 15 else
```

'cruce de animales en la via')

probabilidades 2021 df.sample(10)

import pandas as pd

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning:

`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1

	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos	probabilida
6	domingo	h15	estrellamiento	san sebastian	conducir en exceso de velocidad	1	0	18.4
111	domingo	h08	choque lateral perpendicular	punzara	imprudencia del conductor	1	0	14.1
79	lunes	h06	choque lateral angular	el valle	conducir en exceso de velocidad	3	0	18.4
23	sabado	h21	estrellamiento	el valle	conducir en exceso de velocidad	0	0	18.4
261	viernes	h22	perdida carril	punzara	conducir en exceso de velocidad	0	0	18.4
310	jueves	h05	estrellamiento	sucre	imprudencia del conductor	0	0	18.4
59	sabado	h14	choque lateral perpendicular	sucre	conducir en exceso de velocidad	1	0	14.1
254	sabado	h13	estrellamiento	el sagrario	imprudencia del	1	0	18.4

```
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages

# Crear una figura y un eje
fig, ax = plt.subplots(figsize=(8, 6))

# Eliminar marcas del eje
ax.axis('off')

# Crear la tabla a partir del DataFrame
tabla = pd.plotting.table(ax, probabilidades_2021_df, loc='center', cellLoc='center', fontsize=14)

# Guardar la tabla en un archivo PDF
with PdfPages('tabla_probabilidades_parroquia_urbana_2021.pdf') as pdf:
    pdf.savefig(fig, bbox_inches='tight')
```

	-				,			,	
	min	61	strope birms proproducals	rhapon	na compositor de cuerculos de insoculies	L.		111	BEAGET
			_						
	derings	61	drope Mendargler	seretetin	improduce in delicandaries	2	'	8.0	ence
		61	atolianists		conductors retails de mobilepar			84	Blatt
	inn	61	drope Mendamentining		Ingradiente del conductor	,	,	12	MARCI
March			_		_				
	rendo.	61	árapa cilene	risquis	Improducia del andador			84	CLOS
	min	61	Name in	chapses	Improduce de del produce	t.		14	Blick
	nimin.	e	stronger Mercel arregular		Improduces delicandaries	1		86	61401
Part	riendo.		nor sende		Ingradiente del conductor		,	16	8161
	nierales.		drope birnel proprodicales	poss	ra regerior la camales de inocale	1	'	131	86.6001
March	jenes		atolianists		contains remarks wholed			84	Blick
March	-		190-	erige:	Improduce delicandados			86	ence
March									
							·		6161
	-		shope bireal angular		Improduces delicandador			8.0	61101
	nieto	61	ninderinis	chapuis	Improduce in delicendaries	t.	1	84	ence
March	derings		pethons	pone	conducts are ratically the sembologous			84	gases
March Marc									
March	ins.		peddored	elsale	renderir se recessive orbeidel	1	,	16	6160
March	min.	61	drops slaves	pone	Improduce in this and other			86	BIAN
March	derings		draga Mendamin	chapsis	renderir se ratula de profesiones	2		8.0	Blick
March	ninto								
	-		politicant	pone	contact or recovering of critical			8.0	Blick
March	jeres	61	mpris		Improduces delicated and	2	,	86	esci
March									
March		_	_						
March Marc	min		drope clama	chagosis	improduceia del candante	1	'	8.0	Blick
March Marc	inn.		no sepilo		contact or recovering administr	1		86	gaset.
March Marc	ins.	61	pethons		contact or recovering admitted	,		84	BIACE
March			_			_			
March			_						
Part	min.	61	atolientos	elude	Improduce in delicandaries			8.6	Blick
March Marc	derings	61	drope bireal poperalisator	serviteian	sa cosperine las careades de insendie.		-	12	86.6001
March Marc	ie	-	_	styrenis	Ingrahasi Alicenteia		,	10	giva.
March Marc			_						
March Marc		61	400	rhapois	randoù ar rainh de ambrigan		,	86	ence
March Marc		61	alreague frantial nacestrica		Improduce delicandaries			84	61101
March		61	190-	etge.	Improduce delicated and a			86	ence
March Marc			_						
March 1									
March Marc	-	61	political		contains consult about	2	,	1.0	ence
March Marc	nieroles.		shope blend angle	seretetie	Improduced delicandaries	2		84	Blick
March Marc	state		striferioris	elude	conductors recovered automidal			84	gaset .
March March March Strategy March									
Part			_						
March 1	nimin.	e	shope blood angles	chapute	Improduces delicandador	1		8.0	6160
March Marc	jenn	**	atriferation		conductors retails de medicingues			14	6160
March Marc	min.	61	drope bireal propriate de-		Ingradienta di Cambalan	,		12	86.6001
Marie									
March			_						
March Marc	nimala.	**	190-	poss	Maranton reposition	'	'	8.0	ence
March Marc	design		drope birnel propriativalse	shapois	Improduction delicated action	L		18	BLASS
									Blick
March Marc									
Column	derings			pose		t.			
March Marc	-		drope Mendangdor		conductive ratios de mobiliques	L	1	14	BLUE
March Marc	nini-		ato Garciania	chapute	contains recovering administra			8.0	ence
March Marc									
March Marc									
			_						
March Marc		ec ec	winderion		conductron ratioals de-medicinguas		1	84	BLICE
March Marc	jeres		atolianists	etge.	rando ir se ratula de restriaguas			84	Blick
March Marc			ato Gardeni	1000				84	Blick
March Marc									
Part	jess	61	min projem	serviction	Improduce in this contactor	1		8.6	Blick
Part	similar.		shope birmi poperalisalse	pone	Improduction delicated action	1		121	86.653
March Col. Section March Mar									give
March Marc									
	min.	61	at decision		rendert er rennerte oderskel	7	1	1.0	BLO
	election.	**	at electron	poss	contact or recovering advantad			86	Blick
				g/m					
March Marc			_						
			atriferints						
Part 1				ariga.	Improduces dell'assolution	2		8.6	Blick
Part 1			alian .		conductors returb de medicinguas		,	84	Blick
March Marc	min.			chapute					
	min.	*	sintenen						
Marie Mari	mins election provi	**	sinfaninis práktori	poss	sanda ir na salada da mederiaguas				
10	mins election provi	**	sinfaninis práktori	poss	sanda ir na salada da mederiaguas				
March Marc	miles Missiles jeurs miles	97 96 97	atrollerates profilerated atrollerates	pons	sendo ir ne estado de medidiguas Improduccia dell'osmlatine	,		3.0	Blich
Marco Marc	mates mondes. prom mates specifies	97 96 97 91	striferints problem ord striferints striferints	poss poss	conducto no estudo do condiviagua e improduce in della conductor conducto no estudo da condiviagua e	1	1	3.0	ELCE
pers El deficients um Special and Silvadories 1 1 E.E. DEAD pers El description deligned deliver pers 3 1 2 1 1 2 1 1	meter mendes jeurs meter mendes mendes	90 90 90 90 90 90 90 90 90 90 90 90 90 9	stedientele problemed stedientele stedientele stedientele	power power	sendo ir ne retain de medicingues improdució del conductor sendo ir ne retain de medicingues conductor ne retain de medicingues			3.6	ERACE ERACE
per 81 disprénégopolisé para legislate I I 1 23 8501 em 81 dispréné unité de dispréné de de dispréné de dispréné de dispréné de dispréné de dispréné de dispréné de de dispréné de de dispréné de de dispréné de de de dispréné de de de dispréné de de de dispréné de de de des de de des de de de des de de de des de	stender. jeurs stender. stender. stender. stender. stender.	90 90 90 90 90 90 90 90 90 90 90 90 90 9	skolanski prilikorel skolanski skolanski skolanski skolanski	poses poses poses poses	conductors contain de embriques improdució del conductor conductor en estado de embriques conductors en estado de embrigues conductors en estado de embrigues conductors en estado de embrigues			3.6 3.6 3.6	SEACE SEACE SEACE SEACE
per 81 disprénégopolisé para legislate I I 1 23 8501 em 81 dispréné unité de dispréné de de dispréné de dispréné de dispréné de dispréné de dispréné de dispréné de de dispréné de de dispréné de de dispréné de de de dispréné de de de dispréné de de de dispréné de de de des de de des de de de des de de de des de	stender. jeurs stender. stender. stender. stender. stender.	90 90 90 90 90 90 90 90 90 90 90 90 90 9	skolanski prilikorel skolanski skolanski skolanski skolanski	poses poses poses poses	conductors contain de embriques improdució del conductor conductor en estado de embriques conductors en estado de embrigues conductors en estado de embrigues conductors en estado de embrigues			3.6 3.6 3.6	SEACE SEACE SEACE SEACE
area III desperient westerien reduct metals benings I I I III	material control of the control of t	60 60 60 60 60 60 60 60 60 60 60 60 60 6	aterial personal of the second	power power power power som whether shade	esodo ir estado de ambiegas reproducis del mediate esodo ir estado de ambiegas conducir es estado de ambiegas conducir es estado de ambiegas esodo ir estado de ambiegas repulsar as del ambiegas	1		3.6 3.6 3.6 3.6	ERACE ERACE ERACE ERACE ERACE
	main records pers ander conden ander	10 10 10 10 10 10 10 10 10 10 10 10 10 1	aterial control of the control of th	power po	sendor in estado de celebraçãos improdes in del centrálismo central del centrálismo central de centrálismo centr	1	1	3.6 3.6 3.6 3.6 3.6	ELATE ELATE ELATE ELATE ELATE ELATE ELATE ELATE
design 85 despublished seeded a seed approbate distribution 1 1 3.65 SSSS	min models per service de la constante de la c	42 44 44 44 44 44 44 44 44 44 44 44 44 4	standarde per de la constitución	power power power power stude power power	conducto estado de confengaca ingrederio del confengaca conducto estado de confengaca conducto estado de confengaca conducto estado de confengaca conducto estado de confengaca ingrederio del confendações ingrederio del confendações ingrederio del confenda ingrederio del confenda ingred	1	1	3.6 3.6 3.6 3.8 3.8	ELANT
	min	12 12 12 12 12 12 12 12 12 12 12 12 12 1	pridenti pridenti pridenti didenti	possa possa possa un edetian elado un edetian un edetian un un edetian un un edetian un edetian	conductor solutido confragosos improductos del conductor conductor conductor conductor conductor conductor conductor conductor consumed as electrical conductor conductor conductor improductos del conductor improductos del conductor improductos del conductor improductos del conductor improductor del conductor improductor del conductor improductor del conductor entended or colonial do colonial entended or colonial entended or colonial entended entended entend	1	1	3.6 3.6 3.6 3.6 3.7 3.6	ELAST ELAST ELAST ELAST ELAST ELAST ELAST ELAST ELAST ELAST