## ▼ Variable "hora"

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 DecisionTreeClassifier
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')
df.head()
     /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 t
     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
                                                                   orillas del
                                                                   zamora y
                                                                                                         conduci
                         h11 -3.991993 -79.201155 estrellamiento
        01-01 viernes
                                                                   jose felix
                                                                            urbana
                                                                                               el valle
                                                                                                          estado
                                                                                                         embriac
                                                                        de
                                                                   valdivieso
                                                                   benjamin
                                                          choque
                                                                                                         conduci
                                                                   carrion y
        01-05
               martes
                         h17 -4.020370 -79.217962
                                                          lateral
                                                                             urbana
                                                                                              punzara
                                                                                                          estado
                                                                    gustavo
                                                    perpendicular
                                                                                                         embriaç
                                                                    serrano
                                                                  nueva loja
                                                                                                        no mante
                                                      choque por
                         h12 -3.987230 -79.202984
      2 01-05
                martes
                                                                             urbana
                                                                                                         la dista
                                                                                                sucre
                                                                          ٧
                                                         alcance
                                                                   guaranda
                                                                                                       reglament
                                                                      angel
                                                                                                        imprude
```



3 01-08

viernes

01-09 sabado

Graficamos el mapa de calor de accidentabilidad dentro del cantón Loja

h10 -3 989410 -79 236506

h06 -3 979784 -79 218689

```
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()
```

felicisimo

rojas

isidro

ayora y

habana

urbana

urbana

sucre

sucre

del pea

conduci

embriaç

estado

atropello

choque

angular

lateral

```
`should_run_async` will not call `transform_cell` automatically in the future. Please pass the result 1
#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
     tipologia
                               0
     direccion
     zona
                               0
     parroquia_urbana
                               0
                               0
     causas
     gravedad
                               a
     nro_heridos
                               0
     nro_fallecidos
                               0
     vehiculos_retenidos
                               0
     senalizacion_existente
                               0
     condicion_calzada
     condicion_atmosferica
                               0
     dtype: int64
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
                         0
     hora
     tipologia
                         0
     parroquia_urbana
                         0
     causas
                         0
     nro_heridos
                         0
     nro_fallecidos
                         0
```

dtype: int64

df.sample(10)

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

`should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result 1

	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos
219	lunes	h21	choque lateral perpendicular	el sagrario	imprudencia del conductor	2	0
137	lunes	h11	choque lateral perpendicular	san sebastian	no respetar las senales de transito	0	0
83	viernes	h23	perdida de carril	sucre	conducir en exceso de velocidad	2	0
39	sabado	h16	estrellamiento	carigan	imprudencia del conductor	1	0
57	martes	h07	choque lateral perpendicular	san sebastian	imprudencia del conductor	0	0
187	sabado	h21	estrellamiento	el sagrario	conducir en estado de embriaguez	0	0
101	martes	h21	choque lateral angular	el sagrario	conducir en exceso de velocidad	1	0

df.head()

/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` arguments

# tipos de la variables
ds.dtypes

dia object
hora object
tipologia object
parroquia\_urbana object
causas object
nro\_heridos int64
nro\_fallecidos
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
         no respetar las senales de transito
         fallas mecanicas no previsibles
                                                                                   6
         no mantener la distancia reglamentaria
         condiciones climaticas desfavorables
        no ceder el derecho de via
         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/python 3.10/dist-packages/ipykernel/ipkernel.py: 283: \ Deprecation Warning: 1.00 and 1.00 are also between 1.00 and 1.00 are also between 1.00 are also betwee
         `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` argumer
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
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 if x == 'carigan' else
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' 'arrollamiento' 'volcamiento' 'roce positivo' 'rozamiento'
                                                                                                              'atipico'
           '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
                                    0 if x == 'h01' else
                                    0 if x == 'h02' else
                                    0 if x == 'h03' else
                                    0 if x == 'h04' else
                                    0 if x == 'h05' else
                                    1 if x == 'h06' else
                                    1 if x == 'h07' else
                                    1 if x == 'h08' else
                                    1 if x == 'h09' else
                                    1 if x == 'h10' else
                                    1 if x == 'h11' else
                                    2 if x == 'h12' else
                                    2 if x == 'h13' else
                                    2 if x == 'h14' else
                                    2 if x == 'h15' else
                                    2 if x == 'h16' else
                                    2 if x == 'h17' else
                                    3 if x == 'h18' else
                                    3 \text{ if } x == 'h19' \text{ else}
                                    3 if x == 'h20' else
                                    3 if x == 'h21' else
                                    3 if x == 'h22' else
                                    3)
df.head()
```

ıl.

dia hora tipologia parroquia\_urbana causas nro\_heridos nro\_fallecidos 🥻

# 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

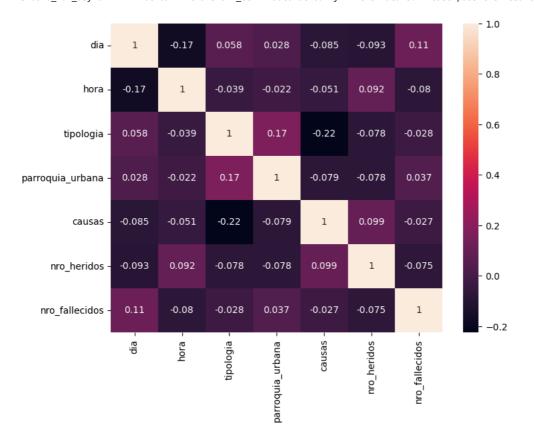
	dia	hora	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos	1	ılı
174	4	3	8	4	2	0	0		
133	6	3	10	5	1	0	0		
179	7	0	13	4	2	0	0		
353	3	2	9	4	2	0	0		
38	7	0	13	4	4	0	0		
83	5	3	14	4	4	2	0		
75	3	2	8	2	2	0	0		
102	5	0	13	4	1	0	0		
369	4	3	10	4	1	0	0		
89	3	3	13	3	1	0	0		

Verificacion de correlacion de variables

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

/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



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

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

	dia	tipologia	parroquia_urbana	causas	nro_heridos	nro_fallecidos
0	5	13	3	1	1	0
1	2	9	5	1	1	0
2	2	10	4	10	0	0
3	5	3	4	9	1	0
4	6	8	4	1	0	0

feature\_names = X.columns.tolist()

```
# Imprimir los nombres de las características
print(feature_names)
```

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

**→** 

### Revisamos la columna a predecir seleccionada

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

	hora	7	ıl.
0	1		
1	2		
2	2		
3	1		
4	1		

### Cargamos de modelo 2018-2020

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

/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

/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 <a href="https://scikit-learn.org/stable/model\_persistence.html#security-maintainability-limitations">https://scikit-learn.org/stable/model\_persistence.html#security-maintainability-limitations</a>

**←** 

### Verificamos los datos a predecir

### print(y\_pred)

<sup>`</sup>should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to `transformed\_cell` arguments

`should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to `transformed\_cell` argumer

```
Configuramos la metrica de clasificación
```

```
# 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()
print(pre)
pre.to_excel("resumen_precision_hora_2021.xlsx")
                                     precision
                                                               recall f1-score
                                                                                                          support
                                        0.357664 0.742424 0.482759
                                                                                                      66.000000
                                        0.000000 0.000000 0.000000
                                                                                                     80.000000
          1
          2
                                        0.409836 0.681818 0.511945 110.000000
                                        0.600000 0.263158 0.365854 114.000000
          3
          accuracy
                                        0.416216 0.416216 0.416216
                                                                                                       0.416216
          macro avg
                                        0.341875 0.421850 0.340139 370.000000
          weighted avg 0.370508 0.416216 0.351036 370.000000
          /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()
```

# Predicciones de los datos obtenidas

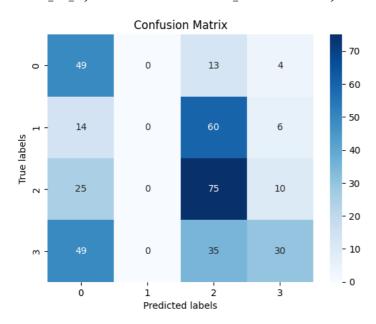
- 0 0
- 1 3
- **2** 2
- **3** 2
- **4** 3

### 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');
```

`should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result 1



Concatenamos los datos de las horas originales con los datos predichos

```
#Para concatenar variables
import pandas as pd

prediccion = pd.DataFrame(y_pred,columns=['hora_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

	hora_prediccion	hora	1	ılı
0	0	1		
1	3	2		
2	2	2		
3	2	1		
4	3	1		
5	2	1		
6	0	2		
7	0	3		
8	3	0		
9	2	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)
```

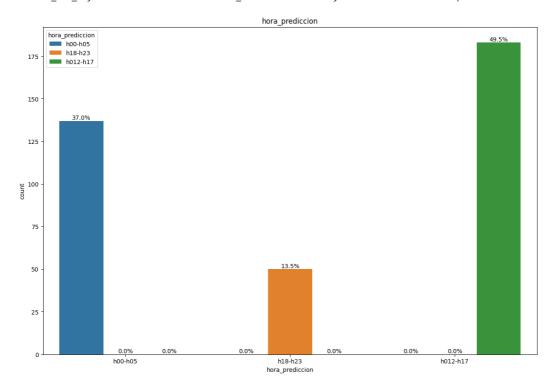
ax[0].savefig("barras\_original\_hora\_2021.png")

```
Exactitud árboles de decisión: 41.62162162162
     Exhaustividad árboles de decisión: 41.62162162162
     Precision árboles de decisión: 41.62162162162162
     /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['hora'] = df_combined['hora'].apply(lambda x:
                                    'h00-h05' if x == 0 else
                                    'h06-h011' if x == 1 else
                                    'h012-h17' if x == 2 else
                                    'h18-h23')
df_combined['hora_prediccion'] = df_combined['hora_prediccion'].apply(lambda x:
                                    'h00-h05' if x == 0 else
                                    'h06-h011' if x == 1 else
                                    'h012-h17' if x == 2 else
                                    'h18-h23')
     /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
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)
#ax[1].plot(df['dia'],df['hora'],color = 'tab:purple')
sns.countplot(x='hora',hue = 'hora', data = df\_combined, ax = ax[1]) #Muestre el conteo de observaciones en cada contenedor categórico u
ax[1].set_title('hora')
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),
           ha='center', va='bottom') # set the alignment of the text
#Guardamos la figura de barras original
```

```
hora
                                                                                                                                                                                                                                                                                                                                                                                                                              h06-h011
                                                                                                                                                                                     29.7%
                                                                                                                                                                                                                                                                                                                                                                                                                                         h012-h17
                                                                                                                                                                                                                                                                                                                                                                                                                             h18-h23
                                                                                                                                                                                                                                                                                                                                                                                                                               h00-h05
                                100
                                   80
  Graficamos las estadisticas predecidas de los datos
                          ē 60 1
 ax = plt.subplots(figsize = (15,10))
ncount=len(df_combined)
sns.countplot(x='hora\_prediccion', hue = 'hora\_prediccion', data = df\_combined, ax = ax[1]) \ \#Muestre \ el \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ de \ observaciones \ en \ cada \ conteo \ en \ cada \
ax[1].set_title('hora_prediccion')
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),
                                                   ha='center', va='bottom') # set the alignment of the text
#Guardamos la figura de barras de predicción
```

ax[0].savefig("barras\_prediccion\_hora\_2021.png")

`should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result 1



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

```
#target_names_str = [str(name) for name in target_names]
variable = df['hora'].apply(lambda x:
```

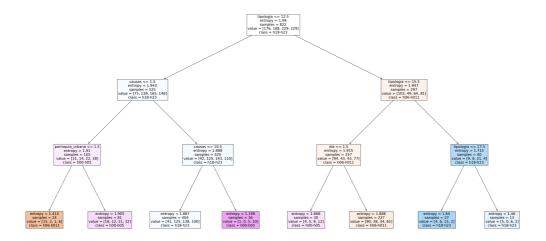
```
'h00-h05' if x == 0 else
'h06-h011' if x == 1 else
'h012-h17' if x == 2 else
'h18-h23')
claseVar = variable.unique().tolist()
target_names_str = [str(name) for name in claseVar]
```

`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\_hora\_2021.png")

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

```
#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: [39.65 28.63 14.98 14.98 28.63 14.98 39.65 28.63 14.98 39.65 28.63
     28.63 14.98 14.98 14.98 39.65 28.63 39.65 28.63 14.98 28.63 39.65 39.65
     14.98 14.98 14.98 14.98 39.65 39.65 14.98 28.63 14.98 14.98 28.63 39.65
     14.98 39.65 39.65 39.65 14.98 39.65 39.65 39.65 39.65 28.63 14.98
      39.65 14.98 39.65 14.98 28.63 39.65 39.65 14.98 28.63 14.98 14.98
     14.98 14.98 39.65 14.98 39.65 14.98 39.65 14.98 14.98 14.98 28.63 14.98
      28.63 28.63 28.63 14.98 39.65 14.98 14.98 14.98 14.98 39.65 39.65 39.65
      14.98 14.98 14.98 14.98 28.63 39.65 14.98 39.65 14.98 28.63 39.65 14.98
      39.65 14.98 39.65 14.98 39.65 14.98 39.65 39.65 14.98 14.98 14.98
     14.98 14.98 39.65 14.98 14.98 28.63 39.65 28.63 14.98 14.98 14.98 28.63
      39.65 14.98 39.65 28.63 28.63 39.65 14.98 39.65 14.98 14.98 39.65 39.65
     39.65 28.63 39.65 39.65 28.63 14.98 14.98 14.98 39.65 14.98 28.63 28.63
     14.98 39.65 39.65 39.65 14.98 14.98 14.98 14.98 14.98 14.98 14.98 14.98
     39.65 14.98 39.65 39.65 28.63 14.98 39.65 14.98 14.98 28.63 14.98 14.98
      39.65 39.65 28.63 14.98 28.63 14.98 14.98 28.63 14.98 14.98 39.65 39.65
      14.98 14.98 14.98 39.65 28.63 39.65 39.65 39.65 39.65 28.63 14.98 39.65
      39.65 14.98 14.98 14.98 39.65 39.65 39.65 14.98 28.63 39.65 14.98 14.98
      14.98 14.98 14.98 39.65 14.98 39.65 14.98 14.98 39.65 39.65 39.65 39.65
      39.65 39.65 14.98 14.98 14.98 39.65 39.65 14.98 14.98 39.65 14.98 39.65
     14.98 14.98 39.65 39.65 39.65 39.65 14.98 39.65 14.98 14.98 14.98 39.65
     14.98 14.98 14.98 39.65 14.98 14.98 14.98 14.98 14.98 39.65 14.98
      14.98 14.98 39.65 39.65 14.98 28.63 14.98 39.65 14.98 39.65 14.98 14.98
```

```
modelo HORA Weka python Calles accidentes 2021.ipynb - Colaboratory
      14.98 14.98 28.63 14.98 39.65 14.98 14.98 39.65 14.98 14.98 39.65 39.65
      14.98 39.65 14.98 14.98 39.65 14.98 14.98 14.98 14.98 39.65 39.65 28.63
      39.65 14.98 28.63 39.65 39.65 39.65 28.63 14.98 14.98 39.65 39.65 39.65
      39.65 39.65 14.98 39.65 39.65 39.65 39.65 39.65 39.65 14.98 39.65 14.98
      28.63 14.98 39.65 14.98 14.98 28.63 39.65 14.98 28.63 14.98 39.65 39.65
      39.65 39.65 39.65 39.65 39.65 39.65 14.98 14.98 39.65 14.98 14.98
      39.65 39.65 14.98 39.65 14.98 28.63 14.98 14.98 14.98 14.98 14.98 14.98
      39.65 14.98 28.63 14.98 14.98 14.98 14.98 14.98 14.98 28.63 14.98 14.98
      14.98 39.65 28.63 14.98 14.98 14.98 14.98 14.98 28.63 28.63 3
print(type(probabilidad_acierto))
columna_probabilidades = pd.DataFrame()
columna_probabilidades['probabilidad'] = pd.DataFrame(probabilidad_acierto)
print(columna_probabilidades)
     <class 'numpy.ndarray'>
          probabilidad
     0
                 39.65
     1
                 28.63
     2
                 14.98
     3
                 14.98
     4
                 28.63
     365
                 14.98
     366
                 14.98
     367
                 14.98
     368
                 28.63
     369
                 28.63
     [370 rows x 1 columns]
     /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
```

### 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
185	7	1	13	6	1	0	0	39.65	
77	5	2	9	1	2	2	0	14.98	
51	3	2	9	6	2	0	0	14.98	
11	5	2	8	4	12	0	0	28.63	
314	7	1	10	1	1	1	0	39.65	
187	6	3	13	1	1	0	0	39.65	
126	2	2	8	3	2	2	0	14.98	
328	6	2	13	5	2	0	0	39.65	
4	6	1	8	4	1	0	0	28.63	
25	2	1	9	1	4	0	0	14.98	

```
probabilidades_2021_df['predicciones']= probabilidades_2021_df['predicciones'].apply(lambda x:
                                    'h00-h05' if x == 0 else
                                    'h06-h011' if x == 1 else
                                    'h012-h17' if x == 2 else
```

'h18-h23') 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-h05' if x == 0 else
                                                                        'h06-h011' if x == 1 else
                                                                        'h012-h17' if x == 2 else
                                                                        'h18-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)
```

```
/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 probabilid

```
import pandas as pd
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_hora_2021.pdf') as pdf:
    pdf.savefig(fig, bbox_inches='tight')
```

			11	odelo_HC	INA_MEK	_pyu1011_	_Calles_a	ccidentes_
pers	8.40	an-species	***	restor or reside de restração e			**	9.40
motors	80.61	alcelanion		contact or records obtained	,		10.	81.01
nimin.	BLAGI	atolianism	pone	contact or recovering arterials		1	265	80.00
wheth	60321	at-danieri	miga-	conductive retrain de embringues			10.	81.01
state	86.60	ati-Geriania	power	conductors extends de contribuyan			945	81.01
mates.	BLOT	altitus	erige.	Improduce is delicated as to	2		111	RUASI
nimales.	86.60	windowien	shapute	conductries extends de medicinguas			946	86.60
jeren	81101	periodeconst	pone	construir no valuale de medicingua e	1		240	86.101
miles	803A21	ato Garcianio		Improduce del conductor			10.	81.01
simile.	81101	atrollaminos	pome	construir no retrada de mediciagos e			240	86.60
state	86.401	anderson.	pone	contains recover at admind			200	81.03
mates:	86.60	at-desire	servicin	conductrum ratiosis de medicingua e		- 1	10	86.60
-	80A0	drope bland promining	risile	ingendenia del cambator			111	#UAL!
	_			ingradienia del cambata				
jere	81.0	sintanini					10.	81.0
jeves	BENET	drope bland population	pone	ingradenia dili sedade			111	#UAL!
	BEAGI	drops slaves	sentetie	conductive ratiosh de mobility as			20	Blich
derings	86,600	drops borist sametrics		Improduce in delicandaries		1	111	WUALI
derings	86,653	drops slaves	chapuis	conductrum ratiosle de medicinguas	1		940	86.101
nimala.	80.621	topols.	serveletien	Improduce del produce	1		211	WELLAST
state	MANEE	drope bired propridingly	stagoria	ingradionia del cardador		1	211	#UAL!
min.	8UA21	drope bireal propositivals	westelle	contains which developes	1		20	Blich
derings	BLCI	atrolaminto	servicin	contains raint de métriques		1	10	86.60
	60A1	drope litted propridente	chaputa	ingradenia di Landaria			111	60.61
meters		_		ingradenia dili andador ingradenia dili andador				
inn	BLOI	atrifactoris	miga:	_	,		20	61/01
inn	86,6751	drope bired populársky	sereletin	sa cogerior la camales de insculie	1	1	211	#UAL!
jeren	86.ME1	stribeted	migra	Improduceia del candante	1	1	10.	81.01
ámiqu	BEAGI	atorilaminoto	servicin	conductron colodo de medelogram	1	1	265	81.63
ninin	BEAGI	atolianiaris	miga-	ingradice is differentiate			90.	81.03
jeren	BLOS	atolianisto	chapute	ingendenia del candador			10.	81.03
sinis	BLACI	sinfanisin	poss	ingradienia del candador			20.	81.0
winds	BEACT.	ato Garden	servicio	Improduce deli-sendante			10	8143
sinis	80At1	atributaria	power	ingraducia di Cambalan			10.	816
	80.00	andreas		ingredients differentiate		-	10	81.0
derings			chapuis	_				
sint:	BLIOT	at-decisio	617	ingredenia del cantador	,	1	16.	86.601
siele	86,600	drope Mendangdor	pone	ingradienta di Cambrina			111	80.421
inn.	86.6003	shope birmi poperalisalu-	ebulle	Improduce is delicated aster			111	#UAI1
min	81101	atorilaminos	wiger	constrair no retrait de medicinguas			240	81.101
nimalin.	MEASES	drope Mend propositivals	pone	na corporter les samules de insoulie	1.		111	WELFALT
i	WD341	drope silene	seretetie	Improduce delicandados			111	8UA01
state	81101	and and a	pone	conductive reduck de embringues		- 1	200	81.61
derings	BLICE	at-deniero	pone	contains which developes			10.	810
sinte	80A0	drope bland paperalisabe		ingradionia del cardador	1	-	111	6UA11
				_				
	86.101	at-decisio	elude	ingendenia dili undata	,	1	10.	81.0
-	86.60	drops Mendangler	shapote	Improduced differentiation	1		111	#UA21
inn	80At1	ato Gardenia		Improduce delicandados			201	BELOT
ámigo	80A21	shoops bireal angular		Improduced delicandados	1	1	211	#UAI1
derings	BLOT	drope bimalangia	***	Improduce delicated and	1		211	#UAL!
min	86.60	drope bland angles	servicin	ingendenia del candador		- 1	111	#UAC1
nimin.	86.601	drope birnel proprietable	rhaporis	Improduce delicandados	1	- 1	111	#UAC1
nimin.	81,01	drope Mendangler	sentelin	ingradienia di Cambalan			111	#UAI!
min.	60A01	rept.		ingradenia di Cambata		-	111	#UAC1
	_							
delap	86.601	perdidupida	shale	contact or consultrational			10.	81.01
mates:	BEARES	drope Mend propositivals	chapute	Improduces delicandados	,	1	111	eu.ur
ine.	BLIGH	at riberton	pone	conductive recovered colonidad		1	20	61.01
-	86.601	drope Mendangler		Improduccia del candante		1	111	80.421
jeres	80A21	mp th	chaponis	improduce del produce	1	1	111	BUALI
nimale.	60AH	drops blend popularies		improduceia del candante			111	80.61
winds	BEAU		elude	ingredenia del candador		- 1	111	8UA21
jeres	86.6001	drope slame		Improduceia del candante			11	60.41
min	BLOT	drope bland paymelinder		improduceia dell'ambator	1	1	211	#UAt1
min	BLICE	(tops sizes	elude	rando ir ar salada da mediningan		-	20	81401
	_	_		_				
inn.	80.61	etics.	elude	ingradionia del cardante	1	1	111	80.41
min	60At1	1901	servicin	ingradenia dili sedado	'		111	#UAC1
min.	BLOT	no repite	shapuis	ingradionia del cardador	1	1	111	#UAZ1
state	BEACE	drops shares	shapuis	conductors exhalt de melologues	1	1	945	86.60
delays	60A21	drope bireal angles	***	conductron ratiosk de medicinguas			20	Blich
jeres	80.61	drope blend poperalisals:		Improduce is delicated as		- 1	111	#UAI1
min.	86.6001	drope birnel propositionie	sentelin	ingredienia del candador	1		111	#UAC1
min.	86.601	drope Mendangler	servicie	ingradienia del cambaire	1	1	111	#UA21
inn.	#DAST	(tops sizes	eige.	ingradienia del cardador		-	111	6UA11
	_							
	BENET	drope Mendangler	sentelian	ingradionia del cardante			**	#UAL!
elecules.		drope Mend prepriatedor	seretelie	conductive extends de embringues		1	20	BLICE
atendo.	Blick	drops slaves		sando ir ne satuda da mederlagua e			201	Blick