Modelo

October 27, 2025

0.1 Propuesta de Negocio

La propuesta consiste en desarrollar una herramienta basada en aprendizaje supervisado que permita a la empresa **evaluar y optimizar la estrategia** de ventas de sus diferentes regiones y modelos de vehículos.

Para ello, se utilizará la base de datos de ventas disponible, la cual contiene información detallada sobre las características de los vehículos, el desempeño comercial y las zonas geográficas de distribución.

El modelo tendrá como objetivo clasificar el desempeño de ventas (por ejemplo, en categorías como bajo, medio o alto) a partir de variables explicativas tales como el tipo de vehículo, la región, el precio promedio, los incentivos aplicados, entre otras.

0.2 Alcance de la Propuesta

Con esta herramienta, la empresa podrá: - **Identificar patrones** y factores clave que determinan el éxito de ventas en cada región. - **Comparar el rendimiento** de diferentes modelos bajo condiciones de mercado específicas. - **Apoyar la toma de decisiones** en la asignación de recursos comerciales y campañas de marketing. En última instancia, el desarrollo del modelo busca fortalecer la estrategia comercial mediante el uso de técnicas de ciencia de datos y aprendizaje automático, ofreciendo un enfoque predictivo y sustentado en evidencia cuantitativa.

0.3 Análisis descriptivo de los datos

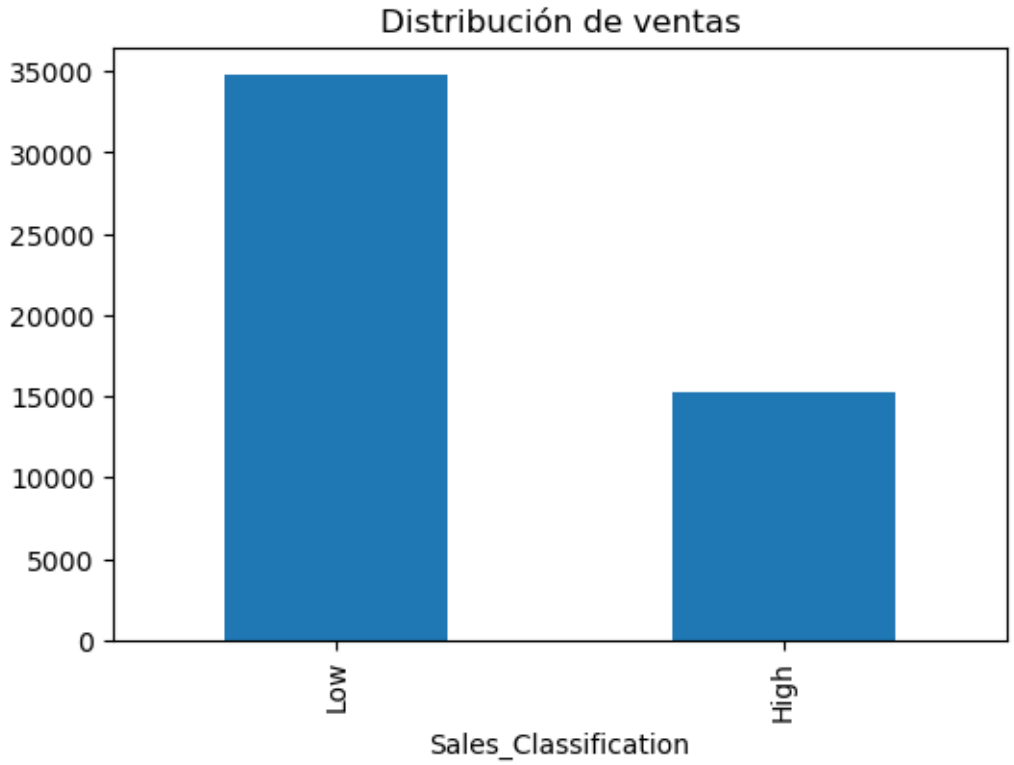
[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
url = "https://raw.githubusercontent.com/jhon1142/Proyecto_Despliegue/main/
↪BMW%20sales%20data%20(2010-2024)%20(1).csv"
df = pd.read_csv(url, encoding='utf-8')

print(df.head())
```

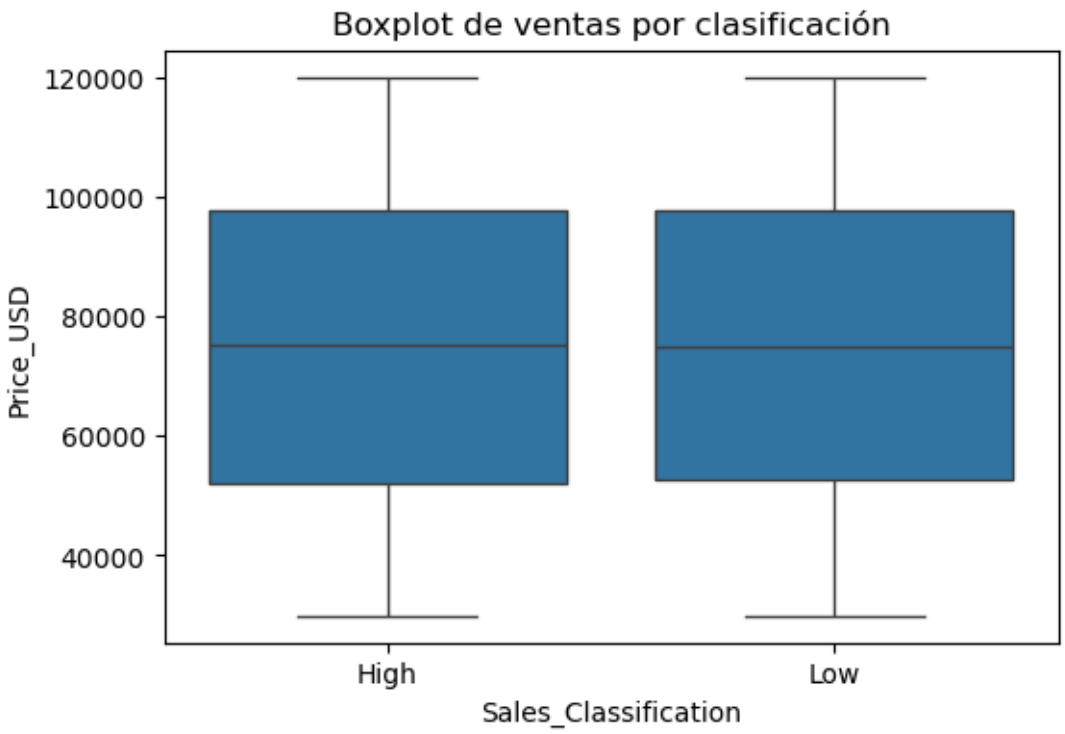
```
      Model  Year         Region  Color Fuel_Type Transmission  Engine_Size_L  \
0  5 Series  2016           Asia    Red    Petrol       Manual            3.5
```

```
1        i8  2013  North America    Red    Hybrid    Automatic            1.6
2  5 Series  2022  North America   Blue    Petrol    Automatic            4.5
3        X3  2024    Middle East   Blue    Petrol    Automatic            1.7
4  7 Series  2020  South America  Black    Diesel       Manual            2.1

   Mileage_KM  Price_USD  Sales_Volume Sales_Classification
0      151748      98740          8300                 High
1      121671      79219          3428                  Low
2       10991     113265          6994                  Low
3       27255      60971          4047                  Low
4      122131      49898          3080                  Low
```

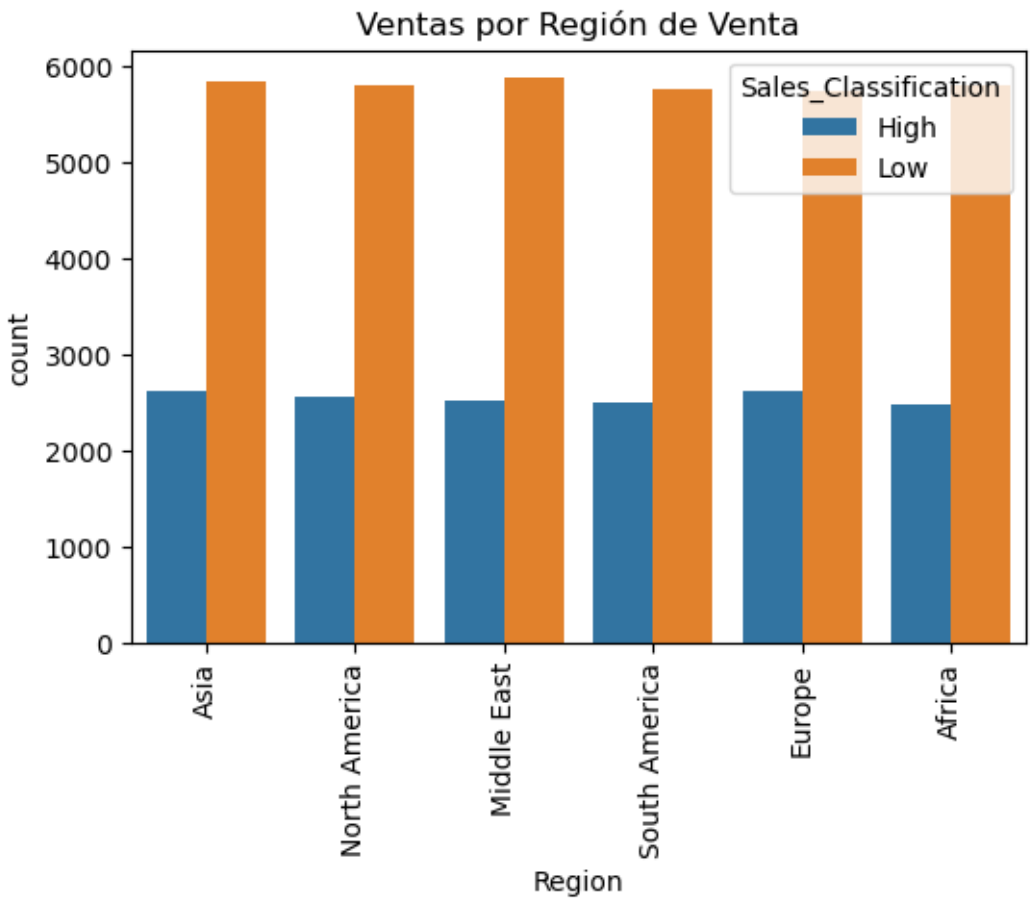
[3]:
```
plt.figure(figsize=(6, 4))
df['Sales_Classification'].value_counts().plot(kind='bar')
plt.title('Distribución de ventas')
plt.show()
```



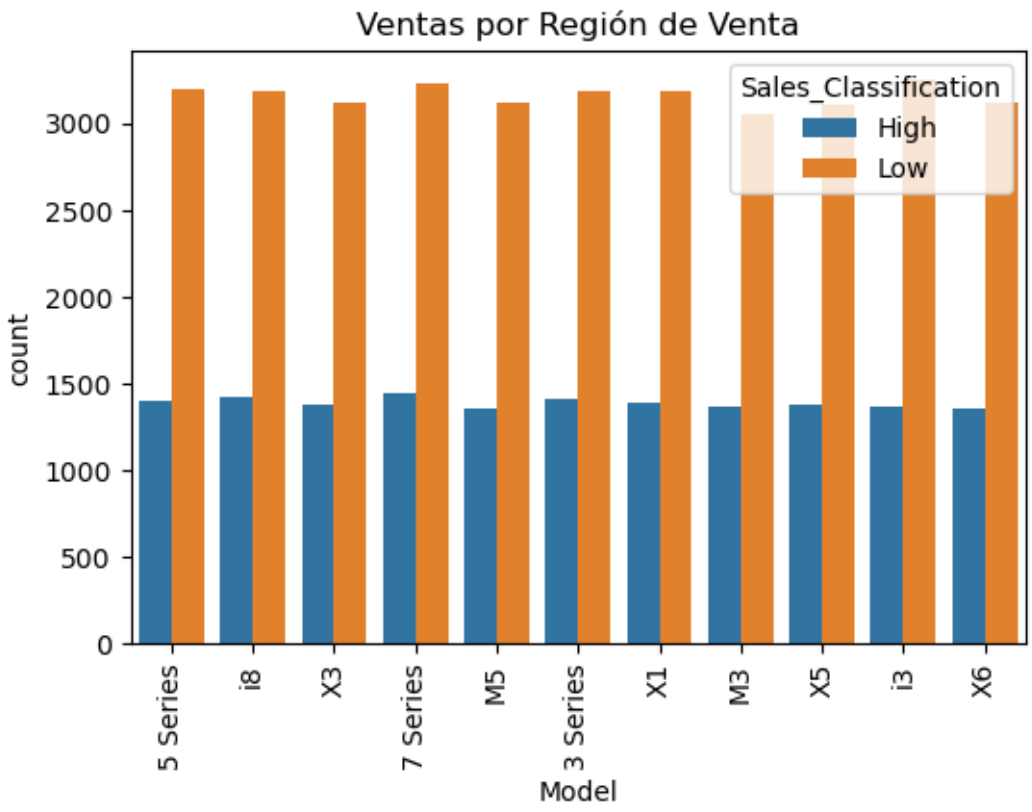
[4]:
```
plt.figure(figsize=(6, 4))
sns.boxplot(x='Sales_Classification', y='Price_USD', data=df)
plt.title('Boxplot de ventas por clasificación')
plt.show()
```



[5]:
```
plt.figure(figsize=(6, 4))
sns.countplot(x='Region', hue='Sales_Classification', data=df)
plt.xticks(rotation=90)
plt.title('Ventas por Región de Venta')
plt.show()
```

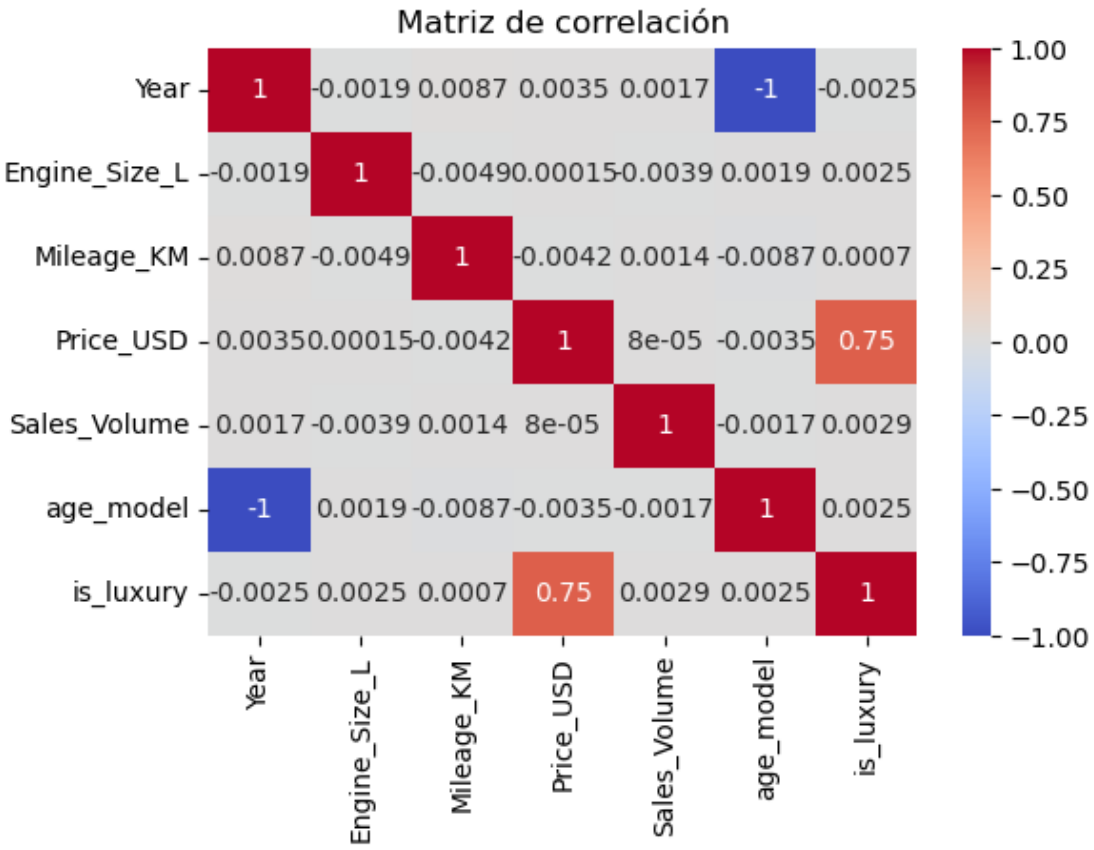


[6]:
```
plt.figure(figsize=(6, 4))
sns.countplot(x='Model', hue='Sales_Classification', data=df)
plt.xticks(rotation=90)
plt.title('Ventas por Región de Venta')
plt.show()
```



[7]:
```
df['age_model'] = 2024 - df['Year']
df['is_luxury'] = (df['Price_USD'] > df['Price_USD'].quantile(0.75)).astype(int)
```

[8]:
```
corr = df.corr(numeric_only=True)
plt.figure(figsize=(6, 4))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Matriz de correlación')
plt.show()
```



0.4 Clasificacion de autos

[9]:
```python
def clasificar_segmento(ref):
    ref = ref.upper()
    if "I3" in ref or "ELECTRIC" in ref or "HYBRID" in ref:
        return "Eléctrico / Híbrido"
    elif "I8" in ref or "M" in ref or "Z" in ref:
        return "Deportivo"
    elif "X" in ref:
        return "Camioneta / SUV"
    elif "7" in ref or "5" in ref:
        return "Ejecutivo"
    elif "3" in ref:
        return "Sedán"
    else:
        return "Otro"

df['Segmento'] = df['Model'].apply(clasificar_segmento)
df
```

[9]:
```
          Model  Year         Region   Color Fuel_Type Transmission  \
0      5 Series  2016           Asia     Red    Petrol       Manual
1            i8  2013  North America     Red    Hybrid    Automatic
2      5 Series  2022  North America    Blue    Petrol    Automatic
3            X3  2024    Middle East    Blue    Petrol    Automatic
4      7 Series  2020  South America   Black    Diesel       Manual
…          …   …            …      …       …          …
49995        i3  2014           Asia     Red    Hybrid       Manual
49996        i3  2023    Middle East  Silver  Electric       Manual
49997  5 Series  2010    Middle East     Red    Petrol    Automatic
49998        i3  2020           Asia   White  Electric    Automatic
49999        X1  2020  North America    Blue    Diesel       Manual

       Engine_Size_L  Mileage_KM  Price_USD  Sales_Volume  \
0                3.5      151748      98740          8300
1                1.6      121671      79219          3428
2                4.5       10991     113265          6994
3                1.7       27255      60971          4047
4                2.1      122131      49898          3080
…              …          …          …            …
49995            4.6      151030      42932          8182
49996            4.2      147396      48714          9816
49997            4.5      174939      46126          8280
49998            3.8        3379      58566          9486
49999            3.3      171003      77492          1764

      Sales_Classification  age_model  is_luxury             Segmento
0                     High          8          1            Ejecutivo
1                      Low         11          0            Deportivo
2                      Low          2          1            Ejecutivo
3                      Low          0          0      Camioneta / SUV
4                      Low          4          0            Ejecutivo
…                    …        …         …                  …
49995                 High         10          0  Eléctrico / Híbrido
49996                 High          1          0  Eléctrico / Híbrido
49997                 High         14          0            Ejecutivo
49998                 High          4          0  Eléctrico / Híbrido
49999                  Low          4          0      Camioneta / SUV

[50000 rows x 14 columns]
```

[10]:
```
y = df['Sales_Classification']  # variable objetivo
X = df.drop(['Sales_Classification'], axis=1)  # variables predictoras
```

[11]:
```
X= pd.get_dummies(
    X,
```

```
    columns=['Model', 'Region', 'Color', 'Fuel_Type', 'Transmission',␣
 ↪'Segmento'],
    drop_first=True
)
X
```

[11]:

	Year	Engine_Size_L	Mileage_KM	Price_USD	Sales_Volume	age_model	\
0	2016	3.5	151748	98740	8300	8	
1	2013	1.6	121671	79219	3428	11	
2	2022	4.5	10991	113265	6994	2	
3	2024	1.7	27255	60971	4047	0	
4	2020	2.1	122131	49898	3080	4	
…	…	…	…	…	…	…	
49995	2014	4.6	151030	42932	8182	10	
49996	2023	4.2	147396	48714	9816	1	
49997	2010	4.5	174939	46126	8280	14	
49998	2020	3.8	3379	58566	9486	4	
49999	2020	3.3	171003	77492	1764	4	

	is_luxury	Model_5 Series	Model_7 Series	Model_M3	…	Color_Silver	\
0	1	True	False	False	…	False	
1	0	False	False	False	…	False	
2	1	True	False	False	…	False	
3	0	False	False	False	…	False	
4	0	False	True	False	…	False	
…	…	…	…	…	…	…	
49995	0	False	False	False	…	False	
49996	0	False	False	False	…	True	
49997	0	True	False	False	…	False	
49998	0	False	False	False	…	False	
49999	0	False	False	False	…	False	

	Color_White	Fuel_Type_Electric	Fuel_Type_Hybrid	Fuel_Type_Petrol	\
0	False	False	False	True	
1	False	False	True	False	
2	False	False	False	True	
3	False	False	False	True	
4	False	False	False	False	
…	…	…	…	…	
49995	False	False	True	False	
49996	False	True	False	False	
49997	False	False	False	True	
49998	True	True	False	False	
49999	False	False	False	False	

	Transmission_Manual	Segmento_Deportivo	Segmento_Ejecutivo	\
0	True	False	True	

```
1                     False                 True                 False
2                     False                False                 True
3                     False                False                False
4                      True                False                 True
…                      …                    …                    …
49995                  True                False                False
49996                  True                False                False
49997                 False                False                 True
49998                 False                False                False
49999                  True                False                False

       Segmento_Eléctrico / Híbrido  Segmento_Sedán
0                             False           False
1                             False           False
2                             False           False
3                             False           False
4                             False           False
…                             …               …
49995                          True           False
49996                          True           False
49997                         False           False
49998                          True           False
49999                         False           False

[50000 rows x 35 columns]
```

[12]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X = X.copy()
X[['Year', 'Engine_Size_L', 'Mileage_KM', 'Price_USD', 'Sales_Volume']] =␣
 ↪scaler.fit_transform(
    X[['Year', 'Engine_Size_L', 'Mileage_KM', 'Price_USD', 'Sales_Volume']]
)
X
```

[12]:
```
           Year  Engine_Size_L  Mileage_KM  Price_USD  Sales_Volume  \
0     -0.234876       0.250548    0.887814   0.911817      1.131530
1     -0.928611      -1.632377    0.368717   0.160951     -0.573911
2      1.152595       1.241561   -1.541504   1.470514      0.674365
3      1.615085      -1.533276   -1.260805  -0.540950     -0.357231
4      0.690105      -1.136871    0.376656  -0.966867     -0.695729
…           …              …           …          …             …
49995 -0.697366       1.340662    0.875423  -1.234811      1.090224
49996  1.383840       0.944257    0.812704  -1.012409      1.662205
49997 -1.622346       1.241561    1.288067  -1.111955      1.124529
49998  0.690105       0.547852   -1.672880  -0.633457      1.546689
```

```
49999  0.690105       0.052345    1.220136   0.094523     -1.156394

       age_model  is_luxury  Model_5 Series  Model_7 Series  Model_M3  …  \
0              8          1            True           False     False  …
1             11          0           False           False     False  …
2              2          1            True           False     False  …
3              0          0           False           False     False  …
4              4          0           False            True     False  …
…          …         …              …               …       …  …
49995         10          0           False           False     False  …
49996          1          0           False           False     False  …
49997         14          0            True           False     False  …
49998          4          0           False           False     False  …
49999          4          0           False           False     False  …

       Color_Silver  Color_White  Fuel_Type_Electric  Fuel_Type_Hybrid  \
0             False        False               False             False
1             False        False               False              True
2             False        False               False             False
3             False        False               False             False
4             False        False               False             False
…             …          …                   …               …
49995         False        False               False              True
49996          True        False                True             False
49997         False        False               False             False
49998         False         True                True             False
49999         False        False               False             False

       Fuel_Type_Petrol  Transmission_Manual  Segmento_Deportivo  \
0                  True                True               False
1                 False               False                True
2                  True               False               False
3                  True               False               False
4                 False                True               False
…                 …                  …                 …
49995             False                True               False
49996             False                True               False
49997              True               False               False
49998             False               False               False
49999             False                True               False

       Segmento_Ejecutivo  Segmento_Eléctrico / Híbrido  Segmento_Sedán
0                    True                         False           False
1                   False                         False           False
2                    True                         False           False
3                   False                         False           False
4                    True                         False           False
```

```
…                …                              …              …
49995            False                            True          False
49996            False                            True          False
49997             True                           False          False
49998            False                            True          False
49999            False                           False          False

[50000 rows x 35 columns]
```

[13]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42, stratify=y)
```

[14]:
```
from sklearn.linear_model import LogisticRegression
# --- MODELO DE REGRESIÓN LOGÍSTICA ---
logreg = LogisticRegression(
    max_iter=1000,
    solver='lbfgs',
    class_weight='balanced'
)
logreg.fit(X_train, y_train)

# --- PREDICCIÓN ---
y_pred = logreg.predict(X_test)
```

[15]:
```
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪accuracy_score
print(" Accuracy:", accuracy_score(y_test, y_pred))
print("\n Classification report:\n", classification_report(y_test, y_pred))
```

```
 Accuracy: 0.9939

 Classification report:
              precision    recall  f1-score   support

        High       0.98      1.00      0.99      3049
         Low       1.00      0.99      1.00      6951

    accuracy                           0.99     10000
   macro avg       0.99      1.00      0.99     10000
weighted avg       0.99      0.99      0.99     10000
```

[16]:
```
# --- MATRIZ DE CONFUSIÓN ---
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred, labels=logreg.classes_)
print(" Matriz de confusión (filas = reales, columnas = predichas):")
```

```
print(pd.DataFrame(cm, index=[f"Real_{c}" for c in logreg.classes_],
                       columns=[f"Pred_{c}" for c in logreg.classes_]))
```

```
 Matriz de confusión (filas = reales, columnas = predichas):
           Pred_High  Pred_Low
Real_High       3049         0
Real_Low          61      6890
```

[17]:
```
coef_df = pd.DataFrame({
    'Variable': X.columns,
    'Coeficiente': logreg.coef_[0]
}).sort_values(by='Coeficiente', ascending=False)

coef_df.head(10)  # Las 10 variables que más aumentan la probabilidad de "High"
```

[17]:

	Variable	Coeficiente
0	Year	7.130376
5	age_model	1.647322
34	Segmento_Sedán	0.795118
12	Model_X3	0.789978
11	Model_X1	0.608213
14	Model_X6	0.596089
13	Model_X5	0.582423
31	Segmento_Deportivo	0.478289
32	Segmento_Ejecutivo	0.474004
33	Segmento_Eléctrico / Híbrido	0.326608

[18]:
```
import numpy as np

coef_df['Odds_Ratio'] = np.exp(coef_df['Coeficiente'])
coef_df['Interpretación'] = coef_df['Coeficiente'].apply(
    lambda x: '↑ Aumenta prob. ventas altas' if x > 0 else '↓ Disminuye prob. ventas altas'
)
coef_df.head(10)
```

[18]:

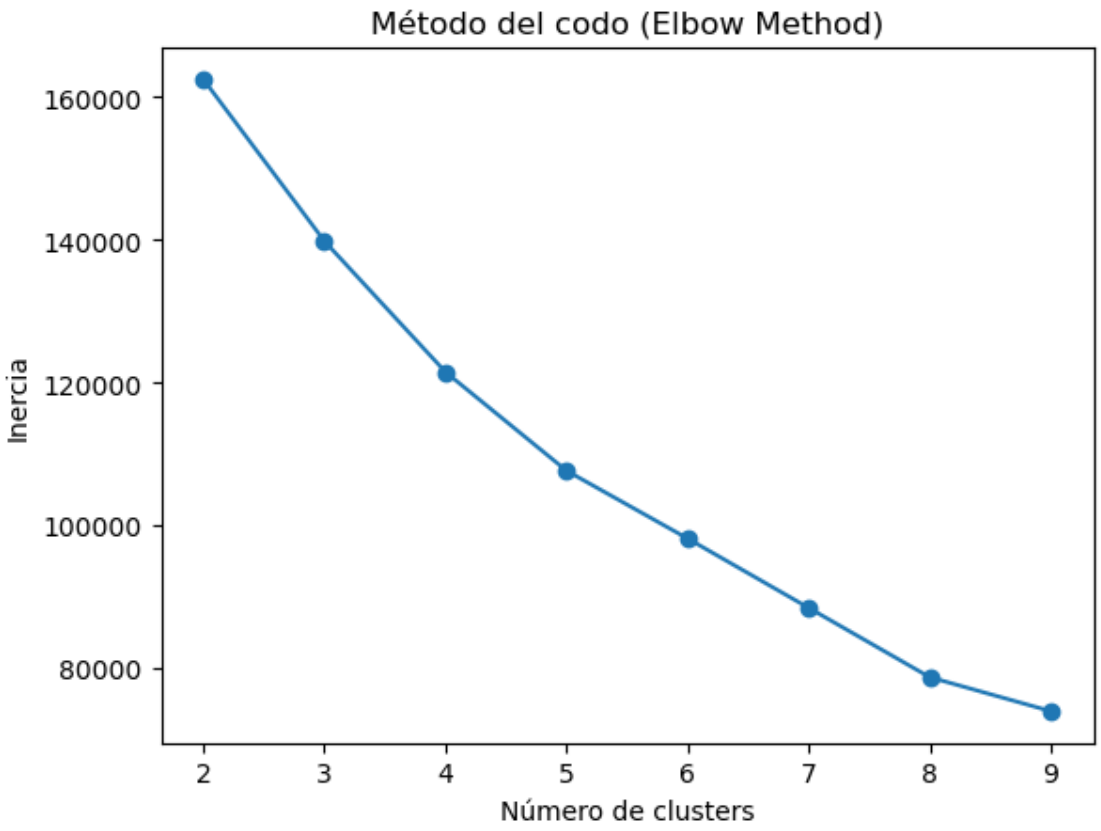
	Variable	Coeficiente	Odds_Ratio	\
0	Year	7.130376	1249.347114	
5	age_model	1.647322	5.193055	
34	Segmento_Sedán	0.795118	2.214703	
12	Model_X3	0.789978	2.203347	
11	Model_X1	0.608213	1.837146	
14	Model_X6	0.596089	1.815006	
13	Model_X5	0.582423	1.790372	
31	Segmento_Deportivo	0.478289	1.613311	
32	Segmento_Ejecutivo	0.474004	1.606413	
33	Segmento_Eléctrico / Híbrido	0.326608	1.386258	

```
                  Interpretación
0   ↑ Aumenta prob. ventas altas
5   ↑ Aumenta prob. ventas altas
34  ↑ Aumenta prob. ventas altas
12  ↑ Aumenta prob. ventas altas
11  ↑ Aumenta prob. ventas altas
14  ↑ Aumenta prob. ventas altas
13  ↑ Aumenta prob. ventas altas
31  ↑ Aumenta prob. ventas altas
32  ↑ Aumenta prob. ventas altas
33  ↑ Aumenta prob. ventas altas
```

0.5 Clustering

[19]:
```python
X_cluster = df[['Engine_Size_L', 'Mileage_KM', 'Price_USD', 'Sales_Volume']]
X_cluster = (X_cluster - X_cluster.mean()) / X_cluster.std()  # escalado
```

[20]:
```python
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

inertia = []
for k in range(2, 10):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_cluster)
    inertia.append(kmeans.inertia_)

plt.plot(range(2, 10), inertia, marker='o')
plt.title('Método del codo (Elbow Method)')
plt.xlabel('Número de clusters')
plt.ylabel('Inercia')
plt.show()
```



[21]:
```
kmeans = KMeans(n_clusters=4, random_state=42)
df['Cluster'] = kmeans.fit_predict(X_cluster)
```

[22]:
```
cluster_summary = df.groupby('Cluster')[['Price_USD', 'Engine_Size_L',␣
↪'Sales_Volume']].mean()
cluster_summary
```

[22]:

Cluster	Price_USD	Engine_Size_L	Sales_Volume
0	99297.829222	3.259359	4432.425593
1	50786.748808	3.248008	5686.481296
2	80912.864108	3.216491	7719.188615
3	68692.701543	3.265164	2405.510206

[ ]: