Proyecto IA

Campaña Ricardo, Chalacama Erik, Delgado Joel, Román Paúl, Salazar Santiago, Ya

Tabla de contenidos


```
import pickle
import numpy as np
import pandas as pd
import sklearn.metrics as sm
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn import metrics
from sklearn import preprocessing
from sklearn.metrics import pairwise
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
import seaborn as sns
from pandas.plotting import parallel_coordinates
from sklearn.preprocessing import PolynomialFeatures
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import make_pipeline
from sklearn import datasets
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.utils import shuffle
```

```
data = pd.read_csv("inec_encuesta-estructural-emprresarial_establecimientos_2019.csv", del
data.sample(5)
```

	id_empresa	tipo_establecimiento_cod	tipo_establecimiento	provincia_cod	provincia_desc
2947	13705202090	Sucursal	Sucursal	TUNGURAHUA	TUNGURAHU
4217	13706014092	Sucursal	Sucursal	PICHINCHA	PICHINCHA
2132	13705121090	Sucursal	Sucursal	LOJA	LOJA
16162	14629457098	Matriz	Matriz	GUAYAS	GUAYAS
3546	13705657097	Único	Único	GUAYAS	GUAYAS

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23524 entries, 0 to 23523
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   id_empresa                23524 non-null  int64
 1   tipo_establecimiento_cod  23524 non-null  object
 2   tipo_establecimiento      23524 non-null  object
 3   provincia_cod             23524 non-null  object
 4   provincia_desc            23524 non-null  object
 5   ciiu_cod                  23524 non-null  object
 6   ciiu_desc                 23524 non-null  object
 7   personal_ocupado          23524 non-null  int64
 8   sueldos                   23524 non-null  int64
 9   ventas                    23524 non-null  int64
dtypes: int64(4), object(6)
memory usage: 1.8+ MB
```

```
data.drop(columns=['id_empresa'], inplace=True)
```

```
data.columns
```

```
Index(['tipo_establecimiento_cod', 'tipo_establecimiento', 'provincia_cod',
       'provincia_desc', 'ciiu_cod', 'ciiu_desc', 'personal_ocupado',
       'sueldos', 'ventas'],
      dtype='object')
```

```
# Elimina las filas en las que al menos una columna contenga un espacio en blanco o cadena
data_cleaned = data[~(data == ' ').any(axis=1) & ~(data == '').any(axis=1)].copy()

# Imprime el número de filas antes y después de la eliminación
print("Número de filas antes de eliminación:", len(data))
print("Número de filas después de limpieza:", len(data_cleaned))
```

```
Número de filas antes de eliminación: 23524
Número de filas después de limpieza: 21695
```

```
# Aplica strip() a todas las celdas del DataFrame para eliminar espacios en blanco
data = data.applymap(lambda x: x.strip() if isinstance(x, str) else x)
```

```
data_cleaned['ciiu_cod'].value_counts()
```

```
VENTA AL POR MENOR DE PRODUCTOS FARMACÉUTICOS Y MEDICINALES, COSMÉTICOS Y ARTÍCULOS DE TOCAD
VENTA AL POR MAYOR DE OTROS ENSERES DOMÉSTICOS.
ACTIVIDADES DE RESTAURANTES Y DE SERVICIO MÓVIL DE COMIDAS.
VENTA AL POR MENOR DE APARATOS ELÉCTRICOS DE USO DOMÉSTICO, MUEBLES, EQUIPO DE ILUMINACIÓN Y
VENTA AL POR MENOR EN COMERCIOS NO ESPECIALIZADOS CON PREDOMINIO DE LA VENTA DE ALIMENTOS, BE

OTRAS ACTIVIDADES DE TECNOLOGÍA DE LA INFORMACIÓN Y DE SERVICIOS INFORMÁTICOS.
FABRICACIÓN DE GAS; DISTRIBUCIÓN DE COMBUSTIBLES GASEOSOS POR TUBERÍAS.
FABRICACIÓN DE EQUIPO ELÉCTRICO DE ILUMINACIÓN.
TRANSMISIONES DE RADIO.
FABRICACIÓN DE PARTES Y PIEZAS DE CARPINTERÍA PARA EDIFICIOS Y CONSTRUCCIONES.
Name: ciiu_cod, Length: 264, dtype: int64
```

```
data_cleaned['ciiu_desc'].value_counts()
```

```
Venta al por menor de productos farmacéuticos y medicinales, cosméticos y artículos de tocad
Venta al por mayor de otros enseres domésticos.
Actividades de restaurantes y de servicio móvil de comidas.
Venta al por menor de aparatos eléctricos de uso doméstico, muebles, equipo de iluminación y
Venta al por menor en comercios no especializados con predominio de la venta de alimentos, be

Otras actividades de tecnología de la información y de servicios informáticos.
Fabricación de gas; distribución de combustibles gaseosos por tuberías.
```

```
Fabricación de equipo eléctrico de iluminación.
Transmisiones de radio.
Fabricación de partes y piezas de carpintería para edificios y construcciones.
Name: ciiu_desc, Length: 264, dtype: int64
```

```
data_cleaned['provincia_desc'].value_counts()
```

```
GUAYAS                          6578
PICHINCHA                       6158
MANABÍ                          1553
AZUAY                           1281
EL ORO                           820
LOS RÍOS                         766
TUNGURAHUA                       686
STO. DOMINGO DE LOS TSÁCHILAS    644
IMBABURA                         453
LOJA                             388
ESMERALDAS                       346
CHIMBORAZO                       296
SANTA ELENA                      276
COTOPAXI                         262
ORELLANA                         214
SUCUMBÍOS                        214
CAÑAR                            175
CARCHI                           126
BOLÍVAR                           90
ZAMORA CHINCHIPE                  77
MORONA SANTIAGO                   76
GALÁPAGOS                         73
PASTAZA                           72
NAPO                              71
Name: provincia_desc, dtype: int64
```

```
data_cleaned['provincia_cod'].value_counts()
```

```
GUAYAS                          6578
PICHINCHA                       6158
MANABI                          1553
AZUAY                           1281
EL ORO                           820
```

```
LOS RIOS                          766
TUNGURAHUA                        686
STO. DOMINGO DE LOS TSACHILAS     644
IMBABURA                          453
LOJA                              388
ESMERALDAS                        346
CHIMBORAZO                        296
SANTA ELENA                       276
COTOPAXI                          262
ORELLANA                          214
SUCUMBIOS                         214
CAÑAR                             175
CARCHI                            126
BOLIVAR                            90
ZAMORA CHINCHIPE                   77
MORONA SANTIAGO                    76
GALAPAGOS                          73
PASTAZA                            72
NAPO                               71
Name: provincia_cod, dtype: int64
```

```
data_cleaned['tipo_establecimiento_cod'].value_counts()
```

```
Sucursal    17916
Matriz       2447
Único        1332
Name: tipo_establecimiento_cod, dtype: int64
```

```
data_cleaned['tipo_establecimiento'].value_counts()
```

```
Sucursal    17916
Matriz       2447
Único        1332
Name: tipo_establecimiento, dtype: int64
```

```
data_cleaned.drop(columns=['ciiu_cod', 'tipo_establecimiento_cod', 'provincia_cod'], inpla
```

```
data_cleaned.columns
```

```
Index(['tipo_establecimiento', 'provincia_desc', 'ciiu_desc',
       'personal_ocupado', 'sueldos', 'ventas'],
      dtype='object')
```

```
data.sample(10)
data_cleaned.sample(5)
```

	tipo_establecimiento	provincia_desc	ciiu_desc	personal_
23236	Sucursal	EL ORO	Venta al por mayor de otros enseres domésticos.	3
4238	Sucursal	PICHINCHA	Venta al por mayor de otros enseres domésticos.	7
13229	Sucursal	IMBABURA	Elaboración y conservación de carne.	50
373	Sucursal	GUAYAS	Venta al por mayor de otros enseres domésticos.	10
22351	Sucursal	GUAYAS	Venta al por menor de productos farmacéuticos ...	2

```
# Convertir las variables categóricas a tipo 'category'
data_cleaned['tipo_establecimiento'] = data_cleaned['tipo_establecimiento'].astype('catego
data_cleaned['provincia_desc'] = data_cleaned['provincia_desc'].astype('category')
data_cleaned['ciiu_desc'] = data_cleaned['ciiu_desc'].astype('category')

# Definir nuevas asignaciones de categorías basadas en los resultados de value_counts()
provincia_desc_categories = {
'GUAYAS': 0,
'PICHINCHA': 1,
'MANABÍ': 3,
'AZUAY': 4,
'EL ORO': 5,
'LOS RÍOS': 6,
'TUNGURAHUA': 7,
'STO. DOMINGO DE LOS TSÁCHILAS': 8,
'IMBABURA': 9,
'LOJA': 10,
'ESMERALDAS': 11,
'CHIMBORAZO': 12,
'SANTA ELENA': 13,
'COTOPAXI': 14,
'ORELLANA': 15,
'SUCUMBÍOS': 16,
'CAÑAR': 17,
'CARCHI': 18,
'BOLÍVAR': 19,
```

```
'ZAMORA CHINCHIPE': 20,
'MORONA SANTIAGO': 21,
'GALÁPAGOS': 22,
'PASTAZA': 23,
'NAPO': 24,
}

################################################
################################################

# Obtener los valores únicos de tipo_establecimiento
unique_tipo_establecimiento = data_cleaned['tipo_establecimiento'].unique()

# Crear el diccionario 'tipo_establecimientor_categories' usando un bucle for
tipo_establecimientor_categories = {}
for index, tipo_establecimiento in enumerate(unique_tipo_establecimiento):
    tipo_establecimientor_categories[tipo_establecimiento] = index

################################################
################################################

# Obtener los valores únicos de tipo_establecimiento
unique_ciiu_desc = data_cleaned['ciiu_desc'].unique()

# Crear el diccionario 'ciiu_desc_categories' usando un bucle for
ciiu_desc_categories = {}
for index, ciiu_desc in enumerate(unique_ciiu_desc):
    ciiu_desc_categories[ciiu_desc] = index
```

```
# Renombrar las categorías utilizando los nuevos códigos
data_cleaned['tipo_establecimiento'] = data_cleaned['tipo_establecimiento'].cat.rename_cat
data_cleaned['provincia_desc'] = data_cleaned['provincia_desc'].cat.rename_categories(prov
data_cleaned['ciiu_desc'] = data_cleaned['ciiu_desc'].cat.rename_categories(ciiu_desc_cate

# Convertir las variables categóricas a enteros
data_cleaned['tipo_establecimiento'] = data_cleaned['tipo_establecimiento'].astype('int64'
data_cleaned['provincia_desc'] = data_cleaned['provincia_desc'].astype('int64')
data_cleaned['ciiu_desc'] = data_cleaned['ciiu_desc'].astype('int64')

# Mostrar una muestra de los datos transformados
data_cleaned.sample(10)
```

	tipo_establecimiento	provincia_desc	ciiu_desc	personal_ocupado	sueldos	ventas
5424	2	3	2	7	46473	1901888
22187	2	3	15	0	0	0
17971	2	0	154	3	33027	4261323
13659	2	3	10	4	31733	496003
8712	2	9	62	80	616888	20393257
17936	2	1	4	15	10936	124962
15463	2	1	74	3	22771	622951
8330	2	0	137	249	2953816	11760923
1546	2	0	74	1	5970	0
8287	2	18	2	7	0	6144810

```
# Mostrar una muestra de los datos transformados

#data_cleaned=data_cleaned[data_cleaned['ventas']!=0]
#data_cleaned=data_cleaned[data_cleaned['ventas']>0]
data_cleaned = data_cleaned.reset_index(drop=True)
data_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21695 entries, 0 to 21694
Data columns (total 6 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   tipo_establecimiento  21695 non-null  int64
 1   provincia_desc        21695 non-null  int64
 2   ciiu_desc             21695 non-null  int64
 3   personal_ocupado      21695 non-null  int64
 4   sueldos               21695 non-null  int64
 5   ventas                21695 non-null  int64
dtypes: int64(6)
memory usage: 1017.1 KB
```

ENTRENAMIENTO DEL MODELO (REGRESION POLINOMIAL Y LINEAL)

```
y = data_cleaned['sueldos'] #variable target
X = data_cleaned.drop(columns=['sueldos']) #variables input
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=20)
print(X_train.shape,X_test.shape)
```

```
# Creamos una nueva observacion
nuevoSueldo= pd.DataFrame(
    [
         {
        "tipo_establecimiento": 1,
        "provincia_desc":4,
        "ciiu_desc":1,
        "personal_ocupado":2,
        "ventas":2117598
         }
    ])
nuevoSueldo
```

```
(15186, 5) (6509, 5)
```

	tipo_establecimiento	provincia_desc	ciiu_desc	personal_ocupado	ventas
0	1	4	1	2	2117598

```
scaler = preprocessing.StandardScaler()
scaler.fit(X_train[['tipo_establecimiento','provincia_desc','ciiu_desc','personal_ocupado'
#transformamos todo el dataset
data_cleaned_normalized=pd.concat([pd.DataFrame(scaler.transform(data_cleaned[['tipo_estab
                                                columns=['tipo_establecimiento','provincia_
                                    data_cleaned[['sueldos']]],axis=1)
print(data_cleaned_normalized.head())

X_train_norm = data_cleaned_normalized.iloc[X_train.index].drop(columns=['sueldos'])
X_test_norm = data_cleaned_normalized.iloc[X_test.index].drop(columns=['sueldos'])

print('Training sample:')
print(X_train_norm.iloc[:14, :])
#print(validNorm[0])

nuevoSueldo_norm = pd.DataFrame(scaler.transform(nuevoSueldo), columns=['tipo_establecimie
print('\nNueva obs:')
print(nuevoSueldo)
print('Nueva obs escalada:')
print(nuevoSueldo_norm)
```

```
   tipo_establecimiento  provincia_desc  ciiu_desc  personal_ocupado  \
0             -2.607276        0.070884  -0.958594          0.263368
1             -1.085246        0.070884  -0.940636         -0.195637
2             -1.085246        0.070884  -0.922678         -0.025361
3             -1.085246        0.070884  -0.904720          0.085689
4             -1.085246        0.070884  -0.886761         -0.069781

     ventas  sueldos
0  0.077898   600616
1 -0.050010    66000
2  0.024593   298257
3 -0.056455   319010
4 -0.063409   193291
Training sample:
       tipo_establecimiento  provincia_desc  ciiu_desc  personal_ocupado  \
10213              0.436784       -0.726481  -0.886761         -0.136410
12035              0.436784        0.270225   0.567846         -0.173427
9041               0.436784       -0.527140   0.531930         -0.121604
11898              0.436784       -0.527140   1.016799         -0.203040
17373              0.436784        0.070884   2.489364         -0.217847
797                0.436784       -0.726481  -0.761055          0.544694
14826              0.436784        0.270225  -0.743097         -0.232653
1285               0.436784        1.266931   0.352348         -0.240057
4090              -2.607276       -0.726481  -0.365976          0.574307
19608             -2.607276       -0.726481  -0.779013         -0.129007
6636               0.436784        1.067590  -0.743097         -0.129007
9298               0.436784       -0.726481  -0.186395         -0.247460
14319              0.436784        2.064295  -0.132521         -0.232653
6495               0.436784       -0.726481  -0.096604         -0.232653

         ventas
10213 -0.070054
12035 -0.058917
9041  -0.076331
11898 -0.078543
17373 -0.080638
797   -0.078744
14826 -0.081419
1285  -0.054861
4090  -0.004939
19608  0.000604
6636  -0.001066
9298  -0.084756
```

```
14319 -0.079642
6495  -0.039279

Nueva obs:
   tipo_establecimiento  provincia_desc  ciiu_desc  personal_ocupado   ventas
0                     1               4          1                 2  2117598
Nueva obs escalada:
   tipo_establecimiento  provincia_desc  ciiu_desc  personal_ocupado   ventas
0             -1.085246        0.070884  -0.940636         -0.232653 -0.05001
```

- Separamos data de training y testing:

```
#Cree y entrene el modelo de regresor lineal
# Create the linear regressor model
linear_regressor = linear_model.LinearRegression()
```

```
# Train the model using the training sets
linear_regressor.fit(X_train_norm, y_train)
```

```
LinearRegression()
```

```
# Realiza la predicción
predicted_price = linear_regressor.predict(nuevoSueldo_norm)

print("Predicted price:", predicted_price)
```

```
Predicted price: [144447.71579988]
```

```
# Predict the output
y_test_pred = linear_regressor.predict(X_test_norm)
```

```
# Calcule los valores para las diferentes métricas, consulte su interpretación
print("Linear regressor performance:")
print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, y_test_pred),
print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
```

```
Linear regressor performance:
Mean absolute error = 152721.72
Mean squared error = 422914277963.96
Median absolute error = 30704.72
Explain variance score = 0.76
R2 score = 0.76
```

```
with open('trained_model.pkl','wb') as file:
    pickle.dump(linear_regressor,file)
```

polinomio grado 2

```
degree = 2
polyreg = make_pipeline(PolynomialFeatures(degree), LinearRegression())
polyreg.fit(X_train_norm, y_train)
y_predicted = polyreg.predict(X_test_norm)
```

```
poly_reg_rmse = np.sqrt(mean_squared_error(y_test, y_predicted))
poly_reg_rmse
```

```
808602.3153096966
```

```
# Measure performance
print("Linear Regressor performance:")
print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_predicted), 2))
print("Mean squared error =", round(sm.mean_squared_error(y_test, y_predicted), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, y_predicted), 2))
print("Explained variance score =", round(sm.explained_variance_score(y_test, y_predicted)
print("R2 score =", round(sm.r2_score(y_test, y_predicted), 2))
```

```
Linear Regressor performance:
Mean absolute error = 170483.77
Mean squared error = 653837704324.2
Median absolute error = 40041.11
Explained variance score = 0.62
R2 score = 0.62
```

SVM regressor

```
# Create Support Vector Regression model
sv_regressor = SVR(kernel='poly',degree=5, C=1.0, epsilon=0.1)
```

```
# Train Support Vector Regressor
sv_regressor.fit(X_train_norm, y_train)
```

```
SVR(degree=5, kernel='poly')
```

```
# Evaluate performance of Support Vector Regressor
y_test_pred = sv_regressor.predict(X_test_norm)
mse = mean_squared_error(y_test, y_test_pred)
r2 = r2_score(y_test, y_test_pred)
print("\n#### Performance ####")
print("Mean squared error =", round(mse, 2))
print("R2 score =", round(r2, 2))
```

```
#### Performance ####
Mean squared error = 1430701216056.59
R2 score = 0.18
```