Entrega 2

Edwin Gutiérrez

Informe de avance para proyecto predictivo titulado: Home Credit Default Risk

Pueden ver la fuente original de la competencia en este hipervínculo, o con el siguiente enlace del kaggle: https://www.kaggle.com/competitions/home-credit-default-risk/data

El video puede verse en este hipervínculo, o en este enlace: https://drive.google.com/file/d/1uAxRYijLftpXXkRINXA1inwbSJ5MDqYy/view?usp=sharing

Sinopsis:

En el informe anterior (entrega 1), se expuso que el problema predictivo a resolver es el de determinar si una persona es apta para un crédito hipotecario, o no, según varias características, como lo son su nivel educativo, sus bienes, su estado civil, etc.

Este problema de predicción es útil para la entidad bancaria, ya que, con base a los resultados, puede disminuir su riesgo al otorgar créditos para la compra de vivienda, manteniendo una buena salud financiera, debido a que los clientes avalados con este método tendrán la manera de pagar las cuotas.

Dado que se necesita predecir dos condiciones (si es apto, o no), se puede decir que es un problema clasificatorio o de clasificación binaria. Así que es posible usar un Support Vector Machine, SVM, visto en los videos de clase, o una regresión clasificatoria.

Esta última no se consideró tan apta, ya que, según lo visto en el contenido del curso, es mucho más efectiva cuando la frontera de decisión entre tus dos clases es lineal, es decir, una línea recta, plano, o hiperplano puede separar tus clases. Como existe una cantidad considerable de características, es probable que no sea tan fácil separar a los individuos en pocas clases. Aun así, se realizaron algunas pruebas con este método.

Adicionalmente, se evaluó la posibilidad de usar un árbol de decisión, ya que la cantidad de variables de los individuos podía permitir que se tomaran varios elementos como decisivos para otorgar (o no) el crédito hipotecario.

Para esta ocasión, se exponen los resultados de las iteraciones realizadas con la regresión clasificatoria, y el Support Vector Machine.

Regresión clasificatoria:

En este proceso, se usó una matriz modificada de la original, ya que esta es muy grande y contiene demasiados datos. Se usó en su lugar una muestra de 1000 filas para entrenamiento, 5000 filas para predicciones, y las columnas más representativas del dataset original, las cuales son:

“CAR_OWN_AGE', = posee vehículo

'CNT_CHILDREN' = cantidad de hijos

'AMT_INCOME_TOTAL' = ingresos totales

'DAYS_EMPLOYED' = días como trabajador

'CNT_FAM_MEMBERS' = cantidad de miembros de la familia

'BASEMENTAREA_AVG' = área del inmueble

'YEARS_BUILD_AVG' = años de construido el inmueble

'COMMONAREA_AVG' = cantidad de áreas comunes

'FLOORSMAX_AVG' = Cantidad de pisos

'CODE_GENDER' = género masculine o femenino

'FLAG_OWN_CAR' = Si es propietario de un vehículo

'FLAG_OWN_REALTY' = Si posee bienes

'NAME_INCOME_TYPE' = Tipo de ingresos o labor

'NAME_EDUCATION_TYPE' = nivel educativo

'NAME_FAMILY_STATUS' = estado civil

'OCCUPATION_TYPE'] = cargo de trabajo

De aquí tenemos que se tuvo que separar la información en variables categóricas y no categóricas, quedando así:

No categóricas: CAR_OWN_AGE', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'DAYS_EMPLOYED',

'CNT_FAM_MEMBERS', 'BASEMENTAREA_AVG', 'YEARS_BUILD_AVG',

'COMMONAREA_AVG', 'FLOORSMAX_AVG']

Categóricas : ['CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'NAME_INCOME_TYPE',

'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'OCCUPATION_TYPE']

Acto seguido, lo que se realizó fue la preparación y depuración del dataset. Como se explicó en los videos del curso, la información no siempre viene tan limpia y bonita como quisiéramos. Por ello, se tuvo que eliminar las columnas que no aportaban ningún tipo de dato relevante, como

También se tuvo que rellenar algunos valores que estaban en blanco, como por ejemplo, de la columna sobre la información de si el individuo posee (o no) un vehículo propio. Para quienes no son propietarios, este valor estaba vacío, y generaba conflictos a la hora de iterar y hacer predicciones.

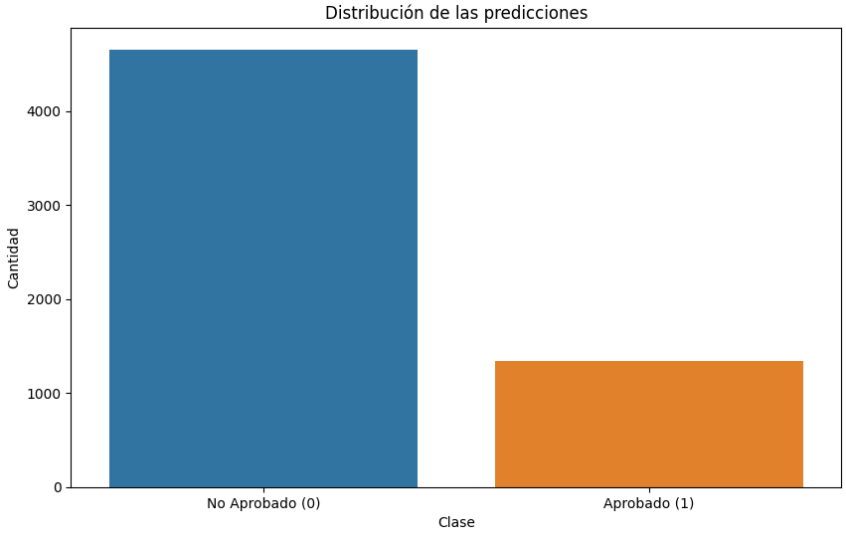
A continuación, se muestra en el código cómo se realizó la carga de datos y las columnas de información:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Función para preprocesar los datos
def preprocess_data(data):
    # Convertir DAYS_EMPLOYED a positivo
    data['DAYS_EMPLOYED'] = data['DAYS_EMPLOYED'].abs()

    # Asignar 0 a CAR_OWN_AGE donde FLAG_OWN_CAR es 'N'
    data.loc[data['FLAG_OWN_CAR'] == 'N', 'CAR_OWN_AGE'] = 0

    # Rellenar los valores faltantes
    num_columns = ['CAR_OWN_AGE', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'DAYS_EMPLOYED',
                   'CNT_FAM_MEMBERS', 'BASEMENTAREA_AVG', 'YEARS_BUILD_AVG',
                   'COMMONAREA_AVG', 'FLOORSMAX_AVG']
    cat_columns = ['CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'NAME_INCOME_TYPE',
                   'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'OCCUPATION_TYPE']

    for col in num_columns:
        data[col].fillna(0, inplace=True)

    for col in cat_columns:
        data[col].fillna('No aplica', inplace=True)
```

Este código puede verse con más detalle en el notebook Colab, o en el siguiente enlace:

https://colab.research.google.com/drive/1SG1CFgbgumSfLmgL4PUw0wDHikg9S9R_?usp=share_link

Las predicciones del modelo determinaron un 28% de individuos aprobados, del total de 5000 muestras (o filas). Pero se debe analizar con detenimiento el accuracy o precisión.



SUPPORT VECTOR MACHINE:

Se hizo también este procedimiento, para determinar si este resultaba mejor para obtener resultados. Se le dieron las mismas columnas, es decir, mismos datos de información. Así se cargaron los datos:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
from sklearn.impute import SimpleImputer

# Cargar datasets
train_data = pd.read_csv('datasetv2train.csv')
test_data = pd.read_csv('datasetv2.csv')  # Este dataset no tiene la columna 'target'

# Separar características y variable objetivo
X_train = train_data.drop('target', axis=1)
y_train = train_data['target']

# Listas de columnas numéricas y categóricas
num_columns = ['OWN_CAR_AGE', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'DAYS_EMPLOYED',
               'CNT_FAM_MEMBERS', 'BASEMENTAREA_AVG', 'YEARS_BUILD_AVG',
               'COMMONAREA_AVG', 'FLOORSMAX_AVG']
cat_columns = ['CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'NAME_INCOME_TYPE',
               'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'OCCUPATION_TYPE']

# Preprocesamiento
num_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])
```

Este código puede verse mejor en [este hipervínculo de Colab](), o en el siguiente enlace:

https://colab.research.google.com/drive/1FF1lCxeMyrDnbYJgL2t4w8iPNGvwWQM0?usp=sharing

El resultado fue que llenó la columna target de acuerdo a lo solicitado (aporbado o no aprobado, como uno y cero, respectivamente). Sedeberá analizar el accuracy o precisión d ellos resultados.

BASEMENTAREA_AVG	YEARS_BUILD_AVG	COMMONAREA_AVG	FLOORSMAX_AVG	target
0.0369	0.6192	0.0143	0.0833	0
0.0529	0.796	0.0605	0.2917	1
				0
				0
				0
				0
				0
				1
				0