Quantum HackMx

¡Mejora, transforma e implementa utilizando tu creatividad y el aprendizaje automático para la detección de fraudes bancarios!

Reto Propuesto por NDS Congitive Labs

Equipo CyberBots
ITESM CEM IRS-ITC, 2do Semestre

Cesar Emiliano Palomé Luna
Paulo Ogando Gulias
José Luis Madrigal Sánchez
Ana Patricia Islas Mainou

NDS COGNITIVE LABS

Introducción

Fraudes Bancarios

Los fraudes bancarios han aumentado recientemente debido al aumento de las transacciones; por lo que es imperativo generar algoritmos capaces de detectarlos de manera oportuna para evitar pérdidas monetarias.

Además, es responsabilidad conjunta de las empresas y de los estudiantes el velar por el bienestar de los clientes y los ciudadanos, por lo que realizar este tipo de proyectos es indispensable.

Redes Neuronales para Deep Learning y Data Science

Se decidió programar redes neuronales por su gran capacidad de adaptación durante el entrenamiento y por ser un método con gran tolerancia a los errores.

NDS COGNITIVE LABS

Algoritmo

Previo a la programación, ser realizó un algoritmo con los puntos más importantes que debía contener el programa.

Se utilizó base de datos **Kaggle Credit Card Fraud Detection**, que contiene un total de **284,807** registros dónde **284,315** son transacciones legales y **492** son fraudes bancarios.

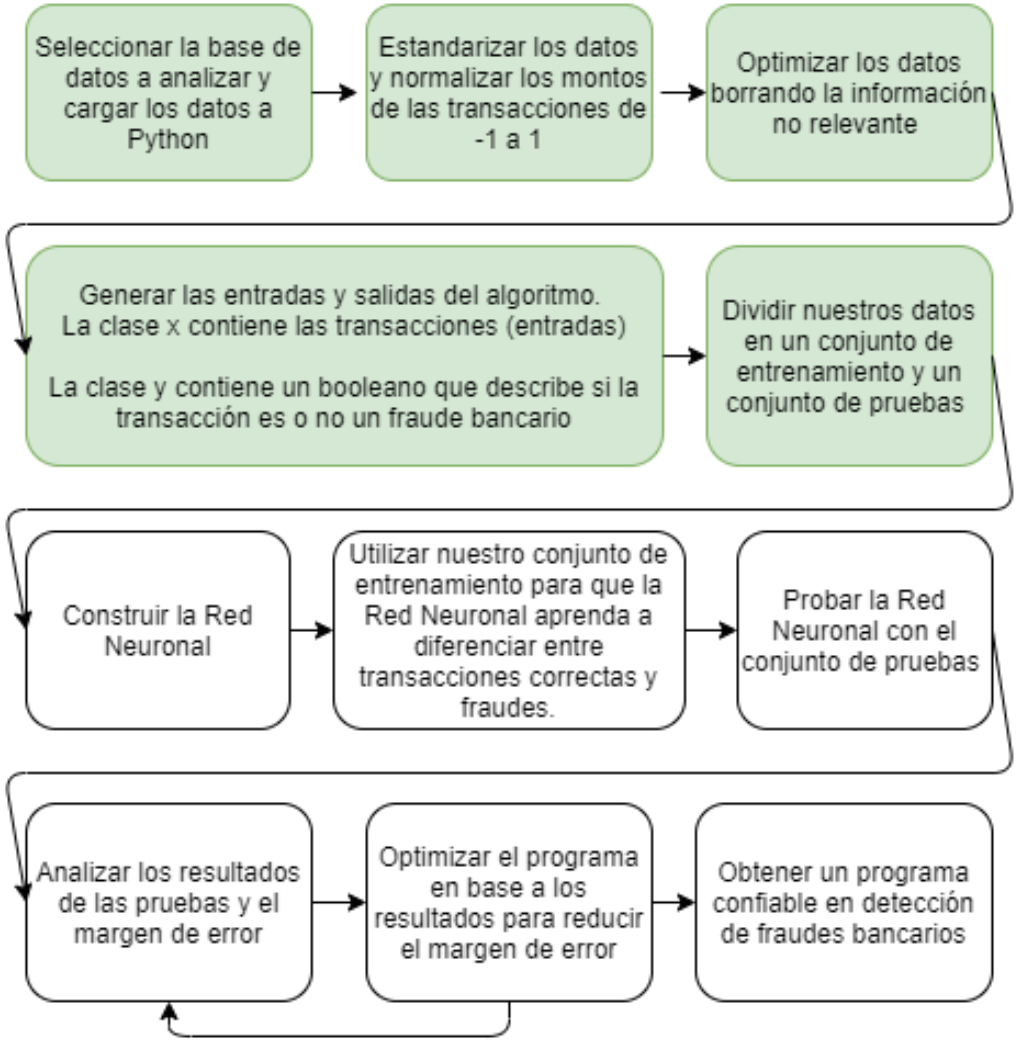
Librerías de Python utilizadas:

Pandas

Sklearn

Tensorflow

Keras



NDS COGNITIVE LABS

Manejo de los datos

Análisis por PCA, Normalización y Optimización de los Datos

```
# Standardizzacion de los datos del monto
scaler = preprocessing.StandardScaler()
# Normalizacion de los montos de -1 a 1
datos["NORMALIZADO"] = scaler.fit_transform(datos["Amount"\
                                                ].values.reshape(-1, 1))

# Borar las columnoas de tiempo y monto no normalizado
datos = datos.drop(["Amount", "Time"], axis = 1)
```

1. **PCA:** se seleccionan los datos relevantes para el análisis, se apoya en eigenvalores, eigenvectores, la covarianza y otras medidas estadísticas.
2. **Normalización:** se escalan los datos del monto de -1 a 1, manteniendo la relación entre ellos para evitar trabajar con números muy grandes.
3. **Optimización:** se eliminan aquellas columnas con datos que no se utilizan en el análisis.

Conjuntos de entrenamiento y pruebas para Machine Learning

Clase X: contiene la información de las transacciones y el monto normalizado.

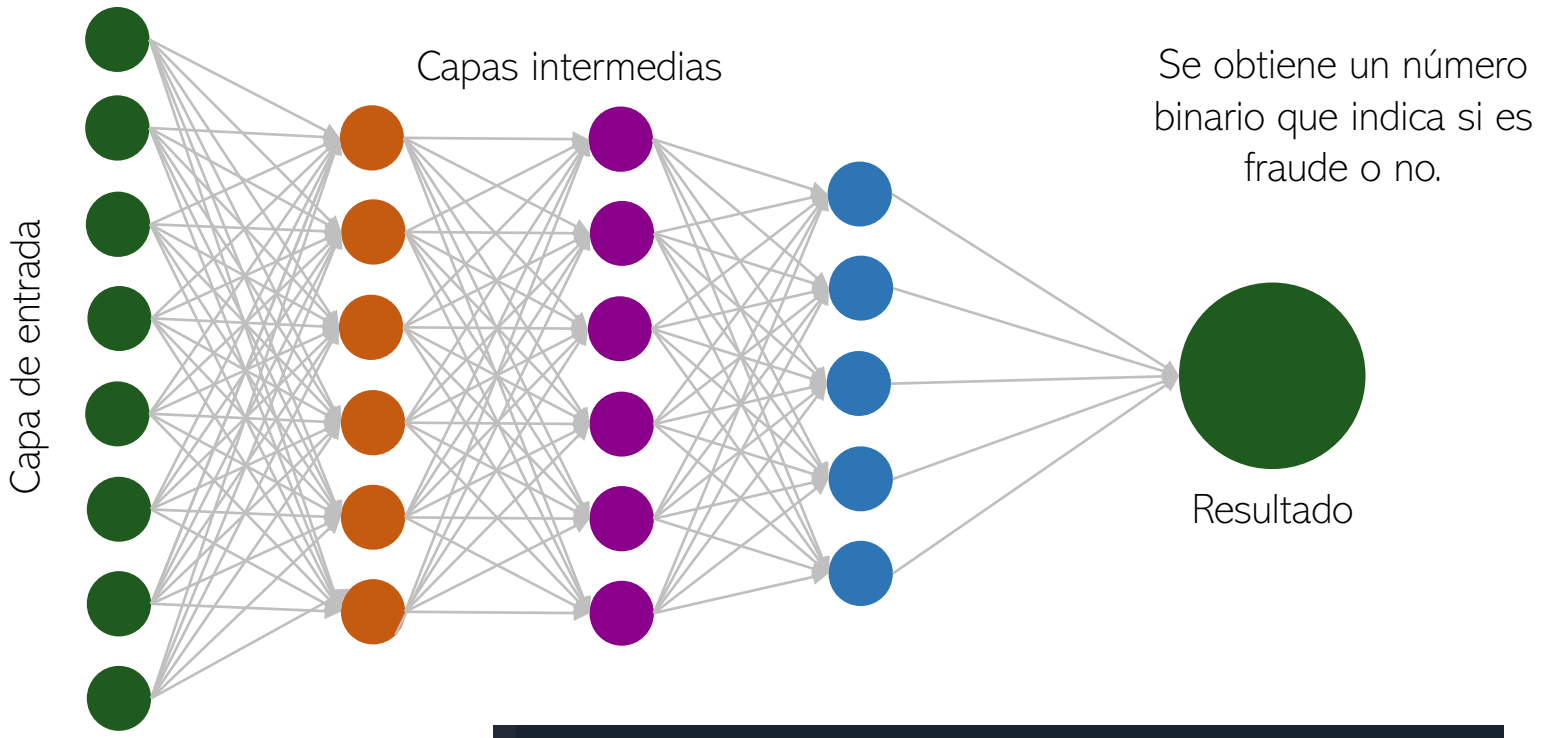
Clase Y: contiene un booleano que dice si es o no fraude bancario.

70% Entrenamiento

30% Pruebas

NDS COGNITIVE LABS

Red Neuronal Artificial




```
# Agreamos las capas de neuronas
modelo.add(layers.Dense(29, activation="relu", name = "capa1"))
modelo.add(layers.Dense(16, activation="relu", name = "capa2"))
modelo.add(layers.Dense(8, activation="relu", name = "capa3"))
modelo.add(layers.Dense(1, activation="sigmoid", name = "capa4"))
```

Cuatro Capas de Neuronas

Tamaño de Activación	Tamaño de la primera capa de neuronas, es el mismo que los datos de entrada.
Capa 1	Función de activación: relu. Tiene 29 neuronas.
Capa 2	Función de activación: relu. Tiene 16 neuronas.
Capa 3	Función de activación: relu. Tiene 8 neuronas.
Capa 4	Función de activación: sigmoid, nos da valores de 0 o 1. Tiene 1 neurona.

Funciones de Activación

- Activación Relu
- Activación Sigmoid

NDS COGNITIVE LABS

Construcción de la Red

Se utiliza el método de Adams para compilar la Red, este permite disminuir la gradiente cuando se entrena la neurona para aproximar mejor los resultados deseados.

```
Model: "sequential_68"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
capa1 (Dense)                multiple                  870
_________________________________________________________________
capa2 (Dense)                multiple                  480
_________________________________________________________________
capa3 (Dense)                multiple                  136
_________________________________________________________________
capa4 (Dense)                multiple                  9
=================================================================
Total params: 1,495
Trainable params: 1,495
Non-trainable params: 0
_________________________________________________________________
```

Entrenamiento de la Red

Se entrena la neurona con el 70% de los datos y se solicita que se realice el entrenamiento 5 veces para supervisar la disminución en el error y el aumento en la efectividad.

```
Epoch 1/5
60/60 [==============================] - 16s 266ms/step - loss: 0.5703 - acc: 0.7202
Epoch 2/5
60/60 [==============================] - 5s 90ms/step - loss: 0.1256 - acc: 0.9983
Epoch 3/5
60/60 [==============================] - 5s 77ms/step - loss: 0.0206 - acc: 0.9983
Epoch 4/5
60/60 [==============================] - 5s 76ms/step - loss: 0.0067 - acc: 0.9989
Epoch 5/5
60/60 [==============================] - 5s 82ms/step - loss: 0.0048 - acc: 0.9992
```

Tiempo | Error | Efectividad

```
modelo.fit(X_train, Y_train, steps_per_epoch= 20, batch_size = 15, \
           epochs = 5, verbose=1)
```

- **Steps per epoch:** Iteraciones por etapa de entrenamiento.
- **Batch size:** Número de datos analizados en cada paso.
- **Epochs:** Etapa de entrenamiento actual.

NDS COGNITIVE LABS

Conclusiones

A La red neuronal programada tiene un porcentaje de efectividad muy alto y un índice de error muy bajo, por lo que es un método de análisis de datos confiable.

B Al analizar datos sensibles, como información bancaria, es imperativo tener certeza de que el algoritmo funciona bien para maximizar la detección de transacciones fraudulentas.

Resultados de la Red Neuronal Artificial

```
PROBAR LA RED NEURONAL----------------------------------------
1/1 [==============================] - 28s 28s/step - loss: 0.0045 - acc: 0.9993
El error es de: 0.004460305441170931
La efectividad es de: 99.93%
```

Agradecimiento Especial al Profesor Antonio Víctor Mejía Olvera y al Profesor Luis Manuel Díaz García

NDS COGNITIVE LABS