

# FRAUD DETECT

ALEJANDRO CASTILLO  
PABLO OLLER  
PABLO SANTOS



# Índice

## Tabla de Contenidos

- 01 Introducción**
- 02 Obtención y Limpieza**
- 03 Exploración y Visualización**
- 04 Modelo de Predicción**
- 05 Modelo de Clusterización**
- 06 Procesamiento del Lenguaje Natural**
- 07 Aplicación Web**
- 08 Conclusiones**



# Introducción

## Descripción de FraudDetect

Fraud-Detect es una web-app diseñada para abordar de forma eficiente y precisa la detección del fraude bancario mediante la aplicación del uso de modelos de IA.

La aplicación genera un detallado análisis que identifica posibles irregularidades financieras como consecuencia de comportamientos fraudulentos entre los clientes.



# Tecnologías



TensorFlow

K Keras



Streamlit



matplotlib



# Obtención de Datos Dataset Kaggle

Para la obtención de datos hemos trabajado con un Dataset muy utilizado para el entrenamiento de modelos orientados al fraude bancario.

Nosotros lo hemos encontrado en la comunidad de científicos de datos Kaggle.

El enlace a este Dataset lo podréis encontrar en la Bibliografía del README.



# Exploración y visualización de datos

En primer lugar se ha cogido una muestra de la información del dataset, a continuación equilibraremos el conjunto de datos para representar y distribuir las clases equitativamente. En caso contrario sesgaría el rendimiento del modelo ya que hay más casos de no fraude que de los que sí.



# Modelo de Predicción

## Preparación de los datos

En nuestro caso tenemos que gestionar valores nulos además de aplicar una transformación de una de las columnas a tipo float.

También hay que hacer un balanceo de datos y un estudio exhaustivo de valores atípicos.

```
1 # Separación del Dataset  
2 X = ccdf.drop('Class', axis=1)  
3 y = ccdf['Class']  
  
1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)  
  
1 X_train.shape, X_test.shape  
((173954, 30), (43489, 30))
```

# Modelo de Predicción

## Creación del Modelo

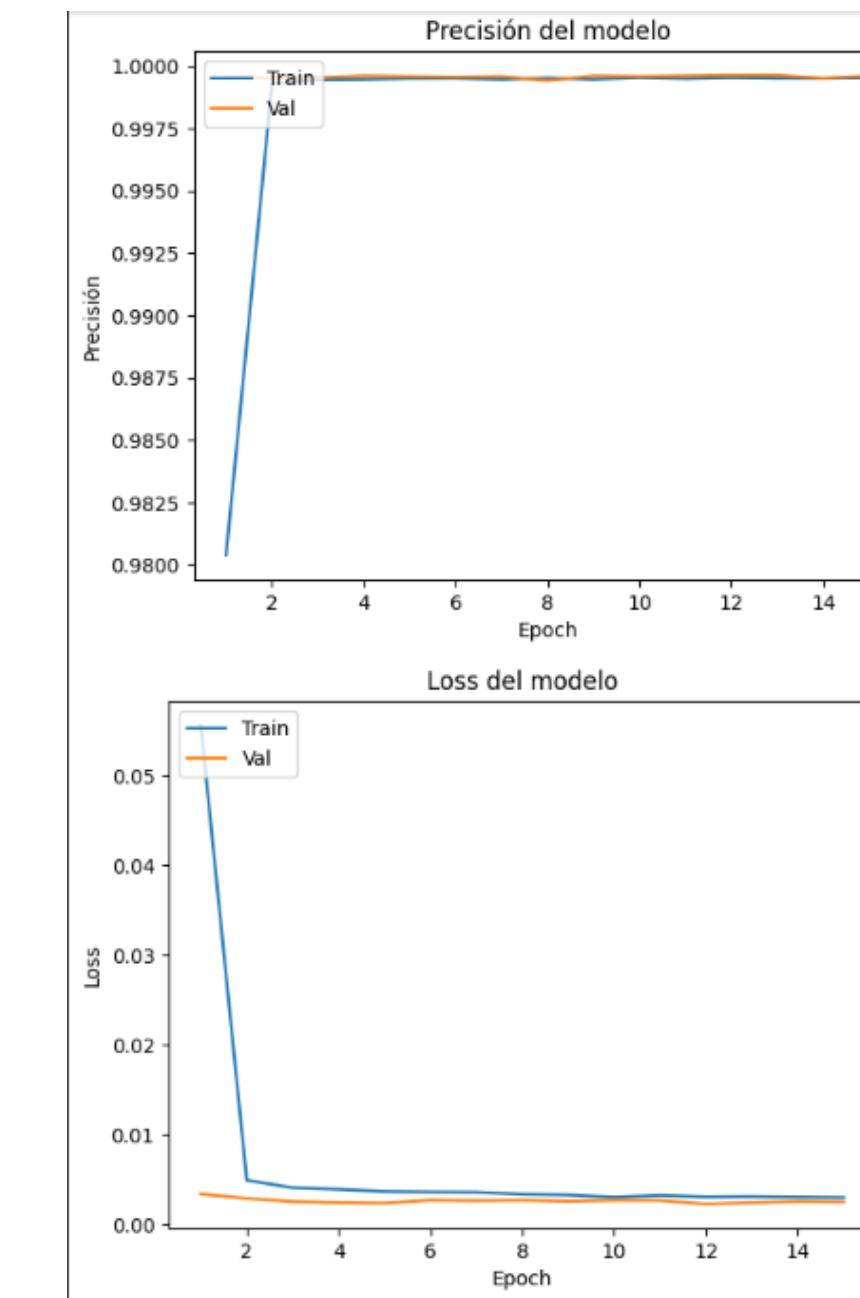
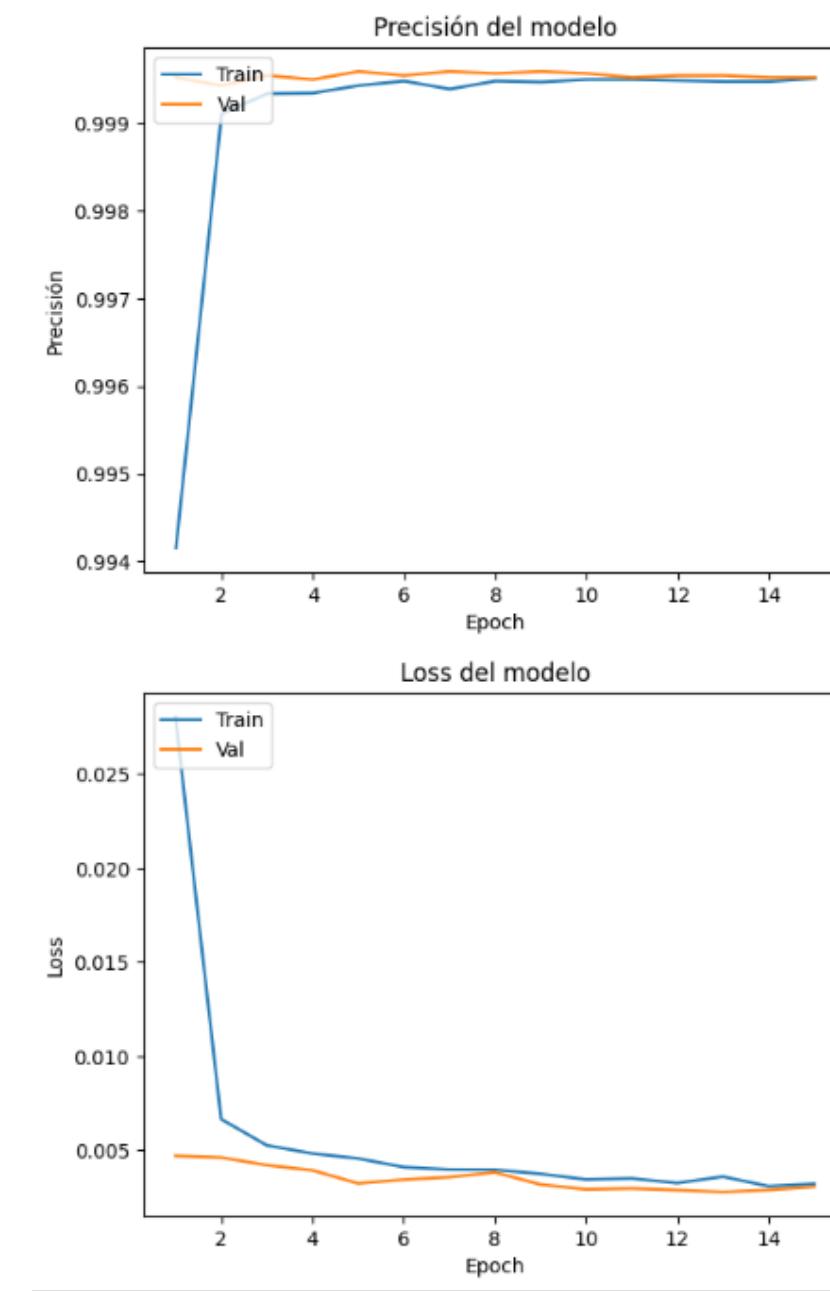
Creación de nuestra red neuronal convolucional utilizando la biblioteca Keras.

```
1 epochs = 15
2 model = Sequential()
3 model.add(Conv1D(filters=32, kernel_size=2, activation='relu', input_shape=X_train[0].shape))
4 model.add(BatchNormalization())
5 model.add(Dropout(0.2))
6
7 model.add(Conv1D(filters=64, kernel_size=2, activation='relu'))
8 model.add(BatchNormalization())
9 model.add(Dropout(0.5))
10
11 model.add(Flatten())
12 model.add(Dense(units=64, activation='relu'))
13 model.add(Dropout(0.5))
14
15 model.add(Dense(units=1, activation='sigmoid'))
```

# Modelo de Predicción

## Métricas de Rendimiento

Comparación del rendimiento con y sin capa MaxPool.



# Modelo de Clusterización

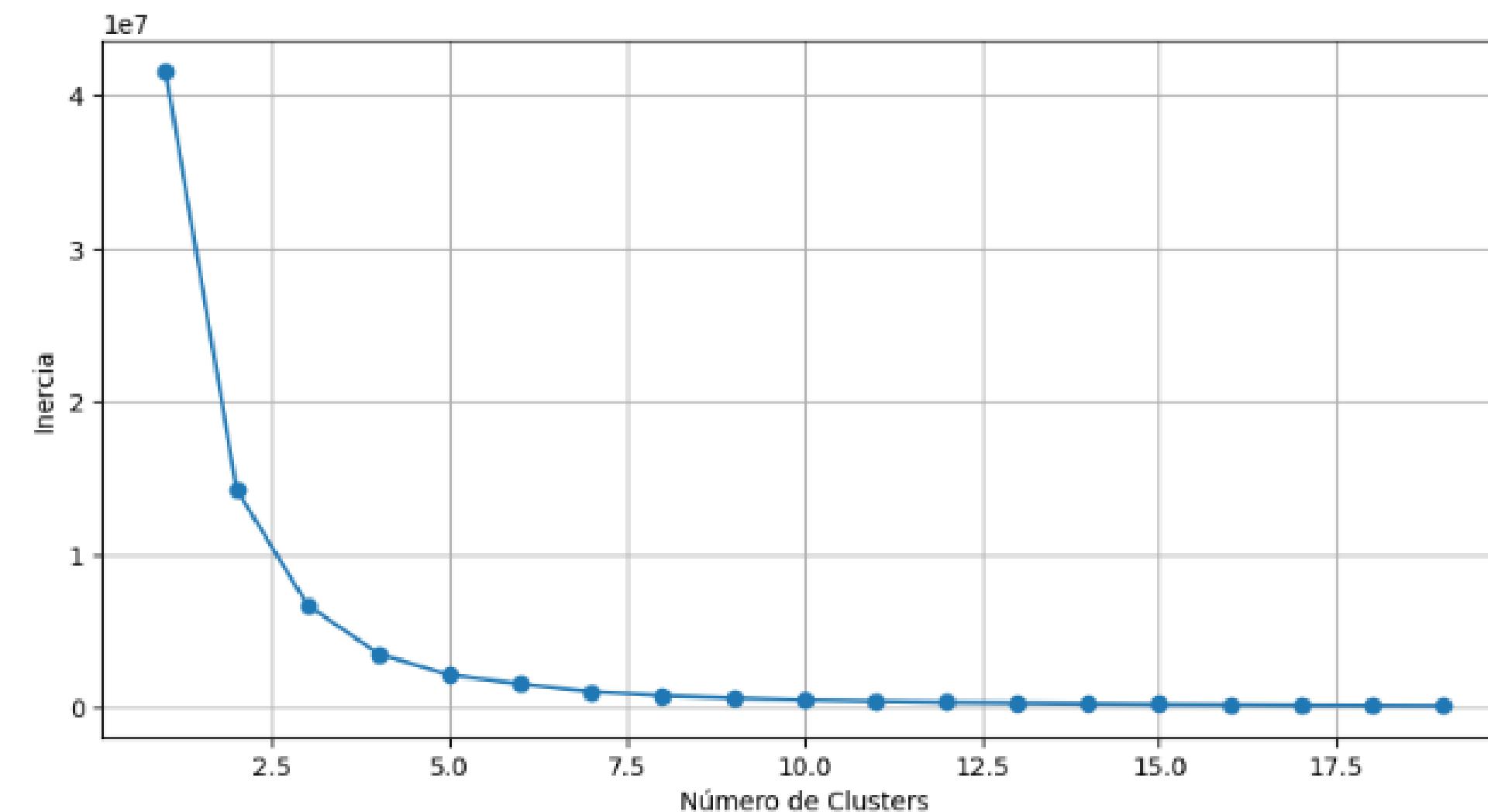
## Preparación de los datos

Aprovechamos el tratamiento anterior y creamos el nuevo atributo 'Median'.

```
ccdf['Median'] = ccdf[['V1", "V2", "V3", "V4", "V5", "V6", "V7", "V8", "V9", "V10",
    "V11", "V12", "V13", "V14", "V15", "V16", "V17", "V18", "V19",
    "V20", "V21", "V22", "V23", "V24", "V25", "V26", "V27", "V28"]].mean(axis=1)
```

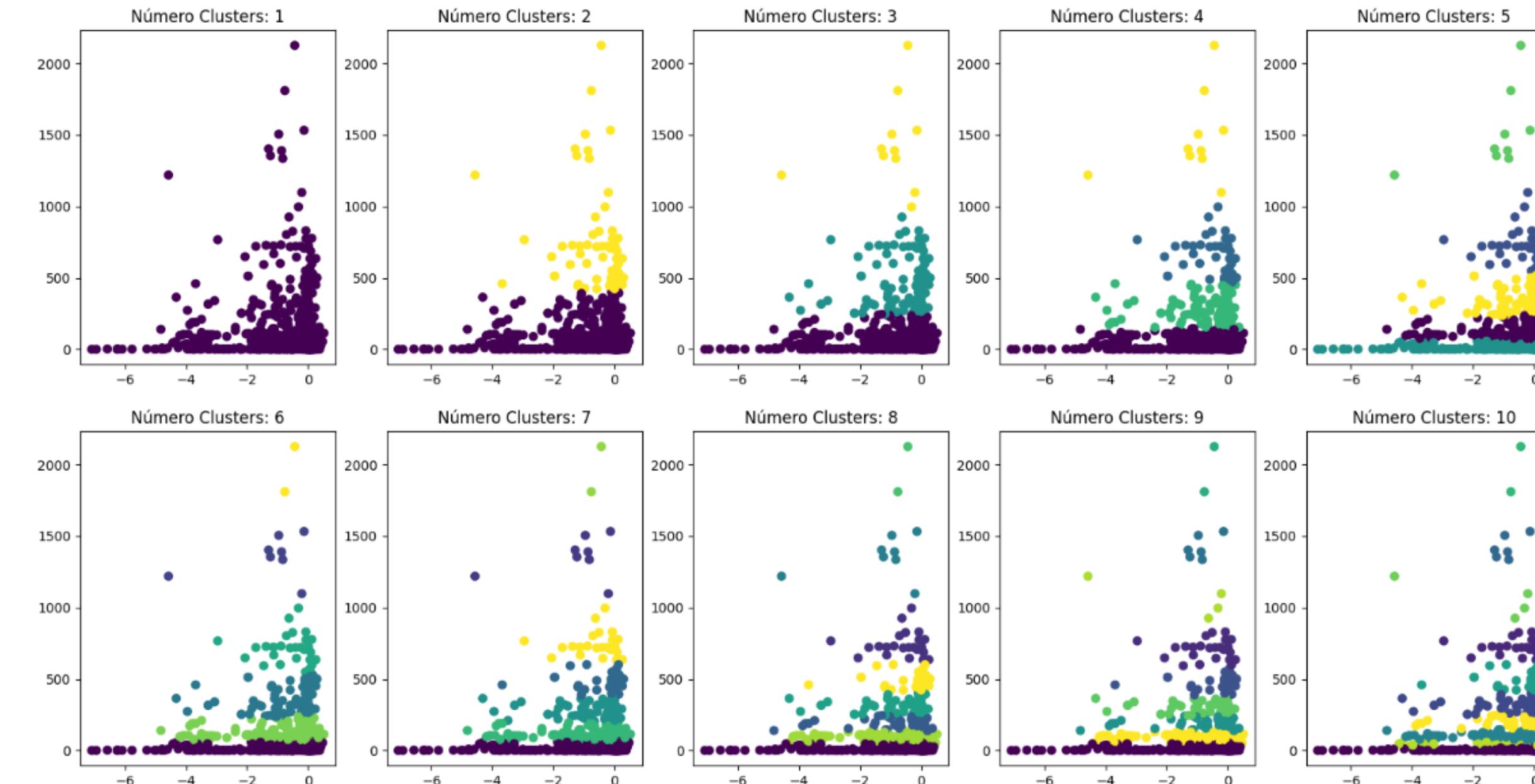
# Modelo de Clusterización Agrupación idónea

Detección del número idóneo de clusters mediante el método Elbow.



# Modelo de Clusterización Agrupación idónea

Aplicación del algoritmo 10 veces y visualización de las diferentes agrupaciones.



# Procesamiento del lenguaje natural



## ¿Qué hemos elegido?

Para abarcar el tema del PLN hemos elegido crear un chatbot llamado Eto'o bot que permitirá al usuario resolver cuestiones sobre los resultados mostrados en la aplicación.

## ¿Por qué?

Es una herramienta muy adaptable que puede resolver problemas de fácil manera, respondiendo directamente a la pregunta del usuario sin tener que abandonar la página para resolver sus dudas.

# Aplicación Web **Fraud-Detect**



# Conclusiones

Con este proyecto hemos querido mostrar las posibilidades que ofrecen las tecnologías relacionadas con la IA.

el superar la dificultad de los problemas que iban apareciendo nos hacia ver todo lo que habíamos aprendido a lo largo del Master.

En particular, queremos destacar la contribución esencial de Pablo Santos, cuyo compromiso y dedicación fueron fundamentales en el desarrollo del proyecto realizando el mayor esfuerzo.

Agradecemos al profesorado por su implicación en la materia, por las explicaciones en clase, tareas y trabajos en grupo que han hecho que lleguemos a este TFM con la mejor preparación posible.



CPIFP Alan Turing



Accenture Málaga