Universidad Nacional de General Sarmiento

LABORATORIO DE CONSTRUCCIÓN DE SOFTWARE

Informe Progreso de Detección de Anomalías en Asistencia (Opción 4)

Alumnos:

- Tomás Altamirano
- Gonzalo Petosa

Elegimos trabajar con la opción número 4 debido a que nos pareció más adaptable a nuestra forma de trabajar con **Isolation Forest**.

Insolation Forest

El **Insolation Forest** es un modelo de machine learning diseñado específicamente para la detección de anomalías, es un modelo no supervisado que no requiere etiquetas en los datos para poder entrenarse. A diferencia de otros modelos que buscan clasificar o predecir una etiqueta, el objetivo de Isolation Forest es identificar puntos de datos que son anómalos o diferentes del resto

¿Cómo se utiliza este modelo?

- **Entrenamiento**: El modelo de **Isolation Forest** se entrena con un conjunto de datos que contiene tanto puntos de datos normales como posibles outliers (sin necesidad de etiquetas). El modelo construye un conjunto de árboles de decisiones binarias (de ahí el término "**Forest**") y utiliza particiones aleatorias de los datos para aislar las observaciones.

- **Predicción**: Después de entrenar el modelo, se puede utilizar para hacer predicciones sobre nuevos datos, y determinar si un punto de datos es "normal" o un "**outlier**". Los puntos que son más fáciles de aislar (requieren menos divisiones) se consideran **outliers**, mientras que los que son más

difíciles de aislar (requieren más divisiones) se consideran normales.

Características clave del modelo:

1. **Modelo no supervisado**: No necesita etiquetas para entrenar. Solo requiere los datos.

2. **Basado en árboles de decisión**: Utiliza múltiples árboles de decisión (un bosque) para aislar los puntos de datos.

3. **Eficiencia**: El modelo es eficiente y funciona bien en grandes volúmenes de datos.

4. **Robusto frente a grandes datasets**: Isolation Forest es especialmente útil cuando se trabaja con grandes conjuntos de datos con muchas características

LUEGO DE DESCARGAR EL CÓDIGO GIT

Antes de ejecutar el código, se debe de instalar mediante consola (CMD) las siguientes librerías de la siguiente manera:

```
pip install numpy pandas matplotlib scikit-learn
```

Si está instalado, podemos comprobarlo al poner por cada librería:

```
pip show numpy
```

```
pip show pandas
```

```
pip show matplotlib
```

```
pip show scikit-learn
```

De esta manera comprobamos si las librerías están instaladas en el equipo, mostrándonos información de la librería como su versión y otras características. En caso de que el programa en donde estamos escribiendo nuestro código estuviera

abierto durante la instalación, hay que cerrarlo y volverlo a abrir para poder utilizar las librerías anteriormente nombradas.

Dataset Ficticio

Nuestro dataset consta de 100 empleados, para los cuales se crea una lista de 30 lugares a modo de representar la asistencia de los mismos durante un mes, para esto se elige al azar un número que puede ser 0 o 1 siendo 0 si no asiste y 1 si asiste, nótese que hay un 10% de que sea 0 y un 90% de 1.

A modo de tener un control más fácil, se insertan 3 empleados con al menos 3 inasistencias aseguradas lo cual permite tener un control de los resultados de modelos.

```
# Simulación de datos de 100 empleados y 30 días de registros
empleados = [f"Empleado_{i}" for i in range(1, 101)]  # Creación de los 100 empleados
```

```
for empleado in empleados:
    asistencias = np.random.choice([0, 1], size=30, p=[0.1, 0.9])  # Genera un arreglo de tamaño 30 con 0 o 1 simulando la asistencia
    # Introducir algunas anomalías (ausencias seguidas)
    if empleado == 'Empleado_1':
        asistencias[5:8] = [0, 0, 0]  # 3 ausencias seguidas (anomalía)
    if empleado == 'Empleado_2':
        asistencias[15:18] = [0, 0, 0]  # 3 ausencias seguidas (anomalía)
    if empleado == 'Empleado_3':
        asistencias[25:28] = [0, 0, 0]  # 3 ausencias seguidas (anomalía)
    asistencia_data.append(asistencias)
```

Primero utilizamos una semilla de aleatoriedad para obtener un mismo resultado cada vez que testeábamos, luego comenzamos a simular los datos de los 100 empleados más los 30 días que estos deberían asistir. Lo siguiente que hicimos fué crear un **DataFrame** el cual es una tabla con los días con el valor de asistencia por empleado. Seguido de esto, usamos una expresión lambda para transformar los **0** en asistencias y los **1** en inasistencias ya que facilita el cálculo de la racha de ausencia.

Entrenamiento y Predicción

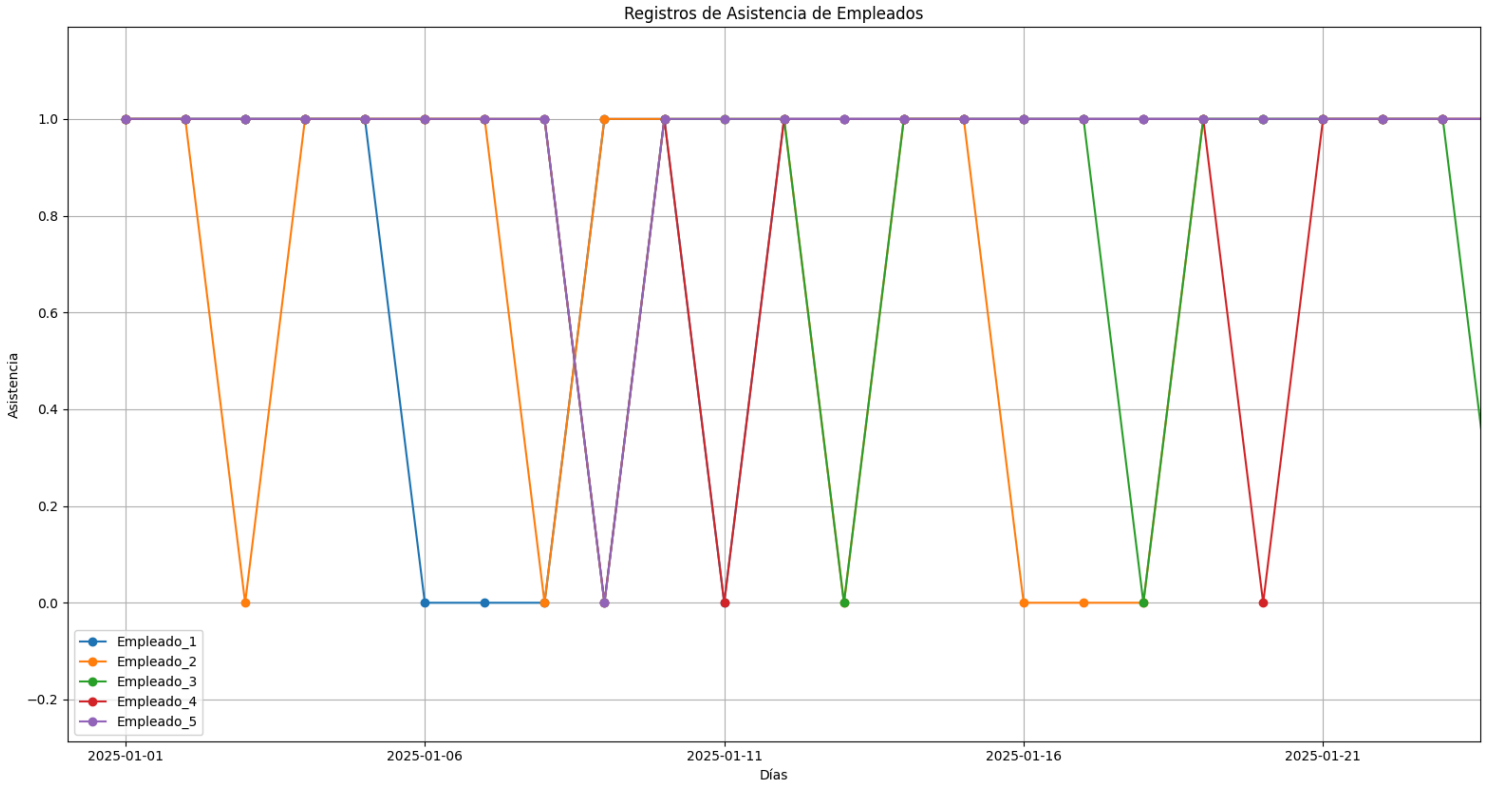
```python
# Aplicar Isolation Forest para detectar anomalías
model = IsolationForest(contamination=0.1, random_state=42)  # El valor de contaminacion esta en 10% dado que esperamos ese porcentaje de ausentismo.
model.fit(df_bin)

# Predecir anomalías
anomalies = model.predict(df_bin)
```

Entrenamiento (fit): El modelo aprende patrones normales de asistencia en tu dataset df_bin, que representa las ausencias (0 = asistió, 1 = se ausentó).
Predicción (predict): El modelo predice si cada fila del DataFrame representa un comportamiento anómalo (-1) o normal (1).

Luego aplicamos el **Insolation Forest** para comenzar a detectar las anomalías, dejando una contaminación en 0.1 ya que el porcentaje de ausentismo de un empleado ronda el 10%, y luego las predijimos en donde **1** es para normal y **-1** es para anomalía y por consiguiente creamos un DataFrame con estas predicciones, mostramos los resultados de estas para luego graficar sus resultados, que primero vamos a graficar la asistencia de los primeros 5 empleados para ver cómo se distribuyen, seleccionamos los empleados para la visualización y finalmente graficamos las asistencias de estos empleados, en dónde se nos muestra las anomalías que se registraron.

PARTE GRÁFICA



En el gráfico adjunto se visualiza un gráfico con un eje X representando los días del mes, y un eje Y representando si asistio (1) o no (0), entonces durante el transcurso de los días del mes, vemos que aquellos que no fueron a trabajar tienen punto coloreados en el valor 0 del eje Y, de esta forma vemos que si hay 3 punto en 0 seguidos estaríamos en presencia de una anomalía de ausentismo.

Nuestro código se encuentra en:

https://github.com/7oXaL/Plataforma-de-Gestion-de-Recursos-Humanos-con-ERP-y-CRM-con-Inteligencia-Artifical/blob/main/DataCet.py