UNIVERSIDAD DE GRANADA

Series Temporales y Minería de Flujos de Datos

Máster en Ciencia de Datos e Ingeniería de Computadores

Práctica: Minería de Flujos de Datos

1. Objetivo.....2
2. Materiales necesarios.....2
3. Descripción del problema.....2
4. Descripción de la tarea a realizar.....3
5. Entrega de la práctica y evaluación.....4

DECSAI

Departamento de Ciencias de la Computación e Inteligencia Artificial

Minería de Flujos de Datos

1. Objetivo

El objetivo de la práctica consiste en que el estudiante se familiarice con bibliotecas para trabajar con Flujos de Datos y resuelva problemas de la temática. **La tarea debe realizarse de forma individual.**

2. Materiales necesarios

Idealmente, se recomienda trabajar con entornos **Anaconda**. En particular, se requiere la instalación de las siguientes bibliotecas:

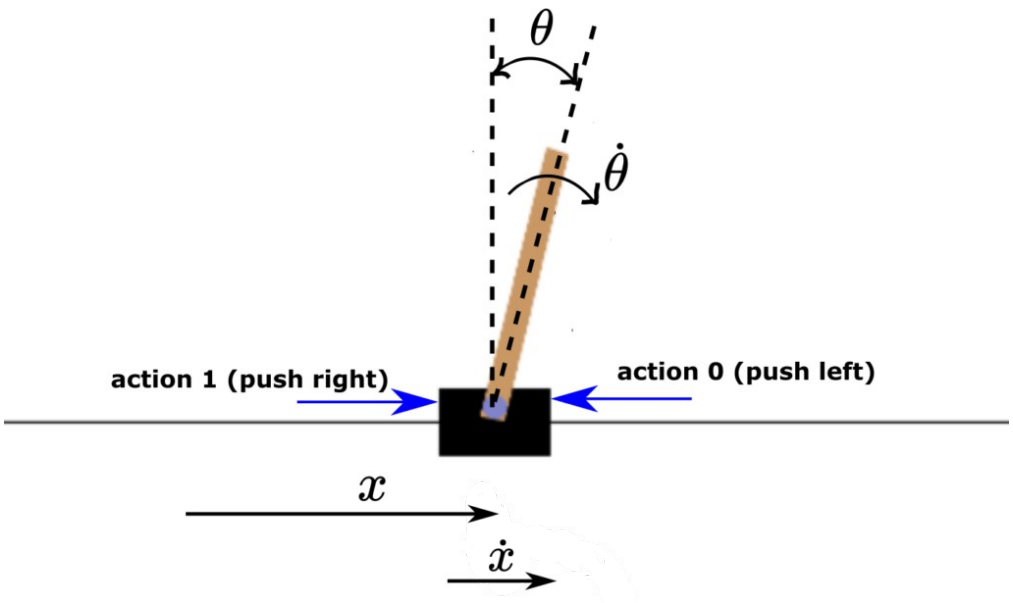
- **River** (Biblioteca para Minería de Flujos de Datos):
  - pip install river
- **Gymnasium** (Biblioteca de Reinforcement Learning usada para resolución de problemas de flujos de datos):
  - pip install gymnasium
  - pip install pygame
  - pip install gymnasium[classic-control]
  - **En caso de problemas de visualización (probado en Ubuntu):**
    - conda install -c conda-forge pyglet
    - conda install -c conda-forge libstdcxx-ng
- **IDE de programación Python** favorito (recomendación: Spyder).
- Fichero **CartPoleInstances.csv** proporcionado por el profesor en la plataforma docente: Contiene 100.000 instancias de un flujo de datos a resolver.
- Fichero **ShowCartPoleFlow.py**: Ejemplo de lectura del fichero CSV que contiene el flujo de datos.
- Fichero **PlayCartPole.py**: Ejemplo de interacción con la biblioteca Gymnasium y cálculo de la recompensa total obtanida (*performance*) de un modelo naive de ejemplo implementado a mano.
- Diapositivas en PDF de descripción de la práctica.

3. Descripción del problema

Se dispone de un sistema hardware simulado en 2D, formado por un carro y un poste en la parte central. En cada momento, el flujo de datos contiene información sobre:

- La posición del carro (Atributo **Cart Position**): Numérico (Valores >0 hacia la derecha, valores < 0 hacia la izquierda, valor=0 en el centro).
- La velocidad del carro (Atributo **Cart Velocity**): Numérico. Contiene la velocidad del carro (<0 hacia la izq ; >0 hacia la derecha ; 0= sin movimiento).
- El ángulo del poste con respecto a la vertical (Atributo **Pole Angle**): Numérico (<0 inclinado a la izq ; >0 inclinado a la derecha).
- La velocidad angular a la que se mueve el poste (Atributo **Pole Angular**

Velocity): Numérico. (<0 hacia la izq ; >0 hacia la dcha.).
- La pseudo-mejor acción posible a realizar (Atributo objetivo o clase, **Action**): Valor entero (0= mover a la izq ; 1= mover a la dcha.).



Por cada instante de tiempo que el poste permanezca en posición vertical (ángulo "cercano" a la vertical), se obtiene una recompensa de +1. La **recompensa total** obtenida es la suma de todos los instantes de tiempo que el poste permanece en vertical. El flujo finaliza una vez que el poste cae.

El **objetivo del problema** consiste en desarrollar un mecanismo de selección de acciones que permitan que el poste permanezca en vertical **de forma indefinida**. No obstante, para evitar flujos infinitos, asumiremos que el poste está indefinidamente en posición vertical tras **500 instantes de tiempo en esa posición**.

4. Descripción de la tarea a realizar

En el fichero **CartPoleInstances.csv** publicado por el profesor en la plataforma docente se proporciona un flujo de datos con 100.000 instancias de un experto en mantener el poste en posición vertical. El objetivo perseguido en la práctica es **elaborar un clasificador de Minería de Flujos de Datos capaz de aprender este comportamiento, y replicarlo posteriormente de forma autónoma**. Para ello, se pide:

- Comparar al menos 3 clasificadores (uno de ellos, usado con método de línea base -baseline-, deberá ser un modelo Naïve-Bayes con modelado Gaussiano de clases).
- Tratar de desarrollar el clasificador que mejor se comporte en base a dos métricas diferentes:
  - En entrenamiento: La mejor tasa de clasificación (selección de acción) posible (**métrica accuracy**).

    - En test: La mejor ***performance*** posible (Recompensa total obtenida).

El estudio es libre, permitiéndose aplicar cualquier técnica o conjunto de técnicas estudiadas en teoría o prácticas. A modo orientativo (no necesariamente obligatorio), se podrían estudiar aspectos como:

- ¿Presenta el flujo de datos algún tipo de Concept Drift? (justificando experimentalmente la propuesta).
- ¿Es un modelo adaptativo mejor que uno estacionario en este problema? (justificando la respuesta).
- ¿Técnicas como ADWIN o similares son relevantes en este problema?
- ¿Se necesita algún tipo de preprocesamiento para resolver el problema?

Requisitos:

- Los 3 modelos de clasificación entrenados deberán compararse estadísticamente en accuracy (training), de modo que se pueda conocer qué modelo es capaz de aproximar mejor al comportamiento del experto.
- Los 3 modelos de clasificación entrenados deberán compararse estadísticamente en test (**Recompensa Total obtenida**), de modo que se pueda conocer cuál es el mejor para resolver el problema planteado.
- ¿Se corresponde el resultado de ambos puntos anteriores? Justifique su respuesta.

5. Entrega de la práctica y evaluación

Se deberá entregar por PRADO, antes de la fecha límite indicada en la plataforma docente:

- Un **fichero PDF (memoria de prácticas)** conteniendo el análisis del flujo de datos realizado, así como una explicación de las soluciones propuestas y su análisis comparativo.
- **Un único fichero Python .py:**
    - El código fuente con la solución para entrenamiento y test de los modelos generados. Deberá poder ser ejecutable por el profesor.

La tarea se valorará de 0 a 10, y se corresponderá con 2.5 puntos de la evaluación final de la asignatura. Se valorará:

- Análisis previo del flujo de datos (2 puntos).
- Código fuente: Funcionamiento y legibilidad con comentarios (3 puntos).
- Análisis de resultados y selección del mejor modelo para las métricas:
    - Preprocesamiento requerido para los modelos (1 punto).
    - Análisis comparativo de modelos (3 puntos).
    - Selección y justificación del mejor modelo, y su aplicación en test (1 punto).