Inteligencia Artificial

Apuntes de clase 14/08/2025

Jose Pablo Quesada Rodríguez
Escuela Ingeniería en Computación
Cartago, Costa Rica
josepabloqr15@estudiantec.cr

***Abstract*—El presente documento pretende funcionar a manera de resumen detallado de la clase del día Jueves 14/08/2025, Se abarca un repaso de los temas vistos en la clase anterior a esta, así tambien como conceptos nuevos y explicación de herramientas utiles para el curso. Se busca ampliar la información mediante referencias bibliográficas y adjuntar el apartado de noticias al final del documento.**

I. Introducción

Se inicia la clase con una introducción a los tipos de aprendizaje, estos siendo un repaso rápido de estos ya que fueron vistos en la clase anterior

TABLE I
Tipos de Aprendizajel

Tipo	Descripción
Supervisado	Conjunto de datos que tienen una etiqueta, la cual supervisa el aprendizaje y determina que tan bien o mal se encuentra la descripcion.
No-supervisado	No requiere etiquetas, utiliza algoritmos de machine learning diseñados para descubrir patrones ocultos o agrupaciones de datos sin la necesidad de intervención humana
Semi-supervisado	Usa pocos datos etiquetados junto a muchos no etiquetados.
Auto-supervisado	Genera etiquetas a partir de los propios datos de entrada.(sus dataset/samples funcionan como sus propias etiquetas)
Refuerzo	Aprende con recompensas o castigos según sus acciones.
Few-shot	Generaliza a partir de pocos ejemplos.
One-shot	Aprende con un único ejemplo.
Zero-shot	Realiza tareas sin ejemplos previos, usando conocimiento previo.

II. Enfoques a Machine Learning

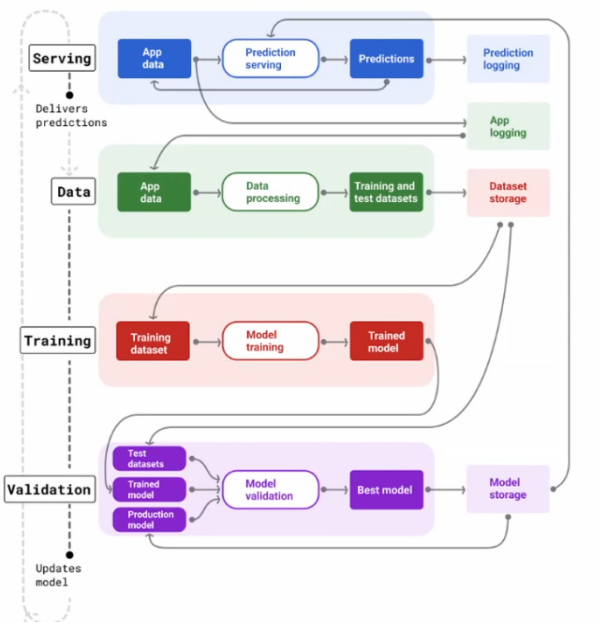
A. Ciencia

- **Generan conocimiento:** Se centran en la investigación y en la producción de nuevo conocimiento y tecnicas especificas sobre como utilizar modelos para generar nuevo conocimiento o pulir el existente
- **Métricas:** Se enfocan en criterios cientificos para definir la métricas como por ejemplo la capacidad explicativa (qué tan bien un modelo ayuda a entender un fenómeno)
- **Data Scientist:** Orientado a extraer conocimiento de los datos mediante experimentación rigurosa, pero con una visión práctica. A menudo trabajan con datos reales y aplican técnicas existentes, pero también pueden ajustar o proponer mejoras metodológicas para que los resultados sean científicamente válidos.
- **Research Scientist:** más enfocado en la teoría y la innovación que en la aplicación práctica. Su objetivo principal es desarrollar nuevos modelos, marcos teóricos o enfoques de aprendizaje automático que expandan el campo.

B. Ingeniería

- **Puesta en Producción de un modelo:** Bajo lo seleccionado por el data scientist del modelo científico, toma el modelo dado y busca ponerlo en producción para usarios masivos
- **Transformar el modelo:** Busca la optimización de este
- **Onnx:** Se encarga de hacer transformaciones en modelos para optimizar los modelos.
- **MLOps:** Debe pensar como tomar un modelo masivo y disponibilizarlo para los usuarios

III. Pipeline IA



El pipeline de Machine learning indica como se debe crear, hacer y mantener una inteligencia artificial según [1] la linea de este pipeline es:

A. *Conseguir data*

En esta seccion se consiguen la data, se obtiene de diferentes formas, esta data va a alimentar el modelo para su entrenamiento, sin embargo esta puede contener información inutil para el entrenamiento del modelo o en formatos diferentes, por lo tanto se necesita limpiar la misma

B. *Limpiar la data(data preparation*

Se procesa la data, buscando eliminar los datos inutiles o rellenar elementos faltantes de los mismos, tambien convertir formatos, pueden llegar data de diferentes formas, SQL o noSql por ejemplo, todo con el objetivo de hacerlos funcionales como datos de entrenamiento, estandarizandolos.

C. *Feature Engineering*

Ayuda a definir las caracteristicas que vayan a definir las predicciones correctas, consiste en la selección correcta de que variables considerar como relevantes para obtener mejores resultados, el objetivo es proporcionar features más informativos y relevantes, elegir que caracteristicas conservar y que deshechar

D. *Selección del modelo*

Consiste en Saber identificar el problema y que modelo amerita su uso, una buena sintesis para definir esto es mediante la característica de explicabilidad, si se trabaja con problemas que van a necesitar una retroalimentación clara de donde viene la información, un modelo de red neuronal no será lo más indicado.

E. *Entrenar el modelo*

El modelo es entrenado con alguno de los metodos de aprendizaje vistos, teniendo en consideración parametros e hiperparámetros, siendo estos ultimos los que ayudarán a dirigir al modelo por el camino deseado, un ejemplo de esto es k-means, el cual es un algoritmo para realizar clustering en un modelo de aprendizaje no-supervisado

F. *Validar el modelo*

En esta parte ya se tiene el modelo entrenado y se busca el testeo del mismo y la validación, se prueba con situaciónes similares a las de entrenamiento, pero no exactamente iguales, con el objetivo de verificar que se comporte de la manera deseada Y al final se realiza una validación del modelo, como una especie de examen final y se compara con producción para verificar que el mismo

IV. Paradigma de Resolución de Problemas

A. *Problemas de búsqueda*

En estos problemas, los algoritmos tratan de seguir un camino para llegar a una solución optima, la solución optima se considera siempre la opción más ”barata”

B. *Problemas de optimización*

En estos problemas se tiene una función matemática con un punto minimo, al encontrar ese punto minimo se tiene la mejor solución, existe la Solución local la cual nos da la mejor solución en un área especifica de búsqueda (sin embargo no la mejor) y la solución Global que da la mejor solución En todo el espacio, (se busca encontrar esta solución)

C. *Predicción y clasificación*

1) ***Predicción****:* Cuando se quiere determinar el valor real de una función o predecir los valores de esta de acuerdo a las caracteristicas que tienen las muestras del data set que lo componen, un ejemplo dado por el profesor fue las partes de los vehiculos en la predicción de cuanto combustible consume, variables dependientes que dependen de variables independientes

2) ***Clasificación****:* En este caso en vez de buscar predecir un valor, se busca hallar la categoría a la que pertenece basado en sus características, ejemplo la marca del vehiculo con base a sus piezas.

D. *Agrupamiento*

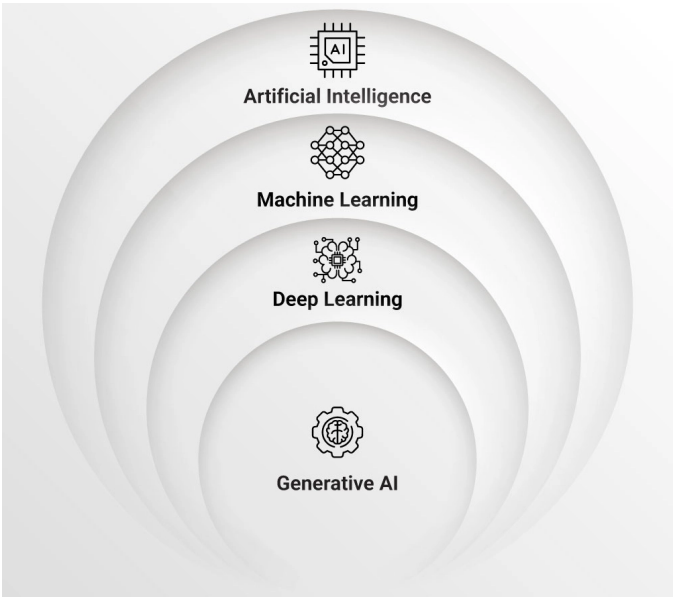
Conjunto de datos no etiquetados y se busca encontrar patrónes de estos mismos datos, aquí aplica el k-means

V. Modelos Determinista O Estocástico

1) ***Determinista****:* Este modelo indica que para una entrada de datos retoran una salida siempre consistente, un ejemplo de esto sería por ejemplo ¿Hay luz al medio día?

2) ***Estocástico****:* Este modelo indica que para una entrada puede retornar un resultado a partir de un conjunto de los posibles resultados, muy aleatorio, un ejemplo sería ¿Cual será el clima a medio día?

VI. Conjunto AI



- **Inteligencia Artificial:** Algoritmos generativos
- **Machine learning:** Utiliza metodos estadisticos, como regresión lineal, logística y arboles de decisión

- **Deep learning:** Redes Neuronales (utilizadas para resolver problemas complejos)
- **Generative AI:** Generación de nuevo contenido utilizando deep learning

VII. Inicio de material nuevo

VIII. Jupyter Notebook

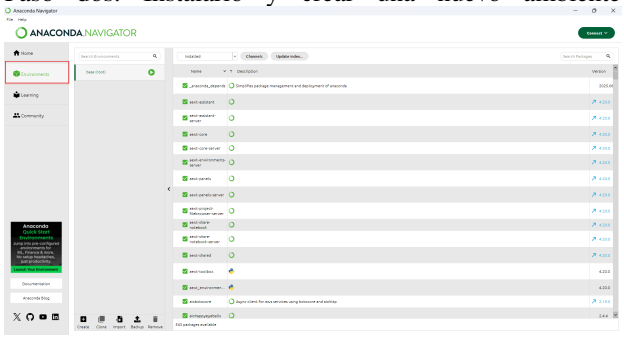
En el curso se ha explicado que para todas las entregas se van a utilizar Jupyter Notebooks debido a la facilidad que tiene esto para representar tanto celadas de texto como celdas de código. **Se puede usar una extensión de visual studio code para esto**

A. Recomendaciones del profesor

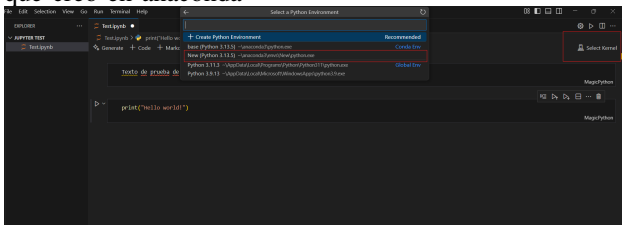
1) ***Utilizar Conda****:* Conda es una herramienta que permite tener diferentes ambientes de desarrollo por separado, lo cual permite trabajar con sin errores de compatibilidad al volver a proyectos antiguos

IX. Tutorial de instalación de Anaconda y Jupyter Notebook

1) Paso uno: Ingresar a la pagina oficial de Anaconda https://anaconda.org/ y descargar la versión requerida de Anaconda de acuerdo a su Sistema Operativo
2) Paso dos: Instalarlo y crear una nuevo ambiente



3) Paso tres: Seleccionar la versión del nuevo ambiente
4) Paso cuatro: Instalar la extensión de jupyter en vscode y crear un archivo con extensión .ipynb
5) Paso cinco: Seleccione en la opción de kernel, el ambiente que creo en anaconda



A. Recomendaciones de profesor

En caso de que falten dependencias utilizar **pip install** con las dependencias requeridas, es buena idea tener un requirements.txt para agilizar el proceso

X. Tensores

¿Que son tensores? Según [2] Los tensores han existido desde que William Hamilton acuñó el término hace 200 años para describir un objeto matemático que representa un conjunto de números con algunas propiedades de transformación.

A. *Definición de Tensor*

Un tensor es un objeto matemático que generaliza escalares, vectores y matrices en espacios de dimensiones superiores. Es un arreglo de números y funciones que abarca magnitudes físicas, transformaciones geométricas y diversas entidades matemáticas.

Existen tensores unidimensionales los cuales son llamados vectores, bidimensionales en formas de matriz y cuando k ¿ 2 ejes se deja de usar un nombre especifico y se le conoce como tensor de orden k

XI. Introducción a Pytorch

A. *Librerias necesarias para la manipulación de tensores, arreglos y estructuras de datos tabulares*

```
import torch
import numpy as np
import pandas as pd
```

B. *¿Como crear Tensores?*

La función arange (n) funciona para generar tensores prellenados con valores espaciados uniformemente, comenzando en 0 (inlcuido) hasta el n (no incluido) **los tensores recien creados se almacenan en memoria principal**

```
x= torch.arange(12,dtype=torch.float32)

x# Tensor unidimensional que puede ser operado
con diversas funciones que pueden ser
invocadas sobre el
```

C. Funciones sobre tensores

- **.numel()** Indica el numero de elementos que tiene el tensor
- **.shape** Se usa para acceder al tamaño de cada eje del tensor
- **.reshape** Se usa para reorganizar las dimensiones del tensor sin copiar los datos se puede pasar de un tensor de una dimensión por ejemplo el ejemplo anterior de 12 elementos, a un tensor bidimensional de 3 filas por 4 columnas
- **torch.zeros((z,x,y)) / torch.ones((z,x,y)) / torch.grandn(x,y)** Se utilizan para crear tensores de diferentes dimensiones, relleno con ceros, unos o numeros random
- **Operaciones elemento a elemento** pytorch permite operaciones aritmeticas entre tensores las cuales se aplicarán elemento a elemento

- **Concatenaciones de tensores** Mediante **torch.cat((X.Y),dim=k** se pueden concatenar tensores, siendo k el eje donde sobre el que se apilarán
- **Indexación lógica** Mediante **X==Y siendo X y Y tensores** se pueden realizar mascaras booleanas

XII. Creacion desde listas de python

```
A= torch.tensot([[2,1,4,3],
                 [1,2,3,4],
                 [4,3,2,1]], dtype=torch.float32)
A

tensor([[2.,1.,4.,3.],
        [1.,2.,3.,4.],
        [4.,3.,2.,1.]])
```

XIII. Indexacion y segmentación (slicing)

```
fila_ultima = A[-1] # ltima fila de X
submatriz = A[1:3] # Filas 1 y 2 de X
fila_ultima, submatriz
```

XIV. Broadcasting

El broadcasting en tensores permite operar tensores de diferentes formas al expandir automaticamente,

```
a = torch.arange(3).reshape((3, 1)) # Forma 3
    x1
b = torch.arange(2).reshape((1, 2)) # Forma 1
    x2
broadcast = a + b # Resultado de forma 3x2
gracias al broadcasting
```

XV. Operaciones In-place

Se indica que se tiene que tener cuidado debido a que cuando se tratan con modelos de machine learning o deep learning, nos encontramos con el hecho de que se ocupa muchisima memoria y hay que procurar ser óptimos en este sentido y al hacer ejecutar operaciones se puede dejar memoria asignada o se apunta a nuevas secciones de memoria, lo cual puede ser innecesario. Por eso se recomienda el uso de operaciónes totalmente in-place

XVI. Conversión a Numpy y Carga de datos desde CSV

Se puede interoperar con NumPy, convirtiendo tensores a arreglos y viceversa sin copiar datos

```
A_np = A.numpy()
print(type(A_np))
A_back = torch.from_numpy(A_np)
print(type(A_back))
```

Para cargar datos de un archivo CSV usaremos pandas para convertir sus columnas a tensores. Además de usar codificación one-hot para completar valores faltantes.

```
df = pd.read_csv('../data/house_tiny.csv')
inputs = df.iloc[:, :2]
inputs = pd.get_dummies(inputs, dummy_na=True)
inputs = inputs.fillna(inputs.mean())
X_csv = torch.tensor(inputs.to_numpy(dtype=
    float))
X_csv
```

XVII. Algebra Lineal Introducción

A. Escalar

Es un valor numérico que representa una sitaución a la vez, basicamente es un número, un unico elemento

B. Vectores

Se puede pensar en un vector como un arreglo de tamaño fijo de escalares, un vector no sería más que un conjunto de escalares

C. Matrices

Se puede ver como un arreglo de arreglos o como un tensor de orden 2

D. Tensores de orden superior

Ya no tienen nombre específico más que tensores de orden k

Una propiedad util de los escalares, vectores, matrices y tensores es que las operaciones elemento a elemento generan resultados que tienen la misma forma que sus operandos

XVIII. Producto Hadamard

Consiste en la multiplicación de elemento a elemento de dos matrices de un mismo tamaño

```
import numpy as np

# Definicin de matrices
X = np.array([[1, 2, 3],
              [4, 5, 6]])

Y = np.array([[7, 8, 9],
              [10, 11, 12]])

# Producto de Hadamard (elemento a elemento)
Z = X * Y

print(Z)
# Resultado:
# [[ 7 16 27]
# [40 55 72]]
```

XIX. Propiedades basicas de la aritmetica de tensores

Sumando o multiplicando un escalan y un tensor, producira un tensor del mismo tamaño como el tensor original. Cada elemento de el tensos es sumado o multiplicado por el escalar.

```
a = 2
X = torch.arange(24).reshape(2, 3, 4)
a + X, (a * X).shape


Resultado:

(tensor([[[ 2, 3, 4, 5],
        [ 6, 7, 8, 9],
        [10, 11, 12, 13]],

       [[14, 15, 16, 17],
        [18, 19, 20, 21],
        [22, 23, 24, 25]]]),
 torch.Size([2, 3, 4]))
```

XX. Reducción

Podemos realizar la suma de tensores se puede invocar sum() sin argumentos, esto hará que se reduzca a un escalar

```
# Crear una matriz A de forma (2, 3)
con valores [0,1,2,3,4,5]
A=torch.arange(6,dtype=torch.float32).reshape
    (2,3)
# Mostrar la forma de A y la suma
de todos sus elementos
A.shape, A.sum()

(torch.Size([2, 3]), tensor(15.))
```

Como lo reduce a lo largo de sus ejes, se puede especificar alguna de sus ejes x o y para sumar a lo largo del respectivo eje usando el parametro axis

```
A, A.sum(axis=0), A.sum(axis=1)

A--->(tensor([[0., 1., 2.],
       [3., 4., 5.]])

 A.sum(axis=0) ---> tensor([3., 5., 7.]),

A.sum(axis=1)--->tensor([ 3., 12.]))
```

Si se reduce a lo largo de todos sus ejes equivale a sumar todos los elementos de la matriz

```
A.sum(axis=[0, 1]) == A.sum()
tensor(True)
```

La media se calcula usando **mean**, la cual se puede definir como la suma de todos los elementos dividido entre el total de estos

```
A.mean(), A.sum() / A.numel()
#Comparacin entre ambas formas de sacar la
    media
A.mean(), A.sum() / A.numel() #Se obtiene el
    mismo resultado
```

XXI. Suma sin reducción

Si se desea conservar el numero de ejes al sumar como cuando se desea aprovechar el broadcasting, se usa

```
sum_A = A.sum(axis=1, keepdims=True)
sum_A, sum_A.shape


(tensor([[ 3.],
       [12.]]),

 torch.Size([2, 1]))
```

Si se desea calcular la suma acumulada de los elementos de un tensor, se puede usar **cumsum**

```
A.cumsum(axis=0)

tensor([[0., 1., 2.],
      [3., 5., 7.]])
```

XXII. Noticias habladas en clase

A. Alexander Wang fundador de Scale

Se habló del caso de Alexander Wang un joven de solo 28 años que fundó la empresa Scale AI, empresa por la cual Meta invirtió 13 mil millones de euros

B. Guerra de plataformas de LLM

Se mencionó que la competencia por ser la mejor IA, existe actualmente como si fuera una guerra entre plataformas, de la misma forma que ha ocurrido en otras cosas en el pasado, como las plataformas de streaming.

References

[1] S. Pacheco Portuguez, "Clase algebra lineal y manipulación de tensores mediante pytorch," *Tecnológico de Costa Rica*, 2025.

[2] I. Valchanov, "¿qué son los tensores?" *365 Data Science*, 2023. [Online]. Available: https://365datascience.com/tutorials/python-tutorials/tensor/