ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

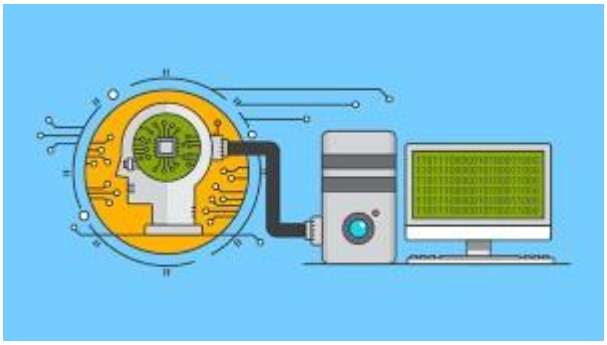
FUNDAMENTOS INTELIGENCIA ARTIFICIAL

ASIGNATURA:	Fundamentos de Inteligencia Artificial
PROFESOR:	Ing. Yadira Franco R
PERÍODO ACADÉMICO:	2025-A

TAREA SEMANA 3

TÍTULO:

Introducción a la Inteligencia Artificial IA



NOMBRE: Cristian Tambaco

Objetivo de la clase: Introducir los conceptos de **aprendizaje supervisado**, enseñar a aplicar modelos de clasificación en Python y evaluar su rendimiento.
Tarea: Crear un conjunto de datos, entrenar y evaluar modelos de clasificación, y comparar los resultados obtenidos.

Link GitHub:

https://github.com/CristianTambaco/Tarea_Introduccion_inteligencia_artificial_Tambaco_Cristian.git

1. Que es **algoritmos de aprendizaje supervisado** y **modelos supervisados, diferencias y uso**

Algoritmos de aprendizaje supervisado

Los algoritmos de aprendizaje supervisado son una herramienta importante en el campo del aprendizaje automático, y se utilizan para hacer predicciones o tomar decisiones basadas en datos. Estos algoritmos se entrenan utilizando datos etiquetados, que consisten en datos de entrada y la salida correcta correspondiente. A partir de esta asignación de entrada-salida, el algoritmo es capaz de hacer predicciones precisas sobre datos no etiquetados.

Un ejemplo común de un algoritmo de aprendizaje supervisado es el regresor lineal, que se utiliza para predecir una variable continua a partir de un conjunto de características. Otro ejemplo es el clasificador basado en árbol de decisión, que se utiliza para predecir una variable discreta a partir de un conjunto de características.

Además de la regresión y la clasificación, los algoritmos de aprendizaje supervisado también se pueden utilizar para hacer predicciones de series temporales, como los precios de las acciones o la demanda de un producto. Los algoritmos de aprendizaje supervisado también se utilizan en aplicaciones de reconocimiento de imágenes y de voz, así como en el filtrado de correo no deseado.

Los algoritmos de aprendizaje supervisado son una herramienta para hacer predicciones precisas a partir de datos etiquetados. Sin embargo, requieren un conjunto de datos etiquetados para el entrenamiento, y pueden tener dificultades para manejar datos muy complejos o datos con ruido. Además, pueden sobreentrenarse a los datos de entrenamiento, lo que reduce su capacidad para generalizar a datos no etiquetados. A pesar de estos posibles inconvenientes, los algoritmos de aprendizaje supervisado siguen siendo una herramienta importante en el aprendizaje automático y tienen una amplia gama de aplicaciones prácticas.

Modelos supervisados

Los modelos supervisados son un tipo de aprendizaje automático (machine learning) en el que el algoritmo se entrena utilizando datos etiquetados, es decir, con ejemplos en los que ya se conocen la entrada y la salida correcta. El objetivo del modelo es aprender una relación entre las entradas

(características o features) y las salidas (etiquetas o labels), para poder predecir con precisión nuevas salidas a partir de nuevas entradas.

Tipos principales de modelos supervisados

1. Clasificación

Su objetivo es predecir una categoría o clase discreta.

Ejemplo de salida: "perro", "gato", "otro" / "fraude", "no fraude"

Modelos comunes:

- Regresión logística
- K Vecinos más Cercanos (KNN)
- Árboles de decisión
- Random Forest
- Support Vector Machines (SVM)
- Redes neuronales
- Naive Bayes

Casos de uso:

- Detección de spam
- Diagnóstico médico (enfermo/sano)
- Clasificación de imágenes
- Análisis de sentimiento

2. Regresión

Su objetivo es predecir un valor numérico continuo.

Ejemplo de salida: Precio de una casa, temperatura, probabilidad de lluvia

Modelos comunes:

- Regresión lineal
- Regresión polinómica
- Árboles de regresión
- Random Forest (también puede hacer regresión)
- Support Vector Regression (SVR)
- Redes neuronales

Casos de uso:

- Predicción de ventas
- Estimación de precios inmobiliarios

- Pronóstico de demanda
- Predicción de crecimiento económico

Diferencias entre modelos de clasificación y regresión

CARACTERÍSTICA	CLASIFICACIÓN	REGRESIÓN
Tipo de salida	Discreta (categórica)	Continua (numérica)
Ejemplo	"Aprobado/Reprobado"	"85.6 puntos"
Métricas comunes	Precisión, F1, AUC-ROC	MSE, MAE, R^2
Función de pérdida	Cross-entropy, log loss	Error cuadrático medio (MSE)
Problemas típicos	Sobreajuste, clases desbalanceadas	Sobreajuste, escala de variables

Se usa clasificación cuando:

- Se quiere asignar una categoría o etiqueta a los datos.
- Se tiene un número finito de categorías posibles.
- Ejemplo: identificar si un correo es spam o no.

Se usa regresión cuando:

- Se necesita predecir un valor numérico.
- La salida puede tomar cualquier valor dentro de un rango.
- Ejemplo: predecir el precio de una acción mañana.

Pasos para un modelo supervisado

- Recopilar y preparar datos etiquetados.
- Dividir los datos en conjuntos: entrenamiento, validación y prueba.
- Seleccionar un modelo adecuado según el tipo de problema (clasificación o regresión).
- Entrenar el modelo con el conjunto de entrenamiento.
- Evaluar el rendimiento con métricas apropiadas.
- Ajustar hiperparámetros y mejorar el modelo si es necesario.
- Usar el modelo para hacer predicciones sobre nuevos datos.

Métricas comunes por tipo de modelo:

Para clasificación:

- Precisión (Accuracy)
- Sensibilidad (Recall)

- Especificidad
- Precisión (Precision)
- F1 Score
- Curva ROC y AUC
- Matriz de confusión

Para regresión:

- Error Cuadrático Medio (MSE)
- Error Absoluto Medio (MAE)
- Coeficiente de determinación (R^2)

2. Cómo funciona aprendizaje supervisado en Python, utilizando el conjunto

El aprendizaje supervisado es una tarea de aprendizaje automático en la que se entrena un algoritmo para encontrar patrones utilizando un conjunto de datos. El algoritmo de aprendizaje supervisado utiliza este entrenamiento para realizar inferencias de entrada y salida sobre conjuntos de datos futuros. De la misma manera que un profesor (supervisor) le daría tareas a un estudiante para que aprenda y aumente sus conocimientos, el aprendizaje supervisado proporciona conjuntos de datos a los algoritmos para que también puedan aprender y realizar inferencias.

Proceso del aprendizaje supervisado:

Recopilación de datos: El primer paso es obtener un conjunto de datos con ejemplos que ya están etiquetados. Por ejemplo, si se está tratando de predecir el precio de casas, los datos de entrenamiento consistirían en características como el tamaño de la casa, la ubicación, el número de habitaciones, etc., y su precio correspondiente.

Preprocesamiento de datos: Los datos deben ser preparados antes de entrenar el modelo. Esto puede incluir la limpieza de datos, manejo de valores faltantes, normalización de características, etc.

División de los datos: Usualmente, los datos se dividen en dos partes: conjunto de entrenamiento (train) y conjunto de prueba (test). El conjunto de entrenamiento se utiliza para enseñar al modelo, mientras que el conjunto de prueba se usa para evaluar su rendimiento.

Elección del modelo: Se selecciona un algoritmo de aprendizaje supervisado. Dependiendo del tipo de problema (regresión o clasificación), se elige un modelo adecuado. Algunos ejemplos comunes son:

Regresión: Si el objetivo es predecir valores continuos (como el precio de una casa), se puede usar un modelo de regresión como la regresión lineal.

Clasificación: Si el objetivo es clasificar los datos en categorías (como "spam" o "no spam"), se puede usar un modelo de clasificación como un árbol de decisión, una máquina de soporte vectorial (SVM) o una red neuronal.

Entrenamiento del modelo: Se usa el conjunto de datos de entrenamiento para ajustar los parámetros del modelo.

Evaluación del modelo: Una vez entrenado, se evalua el modelo con el conjunto de prueba. Algunas métricas comunes para la evaluación de modelos incluyen:

- Regresión: Error cuadrático medio (MSE), R^2.
- Clasificación: Precisión, Recall, F1-Score, etc.

Ajuste de hiperparámetros: A menudo, se pueden realizar ajustes a los parámetros del modelo para mejorar su rendimiento.

Predicción: Finalmente, una vez entrenado y validado el modelo, se puede utilizar para hacer predicciones sobre nuevos datos.

PRACTICA #1 identifique si un estudiante **aprobó o no aprobó** en función de características como el número de horas de estudio, nivel de conocimiento previo, entre otras, puedes seguir esta estructura.

Características (Entradas) para el Dataset:

1. **Horas de Estudio**: Número de horas que el estudiante dedicó al estudio.
2. **Nivel de Conocimiento Previo**: Un valor de 1 a 10, donde 1 es bajo conocimiento y 10 es un nivel alto de conocimiento previo.
3. **Asistencia a Clases**: Porcentaje de clases asistidas, entre 0 y 100.
4. **Promedio de Tareas**: Promedio de calificaciones de las tareas o ejercicios realizados.
5. **Tipo de Estudiante**: Si es un estudiante que estudia solo (1) o en grupo (0).

Salida (Etiqueta):

- **Resultado**: "Aprobado" o "No Aprobado" (0 = No Aprobado, 1 = Aprobado).

Entrenamiento de un modelo: Este dataset puede ser utilizado para entrenar un modelo de clasificación supervisado (por ejemplo, Regresión Logística, Árboles de Decisión, etc.) para predecir si un estudiante aprobará o no en función de las características dadas.

- Se crea el Dataset en Python y lo convierte en un Dataframe

```
# Paso 1: Crear el Dataset
# -------------------------

import pandas as pd
```

```
# Crear el conjunto de datos
data = {
    'Horas de Estudio': [5, 3, 8, 2, 7, 6, 4, 9, 3, 6],
    'Nivel de Conocimiento Previo': [7, 5, 8, 3, 9, 6, 4, 10, 5, 7],
    'Asistencia a Clases': [80, 60, 90, 50, 85, 75, 65, 95, 70, 80],
    'Promedio de Tareas': [8, 5, 9, 4, 8, 7, 6, 9, 5, 7],
    'Tipo de Estudiante': [1, 0, 1, 0, 1, 1, 0, 1, 0, 1],
    'Resultado': [1, 0, 1, 0, 1, 1, 0, 1, 0, 1]  # 1 = Aprobado, 0 = No
Aprobado
}

# Convertir a un DataFrame
df = pd.DataFrame(data)

# Mostrar el DataFrame
print("Dataframe:")
print(df)
```

- Se separan las características de entrada y las etiquetas de salida

```
# Paso 2: Preprocesar los Datos
# --------------------------

from sklearn.preprocessing import StandardScaler

# Separar las características (entradas) y la etiqueta (salida)
X = df.drop('Resultado', axis=1)  # Características
y = df['Resultado']  # Etiqueta (resultado)

# Normalizar las características
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Mostrar las características normalizadas
print("Características normalizadas:")
print(X_scaled)
```

- Se dividen los datos para entrenamiento el 70% para entrenamiento y el 30% para prueba.

```
# Paso 3: Dividir los Datos en Entrenamiento y Prueba
# --------------------------
```

```
from sklearn.model_selection import train_test_split

# Dividir los datos en entrenamiento y prueba (70% entrenamiento, 30%
prueba)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.3, random_state=42)

# Mostrar las dimensiones de los conjuntos de entrenamiento y prueba
print("Dimensiones de los conjuntos de entrenamiento y prueba:")
print(f"Datos de entrenamiento: {X_train.shape}")
print(f"Datos de prueba: {X_test.shape}")
```

- Se elige el modelo “Árbol de Decisión” y se entrena el modelo

```
# Paso 4: Elegir y Entrenar el Modelo
# --------------------------

from sklearn.tree import DecisionTreeClassifier

# Crear el modelo de Árbol de Decisión
model = DecisionTreeClassifier(random_state=42)

# Entrenar el modelo
model.fit(X_train, y_train)
```

- Se hace predicciones sobre el conjunto de pruebas y se evalúa la precisión del modelo

```
# Paso 5: Evaluar el Modelo
# --------------------------

from sklearn.metrics import accuracy_score, confusion_matrix

# Hacer predicciones sobre el conjunto de prueba
y_pred = model.predict(X_test)

# Evaluar la precisión del modelo
accuracy = accuracy_score(y_test, y_pred)
print(f"Precisión del modelo: {accuracy * 100:.2f}%")
```

```
# Mostrar la matriz de confusión
print("Matriz de Confusión:")
print(confusion_matrix(y_test, y_pred))
```

- Se hace las predicciones para la evaluación del modelo, tomando de ejemplo con datos quemados donde un estudiante A tiene 6 horas de estudio, nivel de conocimiento previo 7, 80% de asistencia a clases, promedio de tareas 7 y estudia solo (1). Otro estudiante B tiene 10 horas de estudio, nivel de conocimiento previo 5, 20% de asistencia a clases, promedio de tareas 6 y estudia en grupo (0).

```
# Paso 6: Hacer Predicciones
# --------------------------


# Ejemplo de predicción para un nuevo estudiante
# Un nuevo estudiante tiene:
# Estudiante A
# 6 horas de estudio, nivel de conocimiento previo 7, 80% de asistencia
a clases, promedio de tareas 7 y estudia solo (1)


new_student = pd.DataFrame([[6, 7, 80, 7, 1]], columns=['Horas de
Estudio', 'Nivel de Conocimiento Previo',
                                                        'Asistencia a
Clases', 'Promedio de Tareas', 'Tipo de Estudiante'])


# Estudiante B
# 10 horas de estudio, nivel de conocimiento previo 5, 20% de asistencia
a clases, promedio de tareas 6 y estudia en grupo (0)


# new_student = pd.DataFrame([[10, 5, 20, 6, 0]], columns=['Horas de
Estudio', 'Nivel de Conocimiento Previo',
#                                                         'Asistencia a
Clases', 'Promedio de Tareas', 'Tipo de Estudiante'])


# Normalizar las características del nuevo estudiante
new_student_scaled = scaler.transform(new_student)
```

```python
# Hacer la predicción
prediction = model.predict(new_student_scaled)

# Mostrar el resultado

print("\nNuevo estudiante:\n")
print(new_student)

if prediction == 1:
    print("\nEl estudiante ha aprobado.\n")
else:
    print("\nEl estudiante no ha aprobado.\n")
```

- Se imprime el Dataframe, características normalizadas, las dimensiones de los conjuntos de entrenamiento y prueba, precisión del modelo, matriz de confusión y los resultados del estudiante A, donde el estudiante ha aprobado.

```
Dataframe:
   Horas de Estudio  Nivel de Conocimiento Previo  Asistencia a Clases  \
0                 5                             7                   80
1                 3                             5                   60
2                 8                             8                   90
3                 2                             3                   50
4                 7                             9                   85
5                 6                             6                   75
6                 4                             4                   65
7                 9                            10                   95
8                 3                             5                   70
9                 6                             7                   80

   Promedio de Tareas  Tipo de Estudiante  Resultado
0                   8                   1          1
1                   5                   0          0
2                   9                   1          1
3                   4                   0          0
4                   8                   1          1
5                   7                   1          1
6                   6                   0          0
7                   9                   1          1
8                   5                   0          0
9                   7                   1          1
```

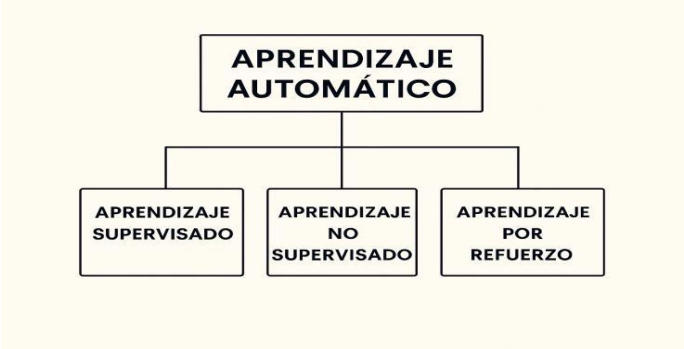
```
Características normalizadas:
[[-0.13678823  0.2847474   0.37796447  0.72231512  0.81649658]
 [-1.04870973 -0.6644106  -1.13389342 -1.08347268 -1.22474487]
 [ 1.23109403  0.7593264   1.13389342  1.32424438  0.81649658]
 [-1.50467048 -1.61356859 -1.88982237 -1.68540194 -1.22474487]
 [ 0.77513328  1.23390539  0.75592895  0.72231512  0.81649658]
 [ 0.31917253 -0.1898316   0.          0.12038585  0.81649658]
 [-0.59274898 -1.13898959 -0.75592895 -0.48154341 -1.22474487]
 [ 1.68705478  1.70848439  1.51185789  1.32424438  0.81649658]
 [-1.04870973 -0.6644106  -0.37796447 -1.08347268 -1.22474487]
 [ 0.31917253  0.2847474   0.37796447  0.12038585  0.81649658]]
Dimensiones de los conjuntos de entrenamiento y prueba:
Datos de entrenamiento: (7, 5)
Datos de prueba: (3, 5)
Precisión del modelo: 100.00%
Matriz de Confusión:
[[2 0]
 [0 1]]


Nuevo estudiante:

   Horas de Estudio  Nivel de Conocimiento Previo  Asistencia a Clases  \
0                 6                             7                   80

   Promedio de Tareas  Tipo de Estudiante
0                   7                   1

El estudiante ha aprobado.
```

- Para el estudiante B, se imprime el Dataframe, características normalizadas, las dimensiones de los conjuntos de entrenamiento y prueba, precisión del modelo, matriz de confusión y los resultados del estudiante B, donde el estudiante no ha aprobado.

```
Dataframe:
   Horas de Estudio  Nivel de Conocimiento Previo  Asistencia a Clases  \
0                 5                             7                   80
1                 3                             5                   60
2                 8                             8                   90
3                 2                             3                   50
4                 7                             9                   85
5                 6                             6                   75
6                 4                             4                   65
7                 9                            10                   95
8                 3                             5                   70
9                 6                             7                   80

   Promedio de Tareas  Tipo de Estudiante  Resultado
0                   8                   1          1
1                   5                   0          0
2                   9                   1          1
3                   4                   0          0
4                   8                   1          1
5                   7                   1          1
6                   6                   0          0
7                   9                   1          1
8                   5                   0          0
9                   7                   1          1
```

```
Características normalizadas:
[[-0.13678823  0.2847474   0.37796447  0.72231512  0.81649658]
 [-1.04870973 -0.6644106  -1.13389342 -1.08347268 -1.22474487]
 [ 1.23109403  0.7593264   1.13389342  1.32424438  0.81649658]
 [-1.50467048 -1.61356859 -1.88982237 -1.68540194 -1.22474487]
 [ 0.77513328  1.23390539  0.75592895  0.72231512  0.81649658]
 [ 0.31917253 -0.1898316   0.          0.12038585  0.81649658]
 [-0.59274898 -1.13898959 -0.75592895 -0.48154341 -1.22474487]
 [ 1.68705478  1.70848439  1.51185789  1.32424438  0.81649658]
 [-1.04870973 -0.6644106  -0.37796447 -1.08347268 -1.22474487]
 [ 0.31917253  0.2847474   0.37796447  0.12038585  0.81649658]]
Dimensiones de los conjuntos de entrenamiento y prueba:
Datos de entrenamiento: (7, 5)
Datos de prueba: (3, 5)
Precisión del modelo: 100.00%
Matriz de Confusión:
[[2 0]
 [0 1]]

  Nuevo estudiante:

     Horas de Estudio  Nivel de Conocimiento Previo  Asistencia a Clases  \
  0                10                             5                   20

     Promedio de Tareas  Tipo de Estudiante
  0                   6                   0

  El estudiante no ha aprobado.
```

Próxima clase TEST SERÁ NOTA DE LAS TAREAS



Resumen de Clasificación:

Tipo de Aprendizaje	Definición	Ejemplo de Tareas	Algoritmos Comunes
Supervisado	Datos etiquetados para predicción de resultados.	Clasificación, Regresión	Regresión Logística, SVM, Árboles de Decisión, KNN, Redes Neuronales
No Supervisado	Datos no etiquetados para encontrar patrones y estructuras.	Clustering, Reducción de Dimensionalidad	K-means, PCA, Agrupamiento Jerárquico
Refuerzo	Aprende mediante recompensas y penalizaciones en decisiones secuenciales.	Juegos, Control de Robots, Optimización de Estrategias	Q-learning, DQN, Algoritmos de Política

Lo que se necesita del Dataset para entrenar y evaluar estos modelos:

1. **Datos Etiquetados**: Necesitamos que el dataset esté correctamente etiquetado, es decir, que tengamos las columnas de características (entradas) y la etiqueta de salida (si el estudiante aprobó o no).
2. **División en Conjuntos de Entrenamiento y Prueba**: Para evaluar el rendimiento de los modelos, dividimos el dataset en **dos partes**: una para entrenar los modelos (usualmente 70-80% del total) y otra para probarlos (usualmente 20-30% del total).
3. **Datos Limpios**: Es importante que el dataset no tenga valores faltantes o datos irrelevantes que puedan afectar la precisión de los modelos.
4. **Escalado de Datos (si es necesario)**: Algunos modelos, como KNN y SVM, pueden beneficiarse de la **normalización** o **escalado** de los datos para asegurar que todas las características tengan la misma escala.

Subir el archivo y código al git hub **INDIVIDUAL LA TAREA**

CONCLUSIONES

- Los modelos supervisados requieren un conjunto de datos etiquetados para entrenarse. Puede ser costoso y lento de obtener, especialmente para problemas complejos.
- Los modelos pueden verse afectados por datos que no siguen patrones claros o que contienen errores.
- El aprendizaje supervisado tiene un amplio rango de aplicaciones en áreas como la medicina, la economía y la tecnología.

Link GitHub:

https://github.com/CristianTambaco/Tarea_Introduccion_inteligencia_artificial_Tambaco_Cristian.git

BIBLIOGRAFÍA

[1] «Algoritmos de aprendizaje supervisado | Interactive Chaos». Accedido: 11 de mayo de 2025. [En línea]. Disponible en: https://interactivechaos.com/es/wiki/algoritmos-de-aprendizaje-supervisado

[2] «Aprendizaje supervisado | Algoritmos, ejemplos - Datarmony». Accedido: 11 de mayo de 2025. [En línea]. Disponible en: https://datarmony.com/aprendizaje-supervisado-algoritmos-ejemplos/

[3] «Supervised Learning With Python: What to Know», Built In. Accedido: 11 de mayo de 2025. [En línea]. Disponible en: https://builtin.com/data-science/supervised-learning-python