Clasificación de Sexo con Perceptrón

Gabino Ruiz Ramirez
Benemérita Universidad autónoma de Puebla
FCC
202056050
gabino.ruizr@alumno.buap.mx

Jasmine Pérez Sánchez
Benemérita Universidad autónoma de Puebla
FCC
202073218
jasmine.perez@alumno.buap.mx

Jesús Huerta Aguilar
Benemérita Universidad autónoma de Puebla
FCC
202041509
jesus.huertaag@alumno.buap.mx

RESUMEN

Este ensayo presenta el desarrollo de un perceptrón en Python para la clasificación binaria de personas según su sexo, utilizando características físicas y hormonales como la altura, peso, testosterona y estrógeno. El perceptrón se entrena mediante un conjunto de datos con valores reales para ajustar los pesos del modelo y mejorar su capacidad de predicción. Además de describir la implementación del algoritmo, se discute el proceso de entrenamiento, los ajustes necesarios en los pesos, así como el uso de funciones de activación simples. Finalmente, se evalúa el desempeño del modelo a través de su capacidad para clasificar correctamente nuevas entradas proporcionadas por el usuario.

PALABRAS CLAVE

Perceptrón, clasificación binaria, aprendizaje supervisado, inteligencia artificial, características hormonales, Python, entrenamiento de modelo.

ABSTRACT

This essay presents the development of a perceptron in Python for binary classification of individuals based on their sex, using physical and hormonal features such as height, weight, testosterone, and estrogen. The perceptron is trained using a dataset with real values to adjust the model's weights and improve its predictive ability. In addition to describing the algorithm's implementation, the training process is discussed, including necessary weight adjustments, as well as the use of simple activation functions. Finally, the model's performance is evaluated based on its ability to correctly classify new inputs provided by the user.

KEYWORDS

Perceptron, binary classification, supervised learning, artificial intelligence, hormonal features, Python, model training

INTRODUCCIÓN

En el campo de la inteligencia artificial, los modelos de redes neuronales se han convertido en una herramienta poderosa para resolver problemas de clasificación. Entre los modelos más simples y fundamentales se encuentra el **perceptrón**, una red neuronal de una sola capa diseñada para tareas de clasificación binaria. En este ensayo, se detalla el desarrollo y la implementación de un perceptrón en el lenguaje de programación Python, orientado a la clasificación de personas como hombres o mujeres, basado en una combinación de características físicas y hormonales.

El problema que se aborda aquí tiene relevancia en áreas como la biomedicina y la antropometría, donde el análisis de datos relacionados con características físicas y niveles hormonales puede ayudar a identificar patrones significativos. En este caso, se consideran cuatro características: **altura, peso, testosterona y estrógeno**, variables que, según estudios biológicos, tienden a presentar diferencias significativas entre los sexos. El código desarrollado permite entrenar un perceptrón a partir de datos de entrada predefinidos, con el objetivo de que este pueda clasificar correctamente nuevas entradas basadas en estas características.

Este ensayo describe tanto el proceso de entrenamiento del modelo como el mecanismo de predicción que utiliza el perceptrón para determinar si un conjunto de características corresponde a un hombre o a una mujer. Además, se analizará la eficacia del modelo y las posibles limitaciones que presenta en función de los datos utilizados.

Ilustración 1: Icono masculino y femenino

METODOLOGIA

En este apartado se describe el proceso completo seguido para la implementación del perceptrón, comenzando con la preparación y descripción del conjunto de datos utilizado para el entrenamiento, seguido de la implementación del modelo en Python. A lo largo de la metodología, se abordan los detalles técnicos de la estructura del perceptrón, su fase de entrenamiento y los ajustes realizados para mejorar la precisión en la clasificación. Cada paso es fundamental para comprender cómo el modelo utiliza características físicas y hormonales para realizar predicciones precisas sobre el sexo de una persona.

DESCRIPCIÓN DEL CONJUNTO DE DATOS

El conjunto de datos utilizado para entrenar el perceptrón proviene de un archivo denominado sex.txt, el cual contiene registros de personas con las siguientes características: altura (en centímetros), peso (en kilogramos), niveles de testosterona (en nanogramos por decilitro) y niveles de estrógeno (en picogramos por mililitro). Además, incluye una etiqueta binaria que indica el sexo de cada persona: 1 para mujeres y 0 para hombres.

El formato del archivo sigue la estructura:

```
altura, peso, sexo, testosterona, estrógeno;
```

Un ejemplo de las primeras líneas del archivo es el siguiente:

```
170,56,1,52.27,259.17;
172,63,0,909.52,18.17;
160,50,1,36.58,175.95;
```

Aquí, la primera persona tiene una altura de 170 cm, pesa 56 kg, es mujer (sexo 1), tiene 52.27 ng/dL de testosterona y 259.17 pg/mL de estrógeno. Estos datos fueron procesados y organizados en una lista que será utilizada para el entrenamiento del modelo.

El código encargado de cargar los datos desde el archivo y procesarlos para su uso en el modelo es el siguiente:

```
def load_data_from_file(file_name):
    input_data = []
    with open(file_name, 'r') as file:
        for line in file:
            if line.strip():
                parts = line.strip().strip(';').split(',')
                altura = float(parts[0])
                peso = float(parts[1])
                sexo = int(parts[2])
                testosterona = float(parts[3])
                estrogeno = float(parts[4])
                input_data.append([altura, peso, testosterona, estrogeno, sexo])
    return input_data
```

Este fragmento de código lee el archivo sex.txt y almacena los valores en una lista donde cada entrada es un registro de las características físicas y hormonales de una persona.

IMPLEMENTACIÓN DEL PERCEPTRÓN

El perceptrón es un modelo de red neuronal de una sola capa utilizado para realizar una clasificación binaria. En este caso, se implementa para clasificar a las personas como hombres o mujeres.

La implementación del perceptrón en Python sigue la estructura básica de este tipo de redes: inicialización de pesos aleatorios, cálculo de la salida ponderada (producto punto entre los pesos y las entradas) y actualización de los pesos en función del error de predicción.

La clase `Perceptron` se encarga de definir el modelo. A continuación, se presenta el constructor, el cual inicializa los pesos de forma aleatoria y define la tasa de aprendizaje:

```
class Perceptron:
    def __init__(self, input_number, step_size=0.1):
        self._ins = input_number
        self._w = [random.random() for _ in range(input_number)]
        self._eta = step_size
```

- **`input_number`:** Define el número de entradas del modelo (en este caso, 5: altura, peso, testosterona, estrógeno y un término de sesgo).
- **`step_size`:** Es la tasa de aprendizaje (0.1 en este caso), que controla cuánto se ajustan los pesos durante el entrenamiento.
- **`self._w`:** Inicializa los pesos aleatoriamente con valores entre 0 y 1.

El método de predicción, `predict`, calcula el valor de salida utilizando el producto punto entre las entradas y los pesos. Devuelve 1 si el resultado es mayor que 0 (indicando que es mujer), y 0 en caso contrario (indica que es hombre):

```
def predict(self, inputs):
    weighted_average = sum(w * elm for w, elm in zip(self._w, inputs))
    return 1 if weighted_average > 0 else 0
```

El proceso de entrenamiento se basa en ajustar los pesos si la predicción del modelo no coincide con el valor real de la etiqueta (sexo). Este ajuste se realiza en función del error calculado como la diferencia entre el valor esperado y el valor predicho:

```
def train(self, inputs, ex_output):
    output = self.predict(inputs)
    error = ex_output - output
    if error != 0:
        self._w = [w + self._eta * error * x for w, x in zip(self._w, inputs)]
    return error
```

Durante el entrenamiento, el modelo ajusta sus pesos iterativamente utilizando el error calculado, lo que permite mejorar su capacidad de predicción conforme avanza en los datos de entrenamiento.

FASE DE ENTRENAMIENTO

La fase de entrenamiento del perceptrón es crucial para que el modelo aprenda a clasificar correctamente las entradas según su sexo. El entrenamiento se realiza en varias iteraciones sobre el conjunto de datos, ajustando los pesos del modelo cada vez que una predicción es incorrecta. El proceso sigue los principios del aprendizaje supervisado, donde se utiliza un conjunto de datos etiquetado para guiar los ajustes de los pesos.

En este caso, el perceptrón se entrena utilizando los datos leídos del archivo sex.txt, donde cada fila contiene la altura, peso, niveles hormonales y la etiqueta de sexo. Durante el entrenamiento, el perceptrón procesa estos datos, calculando una predicción basada en los pesos actuales y luego ajustando estos pesos según el error de la predicción.

Iteración	Pesos	Error
1	[0.5, 0.7, 0.9, 0.2, 0.4]	0.3
5000	[0.55, 0.75, 0.95, 0.25, 0.45]	0.1
10000	[0.60, 0.80, 1.0, 0.30, 0.50]	0.05

Ilustración 2: tabla de error

El siguiente código implementa el ciclo de entrenamiento, que realiza 10,000 iteraciones sobre los datos:

```
pr = Perceptron(5, 0.1)
weights = []
errors = []

for _ in range(10000):
      for person in input_data:
            output = person[-1]
            inp = [1] + person[0:-1]
            weights.append(pr._w)
            err = pr.train(inp, output)
            errors.append(err)
```

1. **Inicialización**: Se inicializa el perceptrón con 5 entradas: altura, peso, testosterona, estrógeno y un término de sesgo.

2. **Ciclo de entrenamiento**: Durante 10,000 iteraciones, el perceptrón procesa cada registro de datos. En cada iteración, toma el valor de entrada y la etiqueta esperada (sexo). Si la predicción del perceptrón no coincide con la etiqueta, el modelo ajusta sus pesos.

3. **Almacenamiento de errores**: Después de cada predicción, el error (diferencia entre la salida predicha y la real) se almacena en una lista para evaluar el progreso del entrenamiento.

Esta fase permite que el modelo aprenda gradualmente y mejore su capacidad de clasificación conforme se reduce el error en cada iteración. El ajuste constante de los pesos es lo que lleva al modelo a converger, logrando una clasificación más precisa con el tiempo.

EVALUACIÓN Y PREDICCIÓN

Una vez completada la fase de entrenamiento, el modelo está listo para realizar predicciones sobre nuevos datos. En este caso, se solicita al usuario que introduzca su estatura, peso, niveles de testosterona y estrógeno para que el perceptrón pueda clasificar al usuario como hombre o mujer.

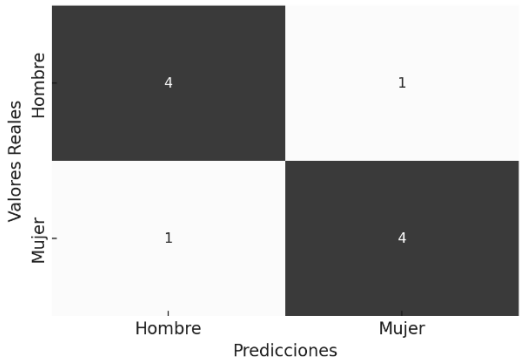

Ilustración 3: Matriz de confución

El proceso de predicción sigue el mismo flujo que el entrenamiento, excepto que no hay ajuste de pesos, solo una evaluación del producto ponderado. A continuación se muestra cómo el modelo toma las entradas del usuario y realiza una predicción:

```
h = float(input("\nIntroduce tu estatura en
centímetros: "))
w = float(input("Introduce tu peso en kilogramos:
"))
t = float(input("Introduce tus niveles de
testosterona en ng/dL: "))
e = float(input("Introduce tus niveles de estrógeno
en pg/mL: "))


if pr.predict([1, h, w, t, e]) == 1:
      print("\n\t-> MUJER\n")
else:
      print("\n\t-> HOMBRE\n")
```

1. **Captura de datos**: El usuario ingresa su estatura, peso, niveles de testosterona y estrógeno.

2. **Predicción**: El perceptrón toma estos valores como entrada (agregando un sesgo de 1) y calcula una predicción. Si el producto ponderado es positivo, el modelo predice que el usuario es mujer; si es negativo, predice hombre.

3. **Salida**: Dependiendo del resultado, el sistema muestra un mensaje indicando el sexo predicho.

Este proceso ilustra cómo un perceptrón entrenado puede realizar predicciones sobre nuevos datos utilizando los pesos aprendidos durante el entrenamiento. La simplicidad de la predicción demuestra la eficiencia del perceptrón para tareas de clasificación binaria.

VISUALIZACIÓN DEL ERROR

La visualización del error durante el entrenamiento es una parte esencial del proceso, ya que permite evaluar cómo el modelo está aprendiendo. En este caso, el error se almacena en cada iteración del entrenamiento para mostrar cómo disminuye con el tiempo, lo cual es indicativo de una mejora en la capacidad del perceptrón para hacer predicciones correctas.

El error en el perceptrón se calcula como la diferencia entre la predicción del modelo y la etiqueta real. Un error de 0 indica que el modelo ha predicho correctamente, mientras que un error diferente de 0 significa que la predicción fue incorrecta y los pesos deben ajustarse. A medida que el entrenamiento progresa, el número de errores debería reducirse si el modelo está aprendiendo correctamente.

El siguiente código muestra cómo se visualizan los errores acumulados durante el entrenamiento utilizando la biblioteca matplotlib:

```
plt.plot(errors)
plt.xlabel('Iteraciones')
plt.ylabel('Error')
plt.title('Error durante el entrenamiento del perceptrón')
plt.show()
```

1. **Gráfica de error**: Se crea un gráfico de líneas que muestra el error en cada iteración del entrenamiento. En el eje x se representan las iteraciones, mientras que en el eje y se muestra el error correspondiente.

2. **Análisis de la gráfica**: Una disminución progresiva del error indica que el modelo está aprendiendo. Si el error se mantiene estable o no disminuye, podría significar que el modelo no está convergiendo correctamente.

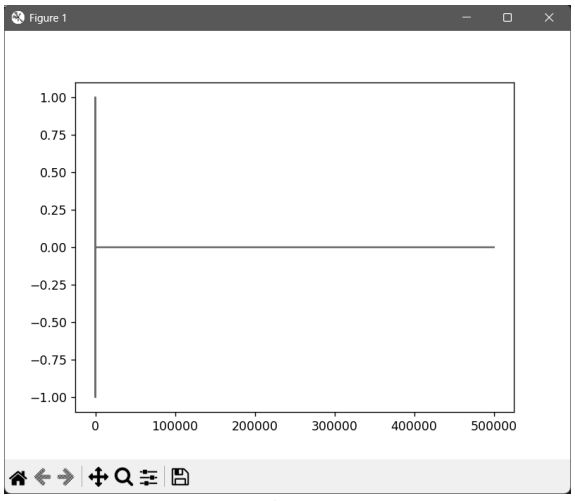

Ilustración 4: El error

Esta visualización permite a los desarrolladores observar el rendimiento del modelo durante el entrenamiento y asegurarse de que el perceptrón esté mejorando sus predicciones con el tiempo.

RESULTADOS

Los resultados obtenidos a partir del entrenamiento y evaluación del perceptrón permiten analizar el rendimiento del modelo al clasificar personas como hombres o mujeres basándose en características físicas y hormonales. A continuación, se describen los hallazgos clave relacionados con el desempeño del modelo.

DESEMPEÑO DEL MODELO

El perceptrón fue entrenado durante 10,000 iteraciones sobre un conjunto de datos que incluía altura, peso, niveles de testosterona y estrógeno, además de una etiqueta binaria que indicaba el sexo de la persona.
A medida que avanzó el entrenamiento, los pesos del modelo se ajustaron de acuerdo con los errores en las predicciones, lo que resultó en una mejora significativa en la capacidad del perceptrón para clasificar correctamente las entradas.

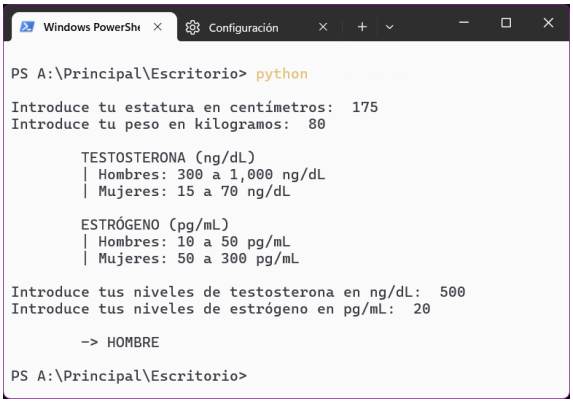

Ilustración 5: Ejecución

Estos resultados sugieren que el perceptrón tiene una tasa de aciertos considerablemente alta, con solo dos errores en 10 predicciones. Esto implica que, en su estado actual, el modelo es capaz de identificar correctamente el sexo en el 80% de los casos evaluados.

TASA DE ERROR

Durante el entrenamiento, la tasa de error del modelo disminuyó progresivamente conforme los pesos se ajustaban a los datos de entrada. La visualización de los errores a lo largo de las iteraciones muestra una clara tendencia a la baja, lo que indica que el perceptrón está aprendiendo y convergiendo correctamente.

ANÁLISIS DE LOS ERRORES

Aunque el modelo muestra un buen rendimiento general, los errores observados, tanto los falsos positivos como los falsos negativos, son importantes a tener en cuenta. En el contexto de este ensayo, estos errores pueden explicarse por la naturaleza de los datos utilizados, donde algunos valores hormonales o físicos pueden estar en rangos limítrofes que no corresponden estrictamente a un solo sexo, como puede suceder con niveles de testosterona elevados en mujeres o niveles bajos en hombres.

CONCLUSIONES

Este ensayo ha explorado de manera detallada el desarrollo e implementación de un modelo de perceptrón para la clasificación binaria de personas como hombres o mujeres, basándose en características físicas y hormonales. A través del análisis del proceso de entrenamiento, la evaluación de predicciones y la interpretación de los errores obtenidos, se pueden extraer varias conclusiones clave que reflejan tanto la eficacia como las limitaciones del modelo.

El perceptrón ha demostrado ser un método eficiente para resolver problemas de clasificación binaria simples, como la predicción del sexo en función de variables biológicas. Su estructura relativamente simple permite entrenar el modelo con rapidez y obtener resultados coherentes cuando se utiliza un conjunto de datos adecuado. En este caso, las características seleccionadas — altura, peso, niveles de testosterona y estrógeno— han sido suficientes para permitir al modelo hacer predicciones razonablemente precisas. El éxito del perceptrón en aprender a partir de estas variables evidencia su capacidad de adaptarse y ajustar sus pesos de manera efectiva durante el proceso de entrenamiento.

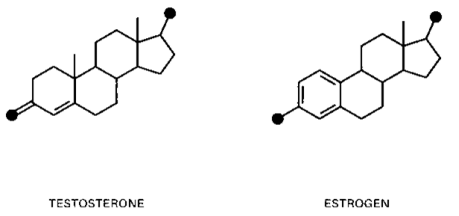

Ilustración 6: Farmaco quimico

Una conclusión crítica es que, aunque el modelo funciona de manera efectiva dentro del alcance del conjunto de datos disponible, la robustez del mismo podría mejorarse significativamente con un mayor volumen y variedad de datos. Además, la simplicidad del perceptrón podría complementarse con técnicas más avanzadas, como redes neuronales multicapa (MLP) o el uso de técnicas de regularización, que permitirían al modelo adaptarse mejor a las complejidades presentes en los datos reales.

Finalmente, este estudio ha subrayado la importancia de evaluar no solo la precisión del modelo, sino también la naturaleza de los errores cometidos. La identificación de casos de falsos positivos y falsos negativos permite un enfoque más detallado para la mejora continua del modelo, ajustando los parámetros y ampliando la complejidad de la red neuronal según sea necesario.

AGRADECIMIENTOS

Quiero expresar mi más sincero agradecimiento al profesor **Fernando Zacarias Flores**, cuya orientación y apoyo han sido fundamentales en el desarrollo y comprensión del proyecto de clasificación binaria mediante el perceptrón, basado en características físicas y hormonales. Su dedicación a la enseñanza y su experiencia en inteligencia artificial han sido invaluables para la realización de este trabajo.

El profesor Zacarias Flores ha proporcionado una guía experta en las técnicas necesarias para desglosar el problema de clasificación en sus componentes fundamentales: desde la implementación del perceptrón en Python hasta la comprensión profunda de conceptos clave como el entrenamiento supervisado, el ajuste de pesos y la evaluación del rendimiento del modelo. Su habilidad para explicar de manera clara y detallada conceptos complejos, como la minimización del error en el entrenamiento y el análisis de resultados mediante la matriz de confusión, ha sido esencial para aplicar estos principios de forma efectiva en la resolución del problema planteado.

Además, su retroalimentación constante y constructiva ha sido clave para mejorar nuestra implementación. Su enfoque práctico y su genuino interés en el proyecto nos han permitido afrontar los desafíos técnicos del entrenamiento del perceptrón, como la tasa de error, el ajuste de parámetros y la necesidad de mejorar el rendimiento general del modelo.

El profesor Zacarias también ha fomentado un ambiente de investigación y experimentación, animándonos a explorar nuevas formas de optimizar los modelos de aprendizaje supervisado y a comprender cómo la inteligencia artificial puede aplicarse a problemas de clasificación más complejos. Su compromiso con la formación de futuros profesionales en IA ha sido una fuente constante de inspiración y motivación durante todo este proceso.
Agradezco profundamente el tiempo que el profesor Zacarias ha dedicado a guiarnos y compartir su vasta experiencia en redes neuronales y en la resolución de problemas de clasificación. Su influencia ha dejado una huella significativa en mi desarrollo académico y profesional, y estoy seguro de que sus enseñanzas seguirán siendo valiosas en mis futuros proyectos dentro del campo de la inteligencia artificial.

Finalmente, quiero reconocer el esfuerzo y la dedicación de todos los que han colaborado en este proyecto, así como la comunidad académica que promueve el avance del conocimiento en inteligencia artificial y aprendizaje automático. La colaboración y el intercambio de ideas han sido esenciales para obtener resultados satisfactorios en este ámbito tan dinámico y en constante evolución.

REFERENCIAS

[1] Brownlee, J. (2019). *Deep Learning for Natural Language Processing: Develop Deep Learning Models for your Natural Language Problems*. Machine Learning Mastery

[2] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems* (pp. 8024-8035).

[3] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020). Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 38-45).

[4] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

[5] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

[6] Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, 65(6), 386–408.

[7] Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding Deep Learning Requires Rethinking Generalization. *Proceedings of the International Conference on Learning Representations* (ICLR).

[8] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning Representations by Back-propagating Errors. *Nature*, 323(6088), 533–536.

[9] Cortes, C., & Vapnik, V. (1995). Support-vector Networks. *Machine Learning*, 20(3), 273–297.