Universidad Tecnológica de chihuahua

Tecnologías de la información

Extracción de Conocimiento en Bases de Datos

Enrique Mascote

III.1. Análisis Supervisado

Marco Duarte – IDGS91N

1. Introducción

El análisis supervisado forma parte fundamental del aprendizaje automático, permitiendo resolver problemas donde se cuenta con datos de entrada y una variable objetivo conocida. En este documento se explora e investiga el funcionamiento de distintos algoritmos de regresión y clasificación, junto con sus métricas más utilizadas, fortalezas y limitaciones. Posteriormente, se desarrolla un caso práctico que integra el diseño, implementación y evaluación de un modelo supervisado, destacando los pasos clave del proceso completo de modelado.

2. Investigación de algoritmos

2.1. Algoritmos de Regresión

Regresión Lineal

Objetivo:

Predecir un valor numérico continuo a partir de una combinación lineal de las variables independientes.

Principio de funcionamiento:

Encuentra la línea recta (o hiperplano) que mejor se ajusta a los datos minimizando el error cuadrático medio. Supone una relación lineal entre las variables.

Métricas típicas:

- MAE, MSE, RMSE, R^2.

Fortalezas:

- Simplicidad e interpretabilidad.
- Muy rápida de entrenar.
- Útil como línea base.

Limitaciones:

• No funciona bien si la relación no es lineal.

• Sensible a valores atípicos.

Árbol de Regresión

Objetivo:

Predecir valores continuos mediante la subdivisión del espacio de datos en regiones.

Principio de funcionamiento:

Construye un árbol donde cada nodo divide los datos en función de la variable que genera menor error dentro de los subconjuntos.

Métricas típicas:

• MAE, MSE, RMSE.

Fortalezas:

• Maneja relaciones no lineales.

• Fácil de interpretar visualmente.

• Requiere poco preprocesamiento.

Limitaciones:

• Propenso al sobreajuste.

• Sensible a pequeñas variaciones en los datos.

2.2. Algoritmos de Clasificación

K-Nearest Neighbors (KNN)

Objetivo:

Clasificar una instancia basándose en las clases de sus vecinos más cercanos.

Principio de funcionamiento:

Calcula la distancia entre el punto nuevo y los puntos existentes, selecciona los K más cercanos y asigna la clase más frecuente.

Métricas típicas:

• Accuracy

• Precision

• Recall

• F1-score

Fortalezas:

• Fácil y eficaz en datasets pequeños.

• No requiere entrenamiento previo.

Limitaciones:

• Lento con conjuntos grandes.

• Sensible a la escala de los datos.

• Rendimiento disminuye cuando hay muchas variables.

Árbol de Decisión (Clasificación)

Objetivo:

Clasificar observaciones aplicando reglas en forma de árbol.

Principio de funcionamiento:

Divide los datos mediante atributos que maximizan la pureza de las clases (entropía o Gini).

Métricas típicas:

• Accuracy

• Matriz de confusión

• Precision, Recall, F1-score

Fortalezas:

• Muy interpretables.

• Manejan características categóricas y numéricas.

• Capturan relaciones no lineales.

Limitaciones:

• Sobreajuste si no se podan.

• Inestables ante cambios en los datos.

3. Caso de estudio y justificación

Caso práctico: Clasificación de clientes

Se desea clasificar a los clientes de una tienda en dos categorías: “Compra” y “No compra”. Las variables disponibles son:

Edad

Ingresos

Número de visitas

Tiempo promedio en el sitio web

El objetivo es identificar qué clientes tienen mayor probabilidad de compra.

Justificación del algoritmo elegido

Se selecciona el Árbol de Decisión debido a:

Su interpretabilidad para entender qué variables influyen más.

Su buen rendimiento con datos tabulares.

La capacidad de modelar relaciones no lineales.

Bajo requerimiento de preprocesamiento.

Esto lo convierte en una opción adecuada para equipos de marketing que requieren explicabilidad.

4. Diseño e implementación

Variables del modelo

X (entrada): edad, ingresos, visitas, tiempo

y (salida): compra (1 = Sí compra, 0 = No compra)

Pipeline propuesto

Carga y limpieza de datos

División entrenamiento/prueba (80/20)

Entrenamiento del modelo

Predicción

Evaluación con accuracy y F1-score

Código de implementación (Python + scikit-learn)

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
import pandas as pd

# Carga de datos
data = pd.read_csv("clientes.csv")

# Selección de variables
X = data[['edad', 'ingresos', 'visitas', 'tiempo']]
y = data['compra']
```

```
# División de datos
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Entrenamiento
model = DecisionTreeClassifier()
model.fit(X_train, y_train)

# Predicción
pred = model.predict(X_test)

# Métricas
print("Accuracy:", accuracy_score(y_test, pred))
print(classification_report(y_test, pred))
```

5. Resultados y evaluación

Los resultados muestran un desempeño adecuado en la precisión del modelo. Sin embargo, el Árbol de Decisión puede sobreajustarse si es demasiado profundo. Se recomienda:

Aplicar poda del árbol (max_depth, min_samples_split).

Comparar con modelos más robustos como Random Forest o Gradient Boosting.

Revisar la calidad de los datos y eliminar ruido.

6. Conclusiones y recomendaciones

El aprendizaje supervisado ofrece herramientas esenciales para resolver problemas de predicción y clasificación. Los algoritmos estudiados presentan características útiles según la naturaleza del problema. El caso práctico permitió comprender el flujo completo desde la investigación hasta la implementación y evaluación del modelo. Para mejorar resultados finales, es recomendable realizar ajuste de hiperparámetros, aumentar los datos disponibles y evaluar modelos más complejos cuando sea necesario.

7. Referencias

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning.

Geron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow.

Scikit-learn Documentation. https://scikit-learn.org