

Práctica de Programación en IA. CE en Inteligencia Artificial y BigData.

Primer Trimestre

Predicción del precio de vehículos de segunda mano mediante Aprendizaje Automático e integración web

Contexto

En esta práctica se desarrollará un sistema completo de predicción del precio de vehículos de segunda mano, abarcando todas las fases habituales de un proyecto de *Machine Learning aplicado*: selección y evaluación de modelos, análisis de métricas, exportación del modelo y su integración en una página web interactiva.

Se parte de un conjunto de datos ya analizado previamente mediante EDA, y en función de los resultados obtenidos podéis valorar si modificar o elegir otro dataset que sea menos complejo (una sola marca de vehículos por ejemplo). El objetivo principal es comparar distintos modelos de aprendizaje automático, seleccionar el más adecuado en base a métricas cuantitativas y desplegarlo en un entorno web accesible para usuarios finales.

Objetivo de la práctica

Construir una aplicación web que permita al usuario introducir las características de un vehículo de segunda mano y obtener una estimación de su precio, utilizando un modelo de aprendizaje automático entrenado y evaluado previamente.

Variables de entrada del modelo

El sistema deberá aceptar como mínimo las siguientes variables (podéis tomar otras decisiones sobre las características si lo consideráis conveniente):

- Marca del vehículo
- Modelo del vehículo
- Tipo de combustible
- Tipo de transmisión
- Kilometraje
- Año de fabricación

La variable objetivo será:

- Precio del vehículo

Parte 1: Evaluación y comparación de modelos (Machine Learning)

1. Modelos a evaluar

Se deberán entrenar y evaluar al menos tres modelos distintos, por ejemplo:

- Regresión Lineal
- Árboles de decisión
- Random Forest
- Gradient Boosting
- KNN
- SVM

La elección de modelos debe estar razonada en función del problema.

2. Preprocesamiento

Se deberá aplicar correctamente:

- Codificación de variables categóricas y/o uso de OneHotEncoding.
- Escalado de variables numéricas (si procede)
- Separación en conjuntos de entrenamiento y test
- Uso de *pipelines* cuando sea posible

3. Métricas de evaluación

Para cada modelo evaluado se deberán calcular y analizar, como mínimo, las siguientes métricas:

- MAE (Mean Absolute Error)
- MSE (Mean Squared Error)
- R^2 (Coeficiente de determinación)

Se deberá incluir:

- Una tabla comparativa de métricas
- Un análisis crítico de los resultados
- Justificación clara del modelo seleccionado

4. Selección y exportación del modelo

Una vez seleccionado el mejor modelo:

- Se entrenará con los mejores hiperparámetros
- Se exportará usando `joblib` o `pickle`
- Se guardarán también los transformadores necesarios (encoders, escaladores, etc.)

Parte 2: Implementación del sitio web

1. Backend

Se deberá implementar un backend que:

- Cargue el modelo entrenado
- Reciba los datos del usuario desde un formulario web
- Realice la predicción del precio
- Devuelva el resultado al usuario

Se puede utilizar, por ejemplo:

- Flask (recomendado por su simplicidad)
- FastAPI

2. Frontend

Se deberá desarrollar una página web que incluya:

- Un formulario con los campos necesarios
- Validación básica de entradas
- Un botón para calcular el precio
- Visualización clara del precio estimado

El diseño no será el aspecto principal, pero se valorará:

- Claridad
- Usabilidad
- Correcta organización del formulario

3. Flujo de la aplicación

1. El usuario introduce los datos del vehículo
2. El frontend envía la información al backend
3. El backend procesa los datos y llama al modelo
4. Se devuelve la predicción
5. El precio estimado se muestra en pantalla

Parte 3: Documentación y entrega

La entrega deberá incluir:

1. Notebook(s) de entrenamiento y evaluación con:
 - Modelos probados
 - Métricas obtenidas
 - Justificación del modelo final
2. Código del sitio web, incluyendo:
 - Backend
 - Frontend
 - Modelo exportado
3. Documento README explicando:
 - Estructura del proyecto
 - Cómo ejecutar la aplicación
 - Tecnologías utilizadas

Procura utilizar:

- Validación cruzada
- Ajuste de hiperparámetros (GridSearch y RandomizeSearch)
- Gráficas de errores reales vs predichos
- Despliegue en la nube