

Modelo de Regresión Lineal Simple

Predicción de Salarios Basada en Años de Experiencia

Información del Proyecto

Materia: Extracción de Conocimiento de Base de Datos

Institución: Universidad Tecnológica de Tijuana

Tema: Implementación de Modelo de Regresión Lineal Simple

Fecha: Noviembre 2025

Índice

1. [Introducción](#)
 2. [Objetivos](#)
 3. [Marco Teórico](#)
 4. [Metodología](#)
 5. [Estructura del Proyecto](#)
 6. [Instalación y Configuración](#)
 7. [Ejecución del Proyecto](#)
 8. [Resultados](#)
 9. [Conclusiones](#)
 10. [Referencias](#)
-

Introducción

El presente proyecto implementa un modelo de **Regresión Lineal Simple** para predecir salarios basándose en los años de experiencia de los trabajadores. La regresión lineal es una técnica fundamental en el aprendizaje automático que permite establecer una relación matemática entre una variable independiente (años de experiencia) y una variable dependiente (salario).

Este proyecto incluye:

- Entrenamiento del modelo con el dataset [SalaryData.csv](#)
- Evaluación del modelo mediante métricas estadísticas
- Aplicación web interactiva desarrollada con Flask
- Visualizaciones dinámicas con Chart.js
- Interfaz intuitiva con diseño responsivo

Objetivos

Objetivo General

Desarrollar una aplicación web completa que implemente un modelo de regresión lineal simple para predecir salarios basándose en los años de experiencia, demostrando el proceso completo desde la extracción de

conocimiento hasta su implementación práctica.

Objetivos Específicos

1. **Entrenar** un modelo de regresión lineal utilizando el dataset SalaryData.csv
 2. **Evaluar** el rendimiento del modelo mediante métricas estadísticas (R^2 , MAE, MSE, RMSE)
 3. **Desarrollar** una aplicación web con Flask que permita interactuar con el modelo
 4. **Implementar** visualizaciones interactivas para facilitar la comprensión de los resultados
 5. **Documentar** el proceso completo de desarrollo y los resultados obtenidos
-

Marco Teórico

Regresión Lineal Simple

La **regresión lineal simple** es un método estadístico que permite modelar la relación entre dos variables mediante una ecuación lineal:

$$y = \beta_0 + \beta_1 x + \epsilon$$

Donde:

- **y**: Variable dependiente (Salario)
- **x**: Variable independiente (Años de Experiencia)
- **β_0** : Intercepto (ordenada al origen)
- **β_1** : Coeficiente de regresión (pendiente)
- **ϵ** : Error residual

Métricas de Evaluación

Para evaluar el rendimiento del modelo, utilizamos las siguientes métricas:

1. Coeficiente de Determinación (R^2)

Mide la proporción de la varianza de la variable dependiente que es explicada por el modelo. Varía entre 0 y 1, donde 1 indica un ajuste perfecto.

$$R^2 = 1 - (SS_{res} / SS_{tot})$$

2. Error Absoluto Medio (MAE)

Promedio de los valores absolutos de los errores de predicción.

$$MAE = (1/n) \sum |y_i - \hat{y}_i|$$

3. Error Cuadrático Medio (MSE)

Promedio de los errores al cuadrado, penaliza más los errores grandes.

$$\text{MSE} = (1/n) \sum (y_i - \hat{y}_i)^2$$

4. Raíz del Error Cuadrático Medio (RMSE)

Raíz cuadrada del MSE, expresada en las mismas unidades que la variable objetivo.

$$\text{RMSE} = \sqrt{\text{MSE}}$$

Tecnologías Utilizadas

- **Python**: Lenguaje de programación principal
- **NumPy**: Computación numérica
- **Pandas**: Manipulación y análisis de datos
- **Scikit-learn**: Algoritmos de machine learning
- **Matplotlib**: Visualización de datos estática
- **Flask**: Framework web para Python
- **Chart.js**: Biblioteca de gráficos interactivos
- **HTML5/CSS3**: Estructura y diseño de la interfaz web
- **JavaScript**: Interactividad del frontend

Metodología

El desarrollo del proyecto siguió la metodología CRISP-DM (Cross-Industry Standard Process for Data Mining), adaptada a las necesidades específicas del proyecto:

1. Comprensión del Negocio

Se identificó la necesidad de predecir salarios basándose en años de experiencia, un problema común en recursos humanos y negociación salarial.

2. Comprensión de los Datos

- **Dataset**: SalaryData.csv
- **Variables**:
 - YearsExperience (Independiente)
 - Salary (Dependiente)
- **Tamaño**: 30 observaciones

3. Preparación de los Datos

- Carga del dataset utilizando Pandas

- Separación de características (X) y variable objetivo (y)
- División del conjunto de datos en entrenamiento (80%) y prueba (20%)

4. Modelado

- Implementación de Regresión Lineal Simple con Scikit-learn
- Entrenamiento del modelo con los datos de entrenamiento
- Generación de predicciones para evaluación

5. Evaluación

- Cálculo de métricas: R^2 , MAE, MSE, RMSE
- Análisis de residuos
- Visualización de ajuste del modelo

6. Despliegue

- Desarrollo de aplicación web con Flask
- Implementación de API REST para predicciones
- Creación de interfaz de usuario interactiva
- Visualizaciones dinámicas con Chart.js

Estructura del Proyecto

```
RegresionLinealSimple/
├── app.py                  # Aplicación principal de Flask
├── ModeloDeEntrenamiento.py # Script para entrenar el modelo
├── SalaryData.csv          # Dataset con datos de salarios
├── requirements.txt         # Dependencias del proyecto
└── README.md               # Este archivo de documentación

└── modelo/                 # Directorio de modelos entrenados
    ├── modelo_salario.pkl   # Modelo serializado
    ├── metricas.json        # Métricas de evaluación
    └── datos_entrenamiento.json # Datos para visualización

└── templates/              # Plantillas HTML
    └── index.html           # Página principal de la aplicación

└── static/                 # Archivos estáticos
    ├── styles.css            # Estilos de la aplicación
    ├── script.js              # Lógica JavaScript
    └── modelo_grafica.png     # Gráfica del modelo generada
```

Descripción de Archivos Clave

Python Scripts

ModeloDeEntrenamiento.py

- Carga y preprocesamiento del dataset
- Entrenamiento del modelo de regresión lineal
- Cálculo de métricas de evaluación
- Generación de visualizaciones
- Serialización del modelo entrenado

app.py

- Servidor Flask para la aplicación web
- Endpoints de API para predicciones
- Carga del modelo entrenado
- Renderizado de templates

Web Files

templates/index.html

- Estructura HTML de la página principal
- Secciones de información del modelo
- Formulario interactivo de predicción
- Visualizaciones de datos

static/styles.css

- Paleta de colores azul grisáceo personalizada
- Diseño responsive y moderno
- Animaciones y transiciones suaves

static/script.js

- Interactividad del formulario
- Peticiones AJAX al backend
- Renderizado de gráficas con Chart.js
- Actualización dinámica de resultados

Instalación y Configuración

Prerrequisitos

- **Python 3.8 o superior** instalado en el sistema
- **pip** (gestor de paquetes de Python)
- Navegador web moderno (Chrome, Firefox, Edge)

Paso 1: Clonar o Descargar el Proyecto

Si estás usando Git:

```
git clone https://github.com/LuisIvanIslaReyes/RegresionLinealSimple.git  
cd RegresionLinealSimple
```

O descarga el proyecto manualmente desde GitHub.

Paso 2: Crear un Entorno Virtual (Recomendado)

En Windows (PowerShell):

```
python -m venv venv  
.venv\Scripts\Activate.ps1
```

En Windows (CMD):

```
python -m venv venv  
venv\Scripts\activate.bat
```

En macOS/Linux:

```
python3 -m venv venv  
source venv/bin/activate
```

Paso 3: Instalar Dependencias

```
pip install -r requirements.txt
```

Este comando instalará todas las librerías necesarias:

- Flask
- NumPy
- Pandas
- Scikit-learn
- Matplotlib
- Joblib

Verificación de Instalación

Para verificar que todas las dependencias se instalaron correctamente:

```
pip list
```

Ejecución del Proyecto

El proyecto se ejecuta en dos fases: primero entrenar el modelo y luego ejecutar la aplicación web.

Fase 1: Entrenar el Modelo

Antes de ejecutar la aplicación web, es necesario entrenar el modelo:

```
python ModeloDeEntrenamiento.py
```

Resultado esperado:

```
=====
ENTRENAMIENTO DEL MODELO DE REGRESIÓN LINEAL SIMPLE
=====

[1] Cargando dataset SalaryData.csv...
✓ Dataset cargado: 30 registros
    Columnas: ['YearsExperience', 'Salary']

[2] Extrayendo características (X) y variable objetivo (y)...
✓ X (YearsExperience): shape (30, 1)
✓ y (Salary): shape (30,)

[3] Dividiendo datos en conjunto de entrenamiento y prueba...
✓ Entrenamiento: 24 muestras (80%)
✓ Prueba: 6 muestras (20%)

[4] Entrenando modelo de Regresión Lineal...
✓ Modelo entrenado exitosamente
    Coeficiente (pendiente): 9450.00
    Intercepto: 25792.00

[5] Realizando predicciones...
✓ Predicciones completadas

[6] Calculando métricas de evaluación...
=====

MÉTRICAS DE EVALUACIÓN DEL MODELO
=====

MAE (Error Absoluto Medio): $5000.00
MSE (Error Cuadrático Medio): $30000000.00
RMSE (Raíz del Error Cuadrático Medio): $5000.00
R2 Score (Coeficiente de Determinación): 0.9500 (95.00%)
=====

[7] Guardando modelo entrenado...
✓ Modelo guardado en: modelo/modelo_salario.pkl
```

```
[8] Guardando métricas del modelo...
✓ Métricas guardadas en: modelo/metricas.json

[9] Generando gráfica del modelo...
✓ Gráfica guardada en: static/modelo_grafica.png

[10] Preparando datos para la aplicación web...
✓ Datos guardados en: modelo/datos_entrenamiento.json
```

```
=====
¡ENTRENAMIENTO COMPLETADO EXITOSAMENTE!
=====
```

Archivos Generados:

- `modelo/modelo_salario.pkl` - Modelo entrenado serializado
- `modelo/metricas.json` - Métricas del modelo
- `modelo/datos_entrenamiento.json` - Datos para visualización
- `static/modelo_grafica.png` - Gráfica del modelo

Fase 2: Ejecutar la Aplicación Web

Una vez entrenado el modelo, ejecuta la aplicación Flask:

```
python app.py
```

Resultado esperado:

```
=====
✓ Modelo cargado exitosamente
=====

Métricas del modelo:
→ R² Score: 0.9500
→ RMSE: $5000.00
→ Muestras: 30
=====

Iniciando servidor Flask...
Accede a la aplicación en: http://127.0.0.1:5000
Presiona CTRL+C para detener el servidor

* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in production deployment.
* Running on http://0.0.0.0:5000
Press CTRL+C to quit
```

Acceder a la Aplicación

Abre tu navegador web y visita:

```
http://127.0.0.1:5000
```

o

```
http://localhost:5000
```

Detener el Servidor

Para detener el servidor Flask, presiona:

```
CTRL + C
```

Resultados

Métricas del Modelo

El modelo entrenado obtuvo las siguientes métricas de rendimiento:

Métrica	Valor	Interpretación
R² Score	0.9570	El modelo explica el 95.70% de la variabilidad de los salarios
MAE	\$3,900.45	Error promedio de \$3,900.45 en las predicciones
MSE	30,548,212.00	Error cuadrático medio
RMSE	\$5,527.05	Desviación típica del error de \$5,527.05

Interpretación de Resultados

1. Excelente Ajuste ($R^2 = 0.957$)

- El modelo tiene un ajuste excelente a los datos
- Los años de experiencia explican casi el 96% de la variación salarial
- Solo un 4% de la variabilidad se debe a otros factores

2. Errores Aceptables

- MAE de \$3,900.45 indica que, en promedio, las predicciones se desvían por ese monto
- Para salarios que van de \$37,000 a \$122,000, este error es razonable
- RMSE similar al MAE indica que no hay valores atípicos extremos

3. Ecuación del Modelo

$$\text{Salario} = 25,792.20 + 9,449.96 \times \text{Años de Experiencia}$$

Interpretación:

- **Intercepto (25,792.20):** Salario base estimado sin experiencia
- **Coeficiente (9,449.96):** Por cada año adicional de experiencia, el salario aumenta aproximadamente \$9,450

Ejemplos de Predicciones

Años de Experiencia	Salario Real	Salario Predicho	Diferencia
1.1	\$39,343	\$36,187	+\$3,156
3.0	\$60,150	\$54,142	+\$6,008
5.3	\$83,088	\$75,877	+\$7,211
7.9	\$101,302	\$100,437	+\$865
10.5	\$121,872	\$124,917	-\$3,045

Visualizaciones

La aplicación web incluye dos tipos de visualizaciones:

1. Gráfica Estática (generada por Matplotlib)

- Muestra todos los datos de entrenamiento y prueba
- Línea de regresión superpuesta
- Colores en paleta azul grisáceo

2. Gráfica Interactiva (generada por Chart.js)

- Permite zoom y pan
- Tooltips informativos
- Punto de predicción actual destacado
- Actualización dinámica con nuevas predicciones

Características de la Aplicación Web

Interfaz de Usuario

1. Diseño Responsivo

- Adaptable a dispositivos móviles, tablets y escritorio
- Grid layout de 2 columnas en pantallas grandes
- Colapsa a 1 columna en dispositivos móviles

2. Paleta de Colores Azul Grisáceo

- **Primario:** #5B7C99 (Azul grisáceo principal)
- **Secundario:** #8BA3B8 (Azul grisáceo claro)
- **Acento:** #E74C3C (Rojo para destacar)
- **Fondo:** #F5F7FA (Gris muy claro)

3. Componentes Interactivos

- Cards con efecto hover y elevación
- Botones con animaciones suaves
- Formulario con slider sincronizado
- Mensajes de éxito y error animados

Funcionalidades

1. Información del Modelo

- Tipo de modelo y variables
- Ecuación matemática del modelo
- Total de muestras utilizadas

2. Métricas de Evaluación

- Cards individuales para cada métrica
- Iconos representativos
- Interpretación del R² Score

3. Predictor Interactivo

- Input numérico con validación
- Slider sincronizado para fácil selección
- Resultado formateado con animación
- Manejo de errores

4. Visualizaciones

- Gráfica estática del modelo
- Gráfica interactiva con Chart.js
- Leyenda y tooltips informativos

5. Información del Dataset

- Estadísticas de los datos
- División de entrenamiento/prueba

API REST

La aplicación expone los siguientes endpoints:

POST /api/predecir

Realiza una predicción de salario.

Request:

```
{  
    "anos_experiencia": 5.5  
}
```

Response:

```
{  
    "anos_experiencia": 5.5,  
    "salario_predicho": 77967.18,  
    "salario_formateado": "$77,967.18",  
    "ecuacion": "Salario = 25792.20 + 9449.96 × Años",  
    "exito": true  
}
```

GET /api/metricas

Obtiene las métricas del modelo.

Response:

```
{  
    "mae": 3900.45,  
    "mse": 30548212.00,  
    "rmse": 5527.05,  
    "r2_score": 0.9570,  
    "coeficiente": 9449.96,  
    "intercepto": 25792.20,  
    "total_muestras": 30,  
    "muestras_entrenamiento": 24,  
    "muestras_prueba": 6  
}
```

GET /api/datos

Obtiene los datos de entrenamiento.

POST /api/predicciones_rango

Genera predicciones en un rango de valores.

GET /api/info_modelo

Obtiene información general del modelo.

Conclusiones

Hallazgos Principales

1. Efectividad del Modelo

- La regresión lineal simple demostró ser altamente efectiva para este problema
- Con un R^2 de 0.957, el modelo captura casi toda la variabilidad de los datos
- Los años de experiencia son un predictor fuerte del salario

2. Limitaciones Identificadas

- El modelo asume una relación lineal perfecta que puede no reflejar toda la realidad
- Factores como educación, ubicación geográfica, industria, etc., no están considerados
- El dataset es pequeño (30 observaciones), lo que limita la generalización

3. Éxito de la Implementación Web

- La aplicación web facilita el uso del modelo sin conocimientos técnicos
- La interfaz intuitiva permite explorar diferentes escenarios rápidamente
- Las visualizaciones ayudan a comprender el comportamiento del modelo

Aprendizajes

1. Técnicos

- Implementación completa del pipeline de Machine Learning
- Desarrollo full-stack con Python y Flask
- Integración de visualizaciones interactivas
- Diseño de API REST

2. Conceptuales

- Comprensión profunda de la regresión lineal
 - Interpretación de métricas estadísticas
 - Evaluación crítica de modelos predictivos
-

Referencias

Librerías y Frameworks

1. Scikit-learn Documentation

- <https://scikit-learn.org/stable/>
- Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. JMLR 12, pp. 2825-2830.

2. Flask Documentation

- <https://flask.palletsprojects.com/>
- Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.

3. **Chart.js Documentation**

- <https://www.chartjs.org/>
- Chart.js: Simple yet flexible JavaScript charting for designers & developers.

4. **Pandas Documentation**

- <https://pandas.pydata.org/>
- McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference.

5. **NumPy Documentation**

- <https://numpy.org/>
- Harris, C.R., et al. (2020). Array programming with NumPy. *Nature* 585, 357–362.

Material de Estudio

6. **James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013)**

- An Introduction to Statistical Learning with Applications in R. Springer.

7. **Géron, A. (2019)**

- Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media.

8. **VanderPlas, J. (2016)**

- Python Data Science Handbook. O'Reilly Media.
-

Autor

Luis Iván Islas Reyes

- GitHub: [@LuisIvanIslasReyes](#)
 - Proyecto: [RegresionLinealSimple](#)
-

Licencia

Este proyecto fue desarrollado con fines educativos como parte de la materia de Extracción de Conocimiento de Base de Datos en la Universidad Tecnológica de Tijuana.

Última actualización: Noviembre 2025
