Curso Data Science

Skill Lab

Prof Me Eng Marcelo Bianchi
Data Scientist

Aula 05 – Machine Learning
Simple Linear Regression

Machine Learning

Simple Linear Regression

Simple Linear Regression

Step 1 – Data Preprocessing

Step 2 – Simple Linear Regression

Theory

Regressão Linear

A regressão linear representa a relação entre as variáveis numéricas, usada para quantificar e fazer previsões baseadas no relacionamento entre variáveis. A relação de linearidade significa que quando uma (ou mais de uma no caso da regressão linear múltipla) variável independente aumenta ou diminui, a variável dependente aumenta ou diminui também. A regressão linear é uma técnica **supervisionada** que faz previsões de valores de **dados contínuos**. Por ser supervisionado, cada ver que treinarmos o modelo, ele estará capturando padrões que depois serão usados para prever valores de novos dados. O objetivo da regressão linear é descobrir como certas variáveis são relacionadas, como uma influencia a outra. Existem basicamente dois **tipos**:

- **Regressão Linear Simples:** examina a relação linear entre duas variáveis. Tem um preditor e uma predição. Ou seja, uma variável independente e uma variável dependente (target).
- **Regressão Linar Múltipla:** examina a relação linar entre mais de duas variáveis. Tem múltiplos preditores e uma predição. Ou seja, várias variáveis independentes e uma variável dependente (target).

Por exemplo, podemos observar que os salários de empregados de uma empresa depende de algumas variáveis como sua experiência, grau de escolaridade, cargo, cidade em que trabalha, e várias outras características. Este é um problema de regressão onde cada empregado representa uma observação e é pressuposto que as variáveis de experiência, escolaridade, cargo e cidade são independentes entre si, enquanto que salário depende delas. Perceba que a regressão linear pode ser usada tanto para descobrir como uma variável influencia a outra, quanto para fazer previsões futuras usando observações passadas. Usando este mesmo exemplo, podemos descobrir quanto o grau de escolaridade impacta no salário. Ao mesmo tempo, podemos usar os dados de observações dos últimos anos para prever como serão os salários nos próximos meses ou anos se informarmos como entrada a experiência, escolaridade, cargo e cidade. Veja a nomenclatura usada aqui:

- **variável independente** = features independentes, inputs, regressores ou variáveis preditoras. Usadas para prever o resultado. (x)
- **variável dependente** = features dependentes, target, alvo, outputs ou responses. É a que queremos descobrir. (y)
- **resíduos** = são os erros de predição, representam a diferença entre a previsão e o que realmente aconteceu.

Avaliação do Modelo

1. **Coeficiente de determinação**: mostra o quanto a variação em y pode ser explicada pela dependência em `x`. Quanto maior o coeficiente, maior a indicação de que o modelo se encaixou ou acertou (**better fit**) e significa que a variação do resultado pode ser explicada pelas diferentes entradas. Mais próximo de 1.0 também corresponde à soma dos erros ao quadrado ou **Sum of Squared Residuals (SSR)** mais próxima de 0 e um modelo perfeitamente inserido nos valores previstos, onde a resposta do previsto e do que realmente aconteceu é idêntica e sem erros.
2. **Overfitting**: são casos onde um modelo aprendeu tão bem que vai acertar em 100% dos casos conhecidos, tem um coeficiente 1.0 ou bem próximo de 1.0.
3. **Underfitting**: tem o valor de coeficiente próximo de 0 e um SSR maior. Evidencia o caso onde o modelo não capturou as dependências e os relacionamentos entre as variáveis e a SSR é alta.
4. **Intercept (b_0)**: mostra o ponto onde a regressão estimada cruza o eixo y quando `x=0`.
5. **Slope (b_1)**: representa o quanto vai aumentar (ou diminuir) a resposta da predição quando x_i aumentar 1.
6. **Fórmula da regressão linear simples**: $y = intercept + slope * x$
7. **Resíduos**: $residuos = y_i - b_0 - b_1 * x_i$ quando `i=1`.

Não existe um número mágico ou um valor exato, mas é importante saber que o valor muito próximo de 0.0 ou de 1.0 podem não ser um bom resultado. Devemos buscar um modelo que seja genérico o suficiente para lidar com novos dados de entrada e que ao mesmo tempo minimize o SSR.

Practical

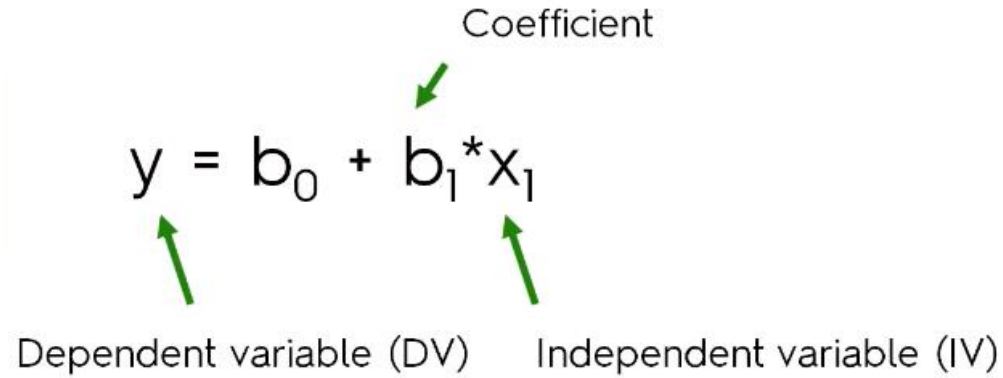
Simple Linear Regression

	YearsExperience	Salary
2	1.1	39343
3	1.3	46205
4	1.5	37731
5	2	43525
6	2.2	39891
7	2.9	56642
8	3	60150
9	3.2	54445
10	3.2	64445
11	3.7	57189
12	3.9	63218
13	4	55794
14	4	56957
15	4.1	57081
16	4.5	61111
17	4.9	67938
18	5.1	66029
19	5.3	83088
20	5.9	81363
21	6	93940
22	6.8	91738
23	7.1	98273
24	7.9	101302

- What is the correlation between the Years of Experience and Salary ?
- Independent Variable = Years of Experience
- Dependent Variable = Salary

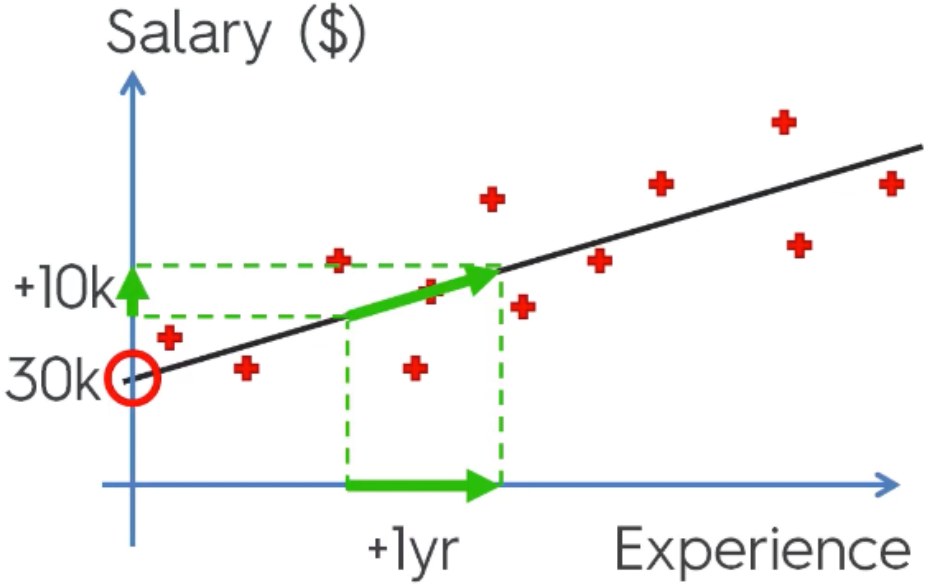
Regressions

Simple Linear Regression



Regressions

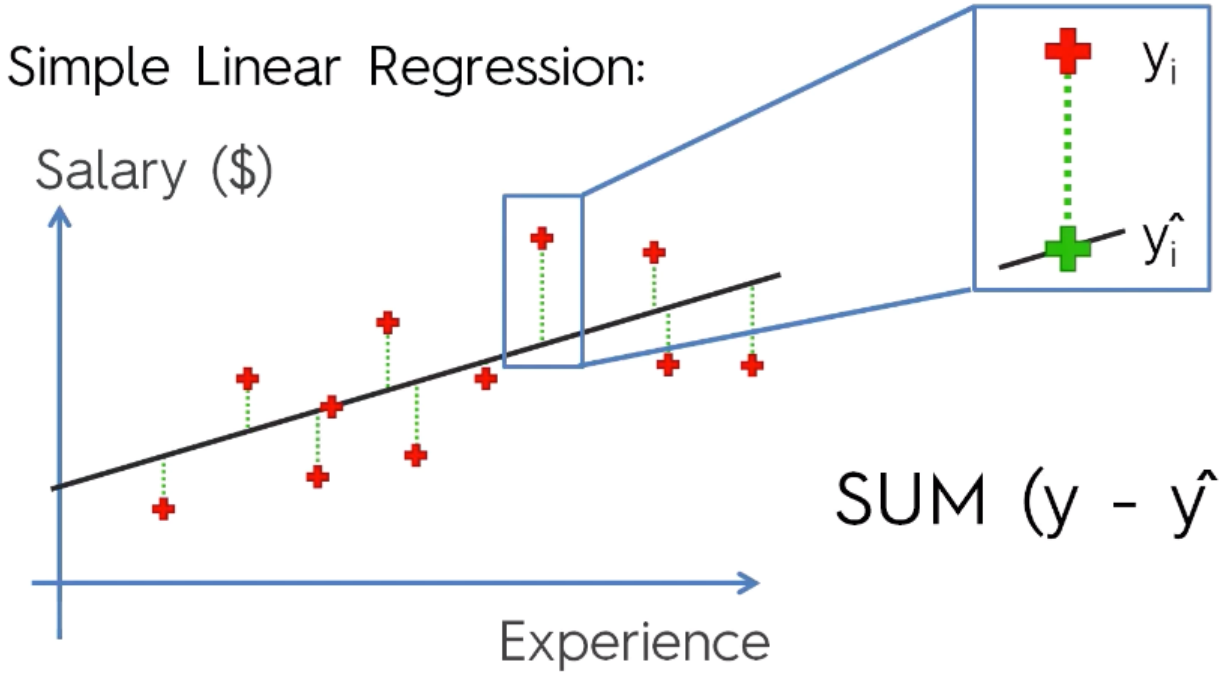
Simple Linear Regression:



$$y = b_0 + b_1 * x$$

Salary = b_0 + b_1 *Experience

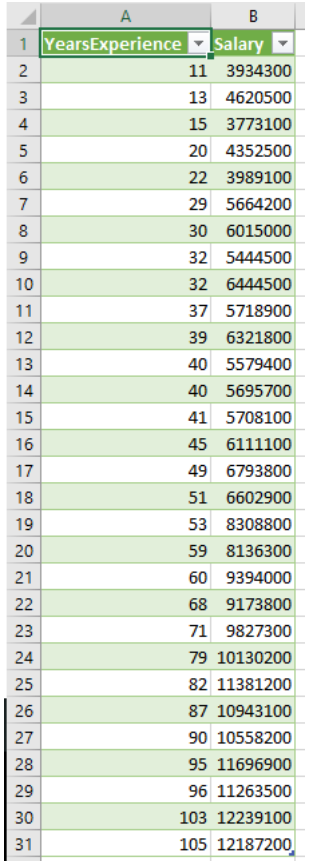
Ordinary Least Squares



$$SUM\ (y - \hat{y})^2 \rightarrow min$$


Salary vs Experience (Training set)
Salary
Years of Experience
120000
100000
80000
60000
40000
2
4
6
8
10


Salary vs Experience (Test set)
Salary
Years of Experience
120000
100000
80000
60000
40000
2
4
6
8
10



	A	B
1	YearsExperience	Salary
2	11	3934300
3	13	4620500
4	15	3773100
5	20	4352500
6	22	3989100
7	29	5664200
8	30	6015000
9	32	5444500
10	32	6444500
11	37	5718900
12	39	6321800
13	40	5579400
14	40	5695700
15	41	5708100
16	45	6111100
17	49	6793800
18	51	6602900
19	53	8308800
20	59	8136300
21	60	9394000
22	68	9173800
23	71	9827300
24	79	10130200
25	82	11381200
26	87	10943100
27	90	10558200
28	95	11696900
29	96	11263500
30	103	12239100
31	105	12187200

Dataset

- Years Experience
- Salary

```
# Simple Linear Regression

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Salary_Data.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 1].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, random_state = 0)
```

```
# Feature Scaling
"""from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
sc_y = StandardScaler()
y_train = sc_y.fit_transform(y_train)"""

# Fitting Simple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting the Test set results
y_pred = regressor.predict(X_test)

# Visualising the Training set results
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Years of Experience')
```

```
plt.ylabel('Salary')
plt.show()

# Visualising the Test set results
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Test set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```

1. Exercício

Você consegue prever a nota de um estudante de acordo com a quantidade de horas que ele estudou para uma prova?

Estudante	Horas_de_Estudo	Nota
1	1	53
2	5	74
3	7	59
4	8	43
5	10	56
6	11	84
7	14	96
8	15	69
9	15	84
10	19	83

Exercício – Regression – Linear Regression

- Crie novo_x contendo um array as seguintes horas de estudo dos novos alunos: 6, 9, 12, 15, 16, 4

- Aplique o modelo criado fazendo uma previsão de notas no novo conjunto de dados

- Conseguiu descobrir quais serão as notas dos novos alunos? Imprima as notas dos novos alunos.

Skill Lab

Muito obrigado!