Dr. Christian Tsoungui Obama

ML Cheat Sheet #3

Deep Dive: Mastering Regression

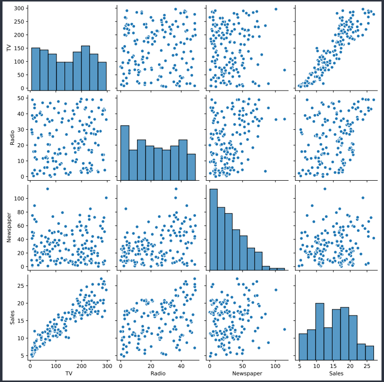
A Foundational Guide for Aspiring Data Scientists

Based on
"Hands-On Machine Learning with Scikit-learn, Keras & TensorFlow"
by

Aurélien Géron

“All Models Are Wrong But Some Are Useful” George P. Box

- Building a machine learning (ML) model means “trying” to find the “unknown” relationship between the target variable and data features
- Models are constructed by making “assumptions” on what the “true” might be based on available data
- In regression problems visualizing scatter plots (point clouds) between the target variable and each feature allows making assumptions on the true model



Linear Regression models assume a linear relationship

Sales	TV	Radio	Newspaper
22.1	230.1	37.8	69.2
10.4	44.5	39.3	45.1
$\vdots$			
18.4	232.1	8.6	8.7

$$\overset{\boldsymbol{y}}{\underbrace{\hphantom{\text{Sales}}}} \qquad \overset{\boldsymbol{X}}{\underbrace{\hphantom{\text{TV} \quad \text{Radio} \quad \text{Newspaper}}}}$$

Linear regression model for target variable $\boldsymbol{y}$ and data features $\boldsymbol{X} = (x_1, x_2, \ldots, x_L)$:

$$\boldsymbol{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_L x_L,$$

where $\boldsymbol{\Theta} = (\theta_0, \theta_1, \theta_2, \ldots, \theta_L)$ are the unknown model parameters. For the advertisement dataset the model is given by

$$\text{Sales} = \theta_0 + \theta_1 \text{TV} + \theta_2 \text{Radio} + \theta_3 \text{Newspaper}.$$

Training ML Model Means Estimating Best Possible Values for $\theta_0, \theta_1, \theta_2, \ldots, \theta_L$

- True model estimates Sales = 22.1 for TV = 230.1, Radio = 37.8, and Newspaper = 69.2, but also Sales = 10.4 for TV = 44.5, Radio = 39.3, and Newspaper = 45.1, and so on for all data entries.
- **Warning: it is impossible to know the true model with certainty**, so the best model is the one that predicts Sales values as closest to their real value for given values of TV, Radio, and Newspaper.
- The best possible model minimizes the error across all samples in the dataset, i.e., the mean squared error (MSE) obtained as

$$MSE(\Theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}^{(i)} - y^{(i)}\right)^2,$$

where N is the sample size and $\hat{y}^{(i)}$ the estimate of the target variable for the i-th sample in the training dataset.

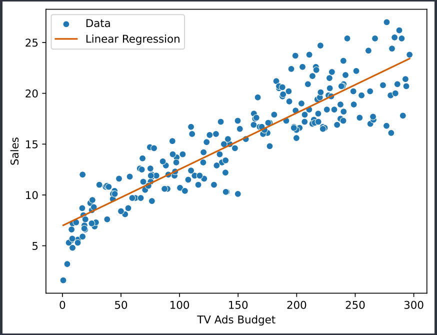
Estimation of best model parameters $\hat{\Theta} = (\hat{\theta}_0, \hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_L)$ in a linear model

Normal Equation: Exact formula to minimize MSE

$$\hat{\Theta} = (X^T X)^{-1} X^T y,$$

where X^T is the transpose of X.

```python
# Implementation of the Normal Equation
import numpy as np

def normal_equation_fit(X, Y):
    X_exp = np.c_[np.ones((X.shape[0], 1)), X] # Add one instance to estimate theta_0
    return np.linalg.inv(X_exp.T.dot(X_exp)).dot(X_exp.T).dot(Y)

# Estimation of best model parameters for Sales and TV data
best_parameters = normal_equation_fit(adsSales[["TV"]], adsSales[["Sales"]])

best_parameters

```



Model visualization: best parameter estimates $\hat{\theta}_0 = 6.975$, $\hat{\theta}_1 = 0.055$

Optimized Normal Equation with Scikit-learn

- Normal Equation might fail if the number of features L is large compared to sample size N or if there are multicollinearities (highly correlated features)
- Typically slow for datasets with large number of data features L
- Scikit-learn solves these issues by using the Moore-Penrose inverse (pseudo-inverse) $\boldsymbol{X}^{+} = (U\Sigma V^{T})^{+}$ of $\boldsymbol{X}$, i.e.,

$$\hat{\boldsymbol{\theta}} = \boldsymbol{X}^{+}\boldsymbol{y},$$

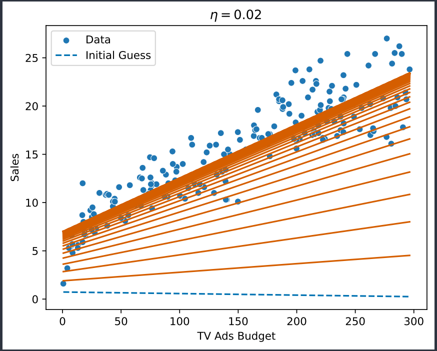
```python
# Training and evaluating on the training set: Linear regression
from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression()
lin_reg.fit(adsSales[["TV"]], adsSales[["Sales"]])

lin_reg.intercept_, lin_reg.coef_

```

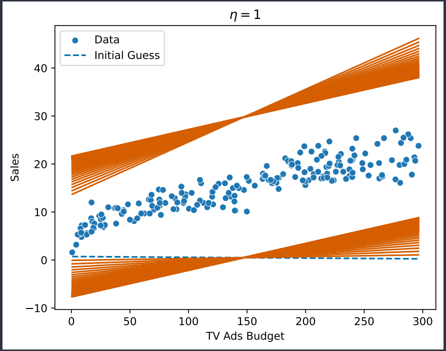
Numerical Approach to Estimate Best Linear Model: Batch Gradient Descent

- For large datasets, the Normal Equation is inefficient
- Batch gradient descent (BGD) finds the best model $\hat{\Theta}$ iteratively, i.e.,

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta \frac{2}{N} X^T \left(X\Theta^{(t)} - y \right).$$

- Learning rate η too small leads to very slow convergence
- η too large causes algorithm to diverge (i.e., does not find the best model parameters)





η **too small:** best model is found but algorithm might take too much time

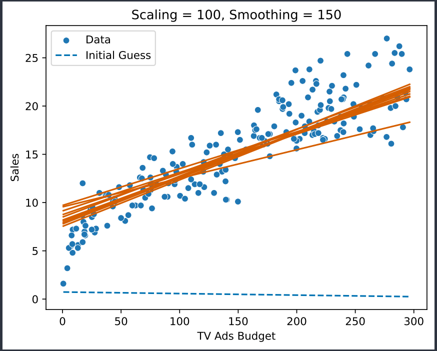
η **too large:** best model might not be found

Stochastic Gradient Descent

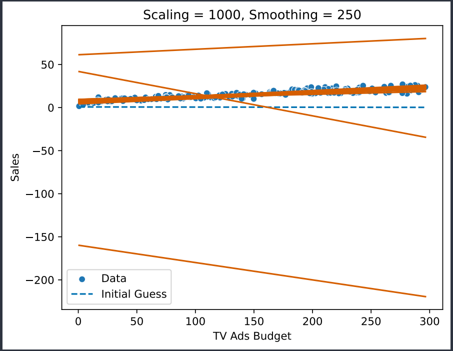
- BGD can get stuck at a local minimum (possibly good but not best model)
- Stochastic gradient descent (SGD) can escape local minima by iteratively training on one random sample per epoch, i.e.,

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta 2 X_{(i)}^{T} \left(X^{(i)} \Theta^{(t)} - y^{(i)} \right).$$

- SGD is good for large datasets and online learning
- SGD might bounce around and never find the best model

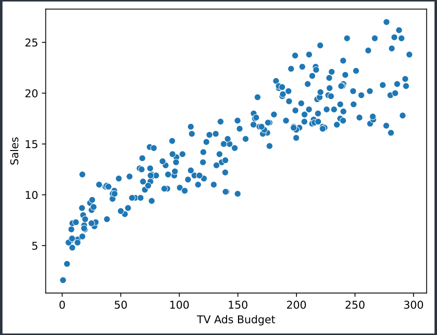


Fast convergence: SGD finds the best model relatively fast



Jumping behavior: SGD might bounce around quite a lot before finding best model

What If Scatter Plot Resembles a Curve?



Scatter plot Sales vs TV: point cloud actually resembles a curve, not a line

Polynomial regression model:

For curves, a *d*-degree polynomial might be better than a line, i.e.,

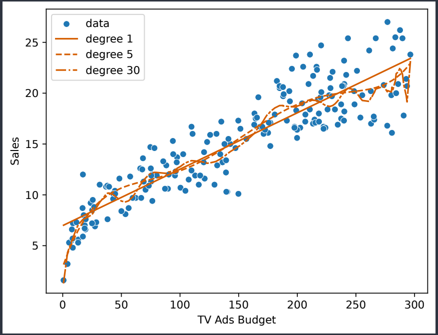
$$\boldsymbol{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \ldots + \theta_d x_1^d,$$

where $\boldsymbol{\Theta} = (\theta_0, \theta_1, \theta_2, \ldots, \theta_d)$ are the unknown model parameters. E.g.,

$$\text{Sales} = \theta_0 + \theta_1 \text{TV} + \theta_2 \text{TV}^2 + \ldots + \theta_d \text{TV}^d.$$

Complex Models Are Not Always The Best: Overfitting vs. Underfitting

- Simple models (e.g., linear models) may not capture important patterns and perform poorly both on training and new data: underfitting
- Complex models (e.g., high-degree polynomial regression) closely fit the data but often perform poorly on new data: overfitting
- **Trade-off:** Complex models are flexible and prone to overfitting, but simple models are too rigid and prone to underfitting.



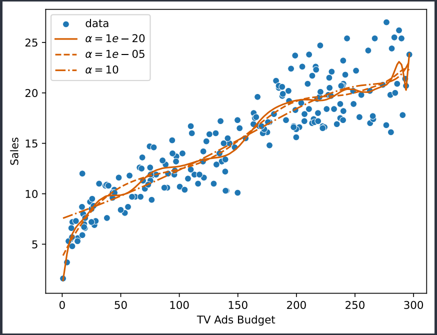
Model complexity: Line seems too simple and degree 30 polynomial too complex

Constraining Models With Regularization: Ridge Regression

- Flexibility of a complex model can be constrained by encouraging small values of Θ
- **Regularization** achieves this by minimizing the norm of Θ
- **Ridge regression:** uses the L_2-norm of Θ when searching for the best model, i.e., minimizes

$$MSE(\Theta) + \alpha \frac{1}{2} \sum_{k=1}^{L} \theta_k^2,$$

where α controls amount of regularization

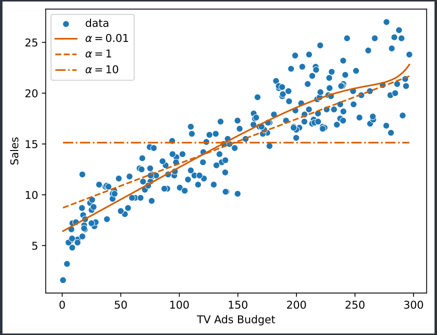


Effect of α on Ridge-regularized polynomial model (degree 30): model simplifies as α increases

LASSO Regression

- Ridge regression minimizes model parameters but never lets them go to zero
- **LASSO regression** can set the model parameter of less important features to zero: **feature selection**
- LASSO regression uses the L_1-norm of Θ and minimizes

$$MSE(\Theta) + \alpha \sum_{k=1}^{L} |\theta_k|.$$



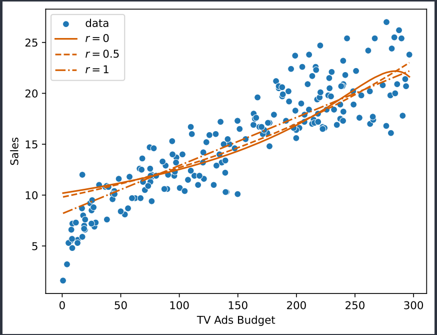
Effect of α on LASSO-regularized polynomial model (degree 30): model simplifies to the intercept θ_0 (horizontal dot-dashed line) as α increases

Elastic Net

- **Elastic Net** combines Ridge and LASSO regression with a mix ratio r such that the model minimizes

$$MSE(\Theta) + r\alpha \sum_{k=1}^{L} |\theta_k| + \frac{1-r}{2}\alpha \sum_{k=1}^{L} \theta_k^2.$$

- $r = 0$ and $r = 1$ correspond to Ridge and LASSO regression, respectively.



Effect of mix ratio r on polynomial model (degree 30) regularized with $\alpha = 0.7$: high similarity between Ridge- and Elastic Net-regulated models

Take-away Information

- Avoid plain linear regression (i.e., no regularization)
- Ridge regression preferable if no need for feature selection
- Elastic Net preferable to LASSO if some features are suspected to be useless

Models		Advantages	Disadvantages
Linear Regression	None	closed-form solution, deterministic, fast to train	assumes linearity, sensitive to outliers & multicollinearity, computationally expensive with many data features
Batch Gradient Descent (BGD)	Learning rate	iterative, faster than linear regression on large dataset	approximates model parameters, might get stuck at local minimum, slow for very large datasets
Stochastic Gradient Descent (SGD)	Learning rate	iterative, faster than BGD, online learning	approximates model parameters, requires learning schedule, oscillates around minimum

Did you find this useful?

Get complete notebook: » GitHub repository (click here) «

Comment below: Can you explain why the SGD oscillates around the minimum?

Follow me: for the next cheat sheet in this series.

Like this post: to help it reach more people.

Let's Connect!

/in/christiantsoungui @TsounguiChris