Weekly Notes on Linear Regression, Hyperparameters, and Feature Engineering

Andrew Leacock

Week 1: Feb 8–14, 2026

1 Introduction

I will try to keep everything as organized as possible but these are my notes, thoughts, and opinions on the subject matter that I have gone over this week.

- Linear Regression
- Hyperparameters
- Feature Engineering

2 Mathematics

Linear Regression (n Parameters)

$$\hat{y} = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$$

Vector Form

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b$$

Loss Function

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \hat{y}_i)$$

Gradient Descent

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta L(\theta_t)$$

Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Standardization

$$x' = \frac{x - \mu}{\sigma}$$

These are the equations I learned this week when going over hyperparameters and feature engineering.

3 Linear Regression

Linear regression is the process of finding the linear function that minimizes the error between predicted and actual values.

$$\hat{y} = wx + b$$

b is the base prediction and w tells you how much $\hat{y}$ changes for a one-unit increase in x.

4 Hyperparameters

Hyperparameters are the parameters set by the programmer that influence how well the machine learning model trains.

Learning Rate: α
Batch Size: m
Epochs: E

4.1 Learning Rate

Learning rate determines the step size used when updating parameters during gradient descent.

If α is too large, the algorithm may overshoot the minimum and fail to converge.

If α is too small, convergence will be very slow.

4.2 Batch Size

Batch size is the number of training samples used to compute the gradient during each parameter update.

4.2.1 Batch Types

There are 3 different types of batches used in gradient descent: stochastic, mini-batch, and full batch.

Stochastic Gradient Descent
Uses one sample for each parameter update.
Mini-Batch Gradient Descent
Uses a subset of the dataset for each update.
Full Batch Gradient Descent
Uses the entire dataset for each update.

4.3 Epochs

Epochs are the number of times the model sees the entire training dataset.

$$\text{Total Updates} = E \cdot \frac{N}{m}$$

5 Feature Engineering

Feature engineering is the process of creating, transforming, or selecting input features to improve a model's performance. It is crucial that the training set is representative of the cases the model must generalize to.

Models perform best when the **capacity** matches the task.
Too little = Underfitting
Too much = Overfitting
Capacity refers to how complex a function a model can represent.

5.1 Feature Scaling

You want your feature values to be on similar numerical scales relative to each other.
Standard

$$-1 \leq x \leq 1$$

If a feature has large numerical values, its corresponding weight tends to be small. If a feature has small numerical values, its corresponding weight tends to be large.

6 More

Machine learning is the process of finding patterns that are likely true for most members of a dataset.

7 Hands on Machine Learning with Scikit Learn and TensorFlow

7.1 Vectors

Predicting the output of a linear model

```
import numpy as np

w = np.array([1.0,2.5,-3.3])
b = 4
x = np.array([10,20,30])

f = np.dot(w,x) + b
print(f)
```

7.2 Linear Regression

Importing libraries

```
import pandas as pd
from sklearn.linear_model import LinearRegression
```

Creating historical vending visit data

```
df = pd.DataFrame({
    "days_since_last_visit": [14, 14, 21, 21, 28,
    "total_vends": [90, 130, 110, 160, 140, 200],
    "empty_selections": [0, 2, 1, 4, 2, 6],
    "minutes_on_site": [5, 12, 7, 18, 9, 22]
})
```

Defining features and label

```
X = df[[
    "days_since_last_visit",
    "total_vends",
    "empty_selections"
]]

y = df["minutes_on_site"]
```

Training the model

```
model = LinearRegression()
model.fit(X, y)
```

Predicting minutes on site

```
X_new = pd.DataFrame(
    [[21, 150, 3]],
    columns=[
        "days_since_last_visit",
        "total_vends",
        "empty_selections"
    ]
)

prediction = model.predict(X_new)

print("Predicted minutes on site:", prediction[0])
```

7.3 Scaling data

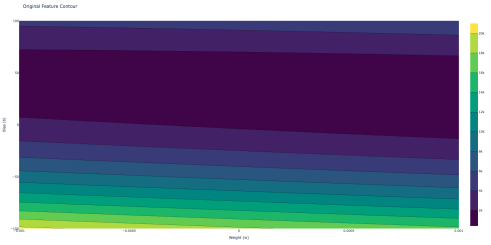

Figure 1: Loss surface using the original feature scale. The contours are elongated, indicating poor conditioning. This geometry causes gradient descent to move inefficiently, often requiring smaller learning rates and more iterations to converge.

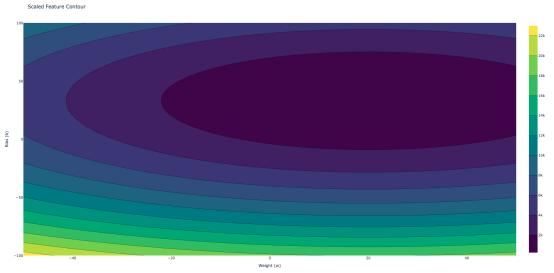

Figure 2: Loss surface after feature standardization. The contours are more symmetric and well-conditioned. This improves optimization stability and allows gradient descent to converge more efficiently.