Exercise Gradient Descent

Cheat Sheet

Algebra of Lines;

Linear function $f(x) = ax + b$ Translating to vector notation and why it works:

The dot product of two 2D vectors is defined as:

$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2$ So if $f(x) = \mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2$

Fix one component to 1 ⟶ no more dependence on the input x, e.g, $b_1 = 1$, the other component should depend on the input x.

That is exactly what we want for a linear regression line. The bias is independent from x, and the other input works with x, so $b_2 = x \longrightarrow \phi(x) = [1, x]$

So if $f(x) = \mathbf{a} \cdot \mathbf{b} = a_1 1 + a_2 b_2$, i.e., $\mathbf{b} = [1, b_1]$, $\mathbf{a} = [a_1, a_2]$ $b_1 = x$

Plugging it back in the to top form gives us a regression line

Optimization using Gradient Descent

$$\min_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$$

Gradient: The gradient $\nabla_{\mathbf{w}} TrainLoss(\mathbf{w})$ is the vector of partial derivatives, pointing in the direction in which the loss function increases the most.

Gradient descent algorithm: Start with an initial set of weights and update the weight vector repeatedly in the direction of the negative gradient (scaled by a *learning rate*):

```
Initialize w = [0, ..., 0]
For t = 1, ..., T  (epochs)
    w ← w - η · ∇TrainLoss(w)
```

Squared Loss for Linear Regression:

$$\text{TrainLoss}(\mathbf{w}) = \frac{1}{|\text{D}_{\text{train}}|} \sum_{(x,y) \in \text{D}_{train}} \frac{1}{2}(\mathbf{w} \cdot \phi(x) - y)^2$$

Gradient of the Squared Loss

The **gradient** of TrainLoss is the vector of partial derivatives with respect to the individual weights. After applying the **chain rule**, we get:

$$\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w}) = \frac{1}{|\text{D}_{\text{train}}|} \sum_{(x,y)\in \text{D}_{train}} \underbrace{(\underbrace{\mathbf{w} \cdot \phi(x) - y}_{\text{prediction}-\text{target}})}_{\text{derivative of outer function}} \cdot \underbrace{\phi(x)}_{\text{derivative of inner function}}$$

Exercise:

This exercise is on paper only. Check your understanding of the gradient descent algorithm on the example of linear regression.

Given are $w = [2, -4]$, $\eta = 0.1$, and the dataset $\text{D}_{\text{train}} = \{(2,2), (1,4), (3,0)\}$. With w[0] = bias and w[1] = weight.

1. Draw the initial regression line $f(x) = 2x - 4$ and the dataset points.

2. Train the regression model for three epochs using the gradient descent algorithm.

Remember each epoch is a full pass through the dataset. This means that you need to take all data points into account for each epoch before updating the weights and calculating the epoch loss.

- 2.1 Calculate the squared loss for each epoch.
- 2.2 Calculate the gradient for each epoch.
- 2.3 Update the weights for each epoch.
- 2.4 Draw the regression line after the third full epoch.

The following table might help you to organize your calculations:

Epoch	x	$f(x)$	Loss	Gradient / $\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$	w
0	2	-6	32.0	[-8, -16]	[2, -4]
0	3	-10	50.0	[-10, -30]	[2, -4]
0	1	-2	18.8	[-6, -6]	[2, -4]
Average / Sum values		-	33.33	[-8, -17.33]	[2.8, -2.27]
1	2	-1.74	6.99	[-3.74 -7.48]	[2.8, -2.27]
1	3	-4.01	8.04	[-4.01 -12.03]	[2.8, -2.27]
1	1	0.53	6.01	[-3.47 -3.47]	[2.8, -2.27]
Average / Sum values		-	7.02	[-3.74 -7.66]	[3.17, -1.5]
2	2	0.17	1.67	[-1.83 -3.66]	[3.17, -1.5]

Epoch	**x**	$f(x)$	**Loss**	**Gradient /** $\nabla_{\mathbf{w}}\text{TrainLoss}(\mathbf{w})$	$\mathbf{w}$
2	3	-1.33	0.88	[-1.33 -3.99]	[3.17, -1.5]
2	1	1.67	2.71	[-2.33 -2.33]	[3.17, -1.5]
Average / Sum values		-	5.27	[-1.83 -3.32666667]	[3.35, -1.17]