Ramtin Behesht Aeen ramtinba145822@gmail.com

Summary of

1 Neural Network

Neural Network Structure:
Simple Python Code:

```python
class Layer:
  def __init__(self):
      self.inp = None
      self.out = None

  def __call__(self, inp: np.ndarray) -> np.
      ndarray:
      return self.forward(inp)

  def forward(self, inp: np.ndarray) -> np.
      ndarray:
      raise NotImplementedError

  def backward(self, up_grad: np.ndarray) -> np.
      ndarray:
      raise NotImplementedError

  def step(self, lr: float) -> None:
      pass
```

- Input features:
$$a \tag{1}$$
- Features weights:
$$a \tag{2}$$
- Bias term:
$$a \tag{3}$$
- Activation function :
$$a \tag{4}$$
- Output of the neuron:
$$y \tag{5}$$

```python
def greet(name):
    print(f"Hello, {name}!")

greet("World")
```

Squared Error(SE): Most Common error function in linear regression is:

$$SE : (y^{(i)} - h(x^{(i)}, w))^2 \tag{6}$$

Sum of Squared Errors (SSE): Cost function should measure all predictions. Thus a choice could be Sum of Squared Error(SE)

$$SSE : \sum_{i=1}^{N} (y^{(i)} - h(x^{(i)}, w)) \tag{7}$$

Solve it analytically for one dimension: Predicted:

$$\widehat{y} = w_0 + w_1 x \tag{8}$$

SSE or Cost Function:

$$J(w_0, w_1) := \sum_{i=1}^{N} (y^{(i)} - \widehat{y}^{(i)})^2 = \sum_{i=1}^{N} (y^{(i)} - w_0 + w_1 x^{(i)})^2 \tag{9}$$

Assumptions:

$$\frac{\partial J}{\partial w_0} = 0\,, \frac{\partial J}{\partial w_1} = 0 \tag{10}$$

$\frac{\partial J}{\partial w_0} = 0$: thus:

$$\frac{\partial}{\partial w_0} (\sum_{i=1}^{N} (y^{(i)} - (w_0 + w_1 x^{(i)}))^2) = 0 \tag{11}$$ [1]

$$-2 \sum_{i=1}^{N} (y^{(i)} - w_0 - w_1 x^{(i)}) = 0 \tag{12}$$

For this equation to equal zero, the following condition must be met:

- $\sum_{i=1}^{N} y^{(i)} = 0 := Y$
- $\sum_{i=1}^{N} -w_0 = 0 := n w_0$
- $\sum_{i=1}^{N} -w_1 x^{(i)} = 0 := X$

Thus:

$$0 = Y - n w_0 - w_1 x^{(i)} \longrightarrow w_0 = \frac{(Y - w_1 x^{(i)})}{n} \tag{13}$$

[1] $fog(x)' = f(g(x))'g(x)'$