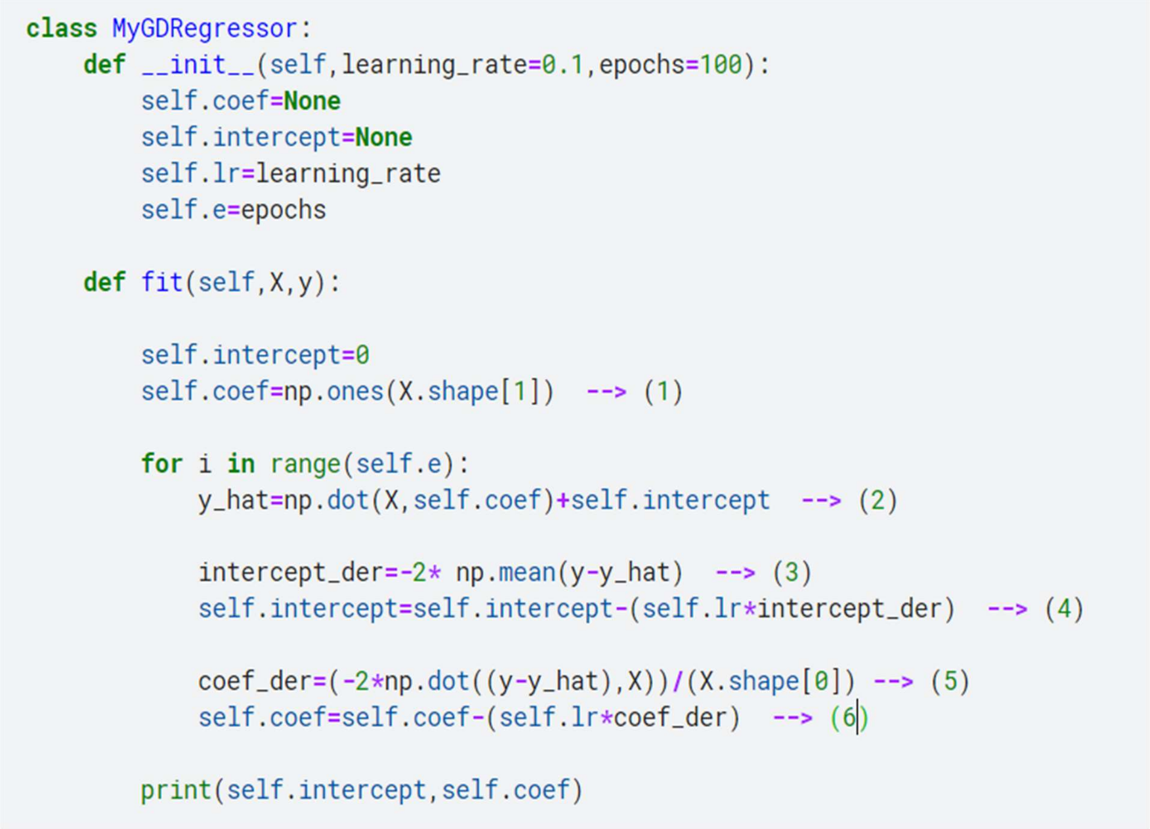
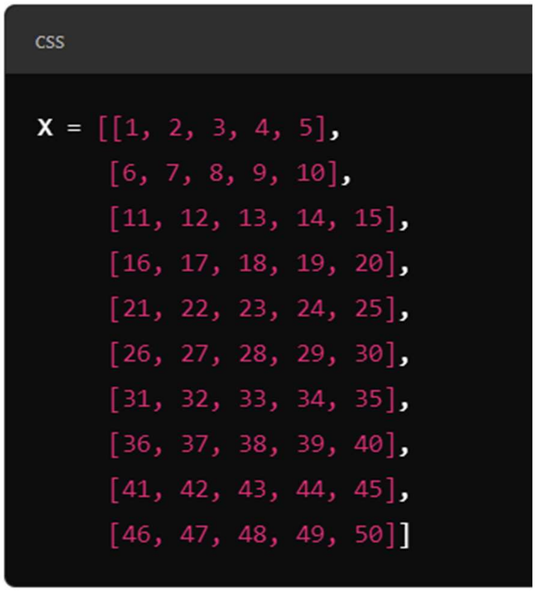
Understanding Gradient Descent

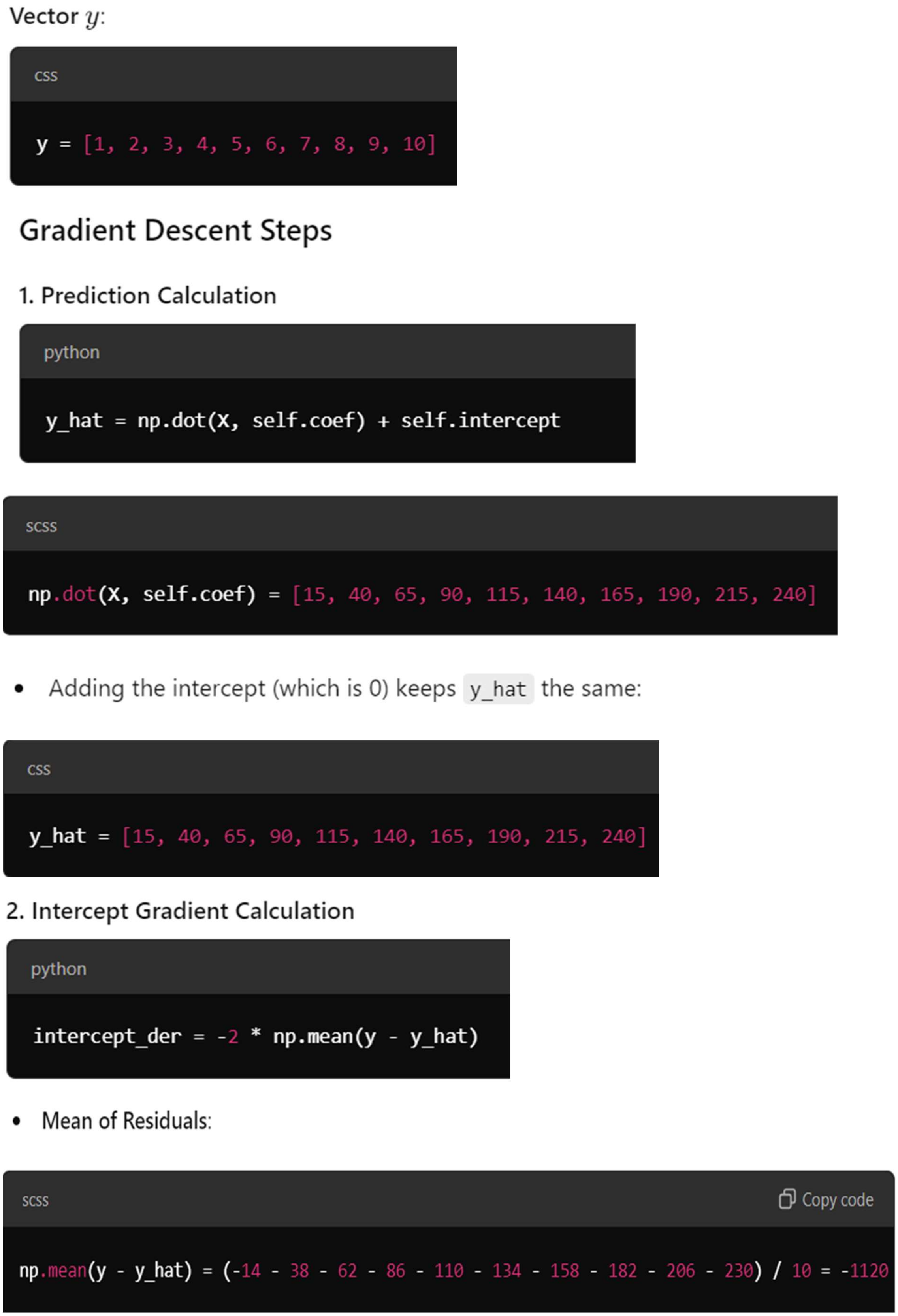


```
class MyGDRegressor:
    def __init__(self,learning_rate=0.1,epochs=100):
        self.coef=None
        self.intercept=None
        self.lr=learning_rate
        self.e=epochs

    def fit(self,X,y):

        self.intercept=0
        self.coef=np.ones(X.shape[1])  --> (1)

        for i in range(self.e):
            y_hat=np.dot(X,self.coef)+self.intercept  --> (2)

            intercept_der=-2* np.mean(y-y_hat)  --> (3)
            self.intercept=self.intercept-(self.lr*intercept_der)  --> (4)

            coef_der=(-2*np.dot((y-y_hat),X))/(X.shape[0]) --> (5)
            self.coef=self.coef-(self.lr*coef_der)  --> (6)

        print(self.intercept,self.coef)
```

Matrix X:



```
X = [[1, 2, 3, 4, 5],
     [6, 7, 8, 9, 10],
     [11, 12, 13, 14, 15],
     [16, 17, 18, 19, 20],
     [21, 22, 23, 24, 25],
     [26, 27, 28, 29, 30],
     [31, 32, 33, 34, 35],
     [36, 37, 38, 39, 40],
     [41, 42, 43, 44, 45],
     [46, 47, 48, 49, 50]]
```



Vector y:

```css
y = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Gradient Descent Steps

1. Prediction Calculation

```python
y_hat = np.dot(X, self.coef) + self.intercept
```

```scss
np.dot(X, self.coef) = [15, 40, 65, 90, 115, 140, 165, 190, 215, 240]
```

- Adding the intercept (which is 0) keeps `y_hat` the same:

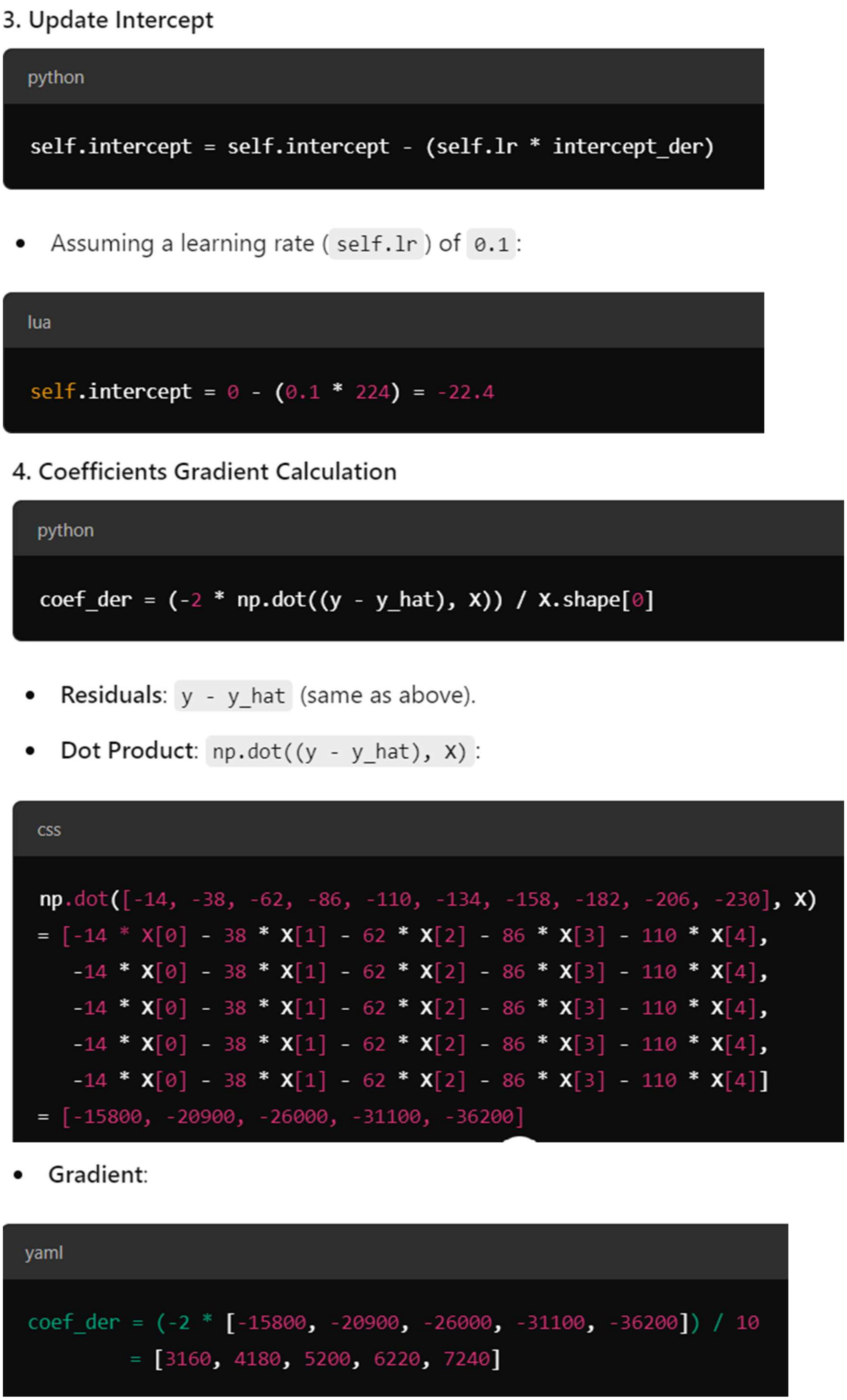
```css
y_hat = [15, 40, 65, 90, 115, 140, 165, 190, 215, 240]
```

2. Intercept Gradient Calculation

```python
intercept_der = -2 * np.mean(y - y_hat)
```

- Mean of Residuals:

```scss
np.mean(y - y_hat) = (-14 - 38 - 62 - 86 - 110 - 134 - 158 - 182 - 206 - 230) / 10 = -1120
```



3. Update Intercept

```python
self.intercept = self.intercept - (self.lr * intercept_der)
```

- Assuming a learning rate (`self.lr`) of `0.1`:

```lua
self.intercept = 0 - (0.1 * 224) = -22.4
```

4. Coefficients Gradient Calculation

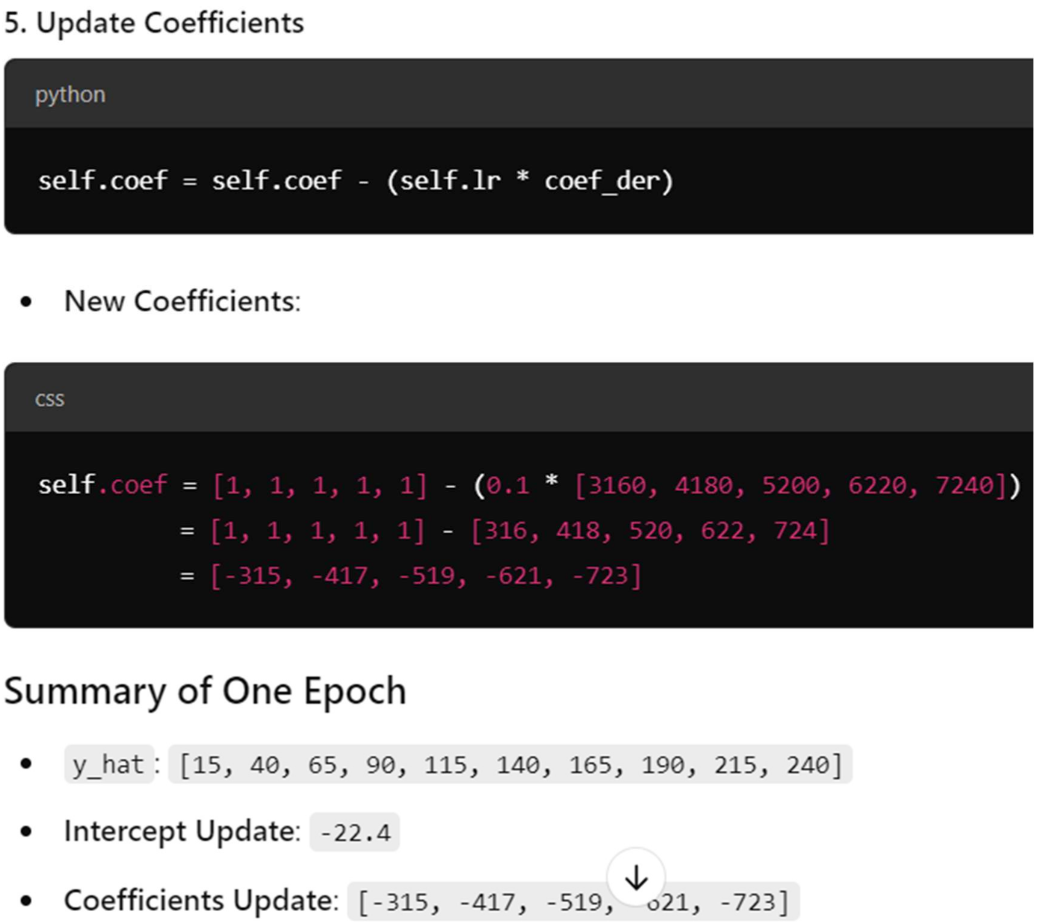
```python
coef_der = (-2 * np.dot((y - y_hat), X)) / X.shape[0]
```

- **Residuals**: `y - y_hat` (same as above).
- **Dot Product**: `np.dot((y - y_hat), X)`:

```css
np.dot([-14, -38, -62, -86, -110, -134, -158, -182, -206, -230], X)
= [-14 * X[0] - 38 * X[1] - 62 * X[2] - 86 * X[3] - 110 * X[4],
   -14 * X[0] - 38 * X[1] - 62 * X[2] - 86 * X[3] - 110 * X[4],
   -14 * X[0] - 38 * X[1] - 62 * X[2] - 86 * X[3] - 110 * X[4],
   -14 * X[0] - 38 * X[1] - 62 * X[2] - 86 * X[3] - 110 * X[4],
   -14 * X[0] - 38 * X[1] - 62 * X[2] - 86 * X[3] - 110 * X[4]]
= [-15800, -20900, -26000, -31100, -36200]
```

- **Gradient**:

```yaml
coef_der = (-2 * [-15800, -20900, -26000, -31100, -36200]) / 10
         = [3160, 4180, 5200, 6220, 7240]
```



5. Update Coefficients

```python
self.coef = self.coef - (self.lr * coef_der)
```

- New Coefficients:

```css
self.coef = [1, 1, 1, 1, 1] - (0.1 * [3160, 4180, 5200, 6220, 7240])
          = [1, 1, 1, 1, 1] - [316, 418, 520, 622, 724]
          = [-315, -417, -519, -621, -723]
```

Summary of One Epoch

- `y_hat`: `[15, 40, 65, 90, 115, 140, 165, 190, 215, 240]`
- Intercept Update: `-22.4`
- Coefficients Update: `[-315, -417, -519, -621, -723]`