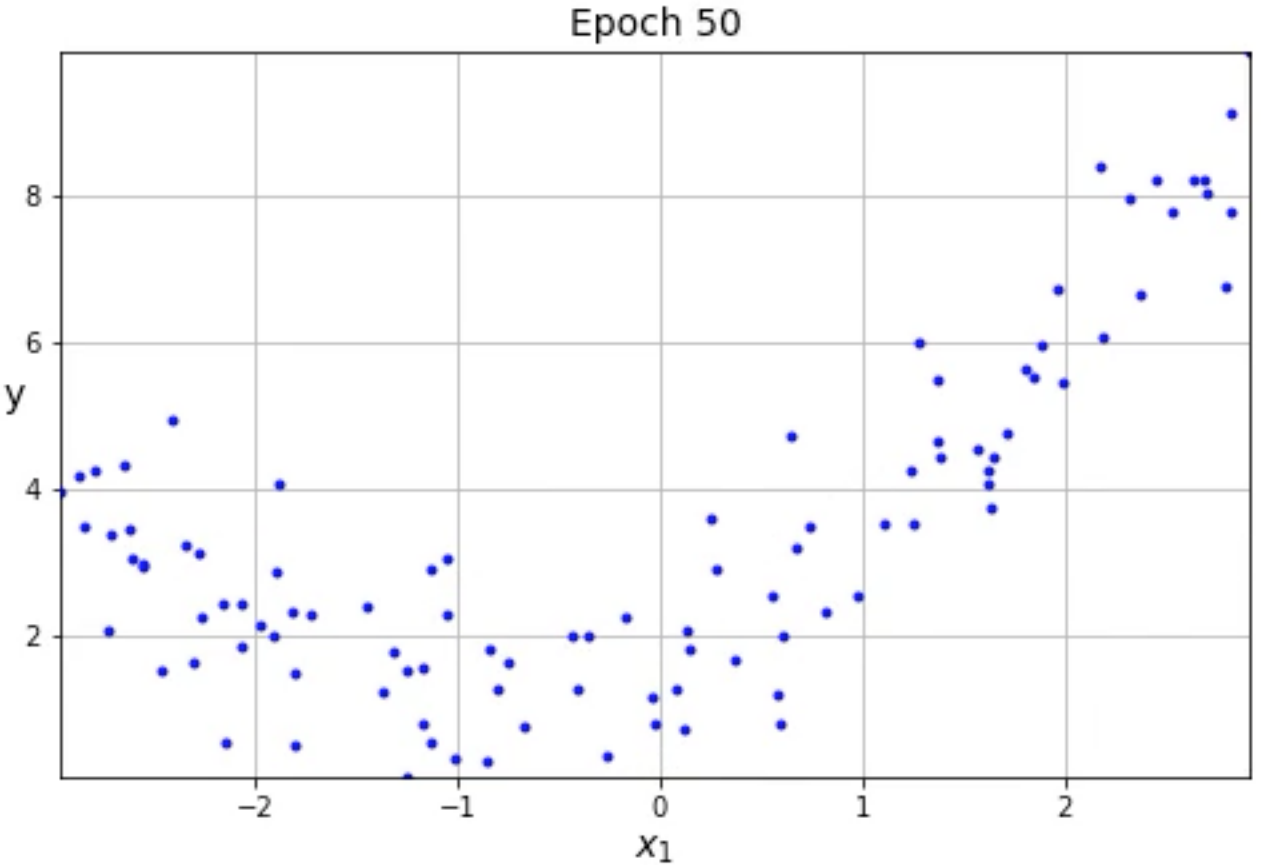
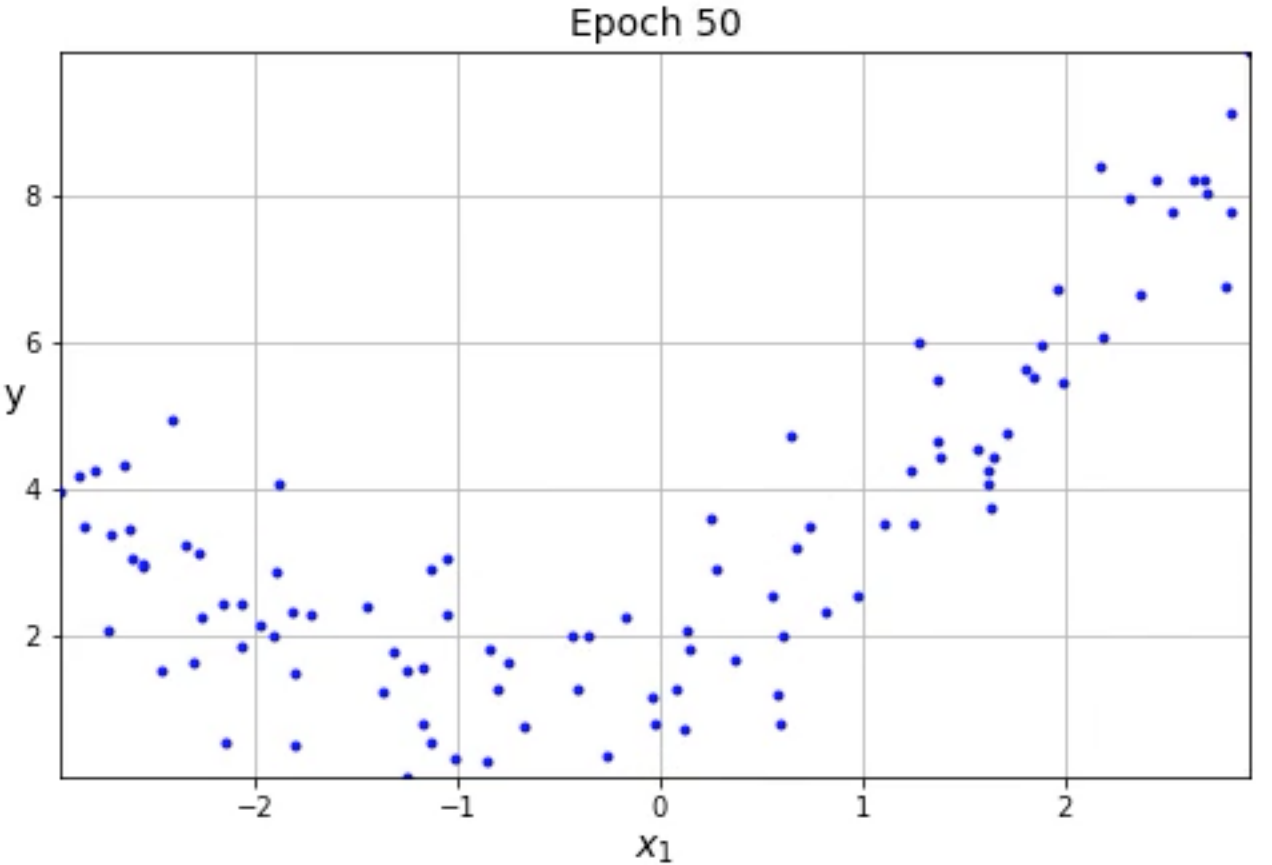
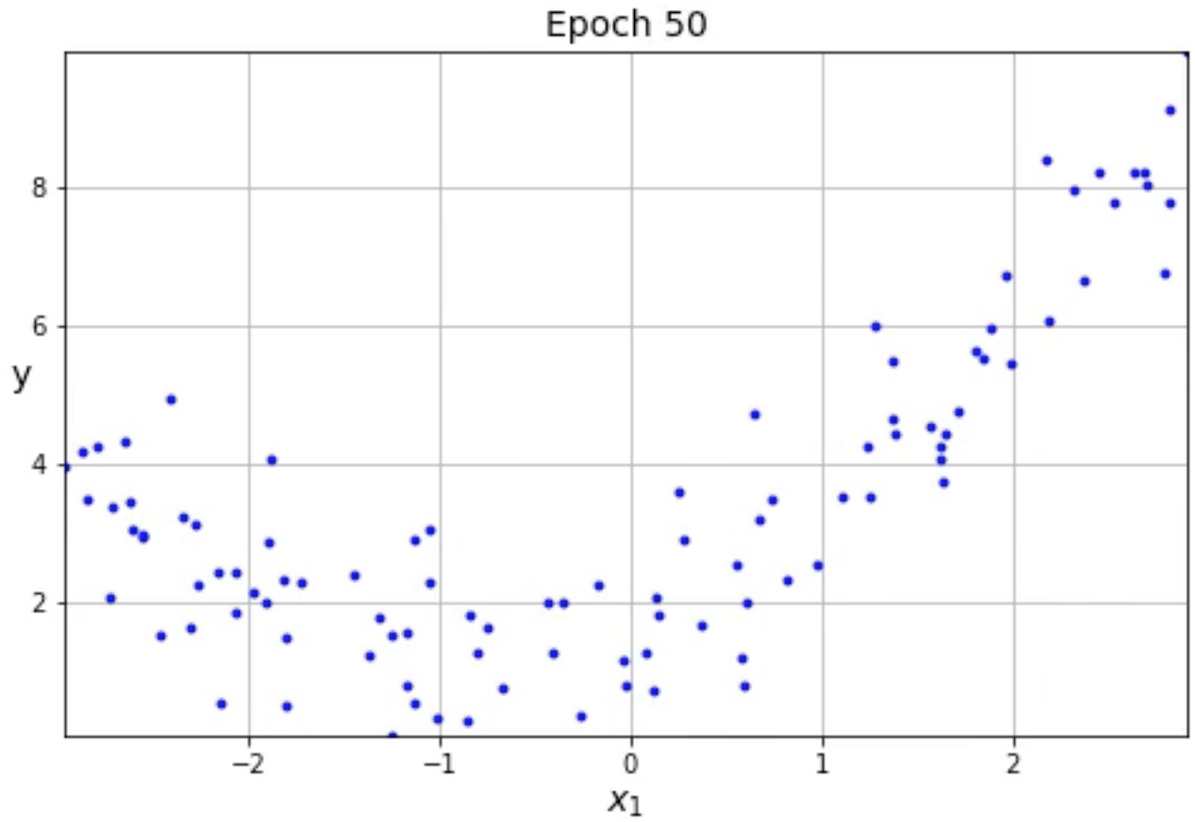
Polynomial Regression


Epoch 50
y
8
6
4
2
−2
−1
0
1
2
x_1

A linear combination of linear functions is a linear function. We need non linearity !


Epoch 50
y
8
6
4
2
−2
−1
0
1
2
x_1

Polynomial Regression



A linear combination of linear functions is a linear function. We need non linearity !

```
def relu(x):
  return np.maximum(0, x)

def reluPrime(x):
  return x > 0

class MLPrelu(MLP):
  def __call__(self, x):
    self.h = relu(np.dot(x, self.w1) + self.b1)
    y_hat = np.dot(self.h, self.w2) + self.b2
    return self.final_activation(y_hat)

  def fit(self, X, Y, epochs = 100, lr = 0.001):
    for e in range(epochs):
     for x, y in zip(X, Y):
x = x[None,:]
y_pred = self(x)
loss = self.loss(y_pred, y).mean()
# Backprop
dldy = self.grad_loss(y_pred, y)
grad_w2 = np.dot(self.h.T, dldy)
grad_b2 = dldy
dldh = np.dot(dldy, self.w2.T)*reluPrime(self.h)
grad_w1 = np.dot(x.T, dldh)
grad_b1 = dldh
# Update (GD)
self.w1 = self.w1 - lr * grad_w1
self.b1 = self.b1 - lr * grad_b1
self.w2 = self.w2 - lr * grad_w2
self.b2 = self.b2 - lr * grad_b2
```