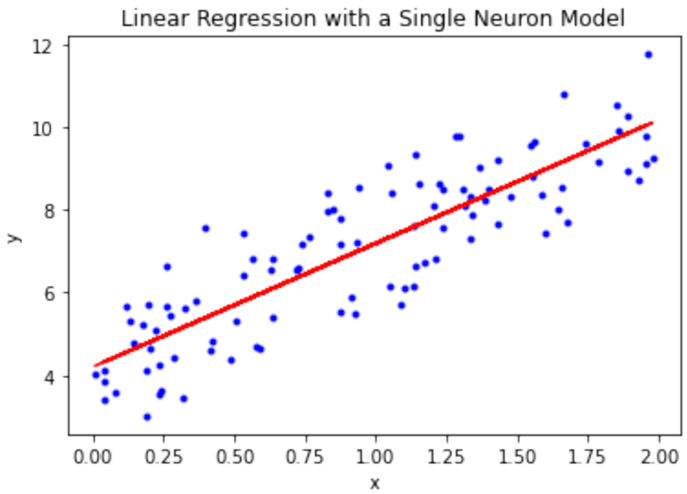
Linear Regression Model with single neuron model

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
# Generate some synthetic data
np.random.seed(0)

X = 2*np.random.rand(100, 1)
y = 4+3*X+np.random.randn(100, 1)
```

In [3]:
```
# Function to perform linear regression using a single neuron model
def linear_regression(X, y, learning_rate = 0.01, epochs = 1000):
    m, n = X.shape
    X_b = np.c_[np.ones((m,1)), X] # add bias term (x0 = 1)
    theta = np.random.randn(n+1, 1) # initialize weights randomly

    for epoch in range(epochs):
        gradients = 2/m * X_b.T.dot(X_b.dot(theta) - y)
        theta -= learning_rate * gradients

    return theta
```

In [4]:
```
# Train the model
theta = linear_regression(X, y)

# Plot the results
plt.plot(X, y, "b.")
plt.plot(X, theta[0]+theta[1]*X, "r-")
plt.xlabel("x")
plt.ylabel("y")
plt.title("Linear Regression with a Single Neuron Model")
plt.show()

# Display the learned parameters
print(f"Intercept (bias): {theta[0][0]}")
print(f"Slope (weight): {theta[1][0]}")
```




```
Intercept (bias): 4.20607718142562
Slope (weight): 2.9827303563323175
```

In [ ]: