Question 1-a

$$\phi(x) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ 1 \end{bmatrix}, \quad w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \end{bmatrix}$$

Question 1-b

$$\phi^T(x_i)\phi(x_j) = x_{i1}^2x_{j1}^2 + x_{i2}^2x_{j2}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2} + 2x_{i1}x_{j1} + 2x_{i2}x_{j2} + 1$$

$$(x_{i1}x_{j1} + x_{i2}x_{j2} + 1)^2 = x_{i1}^2x_{j1}^2 + x_{i2}^2x_{j2}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2} + 2x_{i1}x_{j1} + 2x_{i2}x_{j2} + 1$$

Question 1-c

$\phi^T(x_i)\phi(x_j)$ - 11 multiplications

$(x_i^Tx_j + 1)^2$ - 3 multiplications

Question 2-a

$$(\Phi^T\Phi + \lambda I)^{-1}\Phi^T = \Phi^T(\Phi\Phi^T + \lambda I)^{-1}$$

Question 2-b

We can plug the equivalent expression into $\hat{w}$ to get:

$$\hat{w} = \Phi^T(\Phi\Phi^T + \lambda I)^{-1}y$$

Question 2-c

$$[K]_{i,j} = \phi(x_i)^T\phi(x_j)$$

Question 2-d

$$\phi(x) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ 1 \end{bmatrix}$$

$$[K]_{i,j} = x_{i1}^2 x_{j1}^2 + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2} + 2x_{i1}x_{j1} + 2x_{i2}x_{j2} + 1 = (x_i^T x_j + 1)^2$$

Question 2-e

$$\text{from slides:} \quad \alpha = (\Phi\Phi^T + \lambda I)^{-1} y$$

$$\text{from slides:} \quad [\phi(x)^T \Phi^T]_j = \phi(x)^T \phi(x_j)$$

$$\hat{y}(x) = \phi(x)^T \Phi^T (\Phi\Phi^T + \lambda I)^{-1} y = \phi(x)^T \Phi^T \alpha$$

$$\hat{y}(x) = \sum_{j=1}^{N} K(x, x_j)\alpha_j$$

Question 3-a

$$K(x, x_j) = \phi(x)^T \phi(x_j)$$

$$K(x, x_j) = x^T x_j$$

Question 3-b

$$\hat{y}(x) = \sum_{j=1}^{N} K(x, x_j)\alpha_j = \sum_{j=1}^{N} x^T x_j \alpha_j = x^T \sum_{j=1}^{N} x_j \alpha_j$$

Linear combo of $x_j \alpha_j$

Question 4-a

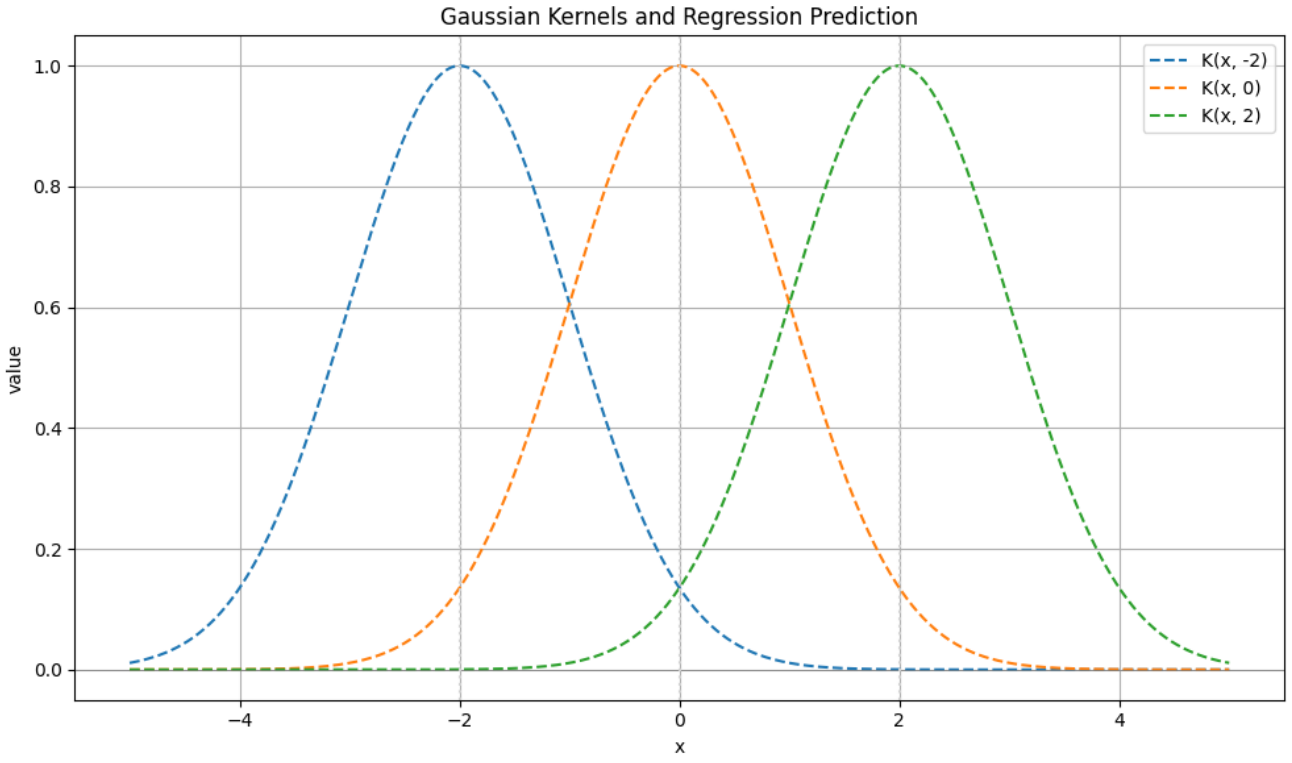
In [2]:
```python
import numpy as np
import matplotlib.pyplot as plt

# Define the Gaussian kernel
def gaussian_kernel(x, xj, sigma=1.0):
    return np.exp(-((x - xj) ** 2) / (2 * sigma ** 2))
```

```
# Setup
x = np.linspace(-5, 5, 400)
x1, x2, x3 = -2, 0, 2

# Compute individual kernel values
K1 = gaussian_kernel(x, x1)
K2 = gaussian_kernel(x, x2)
K3 = gaussian_kernel(x, x3)

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(x, K1, label='K(x, -2)', linestyle='--')
plt.plot(x, K2, label='K(x, 0)', linestyle='--')
plt.plot(x, K3, label='K(x, 2)', linestyle='--')
plt.title("Gaussian Kernels and Regression Prediction")
plt.xlabel("x")
plt.ylabel("value")
plt.axhline(0, color='gray', linewidth=0.5)
plt.axvline(-2, color='lightgray', linestyle=':')
plt.axvline(0, color='lightgray', linestyle=':')
plt.axvline(2, color='lightgray', linestyle=':')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



Question 4-b

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt
```

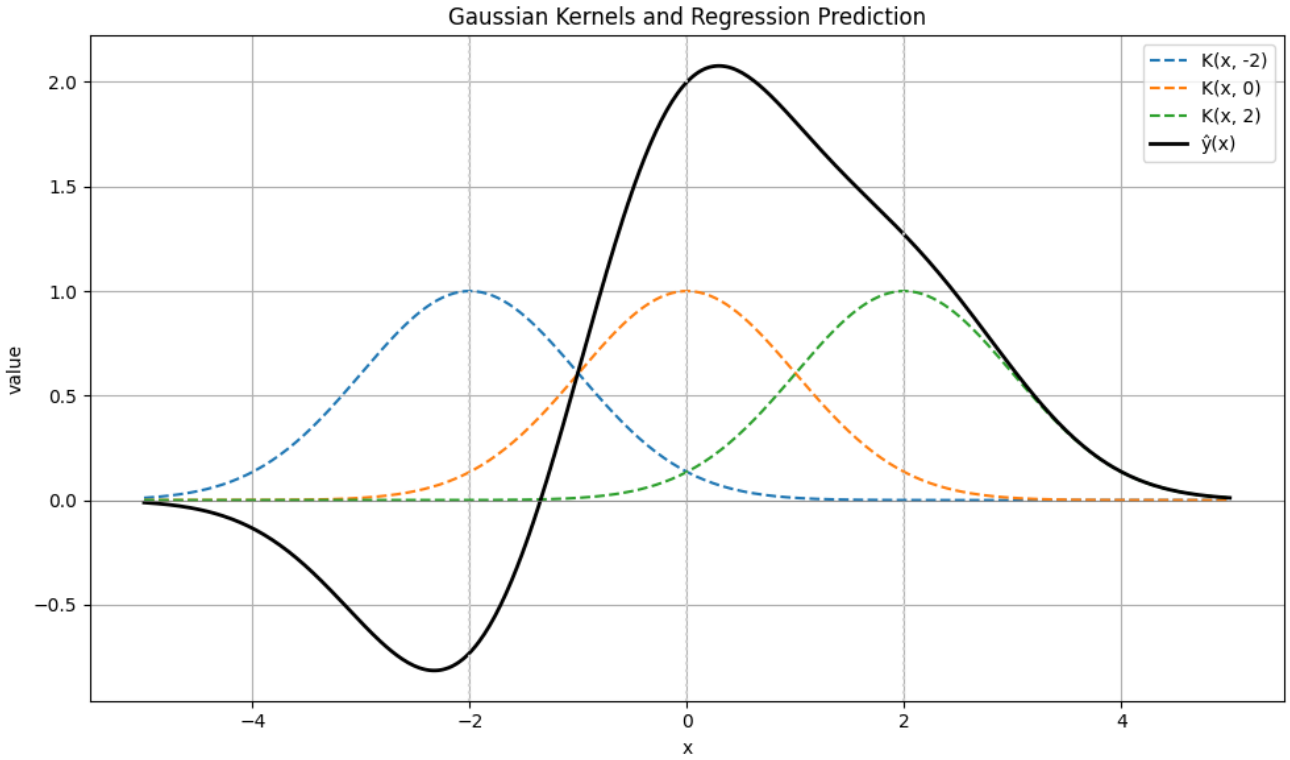
```python
# Define the Gaussian kernel
def gaussian_kernel(x, xj, sigma=1.0):
    return np.exp(-((x - xj) ** 2) / (2 * sigma ** 2))

# Setup
x = np.linspace(-5, 5, 400)
x1, x2, x3 = -2, 0, 2
alpha1, alpha2, alpha3 = -1, 2, 1

# Compute individual kernel values
K1 = gaussian_kernel(x, x1)
K2 = gaussian_kernel(x, x2)
K3 = gaussian_kernel(x, x3)

# Compute regression output
y_hat = alpha1 * K1 + alpha2 * K2 + alpha3 * K3

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(x, K1, label='K(x, -2)', linestyle='--')
plt.plot(x, K2, label='K(x, 0)', linestyle='--')
plt.plot(x, K3, label='K(x, 2)', linestyle='--')
plt.plot(x, y_hat, label='ŷ(x)', linewidth=2, color='black')
plt.title("Gaussian Kernels and Regression Prediction")
plt.xlabel("x")
plt.ylabel("value")
plt.axhline(0, color='gray', linewidth=0.5)
plt.axvline(-2, color='lightgray', linestyle=':')
plt.axvline(0, color='lightgray', linestyle=':')
plt.axvline(2, color='lightgray', linestyle=':')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



Question 4-c

weighted

kernal

training samples