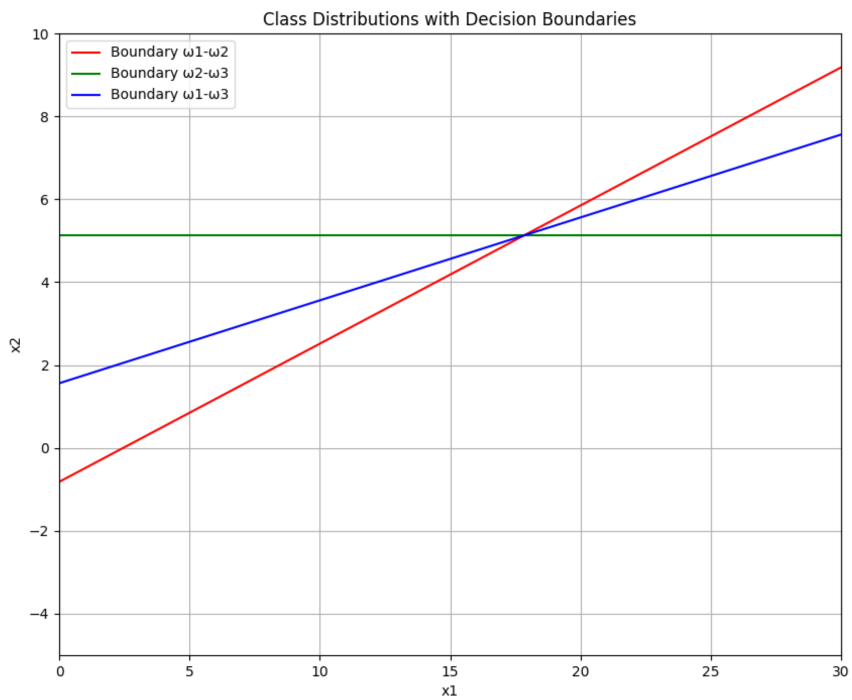
Άσκηση 1.3

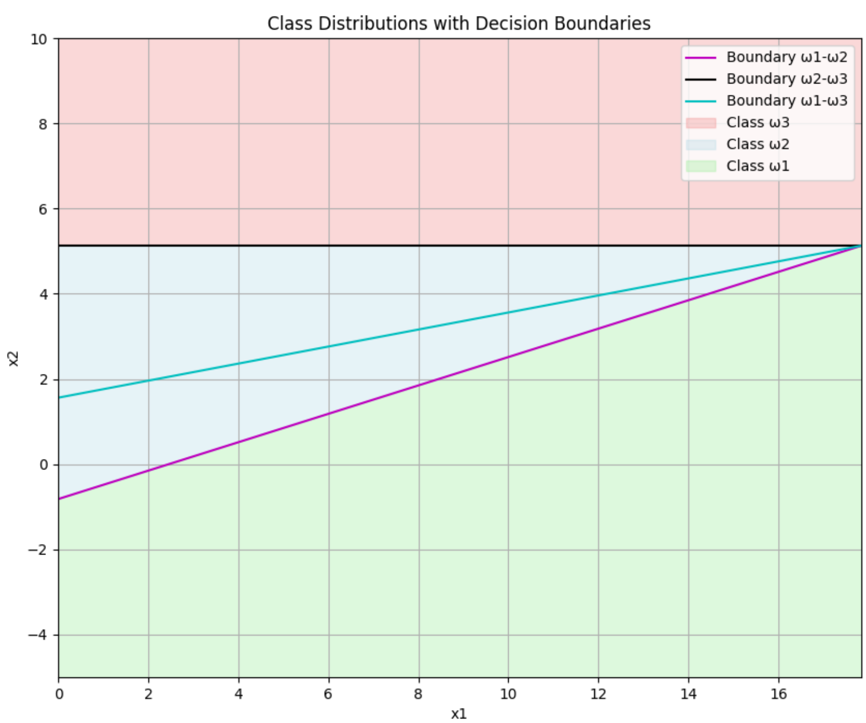
(α) Python Code για το υπολογισμό πιθανοτήτων

```
import numpy as np

from scipy.stats import multivariate_normal

# Given data
mu1 = np.array([3, 2])
mu2 = np.array([4, 3])
mu3 = np.array([6, 7])
cov_matrix = np.array([[1, 0.5], [0.5, 1]])
priors = np.array([0.2, 0.5, 0.3])
x = np.array([3, 5])

# Calculate likelihoods
p_x_given_w1 = multivariate_normal.pdf(x, mean=mu1, cov=cov_matrix)
p_x_given_w2 = multivariate_normal.pdf(x, mean=mu2, cov=cov_matrix)
p_x_given_w3 = multivariate_normal.pdf(x, mean=mu3, cov=cov_matrix)

# Calculate posteriors using Bayes' rule
evidence = p_x_given_w1 * priors[0] + p_x_given_w2 * priors[1] +
p_x_given_w3 * priors[2]
p_w1_given_x = (p_x_given_w1 * priors[0]) / evidence
p_w2_given_x = (p_x_given_w2 * priors[1]) / evidence
p_w3_given_x = (p_x_given_w3 * priors[2]) / evidence

print("Posterior probabilities:")
print(f"P(ω1 | x) = {p_w1_given_x:.4f}")
print(f"P(ω2 | x) = {p_w2_given_x:.4f}")
print(f"P(ω3 | x) = {p_w3_given_x:.4f}")
```

```
Output:

Posterior probabilities:
P(ω1 | x) = 0.0618
P(ω2 | x) = 0.5864
P(ω3 | x) = 0.3518
```

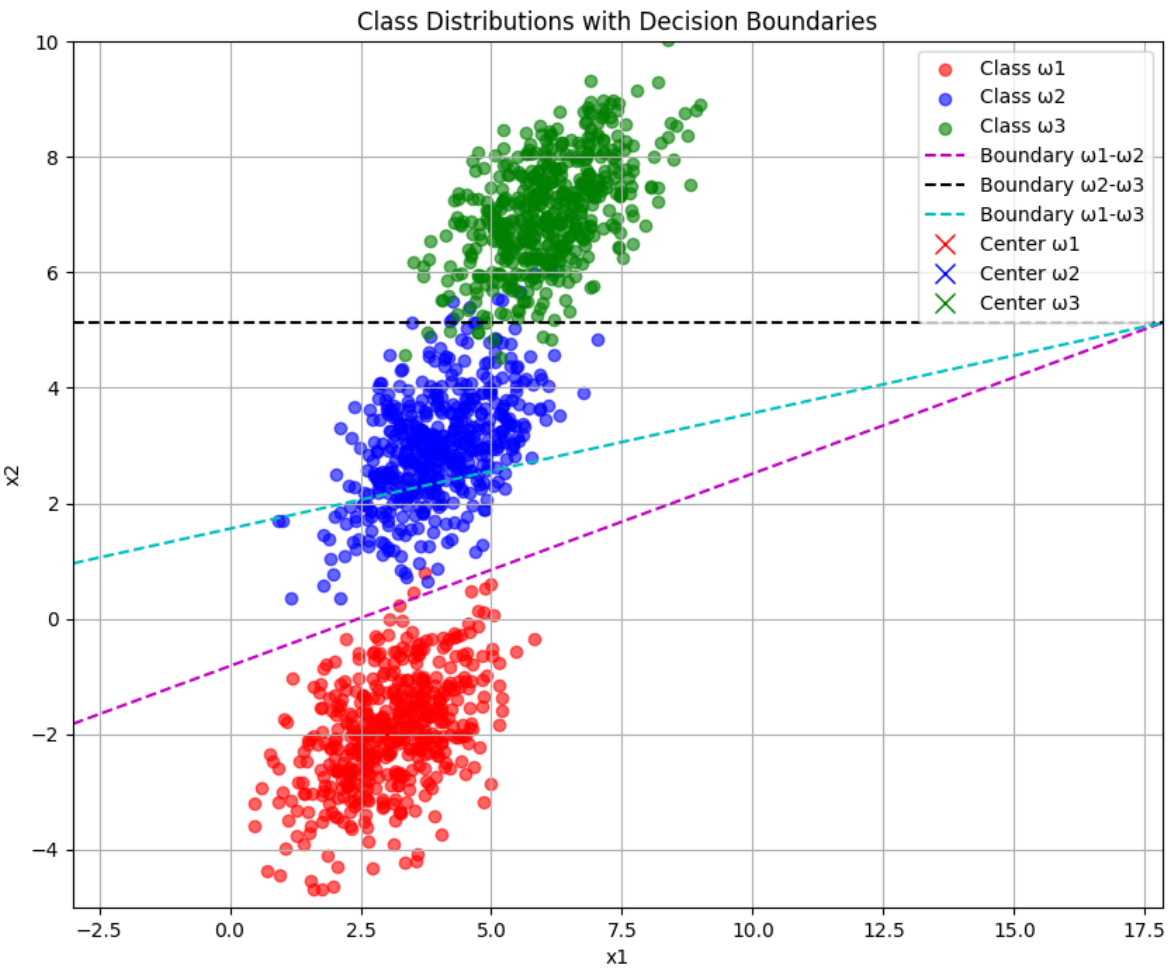
(γ) Παρακάτω φαίνονται οι καμπύλες απόφασης που βρήκαμε.





```
import numpy as np
import matplotlib.pyplot as plt
x_vals = np.linspace(0, 30, 1000)
# Boundary line between ω1 and ω2: 3x1 - 9x2 = 7.3744
boundary12_y = (3 * x_vals - 7.3744) / 9
# Boundary line between ω2 and ω3: x2 = 5.1277 (horizontal line)
boundary23_y = np.full_like(x_vals, 5.1277)
# Boundary line between ω1 and ω3: -3x1 + 15x2 = 23.3918
boundary13_y = (3 * x_vals + 23.3918) / 15
# Plotting
plt.figure(figsize=(10, 8))
# Plot decision boundaries
plt.plot(x_vals, boundary12_y, 'r', label='Boundary ω1-ω2')
plt.plot(x_vals, boundary23_y, 'g', label='Boundary ω2-ω3')
plt.plot(x_vals, boundary13_y, 'b', label='Boundary ω1-ω3')
# Additional plot settings
plt.xlabel('x1')
plt.ylabel('x2')
plt.title('Class Distributions with Decision Boundaries')
plt.legend()
plt.grid(True)
plt.xlim(0, 30)
plt.ylim(-5, 10)
# Plotting
plt.figure(figsize=(10, 8))
# Plot decision boundaries
plt.plot(x_vals, boundary12_y, 'm', label='Boundary ω1-ω2')
plt.plot(x_vals, boundary23_y, 'k', label='Boundary ω2-ω3')
plt.plot(x_vals, boundary13_y, 'c', label='Boundary ω1-ω3')
# # Fill areas based on classification regions
plt.fill_between(x_vals, boundary23_y, 10, color='lightcoral', alpha=0.3,
label='Class ω3')
plt.fill_between(x_vals, boundary12_y, boundary23_y, where=(boundary12_y
<= boundary23_y),color='lightblue', alpha=0.3, label='Class ω2')
plt.fill_between(x_vals, -10, boundary12_y, color='lightgreen', alpha=0.3,
label='Class ω1'
# Additional plot settings
plt.xlabel('x1')
plt.ylabel('x2')
plt.title('Class Distributions with Decision Boundaries')
plt.legend()
plt.grid(True)
plt.xlim(0, 17.84126)
plt.ylim(-5, 10)
plt.show()
```

(δ) Δίνεται το ζητούμενο σχήμα:



Για το ζητούμενο σχήμα χρησιμοποιήθηκε ο εξής κώδικας:

```
import numpy as np
import matplotlib.pyplot as plt

# Define parameters for each class
mean1 = np.array([3, -2])
mean2 = np.array([4, 3])
mean3 = np.array([6, 7])
cov_matrix = np.array([[1, 0.5], [0.5, 1]])  # Shared covariance matrix
priors = np.array([0.2, 0.5, 0.3])
```

```python
# Generate 500 points for each class
np.random.seed(0)  # For reproducibility
p_x_given_w1 = np.random.multivariate_normal(mean1, cov_matrix, 500)
p_x_given_w2 = np.random.multivariate_normal(mean2, cov_matrix, 500)
p_x_given_w3 = np.random.multivariate_normal(mean3, cov_matrix, 500)

# Define decision boundaries as lines based on given equations
x_vals = np.linspace(-3, 20, 1000)
# Boundary line between ω1 and ω2: 3x1 - 9x2 = 7.3744
boundary12_y = (3 * x_vals - 7.3744) / 9
# Boundary line between ω2 and ω3: x2 = 5.1277 (horizontal line)
boundary23_y = np.full_like(x_vals, 5.1277)
# Boundary line between ω1 and ω3: -3x1 + 15x2 = 23.3918
boundary13_y = (3 * x_vals + 23.3918) / 15
# Plotting
plt.figure(figsize=(10, 8))
# Scatter plot for each class
plt.scatter(p_x_given_w1[:, 0], p_x_given_w1[:, 1], color='red',
label='Class ω1', alpha=0.6)
plt.scatter(p_x_given_w2[:, 0], p_x_given_w2[:, 1], color='blue',
label='Class ω2', alpha=0.6)
plt.scatter(p_x_given_w3[:, 0], p_x_given_w3[:, 1], color='green',
label='Class ω3', alpha=0.6)

# Plot decision boundaries
plt.plot(x_vals, boundary12_y, 'm--', label='Boundary ω1-ω2')
plt.plot(x_vals, boundary23_y, 'k--', label='Boundary ω2-ω3')
plt.plot(x_vals, boundary13_y, 'c--', label='Boundary ω1-ω3')
# Plot centers of each class
plt.plot(mean1[0], mean1[1], 'ro', marker='x', markersize=10,
label='Center ω1')
plt.plot(mean2[0], mean2[1], 'bo', marker='x', markersize=10,
label='Center ω2')
plt.plot(mean3[0], mean3[1], 'go', marker='x', markersize=10,
label='Center ω3')

# Additional plot settings
plt.xlabel('x1')
plt.ylabel('x2')
plt.title('Class Distributions with Decision Boundaries')
plt.legend()
plt.grid(True)
plt.xlim(-3, 17.84126)
plt.ylim(-5, 10)
plt.show()
```

(ε) Υπολογίστηκε με δύο τρόπους. Αρχικά, πήρα 10000 δείγματα από την κατανομή x | $\omega_2 \sim N(\boldsymbol{\mu_2}, \Sigma)$ και βλέπουμε το ποσοστό που αυτά σύμφωνα με τον ταξινομιτή Bayes γίνονται misclassified.

Αποτέλεσμα: Probability of misclassification of Class w2 is: 0.0191

Κώδικας:

```
import numpy as np
from scipy.stats import multivariate_normal
# Data
mu1 = np.array([3, -2])
mu2 = np.array([4, 3])
mu3 = np.array([6, 7])
sigma = np.array([[1, 0.5], [0.5, 1]])
priors = [0.2, 0.5, 0.3]

# Creating 10000 samples
num_samples = 10000
samples = multivariate_normal.rvs(mean=mu2, cov=sigma, size=num_samples)

# Miclassification initialization
misclassified_count = 0

# Compute posterior probability and classify them with Bayes
for x in samples:
    # Computation of PDF for every distribution
    p_x_given_w1 = multivariate_normal.pdf(x, mean=mu1, cov=sigma) *
priors[0]
    p_x_given_w2 = multivariate_normal.pdf(x, mean=mu2, cov=sigma) *
priors[1]
    p_x_given_w3 = multivariate_normal.pdf(x, mean=mu3, cov=sigma) *
priors[2]

    # Classification with Bayes classifier
    predicted_class = np.argmax([p_x_given_w1, p_x_given_w2,
p_x_given_w3])

    # If misclassification then increase variable
    if predicted_class != 1:
        misclassified_count += 1
# Final Output
error_probability = misclassified_count / num_samples
print("Probability of misclassification of Class w2 is:",
error_probability)
```

Ο δεύτερος τρόπος είναι ο υπολογισμός του ολοκληρώματος $p(x\ |\omega_2)$ στις περιοχές που δεν ανήκει στην κλάση ω_2.

Αποτέλεσμα: Probability of Misclaffication is: 0.019090338054833369

Κώδικας: `import numpy as np`

```
from scipy.stats import multivariate_normal
from scipy.integrate import dblquad
# Given parameters
mean_w2 = [4, 3]
cov_matrix = [[1, 0.5], [0.5, 1]]
# Define the distribution for p(x|w2)
rv_w2 = multivariate_normal(mean=mean_w2, cov=cov_matrix)

# Define the boundaries
boundary12 = lambda x1: (3 * x1 - 7.3744) / 9  # Boundary ω1-ω2
boundary23 = 5.1277  # Boundary ω2-ω3

# Misclassification region for ω3 (x2 > boundary23)
def integrand_w3(x2, x1):
    return rv_w2.pdf([x1, x2])

# Misclassification region for ω1 (x2 < boundary12)
def integrand_w1(x2, x1):
    return rv_w2.pdf([x1, x2])

# Integration limits
x1_min, x1_max = -np.inf, 17.84126
# Compute the probability for ω3 region
P_error_w3, _ = dblquad(
    integrand_w3,
    x1_min, x1_max,
    lambda x1: boundary23,   # Lower bound for x2
    lambda x1: np.inf        # Upper bound for x2
)
# Compute the probability for ω1 region
P_error_w1, _ = dblquad(
    integrand_w1,
    x1_min, x1_max,
    lambda x1: -np.inf,             # Lower bound for x2
    lambda x1: boundary12(x1)       # Upper bound for x2
)
# Total misclassification probability
P_misclassification = P_error_w1 + P_error_w3
print(f"Probability of Misclassification is: {P_misclassification}")
```