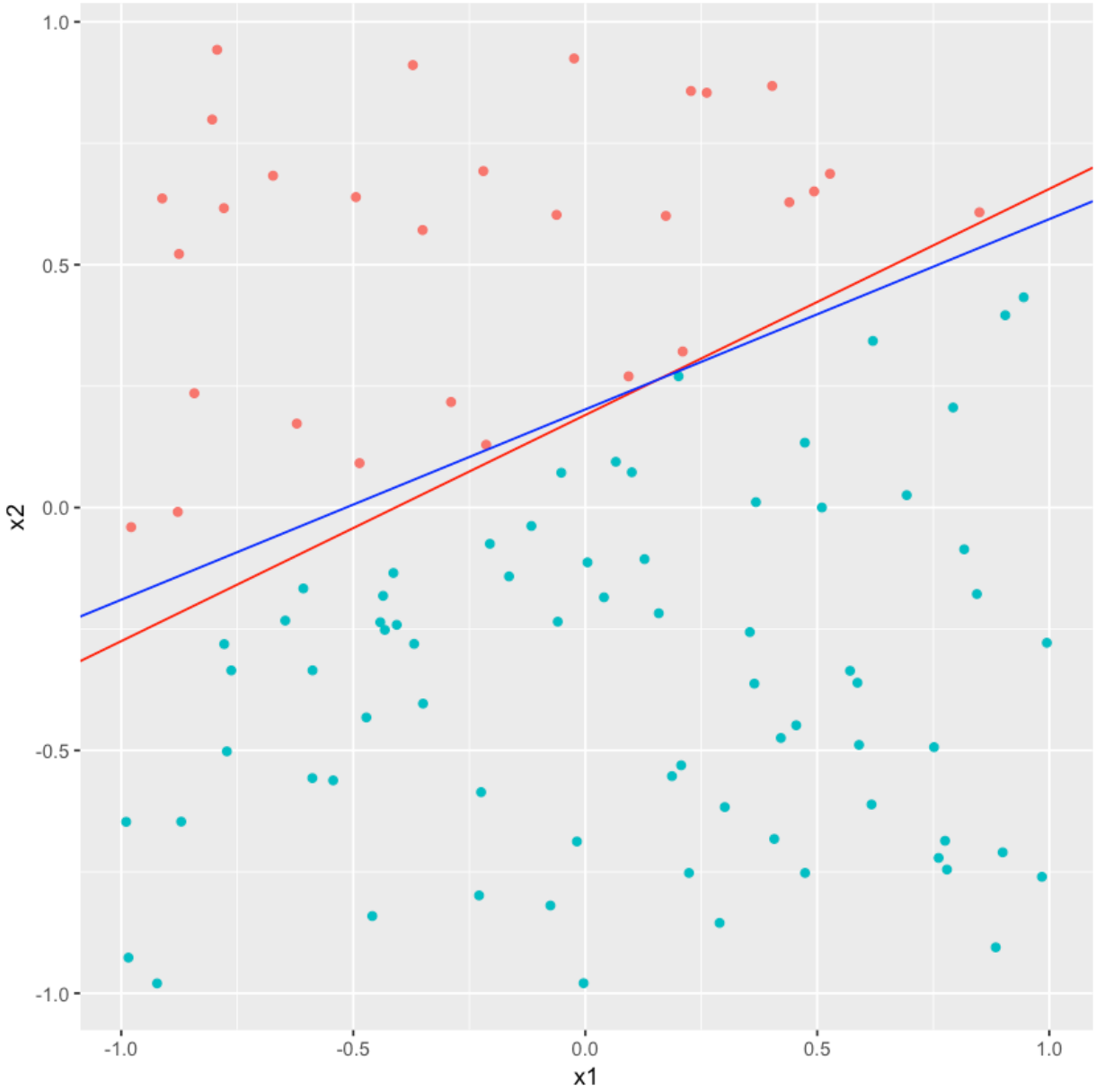
In [ ]:
```
(a) Below, we generate a training data set of size 100 and a test data set c
We also plot the target function f (in red) and the final hypothesis g (in b
value of 5 instead of 100 to simplify the values computed].
```

In [3]:
```
library(ggplot2)

set.seed(1975)
h <- function(x, w) {
scalar_prod <- cbind(1, x$x1, x$x2) %*% w
    return(as.vector(sign(scalar_prod)))
}
w0 <- runif(1, min = -999, max = 999)
w1 <- runif(1, min = -999, max = 999)
w2 <- runif(1, min = -999, max = 999)
f <- function(x) { return(h(x, c(w0, w1, w2)))
}
D_train <- data.frame(x1 = runif(100, min = -1, max = 1), x2 = runif(100, mi
D_train <- cbind(D_train, y = f(D_train))
D_test <- data.frame(x1 = runif(10000, min = -1, max = 1), x2 = runif(10000,
D_test <- cbind(D_test, y = f(D_test))
iter <- 0
eta <- 5
w <- c(0, 0, 0)
 repeat {
y_pred <- h(D_train, w)
D_mis <- subset(D_train, y != y_pred)
     if (nrow(D_mis) == 0)
break
obs_t <- D_mis[sample(nrow(D_mis), 1), ]
     x_t <- c(1, as.numeric(obs_t[1:2]))
y_t <- as.numeric(obs_t[3])
s_t <- sum(w * x_t)
if (y_t * s_t <= 1)
w <- w + eta * (y_t - s_t) * x_t
     iter <- iter + 1
if (iter == 1000)
break
}
test_error <- mean(h(D_test, w) != D_test$y)
p <- ggplot(D_train, aes(x = x1, y = x2, col = as.factor(y + 3))) + geom_poi
p_g <- p + geom_abline(slope = -w1 / w2, intercept = -w0 / w2, colour = "red
p_g
test_error
```

0.0223



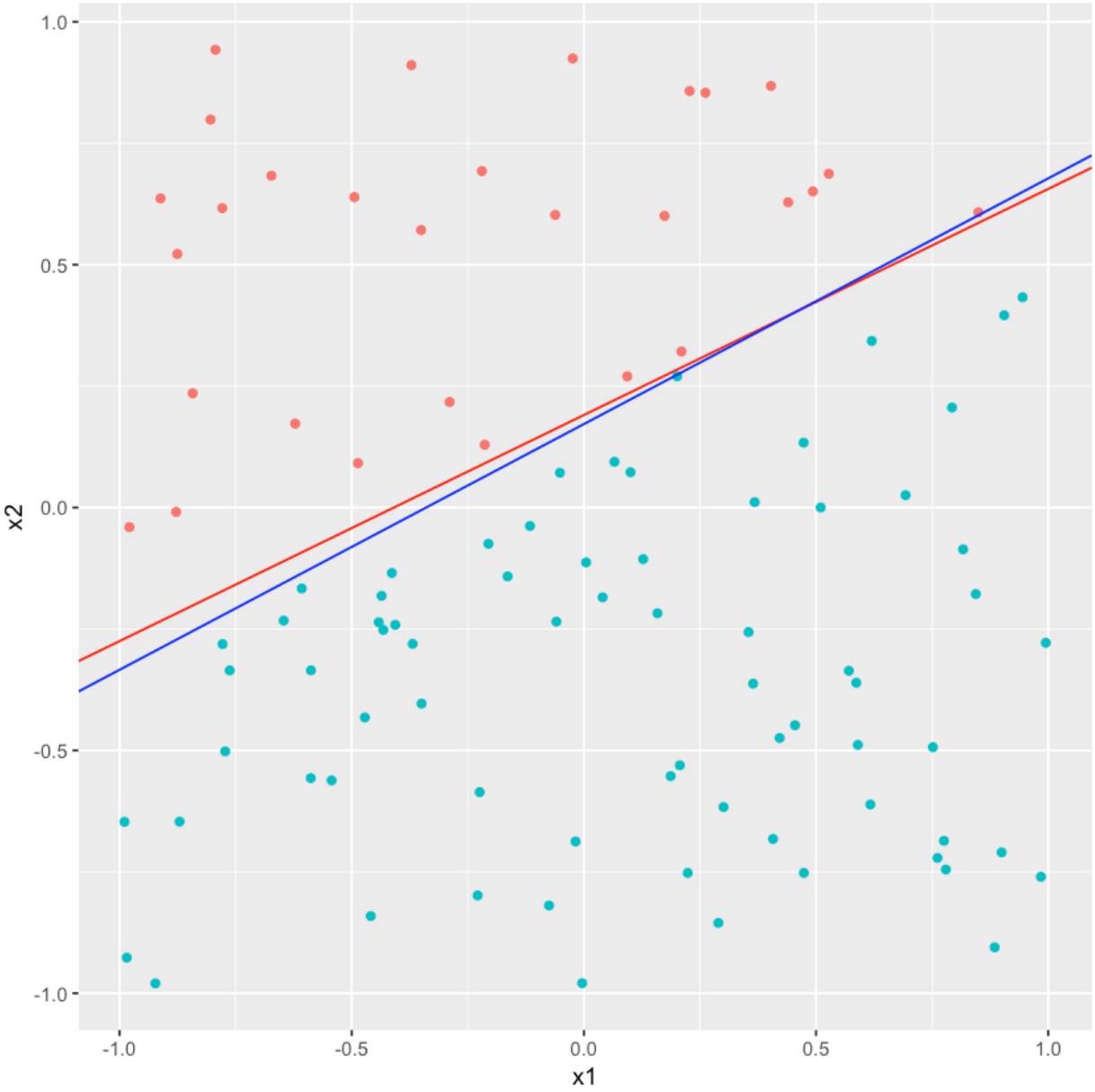
In [4]:
```
We have a classification error rate of 2.23% on the test set
```

```
Error in parse(text = x, srcfile = src): <text>:1:4: unexpected symbol
1: We have
       ^
Traceback:
```

In [ ]:
```
(b) Now we repeat everything we did in (a) with η = 1.
```

In [5]:
```
iter <- 0
eta <- 1
w <- c(0, 0, 0)
repeat {
y_pred <- h(D_train, w)
D_mis <- subset(D_train, y != y_pred)
    if (nrow(D_mis) == 0)
break
obs_t <- D_mis[sample(nrow(D_mis), 1), ]
    x_t <- c(1, as.numeric(obs_t[1:2]))
y_t <- as.numeric(obs_t[3])
s_t <- sum(w * x_t)
if (y_t * s_t <= 1)
w <- w + eta * (y_t - s_t) * x_t
    iter <- iter + 1
if (iter == 1000)
break
}
test_error <- mean(h(D_test, w) != D_test$y)
p <- ggplot(D_train, aes(x = x1, y = x2, col = as.factor(y + 3))) + geom_poi
p_g <- p + geom_abline(slope = -w1 / w2, intercept = -w0 / w2, colour = "red
p_g
test_error
```

0.0123

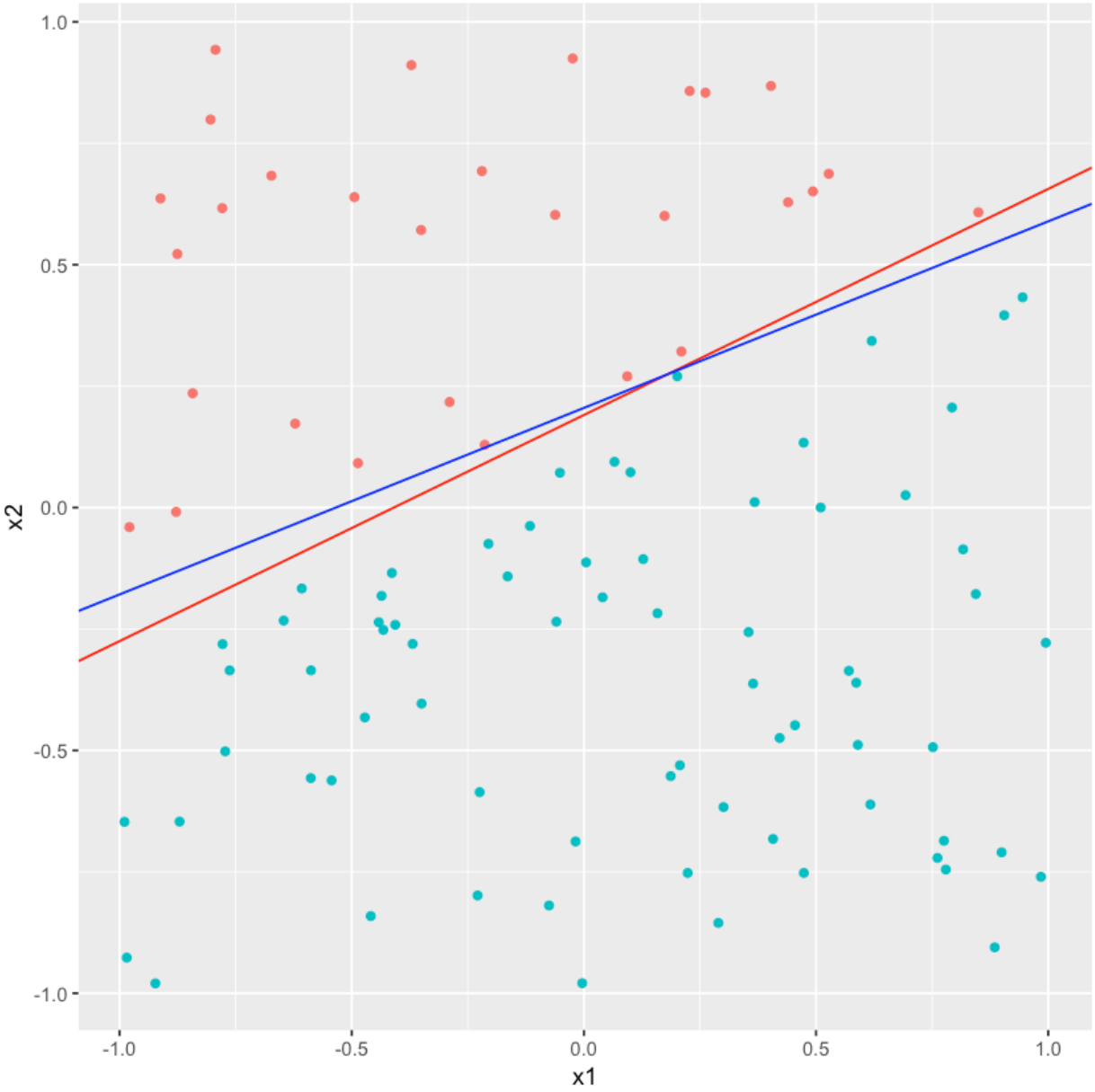


In [ ]:
```
We may see that the classification error rate has now decreased to 1.23% on
```

In [ ]:
```
(c) Now we repeat everything we did in (a) with η = 0.01
```

In [6]:
```
iter <- 0
eta <- 0.01
w <- c(0, 0, 0)
repeat {
y_pred <- h(D_train, w)
D_mis <- subset(D_train, y != y_pred)
    if (nrow(D_mis) == 0)
break
obs_t <- D_mis[sample(nrow(D_mis), 1), ]
    x_t <- c(1, as.numeric(obs_t[1:2]))
y_t <- as.numeric(obs_t[3])
s_t <- sum(w * x_t)
if (y_t * s_t <= 1)
w <- w + eta * (y_t - s_t) * x_t
    iter <- iter + 1
if (iter == 1000)
break
}
test_error <- mean(h(D_test, w) != D_test$y)
p <- ggplot(D_train, aes(x = x1, y = x2, col = as.factor(y + 3))) + geom_poi
 p_g <- p + geom_abline(slope = -w1 / w2, intercept = -w0 / w2, colour = "re
p_g
test_error
```

0.0243

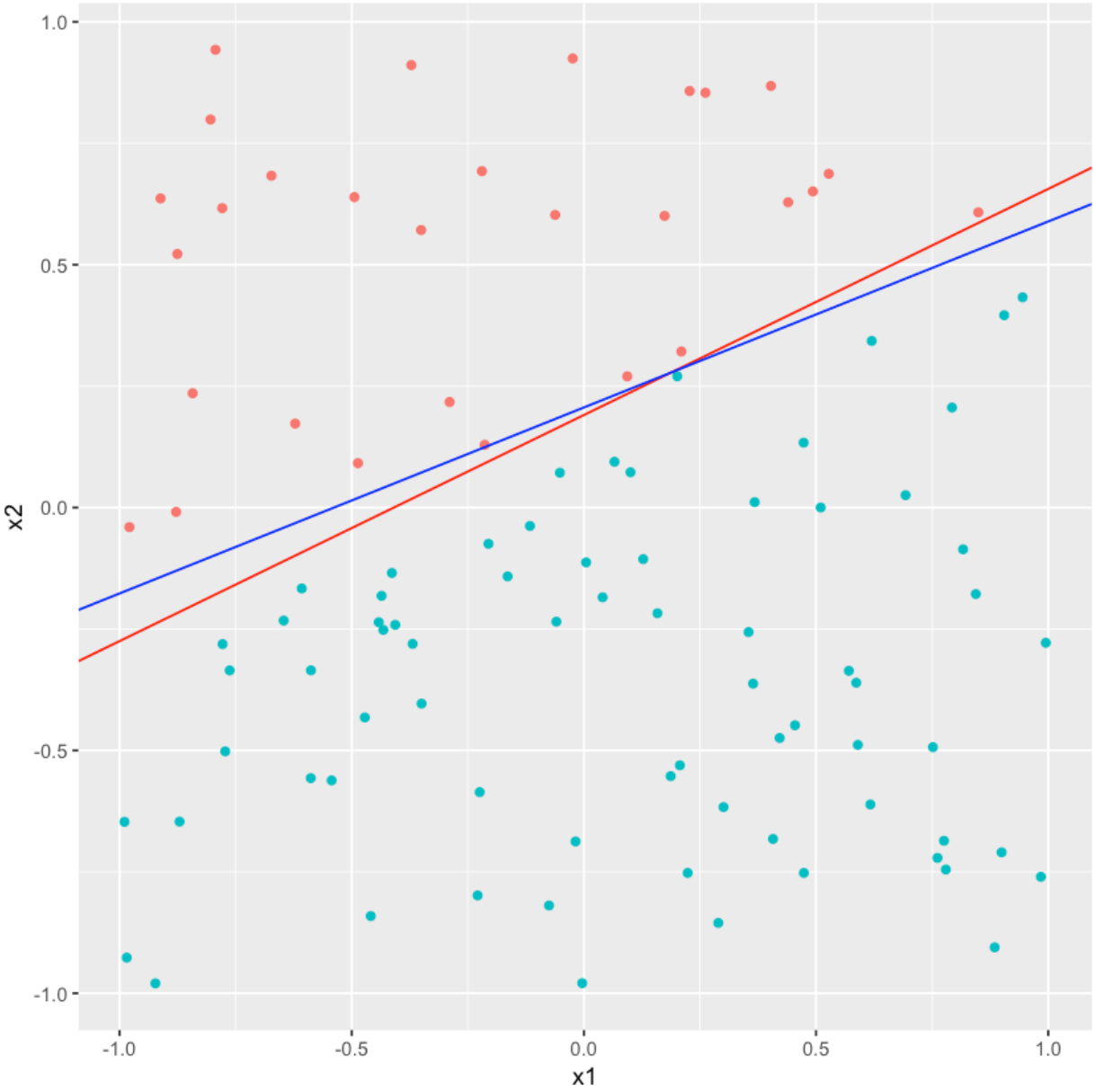


In [ ]:
```
We may see that the classification error rate has now increased to 2.43% on
```

In [ ]:
```
(d) Now we repeat everything we did in (a) with η = 0.0001.
```

In [7]:
```
iter <- 0
eta <- 0.0001
w <- c(0, 0, 0)
repeat {
y_pred <- h(D_train, w)
D_mis <- subset(D_train, y != y_pred)
    if (nrow(D_mis) == 0)
break
obs_t <- D_mis[sample(nrow(D_mis), 1), ]
    x_t <- c(1, as.numeric(obs_t[1:2]))
y_t <- as.numeric(obs_t[3])
s_t <- sum(w * x_t)
if (y_t * s_t <= 1)
w <- w + eta * (y_t - s_t) * x_t
    iter <- iter + 1
if (iter == 1000)
break
}
test_error <- mean(h(D_test, w) != D_test$y)
p <- ggplot(D_train, aes(x = x1, y = x2, col = as.factor(y + 3))) + geom_poi
p_g <- p + geom_abline(slope = -w1 / w2, intercept = -w0 / w2, colour = "red
p_g
test_error
```

0.025



In [ ]:
```
We may see that the classification error rate has now increased to 2.5% on t
```

In [ ]:
```
(e) We may conclude that the η value that results in the minimum classificat
```