knitr::opts\_chunk$set(echo = TRUE, warning = FALSE, message = FALSE)  
library(caret)  
library(doParallel)  
registerDoParallel(cores = detectCores() - 1)   
  
my\_data <- read.csv("data/raceData.csv", header = TRUE, stringsAsFactors = TRUE)  
  
set.seed(1337)  
  
index <- createDataPartition(my\_data$Finish.Position, p = .6, list = FALSE)  
  
training\_data <- my\_data[index, ]  
validation\_data <- my\_data[-index, ]  
  
all\_levels <- union(levels(training\_data$Genotype), levels(validation\_data$Genotype))  
training\_data$Genotype <- factor(training\_data$Genotype, levels = all\_levels)  
validation\_data$Genotype <- factor(validation\_data$Genotype, levels = all\_levels)  
  
#   
# # This should be fixing this error   
# # > predictions <- predict(model\_lm, newdata = validation\_data)  
# # Error in model.frame.default(Terms, newdata, na.action = na.action, xlev = object$xlevels) :   
# # factor Genotype has new levels Z111, Z128, Z148, Z153, Z195  
  
  
# Model 1  
  
model\_lm <- lm(Finish.Position ~ ., data = training\_data)  
  
print(model\_lm)  
  
predictions <- predict(model\_lm, newdata = validation\_data)  
  
actual <- validation\_data$Finish.Position  
test\_r\_squared <- cor(actual, predictions)^2  
rmse\_test <- sqrt(mean((predictions - actual)^2))  
cat("Test Set R-squared:", test\_r\_squared, "\n")  
cat("Test Set RMSE:", rmse\_test, "\n")  
  
residuals <- validation\_data$Finish.Position - predictions  
  
results <- data.frame(  
 Actual = validation\_data$Finish.Position,  
 Predicted = predictions,  
 Residual = residuals  
)  
  
correct\_threshold <- 1  
  
results$Correctly\_Predicted <- abs(results$Residual) <= correct\_threshold  
  
library(ggplot2)  
ggplot(results, aes(x = Actual, y = Predicted)) +  
 geom\_point() +  
 geom\_abline(intercept = 0, slope = 1, color = "red") +  
 labs(title = "Actual vs. Predicted Finish Positions", x = "Actual Finish Position", y = "Predicted Finish Position") +  
 xlim(0, 12) + ylim(0, 12) # Set the x and y-axis limits to 0-12  
  
# Model 2   
  
train\_control <- trainControl(method = "cv", number = 10, verboseIter = TRUE)  
  
stepped\_model <- step(model\_lm, direction="both", trace=FALSE)  
  
system.time({  
 model\_lm\_step <- train(stepped\_model, data=training\_data, method="lm", trControl=train\_control)  
})  
  
stopImplicitCluster()  
  
print(model\_lm\_step)  
  
predictions\_stepped <- predict(stepped\_model, newdata = validation\_data)  
actual\_stepped <- validation\_data$Finish.Position  
test\_r\_squared\_stepped <- cor(actual\_stepped, predictions\_stepped)^2  
rmse\_test\_stepped <- sqrt(mean((predictions\_stepped - actual)^2))  
cat("Test Set R-squared:", test\_r\_squared\_stepped, "\n")  
cat("Test Set RMSE:", rmse\_test\_stepped, "\n")  
  
residuals <- validation\_data$Finish.Position - predictions\_stepped  
  
results\_stepped <- data.frame(  
 Actual = validation\_data$Finish.Position,  
 Predicted = predictions,  
 Residual = residuals  
)  
  
results\_stepped$Correctly\_Predicted <- abs(results\_stepped$Residual) <= correct\_threshold  
  
ggplot(results\_stepped, aes(x = Actual, y = Predicted)) +  
 geom\_point() +  
 geom\_abline(intercept = 0, slope = 1, color = "red") +  
 labs(title = "Actual vs. Predicted Finish Positions", x = "Actual Finish Position", y = "Predicted Finish Position") +  
 xlim(0, 12) + ylim(0, 12) # Set the x and y-axis limits to 0-12  
  
  
# Cross Validation   
  
model\_lm\_cv <- train(Finish.Position ~ ., data = training\_data, method = "lm", trControl = train\_control)  
  
  
predictions <- predict(model\_lm\_cv, newdata = validation\_data)  
  
actual <- validation\_data$Finish.Position  
test\_r\_squared <- cor(actual, predictions)^2  
rmse\_test <- sqrt(mean((predictions - actual)^2))  
cat("Test Set R-squared:", test\_r\_squared, "\n")  
cat("Test Set RMSE:", rmse\_test, "\n")