Why do we need a test data?

MSEs calculated in the validation step can be biased when many models are compared

An example on R. We believe that two genes work together to increase the level of a carcinogenic protein. We have a dataset where we measured, for 200 people, the level of the carcinogenic protein (Y) and 1,000 genes (V1, V2...).

Our goal is to find those two genes that work together.

Step 1. Create **random data**

```
# Create data of random data, 100 observations, 1000 predictors. Mean = 0, STD = 1.
data <- as_tibble(matrix(rnorm(100*1000),nrow=100))
# Names of the variables (by default V1, V2...)
X_vars <- names(data)
# Adding the level of the carcinogenic protein (also random)
data$Y <- rnorm(100)
# Split into train, validation and test
data_train <- data[1:50, ]
data_val   <- data[51:75, ]
data_test  <- data[76:100, ]
```

Step 2. Generate all possible formulas (combinations of two genes)

```
# This uses a external function that we will use in the next lab
formulas <- generate_formulas(2, X_vars, "Y")
print(paste("Total models to compare = (1000*999)/2:", length(formulas)))
```

```
## [1] "Total models to compare = (1000*999)/2: 499500"
```

```
print(formulas[1:10])
```

```
##  [1] "Y ~ V1 + V2"  "Y ~ V1 + V3"  "Y ~ V1 + V4"  "Y ~ V1 + V5"  "Y ~ V1 + V6"
##  [6] "Y ~ V1 + V7"  "Y ~ V1 + V8"  "Y ~ V1 + V9"  "Y ~ V1 + V10" "Y ~ V1 + V11"
```

Step 3. Find the best model, **fitting the model in the training dataset and obtaining the R^2 on the validation dataset**

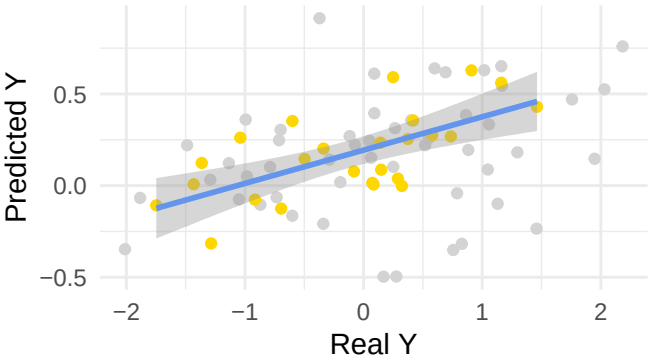
```
fit_validate <- function(formula) {
  # Train on the training dataset
  m1 <- lm(as.formula(formula), data = data_train)
  # Only consider observations when the r^2 in the training dataset is over 10% (correlation over 31%)
  if (summary(m1)$r.squared < 0.1) {
    return(0)
  }
  # Predict values of the validation dataset and calculates R^2
  y_pred <- predict(m1, newdata = data_val)
  return(cor(data_val$Y, y_pred)^2)
}

# Loop over formulas, train model and calculate R^2 on validation dataset
# (this could be made more efficient with sapply(formulas, fit_validate))
corrs <- rep(0, length(formulas))
for (j in seq(1, length(formulas))) {
  formula <- formulas[j]
  corrs[j] <- fit_validate(formula)
}
```

What is the best model?

We use the validation dataset to find the best model (the one with the highest R^2)

```
# Get and print formula and r^2 of the best model
max_corr <- corrs[which.max(corrs)]
best_formula <- formulas[which.max(corrs)]
print(max_corr)
```

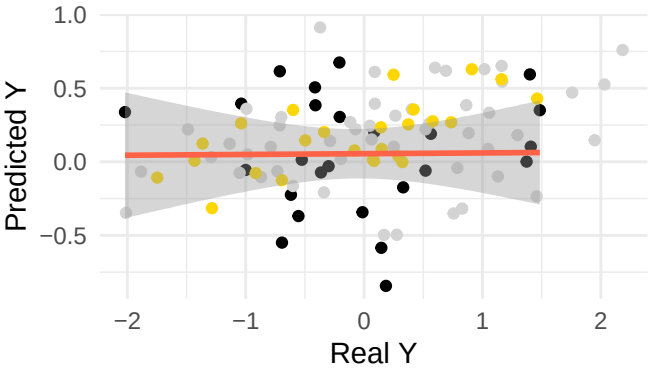
```
## Y ~ V304 + V863
##       0.4293677
```



We can calculate an unbiased MSE using the test dataset

```
# Fit the model in the train+validation dataset (increase observation)
m_test <- lm(as.formula(best_formula), data = bind_rows(data_train, data_val))
# Predict ys in the test dataset and calculate r^2
y_pred <- predict(m_test, newdata = data_test)
cor(data_test$Y, y_pred)^2
```

```
## [1] 0.000126642
```



Question: Why is the MSE from the validation dataset so high? Would selecting the best model using cross-validation have helped reduce this problem? Would having more data help reduce this problem?

We selected that model *because* it had a high R^2. This creates a bias (the model has already seen the data indirectly).