Exercise A.2 for Predictive Modeling

Pablo Vidal Fernández 100483812
José Ignacio Díez Ruiz 100487766
Carlos Roldán Piñero 100484904

2023-01-08

Exercise A.2

For the challenger.txt dataset, do the following:

a. Do a Poisson regression of the total number of incidents, *nfails.field* + *nfails.nozzle*, on *temp*. Interpret the regression. Are the effects of temp significant with $\alpha = 0.01$?

We have to define our dependent variable as the sum of the two indicated variables. Then, we can do the regression:

```
load("10.RData")
challenger$total_fails <- challenger$nfails.field + challenger$nfails.nozzle
fit <- glm(total_fails ~ temp, family = poisson, data = challenger)
```

	Estimate	CI (lower)	CI (upper)	Std. Error	z value	Pr(>|z|)	
(Intercept)	2.9438625	1.1530300	4.5697837	0.8665331	3.397288	0.001	***
temp	-0.1432049	-0.2328168	-0.0523386	0.0457475	-3.130333	0.002	**

The interpretations of the coefficients are the following:

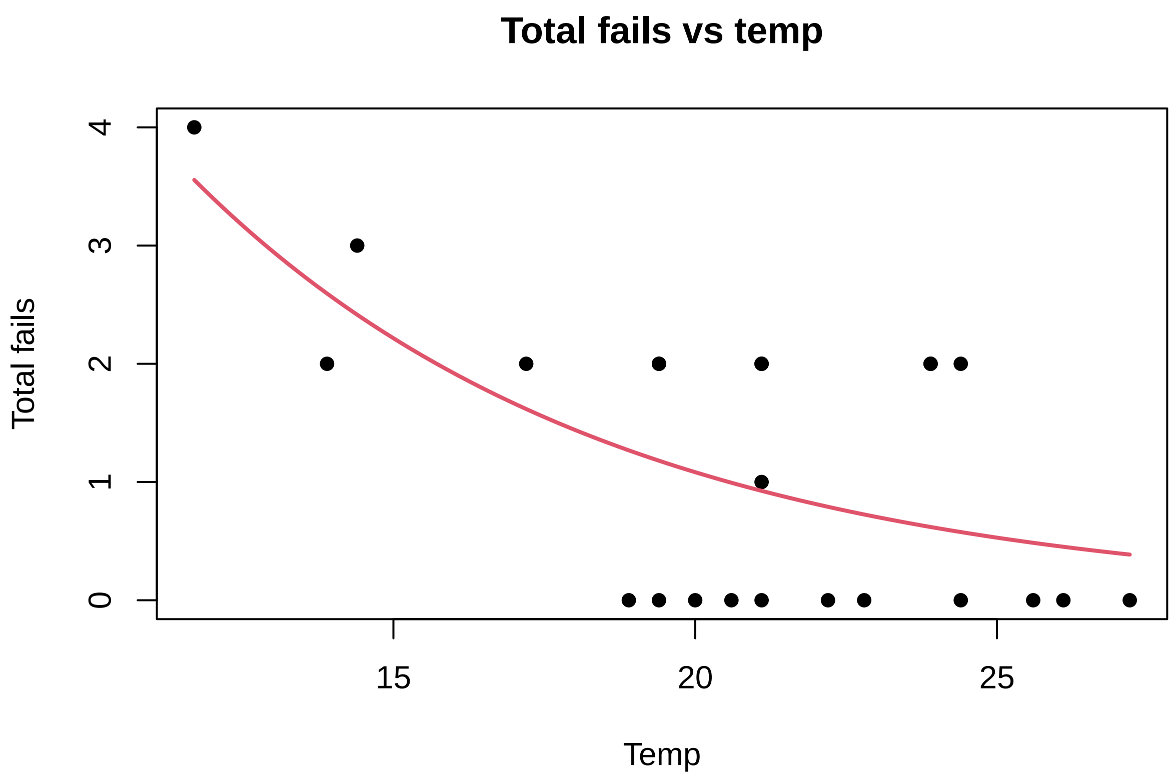
- $e^{\hat{\beta}_0} = e^{2.943} = 18.91$ is the expected number of total fails when *temp* is equal to 0.
- $e^{\hat{\beta}_1} = e^{-0.143} = 0.866$ is the the factor by which the expected number of total fails is going to be multiplied when there is an unit change in *temp* (13.4% reduction).

We can see that the effects of *temp* are significant with $\alpha = 0.01$, as the p-value is smaller and thus we can reject the null hypothesis of the coefficient being equal to 0.

b. Plot the data and the fitted Poisson regression curve.

We create of sequence of 100 points evenly distributed between the minimum and the maximum observations of *temp*, and predict the expected number of total fails for each of them.

```
x <- challenger$temp
y <- challenger$total_fails

plot(x, y, main = "Total fails vs temp", xlab = "Temp",
     ylab = "Total fails", pch = 16)

pred <- predict(fit, data.frame(temp = seq(min(x), max(x),
                    length.out = 100)), type = "response")

lines(seq(min(x), max(x), length.out = 100), pred, col = 2, lwd = 2)
```



c. Predict the expected number of incidents at temperatures -0.6 and 11.67.

The exercise asks for the expected number of incidents, so we will have to specify *type* = *response*. If we didn't specify it, the default argument *link* would give us $\hat{\eta}$, and we are interested in $e^{\hat{\eta}}$.

```
new_x <- data.frame(temp = c(-0.6, 11.67))
prediction <- predict(fit, new_x, type = "response")

df_pred <- cbind(new_x, prediction)
```

Temp	Prediction
-0.60	20.692793
11.67	3.570342

We feel obligated to mention that the minimum observation for *temp* is 11.7 While the prediction for *temp* = 11.67 could be fine, as it is very close to observed data, we cannot be sure about the prediction for *temp* = -0.6. The model has not been trained on data with observations even near that temperature.

d. What are the confidence intervals for the expected number of incidents at the previous temperatures? Draw the confidence intervals curves onto the plot of Part a.

As there is no function that gives the confidence intervals, we have to construct them ourselves using the *se.fit* argument of the *predict* function.

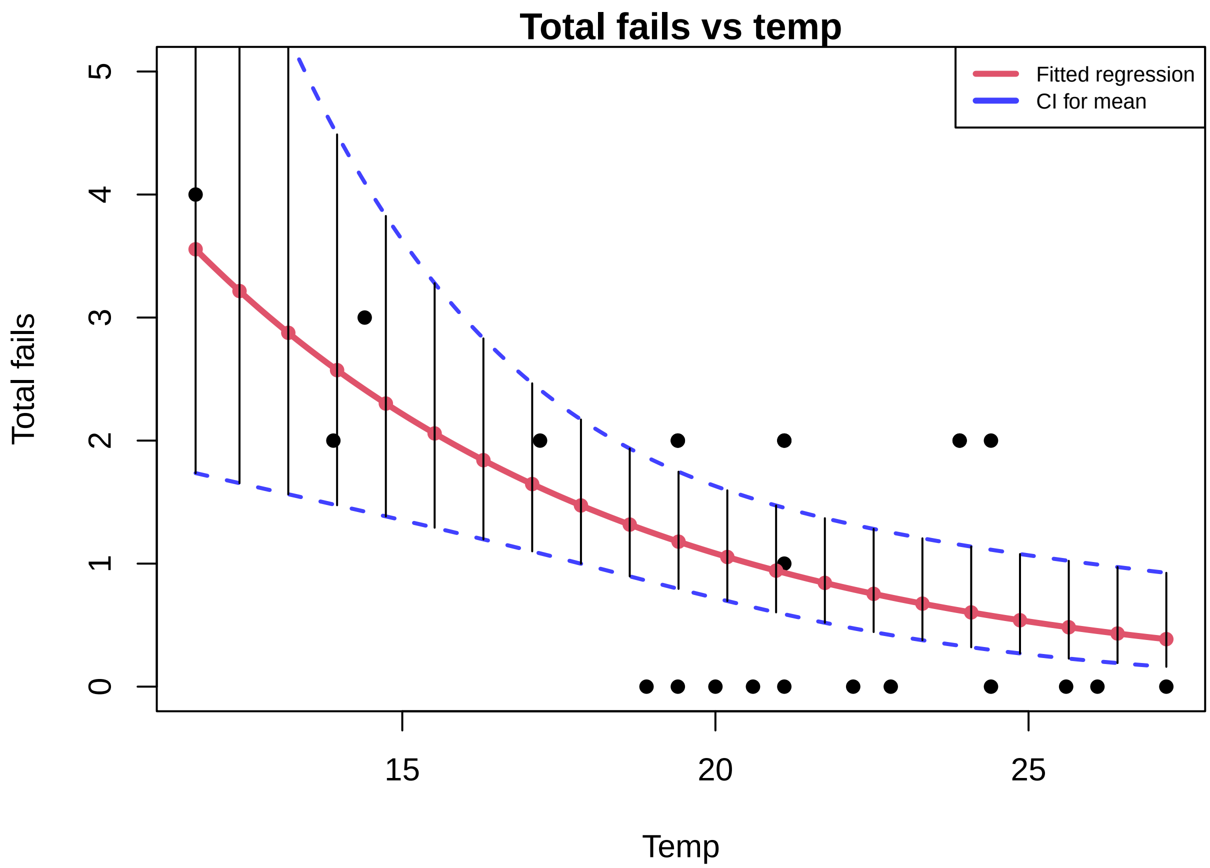
```
predictCIsPoisson <- function(object, newdata, level = 0.95) {
  pred <- predict(object = object, newdata = newdata, se.fit = TRUE)

  za <- qnorm(p = (1 - level) / 2)
  lwr <- pred$fit + za * pred$se.fit
  upr <- pred$fit - za * pred$se.fit

  fit <- exp(pred$fit)
  lwr <- exp(lwr)
```

```r
  upr <- exp(upr)

  result <- cbind(fit, lwr, upr)
  colnames(result) <- c("fit", "lwr", "upr")
  return(result)
}
```

Fit	Lower	Upper
20.692793	3.592928	119.176264
3.570342	1.739169	7.329557

To draw the confidence interval, we repeat the same plot of part a but computing the confidence interval for each point:

```r
coarse <- c(1, seq(10, 200, by = 10))
blue <- rgb(0, 0, 1, alpha = 0.75)
par(mar = c(4, 4, 1, 1) + 0.1, oma = rep(0, 4))
seq_x <- seq(min(x), max(x), length.out = 200)
pred <- predictCIsPoisson(fit, data.frame(temp = seq_x))
plot(x, y, main = "Total fails vs temp", xlab = "Temp",
     ylab = "Total fails", pch = 16, ylim = c(0, 5))
lines(seq_x, pred[,1], col = 2, lwd = 3)
lines(seq_x, pred[,2], col = blue, lwd = 2, lty = 2)
lines(seq_x, pred[,3], col = blue, lwd = 2, lty = 2)
points(seq_x[coarse], pred[coarse, 1],
       col = 2, pch = 16)
segments(x0 = seq_x[coarse], x1 = seq_x[coarse],
         y0 = pred[coarse, 2], y1 = pred[coarse, 3])
legend("topright", legend = c("Fitted regression", "CI for mean"),
       lwd = 3, col = c(2, blue), cex = 0.7)
```



e. What is the probability of having strictly more than five incidents at temperatures -0.6 and 11.67?

We know that $Y|X = x \sim P(e^{\eta})$, so using that we have to calculate either $P(X > 5)$ or $1 - P(X \leq 5)$ using the distribution function **ppois**:

```
sprintf("The probability of having more than 5 incidents with temp = %s
        is %s", new_x$temp[1], round(ppois(5, prediction[1], lower.tail = F), 4))
```

```
## [1] "The probability of having more than 5 incidents with temp = -0.6 \n        is 1"
```

```
sprintf("The probability of having more than 5 incidents with temp = %s
        is %s", new_x$temp[2], round(ppois(5, prediction[2], lower.tail = F), 4))
```

```
## [1] "The probability of having more than 5 incidents with temp = 11.67 \n        is 0.1518"
```

f. Can you improve the explanation of nfails.field + nfails.nozzle by using a Poisson regression with polynomial effects? Explore and comment on your results.

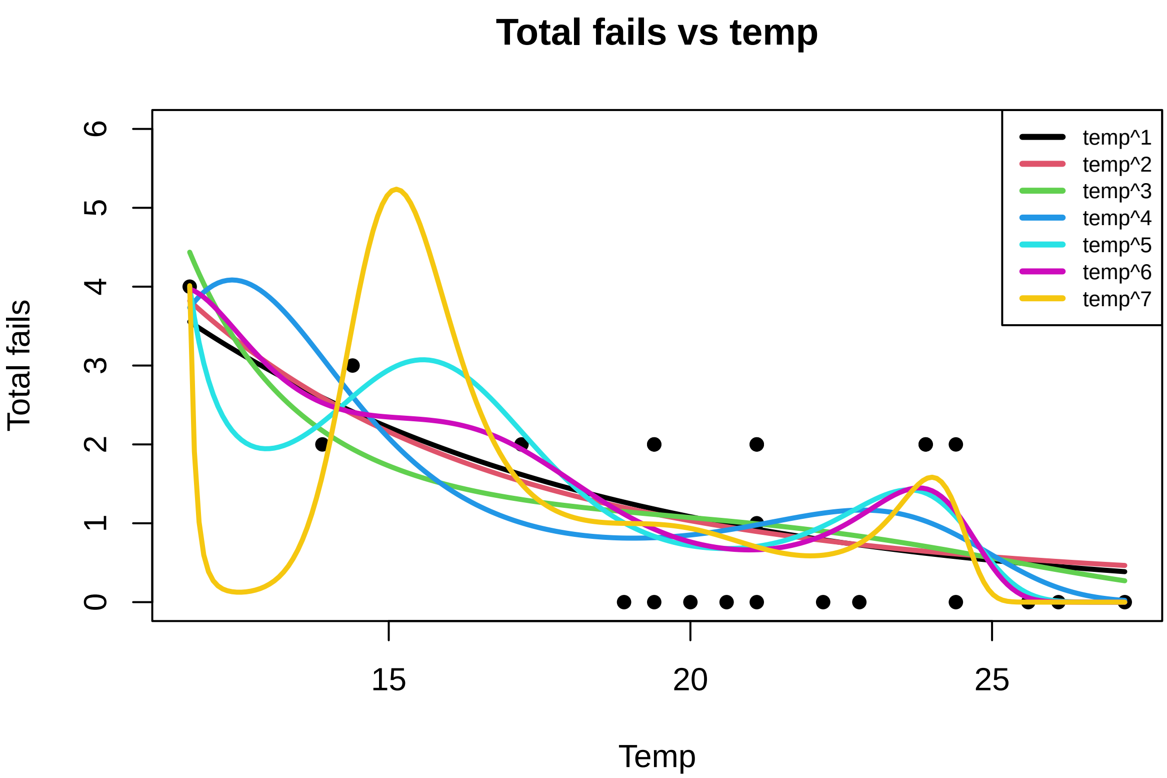
We will create models with polynomials of degree i, $i = 1, ..., 10$ and see their AIC and BIC:

```
mat_pred <- data.frame(x = seq_x)
new_x <- data.frame(temp = seq_x)

df <- data.frame(Degree = 1:7, Deviance = 0, AIC = 0, BIC = 0)

for (i in 1:7){
  fit_temp <- glm(total_fails ~ poly(temp, degree = i), family = poisson, data = challenger)
  df$Deviance[i] <- fit_temp$deviance
  df$AIC[i] <- fit_temp$aic
  df$BIC[i] <- BIC(fit_temp)
```

```r
  mat_pred <- cbind(mat_pred, predict(fit_temp, new_x, type = "response"))
}

colnames(mat_pred)[2:8] <- paste0("temp^", 1:7)
```

Degree	Deviance	AIC	BIC
1	26.94526	62.72621	64.99720
2	26.86420	64.64515	68.05163
3	26.34867	66.12962	70.67160
4	24.03421	65.81516	71.49263
5	21.43129	65.21224	72.02521
6	21.22733	67.00828	74.95674
7	19.84244	67.62339	76.70735

```r
plot(x, y, pch = 16, ylim = c(0,6), ylab = "Total fails",
     xlab = "Temp", main = "Total fails vs temp")
matlines(x = mat_pred[,1], y = mat_pred[,-1],
      lty = 1, lwd = 2.5, col = 1:7)
legend("topright", legend = colnames(mat_pred)[2:8], lty = 1, lwd = 3,
       col = 1:7, cex = 0.7)
```



We can see that, as we increase the degree of the polynomial, the deviance decreases and thus the explanation of the number of incidents improves. However, we can also see that the first (and simplest) model offers both the lowest AIC and BIC. We can conclude that, in this case, considering the small sample size and the information at hand, that the best model is the simplest one (no polynomials), as the more complex ones are more than probably overfitting.