Smoothing Splines Example

D2K Course Staff

July 25 2019

Introduction

Parametric functions can take you pretty far in the world of regression. Many problems can be fairly well-described by fitting a line on some transformed version of your data, or by fitting a polynomial regression line. However, there will come a point in your life where linear regression simply isn't good enough anymore. One day, you'll be sitting at your computer, fitting a regression, and you'll realize that no transformation works because the relationship between the response and predictors simply doesn't follow a nice parametric shape. That residual plot will never look IID, not in a million years, no matter what you try. What will you do then? Are you out of luck? Is disaster forthcoming? Has all possible hope been lost?

No. No it is not. Because you are a smart data scientist, and you know in this case that you should use a smoothing spline regression.

The goal of smoothing splines is to find the relationship between our predictor and response variables as the form

$$Y = f(x) + \epsilon,$$

where ϵ are your residuals and have their standard assumptions (0 mean, independent, constant variance) and $f(x)$ is some arbitrary function. The smoothing spline finds the $f(x)$ which minimizes:

$$\sum_i (Y_i - f(x_i))^2 + \lambda \sum_i f''(x_i)^2.$$

In this summation, we are trying to balance the accuracy of $f(x)$ to the given Y via squared error loss while maintaining smoothness via minimizing the magnitude of the 2nd derivative of $f(x)$. The regularzation parameter λ controls the balance, with larger values giving smoother curves.

Of course, this all assumes that our function $f(x)$ is always twice differentiable. How is this guaranteed? Well, the easiest way is to just fit a cubic function in between all of the points in your data. This is how the basic cubic smoothing spline in R operates. The points in your data are also called "knots" - these are the points at which the cubic spline functions are connected. (Other spline methods have other ways of choosing knots.)

Example

In this example, we will analyze the `Boston` data set from the `MASS` package.

```
library(MASS)

data(Boston)
head(Boston)
```

```
     crim zn indus chas   nox    rm  age    dis rad tax ptratio  black
1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90
2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90
3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83
4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63
5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90
6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12
  lstat medv
1  4.98 24.0
2  9.14 21.6
3  4.03 34.7
4  2.94 33.4
5  5.33 36.2
6  5.21 28.7
```

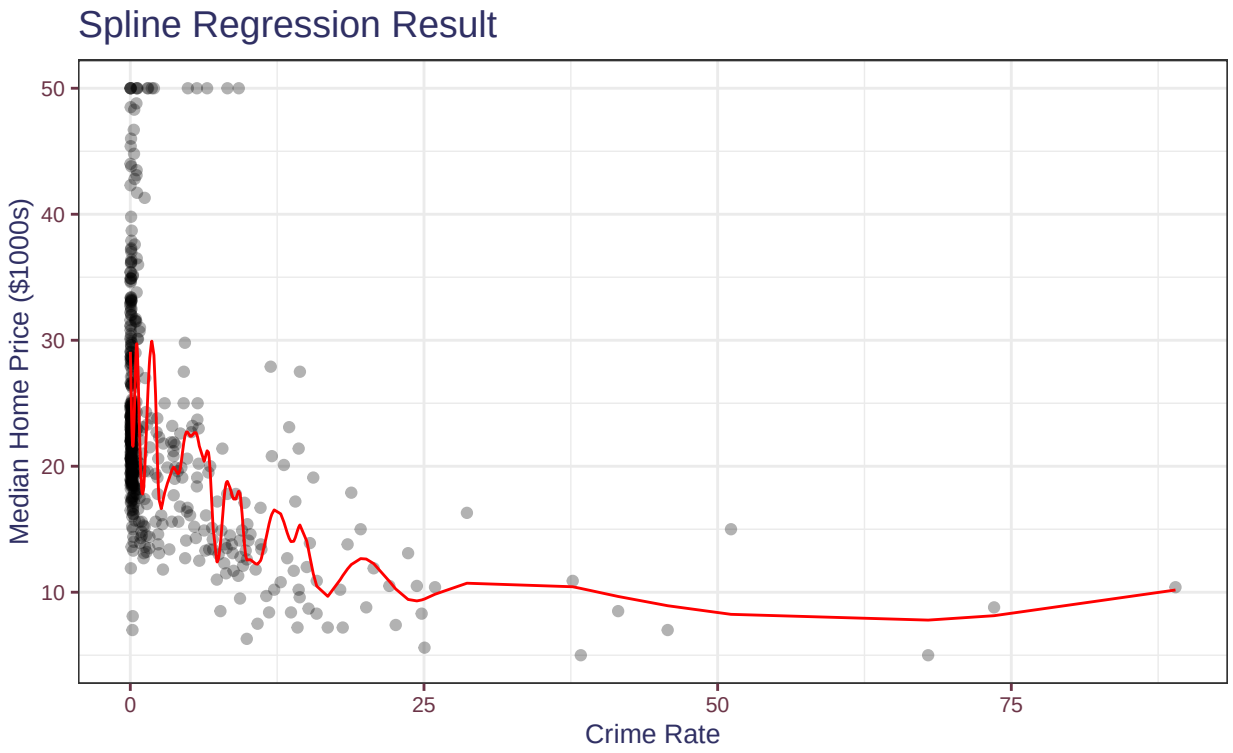
Say we want to predict the median price of homes in the Boston suburbs based on the local crime rate. We use the `smooth.spline` function from base R to fit the model.

```
mod1 <- smooth.spline(x = Boston$crim, y = Boston$medv)
```

If we then want to see the predictions on our data, we use the `predict` function on the fitted model.

```
pred1 <- predict(mod1)

ggplot() +
  geom_point(aes(x = Boston$crim, y = Boston$medv),
             alpha = 0.3) +
  geom_path(aes(x = pred1$x, y = pred1$y),
            color = "red") +
  labs(x = "Crime Rate",
       y = "Median Home Price ($1000s)",
       title = "Spline Regression Result") +
  theme1
```

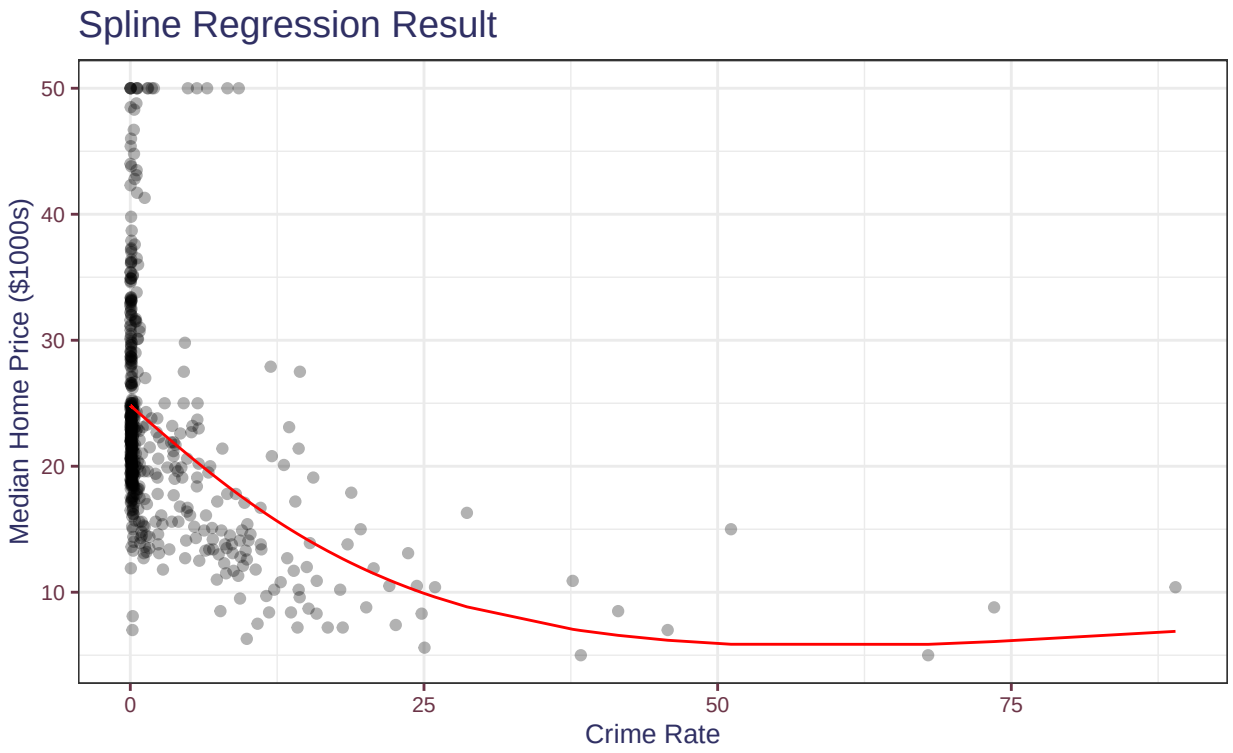


The curve seems a little too squigly near 0 - let's try changing the smoothness penalty parameter.

```
mod2 <- smooth.spline(x = Boston$crim, y = Boston$medv, lambda = 0.05)

pred2 <- predict(mod2)

ggplot() +
  geom_point(aes(x = Boston$crim, y = Boston$medv),
             alpha = 0.3) +
  geom_path(aes(x = pred2$x, y = pred2$y),
            color = "red") +
  labs(x = "Crime Rate",
       y = "Median Home Price ($1000s)",
       title = "Spline Regression Result") +
  theme1
```



Looks a bit more reasonable. You can see the individual predict values (or predict on new x values) all with the `predict` function.

Hypothesis Testing and Extensions

Smoothing splines can be added to any other type of regression model - e.g., we can have a model of the form

$$Y = X\beta + f_{\text{spline}}(X) + \epsilon,$$

which includes the predictor variables as both part the linear and spline parts of the regression equation. We can even choose to have some predictor variables as linear terms and others as spline terms if we so wish. This can be done by first fitting the linear part using `lm`, then taking the residuals and fitting those the desired splines. The same thing can also be done to GLMs or mixed effects models in the same fashion.

Hypothesis testing can be done by an ANOVA procedure via fitting nested models and running an F test.. This probably requires more than the`anova` function in R. However, you should be able to do this without needing a convenient built in function (as long as we can assume normality of the residuals, which should be the case for a spline). Time for you to brush up on your likelihood ratio tests! Remember the form of the F-test:

$$\frac{(RSS_1 - RSS_2)/(df_2 - df_1)}{RSS_2/(n - df_2)} \sim F_{df_2 - df_1, n - df_2}$$

where model 2 is the larger model, model 1 is the smaller null model, and *df* is the degrees of freedom *used* in each model. (You can pull out the degrees of freedom used in `smooth.spline`.)