assignment_05

2023-10-17

Polynomial Regression

I've hidden the code, but we set the seed for this assignment to `1`.

Here, we set up the `x` and `eps` as required:

```
x <- rnorm(100, mean = 0, sd = 1)
eps <- rnorm(100, mean = 0, sd = 0.5)
```

Then, we create the vector `y`:

```
y <- -1 + 0.5*x + 0.3*x^2 + eps
```

```
print(paste0("The length of y is ", length(y), "."))
```

```
## [1] "The length of y is 100."
```

The values of the coefficients are $\beta_0 = -1$, $\beta_1 = 0.5$, and $\beta_2 = 0.3$.

Scatter Away

Here is our scatterplot:

Then, we fit a quadratic model to predict `y`. I decided against using the `I(x^2)` method, as I think this way looks a bit cleaner:

```
x_2 <- x^2
quad_model <- lm(y ~ x + x_2)
```

Now, we look at the coefficients of our model:

```
## (Intercept)           x         x_2
##  -0.9716425   0.5085804   0.2405394
```

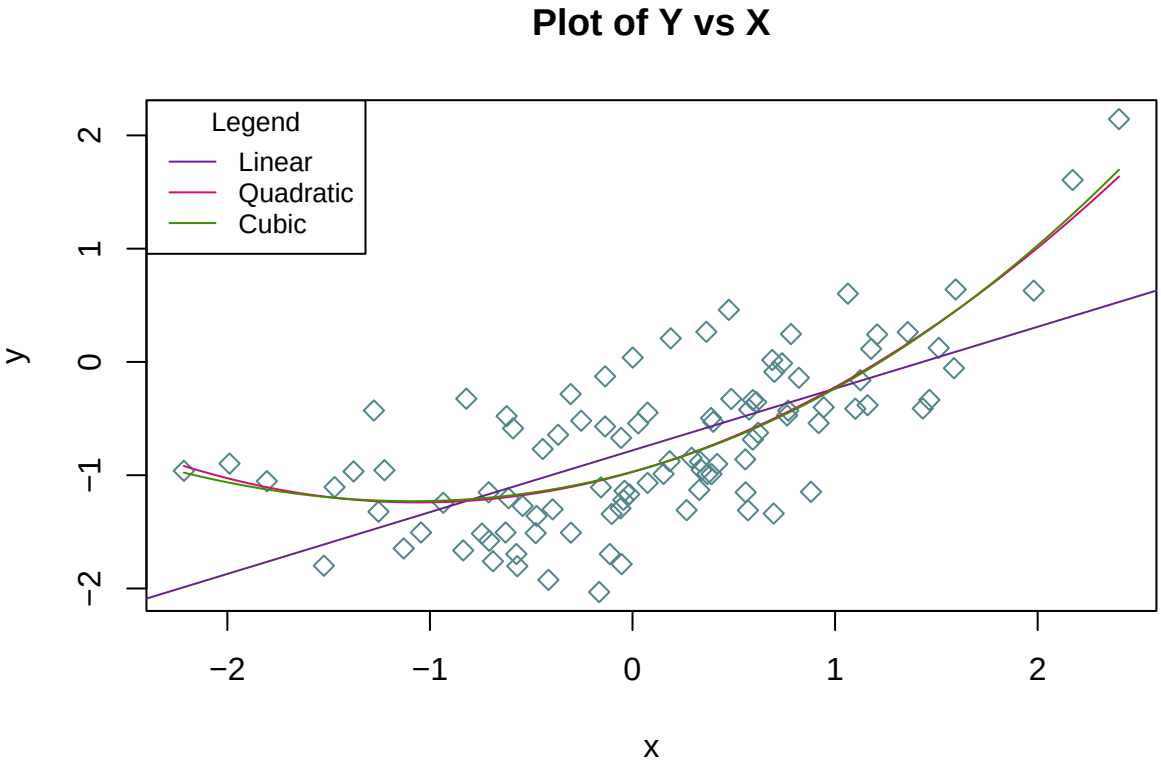
With the above coefficients and then rounded to 3 s.f., we have $\hat{\beta}_0 = -0.972$, $\hat{\beta}_1 = 0.509$, and $\hat{\beta}_2 = 0.241$. The values seem close to the original β_0, β_1, and β_2.

Now, we perform more regressions and use these regression to generate the predicted `y` values for each model:

```
#setting a sequence so we have more even gaps between the x values
x_pred <- seq(min(x), max(x), length = 100)
linear_model <- lm(y~x)
x_3 <- x^3
cubic_model <- lm(y ~ x + x_2 + x_3)
#predicting the y values using the cubic and quadratic models
cubic_pred <- predict(cubic_model, list(x = x_pred, x_2 = x_pred^2, x_3 = x_pred^3))
quad_pred <- predict(quad_model, list(x = x_pred, x_2 = x_pred^2))
```

During this assignment (and the peer tutoring session), I found out that we can't just use `lines(x, cubic_model$fitted.model)` because the `x` values are not sorted, so it was giving me this super messy criss cross plot. If we wanted to do this method, we would have to put all the values of `x` and the predicted values into a dataframe, sort them, and then we would be able to plot using these numbers. However, since `x` is generated randomly from the normal distribution, it's possible that there are large gaps between each value of `x`, which might result in a less even line, so doing it using `predict()` is still better.

Now, we do the required plot with all the models:



Which Regression is Better?

Here, I look at the R^2 values of each model:

```
## [1] "The $R^2$ value of the linear model is 0.44488899572941."
```

```
## [1] "The $R^2$ value of the quadratic model is 0.583269678911654."
```

```
## [1] "The $R^2$ value of the cubic model is 0.583596175532876."
```

It does seem like there is a sizable difference. Visually, the quadratic line seems to fit the original scatter plot better than the linear model. If we look at the numbers, the R^2 value, rounded to 3 s.f., for the quadratic model is `0.583`, while the R^2 value, rounded to 3 s.f., for the linear model is only `0.445`. So, it seems that the quadratic model accounts for a greater proportion of variance. Hence, the quadratic model is a better model.

On the other hand, just looking at the graph, we can't really tell if the cubic model is better than the quadratic model. If we looked at the numbers, the R^2 value, rounded to 3 s.f., for the quadratic model is `0.583`, and the R^2 value, rounded to 3 s.f., for the cubic model is 0.584. That's only a difference of `0.001`. I don't think we can call this a significant improvement.

One possible way to check which model is better is if we did this modeling on a portion of the `x` and `y` values instead, and then used the remaining portion to check and see which model predicts values that are closer to the actual values of `y` for the remaining values (so, similar to what we just did in class on Tuesday).