Pset 1, Machine Learning

Rohen Shah

1/13/2020

Statistical and Machine Learning

Supervised learning involves increasing predictive accuracy by training a learner via iterative feedback. That is, a program will use training data to make estimations (predictions for an outcome variable) and then, the learner will receive feedback in the form of its error rate (or information on the actual/observed outcome variables). In this sense, the learning is "supervised" because there is feedback that then generates a new round of improved predictions.

Unsupervised learning, on the other hand, doesn't involve receiving feedback (supervision) since there is no outcome (Y) variable available, and therefore no "error" rate that can be used for feedback.

The goal of unsupervised learning is usually either exploratory or data-reduction, whereas the goal of supervised learning is to improve predictions (minimize errors). In particular, supervised learning involves both an X and a Y variable, where characteristics of X are used to make increasingly better predictions of Y, initially using training data. Unsupervised learning on the other hand, only involves an X variable – and we are interested in discovering more about the underlying distribution of this variable. That is, we want to learn more about the data generating process of X in unsupervised learning. The target isn't clearly established in unsupervised learning, whereas the target parameters are well defined and increasingly better estimated in supervised learning.

The "Learning" process in supervised learning is conceptualized as the learner's response to the feedback of its error rate. Once a proposed model observes its error rate, it can then iteratively test other potential models and rule out ones with even higher error rates, and switch over to ones with lower error rates. Iteratively carrying this out can result in extremely high predictive accuracy since every combination - including polynomials, interaction terms, and even non-parametric estimation - is on the table in a way that a human researcher cannot feasibly test in a reasonable amount of time. The supervised learner can approach extremely high predictive accuracy based on the number of iterations and the quality and consistency of the training data used to "supervise" the program. There is no analogous "supervision" process in unsupervised learning, but rather the iterations will help uncover previously unknown patterns in the input (X) variables.

Linear Regression Regression

Part (a)

The regression predicted values and parameters are estimated as:

$$\hat{y_i} = 37.885 - 2.875x_i$$

Where $\hat{y_i}$ is the predicted value of miles per gallon and x_i is cylinders. The code that gives us is as follows:

```
reg1 <- lm(mpg~cyl, data=mtcars)
summary(reg1)
```

```
##
## Call:
## lm(formula = mpg ~ cyl, data = mtcars)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.9814 -2.1185  0.2217  1.0717  7.5186
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.8846     2.0738   18.27  < 2e-16 ***
## cyl          -2.8758     0.3224   -8.92 6.11e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.206 on 30 degrees of freedom
## Multiple R-squared:  0.7262, Adjusted R-squared:  0.7171
## F-statistic: 79.56 on 1 and 30 DF,  p-value: 6.113e-10
```

Part (b)

The population regression function is

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

Where y_i is miles per gallon, x_i is cylinders, and ε_i is the population error term for individual i.

Part (c)

After adding weight as a covariate, we can see that the *magnitude of the coefficient on cylinders has decreased*. Specifically, an increase of one cylinder is now associated with **1.5 fewer miles per gallon** rather than 2.9 fewer miles per gallon. Though the magnitude is lower, the effect is still statistically significant at the .01 level. The newly added weight coefficient is also negative and statistically significant at the .01 level, implying that a one unit increase in weight is associated with **3.2 fewer miles per gallon.**

Since the value of the coefficient increased, it implies that the *ommitted variable bias was negative*. Since we know that the coefficient on weight is negative, this implies that *weight and cylinders must be positively correlated* with each other. This further is consistent with the fact that including weight lowered the magnitude of the cylinders coefficient, because some of the magnitude (still negative in direction) has shifted from the cylinders coefficient to the weight coefficient.

```
reg2 <- lm(mpg~cyl+wt, data=mtcars)
summary(reg2)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2893 -1.5512 -0.4684  1.5743  6.1004
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  39.6863     1.7150  23.141  < 2e-16 ***
## cyl          -1.5078     0.4147  -3.636 0.001064 **
## wt           -3.1910     0.7569  -4.216 0.000222 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 2.568 on 29 degrees of freedom
## Multiple R-squared:  0.8302, Adjusted R-squared:  0.8185
## F-statistic: 70.91 on 2 and 29 DF,  p-value: 6.809e-12
```

Part (d)

Including the interaction term between weight and cylinder still results in a statistically significant negative coefficient for both cylinders and weight, **however both coefficients are much larger in magnitude now**. By including the interaction term, we are theoretically implying that the *effect of weight on miles per gallon is different for different levels of cylinders*. Since the interaction term has a positive coefficient, this would imply that weight lowers mpg *more for a 4-cylinder car than it does for a 6-cylinder* car, for example, since the positive interaction term would cancel out some of the negative coefficient on weight, when cylinders is higher.

```
reg3 <- lm(mpg~cyl + wt + cyl*wt, data=mtcars)
summary(reg3)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + wt + cyl * wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2288 -1.3495 -0.5042  1.4647  5.2344
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  54.3068     6.1275   8.863 1.29e-09 ***
## cyl          -3.8032     1.0050  -3.784 0.000747 ***
## wt           -8.6556     2.3201  -3.731 0.000861 ***
## cyl:wt        0.8084     0.3273   2.470 0.019882 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.368 on 28 degrees of freedom
## Multiple R-squared:  0.8606, Adjusted R-squared:  0.8457
## F-statistic: 57.62 on 3 and 28 DF,  p-value: 4.231e-12
```

Non-Linear Regression

Part (a)

The estimated regression model here is

$$\hat{y_i} = -10.425 + 5.294x_i - .053x_i^2$$

Where $\hat{y_i}$ is the predicted wage, and x_i is the age. The coefficient on both x_i and x_i^2 are statistically significant, while the y-intercept term is not. The fact that the coefficient on x_i^2 is negative implies a negative quadratic relationship, where it increases at a decreasing rate and then decreases even steeper. This also implies that **there will exist a maximum value for wage**, which occurs when the value of age solves the First Order Condition of $\frac{\partial y}{\partial x} = 5.294 - 2(0.053)x_i = 0$.

```
wage <- read_csv("wage_data.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   year = col_double(),
##   age = col_double(),
##   maritl = col_character(),
##   race = col_character(),
##   education = col_character(),
##   region = col_character(),
##   jobclass = col_character(),
##   health = col_character(),
##   health_ins = col_character(),
##   logwage = col_double(),
##   wage = col_double()
## )
```

```
reg <- lm(wage~age + I(age^2), data=wage)
summary(reg)
```

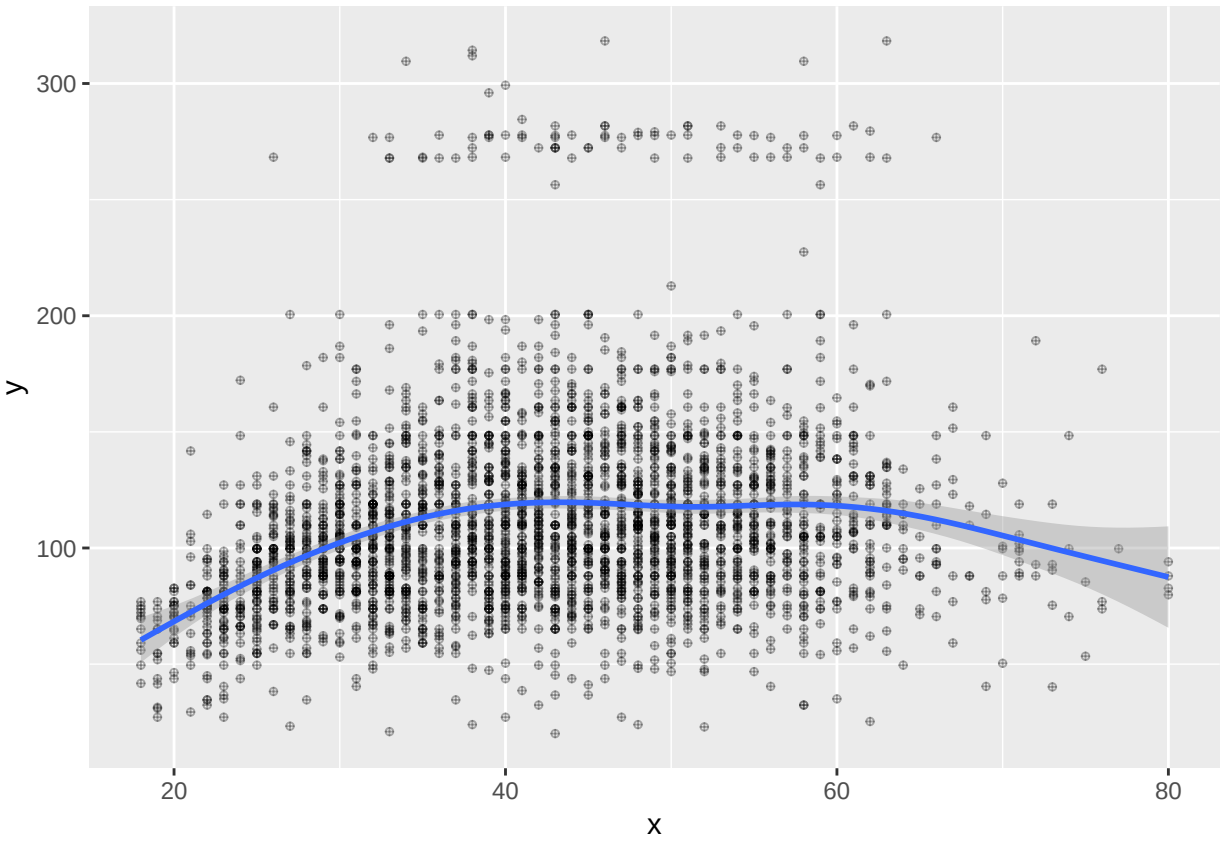
```
##
## Call:
## lm(formula = wage ~ age + I(age^2), data = wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -99.126 -24.309  -5.017  15.494 205.621
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -10.425224   8.189780  -1.273    0.203
## age           5.294030   0.388689  13.620   <2e-16 ***
## I(age^2)     -0.053005   0.004432 -11.960   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.99 on 2997 degrees of freedom
## Multiple R-squared:  0.08209,    Adjusted R-squared:  0.08147
## F-statistic:   134 on 2 and 2997 DF,  p-value: < 2.2e-16
```

Part (b)

The plot of this polynomial, with bounds for the confidence interval, is as follows:

```
df<-data.frame("x"=wage$age, "y"=wage$wage)
plot <- ggplot(df, aes(x, y)) +
  geom_point(alpha=3/10, shape=10, fill="blue", colour="black", size=1) +
  geom_smooth(formula=y~poly(x,2),se=TRUE)
plot
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



Part (c)

We see here that the wage is in fact maximized at a certain age (near the middle of our data). So, a polynomial fit is a much better choice than a linear model. Additionally, even though the data has a very high variance, our confidence interval around the prediction is fairly narrow (at least from the naked eye) which is reasonable given that the very high sample size will make the standard errors very small, thereby leading to a narrow confidence interval.

Part (d)

Statistically, a polynomial regresssion differs from a bivariate linear regression is that we technically have one additional dimension (variable) that we are controlling for: the square of the x variable. Statistically it is treated no differently than a completely new variable added to the model, and its coefficient is solved as such. We also therefore have one less degree of freedom because we have to estimate one more parameter.

Substantively, a polynomial regression is a much more flexible version of a linear regression because not only can it capture shapes and curvatures that a linear regression cannot, but a polynomial regression can capture anything that a linear regression can: a linear regression is a special case of a polynomial regression where one of the coefficients is 0 (the square term). That is, if the true relationship was in fact perfectly linear, we wouldn't get a biased coefficient by choosing a polynomial because our best estimate of the x^2 coefficient would simply be 0 (and our regression output in R will tell us that) which means our best predicted polynomial will simplify to a line. So, with the cost of one degree of freedom, a polynomial is a fairly good deal given its flexibility compared to a linear model, and we should use it if we are less than certain that the true underlying relationship is linear.