Problem Set 1

Artemis Yang

1/15/2020

Supervised and Unsupervised Learning

A fundamental difference between supervised and unsupervised learning lies in whether the data is well-labeled. The data we use in supervised learning is already associated with well-defined labels. Unsupervised learning, on the other hand, deals with unlabeled data and, the machine is on itself to identify patterns or useful information from the data. For example, suppose you have a dataset consisting of pictures of different fruits. In supervised learning, the pictures are associated with the name of the fruit, such as “apple” or “orange”. You as the researcher can use your preferred algorithm to train the machine using these data labeled with fruit names. Then you will be able to use your trained model to identify the type of fruit in some new fruit pictures. In unsupervised learning, your fruit pictures are not assigned labels of the correct name of the fruit type, but you can still use a learning algorithm to recover some information of the uncategorized data. The information could be the shape or color of the fruits. In this case, your machine can still tell the difference between apples and oranges, even though it does not know the correct names of the fruits.

The word “supervised” and “unsupervised” themselves can somehow describe the difference in the learning process: Supervised learning takes place as if a teacher is supervising the learning process. Supervised learning algorithms usually have some embedded criteria to test on model performance. For example, it can check whether your model correctly identifies an apple as an apple. If the model assigns value “orange” to an apple picture, then you will know this model is making a mistake. Such a process is similar to a situation where a teacher knows the right answer and he/she can point out the mistakes the students make. In unsupervised learning, there is no such a “teacher”. This is potentially a drawback of unsupervised learning: There is no way to check on model performance.

The learning goals also differ for supervised and unsupervised learning. The basic target for supervised learning is to approximate the function that maps input data (X) with the output data (Y). A supervised model will use training data to find the relationship between the input and the output. However, unsupervised learning aims at learning the inherent structure of the input data and it does not use output data. Take our fruit picture example again: In supervised learning, the input data could be the shape or color of the fruit, and the output would be the name of the fruit. In unsupervised learning, we can still group the pictures of the same fruit type together, but we are unable to assign any output values to the groups. In a real-world case, we can choose our preferred type of learning based on the different goals the two types of learning can achieve. If we have a specific outcome we would like to learn and we know this outcome can be affected by some input variables, then we should use supervised learning to predict the outcome using the input variables. If we have a dataset with information that we do not have any previous knowledge on, then we might want to use unsupervised learning to discover some patterns of the data for us.

The most commonly seen techniques for supervised learning are classification and regression. Regression can use the training data to predict a single output value. For instance, you can predict air quality using input data of pollution level, precipitation, wind, etc. Classification can group output values into different classes. The fruit picture example we discuss before is a typical classification problem. Unsupervised learning techniques include clustering and association. Association allows you to establish associations between variables in large data, which can be thought of as an unsupervised version of regression. Clustering algorithms will find natural clusters(groups) in the data. For example, we can group fruit pictures based on the difference in their appearances.

Linear Regression

a. Predict *mpg* as a function of *cyl*:

```r
names(mtcars)
```

```
##  [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear"
## [11] "carb"
```

```r
lm1 <- lm(mpg ~ cyl, data=mtcars)
```

This is the output:

```r
summary(lm1)
```

```
##
## 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
```

The parameter value (coefficient) for cylinders is -2.8758 with standard error of 0.3224. The interpretation is that for every one unit of increase in cylinders, the miles per gallon for the car decreases by 2.8758. The effect is statistically significant at the 1% level.

b.Regression function: $mpg = \beta_0 + \beta_1 cyl + e$

c.Predict *mpg* as a function of *cyl* and *wt*:

```r
lm2 <- lm(mpg ~ cyl+wt, data=mtcars)
```

This is the output:

```r
summary(lm2)
```

```
##
## 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
```

The coefficient for cylinders becomes -1.5078 with standard error of 0.4147. The magnitude of this coefficient is smaller than in part a. This is because the model in part a might suffer from omitted variable bias, adding vehicle weight will likely reduce the bias to some extent. We can interpret this coefficient in this way: For every one unit of increase in cylinders, the miles per gallon for the car decreases by 1.5078. The coefficient for vehicle weight is -3.191 with standard error of 0.7569. We can interpret this coefficient in this way: For every one unit of increase in vehicle weight, the miles per gallon for the car decreases by 3.191. The effects are statistically significant at the 1% level.

d.Predict *mpg* as a function of *cyl*, *wt*, and their interaction:

```r
lm3 <- lm(mpg ~ cyl*wt, data=mtcars)
summary(lm3)
```

```
##
## Call:
## lm(formula = mpg ~ 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
```

The sign for the coefficiencts of *cyl* and *wt* are still negative, which is consistent with the previous results, but the magnitudes of the coefficients change a lot. By including an interaction term, we are asserting that the effect of of cylinders on mpg depends on the effect of vehicle weight on mpg and similarly, the effect of of vehicle weight on mpg depends on the effect of cylinders on mpg. The interpretation would be: For every one unit of increase in cylinders, the miles per gallon for the car decreases by 2.9948 (3.8032-0.8084). For every one unit of increase in vehicle weight, the miles per gallon for the car decreases by 7.8472 (8.6556-0.8084). The effects are statistically significant at the 5% level.

Nonlinear Regression

a. Predict *wage* as a function of a second order polynomial for *age*:

```r
library(ggplot2)
setwd("/Users/artemisyang/Dropbox/MACS33002 Introduction to Machine Learning/PS1")
wage_data <- read.csv(file="wage_data.csv")
fit <- lm(wage ~ poly(age,2,raw=T), data=wage_data)
```

This is the output:

```r
summary(fit)
```

```
##
## Call:
## lm(formula = wage ~ poly(age, 2, raw = T), data = wage_data)
##
## 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
## poly(age, 2, raw = T)1   5.294030   0.388689  13.620   <2e-16 ***
## poly(age, 2, raw = T)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
```

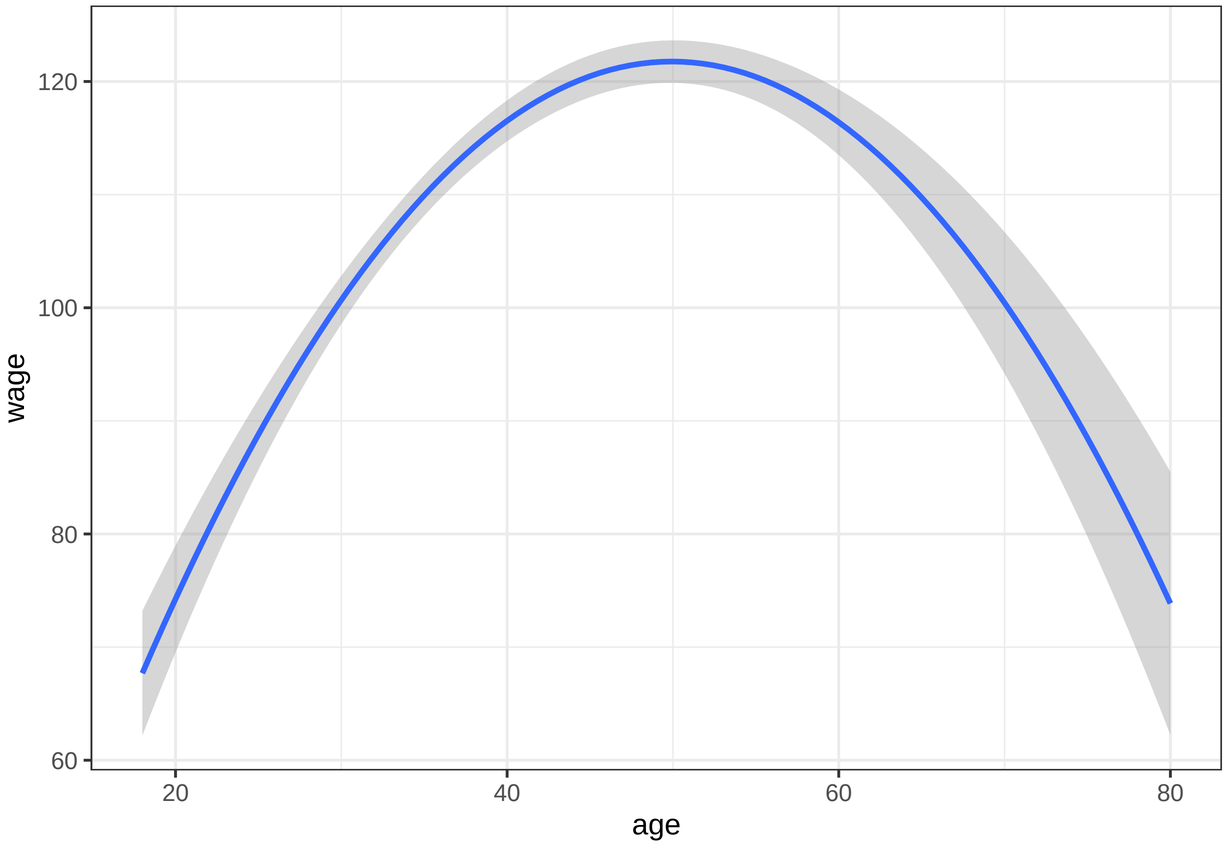
In the second order polynomial regression, the effect of *age* on *wage* is given by the total derivative with respect to age. The interpretation is that when *age* changes from some value a to $a+1$, wage will increase by $5.294 - 0.053 * (2a + 1)$. The effects are statistically significant at the 0.01 significance level. The R^2 is about 0.08, meaning that 8% of the variance in the wage data can be explained by this model.

b. Plot with 95% CI:

```r
confint(fit, level=0.95)
```

```
##                                  2.5 %      97.5 %
## (Intercept)               -26.48338385  5.63293533
## poly(age, 2, raw = T)1   4.53190664  6.05615343
## poly(age, 2, raw = T)2  -0.06169479 -0.04431534
```

```
prd <- data.frame(age = seq(from = range(wage_data$age)[1], to = range(wage_data$age)[2],length.out = 1
err <- predict(fit, newdata = prd, se.fit = TRUE)
prd$lci <- err$fit - 1.96 * err$se.fit
prd$fit <- err$fit
prd$uci <- err$fit + 1.96 * err$se.fit
ggplot(prd, aes(x = age, y = fit)) +
       labs(y= "wage", x = "age")+
       theme_bw() +
       geom_line() +
       geom_smooth(aes(ymin = lci, ymax = uci), stat = "identity")
```



c. We can see that the function is concave and, wage reaches its highest level when age is approximately 50. This implies that we can predict that a person can achieve his/her highest wage at age of 50. By fitting a polynomial regression, we are asserting that age does not have a linear effect on wage. From the graph we can see that age has a positive effect on wage before 50 and a negative effect after 50.

d. In a linear regression, the effect of the independent variable on the dependent variable is monotonic, i.e. either positive or negative. Also, in a linear regression, the rate of change in the dependent variable with respect to the independent variable is constant (the derivative is a constant, so the slope does not change). However, in a non-linear polynomial regression, the sign of the effect of the independent variable on the dependent variable is subject to the value of the independent variable. The rate of change in the dependent variable with respect to the independent variable is no longer constant and, the coefficient on the quadratic term indicates the direction and steepness of the function's curvature. In reality, we need to be very careful about choosing the form of regression function in order to fit the data correctly.