3 Linear regression

ISLR Chapter 3 uses linear regression to illustrate many concepts and issues in regression modeling that are also applicable in other types of regression (e.g., logistic, Poisson, Cox, etc.). Understanding these issues will allow us to see the limitations of various methods and how far we can interpret the results from regression analyses.

Seven questions using `Advertising` data as an example; to be answered in ISLR 3.4. All except Q5 are inference questions. Synergy effect (in marketing) is called interaction effect or effect modification in Statistics. (`Advertising` data: `sales` in thousands of units, `TV`/`radio`/`newspaper` in thousands of dollars.)

```
Advertising = read.table("Advertising.csv", header=T, sep=',', row.names=1)
names(Advertising); dim(Advertising)
Advertising$totalcost = with(Advertising, TV + radio + newspaper)
cor(Advertising)
```

To visualize pairwise relationship, we can use one of the following:

```
pairs(Advertising)
car::scatterplotMatrix(Advertising)
GGally::ggpairs(Advertising)
```

3.1 ISLR 3.1: Simple linear regression (a single predictor X)

Most of the concepts in ISLR 3.1 are applicable to multiple linear regression and other regression models.

- 3.1.1: **Least squares** is a model fitting criterion. It implicitly assumes (1) equal variance of the residuals across observations and (2) symmetry in the residuals.

For the model, sales $= \beta_0 + \beta_1$TV, the equal variance assumption is incorrect. Fitting the model using least squares allows high variance observations on the right to have a high influence on the fitted model. The linearity assumption is also incorrect for low values of `TV`.

```
with(Advertising, plot(TV, sales))
fittv = lm(sales ~ TV, data=Advertising)
abline(fittv, col=2, lwd=2)
```

In R, a fitted model is a list; `summary()` also returns a list. Use `names()` to learn what a list contains.

```
names(fittv)
sumfittv = summary(fittv)
names(sumfittv)
sumfittv$r.squared
```

Figure 3.2 in the book is incorrect. As shown in its code (https://github.com/JWarmenhoven/ISLR-python/issues/1), a linear model $y = \alpha_0 + \alpha_1 x$ was fit with `y=sales` and `x=TV-mean(TV)` to obtain $\hat{\alpha}_0 = 14.02$ and $\hat{\alpha}_1 = 0.0475$, and the RSS was calculated over a grid of (α_0, α_1) but plotted over a grid of $(\alpha_0 - 0.0475 \cdot \text{meanTV}, \alpha_1)$. The relationship between this model and the intended model, sales $= \beta_0 + \beta_1$TV, is that $\beta_0 = \alpha_0 - \alpha_1 \cdot \text{meanTV}$ and $\beta_1 = \alpha_1$. Here, β_0 depends on both α_0 and α_1, and thus the grid of (β_0, β_1) is not just a shift of α_0 on the grid of (α_0, α_1). While $\hat{\alpha}_0$ and $\hat{\alpha}_1$ are indepedent due to the centering of `TV`, $\hat{\beta}_0$ and $\hat{\beta}_1$ are dependent. In addition, the book figure shows RSS divided by 1000, for no obvious reason. The following code generates correct figures.

```
ngrid = 100
grid1 = seq(5, 9, length.out=ngrid)
grid2 = seq(0.028, 0.068, length.out=ngrid)
grid3 = expand.grid(grid1, grid2) # gives the same results

RSS = NULL
for(i in 1:(ngrid^2))
  RSS[i] = with(Advertising, sum((sales - cbind(1, TV) %*% t(grid3[i,]))^2))
```

```
## static 3D plots
contour(grid1, grid2, matrix(RSS, ngrid), xlab='beta0', ylab='beta1', nlevels=25, xlim=c(5,9))
persp(grid1, grid2, matrix(RSS, ngrid), theta=30, phi=20, xlab='beta0', ylab='beta1', zlab='RSS')
## interactive 3D plot using the rgl package
rgl::plot3d(grid3$Var1, grid3$Var2, RSS)
```

A lesson: Analysts and programmers sometimes implement shortcut algorithms. When a shortcut is not equivalent to the intended approach, there can be unexpected consequences.

- 3.1.2: Suppose we fit a linear model to a dataset. Consider 4 things and their value at $X = x_0$:

(1) **least squares line** $\hat{f}(x)$ (estimated from data), and $\hat{f}(x_0)$;
(2) **population regression line** $E(\hat{f})(x)$ (unknown), and $E(\hat{f})(x_0)$;
(3) **true curve** $f(x)$ (unknown), and $f(x_0)$;
(4) a **future outcome** y_0 (unknown).

At $X = x_0$, (1) yields the predicted value $\hat{f}(x_0)$. The expected squared error loss at $X = x_0$ is

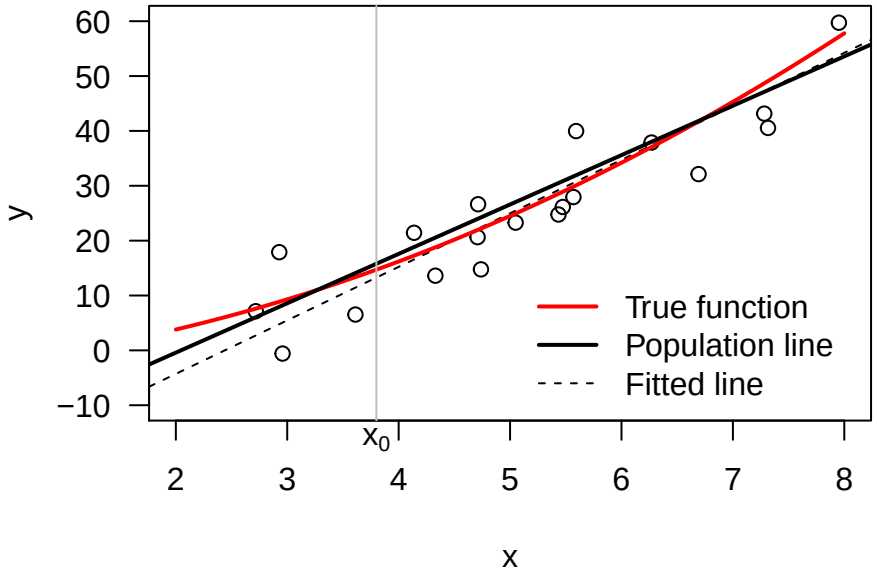
$$E(y_0 - \hat{f}(x_0))^2 = Var[\hat{f}(x_0)] + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon). \tag{2.7}$$

- $Var[\hat{f}(x_0)]$ reflects the discrepancy between (1) and (2) (due to sampling variation),
- $[Bias(\hat{f}(x_0))]^2$ reflects the discrepancy between (2) and (3) (due to modeling choice), and
- $Var(\epsilon)$ reflects the discrepancy between (3) and (4) (due to irreducible error).

An example is below:

```
truef = function(x) -3 + 2*x + 0.7*x^2
set.seed(209); N = 20; x = runif(N, 2, 8); y = truef(x) + rnorm(N, 0, 5)  ## real data

plot(x,y, xlim=c(2,8), ylim=c(-10,60), las=1)                              ## data
abline(lm(y ~ x), lty=2, lwd=1)                                            ## fitted linear line
curve(truef, add=T, lwd=2, col=2)                                          ## true curve
n = 1e6; xpop = runif(n, 2, 8); ypop = truef(xpop) + rnorm(n, 0, 5)
abline(lm(ypop ~ xpop), lwd=2)                                             ## population regression line
abline(v=3.8, col='grey'); mtext(expression(x[0]), side=1, at=3.8)         ## x0
legend(5, 15, bty='n', col=c(2,1,1), lty=c(1,1,2), lwd=c(2,2,1),
       c("True function", "Population line", "Fitted line"))
```



Parameters: In the linear model $y = \beta_0 + \beta_1 x$, there are 3 parameters: regression coefficients β_0 and β_1, and residual variance σ^2.

Every regression coefficient has a unit. For example, $\hat{\beta}_1 = 0.0475$ means 0.0475 thousand units sold for every \$1000 spent on TV advertising. If `TV` were recorded in dollars, $\hat{\beta}_1$ would be 0.0000475; or if `sales` were recorded in units, $\hat{\beta}_1$ would be 47.5.

Parameter estimation: *Point estimate* $\hat{\beta}_1$. *Standard error* $\text{SE}(\hat{\beta}_1)$ is the accuracy of the estimate $\hat{\beta}_1$. A *confidence interval* $(\hat{\beta}_{1l}, \hat{\beta}_{1u})$ is an interval estimate of β_1. IF the model is correct, a 95% CI should cover the true β_1 95% of the time (i.e., for 95% of datasets). A 99% CI is wider than a 95% CI. Many of these measures are calculated under the assumption that the model is correct. They can be misleading when the linearity assumption is invalid.

```
fittv = lm(sales ~ TV, data=Advertising)
summary(fittv)
summary(fittv)$coef; summary(fittv)$r.squared  ## extract results
confint(fittv)  ## CI for parameter estimates
```

- *Bias* of $\hat{\beta}_1$ is $Bias(\hat{\beta}_1) = E(\hat{\beta}_1) - \beta_1$. It is a long-term property. An estimator is unbiased if its bias is zero.
- *Mean squared error* (MSE) of $\hat{\beta}_1$ is $E[(\hat{\beta}_1 - \beta_1)^2] = Var(\hat{\beta}_1) + [Bias(\hat{\beta}_1)]^2$. MSE is a better measure of performance of an estimator than unbiasedness because MSE takes both variance and bias into account.

Hypothesis testing: $H_0 : \beta_1 = 0$ vs. $H_a : \beta_1 \neq 0$.

The two hypotheses can be viewed as **nested models**: the model under H_a is a "full" model, and that under H_0 is a "reduced" model. We test if the "full" model explains the outcome significantly better than the "reduced" model.

p-value: The two hypotheses are not treated equally: p-values are computed under H_0 (i.e., under the assumption that H_0 is correct), but their interpretation favors H_a.

(1) *Rationale*: Even if H_0 is correct, there may be some level of "inconsistency" between the data and H_0. A p-value is the probability of seeing the same level of "inconsistency" or higher when H_0 is correct. Different ways of quantifying "inconsistency" lead to different tests and different p-values.
(2) *Interpretation*: The smaller a p-value is, the more statistically significant "inconsistency". If a threshold α is chosen so that we declare statistical significance whenever $p < \alpha$, the threshold is effectively our tolerance for false positives. However, a non-significant p-value does not mean we accept H_0.
(3) **Absence of evidence is not evidence of absence.** — Altman and Bland (1995, BMJ)

This is similar to what a court would do to a suspect: assume innocence, and declare guilty only with enough evidence. *But* in the "hypothesis testing court", a suspect is not aquitted for lack of evidence (unless there is a strong "consistency" between data and H_0).

Notes: (1) A p-value is **not** the probability that H_a is correct. (2) Statistical significance may not imply biological/practical significance. (3) For simple tests, there is often a correspondence between p-value and CI (e.g., $p < 0.05$ corresponds to 95% CI not covering the true β). (4) Ignore the p-value for the intercept and for the main effects when interaction terms are present.

- 3.1.3: **Measures of fit**: Residual SE (RSE) depends on the scale of outcome. R^2 is scale independent. R^2 increases with model complexity, and thus it cannot be used for model selection. R^2 has three interpretations: (1) $R^2 = [corr(y, \hat{y})]^2$; (2) R^2 is the fraction of outcome total SS explained by the model; (3) R^2 is closely related to the likelihood-ratio statistic when the residuals are assumed to be normal.

3.2 ISLR 3.2 Multiple linear regression (multiple predictors)

- 3.2.1: **Interpretation of p-values in a regression table**: A p-value for a predictor is the significance of the predictor **after** all other features in the regression have been taken into account. In other words, the interpretation of an individual p-value always depends on the context of what other variables or transformations have been included.

As a result, the p-value for a variable should be ignored if the variable is also involved in a transformation or an interaction term (details below in 2.1.5 and 2.2.1). Also see my comment on ISLR 3.3.2 for interpretion of the coefficient of a variable when it is also involved in an interaction term.

Also as a result, the following can happen in practice:

- x_1 becomes less significant after x_2 is added to regression (often due to collinearity between x_1 and x_2)
- x_1 becomes more significant after x_2 is added to regression (often due to both being important predictors)

For example: (1) `TV` and `radio` are nearly independent, and both have effects on `sales`. When both are in the model, each benefits from the other's presence and becomes a lot more significant than being alone in a model. This is because the residual noise is a lot reduced when both are in the model. Similar patterns occur between `TV` and `newspaper`. (2) `radio` and `newspaper` are correlated. When both are in the model, both have reduced significance.

```
cor(Advertising)
summary(lm(sales ~ TV, data=Advertising))$coef                    ## Table 3.1
summary(lm(sales ~ radio, data=Advertising))$coef                 ## Table 3.3
summary(lm(sales ~ newspaper, data=Advertising))$coef             ## Table 3.3
summary(lm(sales ~ TV + radio, data=Advertising))$coef
summary(lm(sales ~ radio + newspaper, data=Advertising))$coef
summary(lm(sales ~ TV + newspaper, data=Advertising))$coef
```

- 3.2.2: 4 questions:
  1. F-test for $H_0 : \beta_a = \cdots = \beta_b = 0$ vs. H_a : Any of these $\neq 0$. We are comparing two **nested models**: the "full" model under H_a and the "reduced" model under H_0.
  2. **Model selection criteria**: Mallow's C_p, AIC, BIC, adjusted R^2 (details in ISLR 6). Cross-validation is a powerful alternative to these traditional measures. For **model building**, stepwise procedures with a stopping rule (as mentioned in ISLR 3.2.2) should be avoided because they likely end up overfitting the data. A stepwise procedure followed by appropriate model selection is okay.
  3. Measures of fit: RSE, R^2.
  4. Confidence interval vs. prediction interval.

```
fittv = lm(sales ~ TV, data=Advertising)
predict(fittv, data.frame(TV=seq(100,300,100)), interval="confidence")  ## CI for fitted values
predict(fittv, data.frame(TV=seq(100,300,100)), interval="prediction")  ## PI for predicted values
```

3.3 3D plots in R

Various ways of generating scatter plots in R: http://www.statmethods.net/graphs/scatterplot.html .

```
fit1 = lm(sales ~ TV + radio, data=Advertising)
## traditional residual plot
with(Advertising, plot(TV, resid(fit1))); abline(h=0)

library(scatterplot3d)  ## static 3D graphs using the scatterplot3d package
s3d = with(Advertising, scatterplot3d(TV, radio, sales, highlight.3d=TRUE, angle=20))
s3d$plane3d(fit1)

library(rgl)            ## interactive 3D graphs using the rgl package
with(Advertising, plot3d(TV, radio, sales, col=2, size=5))
planes3d(fit1$coef[2], fit1$coef[3], -1, fit1$coef[1], alpha=.5)
```

3.4 A model building exercise

We demonstrate two ways of model building, one traditional and one with cross-validation. Note that high-degree polynomials are not a good choice of models (explained in my ISLR Chapter 7 notes). I use them here just for the demonstration.

The plots above show a clear lack of fit in model `fit1`. We now fit an interaction model, where the effect of `TV` on `sales` is linear, but with both intercept and slope changing as linear functions of `radio` (and vise versa).

$$\begin{aligned}\text{sales} &= (\beta_0 + \beta_1 \text{radio}) + (\beta_2 + \beta_3 \text{radio}) \cdot \text{TV} = \beta_0 + \beta_1 \text{radio} + \beta_2 \text{TV} + \beta_3 \text{radio} \cdot \text{TV} \\ &= (\beta_0 + \beta_2 \text{TV}) + (\beta_1 + \beta_3 \text{TV}) \cdot \text{radio}.\end{aligned}$$

```
fiti1 = lm(sales ~ TV * radio, data=Advertising)
summary(fiti1)
anova(fit1, fiti1)  ## significant improvement
with(Advertising, plot(TV, resid(fiti1))); abline(h=0)      ## residual plot

## draw the fitted curve
ngrid = 200
grid1 = seq(min(Advertising$TV), max(Advertising$TV), length.out=ngrid)
grid2 = seq(min(Advertising$radio), max(Advertising$radio), length.out=ngrid)
grid3 = data.frame(TV=rep(grid1, ngrid), radio=rep(grid2, each=ngrid))

with(Advertising, plot3d(TV, radio, sales, col=2, size=5))  ## rgl plot
plot3d(grid3$TV, grid3$radio, predict(fiti1, grid3), add=T, alpha=.5)
```

There is still a clear lack of fit in model `fiti1`. We model the effect of `TV` using a quadratic function, with all three coefficients changing linearly with `radio`.

$$\begin{aligned}\text{sales} &= (\beta_0 + \beta_1\text{radio}) + (\beta_2 + \beta_3\text{radio}) \cdot \text{TV} + (\beta_4 + \beta_5\text{radio}) \cdot \text{TV}^2 \\ &= \beta_0 + \beta_1\text{radio} + \beta_2\text{TV} + \beta_4\text{TV}^2 + \beta_3\text{radio} \cdot \text{TV} + \beta_5\text{radio} \cdot \text{TV}^2 \\ &= (\beta_0 + \beta_2\text{TV} + \beta_4\text{TV}^2) + (\beta_1 + \beta_3\text{TV} + \beta_5\text{TV}^2) \cdot \text{radio}.\end{aligned}$$

Thus, models `fiti2` and `fiti2b` below are the same, although they have different names for the model items.

```
fiti2 = lm(sales ~ poly(TV, 2, raw=T) * radio, data=Advertising)
fiti2b = lm(sales ~ TV * radio + I(TV^2)*radio, data=Advertising)
summary(fiti2)$coef
summary(fiti2b)$coef
anova(fiti1, fiti2)  ## significant improvement
with(Advertising, plot(TV, resid(fiti2))); abline(h=0)      ## residual plot

with(Advertising, plot3d(TV, radio, sales, col=2, size=5))  ## rgl plot
plot3d(grid3$TV, grid3$radio, predict(fiti2, grid3), add=T, alpha=.5)
```

There is still noticeable lack of fit in model `fiti2`. We model the effect of `TV` using a cubic function, with all four coefficients changing as linear functions of `radio`.

```
fiti3 = lm(sales ~ poly(TV, 3, raw=T) * radio, data=Advertising)
summary(fiti3)
anova(fiti2, fiti3)  ## significant improvement
with(Advertising, plot(TV, resid(fiti3))); abline(h=0)      ## residual plot

with(Advertising, plot3d(TV, radio, sales, col=2, size=5))  ## rgl plot
plot3d(grid3$TV, grid3$radio, predict(fiti3, grid3), add=T, alpha=.5)
```

Every model is a significant improvement over the previous one.

```
anova(fit1, fiti1, fiti2, fiti3)
```

There is no longer a clear sign of lack of fit in `fiti3`, although the model can be further improved "significantly", as is shown below. Note that a model building process like this involves multiple testing, and may overfit the data.

```
fiti4 = lm(sales ~ poly(TV, 4, raw=T) * radio, data=Advertising)
fiti5 = lm(sales ~ poly(TV, 5, raw=T) * radio, data=Advertising)
fiti6 = lm(sales ~ poly(TV, 6, raw=T) * radio, data=Advertising)
fiti7 = lm(sales ~ poly(TV, 7, raw=T) * radio, data=Advertising)
fiti8 = lm(sales ~ poly(TV, 8, raw=T) * radio, data=Advertising)
fiti9 = lm(sales ~ poly(TV, 9, raw=T) * radio, data=Advertising)
anova(fit1, fiti1, fiti2, fiti3, fiti4, fiti5, fiti6, fiti7, fiti8, fiti9)
```

The model fiti8 has a really nice fit.

```
with(Advertising, plot(TV, resid(fiti8))); abline(h=0)      ## residual plot
with(Advertising, plot3d(TV, radio, sales, col=2, size=5))  ## rgl plot
plot3d(grid3$TV, grid3$radio, predict(fiti8, grid3), add=T, alpha=.5)
```

Cross-validation: Here the degree of polynomial is a hyperparameter. It can be determined using cross-validation (CV). We use the `caret` package to do it. Since $n = 200$ is small, the results can vary from one split to another and vary with the proportion of the test set. Here we do 10 times 5-fold CV. We first split the data into training and test sets (80%-20%).

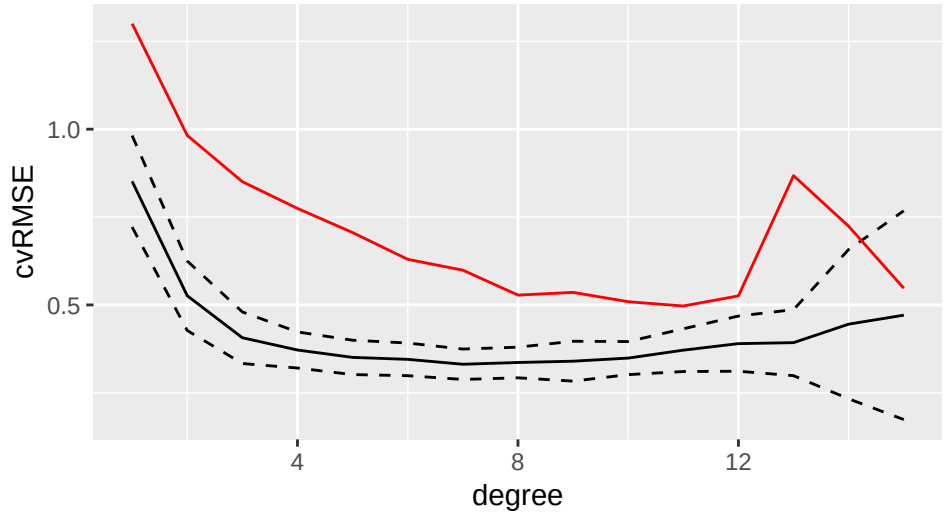
```
library(caret, quietly = T)
Advertising = read.table("Advertising.csv", header=T, sep=',', row.names=1)
set.seed(201)
inTraining = createDataPartition(Advertising$sales, p=.8, list=FALSE)
training = Advertising[inTraining, ]; dim(training)
testing = Advertising[-inTraining, ]; dim(testing)
```

We then set up 10 repeats of 5-fold CV of the training set, and compute a few measures.

```
fitControl = trainControl(method="repeatedcv", number=5, repeats=10)
degree = 1:15
cvRMSE = cvRMSESD = testRMSE = NULL
for(dd in degree) {
    polymod = train(as.formula(bquote(sales ~ poly(TV, .(dd))*radio)),
                    data=training, method="lm", trControl=fitControl)
    cvRMSE[dd] = polymod$results$RMSE
    cvRMSESD[dd] = polymod$results$RMSESD
    testRMSE[dd] = sqrt(mean((testing$sales - predict(polymod, testing))^2))
}
```

We then draw the results:

```
library(ggplot2)
polymodresults = data.frame(degree, cvRMSE, cvRMSESD, testRMSE)
ggplot(polymodresults, aes(x=degree, y=cvRMSE)) +
    geom_line() +
    geom_line(aes(y=cvRMSE+cvRMSESD), lty=2) +
    geom_line(aes(y=cvRMSE-cvRMSESD), lty=2) +
    geom_line(aes(y=testRMSE), color='red')
```



Note that model training, including selection of hyperparameters, should be performed on the training set. We include the performance results on the test set for all models to just show how well those models would perform.

3.5 In a model with a quadratic term, what can be tested?

```
fit1b = lm(sales ~ TV + I(TV^2) + radio, data=Advertising)
```

We can test: (1) if the quadratic term `TV^2` is significant after `radio` is taken into account (in two equivalent ways);

```
## (1a) t-test for I(TV^2)
summary(fit1b)  ## p-value is summary(fit1b)$coef[3,4]
## (1b) F-test for I(TV^2)
fit1 = lm(sales ~ TV + radio, data=Advertising)
anova(fit1, fit1b)
```

(2) if TV advertising cost as modeled in `fit1b` contributes significantly after `radio` is taken into account. The answer might be different when the effect of `TV` is modeled differently (e.g., without the quadratic term).

```
fitra = lm(sales ~ radio, data=Advertising)
anova(fitra, fit1b)  ## comparison when TV's contribution is modeled as TV+TV^2
anova(fitra, fit1)   ## comparison when TV's contribution is modeled as TV
```

It is meaningless to test (3) if `TV` "itself" is significant (while `TV^2` is still in the model).

3.6 ISLR 3.3

- 3.3.1: Qualitative predictors: **Choice of baseline** when there are more than two categories. When a categorical variable is converted to a factor in R, by default its levels are determined by alphabetical order. When a predictor is a factor, by default its first level is the baseline category, and indicator variables are created for all other categories. The regression results are effectively the comparisons of all other categories with the baseline category. If the baseline has a small sample size, the results can be unreliable. Thus it is often desirable to choose the most common category as the baseline (e.g., in epidemiology and genetics).

```
Credit = read.table("Credit.csv", header=T, sep=',', row.names=1)
names(Credit); dim(Credit)
str(Credit)
```

```
summary(lm(Balance ~ Gender, data=Credit))
Credit$Gender           ## Hard to see that males are coded as " Male"
summary(lm(Balance ~ I(Gender=="Male"), data=Credit))  ## incorrect
levels(Credit$Gender)   ## Now it is clear how the categories are coded
summary(lm(Balance ~ I(Gender==" Male"), data=Credit))
```

By default, the first level of a factor variable is the baseline category.

```
summary(lm(Balance ~ Ethnicity, data=Credit))$coef
levels(Credit$Ethnicity)
table(Credit$Ethnicity)
## Use "Caucasian" as the baseline
summary(lm(Balance ~ I(Ethnicity=="Asian") + I(Ethnicity=="African American"), data=Credit))$coef
```

Now suppose we only have 9 AA subjects in the data. Let us see what happens:

```
toremove = sample(which(Credit$Ethnicity == "African American"), 90)
Credittest = Credit[-toremove, ]  ## remove 90 AA subjects
table(Credittest$Ethnicity)
## AA as the baseline
summary(lm(Rating ~ Ethnicity, data=Credittest))$coef
## Caucasian as the baseline
summary(lm(Rating ~ I(Ethnicity=="Asian") + I(Ethnicity=="African American"), data=Credittest))$coef
```

- 3.3.2: **Modeling non-additive effects** between two predictors. **Interaction** is also called effect modification. One type of interaction is product interaction:

$$\begin{aligned} Y &= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \epsilon \\ &= (\beta_0 + \beta_2 X_2) + (\beta_1 + \beta_3 X_2) X_1 + \epsilon && = \alpha_0 + \alpha_1 X_1 + \epsilon \\ &= (\beta_0 + \beta_1 X_1) + (\beta_2 + \beta_3 X_1) X_2 + \epsilon && = \gamma_0 + \gamma_1 X_2 + \epsilon \end{aligned} \tag{3.31}$$

Product interaction is just one type of interaction. Because it is simple (and sometimes the only type of interaction taught in class), it has been widely used and sometimes mistakenly treated as **the** interaction. Note that when both predictors are binary, there is only one way of non-additivity (to be described in Chapter 4 notes), and the product interaction can be treated as the interaction.

When interaction terms are included, the "**main effect**" term is often a misnomer. With x_1, x_2, and $x_1 x_2$ in a model, the "main effect" for x_1 often reflects the effect of x_1 when $x_2 = 0$ (i.e., x_2 is at its baseline). For example,

$$\begin{aligned} \text{balance} &= \beta_0 + \beta_1 \text{income} + \beta_2 I(\text{student}) + \beta_3 \text{income} \cdot I(\text{student}) + \epsilon \\ &= \begin{cases} (\beta_0 + \beta_2) + (\beta_1 + \beta_3)\text{income} + \epsilon & \text{if student} \\ \beta_0 + \beta_1 \text{income} + \epsilon & \text{if not student} \end{cases} \end{aligned} \tag{3.35}$$

In this model, β_1 is just the effect of income among non-students (baseline, because non-student is coded as 0), not the overall effect of income on balance. The "main effect" for student, β_2, is the shift of intercept for a student vs. a non-student, not the overall effect of being a student. Being a student modifies not only the intercept but also the slope; β_2 could be viewed as the effect of being a student when income=0. In this model, the effect of income is β_1 for non-students and $\beta_1 + \beta_3$ for students; being a student modifies the effect of income, thus the name "effect modification".

In an interaction model, what can be tested?

```
fit2 = lm(sales ~ TV * radio, data=Advertising)
summary(fit2)
```

We can test the following: (1) if the interaction term is significant (in two equivalent ways):

```
## (1a) t-test for TV:radio
summary(fit2)      ## p-value is summary(fit2)$coef[4,4]
## (1b) F-test for TV:radio
fit1 = lm(sales ~ TV + radio, data=Advertising)
anova(fit1, fit2)  ## p-value is anova(fit1, fit2)$Pr[2]
```

(2) if TV advertising cost as modeled in `fit2` contributes significantly after `radio` has been taken into account. (This is different from the question if `TV` as modeled in `fit1` is significant.)

```
fitra = lm(sales ~ radio, data=Advertising)
anova(fit2, fitra)
```

(3) if radio advertising cost contributes significantly in the interaction model after `TV` has been taken into account.

```
fittv = lm(sales ~ TV, data=Advertising)
anova(fit2, fittv)
```

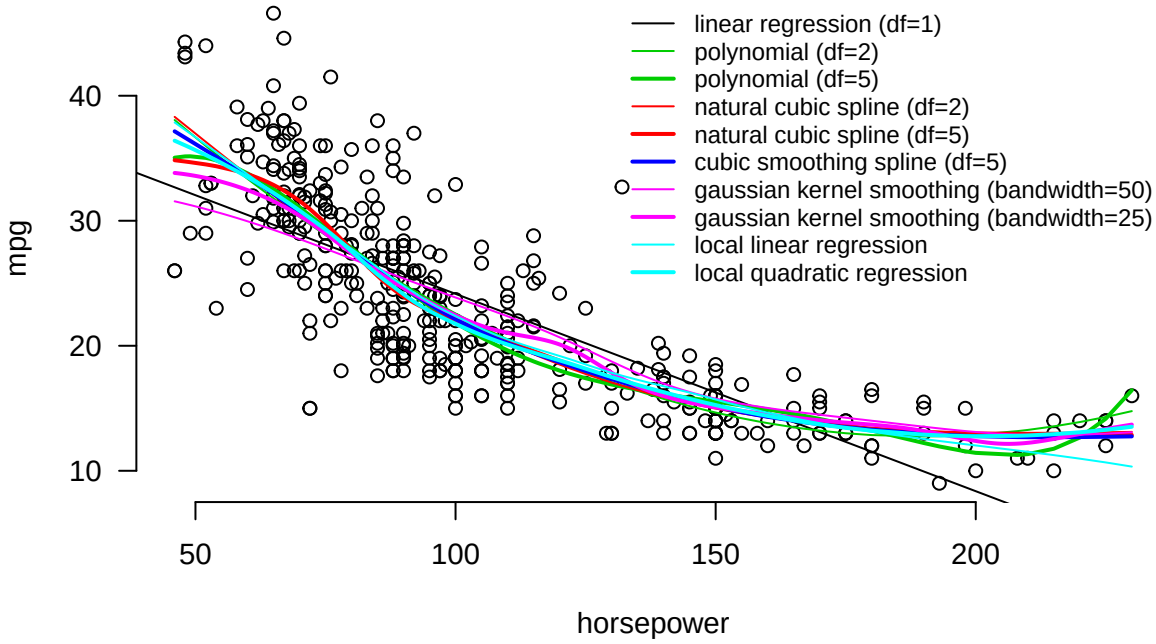
It is meaningless to test (4) if `TV` "itself" is significant (when `TV:radio` is still in the model); or (5) if `radio` "itself" is significant (when `TV:radio` is still in the model). That is, ignore the p-values for these "main effect" terms.

Modeling nonlinearity of a predictor. Polynomials are not as good as splines (ISLR 7). Here we demonstrate various nonlinear models using polynomials, natural splines, smoothing spline, kernel smoothing, and lowess.

```
library(splines)  ## for splines, ns()

with(ISLR::Auto, {
par(cex=.9)
plot(horsepower, mpg, bty='n', las=1)
```

```
## linear model fitted line in black; df=1 from x; total df=2
abline(lm(mpg ~ horsepower))
idx = order(horsepower)
## 2nd order polynomial (quadratic model) fitted curve in green; df=2 from x
points(horsepower[idx], fitted(lm(mpg ~ poly(horsepower, 2, raw=T)))[idx], type='l', col=3, lwd=1)
## 5th order polynomial fitted curve in green (thick); df=5 from x
points(horsepower[idx], fitted(lm(mpg ~ poly(horsepower, 5, raw=T)))[idx], type='l', col=3, lwd=2)
## natural cubic spline in red; df=2 from x
points(horsepower[idx], fitted(lm(mpg ~ ns(horsepower, df=2)))[idx], type='l', col=2, lwd=1)
## natural cubic spline in red (thick); df=5 from x
points(horsepower[idx], fitted(lm(mpg ~ ns(horsepower, df=5)))[idx], type='l', col=2, lwd=2)
## cubic smoothing spline in blue (thick); df=5 from x
lines(smooth.spline(horsepower, mpg, df=5), col=4, lwd=2)
## kernel smoothing
lines(ksmooth(horsepower, mpg, "normal", bandwidth=50), col=6, lwd=1)
lines(ksmooth(horsepower, mpg, "normal", bandwidth=25), col=6, lwd=2)
## loess
lines(horsepower[idx], predict(loess(mpg ~ horsepower, degree=1))[idx], col=5, lwd=1)
lines(horsepower[idx], predict(loess(mpg ~ horsepower, degree=2))[idx], col=5, lwd=2)
legend(130, 48, bty='n', cex=.8, col=c(1,3,3,2,2,4,6,6,5,5), lwd=c(1,1,2,1,2,2,1,2,1,2),
       c('linear regression (df=1)', 'polynomial (df=2)', 'polynomial (df=5)',
         'natural cubic spline (df=2)', 'natural cubic spline (df=5)',
         'cubic smoothing spline (df=5)',
         'gaussian kernel smoothing (bandwidth=50)',
         'gaussian kernel smoothing (bandwidth=25)',
         'local linear regression',
         'local quadratic regression'))
})
```



- 3.3.3: 6 issues in regression (shown in 7 Figures 3.9 – 3.15):
  1. **Nonlinearity**.
  2. **Correlation of errors** for time series data. Also for clustered and longitudinal data.
  3. **Non-constant variance**. Also called heteroscedasticity.
  4. **Outliers**: In regression, an outlier is one that has a very large discrepancy between observed and fitted values compared to the other observations. It is about y and can be flagged using studentized residuals (Figure 3.12). In other context, an outlier is an observation that is far from most other observations.

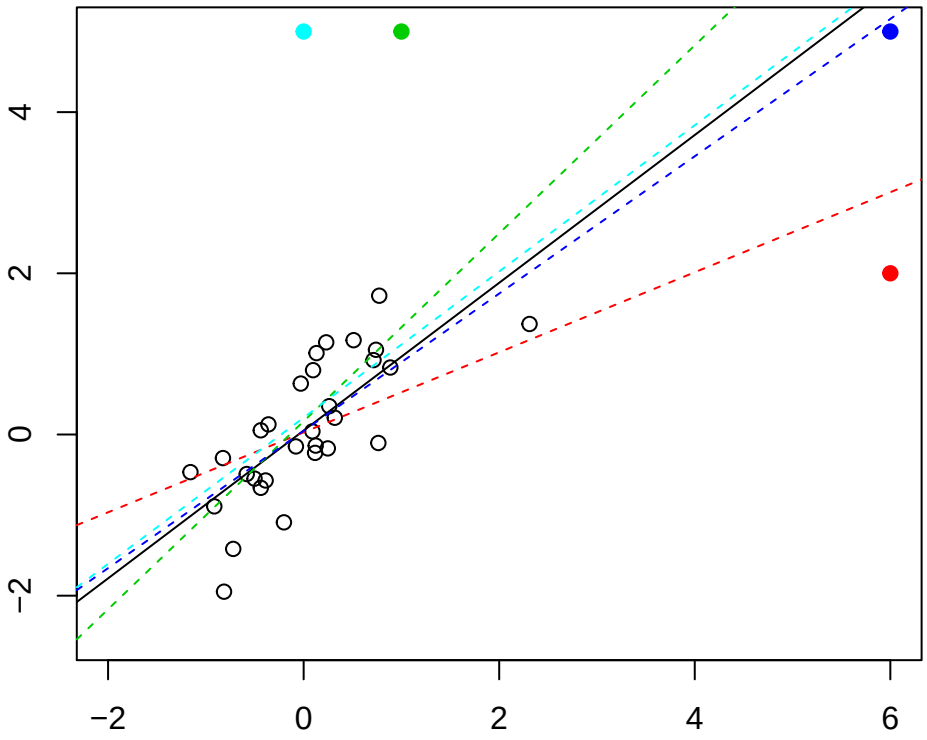
Outliers may not be errors.

5. **High-leverage points**: depends only on X. The **leverage** of x_i is the i-th value on the diagonal of the design matrix $H = X(X'X)^{-1}X'$. It is $\frac{1}{n} + \frac{1}{n-1}d_i^2$, where d_i is the Mahalanobis distance (i.e. sphered distance) between x_i and the center $\bar{x}$ of all x's. The average leverage is $\frac{p+1}{n}$.
   An **influential observation** is one that, once removed, has a large effect on parameter estimates. It can be an outlier or an observation with a high leverage.

A high-leverage point can make SE(slope) a lot smaller, giving misleadingly high "confidence" on the accuracy of the slope estimate. An outlier can make SE(slope) a lot larger. We demonstrate these two concepts below:

```
set.seed(100)
n = 30
x0 = rnorm(n)
y0 = x0 + rnorm(n, 0, .5)
mod0 = lm(y0~x0)
x1 = c(x0, 6); y1 = c(y0, 2); mod1 = lm(y1~x1)  ## add (6,2), high leverage
x2 = c(x0, 1); y2 = c(y0, 5); mod2 = lm(y2~x2)  ## add (1,5), outlier
x3 = c(x0, 6); y3 = c(y0, 5); mod3 = lm(y3~x3)  ## add (6,5), high leverage, not influential
x4 = c(x0, 0); y4 = c(y0, 5); mod4 = lm(y4~x4)  ## add (0,5), outlier, not influential

par(mar=c(2,2,1,1))
plot(x0, y0, xlim=c(-2, 6), ylim=c(-2.5, 5)); abline(mod0)
points(6, 2, col=2, pch=19); abline(mod1, col=2, lty=2)
points(1, 5, col=3, pch=19); abline(mod2, col=3, lty=2)
points(6, 5, col=4, pch=19); abline(mod3, col=4, lty=2)
points(0, 5, col=5, pch=19); abline(mod4, col=5, lty=2)
```



- 6. **Collinearity**. Variance inflation factor $\text{VIF}(x_j) = (1 - R^2_{x_j|X_{-j}})^{-1}$. The higher $R^2_{x_j|X_{-j}}$ the higher VIF. Rule of thumb: VIF>5 is high collinearity. VIF is also $var_F(\hat{\beta}_j)/var_j(\hat{\beta}_j)$, the ratio of variances of $\hat{\beta}_j$, one from the full model and one from the model with only x_j.

```
Credit = read.table("Credit.csv", header=T, sep=',', row.names=1)  ## simulated data
names(Credit); dim(Credit)
pairs(Credit[, c(1:3,11)])
```

```
cor(Credit[, c(1:3,11)])
car::vif(lm(Balance ~ Limit + Rating + Income, data=Credit))  ## VIF
```

When viewed from a certain angle in 3D, two layers are present (simulation artifact?).

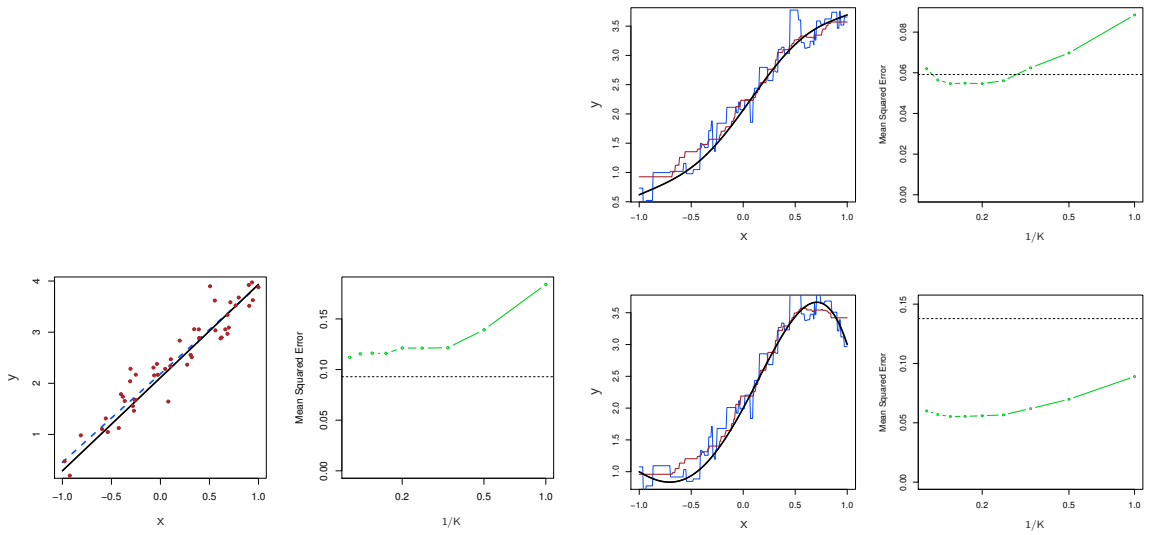
```
with(Credit, rgl::plot3d(Rating, Income, Balance))
```

3.7 ISLR 3.4

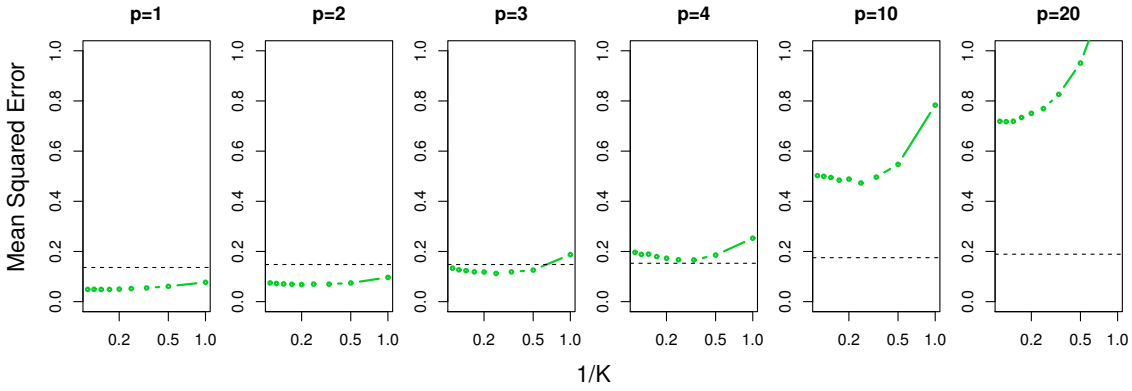
Answer the 7 questions at the beginning of chapter. For Q1, I would also use the total advertising cost.

3.8 ISLR 3.5

Linear regression vs. KNN (Figures 3.18–3.19). The performance of linear regression is the horizontal dashed line.



Curse of dimensionality: In Figure 3.20, only one predictor is relevant to the outcome in a non-linear way, and all others are noise.



In general, for local estimation methods (e.g., KNN), the more noise features the worse performance. When many noise features are present, two points can be "close" because they have similar values for the noise features, not the true predictors. Points that are close with respect to the true predictors may appear to be far from each other because of the noise features. We try to demonstrate this below. We firt generate 150 observations with 3 features, in which only feature x_1 has an effect on y.

```
set.seed(4719); N=150; p=3  ## setup: N obs, p features
x1 = runif(N); y = (x1 + rnorm(N, 0, .2)) > 0.5  ## only x1 has effect on y
xx = cbind(x1, matrix(runif(N*(p-1)), N))  ## all features: x1 plus p-1 noise features

mod1 = glm(y~x1, family=binomial)  ## logistic regression on x1 (not correct, but not bad)
mod2 = glm(y~xx, family=binomial)  ## logistic regression on all features (not bad, too)
summary(mod1)
summary(mod2)
```

Suppose we will do 9-NN. We identify the 9 observations closest to (0.55) on x_1 and the 9 observations closest to $(0.55, 0.5, 0.5)$ on $(x_1, x_2, x_3$.

```
kk = 9; xeval = c(0.55, rep(0.5, p-1))  ## 9-NN at xeval=(0.55, 0.5, ..., 0.5)
distpd = apply(xx, 1, function(x) sqrt(sum((x-xeval)^2)))  ## p-dim distance to xeval
mysetpd = order(distpd)[1:kk]  ## positions of the kk observations closest to xeval
dist1d = abs(x1-xeval[1])      ## 1-dim distance on x1 to xeval[1]
myset1d = order(dist1d)[1:kk]  ## positions of the kk observations closest to xeval[1]
myrest = (1:N)[-c(mysetpd, myset1d)]

library(rgl)
mycol = ifelse(y==1, 2, 1)
plot3d(x1, x2, x3, type='n')  ## plot x1 and two noise predictors
pch3d(0.55, 0.5, 0.5, pch=1, radius=.05)
points3d(x1[myrest], xx[myrest,2], xx[myrest,3], col=mycol[myrest], size=4)
text3d(x1[mysetpd], xx[mysetpd,2], xx[mysetpd,3], col=mycol[mysetpd], "p", cex=1)
text3d(x1[myset1d], xx[myset1d,2], xx[myset1d,3], col=mycol[myset1d], "1", cex=1)
```

3.9 Chapter 3 R Labs

For a linear model, the following functions can be used: `confint()`, `predict()`, `plot()`, `residuals()`, `rstudent()`, `hatvalues()`, `car::vif()`.

3.10 Assignment

1. **Homework**: ISLR Chapter 3 Exercises 9 (`Auto` dataset in ISLR), 15 (`Boston` dataset in MASS)
2. **Homework**: Play with the Titanic dataset to predict survival status. Do all model training and tuning on the training set. The data are available in an R package `titanic`; same data in .csv format are available from https://www.kaggle.com/c/titanic/data . The test set does not contain the outcome. R also has an object `Titanic`, which is a simple data tabulation with a slight discrepancy with the `titanic` package: `Titanic` indicates there were 1316 passengers, while the `titanic` package contains 1309 (891 in training set and 418 in test set). In the R package, ignore the two objects `titanic_gender_class_model` and `titanic_gender_model`. They are probably leftover objects from the package creator's own models. There are many analysis demonstrations of this dataset online, both in R and Python. Do not look at them. Try to get as much as possible yourself in 1 week. Then look at other poeple's solutions online (by just googling). Summarize what you learned from them and what you did better than most of them. Turn in your final prediction model and the predictions on the test set (as a 418×2 object or a file with two columns: PassengerId, Survived).