Based on the table, suggest one way to reduce the number of factor levels.

Solution. We can see that the target variable has similar mean and median within levels 1 and 2, and levels 3 and 4 also involve similar means and medians. Therefore, one way to reduce the number of factor levels is to combine levels 1 and 2 as one group and levels 3 and 4 as another group. As level 5 has only 20 observations, we may fold it into levels 3 and 4 due to the low mean and median. Overall, the two new factor levels are $\{1,2\}$ and $\{3,4,5\}$, which we can relabel as levels A and B. □

The dimension reduction methods above are generic ones that apply regardless of the predictive model you are using. In Subsection 3.2.4 and Chapter 5, we will introduce algorithmic feature selection methods specific to linear models and decision trees, respectively.

3.2 Linear Models: Theoretical Foundations

In this section we provide an overview of linear models for prediction and show how they embody the fundamental concepts in predictive analytics introduced in Section 3.1. If you have taken Exam SRM, then you will probably be familiar with some, if not most of the material in this section, but our overview here is not a general one, but is geared towards Exam PA with a heavy predictive analytic flavor. Do note that many of the considerations presented in this section are also useful for GLMs to be studied in the next chapter.

EXAM NOTE

In Exam PA, it is true that you will ask R to implement all the model fitting, model selection, and model evaluation without having to worry about the technical details behind the scenes, but you are still expected to understand *conceptually* what is being done in each step. In a number of exam tasks, you are asked to provide a high-level description of your predictive models, feature selection process, and model validation process.

3.2.1 Model Formulation

Model equation. Linear models postulate that the target variable Y, which is assumed to be continuous, is related to p predictors $X_1, X_2, \ldots, X_p$ via the approximately[iii] linear relationship

$$Y = \underbrace{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p}_{\text{regression function}} + \varepsilon, \tag{3.2.1}$$

where:

- p is the number of predictors.
- $\beta_0, \beta_1, \ldots, \beta_p$ are unknown regression coefficients (or parameters).

[iii]The linear relationship is only approximate due to the presence of the random error ε.

- ε is the unobservable zero-mean random error term that accounts for the fluctuation of Y about its mean.

When $p = 1$, (3.2.1) is called a *simple linear regression model* (perhaps because it is "simple!"); when $p \geq 2$, the term *multiple linear regression model* is used to recognize the fact that there are multiple predictors. In general, we refer to (3.2.1) as a *linear model*, with Y regressed on $X_1, X_2, \ldots, X_p$, because the expected value of Y, captured by the signal function

$$\mathbb{E}[Y] = f(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p,$$

is linear in the regression coefficients $\beta_0, \beta_1, \ldots, \beta_p$. It is, however, not necessarily linear in the predictors. For example, the quadratic model $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$, which is quadratic in X but linear in $\beta_0, \beta_1, \beta_2$, is still considered a linear model.

For convenience, we refer to

- β_0 as the *intercept*, which is the expected value of Y when all X_j's are zero.
- β_j as the regression coefficient or the *slope* coefficient attached to the jth predictor for $j = 1, \ldots, p$. These coefficients quantify the expected effect of the predictors on Y, *holding all other predictors fixed* (for the precise interpretations, see Subsection 3.2.3).

In Exam SRM, you may have learned a lot of inference tools such as hypothesis tests and confidence intervals involving the regression coefficients β_j's. These tools allow us to infer, on the basis of the observed (training) data, the characteristics of the underlying population. In Exam PA, however, our main focus is on using the linear model to make predictions for the target variable. Instead of doing classical analysis like performing hypothesis tests and constructing confidence intervals, you will spend most of your time building different linear models and evaluating their predictive performance.

Model fitting by ordinary least squares. To estimate the unknown coefficients $\beta_0, \beta_1, \ldots, \beta_p$, we assume that we are given a training set of n independent pairs of observations $\{(Y_i, \mathbf{X}_i)\}_{i=1}^n = \{(Y_i, X_{i1}, X_{i2}, \ldots, X_{ip})\}_{i=1}^n$ governed by

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} + \varepsilon_i, \quad i = 1, \ldots, n, \tag{3.2.2}$$

which can be considered the sample version of (3.2.1) with the subscript i inserted. For notational consistency, in the remainder of this manual very often the subscript i is used to index the observations (from 1 to n) and the subscript j is used to index the predictors (from 1 to p). The dataset can be displayed in the form of a data frame (recall Subsection 1.2.3) as in Table 3.1, where observations are shown across the rows of the table and the corresponding predictor variable values are shown across the columns.

When a linear model in the form of (3.2.2) is used, it is often assumed that the random error terms follow a zero-mean normal distribution with a common variance σ^2, i.e.,

$$\varepsilon_i \overset{\text{i.i.d.}}{\sim} \mathrm{N}(0, \sigma^2). \tag{3.2.3}$$

In statistical parlance, the common variance assumption is known as the *homoscedastic* assumption. Central to much of the theory of linear models, the normality assumption effectively implies that the target variable Y is also normally distributed as

$$Y_i \sim \mathrm{N}(\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip}, \sigma^2).$$

	Target	Predictors			
Observation	Y	X_1	X_2	$\cdots$	X_p
1	Y_1	X_{11}	X_{12}	$\cdots$	X_{1p}
2	Y_2	X_{21}	X_{22}	$\cdots$	X_{2p}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
n	Y_n	X_{n1}	X_{n2}	$\cdots$	X_{np}

Table 3.1: Typical data structure of a linear model.

In Exam PA, whenever the term "linear model" is used, we are implicitly assuming that the random errors are normally distributed in accordance with (3.2.3), unless otherwise stated.

While there are many different ways to estimate the regression coefficients of a linear model, one of the most popular is the *ordinary least squares approach*. As its name suggests, this approach consists in choosing the estimates of the β_j's to make the sum of the "squared" differences between the observed target values and the fitted values under the model the "least":

$$\min_{\beta_0,\beta_1,\ldots,\beta_p} \sum_{i=1}^{n} [Y_i - \underbrace{(\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip})}_{\text{candidate fitted value}}]^2.$$

The solutions, denoted by $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ and called the *ordinary least squares estimators*, are readily available from built-in functions in R, so we shall not concern ourselves with their mathematical formula.[iv] Do note that when the random errors are normally distributed, the least squares estimators $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ coincide with the maximum likelihood estimators.

Matrix representation of linear models. For both theoretical and practical reasons, it is instructive to recast the equation of a linear model, (3.2.2), compactly in terms of matrix notation:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{3.2.4}$$

or, on a component-wise basis,

$$\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix} = \begin{pmatrix} 1 & X_{11} & X_{12} & \cdots & X_{1p} \\ 1 & X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_{n1} & X_{n2} & \cdots & X_{np} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix},$$

where:

- $\mathbf{Y}$ is the $n \times 1$ response vector.
- $\mathbf{X}$ is the *design matrix* containing values of the predictors, with the first column of 1's denoting the intercept and X_{ij} denoting the value of the jth predictor in the ith observation, for $i = 1, \ldots, n$ and $j = 1, \ldots, p$.
- $\boldsymbol{\beta}$ is the vector of $p + 1$ regression coefficients.

[iv] By matrix calculus, the formula for the vector of least squares estimates is given by $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1}(\mathbf{X}^\top \mathbf{y})$. Remember this from Exam SRM?

- ε is the $n \times 1$ vector of random errors.

Although there will hardly be any instances in Exam PA in which you will manipulate matrices by brute force, we will see in Subsection 3.4.4 that some of the R functions for fitting linear models require an explicit specification of the design matrix. For this reason, keeping the notion of a design matrix in mind will be useful, at least from a programming point of view.

Model quantities. As soon as the least squares estimates have been computed, a number of useful quantities about the fitted model can be computed. Some of them are confined to the training set, while some can be computed on both the training and test sets. Regardless, all of them are part of the standard computer output whenever a linear model is fitted in R, so we will focus on what they are and what they try to capture instead of their computational details.

- *Predicted value* $\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1^* + \cdots + \hat{\beta}_p X_p^*$, where $(1, X_1^*, \ldots, X_p^*)$ is a vector of predictor values of interest

  To predict the value of the target variable at a particular vector of predictor values, say $(1, X_1^*, \ldots, X_p^*)$, we simply replace the regression coefficients in the model equation (3.2.2) by their ordinary least squares estimates and the random errors by their theoretical mean level, which is zero, yielding the predicted value

  $$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1^* + \cdots + \hat{\beta}_p X_p^*.$$

  Here $(1, X_1^*, \ldots, X_p^*)$ can be a general vector of predictor values of interest, not necessarily those in the training set. When $(1, X_1^*, \ldots, X_p^*)$ indeed is an observation in the training set, say the ith observation, the predicted value is more commonly known as the *fitted* value of the ith training observation.

  Ideally, we would like the fitted (or predicted) value of each observation in the training set to be sufficiently close to the corresponding observed target value, but not overly close in order to avoid overfitting. On the test set, however, we would like the predicted values to be as close as possible to the observed target values so that our linear model is as predictive as possible.

- *Residual* $e_i = Y_i - \hat{Y}_i$

  Residuals, also called raw residuals, represent the discrepancy between the observed target value and the corresponding predicted value on either the training set or test set. These quantities often hint at patterns in the underlying population that our model fails to capture and, as we will discuss in the next subsection, will play a crucial role in model diagnostics.

- *t-statistic* $t(\hat{\beta}_j) = \hat{\beta}_j / (\text{standard error of } \hat{\beta}_j)$

  The t-statistic for a particular regression coefficient is defined as the ratio of the corresponding least squares estimate to its estimated standard deviation, often called the *standard error*. It can be used to test the hypothesis $H_0 : \beta_j = 0$,[v] which means that the jth predictor can be dropped out of the linear model due to its insignificance *in the presence of all other predictors*. To quantify the amount of evidence contained in the t-statistic against the null hypothesis, we often appeal to the notion of the *p-value*, which is the probability that the test statistic takes a value which is as extreme as or more extreme than its observed value, given that the

[v] Slide 66 of Module 6 says that "the null hypothesis is that the corresponding *predictor* is equal to 0." It should be the regression *coefficient* that should be set to 0.

null hypothesis is true. The smaller the p-value, the more evidence we have against the null hypothesis in favor of the alternative hypothesis and the more important the predictor that is being tested.

- *Coefficient of determination* R^2

  The coefficient of determination R^2 and the F-statistic are defined in terms of more involved formulas which are of limited use in Exam PA, so here we will concentrate on what they are for. In words, R^2 is the proportion of the variation of the target variable that can be explained by the fitted linear model. On the training set, R^2 is a measure on a scale of 0 to 1 of the goodness of fit of a linear model, with a value of R^2 close to 1 being indicative of a model with a good fit.

  One problem with R^2 is that its value always increases[vi] when a predictor, however good or bad it is, is added. This motivates the use of an adjusted version of the R^2 that adds a penalty for the number of parameters of a linear model. For the inclusion of a predictor to increase the value of the adjusted R^2, the improvement in the model quality has to outweigh the penalty due to increasing the number of model parameters.

- *F-statistic*

  Unlike the t-statistic, which is for a single regression coefficient, the F-statistic is for testing $H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$, which assesses the joint significance of the entire set of predictors (excluding the intercept), against the alternative H_a : At least one of $\beta_1, \beta_2, \ldots, \beta_p$ is non-zero. The corresponding p-value is a measure of the evidence against H_0 in favor of H_a. If H_0 is rejected, then we have strong evidence that *at least one* of the p variables is an important predictor for the target variable. However, the test itself does not say which variables are really predictive—further analysis is required.

In Exam PA, traditional statistical inference tools such as t-statistics and F-statistics are not emphasized. They are introduced here because they figure prominently in the model summary produced by R and they are covered in the PA e-learning modules, so it is a good idea to know what they mean. In the case study sections of this chapter, we will be mostly concerned with making predictions for different linear models and comparing their predictive performance.

3.2.2 Model Evaluation

Having fitted a linear model, it is natural for us to evaluate its quality and diagnose it for any abnormalities. If any apparent deficiencies are present, the model may be refined taking these deficiencies into account.

Performance metrics. The target variable of a linear model is numeric (in fact, continuous), so a commonly used measure of the predictive performance of the model on unseen data is the *test* RMSE defined in Subsection 3.1.2. Other performance measures include the loglikelihood and the R^2 computed on the *test* set. In fact, under the assumption that the random errors follow (3.2.3), the test RMSE, R^2, and test loglikelihood are all functions of the sum of squared prediction errors $\sum_{i \in \text{test set}} (Y_i - \hat{Y}_i)^2$ on the test set and thus are equivalent performance metrics. Among the three, the RMSE is most often used because it is the most interpretable (due to it having the same unit as the target variable) and gives us a sense of the typical magnitude of the prediction error.

[vi] More precisely, R^2 never drops when a new predictor is added to a linear model.

For linear models and, more broadly, GLMs, the target variable is known to follow a particular distribution (e.g., normal, Poisson, gamma, binomial), so we can also employ performance metrics based on penalized likelihood. Here are two such metrics commonly used to rank competing models:

- *Akaike Information Criterion (AIC):* It is defined as $-2l + 2p$, where l is the loglikelihood of the model on the training set and p is the number of parameters of the model. The AIC balances the goodness of fit of a model to the training data (the higher the value of l, the better the fit) with the complexity of the model captured by the number of parameters, which acts as a penalty term penalizing an overfitted model. The *smaller* the AIC, the better the model, so when the AIC is used as the model selection criterion, the best model is the one with the *smallest* AIC.

- *Bayesian Information Criterion (BIC):* It is defined as $-2l + \ln(n_{\text{tr}})p$, where n_{tr} is the size of the training set, and shares the same spirit as the AIC. The smaller the BIC, the better the model.

Both the AIC and BIC demand that for the inclusion of an additional feature to improve the performance of a model, the feature must increase the (training) loglikelihood by at least a specific amount. This amount, which penalizes a model by its complexity and accounts for overfitting, is 2 for the AIC and the logarithm of the number of training observations, $\ln(n_{\text{tr}})$, for the BIC. For almost all reasonable values of n_{tr},[vii] the penalty per parameter is higher for the BIC than for the AIC, so the BIC is more stringent than the AIC for complex models and, as we will see in Subsection 3.2.4, represents a more conservative approach to feature selection.

Model diagnostics. In addition to evaluating its predictive performance, we can also diagnose a linear model with respect to the extent to which the model assumptions, particularly (3.2.3), are satisfied. In general, *model diagnostics* refers to a set of quantitative and graphical tools that are used to identify evidence against the model assumptions and, if found, to refine the specification of the model in an effort to improve its adequacy. For this purpose, the residuals of the linear model serve as proxies for the unobservable random errors and will play a pivotal role.

If you recall what you learned in Exam SRM, commonly used quantitative diagnostic tools for linear models include:

- Residuals, as defined in the preceding subsection (taken individually or as a group, to detect abnormality with respect to the target variable values), and their standardized counterparts

- Leverages (to detect abnormality with respect to predictor variable values)

- Cook's distance (to detect influential observations, which are abnormal with respect to both the target variable values and predictor variable values)

In Exam PA, these quantitative tools are de-emphasized. In fact, there is hardly any mention of leverages and Cook's distance in the PA e-learning modules, and only graphical tools are discussed. Two of the most important plots, which were referred to in the June 2019 PA exam model solution, are: (You will have a better idea of how these plots look like in Section 3.4.)

- *"Residuals vs Fitted" plot:* As you can tell from its name, this plot graphs the residuals of the model against the fitted values (or the predicted values on the training set). This plot can be

[vii] For $\ln(n_{\text{tr}}) > 2$, or $n_{\text{tr}} > e^2 = 7.3891$, to be precise.

used to check for the model specification as well as the homogeneity of the error variance. If the relationship between the target variable and the predictors is correctly described according to the model equation (3.2.1) and the random errors possess the same variance, then the residuals, as a function of the fitted values, should display no discernible patterns and spread symmetrically in either direction (positive and negative). Any systematic patterns (e.g., U-shape) and significantly non-uniform spread in the residuals are symptomatic of an inadequate model equation and *heteroscedasticity* (i.e., the random errors have non-constant variance), respectively.

- *"Normal Q-Q" plot:* This plot, with "Q" standing for "quantiles," graphs the quantiles of the standardized residuals (which are the residuals divided by their standard error) against the theoretical standard normal quantiles and can be used for checking the normality of the random errors. If the random errors are indeed normally distributed, then the residuals, as proxies for the random errors, should also be normally distributed. In this case, the points in the Q-Q plot are expected to lie closely on a 45° straight line passing through the origin. Systematic departures from the 45° straight line, often in the two tails, suggest that the normality assumption is not entirely fulfilled.

3.2.3 Feature Generation

The preceding two subsections discussed how a given linear model can be analyzed and evaluated and what model quantities can be computed. This important subsection addresses the important question of how to specify a linear model in the first place. Having come up with a list of potentially useful predictors (usually directly provided in the exam), we can fine-tune the form we want these predictors to enter the model equation with the goal of improving the flexibility of our model. In the course of doing so, we generate useful features that can be added to the model to enhance its prediction accuracy.

The representation of the predictors in question depends on whether they are numeric or categorical. The distinction between numeric and categorical variables often has important implications for modeling (especially for GLMs), so we will treat these two types of variables separately.

Numeric predictors. A numeric predictor X_j is rather easy to handle in a linear model. For simplicity, we suppose that X_j is continuous. In the simplest form of the model, X_j can be included as-is, with a single regression coefficient attached, so that the model equation is

$$Y = \beta_0 + \beta_j X_j + \begin{matrix}\text{terms not} \\ \text{involving } X_j\end{matrix} + \varepsilon.$$

In this case, the regression coefficient $\beta_j = \partial \mathbb{E}[Y]/\partial X_j$ can be interpreted as

> the *expected* change in the target variable Y per unit increase in X_j, *holding all other predictors fixed.*

EXAM NOTE

Interpreting the results of a fitted predictive model is one of the most important items in Exam PA and therefore a skill that you should definitely master. For linear models, a useful way to make sense of their output is to interpret the sign and magnitude of their coefficient estimates. The precise statements depend closely on the specific type (numeric or categorical) of the predictors in question.

In situations where the relationship between Y and X_j does not appear to be linear, it may be desirable to relax the linear form of the regression function and expand it to higher powers of X_j using *polynomial regression*, i.e.,

$$Y = \beta_0 + \boxed{\beta_1 X_j + \beta_2 X_j^2 + \cdots + \beta_m X_j^m} + \begin{matrix}\text{terms not} \\ \text{involving } X_j\end{matrix} + \varepsilon,$$

for some positive integer m, with $X_j, X_j^2, \ldots, X_j^m$ treated as separate features. In this polynomial model, the expected effect of X_j on Y is given by

$$\frac{\partial \mathbb{E}[Y]}{\partial X_j} = \beta_1 + 2\beta_2 X_j + \cdots + m\beta_m X_j^{m-1},$$

which now depends on the value of X_j. By means of polynomial regression, we are able to take care of substantially more complex relationships between the target variable and predictors than linear ones. The more polynomial terms are included, the more flexible the fit that can be achieved. On the downside, the regression coefficients become much more difficult to interpret. It is now difficult to give a precise meaning to, for example, β_1. We cannot say β_1 is the expected change in Y due to a unit change in X_j, holding all other variables fixed—the other polynomial terms $X_j^2, \ldots, X_j^m$ cannot be kept fixed when X_j varies!

An alternative to polynomial regression for incorporating non-linearity into a linear model is binning, which is very briefly mentioned in the PA e-learning modules. Instead of looking at the exact values of X_j, we create a new categorical predictor whose levels are defined as non-overlapping intervals over the range of X_j. Categorical predictors are discussed next.

Categorical predictors. The representation of categorical predictors in a linear model is more delicate. Because categorical predictors are merely (possibly non-numerical) labels of group membership, they cannot be handled algebraically in the same way as quantitative predictors and require special treatment. For example, if we use smoking status as a categorical predictor defined as

$$\texttt{Smoking} = \begin{cases} \texttt{Smoker} \\ \texttt{Non-smoker} \\ \texttt{Unknown} \end{cases},$$

we cannot enter it directly to the model equation and use

$$Y = \beta_0 + \beta_1 \times \texttt{Smoking} + \varepsilon.$$

The three products $\beta_1 \times \texttt{Smoker}$, $\beta_1 \times \texttt{Non-smoker}$, and $\beta_1 \times \texttt{Unknown}$ do not make sense!

To properly incorporate a categorical predictor into a linear model, we have to perform the process of *binarization*,[viii] which is an important feature generation method. As its name suggests, binarization turns a given categorical predictor into a collection of "binary" variables (also known as *dummy* or *indicator variables*), each of which serves as an indicator of one and only one level of the categorical predictor. More precisely, for each level of the categorical predictor, the process creates a binary variable that equals 1 in every row of the dataset where the categorical predictor assumes that level, and 0 otherwise. This way, the binary variable highlights whether each observation in the dataset does or does not possess a certain characteristic.

It is easier to see this with an example. For the three-level `smoking` variable above, we can binarize it by generating the three binary variables `SmokingSmoker`, `SmokingNon-smoker`, and `SmokingUnknown` (their long names follow the naming convention in R when categorical variables are binarized) defined as follows:

	Value		
Level of `Smoking`	`SmokingSmoker`	`SmokingNon-smoker`	`SmokingUnknown`
`Smoker`	1	0	0
`Non-smoker`	0	1	0
`Unknown`	0	0	1

Here `SmokingSmoker`, `SmokingNon-smoker`, and `SmokingUnknown` are indicators of the levels `Smoker`, `Non-smoker`, and `Unknown`, respectively. If we have a small dataset given by

Observation	Level of `Smoking`
1	`Smoker`
2	`Unknown`
3	`Non-smoker`
4	`Non-smoker`
5	`Smoker`
6	`Non-smoker`

then the binarized dataset is

Observation	`SmokingSmoker`	`SmokingNon-smoker`	`SmokingUnknown`
1	1	0	0
2	0	0	1
3	0	1	0
4	0	1	0
5	1	0	0
6	0	1	0

Since the three binary variables are numeric in nature (equal to either 0 or 1), they can be manipulated algebraically and entered into the model equation as separate features to collectively represent `Smoking`. One possible configuration is

$$Y = \beta_0 + \beta_1 \times \texttt{SmokingSmoker} + \beta_2 \times \texttt{SmokingUnknown} + \varepsilon,$$

with `SmokingNon-smoker` left out. You may wonder: Why do we not use all of the three binary variables? Observe that the three binary variables are perfectly linearly related as

$$\texttt{SmokingSmoker} + \texttt{SmokingNon-smoker} + \texttt{SmokingUnknown} = 1.$$

[viii] This term is probably new to you even if you have taken Exam SRM.

Intuitively, it suffices for us to know the values of any two of these three binary variables and the value of the remaining one can be deduced from the perfect linear relationship above, so including all three binary variables in the model equation results in a duplication of information. Technically, the perfect linear relationship among the predictors in a linear model will destabilize the model fitting procedure and undermine the precision of the coefficient estimates. When dealing with a categorical predictor in linear models (and, more generally, GLMs), one of its levels should be left out and termed the *baseline* (or *reference*) level. That is, a categorical predictor with k levels should be represented by $k-1$ binary variables. All of the binary variables that correspond to the non-baseline levels then enter the model equation, each with a single regression coefficient assigned. In this context, these coefficients represent the *expected* difference between the value of the target variable when the categorical predictor is in the corresponding non-baseline level and that when the categorical predictor is in the baseline level, *holding all other predictors fixed.*

How do we go about choosing the baseline level?[ix] In R, the convention is to select the level that comes first alpha-numerically as the baseline level (e.g., `Non-smoker` for the `Smoking` variable), although the default can be overridden to set the baseline level to a particular level with respect to which comparisons of interest are made. Note that while the choice of the baseline level has no effect on the predicted values generated by the linear model or the overall quality of the model, it does affect the interpretation of the coefficient estimates (because you are changing the reference category) and, more importantly, the results of hypothesis tests for removing individual factor levels. If there is no obvious choice for the baseline level, it is a good idea to set the baseline level to the level that carries the most observations.

It deserves mention that although most of the R functions for fitting linear models are capable of dealing with categorical variables by internal binarization (i.e., they will binarize without explicitly being asked to), the ability to perform explicit binarization has important merits when it comes to feature selection, as we will see in Subsection 3.4.2.

[IMPORTANT!] Interactions. The notion of interactions between predictors figures prominently in Exam PA. Out of the four released PA projects, three, including the December 2018 and June 2019 exams, deal with interactions in some way, so it is essential that you have a firm grasp of this modeling concept.

To motivate the need for interactions, recall that the equation of a generic linear model is

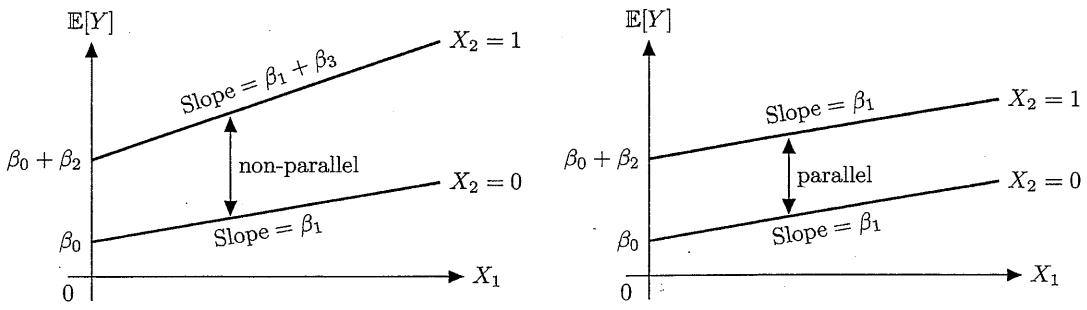
$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon.$$

In this basic form of a linear model, it is assumed that the expected effect of each predictor on the target variable Y is independent of the values of other predictors.[x] In other words, the predictors act on the target variable in an additive or separate fashion. This is likely not the case for many real-world datasets, where two or more predictors interact and have a *joint* effect on the target variable, or equivalently, the expected effect of one predictor on the target variable *depends on the value (or level) of another predictor.*[xi] Here is a simple but interesting example mentioned in ISLR:

[ix] In September 2019, the SOA added a new slide in Module 6 (Slide 130) discussing considerations with factor variables.

[x] If a numeric predictor is represented by polynomial terms, then we group all of those polynomial terms and consider their joint effect; do the same for all the binary variables of a categorical predictor. In either case, the joint effect of the grouped features is independent of the values of other features.

[xi] Slide 54 of PA Module 6 says that "[a]n interaction occurs when the response depends on the relationship between a combination of features." This is not precise enough.



(a) With interaction ($\beta_3 \neq 0$). (b) Without interaction ($\beta_3 = 0$).

Figure 3.2.1: Graphical illustration of a linear model with interactions between continuous and categorical predictors.

> If Y, X_1, and X_2 represent the number of units produced, number of production lines, and number of workers in a factory, then the increase in $\mathbb{E}[Y]$ due to a unit increase in X_1 is likely to depend on the number of workers operating the lines. If there are no workers, then increasing the number of production lines will not raise production (much). The more workers we have, the greater the impact of opening more lines on productivity.

As we will see in the case studies sections of this chapter, the ability to detect and accommodate interaction effects can go a long way towards designing an effective predictive model.

We now discuss how to detect interactions between different kinds of predictors graphically and, once detected, how to tweak a linear model to incorporate interaction effects.

- *Interactions between continuous and categorical predictors:* The interaction between a continuous predictor and a categorical predictor has a nice geometric representation that can be identified rather easily. To see this, we consider a linear model with one continuous predictor X_1 and a binary variable X_2 (representing a certain binary predictor). Apart from the regression coefficients β_1 and β_2 assigned respectively to X_1 and X_2, an additional coefficient β_3 is attached to the product term X_1X_2, treated as an extra feature. The model equation is

$$\begin{aligned}\mathbb{E}[Y] &= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 \boxed{X_1 X_2} \\ &= \begin{cases} \beta_0 + \beta_1 X_1, & \text{if } X_2 = 0, \\ (\beta_0 + \beta_2) + (\beta_1 + \beta_3) X_1, & \text{if } X_2 = 1, \end{cases}\end{aligned}$$

  which carries two separate straight lines with respect to X_1, one for $X_2 = 0$ and one for $X_2 = 1$. Note that the two lines possess not only different intercepts, but also different slopes with respect to X_1 due to the *interaction term* X_1X_2; see the left panel of Figure 3.2.1. For the baseline group (i.e., $X_2 = 0$), the effect of a unit increase in X_1 is to increase $\mathbb{E}[Y]$ by β_1, while the increment in $\mathbb{E}[Y]$ becomes $\beta_1 + \beta_3$ for the other group. If $\beta_3 \neq 0$, then the expected effect of X_1 on Y differs according to whether $X_2 = 0$ or $X_2 = 1$, a manifestation of interaction; if $\beta_3 = 0$, then interaction vanishes and we are simply fitting two parallel lines (with different intercepts) to the data; see the right panel of Figure 3.2.1.

  To assess the extent of interaction graphically, we can use `ggplot2` to make a scatterplot of the target variable against X_1, use the `color` aesthetic to color-distinguish the data points

according to the levels of X_2, and use `geom_smooth()` to fit a separate regression line to each group. If the two fitted lines differ considerably with respect to their slope, then the interaction effect is remarkable and should be taken care of in the linear model.

Example 3.2.1. (SRM-type question with implications for Exam PA: Interaction between a numeric predictor and a multi-level categorical predictor) Consider a continuous target variable Y, a continuous predictor X_1, and a three-level categorical predictor represented by two binary variables X_2 and X_3.

Determine which of the following linear models best accommodates the interaction effect between X_1 and the categorical predictor. Explain how the interaction effect manifests.

(If there are two or more models that can accommodate the interaction effect, choose the simpler/simplest model.)

(A) $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \varepsilon$

(B) $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_1 X_2 + \varepsilon$

(C) $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_1 X_3 + \varepsilon$

(D) $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_1 X_2 + \beta_5 X_1 X_3 + \varepsilon$

(E) $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_1 X_2 + \beta_5 X_1 X_3 + \beta_6 X_2 X_3 + \varepsilon$

Solution. To effect interaction between the numeric predictor X_1 and the three-level categorical predictor, we should multiply X_1 by each of the two binary variables and insert the two products X_1X_2 and X_1X_3 as additional features. Together with the main effects terms X_1, X_2, and X_3, the model equation is

$$\begin{aligned}\mathbb{E}[Y] &= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \boxed{\beta_4 X_1 X_2 + \beta_5 X_1 X_3} \\ &= \begin{cases} \beta_0 + \beta_1 X_1, & \text{if } X_2 = X_3 = 0, \text{ (baseline)} \\ (\beta_0 + \beta_3) + (\beta_1 + \beta_5) X_1, & \text{if } X_2 = 0, X_3 = 1, \\ (\beta_0 + \beta_2) + (\beta_1 + \beta_4) X_1, & \text{if } X_2 = 1, X_3 = 0. \end{cases}\end{aligned}$$

Observe that the expected effect of X_1 on Y differs for each combination of the two binary variables X_2 and X_3 (unless β_4 or β_5 is zero), showing that an interaction effect between X_1 and the three-level categorical predictor exists. **(Answer: (D))** □

Remark. (i) While R will take care of the interaction variables automatically (see Subsection 3.3.2 about how interaction effects in a linear model can be handled in R), it is important to know the form of the resulting model equation—we have to know exactly what model we are dealing with!

(ii) Under the model in (D), the fact that there is no interaction effect between X_1 and the three-level categorical predictor is equivalent to $H_0 : \beta_4 = \beta_5 = 0$.

(iii) Because $X_2X_3 = 0$ (the two binary variables cannot be both 1), the model in (E) is in fact identical to that in (D), although the former looks bulkier.

Example 3.2.2. (Interpreting a linear model with interaction) Consider the fitted linear model

$$\hat{Y} = 40 + 20X_1 + 0.07X_2 + 35X_3 + 0.01X_1X_2 - 10X_1X_3,$$

where

$$\begin{aligned} \hat{Y} &= \text{predicted starting salary after graduation,} \\ X_1 &= \text{GPA,} \\ X_2 &= \text{IQ,} \\ X_3 &= \text{Gender (1 for Female and 0 for Male).} \end{aligned}$$

Determine which of the following statements is/are true.

I. The predicted starting salary for a female with a GPA of 4.0 and an IQ of 110 is 127.1.

II. For a fixed value of GPA and IQ, females earn more on average than males.

III. For a fixed value of GPA and IQ, males earn more on average than females provided that the GPA is high enough.

(A) None

(B) I and II only

(C) I and III only

(D) II and III only

(E) The correct answer is not given by (A), (B), (C), or (D).

Solution. I. True. The predicted starting salary when $X_1 = 4.0$, $X_2 = 110$, and $X_3 = 1$ is

$$\hat{Y} = 40 + 20(4) + 0.07(110) + 35(1) + 0.01(4)(110) - 10(4)(1) = 127.1.$$

II&III. II is false but III is true. Note that the fitted regression function is

$$\hat{Y} = \begin{cases} 40 + 20X_1 + 0.07X_2 + 0.01X_1X_2 + 35 - 10X_1, & \text{for a female,} \\ 40 + 20X_1 + 0.07X_2 + 0.01X_1X_2, & \text{for a male.} \end{cases}$$

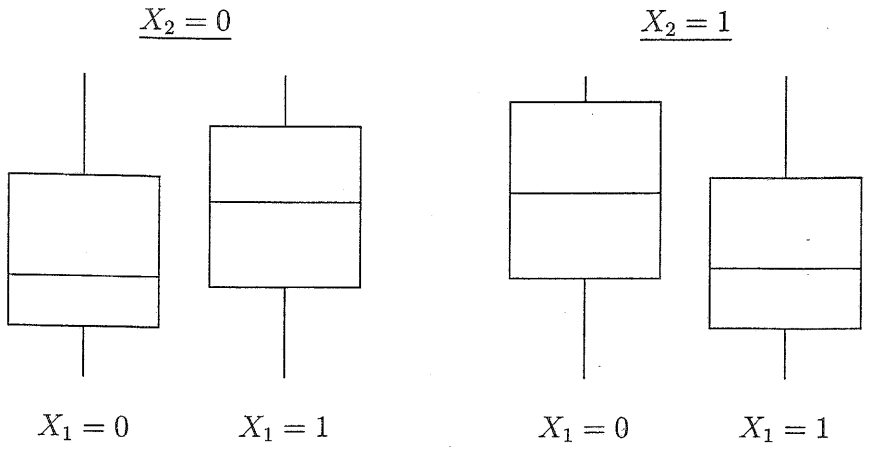
For a fixed value of GPA (X_1) and IQ (X_2), the difference between the predicted starting salary for a female and that for a male is

$$35 - 10X_1 \begin{cases} > 0, & \text{when } X_1 < 3.5, \\ \leq 0, & \text{when } X_1 \geq 3.5. \end{cases}$$

Thus provided that $X_1 \geq 3.5$, a male earns more on average. **(Answer: (C))**

□

Remark. Although the estimated coefficient of Gender is positive, we cannot ignore interaction effects and say females earn more on average for every combination of GPA and IQ values. That the effect of gender on starting salary differs for different values of GPA is a clear manifestation of the interaction between gender and GPA.



(a) The expected effect of X_1 on Y when $X_2 = 0$.

(b) The expected effect of X_1 on Y when $X_2 = 1$.

Figure 3.2.2: Graphical illustration of a linear model with interactions between categorical predictors.

- *Interactions between two categorical predictors:* Algebraically, the interaction between two categorical predictors can be dealt with in the same way as that between a continuous predictor and a categorical predictor. The interaction term is still a product of appropriate features. If the two categorical predictors are binary and represented by the binary variables X_1 and X_2, then the model equation takes the same form:

$$\mathbb{E}[Y] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 \boxed{X_1 X_2} = \begin{cases} \beta_0, & \text{if } X_1 = X_2 = 0, \\ \beta_0 + \beta_1, & \text{if } X_1 = 1, X_2 = 0, \\ \beta_0 + \beta_2, & \text{if } X_1 = 0, X_2 = 1, \\ \beta_0 + \beta_1 + \beta_2 + \beta_3, & \text{if } X_1 = X_2 = 1. \end{cases}$$

  Observe that the change in $\mathbb{E}[Y]$ when switching from $X_1 = 0$ to $X_1 = 1$ equals β_1 when $X_2 = 0$ and equals $\beta_1 + \beta_3$ when $X_2 = 1$ (equivalently, the change in $\mathbb{E}[Y]$ when switching from $X_2 = 0$ to $X_2 = 1$ equals β_2 when $X_1 = 0$ and equals $\beta_2 + \beta_3$ when $X_1 = 1$). That the two changes differ is an illustration of the interaction between the two predictors.

  Graphically, instead of using scatterplots, we may use box plots split by the levels of one categorical predictor faceted by the levels of another categorical predictor to assess the extent of interactions between the two predictors. An example is given in Figure 3.2.2, where the relationship between $\mathbb{E}[Y]$ and X_1 varies with the value of X_2. When $X_2 = 0$ (resp. $X_2 = 1$), switching from $X_1 = 0$ to $X_1 = 1$ increases (resp. decreases) the value of $\mathbb{E}[Y]$.

3.2.4 Feature Selection

Now that we have identified a list of potentially useful predictors, generated potentially desirable new features, and decided on the form of the model, we proceed to select which of these features are genuinely predictive of the target variable. Stated equivalently, we are going to remove features with weak predictive power to prevent overfitting and, by extension, improve predictive performance.

For linear models, feature removal is equivalent to setting some of the regression coefficients to zero, so that the corresponding features have no effect on the target variable. There are many different ways to look for these features of little predictive power. One may use traditional variable selection techniques such as the t-test introduced in Subsection 3.2.1 to assess the statistical significance of each feature in the presence of other features and remove features with a large p-value, as you did in a classical applied statistics course. In Exam PA, we will use modern automated feature selection techniques such as *stepwise selection algorithms*,[xii] which come in different versions:

- *Backward selection:* With backward selection, we start with the model with all features and work "backward." In each step of the algorithm, we drop the feature that causes, in its absence, the greatest improvement in the model according to a certain criterion (see Subsection 3.2.2). The process is terminated until no features can be dropped to improve the model.
- *Forward selection:* Forward selection is the opposite of backward selection. This time we start with the simplest model, namely the model with only the intercept but no features, go "forward," and augment the model by progressively adding the feature that results in the greatest improvement in the model, until no features can be added to improve the model.

These algorithms are implemented automatically in R, so we shall not be concerned with their mathematical details.

One thing to note is that although stepwise selection algorithms are computationally efficient to execute, they look at only a restricted list of candidate models and as a compromise there is no guarantee that they will yield the best subset of features among all possible combinations.

3.2.5 Regularization

Regularization is an alternative to forward or backward selection for feature selection, or equivalently, for reducing model complexity. Instead of going through a list of candidate models and identifying non-predictive features whose regression coefficients should be set to zero, regularization tackles feature selection by shrinking the magnitude of the estimated coefficients of features with limited predictive power towards zero, thereby reducing their expected effect on the target variable. In some cases, the effect of regularization is so strong that the coefficient estimates of non-predictive features are forced to exactly zero.

How does regularization work? Recall that the ordinary least squares fitting procedure lies in minimizing the (training) residual sum of squares

$$\text{RSS} = \sum_{i=1}^{n} [Y_i - (\beta_0 + \beta_1 X_{i1} + \cdots + \beta_p X_{ip})]^2$$

to produce the estimates of the regression coefficients, the $\hat{\beta}_j$'s. Regularization generates coefficient estimates by minimizing a slightly different objective function. Using the RSS as the starting point, regularization incorporates a penalty term that reflects the complexity of the model and minimizes

$$\sum_{i=1}^{n} [Y_i - (\beta_0 + \beta_1 X_{i1} + \cdots + \beta_p X_{ip})]^2 + \lambda f_R(\boldsymbol{\beta}),$$

where

[xii] These algorithms as described in the PA e-learning modules are not exactly the same as those described in Subsection 6.1.2 of ISLR.

- $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)$ is the vector of regression coefficients to be estimated.
- $\lambda \geq 0$ is the *regularization parameter* that controls the extent of regularization and quantifies our preference for simple models.
- $f_R(\boldsymbol{\beta})$ is the *regularization penalty* that captures the size of the regression coefficients.

There are two common choices of $f_R(\boldsymbol{\beta})$ (which you probably have seen in Exam SRM):

- When $f_R(\boldsymbol{\beta}) = \sum_{j=1}^{p} \beta_j^2$, the sum of the squares of the slope coefficients, we are performing *ridge regression.*
- When $f_R(\boldsymbol{\beta}) = \sum_{j=1}^{p} |\beta_j|$, the sum of the absolute values of the slope coefficients, we are performing *lasso.*

A more general regularization method that captures both ridge regression and lasso as special cases is *elastic net regression*, which corresponds to the regularization penalty given by

$$f_R(\boldsymbol{\beta}) = (1-\alpha) \underbrace{\sum_{j=1}^{p} \beta_j^2}_{\text{ridge regression}} + \alpha \underbrace{\sum_{j=1}^{p} |\beta_j|}_{\text{lasso}}, \tag{3.2.5}$$

where α is the *mixing coefficient* determining the relative weight of the ridge and lasso regularization penalties. If $\alpha = 0$, we are effectively performing ridge regression; if $\alpha = 1$, we retrieve the lasso.

Regardless of the specification of the regularization penalty, the essence of regularization is to balance the two competing forces, model fit as captured by the RSS and model complexity measured by the regularization penalty, and produce a set of coefficient estimates that not only fit the training data sufficiently well, but also produce a reasonably simple model. By reducing the complexity of the model and thus the variance of the predictions at the expense of a small increase in bias, we hope to ultimately improve the prediction accuracy of the model.

Note that the intercept is not part of the regularization penalty. It measures the expected value of the target variable when all other features are zero. Indeed, β_0 is not associated with any features and it does not hurt even if β_0 is large in magnitude.

Effects of λ. For simplicity, we will focus on ridge regression and lasso, which are the most commonly used regularization methods (the June 2019 PA exam tests ridge regression and lasso, but not elastic net), and investigate the effects of the regularization parameter λ on different aspects of the fitted linear model.

Note that unlike the ordinary least squares method, regularization produces a family of coefficient estimates $\{\hat{\boldsymbol{\beta}}_\lambda = (\hat{\beta}_{0,\lambda}, \hat{\beta}_{1,\lambda}, \ldots, \hat{\beta}_{p,\lambda}) : \lambda \geq 0\}$ indexed by the regularization parameter λ, with each value of λ giving rise to a particular set of estimates easily determined in R. The trade-off between model fit and model complexity is quantified by λ, which plays its role as follows:

- When $\lambda = 0$, the regularization penalty vanishes and the coefficient estimates are identical to the ordinary least squares estimates.
- As λ increases, the effect of regularization becomes more severe. The coefficient estimates become closer and closer to zero and the flexibility of the model drops, resulting in a decreased variance but an increased bias of the coefficient estimates. In most practical applications, the initial increase in λ causes a substantial reduction in the variance at the cost of only a slight rise in the bias. By trading off a minuscule increase in bias for a large drop in variance, we improve upon the prediction accuracy of the model.

- In the limiting case as $\lambda \to \infty$, the regularization penalty dominates and the estimates of the slope coefficients have no choice but to be all zero, i.e., $\hat{\beta}_{j,\lambda} \to 0$ for all $j = 1, \ldots, p$, and the linear model becomes the intercept-only model (i.e., the model with no features).

Example 3.2.3. (SOA Exam SRM Sample Question 8 (Reworded): Why use alternative fitting procedure?) Determine which of the following statements describe the advantages of using an alternative fitting procedure, such as subset selection and shrinkage, instead of least squares.

I. Doing so will result in a simpler model.

II. Doing so will likely improve prediction accuracy.

III. The results are easier to interpret.

(A) I only

(B) II only

(C) III only

(D) I, II, and III

(E) The correct answer is not given by (A), (B), (C), or (D)

Solution. Alternative fitting procedures tend to remove irrelevant variables from the full set of predictors, thus resulting in a simpler (I) and more interpretable (III) model. Prediction accuracy will likely be improved due to a substantial reduction in variance (II) (so long as the reduction outweighs the increase in squared bias). **(Answer: (D))** □

Ridge regression vs. lasso. While ridge regression and lasso are both regularization methods that serve to reduce model complexity and prevent overfitting, they differ in one important respect:

> The lasso has the effect of forcing the coefficient estimates to be *exactly zero*, effectively removing the corresponding features, when the regularization parameter λ is sufficiently large. In contrast, for ridge regression the coefficient estimates are reduced but none are reduced to exactly zero (and so all features are retained) even by making λ arbitrarily large.

As a result, the lasso tends to produce simpler and hence more interpretable models carrying fewer features. These models are called *sparse* models, meaning that they only involve a small subset of the potentially useful features. If you are interested in identifying only the *key* factors affecting the target variable, as is the case of the June 2019 PA exam, then the lasso is preferred due to its feature selection property, unless the fitted model is much inferior to that of ridge regression with respect to prediction accuracy.

Hyperparameter tuning. In the context of regularization, λ and α are called *hyperparameters*, which are pre-specified inputs that go into the model fitting process and are not determined as

part of the optimization procedure. Hyperparameters can be tuned by cross-validation discussed in Subsection 3.1.2. We construct a grid of values of (λ, α) (we only have to consider λ if we are performing ridge regression or lasso), compute the cross-validation error for each pair of values of (λ, α), and choose the pair that gives the lowest cross-validation error. Given the optimal hyperparameters, the penalized regression model is fitted to all of the training data and its prediction accuracy is evaluated on the test data. All of these steps can be readily implemented in R, so there is no need to worry about their computational aspects.

Pros and cons of regularization techniques for feature selection. How do regularization techniques compare to the stepwise feature selection techniques in Subsection 3.2.4? As you expect, none of the feature selection methods are universally superior and there are relative merits.

Merits.

1. The `glmnet()` function that implements penalized regression in R requires the binarization of categorical predictors in advance. As we will see in Subsection 3.4.2, this allows us to assess the significance of individual factor levels, not just the significance of the entire categorical predictor.
2. Regularization is computationally more efficient than stepwise selection algorithms. As stated on page 219 of *Introduction to Statistical Learning* (ISLR), the computations required to fit a penalized regression model "simultaneously for all values of λ are almost identical to those for fitting a model using least squares."

Demerits.

1. Regularization techniques may not produce the most interpretable model, especially for ridge regression, for which all features are retained.
2. During the model fitting process, all numeric features are standardized to ensure that they are on the same scale. This makes the interpretation of the coefficient estimates less intuitive than under linear models fitted by ordinary least squares.
3. The `glmnet()` function is restricted in terms of model forms. As suggested in the June 2019 PA exam, the function can accommodate some, but not all of the distributions for GLMs. The gamma model, for instance, is not covered.

3.3 Case Study 1: Fitting a Linear Model in R

In this section we will illustrate how a linear model can be fitted and used to make predictions in R by means of the `Advertising` dataset that is associated with ISLR. Although this dataset is only a simulated one (i.e., the observations are generated by computer algorithms, but are not from real data) and you may have already come across it when studying for Exam SRM, we have decided to make use of (or revisit) this dataset due to the following reasons:

- Its marketing context demonstrates how linear models can be applied to aid decision-making in practical situations.
- It allows us to illustrate many of the modeling concepts such as polynomial regression and interaction effects.