Chapter 3

Linear Models

EXAM PA LEARNING OBJECTIVES

6. Topic: Generalized Linear Models

(Note: Linear models, the topic of this chapter, are an important special case of generalized linear models.)

Learning Objectives

The Candidate will be able to describe and select a Generalized Linear Model (GLM) for a given data set and regression or classification problem.

a) Implement ordinary least squares regression in R and understand model assumptions.

b) Understand the specifications of the GLM and the model assumptions.

c) Create new features appropriate for GLMs.

d) Interpret model coefficients, interaction terms, offsets, and weights.
(Note: Offsets and weights will be introduced in Chapter 4 in the context of GLMs.)

e) Select and validate a GLM appropriately.

f) Explain the concepts of bias, variance, model complexity, and the bias-variance trade-off.

g) Select appropriate hyperparameters for regularized regression.

Chapter overview: Armed with the basics of R programming and data visualization techniques covered in Chapters 1 and 2, in Part II of this study manual we are ready to set foot in the predictive analytics arena and study predictive models that play a central role in Exam PA. This chapter is concerned with *linear regression models*, or simply *linear models*, which are widely considered one of the simplest but important predictive models and are the natural place to begin our study of predictive analytics. Despite their simplicity, linear models are perfect illustrations of a number of fundamental predictive analytic concepts, such as bias-variance trade-off, under- and overfitting, feature generation, and feature selection. Although linear models *per se* are often not the best tool for analyzing insurance data, they are the basic building block of more sophisticated predictive models that we will study in later chapters. A thorough understanding of linear models is therefore

crucial to a firm grasp of much of the material that follows.

To set the scene for the rest of this study manual, this foundational chapter begins with a broad overview of the fundamental concepts in predictive analytics. In Section 3.1, you will be introduced to a number of important considerations that guide the construction of an effective predictive model. These considerations are illustrated for linear models in Section 3.2, which lays the theoretical groundwork of linear models in the context of predictive analytics. Sections 3.3 and 3.4 demonstrate the practical implementation of linear models via two case studies. We will first look at a small-scale case study to familiarize ourselves with how to fit a linear model in R and make predictions. Section 3.4 presents a more involved case study that illustrates feature generation, feature selection, model diagnostics, and regularized regression, and is structured around a series of well-defined tasks that resemble the real exam. At the completion of this chapter, you should have a good understanding of the most important issues in predictive analytics and be comfortable with constructing and evaluating linear models in R.

3.1 A Primer on Predictive Analytics

This section streamlines the material scattered in different parts of the PA e-learning modules to provide a coherent introduction to predictive analytics.

3.1.1 Basic Terminology

Predictive analytics in a nutshell. Loosely speaking, *predictive analytics* refers to a vast set of statistical tools for "predicting" a target variable based on a set of closely related variables. Using predictive analytics, we can provide a data-driven response to many questions of practical interest. Here are some examples:

- Do age, gender, education level, and years of experience play an important role in determining salary? What is the form of the relationship between wage and such factors? Unraveling such a relationship can help employers set equitable wages for their employees and attract new talents.
- *(Ratemaking)* Predictive analytics plays a pivotal role in insurance *ratemaking*, which is the process of setting future year's premiums based on prior year loss experience and risk characteristics of policy holders.
- *(December 2018 and June 2019 PA exams)* What are the key factors affecting injury rates in a certain location, such as a coal mine and a road intersection? The ability to identify such factors will inform regulatory bodies when it comes to designing safer coal mines and road intersections in the future.

Classification of variables. Generally speaking, there are two ways to classify variables in a predictive analytic context: By their role in the study (intended use) or by their nature (characteristics).

- *By role:* The variable which we are interested in predicting is called the *target variable* (or *response variable*, *dependent variable*). The variables that are used to predict the target variable go by different names, such as *predictors*, *explanatory variables*, or sometimes simply *variables* if no confusion arises. We will mostly use the term "predictors" and "features" (to

be defined in Subsection 3.1.4) in the remainder of this manual. In an actuarial context, predictors are also known as risk factors or risk drivers.

Throughout your study of predictive analytics, it is useful to think of the relationship between the target variable Y and the set of predictors $\mathbf{X}$ (as a vector) as

$$\underset{\text{target}}{Y} = \underset{\text{systematic part / signal}}{f(\mathbf{X})} + \underset{\text{idiosyncratic part / noise}}{\varepsilon},$$

where:

- ▷ The (unknown but non-random) real function f is the main target of interest carrying the systematic information that the predictors offer about the target variable.
- ▷ ε is the random error term carrying information that cannot be captured by the systematic component of the model.

For convenience, we will also refer to f and ε respectively as the *signal function* and the *noise*, which are engineering terms. The goal of predictive analytics is often to filter out the noise and learn as much about the signal as possible based on the data you have.

- *By nature:* Variables can also be classified as *numeric* variables or *categorical* variables. Such a classification has important implications for developing an effective predictive model that aligns with the character of the target variable and predictors to produce realistic output.
  - ▷ *Numeric variables:* Numeric variables take the form of numbers with an associated range. They can be further classified as *continuous* variables, which, at least in theory, can assume any value in a continuum, or *discrete*[i] variables, which are restricted to only certain values (e.g., non-negative integers) in that range. An example of a continuous variable is the amount of insurance loss incurred by a policyholder in a certain period and an example of a discrete variable is the number of claims submitted. The distinction between continuous and discrete variables is unimportant when they serve as predictors, but needs to be properly taken into account with a target variable. Some predictive models (e.g., linear models) work well for continuous target variables while some (e.g., GLMs and decision trees) apply to both continuous and discrete target variables.
  - ▷ *Categorical variables:* Categorical variables take predefined values, called *levels*, out of a countable collection of categories. When a categorical variable takes only two possible levels, it is called a *binary* variable and the two levels indicate whether an event has taken place or whether a characteristic is present. In R, categorical variables are treated as factor variables (recall that factors are discussed on page 19). Examples of properties that can be described by categorical variables include gender (typically male or female), smoking status (smoking or non-smoking), marital status (single, married, divorced, widowed), and risk group (standard or substandard).

    Note that associating the levels of a categorical variable with numbers does not make the variable numeric. Consider, for instance, the variable X which represents smoking status and is defined as

$$X = \begin{cases} 0, & \text{for a non-smoker,} \\ 1, & \text{for a smoker,} \\ 2, & \text{for an individual with unknown smoking status.} \end{cases}$$

[i] Ordinal categorical variables can be treated technically as discrete variables.

Even though the three levels "non-smoker," "smoker," and "unknown" have been coded as "0," "1," and "2," respectively, the three numbers are merely labels without an implicit order and cannot be compared in an algebraic fashion. For example, "0" is not less than "1" in this case. It is crucially important that such a "numeric" variable, when serving as a predictor in a model, is treated as a categorical variable, or else you may get results which do not make sense.

Supervised vs. unsupervised problems. Predictive analytic problems can also be classified into *supervised* and *unsupervised learning* problems.

- *Supervised learning methods:* They refer to those for which there is a target variable "supervising" or guiding our analysis, and our goal is to understand the relationship between the target variable and the predictors and/or make accurate predictions for the target based on the predictors.

  There are only two supervised learning methods covered in Exam PA. They are GLMs and decision trees, which are covered in Chapters 3 to 5 of this manual.

EXAM NOTE

Since the exam has GLMs and decision trees as the only supervised learning methods and the PA exam must be about making predictions, at least one of these two methods must be heavily tested in each exam project. In the Hospital Readmissions sample project and the June 2019 exam, candidates were asked to construct different kinds of GLMs and, towards the end of the project, consider the pros and cons of using decision trees as an alternative.

(Challenge: Can you "predict" which supervised learning method would be tested in the June 2020 exam? ☺)

- *Unsupervised learning methods:* For unsupervised learning methods, a target variable is absent, and we are interested in extracting relationships and structures between different variables in the data. Again, only two unsupervised learning methods are in the PA exam syllabus, namely, principal components analysis and cluster analysis, which are the subject of Chapter 6 of this manual.

  Note that supervised and unsupervised learning methods are sometimes used in conjunction with one another. As we will see in Chapter 6, unsupervised learning methods can be used for the purpose of data exploration, which enables us to predict the target variable more accurately.

Regression vs. classification problems. In predictive analytics, it is customary to refer to supervised learning problems with a numeric target variable as *regression* problems (an exception is logistic regression, for which the target variable is binary). In contrast, when the target variable is categorical in nature, we are dealing with *classification* problems. A predictive model for predicting a categorical target variable is called a *classifier*.

Both regression and classification problems are of importance in Exam PA. Among the four released PA projects, two center on regression problems and the other two test classification problems. Both kinds of predictive analytic problems have unique features and will be covered in detail in Part II of this manual.

3.1.2 Model Validation

In predictive analytics, where the predominant concern is prediction accuracy, the fundamental question is:

> How do we go about designing a predictive model that performs well when it is applied to new, *previously unseen* data? In other words, how do we make sure that the model excels in predicting *future* target values on the basis of *future* predictor values?

Note that here the focus is on the prediction accuracy with *future* data, not with past data. If, for instance, you are developing a classifier that predicts whether a patient has cancer on the basis of his/her biomedical measurements, our interest is not in whether or not the classifier predicts accurately cancer risk for patients that are used to develop the classifier, because we already know which of those patients have cancer. Instead, we are interested in making accurate classifications for future patients who have developed cancer-like symptoms. To ensure that our model does a good job of describing the past data at hand and, more importantly, predicting new, *future* observations, an unconventional way of leveraging your data is needed.

Training/validation/test split. To construct a model that thrives on future data, it is common practice in predictive analytics to partition our data into two or three parts, each of which plays a different role in the model development process. This is in stark contrast to classical statistics where you will use all of your observations to fit your model.

- *Training set:* Typically the largest part of your data, the training set is where you "train" or develop your predictive model to estimate the signal function f and, if needed, the model parameters. If you have multiple candidate models, the same training set should be used to fit each model to ensure that all of them are compared on a fair basis.

- *Validation set:* The validation set is where you assess the predictive performance of your predictive model(s) using a certain performance metric (see the paragraph entitled "Common performance metrics" below). Observations in the validation set did not participate in the model training process and will provide a more objective ground for evaluating prediction accuracy. If you have multiple candidate models, you can use the validation set to differentiate the quality of these models and choose the one that performs the best on validation data.

- *Test set:* The test set is where you evaluate the predictive performance of your chosen model and obtain an independent measure of its prediction accuracy when it is applied to data that it has not seen before.

As suggested in the PA e-learning modules, a common split of data among the three sets above is 70%/20%/10%, but anything from 60% to 80% for the training set, with the rest divided between the validation and test sets, is acceptable. The split can be done either randomly according to these pre-specified proportions or with special statistical techniques (e.g., stratified sampling to be discussed on page 171) ensuring that the distributions of the target variable in the three sets are comparable.

Note that the terms "validation set" and "test set" are sometimes used interchangeably, but with subtly different meanings. In all of the PA sample and exam projects, we are primarily concerned with finding the best predictive model and less with exactly how well it performs on independent test data (unless your predictive model is required to beat an existing model), so we will split the data into the training set and the validation set only, and the latter is simply referred to as the test set. There is no independent test data set aside for final evaluation.

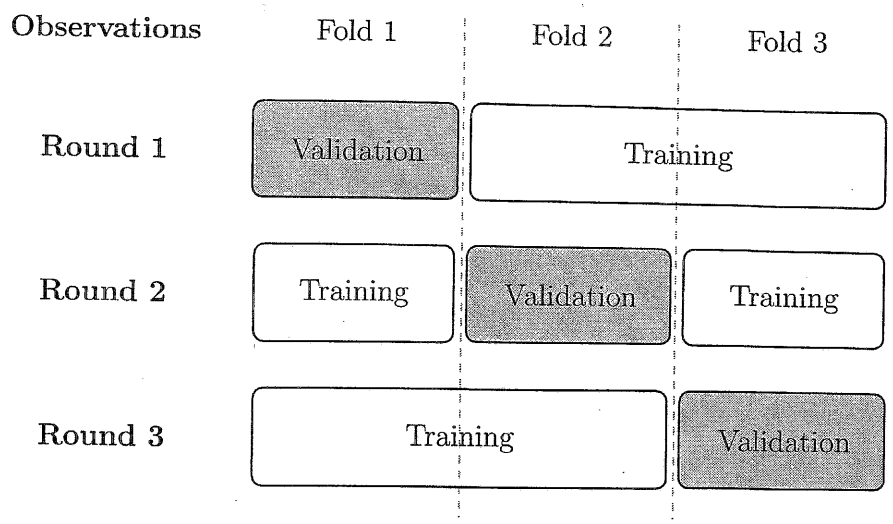


Figure 3.1.1: A schematic diagram of cross-validation with $k = 3$.

Cross-validation. When the size of your data is small, the training/validation/test split may lead to each set containing too few observations and undermine the credibility of the model development process. In this case, an alternative to the training/validation/test split is *cross-validation*, which performs the training-validation split repeatedly across (hence the qualifier "cross") the dataset. The procedure is as follows:

- Set aside the test data, if present, and do not worry about it until the end of the analysis.
- Split the remaining data into k folds of approximately equal size, for some given positive integer k. Common choices for k are 5 and 10.
- The predictive model is then fit k times. Each of the k folds serves in turn as the validation set and the remaining $k - 1$ folds serve as the training set. As a result, each fold is used exactly once for validation and predictions are made for every observation. These predictions can be combined and the overall predictive performance of the model can be assessed.

A schematic diagram of how cross-validation works when $k = 3$ is shown in Figure 3.1.1.

In fact, cross-validation is such a useful validation technique that it is automatically built into some model fitting algorithms such as regularization (see Subsection 3.4.4) and decision trees (see Chapter 5) for determining model parameters.

A TERMINOLOGY NOTE

Unless otherwise stated, in the rest of this manual we will follow the usage in the PA exam projects and use the term "test set" to mean "validation set" so long as no confusion arises.

Example 3.1.1. (CAS Exam MAS-I Spring 2019 Question 34: Mechanics of k-fold CV) A statistician has a dataset with $n = 50$ observations and $p = 22$ independent predictors. He is using 10-fold cross-validation to select from a variety of available models.

Calculate the number of times that the first observation will be included in the training set as part of this procedure.

(A) 0

(B) At least 1, but less than 5

(C) At least 5, but less than 10

(D) At least 10, but less than 20

(E) At least 20

Solution. When doing k-fold cross-validation, any particular observation will be included in the training dataset $k-1$ times; it is not included when the fold that contains the observation serves as the validation set. When $k = 10$, this is $\boxed{9}$. **(Answer: (C))** □

Remark. If "training set" in the question is changed to "validation set," then the answer will be 1.

Common performance metrics. After making the training/test split and fitting the predictive model to the training set, it is time for us to evaluate the predictive performance of the model on the test set with respect to an appropriate performance metric. The choice of the performance metric depends on the nature of the target variable (numeric or categorical), or equivalently, the nature of the prediction problem (regression or classification), as well as the type of predictive model used.

- *Regression problems:* When the target variable is numeric, the predictive performance of a model can be assessed by looking at the size of the discrepancy between the observed value of the target variable on the test set and the corresponding predicted value. This discrepancy is referred to as the prediction error for that observation in the test set. A common numeric predictive metric that aggregates all of the prediction errors on the test set and provides an overall measure of prediction accuracy is the test *root mean squared error* (RMSE), defined as

$$\boxed{\text{test RMSE} = \sqrt{\frac{1}{n_{\text{test}}} \sum_{i \in \text{test set}} (\underbrace{y_i - \hat{y}_i}_{\text{prediction error}})^2},}$$

  where n_{test} is the size of the test set and y_i and $\hat{y}_i$ are respectively the observed and predicted values of the target variable for the ith observation in the test set. The individual prediction errors are squared (so that positive and negative errors will not cancel out), summed, and averaged, followed by taking a square root, to yield the test RMSE. As you can expect, the *smaller* the RMSE, the more predictive the model.

  In your prior studies, you may have seen the concept of test mean squared error (MSE) given by

$$\text{test MSE} = \frac{1}{n_{\text{test}}} \sum_{i \in \text{test set}} (y_i - \hat{y}_i)^2 = (\text{test RMSE})^2.$$

  The advantage of using the RMSE over the MSE is that the former has the same unit as the target variable, making its value easier to interpret.

Example 3.1.2. (SOA Exam SRM Sample Question 11 (Adapted): Calculation of test RMSE) You have fitted a predictive model and applied it to the following test set with five observations:

Observation number (i)	y_i	$\hat{f}(x_i)$
1	2	4
2	5	3
3	6	9
4	8	3
5	4	6

Calculate the test root mean squared error.

Solution. The test RMSE is the square root of sum of the squared prediction errors:

$$\begin{aligned}\text{test RMSE} &= \sqrt{\frac{1}{5}\sum_{i=1}^{5}[y_i - \hat{f}(x_i)]^2} \\ &= \sqrt{\frac{1}{5}[(2-4)^2 + (5-3)^2 + (6-9)^2 + (8-3)^2 + (4-6)^2]} \\ &= \boxed{3.0332}.\end{aligned}$$

□

- *Classification problems:* For categorical target variables, the predicted values are simply labels which cannot be handled algebraically, so the difference $y_i - \hat{y}_i$ may not be well-defined or may not make sense even if it is defined (recall the smoking status variable X defined on page 113). We may instead look at the *classification error rate*, which is the proportion of observations in the test set that are incorrectly classified. The smaller the classification error rate, the more predictive the classifier.

  We will look at the performance metrics for categorical target variables more closely in Chapters 4 and 5.

Both the RMSE and classification error rate are general performance metrics that apply to general regression and classification problems, respectively. They can be computed on the training set as well as the test set, but as discussed above, our main interest is in these performance metrics on the test set. They are also distribution-free in the sense that their computations do not rely on a particular distribution assumed for the target variable. If the target variable is known a priori to follow a certain distribution (e.g., normal, Poisson), then additional likelihood-based performance metrics are available; see Subsection 3.2.2.

3.1.3 Bias-Variance Trade-off

Now that we know how to partition our data to evaluate the predictive performance of a model fairly, this subsection introduces the considerations that go into designing a good predictive model and puts them in a statistical framework.

Key idea: A complex model $\not\Rightarrow$ A predictive model. If you are just beginning to learn predictive analytics, you may be tempted to create a super sophisticated model with a huge number of predictors, some of which may be only marginally related to the target variable, in an attempt to enhance your prediction accuracy. To your surprise, such a bulky model does not necessarily make good predictions. When doing predictive analytics, an extremely important mentality to keep in mind is:

Prediction accuracy is not the same as goodness-of-fit.

In classical statistics, much effort is devoted to developing a model that fits well to the data on which the model is trained (the training data introduced above). The more complex the model, the lower the training error. In predictive analytics, our prime interest is in developing a model that makes accurate predictions on *future* data. Unfortunately, goodness-of-fit measures, which quantify how well a model describes *past* data, are not necessarily good measurements of how well the model will perform on *future* data. As we will see below, when it comes to making predictions for new observations, using an overly complex model may backfire. In fact, a central theme of predictive analytics is selecting the right level of complexity for constructing an effective prediction model. As the old adage goes, sometimes

"less (complexity) is more (prediction accuracy)!"

Decomposition of expected test prediction error. To better understand how model complexity impacts on prediction accuracy on the test set, it is instructive to break down the expected test prediction error into several components. Although such a decomposition is mainly of theoretical nature and you will rarely calculate the individual components of the formula in Exam PA, keeping the decomposition in mind will help you understand what it takes to produce accurate predictions. In particular, it reinforces the practically important point that more complex models are not necessarily superior models.

For simplicity, we will consider a numeric target variable so that the (R)MSE is an appropriate performance metric (categorical target variables admit a similar decomposition). For a given target variable Y_0 on the test set with $\mathbf{X}_0$ being its associated vector of predictor values, the decomposition reads

$$\underbrace{\mathbb{E}_{\text{Tr},Y_0}\left[\left(Y_0-\hat{f}(\mathbf{X}_0)\right)^2\right]}_{\text{expected test error @ }\mathbf{X}_0} = \overbrace{\underbrace{\text{Var}_{\text{Tr}}[\hat{f}(\mathbf{X}_0)]}_{\substack{\text{measures variability/}\\ \text{precision of }\hat{f}}} + \underbrace{[\text{Bias}_{\text{Tr}}(\hat{f}(\mathbf{X}_0))]^2}_{\substack{\text{measures accuracy}\\ \text{of }\hat{f}}}}^{\text{reducible error}} + \overbrace{\text{Var}(\varepsilon)}^{\text{irreducible error}}, \tag{3.1.1}$$

where $\hat{f}$ is the estimated signal function using the training set and the expectation on the left is taken over the new (random) target Y_0 as well as the training data (remember, the target variable values on the training set are only realizations from the distribution of the target variable). This identity breaks down the expected test prediction error into three salient components and sheds light on how model complexity affects the quality of predictions:

- *Bias:* The bias of $\hat{f}(\mathbf{X}_0)$ as a predictor of Y_0 is the difference between the expected value of $\hat{f}(\mathbf{X}_0)$ and the true value of the signal function f at $\mathbf{X}_0$, i.e., $\text{Bias}_{\text{Tr}}(\hat{f}(\mathbf{X}_0)) = \mathbb{E}_{\text{Tr}}[\hat{f}(\mathbf{X}_0)] - f(\mathbf{X}_0)$. In general, the more complex a model, the lower the bias (in magnitude) due to its higher ability to capture the underlying signal.

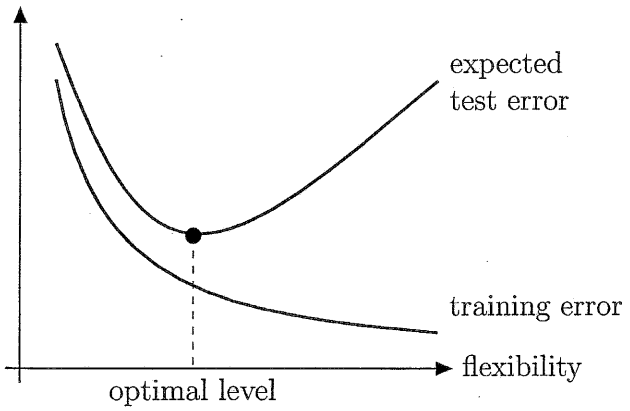


Figure 3.1.2: The qualitative behavior of training error and expected test error.

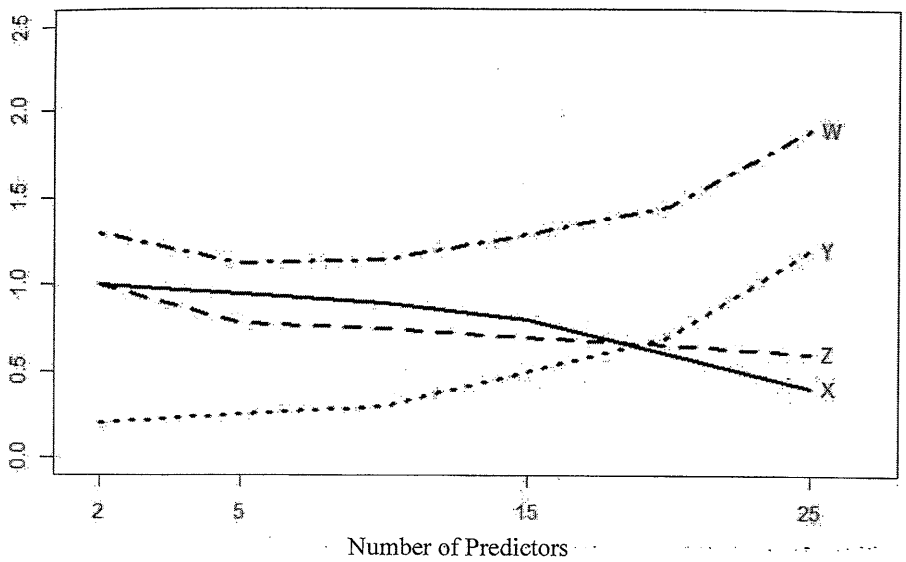
- *Variance:* The variance of $\hat{f}(\mathbf{X}_0)$ quantifies the amount by which $\hat{f}(\mathbf{X}_0)$ would change if we estimated it using a different training set. Ideally, $\hat{f}(\mathbf{X}_0)$ as a predictor of Y_0 should be relatively stable across different training sets. In general, the more complex a model, the higher the variance because a small perturbation in the training data can cause massive changes in $\hat{f}$. Consider, for example, the extreme case when $\hat{f}$ is so sophisticated that it can perfectly fit all of the training observations. Then any change in the training observations will lead to the same change in the predicted values.

- *Irreducible error:* This is the variance of the noise, which is independent of the choice of the predictive model, but inherent in the target variable. As this component cannot be reduced no matter how good your predictive model is, it is called the *irreducible error* in the expected prediction error. In contrast, the bias and variance of $\hat{f}(\mathbf{X}_0)$ constitute the reducible error as the performance of $\hat{f}(\mathbf{X}_0)$ can be improved by using a more appropriate predictive model.

As you can see, there is generally an intrinsic conflict between having a predictive model with low bias and having a predictive model with low variance. All else equal, a *more complex* model has a *lower bias* but a *higher variance* than a less flexible model (in addition to being less interpretable and more prone to computational issues). Such an inverse relationship between the model variance and the model bias is commonly referred to as the *bias-variance trade-off*, one of the most fundamental concepts in predictive analytics. The relative rate of change of these two competing quantities determines the overall behavior of the expected test prediction error. Typically, as the flexibility of a model increases, the bias initially tends to drop faster than the variance increases, so the expected test prediction error decreases. When the model becomes overwhelmingly complex, the drop in the bias is outweighed by the rise in the variance, so the expected test prediction error eventually rises, displaying a U-shape behavior (see Figure 3.1.2). In this case, we say that the unnecessarily complex model is *overfitting* the data; it is trying too hard to capture the noise in the training data and mistreats the noise as if it were a signal.

From a theoretical perspective, the construction of a good predictive model boils down to locating the right level of model complexity that corresponds, at least approximately, to the minimum point of the U-shape curve followed by the expected test error. Of course, the search of this minimum point when only a limited amount of data is available is no easy task and is one of the most important activities throughout this study manual!

Example 3.1.3. (Based on CAS Exam MAS-I Sample Question 2: Matching the four curves) You want to fit a linear regression model to a large dataset and need to determine the number of predictors to use. Below is a chart of four statistics from this model valued for various numbers of predictors:

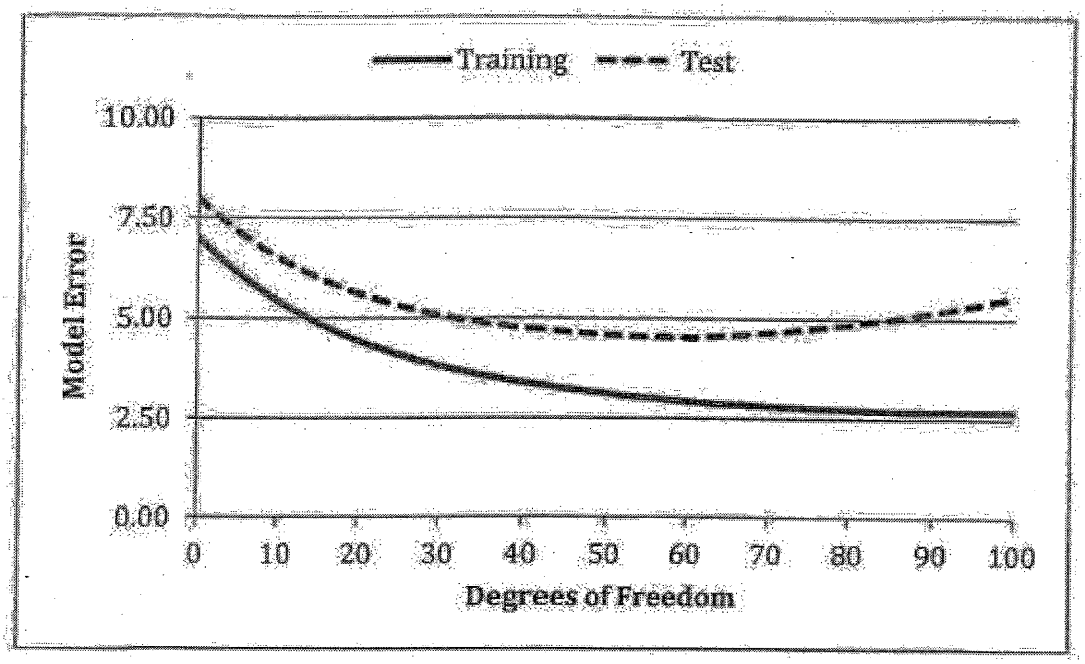


Determine which set of statistics below best describes each curve.

(A)	W is Test MSE	X is Variance	Y is Squared Bias	Z is Training MSE
(B)	W is Variance	X is Squared Bias	Y is Test MSE	Z is Training MSE
(C)	W is Training MSE	X is Test MSE	Y is Variance	Z is Squared Bias
(D)	W is Test MSE	X is Training MSE	Y is Variance	Z is Squared Bias
(E)	W is Variance	X is Training MSE	Y is Test MSE	Z is Squared Bias

Solution. The number of predictors is a natural measure of the flexibility carried by the linear model. As flexibility increases, the training MSE and squared bias both decrease, but the variance increases. The test MSE generally exhibits a U-shape. All of these are captured by and only by **Answer (D)**. □

Example 3.1.4. (CAS Exam 8 Fall 2017 Question 4 (a)-(b) (Reworded): Examining training and test errors graphically) An actuary has split a dataset into training and test sets for a model. The chart below shows the relationship between model performance and model complexity. Model performance is represented by model error (on the training or test sets) and model complexity is represented by degrees of freedom.



(a) Briefly explain why it is desirable to split the data into training and test sets.

(b) Briefly describe whether each of the following models has an optimal balance of complexity and performance.

(i) Model 1: 10 degrees of freedom

(ii) Model 2: 60 degrees of freedom

(iii) Model 3: 100 degrees of freedom

Solution. (a) Recall that the more flexible the model, the better the fit to the training data, but not necessarily the fit to independent test data. With the training/test split, we use the training set to fit the model but then test its predictive power on the test set (which is data "unseen" by the model) to avoid overfitting to the noise of the data and to ensure that the model is predictive.

(b) (i) This model does not have an optimal balance as the model error on both the training and test data is relatively high and the test error can be decreased by making the model more complex.

(ii) This model has an optimal balance as the test error is approximately at its lowest point, meaning that adding or removing parameters will raise the test error.

(iii) This model does not have an optimal balance. While adding more parameters has lowered the training error, the test error has increased, suggesting overfitting.

□

Visualizing bias-variance trade-off: A toy example. As a concrete pictorial illustration of the bias-variance trade-off, we consider fitting five linear models of different degrees of model complexity and investigate the predictive performance of these five models on independent test data by simulation. Specifically, we suppose that the true relationship between the target variable Y and the single predictor X is $Y = f(X) + \varepsilon$, where the signal function is the square function $f(X) = X^2$ and the noise ε is normally distributed with a zero mean and a standard deviation of 0.5. Without knowing the true f in advance, we fit five linear models with polynomial regression functions of varying degrees (0, 1, 2, 4, and 8) to the training data, of size 300, and make a prediction for the target variable at $X_0 = 0.75$. The higher the degree, the more flexible the linear model. Since $f(X) = X^2$, the true value of the signal function at X_0 is $0.75^2 = 0.5625$, which allows us to judge the predictive performance of the five linear models. This process is repeated 2,000 times to appreciate the distribution of the predictions for each model. In particular, we are interested in estimating the expected test MSE for each model from the results of the 2,000 rounds of simulation.

In CHUNK 1, we use a `for` loop to perform the 2,000 rounds of simulation and calculate the average test MSE for each linear model. Don't worry too much about the code as you are not expected to write any involved code in Exam PA and some of the functions such as `lm()` and `predict()` will be covered later in this chapter. For now, just focus on the size of the five average test MSEs.

```
# CHUNK 1
n <- 300  # size of each sample
n_sims <- 2000  # number of rounds of simulation
x0 <- 0.75  # predictor value of interest
df.test <- data.frame(x = x0)  # test set of one observation
pred <- matrix(NA, nrow = n_sims, ncol = 5)
mse <- matrix(NA, nrow = n_sims, ncol = 5)

set.seed(1)
for (i in 1:n_sims) {
  x <- runif(n)
  e <- rnorm(n, sd = 0.5)
  y <- x^2 + e

  # the training set for each round
  df.train <- data.frame(x, y)  # training set of n observations

  # fit the five linear models to training set
  model.0 <- lm(y ~ 1, data = df.train)  # intercept-only model
  model.1 <- lm(y ~ x, data = df.train)
  model.2 <- lm(y ~ poly(x, 2), data = df.train)
  model.4 <- lm(y ~ poly(x, 4), data = df.train)
  model.8 <- lm(y ~ poly(x, 8), data = df.train)

  y0 <- x0^2 + rnorm(1, sd = 0.5)  # random target of prediction

  # calculate the predicted value for each linear model
  pred[i, 1] <- predict(model.0, newdata = df.test)
```

```
  pred[i, 2] <- predict(model.1, newdata = df.test)
  pred[i, 3] <- predict(model.2, newdata = df.test)
  pred[i, 4] <- predict(model.4, newdata = df.test)
  pred[i, 5] <- predict(model.8, newdata = df.test)

  # calculate the MSE for each linear model
  mse[i, 1] <- (y0 - pred[i, 1])^2
  mse[i, 2] <- (y0 - pred[i, 2])^2
  mse[i, 3] <- (y0 - pred[i, 3])^2
  mse[i, 4] <- (y0 - pred[i, 4])^2
  mse[i, 5] <- (y0 - pred[i, 5])^2
}

for (i in 1:5) {
  print(mean(mse[, i]))
}

## [1] 0.3061781
## [1] 0.2546873
## [1] 0.254079
## [1] 0.2561206
## [1] 0.2586937
```

We can see that as the degree of the polynomial regression function increases from 0 (i.e., no predictors are used), 1, to 2, the average test MSE decreases. The models of degrees 0 and 1 appear to be *under*fitting the data and fail to capture the signal sufficiently. As we further raise the polynomial degree, the average test MSE rises and follows the U-shape behavior discussed above. The models of degrees 4 and 8 are *over*fitting the data and following the noise in the training data too closely. As we expect, the lowest average test MSE is achieved by the linear model of degree 2, which has the true level of model complexity. (Question: The average test MSEs are all greater than $0.5^2 = 0.25$. Do you know why? Hint: (3.1.1).)

We can gain a lot more insights into the behavior of the predictions produced by the five models by making separate box plots for these predictions, as in CHUNK 2 (see Figure 3.1.3). The box plots reveal that the models of degrees 0 and 1 perform poorly in terms of bias—their predictions are far away from the true signal value $f(X_0) = 0.5625$, which is indicated by a dashed horizontal line. The models of degrees 2, 4, and 8 all make accurate predictions in the sense that their predictions, on average, equal the true signal value. However, the higher the degree of the regression function, the greater the amount of fluctuation around the average value. It follows that the average test MSEs of the models of degrees 4 and 8 are higher than that of the model of degree 2. Overall, the findings in Figure 3.1.3 conform to the bias-variance trade-off and provide a good illustration of the interplay between model complexity and predictive performance.

3.1.4 Feature Generation and Selection

A practical implication of the bias-variance trade-off is that it is vitally important to strike a balance between having a simple model that produces stable (but possibly inaccurate) predictions and a flexible model that is capable of capturing complex signals but may make highly variable predictions.

```
# CHUNK 2

# Combine the predicted values and MSEs of the five models
# as a single vector for plotting purposes
pred <- c(pred[, 1], pred[, 2], pred[, 3], pred[, 4], pred[, 5])
mse <- c(mse[, 1], mse[, 2], mse[, 3], mse[, 4], mse[, 5])
model_deg <- as.factor(rep(c(0, 1, 2, 4, 8), each = n_sims))
df <- data.frame(model_deg, pred, mse)

library(ggplot2)
ggplot(df, aes(x = model_deg, y = pred)) +
  geom_boxplot(fill = "red") +
  geom_hline(yintercept = x0^2, linetype = "dashed") +
  labs(x = "Degree", y = "Predictions")
```

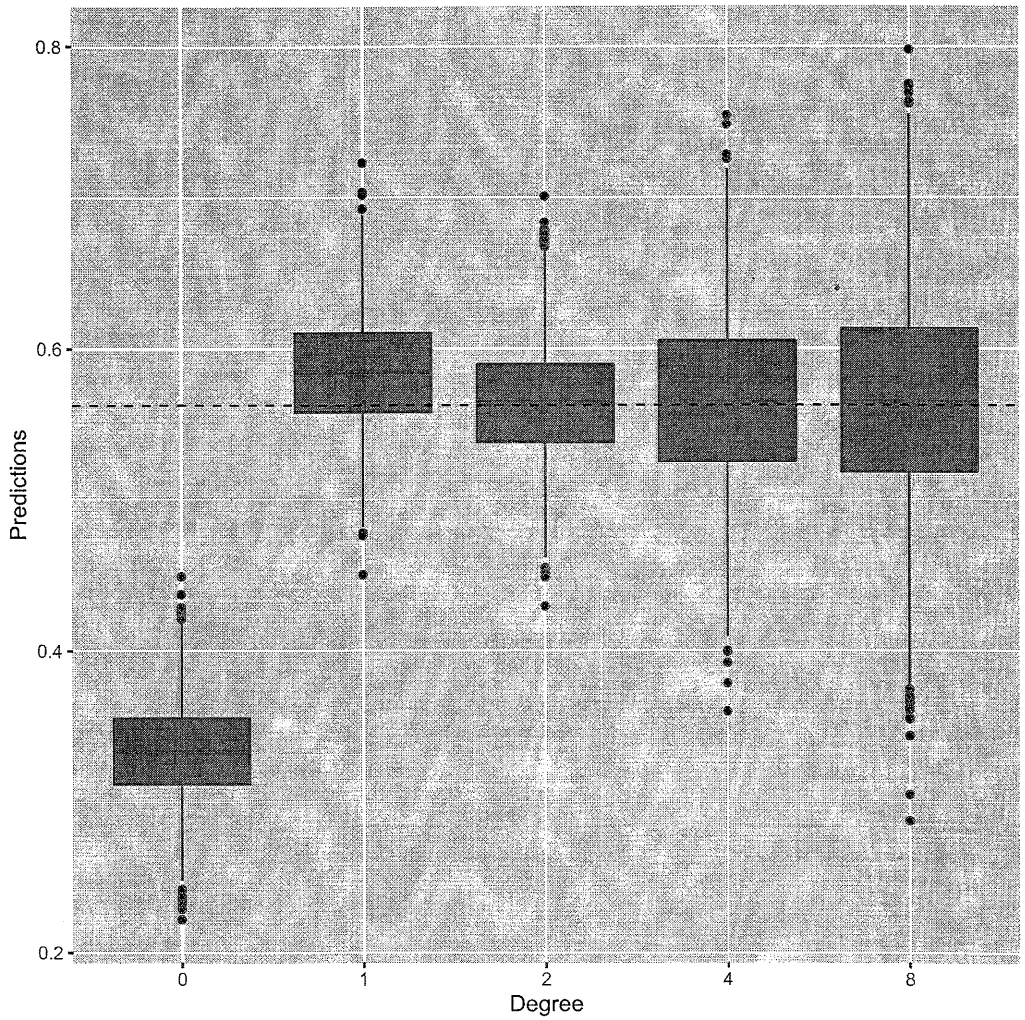


Figure 3.1.3: Box plots for the predictions produced by linear models with regression functions of five different degrees.

This in turn boils down to an effective control of model complexity, which can be achieved by feature generation and selection.

Variables vs. features. To many, the terms "variables" and "features" are synonyms meaning the predictors used in a predictive model.[ii] In a predictive analytic context (and in Exam PA, in particular), however, there is a subtle distinction between variables and features that we should keep in mind.

- *Variables:* In statistics literature, a *variable* is synonymous with a predictor in a model. In the predictive analytics arena, a variable more precisely means a raw measurement that is recorded and constitutes the original dataset prior to any data transformation.
- *Features:* In the machine learning community, *features* refer to derivations from the original variables and provide an alternative, more useful view of the information contained in the dataset. In the language of Exam MFE/IFM, features are "derivatives" of raw variables.

Feature generation. *Feature generation* is the process of developing new features based on existing variables in the data. These new derived variables can then serve as final inputs into our predictive model. Within the context of bias-variance trade-off, feature generation seeks to enhance the flexibility of the model and lower the bias of the predictions at the expense of an increase in variance.

There are many methods for generating new features, many of which are motivated from practical considerations. Here are some common examples: (Example 2 is illustrated in the Hospital Readmissions sample project)

Example	Variable(s)	Feature
1	Issue year of an insurance policy	Policy duration (= current year − issue year)
2	Age	An indicator variable that flags whether an individual is under 65 or above 65 (such a feature may be useful for explaining health- or retirement-related target variables)
3	Gender (e.g., male and female) and smoking status (smoker and non-smoker)	A single indicator of gender and smoking status (e.g., male smoker, female non-smoker)
4	(Generic variables)	Principal components, which are linear combinations of existing variables (see Chapter 6 for details)

Out of the two supervised learning techniques covered in the PA exam syllabus, feature generation plays a more prominent role in GLMs. When constructing GLMs, very often you will create functional transformations of numeric variables, dummy variables for representing categorical variables, and interaction terms for both numeric and categorical variables. This will be covered in detail in Subsection 3.2.3 and illustrated in Sections 3.3 and 3.4.

[ii]The bottom of page 15 of *Introduction to Statistical Learning* (ISLR) says that the terms "predictors," "independent variables," "features," and "variables" mean essentially the same thing. We will not follow this usage in Exam PA.

Feature selection. *Feature selection* (or feature removal), the opposite of feature generation, is the procedure of dropping features with limited predictive power and therefore reducing the dimension of the data. Within the context of bias-variance trade-off, it is an attempt to control model complexity and prevent overfitting.

Feature selection is an important predictive analytic concept that applies to both GLMs and decision trees. It is particularly relevant to categorical predictors, each category of which is technically a feature in the model. If a categorical predictor has multiple categories, then they can substantially inflate the dimension of the data and undermine prediction precision. Some commonly used strategies for reducing the dimension of a categorical predictor are:

- *Combining similar categories:* Remember that predictors are meant to differentiate the target variable. In the case of a categorical predictor, the ideal case is that the values (or levels) of the target variable are relatively homogeneous within each category but are markedly different across different categories. If the target variable behaves similarly (with respect to the mean, median, or other distributional measures) in two categories, then these two categories can be combined without losing much information while reducing the dimension of the data.

- *Combining sparse categories with others:* Categories with very few observations are also natural candidates to be combined with other categories. Due to their small sample size, it is difficult to estimate the effects of these categories on the target variable reliably. It is advisable to fold these sparse categories into other categories in which the target variable exhibits a similar behavior. If desired, we may label the combined categories as "other" to distinguish them from categories most relevant to the analysis.

These category combination strategies are tested in Task 2 of the June 2019 PA exam, which reads:

> 2. *(5 points)* Reduce the number of factor levels where appropriate
>
> Several of the variables have a small number of observations at some of the factor levels. Consider using knowledge of the factor levels as well as evidence from Task 1 (see page 83 of this manual) to combine some of them into factor levels with more observations.

The model solution says that "[t]he best candidates found *multiple cases* where factor levels could be combined, considered the *similarity of means/medians*, and provided adequate rationale for combining levels. It was *not sufficient* to only combine those levels with *extremely low counts*." As you can imagine, there are many possible ways to do the category combinations and some degree of subjectivity is involved. As the SOA says, "[t]he number of variables for which factors were combined was less important than the *quality* of the combinations made and the *supporting evidence*."

Example 3.1.5. (A simplified version of Task 2 of June 2019 PA exam) The following table shows the mean and median of a numeric target variable split by different levels of a categorical predictor:

Level	Mean	Median	Number of Observations
1	11.23	10.95	10,000
2	10.79	10.03	2,000
3	9.47	9.12	500
4	8.63	8.42	200
5	7.18	6.97	20

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.