Risk Analytics

Machine Learning and Optimization
for Data-Driven Decision Making

Fernando S. Oliveira

Draft version — May 14, 2026

Chapter 8
Supervised Learning for Risk Prediction

Chapter 8

Supervised Learning for Risk Prediction

8.1 Introduction: From Risk Segments to Risk Predictions

Chapter 7 introduced classification trees as transparent tools for risk segmentation. A tree helps the analyst explain why some borrowers, transactions, suppliers, assets, or projects appear riskier than others. It converts data into visible if–then rules and therefore supports communication, governance, and managerial understanding.

This chapter moves from segmentation to prediction. In many risk problems, the analyst does not only want to know which group a case belongs to. The analyst wants to estimate how likely an adverse event is, how cases should be ranked, and how predicted probabilities translate into expected loss. A lender may want to estimate the probability that an account will default. An insurer may want to estimate the probability that a claim will occur. A procurement team may want to estimate the probability that a supplier will fail. An operations manager may want to identify machines, projects, or shipments that deserve attention before losses materialize.

The practical question is therefore not only, "Is this case high risk?" It is also, "How high is the risk, how reliable is the estimate, and what economic exposure does it create?" This distinction matters because risk decisions usually involve scarce resources. Organizations cannot investigate every transaction, review every account, inspect every supplier, or intervene in every project. They need models that help them prioritize attention, allocate capacity, and estimate the loss consequences of different actions.

The chapter focuses on three supervised-learning model families. Logistic regression provides a transparent probability baseline and remains important in credit scoring and other regulated prediction settings because it links predictors to event probabilities through the log-odds transformation [4, 1, 9].

Random forests improve tree-based prediction by averaging many decorrelated trees, reducing the instability of a single tree [2, 12]. Gradient boosting builds trees sequentially to reduce prediction error and can capture nonlinearities, thresholds, and interactions that simpler models may miss [7, 3].

These models are not introduced as a general machine-learning survey. They are introduced because they illustrate three distinct ways of estimating risk: a transparent parametric benchmark, a stable nonlinear ensemble, and a high-performing sequential ensemble. The emphasis is applied throughout. A model is useful only if its output supports the risk decision being made. A class label may be sufficient for a simple routing decision, but risk analytics usually requires more: probability estimates, risk rankings, concentration diagnostics, expected-loss calculations, and managerial interpretation.

The central message is that supervised-learning outputs become risk measures only when they are connected to exposure, severity, and decision context. A probability estimate is not yet a loss estimate. A high score is not yet a decision. A model with strong ranking performance may still require calibration before its probabilities are used in expected-loss calculations. Conversely, a simpler and more transparent model may remain valuable as a benchmark even when a more flexible model performs better.

The chapter proceeds as follows. Section 8.2 defines supervised learning as a risk-prediction problem. Section 8.3 links probability prediction to exposure and expected loss. Section 8.4 introduces logistic regression as a transparent baseline probability model. Sections 8.5 and 8.6 present random forests and gradient boosting. Section 8.7 compares these models in a credit-card default case study and shows how probability estimates can be converted into expected-loss and concentration diagnostics. Section 8.8 summarizes the managerial lessons.

8.2 Supervised Learning as Risk Prediction

In supervised learning, the analyst observes historical examples of inputs and outcomes. Let

$$\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{n}$$

denote a dataset with n observations. The vector

$$X_i = (X_{i1}, \ldots, X_{ip})$$

contains the p observable characteristics of case i. Thus, X_i denotes the whole predictor vector for case i, while X_{ij} denotes the value of predictor

j for that case. The variable Y_i is the outcome to be predicted. In risk analytics, a case may be a borrower, account, transaction, policyholder, supplier, patient, machine, project, shipment, region, or time period. The features in X_i describe what is known before the outcome is realized. The outcome Y_i records what eventually happened.

A supervised-learning model uses the historical data to learn a prediction rule. Let f denote a candidate prediction function and let $\mathcal{F}$ denote the set of functions the analyst is willing to consider. Model fitting can be written generically as

$$f^* \in \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(X_i), Y_i),$$

where f^* is the selected prediction function and ℓ is the loss function used during estimation. This expression is useful because it separates three modelling choices: the data used for learning, the type of prediction functions allowed, and the errors emphasized during estimation. Logistic regression, random forests, and gradient boosting differ because they represent different families of prediction functions and learn from error in different ways.

For a binary adverse event,

$$Y_i = \begin{cases} 1, & \text{if the adverse event occurs,} \\ 0, & \text{otherwise.} \end{cases}$$

Examples include default, fraud, claim occurrence, churn, supplier failure, machine breakdown, hospital readmission, cyber breach, safety incident, or severe delay. The fitted model may produce a risk score that ranks cases from lower to higher estimated risk. More importantly for risk analytics, it may produce a predicted probability. Throughout this chapter,

$$p_i$$

denotes the model-predicted probability that the adverse event occurs for case i, based on the information in X_i. The notation is intentionally simple: p_i is the probability output used by the analyst, not the unknown true probability.

The probability estimate is usually more useful than a simple high-risk or low-risk label because it can later be combined with exposure, severity, and decision rules. A predicted default probability of 8% does not itself decide whether a borrower should be accepted, rejected, priced differently, or reviewed manually. Those choices depend on exposure, expected revenue, loss severity, capital cost, risk appetite, legal constraints, and operational

capacity. The modelling task is to estimate quantities that can later enter those decisions.

Supervised learning can also be used when the outcome is continuous rather than binary. Examples include claim amount, repair cost, recovery time, delivery delay, downtime, or project overrun. In such cases, the model estimates a numerical outcome directly. This chapter focuses mainly on binary adverse-event prediction because it provides a clear bridge from probability estimation to expected-loss analysis.

8.3 Probability Prediction, Exposure, and Expected Loss

The probability estimate is the bridge between supervised learning and quantitative risk analysis. If the adverse event would generate a loss c_i, the predicted expected loss for case i is

$$e_i = p_i c_i,$$

where e_i denotes the model-implied expected loss for case i. This expression is deliberately simple. It shows that the classifier is not the entire risk model. The classifier supplies the probability component; the analyst must still specify the loss, severity, or exposure component.

In credit risk, the same logic is usually written in terms of probability of default, loss given default, and exposure at default. In the notation used here, g_i denotes the loss rate if default occurs and a_i denotes the exposure of account i. The predicted expected loss for account i is then

$$e_i = p_i g_i a_i. \tag{8.1}$$

The probability component p_i comes from the prediction model. The severity component g_i and the exposure component a_i may come from separate models, accounting systems, expert assumptions, or scenario analysis.

This decomposition clarifies why probability ranking and economic-risk ranking are not always the same. A borrower with a lower default probability may have a higher expected loss if the exposure is much larger. A supplier with a moderate disruption probability may be critical if it provides a key component. A machine with only moderate failure probability may deserve priority if downtime is very costly. Probability is therefore one component of risk, not the whole risk measure.

For a portfolio of n cases, the average predicted loss is

$$\overline{e} = \frac{1}{n} \sum_{i=1}^{n} e_i.$$

In applied work, this average can be scaled by the number of cases to obtain the portfolio total when needed. More importantly, the case-level values e_i identify which borrowers, suppliers, transactions, assets, or projects contribute most to predicted loss.

The empirical case below does not estimate a full severity model because the credit-card dataset contains the default indicator but not realized loss-given-default. Instead, the analysis uses credit limit as an exposure proxy and applies alternative severity assumptions. This is sufficient for the chapter's purpose: to show how predicted probabilities become expected-loss inputs. When realized severity data are available, the same framework can be extended by modelling occurrence and severity separately.

Probability prediction also changes how model quality is interpreted. A model used only to rank cases may be useful even if its probability levels are imperfect. A model used for expected-loss estimation must produce probabilities with numerical meaning. If a model systematically overstates or understates event probability, expected losses will be biased. Thus, when the target is expected loss, the probability estimate must be treated as a quantitative input, not merely as a ranking score.

8.4 Logistic Regression as a Baseline Risk Model

Logistic regression is one of the most important baseline models for binary risk prediction. It was introduced as a statistical model for binary response data and remains widely used because it estimates event probabilities directly [4, 9]. In credit scoring and financial-risk assessment, this probability interpretation is often more useful than a class label [1].

For each case i, logistic regression uses the predictor vector

$$X_i = (X_{i1}, \ldots, X_{ip}).$$

The model first forms a linear risk index,

$$z_i = \beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}.$$

Here, β_0 is the intercept and β_j is the coefficient associated with predictor j. The index z_i is a weighted sum of the observed predictor values for case i. It can take any real value, so it cannot itself be interpreted as a probability.

Logistic regression converts this index into a probability using the logistic function,

$$\Lambda(z_i) = \frac{1}{1 + \exp(-z_i)}.$$

The model-predicted event probability is therefore

$$p_i = \Pr(Y_i = 1 \mid X_i) = \Lambda(z_i) = \frac{1}{1 + \exp\left[-\left(\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}\right)\right]}.$$

Equivalently, the log-odds are linear in the predictors:

$$\log\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}.$$

This representation is central to the model's practical value. A positive coefficient increases the log-odds of the adverse event, while a negative coefficient decreases them, holding the other predictors constant. The quantity $\exp(\beta_j)$ is the odds ratio associated with a one-unit increase in predictor j. The model therefore provides both a probability estimate and an interpretable direction of effect.

Logistic regression is usually estimated by maximum likelihood, a principle formalized in classical statistical theory by Fisher [6]. For observations $Y_i \in \{0, 1\}$, the likelihood contribution is

$$p_i^{Y_i}(1 - p_i)^{1 - Y_i},$$

and the log-likelihood is

$$\ell(\beta) = \sum_{i=1}^{n} \left[Y_i \log(p_i) + (1 - Y_i) \log(1 - p_i)\right],$$

where $\beta = (\beta_0, \beta_1, \ldots, \beta_p)$ collects the intercept and all slope coefficients. Maximizing this log-likelihood is equivalent to minimizing binary cross-entropy or log-loss. A logistic model is therefore not only a classifier; it is a probability model for a binary risk event.

A class prediction is obtained only after an additional decision step. With the conventional threshold 0.5, the model predicts the adverse class when

$p_i \geq 0.5$. The corresponding decision boundary is

$$\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij} = 0.$$

This shows both the strength and limitation of logistic regression. The probability curve is nonlinear in the risk index, but the decision boundary is linear in the predictors. If the true risk surface contains nonlinearities, thresholds, or interactions, a basic logistic regression will miss them unless the analyst includes transformed variables, splines, bins, or interaction terms.

This limitation is not necessarily a weakness. Logistic regression is often chosen because it is stable, transparent, and relatively easy to audit. In credit-risk applications, continuous variables may be binned, categorical indicators represented by dummy variables, and fitted log-odds translated into a points-based scorecard. In the present chapter, logistic regression supplies an interpretable probability baseline against which more flexible models can be compared.

The model still requires care. Categorical variables need appropriate coding. Continuous variables may require transformation, centering, or nonlinear terms. Sparse categories and rare-event labels can create complete or quasi-complete separation. Small-sample bias can arise in rare-event settings, motivating penalized likelihood or bias-reduction approaches such as Firth's correction [5]. Regularized logistic regression can also help when predictors are numerous or correlated: ridge penalties shrink coefficients, lasso penalties can perform variable selection, and elastic net combines both ideas [8, 10].

The role of logistic regression is therefore clear. It is not presented as the most flexible model. It is the disciplined probability benchmark. If ensembles improve substantially over logistic regression, the gain can be interpreted as the value of modelling nonlinearities, interactions, and threshold effects beyond a transparent log-odds baseline.

Table 8.1: Why logistic regression remains useful in risk prediction

Feature	Risk value	Caution
Probability output	Produces event probabilities directly	Requires representative data and calibration checks
Log-odds structure	Gives interpretable coefficients and odds ratios	Effects are linear in log-odds unless features are engineered
Transparency	Supports explanation, audit, and benchmarking	May miss nonlinear patterns and interactions
Regularization	Helps with many or correlated predictors	Penalties affect interpretation and probability levels
Rare-event baseline	Provides a disciplined benchmark for default, fraud, and failure models	May require bias correction or careful sampling

8.5 Random Forests

A single classification tree is easy to explain, but it can be unstable. A small change in the training data may change an early split, and that early change can alter the entire tree. Random forests address this weakness by replacing one tree with many trees and averaging their predictions [2].

The basic idea is simple. Instead of asking one tree to make the prediction, the analyst grows many different trees. Each tree is trained on a bootstrap sample, that is, a sample drawn from the training data with replacement. Because each bootstrap sample differs slightly, each tree sees a slightly different version of the data. In addition, when a tree searches for the best split at a node, it does not examine all predictors. It examines only a random subset of predictors. This second source of randomness prevents the same dominant variable from driving every tree.

For a binary adverse event, tree b produces a probability p_{ib} for case i. If the forest contains B trees, the random-forest probability for case i is

$$p_i = \frac{1}{B}\sum_{b=1}^{B} p_{ib}.$$

Thus, the forest prediction is an average of many tree predictions.

The strength of the method comes from variance reduction. A deep tree can capture nonlinearities, thresholds, and interactions, but it can also adapt too closely to random noise in the training data. Averaging many trees reduces this instability. The individual trees do not need to be perfect. They

need to be useful and sufficiently different from one another. If their errors are not perfectly correlated, the average prediction is more stable than the prediction of a single tree.

This logic is valuable in risk analytics. Credit-risk data often contain several related repayment-history variables. Fraud data may contain many transaction-frequency and transaction-amount indicators. Supplier-risk data may contain correlated quality, delivery, financial, and geopolitical indicators. A single tree may rely too heavily on one variable. A random forest can combine many alternative partitions of the data and produce a more robust ranking of risky cases.

Random forests also handle nonlinear relationships naturally. For example, default risk may increase sharply only after repayment delay exceeds a certain threshold. Fraud risk may rise only when unusual transaction size is combined with unusual location or timing. Supplier risk may depend on combinations of financial weakness, delivery delays, and country exposure. A random forest can represent such interactions without requiring the analyst to specify them manually.

The main tuning choices are the number of trees, the maximum depth of each tree, the minimum number of observations required in a leaf, and the number of predictors considered at each split. More trees usually improve stability, although with diminishing returns. Deeper trees increase flexibility but may overfit if leaves become too small. Larger leaves produce smoother probability estimates. The number of predictors considered at each split controls how different the trees are from one another.

Random forests also provide useful diagnostics. Variable-importance measures can indicate which predictors contribute most to prediction. Partial-dependence plots can summarize how predicted risk changes with one predictor while averaging over others. These tools are helpful, but they should not be confused with full transparency. A random forest is not a small set of rules. It is an ensemble of many trees. Its interpretation is therefore summary-based rather than path-based.

For expected-loss applications, probability calibration remains important. A random forest may rank cases well but still produce probabilities that are too compressed or poorly calibrated. If the output is used only for prioritization, ranking quality may be sufficient. If the output is used in the expected-loss calculation in equation (8.1), then the numerical level of p_i matters. Calibration checks should therefore accompany random-forest models when they are used as probability inputs to risk calculations.

The role of the random forest in this chapter is therefore clear. It is a strong nonlinear benchmark that improves the stability and predictive power

of tree-based risk models, while sacrificing the simple rule-based transparency of a single tree.

8.6 Gradient Boosting

Gradient boosting also combines many trees, but it does so in a different way from a random forest. A random forest builds many trees independently and averages them. Gradient boosting builds trees sequentially. Each new tree is added to improve the errors left by the current model [7].

The intuition is similar to learning from mistakes. The first tree provides a rough prediction. The next tree focuses on the remaining errors. Later trees continue to correct patterns that earlier trees did not capture. To describe the fitted model, keep the same convention used throughout the chapter: X_i is the full predictor vector for case i, and X_{ij} is predictor j for that case. A boosting model does not usually write the prediction as a simple sum over the individual predictors. Instead, it adds tree functions, where each tree can use several predictors and split the data into regions.

Let $H_m(X_i)$ denote the numerical contribution made by the tree added at boosting step m for case i. After M boosting steps, the model score is

$$F_M(X_i) = \sum_{m=1}^{M} \nu H_m(X_i),$$

where M is the number of boosting steps and ν is the learning rate. The learning rate controls how strongly each new tree changes the current model. A smaller value of ν makes each update more cautious; a larger value makes each tree more influential.

The quantity $F_M(X_i)$ is a model score, not yet a probability. For binary risk prediction, the score is converted into a probability using the logistic transformation:

$$p_i = \frac{1}{1 + \exp[-F_M(X_i)]}.$$

The model is usually trained by reducing log-loss, so it is directly oriented toward probability prediction rather than only class assignment.

The term "gradient" refers to the way the algorithm decides what the next tree should learn. At each step, the model examines the direction in which the loss function can be reduced most quickly. The next tree is fitted to approximate that direction. In simple terms, boosting repeatedly asks: where is the current model still wrong, and what small tree can reduce those errors?

This makes gradient boosting powerful for tabular risk data. It can capture nonlinearities, thresholds, and interactions, while focusing learning effort on cases that are difficult to predict. In credit risk, this may mean accounts whose repayment history is ambiguous. In fraud detection, it may mean transactions that do not look obviously normal or obviously fraudulent. In supplier risk, it may mean firms with mixed signals: some indicators appear stable while others suggest stress.

The flexibility of gradient boosting comes with more tuning choices than logistic regression or a random forest. Important hyperparameters include the number of trees, learning rate, maximum tree depth, minimum leaf size, row subsampling, column subsampling, and regularization. These choices control the balance between learning useful structure and fitting noise.

The learning rate is especially important. A smaller learning rate means that each tree makes only a small correction. This often improves generalization but requires more trees. A larger learning rate allows faster fitting but increases the risk of overfitting. Tree depth controls the complexity of interactions. Shallow trees capture simple effects and low-order interactions. Deeper trees capture more complex interactions but may become unstable if the training sample is not large enough.

XGBoost is a widely used implementation of gradient-boosted trees [3]. Its popularity comes from combining the boosting idea with regularization, shrinkage, subsampling, efficient split search, and strong computational performance. These features are useful in applied risk analytics because risk datasets often include many predictors, missing values, sparse indicators, nonlinear effects, and interactions.

Gradient boosting can produce very strong predictive performance, but it should not be treated as automatically superior. It is less transparent than logistic regression and less directly explainable than a single tree. It can also overfit if tuned poorly. In high-stakes risk settings, the analyst should therefore examine out-of-sample performance, calibration, stability, and the plausibility of the main drivers.

For expected-loss analysis, the same caution applies as with random forests. A boosted model may rank risky cases effectively, but the estimated probabilities must be checked before they are multiplied by exposure and severity. If the output is used in the expected-loss calculation in equation (8.1), then the probability estimate is no longer just a score. It is a numerical input into a monetary calculation.

Gradient boosting therefore plays a specific role in this chapter. It is the flexible high-performance model. It shows how supervised learning can move beyond transparent segmentation and simple log-odds models toward

richer nonlinear risk prediction. Its value must nevertheless be judged by the quality of its probabilities, its out-of-sample stability, and its usefulness for the risk decision being supported.

Table 8.2: Tree-based model families in risk prediction

Model family	**Main modelling logic**	**Risk-analytics implication**
Single tree	One recursive partition of the feature space	Transparent segmentation benchmark, but potentially unstable
Random forest	Many decorrelated trees averaged together	Stable nonlinear prediction; useful benchmark for ranking and probability estimation
Gradient boosting	Trees added sequentially to reduce loss	Flexible high-performance prediction; requires careful tuning and validation
XGBoost	Regularized and computationally efficient gradient boosting	Strong tabular-data benchmark for default, fraud, operational-risk, and severity models

8.7 Empirical Case Study: Comparing Default-Prediction Models

This section extends the credit-card default analysis developed in Chapter 7. The earlier analysis used classification trees to create transparent risk segments. The purpose here is different: the same empirical setting is used to compare supervised-learning models as probability-prediction systems and to show how their outputs become expected-loss quantities once combined with exposure and severity assumptions.

The data are the Taiwan credit-card default dataset, containing 30,000 accounts and a binary indicator of default in the next month [11]. The explanatory variables include credit limit, demographic variables, recent repayment status, bill amounts, and payment amounts. To maintain continuity with the earlier analysis, the case also uses aggregate repayment and exposure variables: average repayment status, maximum repayment status, average bill amount, average payment amount, average utilization, and a payment-to-bill ratio. These variables summarize recent repayment behaviour and exposure intensity without changing the substantive interpretation of the dataset.

The data are split into training, validation, and test partitions using stratified sampling. The training set contains 18,000 accounts, the validation set contains 6,000 accounts, and the test set contains 6,000 accounts. The overall default rate is 22.12%, and the test-set default rate is 22.12%. The test set is used only for the final reported comparison. This separation is important because the empirical question is not whether a model can fit historical observations, but whether it can provide useful probability estimates for accounts not used during fitting.

8.7.1 Model design

The comparison includes four models. The first is the shallow interpretable tree from the previous analysis, represented here by a Gini tree with maximum depth three. It is included as a benchmark because it captures the segmentation logic already developed. The second model is logistic regression, which provides a transparent probability baseline. A regularized logistic model was also estimated as a robustness check, but it produced virtually identical test-set results to the unpenalized logistic model. To keep the comparison compact, only the standard logistic regression is reported. The third model is a random forest, which averages many decorrelated trees. The fourth model is gradient boosting, which builds trees sequentially to reduce prediction error.

All models produce a predicted default probability p_i for each account. The analysis reports probability-oriented metrics rather than a full threshold-policy analysis. AUC and precision–recall AUC summarize ranking quality. The Brier score and log-loss evaluate probability quality. Expected loss is then computed by combining the predicted probability with exposure and an assumed severity parameter.

8.7.2 Predictive performance

Table 8.3 reports test-set predictive performance. The ensemble models improve materially over the interpretable tree in ranking and probability quality. The random forest increases AUC from 0.745 to 0.778, while gradient boosting increases it to 0.778. The improvement is larger in precision–recall AUC, which is especially relevant because default is the minority class. The interpretable tree has PR-AUC of 0.489; random forest and gradient boosting increase this to 0.557 and 0.559, respectively. The Brier score and log-loss also improve for the ensemble models.

Table 8.3: Predictive performance on the test set

Model	**Mean p**	**AUC**	**PR-AUC**	**Brier**	**Log-loss**
Interpretable tree	0.221	0.745	0.489	0.138	0.442
Logistic regression	0.221	0.724	0.498	0.145	0.467
Random forest	0.221	0.778	0.557	0.135	0.431
Gradient boosting	0.221	0.778	0.559	0.135	0.432

The logistic model is useful as a transparent probability baseline, but in this empirical run it does not dominate the shallow tree in overall AUC or probability loss. Its value is therefore not superior nonlinear prediction; it is interpretability, coefficient structure, and a disciplined probability benchmark. The random forest and gradient boosting models create value by capturing nonlinearities and interactions that the shallow tree cannot represent. Relative to the tree benchmark, the random forest increases AUC by 0.033 and PR-AUC by 0.068. Gradient boosting produces a similar AUC gain and the highest PR-AUC in this comparison.

Figure 8.1 shows the ROC curves, while Figure 8.2 shows the precision–recall curves. The ROC curves show that the ensemble models improve ranking over a broad range of false-positive rates. The precision–recall curves are more informative for the high-risk region because they focus on alert quality and default capture when the adverse class is relatively uncommon.

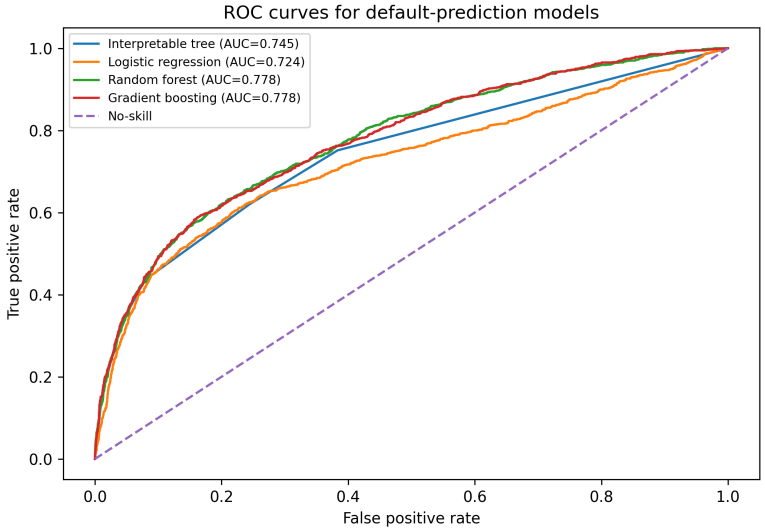

Figure 8.1: ROC curves for the default-prediction models.

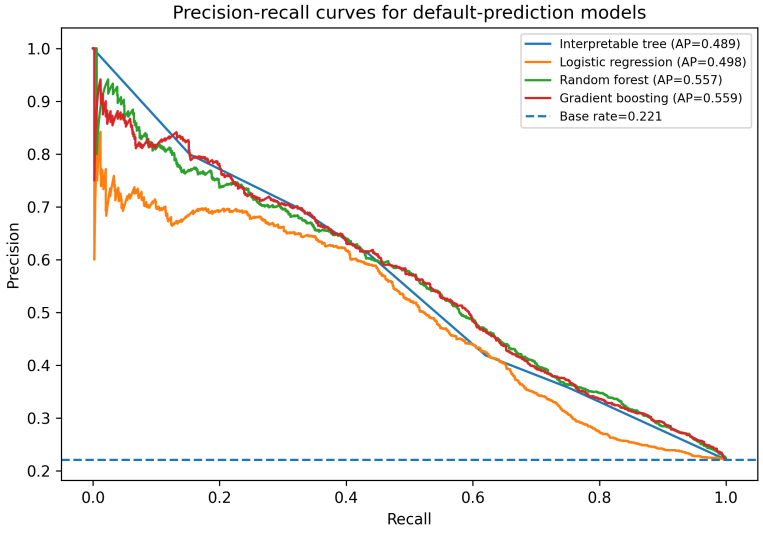

Figure 8.2: Precision–recall curves for the default-prediction models.

8.7.3 Expected loss and sensitivity to severity assumptions

The previous results evaluate default probabilities. To connect probability prediction to risk analytics, the next step combines default probability with exposure and severity. In this empirical illustration, a_i denotes the credit limit of account i, which is used as a simple proxy for exposure. The dataset does not contain realized recovery values or realized loss-given-default, so the severity parameter is treated as a scenario assumption. Let g denote the assumed loss rate if default occurs. Expected loss is computed using equation (8.1), with $g_i = g$ for all accounts in a given severity scenario.

This setup is intentionally simple. It does not claim that the credit limit is a perfect measure of exposure, nor that the assumed severity is estimated from the data. Rather, it shows how a probability model can be connected to an economic risk calculation once the analyst supplies an exposure proxy and a severity assumption.

Table 8.4 reports expected-loss diagnostics under a baseline severity assumption of $g = 50\%$. The value $g = 50\%$ is chosen as a transparent central scenario, not as an empirical estimate. The alternative values in Table 8.5 show how the loss calculation changes when the analyst uses lower or higher severity assumptions. The final column in Table 8.4 shows how much predicted loss is concentrated in the highest-risk 10% of accounts.

Table 8.4: Expected-loss diagnostics for $g = 50\%$

Model	**Mean loss per account**	**Total loss (millions)**	**Top 10% loss share**
Interpretable tree	15,552	93.31	36.2%
Logistic regression	14,175	85.05	32.8%
Random forest	14,698	88.19	37.4%
Gradient boosting	15,019	90.11	35.7%

Notes: Expected loss is computed using equation (8.1). Here a_i is the account credit limit, used as an exposure proxy, and $g = 50\%$ is an illustrative severity scenario. The dataset contains default outcomes but not realized loss-given-default.

Table 8.5 shows the sensitivity of total predicted loss to alternative severity assumptions of 30%, 50%, and 70%. The relationship is mechanically linear in g, but the exercise is useful because it separates two parts of the risk calculation. Model choice determines the distribution of predicted probabilities across accounts. The severity assumption then scales those probabilities into expected loss.

Table 8.5: Sensitivity of total predicted loss to the severity assumption (test portfolio, millions)

Model	$g = 30\%$	$g = 50\%$	$g = 70\%$
Interpretable tree	55.99	93.31	130.63
Logistic regression	51.03	85.05	119.07
Random forest	52.91	88.19	123.46
Gradient boosting	54.07	90.11	126.16

8.7.4 Risk concentration and selected portfolio size

A second sensitivity exercise examines what happens when accounts are selected by predicted risk. This is not a full threshold policy. It is a diagnostic of whether the models concentrate default risk in the highest-scored accounts. Table 8.6 reports the share of observed defaults captured when the top 5%, 10%, and 20% of accounts are selected by predicted probability.

Table 8.6: Defaults captured at different review-capacity levels

Model	**Top 5%**	**Top 10%**	**Top 20%**
Interpretable tree	17.4%	31.9%	47.9%
Logistic regression	15.6%	29.9%	48.5%
Random forest	17.3%	31.0%	51.1%
Gradient boosting	17.9%	31.6%	51.0%

Notes: Accounts are sorted from highest to lowest predicted default probability. Each entry reports the share of all observed defaults contained in the selected group.

Table 8.6 gives the analysis a direct managerial interpretation. If only the highest-risk 5% of accounts can be reviewed, all models identify a group containing about 16–18% of observed defaults. If review capacity expands to the highest-risk 20%, the random forest and gradient boosting models capture about 51% of all observed defaults. The table therefore translates model ranking into an operational question: how many cases can the organization afford to review, and how much default risk is concentrated in that selected group?

The decile analysis in Table 8.7 uses the best model by log-loss, which in this run is the random forest. Accounts are sorted by predicted default probability and divided into ten equally sized groups. The table has two purposes. First, it checks whether predicted probabilities increase in line with observed default rates. Second, it shows whether expected loss is concentrated in the highest-risk groups.

Table 8.7: Selected risk-decile diagnostics for the best probability model

Risk group	**Accounts**	**Mean p**	**Observed default**	**Loss share**
Lowest 10%	600	5.0%	4.5%	5.0%
Middle 40–60%	1,200	12.5%	12.2%	14.2%
High 70–80%	1,200	24.0%	22.4%	22.5%
Highest 10%	600	67.3%	68.7%	23.1%
Highest 20%	1,200	55.2%	56.5%	39.6%

Notes: Accounts are sorted by predicted default probability. Loss share is computed using equation (8.1), with a_i equal to the account credit limit and $g = 50\%$. The value $g = 50\%$ is an illustrative severity scenario, not an estimate from the dataset.

The pattern is clear. The lowest-risk 10% of accounts has an observed default rate of only 4.5%, while the highest-risk 10% has an observed default rate of 68.7%. The highest-risk 20% of accounts accounts for 39.6% of

predicted expected loss. This is the managerial value of risk ranking: it identifies where attention, monitoring, or intervention is likely to matter most. Figure 8.3 shows the full decile-level concentration pattern.

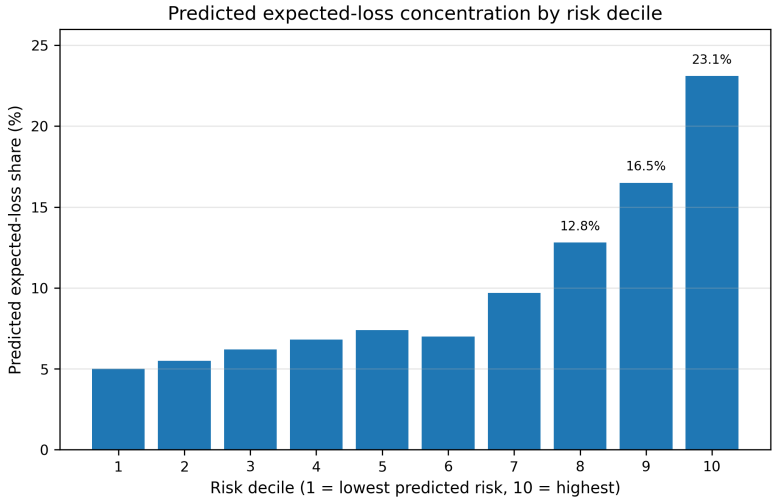

Figure 8.3: Predicted expected-loss concentration by risk decile for the best probability model.

Figure 8.4 compares mean predicted default probability with the observed default rate across risk deciles. The alignment is not perfect, but it is sufficiently close to show that the model's probabilities have plausible empirical meaning for this expected-loss illustration. A full calibration analysis is a separate decision-support task.

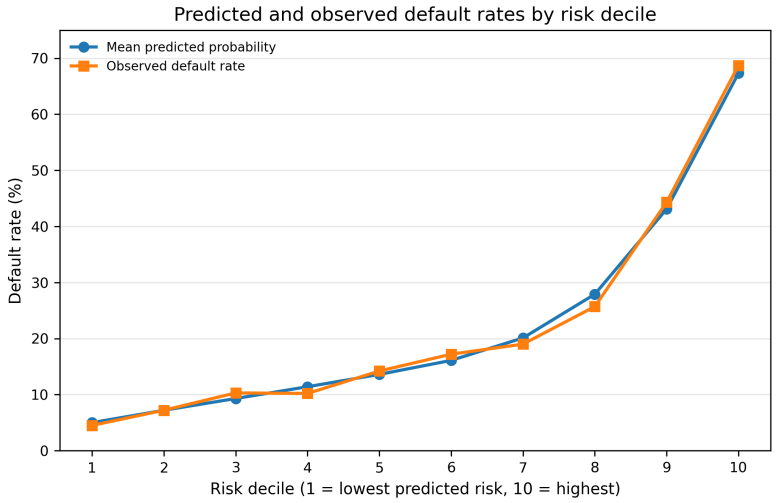

Figure 8.4: Mean predicted probability and observed default rate by risk decile for the best probability model.

8.7.5 Model drivers and managerial interpretation

The final part of the case study examines which variables drive the ensemble models. This is not a substitute for causal analysis. Feature-importance scores show which variables a model uses most strongly for prediction; they do not prove that a variable causes default. Even so, they are useful for model interpretation because they indicate which business signals should be examined when explaining predictions.

The variable names in the original dataset are compact and not always self-explanatory. The variable `PAY_0` records the most recent repayment status. The variables `PAY_2`, `PAY_3`, and `PAY_4` record earlier repayment-status measures. The constructed variable `PAY_STATUS_MEAN` summarizes average repayment status across recent months, while `PAY_STATUS_MAX` records the worst recent repayment status. The variable `PAY_AMT_MEAN` summarizes average recent payments, `BILL_AMT_MEAN` summarizes average recent bill balances, `UTILIZATION_MEAN` measures average use of the credit limit, and `PAY_TO_BILL_MEAN` summarizes repayment intensity relative to billed amounts.

Figures 8.5 and 8.6 report the main feature-importance patterns for the random forest and gradient-boosting models. The two figures lead to a consistent interpretation: recent repayment behaviour is the dominant risk signal. Both models place the most recent repayment status and summary

measures of repayment history among the most important predictors. This is substantively plausible because recent delinquency and persistent repayment weakness are direct indicators of default risk.

The two models differ in how concentrated their importance scores are. The random forest spreads importance across several repayment-history variables, reflecting its averaging over many alternative trees. Gradient boosting places much more weight on the most recent repayment status, indicating that this variable is especially useful in the sequential loss-reduction process. The managerial conclusion is not that one variable mechanically causes default, but that repayment behaviour is the first area to examine when explaining model predictions, designing monitoring rules, or communicating risk drivers.

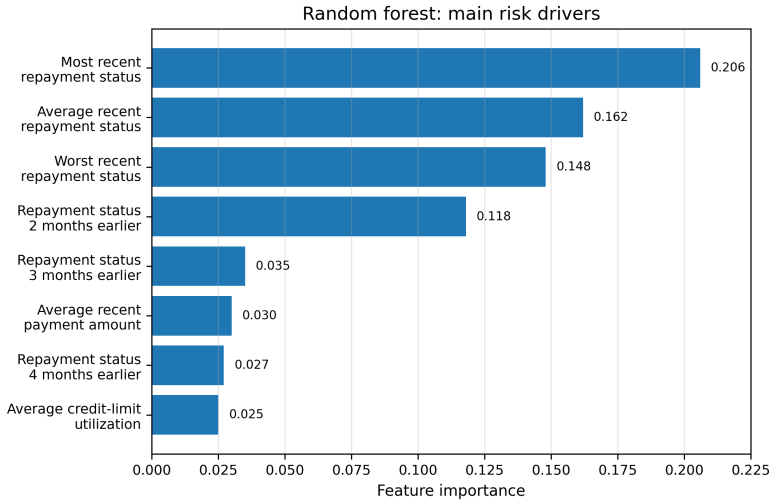

Figure 8.5: Top feature importances for the random forest model.

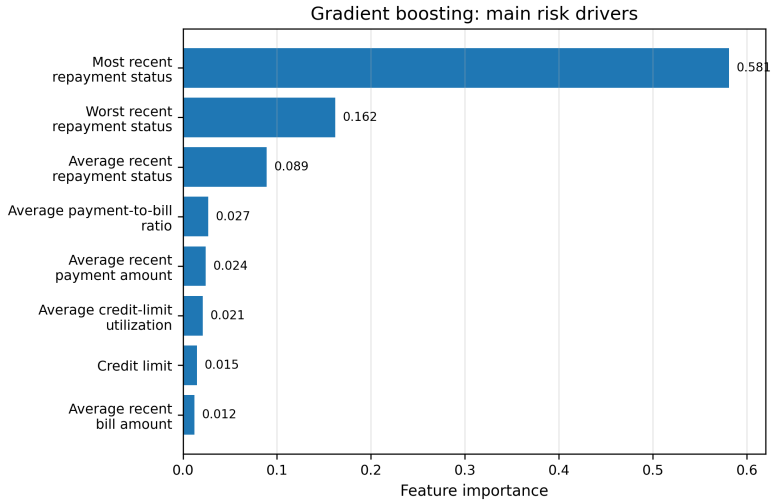

Figure 8.6: Top feature importances for the gradient-boosting model.

8.8 Summary and Managerial Lessons

This chapter extended the move from risk segmentation to risk prediction. Classification trees remain useful because they make risk logic visible, but many applications require more than transparent groups. They require probabilities, rankings, and expected-loss inputs.

The chapter developed a simple applied framework. A supervised-learning model produces a probability p_i for each case. That probability becomes a risk quantity only when it is connected to exposure and severity. This chapter used equation (8.1) to express that link: probability is not the whole of risk, but it is the bridge between prediction and expected loss.

Logistic regression was presented as the transparent probability benchmark. Random forests were introduced as a way to stabilize and enrich tree-based prediction by averaging many decorrelated trees. Gradient boosting was introduced as a sequential ensemble method that can capture nonlinear patterns and interactions in tabular risk data. The empirical credit-card case showed that ensemble models can improve ranking and probability performance, while transparent models remain useful for communication and governance.

The managerial lesson is direct: model choice should be driven by the decision being supported. If the task is explanation, a transparent model may be preferred. If the task is review prioritization, ranking and concentration

diagnostics matter most. If the task is expected-loss estimation, probability quality and calibration become essential.

Managerial takeaways

- Use predicted probabilities when decisions depend on exposure or severity.
- Convert probabilities into expected loss before ranking economic risk.
- Keep transparent benchmarks even when ensemble models predict better.
- Evaluate models by their intended use: explanation, ranking, monitoring, pricing, or intervention.
- Communicate model drivers in business language, not only as variable names.

Bibliography

[1] Edward J. Anderson. *Business Risk Management: Models and Analysis*. John Wiley & Sons, Ltd, Chichester, West Sussex, UK, 2014.

[2] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[3] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[4] David R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–242, 1958.

[5] David Firth. Bias reduction of maximum likelihood estimates. *Biometrika*, 80(1):27–38, 1993.

[6] Ronald A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A*, 222:309–368, 1922.

[7] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[8] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, 2 edition, 2009.

[9] Joseph M. Hilbe. *Practical Guide to Logistic Regression*. CRC Press, Boca Raton, FL, 2015.

[10] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer, New York, 2 edition, 2021.

[11] I-Cheng Yeh and Che-hui Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2):2473–2480, 2009.

[12] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. CRC Press, 2012.

