# Improving Model Accuracy with Probability Scoring Machine Learning Models

Juily Vasandani
Krannert School of Management
Purdue University
West Lafayette, IN
jvasanda@purdue.edu

Saumya Bharti
Krannert School of Management
Purdue University
West Lafayette, IN
bhartis@purdue.edu

Deepankar Singh
Krannert School of Management
Purdue University
West Lafayette, IN
singh681@purdue.edu

Shreeansh Priyadarshi
Krannert School of Management
Purdue University
West Lafayette, IN
spriyad@purdue.edu

***Abstract*—Binary classification problems are exceedingly common across corporations, regardless of their industry. Popular examples include classifying a patient as high-risk or low-risk or predicting if a client will convert or not. The motivation for this research is determining techniques to improve prediction accuracy for operationalized models. Collaborating with a national partner, we conducted feature importance tests and engineering experiments to isolate industry-agnostic factors with the most significant impact on the conversion rate. We also use probability scoring to highlight incremental changes in accuracy as we applied several improvement techniques to determine which would significantly increases a model's predictive power. We compare five algorithms: XGBoost, LGBoost, CatBoost, and MLP, and an ensemble of all four, while our results highlight the superior accuracy of the ensemble, with a final log loss value of 0.5784. We also note that the highest levels of improvement in log loss occurs at the beginning of the process, after downsampling and using engineered custom metrics as inputs to the models.**

***Keywords—conversion, binary classification, probability scoring, log loss, downsampling, hyperparameter tuning, feature engineering, XGBoost, LGBoost, CatBoost, MLP, ANN, ensemble***

## I. Introduction

Binary classification is the simple task of making a prediction as to which of two groups a given observation belongs to on the basis of a specified classification rule. They also exist everywhere in society – from hospitals predicting high-risk vs low-risk patients to credit card companies categorizing applications as good vs bad credit, and even quality control in manufacturing processes (pass/fail). Its goal is simply to categorize data points into one of two different buckets, making it one of the simplest kinds of supervised learning problems, and machine-learning models are quickly becoming the immediate solution across different industries. These models extract information automatically using computational and statistical methods, giving software the ability to build knowledge from experience, derived from patterns and rules extracted from a large volume of data [1].

Typical classifiers commonly used for these problems include decision trees, logistic regression, support vector machines, and neural networks. Each method performs best under specific circumstances – depending on the number of observations, the dimensionality of the predictors, the noise in the data, and many other factors. To add to the complexity, there are also several known evaluation metrics that can be used to measure the performance of a classification model ranging from simple accuracy or misclassification measures to precision, recall, log loss, ROC curve, and the AUC score, which can show more insightful information about the classifier's performance [2]. Deciding on the eventual performance measure tends to be subjective since they have to align with the project objectives, and different measures may alter final predictions, making this choice a significant part of the overall classification problem.

While much of the research in this space has focused on the accuracies produced by different classifiers utilizing various probability scoring methods, no comparison of other accuracy improving methods (e.g. hyperparameter tuning, feature engineering, or ensemble techniques) has yet been performed. This paper presents a comprehensive comparison of different machine- learning models designed to solve a binary classification problem, using proprietary data from our partner company to understand the drivers that impact their conversion rate. Though traditionally defined within the context of digital marketing and e-commerce, a conversion generally occurs when users (or clients) take a desired action [3], not limited to a sale or a purchase. A conversion can be any key performance indicator (KPI) relevant to a firm and for the purpose of this study, it is defined as the instance a sales representative completes a scheduled home visit with a lead generated by the company's marketing team. Our study not only evaluates each model, but various methods to improve each classifier's accuracy as well, particularly those that have not yet been addressed in the literature. Based on our experimental results, binary classification problems can be solved using different combination of models and accuracy tuning methods, which can be then evaluated using different criteria that works with the overall business problem. The motivation behind our study is to evaluate different ways to improve models already in production, since many organizations would likely already have basic models in place to solve their classification problems. Our aim is to answer the following research questions:

1. What industry-agnostic factors have a significant impact on conversion rates?
2. Which accuracy improvement techniques (e.g. sampling methods, hyperparameter tuning, feature engineering, etc.) can significantly increase a model's predictive power?

The following section reviews the past literature on various criteria and methods used to solve binary classification problems, including evaluating different models and

performance measures traditionally used in this space. It is immediately followed by a deeper look into the context and background of the data used in our study, before a dissection of our methodology in the third section. We outline the different models we investigated, as well as the various methods tested against them to improve their accuracy. Finally, we present our results, discuss our conclusions and lay out the framework for further research and potential future scope of this project.

## II. Literature Review

### A. Model Selection

Regardless of industry, every person – whether they have made a single purchase or have been a loyal customer for many years – will eventually cease their relationship with a business. This loss, commonly known as customer attrition or churn, is one of the most popular business applications of the binary classification problem, with a wealth of resources dedicated to studying various solutions and their accuracy. Churn models aim to identify early churn signals and recognize customers with an increased likelihood to leave voluntarily, using various parameters and different evaluation methods to predict the attrition occurrence. A performance comparison between Monte Carlo simulations of five well-established techniques used for churn prediction showed that the best classifier was the SVM-POLY model tuned with AdaBoost [4]. However, later conclusions studying telecommunications churn found tree-based algorithms to be the best models to apply towards a real-time attrition problem [5, 6]. These contradicting results serve to show us that whilst algorithm selection is very important, there are several other factors that impact the effectiveness of a chosen machine learning solution.

Although the appropriate choice for a supervised learning algorithm depends on the task at hand, each of these algorithms has their own pros and cons, and there are cases when practitioners may select an inappropriate algorithm for their solution. No single model can uniformly outperform the others across all datasets, and the comparative summary of classification techniques included in Table 2.1 below provides a truncated overview of model performance across several features [7]. When dealing with this type of machine learning problem, the key question is not whether a learning algorithm is superior to others, but under which conditions can a model outperform others on an applied problem. Although it may not give us a deterministic answer for model choice, it gives us a preliminary idea of which models to start fitting to our data based on the given problem.

| | **Algorithms** | | |
|---|---|---|---|
| | ***Decision Trees*** | ***Neural Networks*** | ***Naïve Bayes*** |
| Accuracy in general | ** | *** | * |
| Speed of learning with respect to number of attributes and instances | *** | * | **** |
| Speed of classification | **** | **** | **** |
| Tolerance to missing values | *** | * | **** |
| Tolerance to irrelevant attributes | *** | * | ** |
| Tolerance to redundant attributes | ** | ** | * |
| Tolerance to highly interdependent attributes (e.g. parity problems) | ** | *** | * |
| Dealing with discrete, binary, or continuous attributes | **** | *** (not discrete) | ***(not cont.) |
| Tolerance to noise | ** | ** | *** |
| Dealing with danger of overfitting | ** | * | *** |
| Attempts for incremental learning | ** | *** | **** |
| Explanation ability/transparency of knowledge/classifications | **** | * | **** |
| Model parameter handling | *** | * | **** |

**Table 2.1:** Comparing algorithms (**** represents the best and * the worst performance)

An ensemble model combines classifiers as an extension of algorithm selection, improving the performance of individual classifiers. There are many ways to build an ensemble, and some of these mechanisms include: i) using different subsets of training data with a single learning method, ii) using different training parameters with a single training method, and iii) using different learning methods [7]. Despite the many varied base models used to predict an outcome within an ensemble model, the ensemble acts and performs as a single model. During this process, the generalization error of the prediction is reduced, as long as the base models are diverse and independent of each other [8].

### B. Probability Scoring

Model fit depends on the kind of data available and interpretation depends on the evaluation measure chosen, a model's performance can also be tuned towards better performance. Tuning a machine learning model is very much like turning the switches and knobs of an antique TV in order to get a clearer picture. The eventual goal is to improve model accuracy, which can be achieved by optimization techniques such as sampling methods, feature extraction and engineering, metric creation and selection, hyperparameter tuning, and ensemble methods, among others. The ability to effectively gauge the impact of each optimization technique on overall model accuracy is a significant objective to address the research questions in this study, making our choice of evaluation metric very important.

A classification algorithm's prediction score indicates its certainty that the given observation belongs to the positive class. To make the decision about whether the observation should be classified as positive or negative, a classification threshold (cut-off) is selected and compared against its score. Any observations with scores higher than the threshold are then predicted as the positive class and scores lower than the threshold are predicted as the negative class. Once observations are classified, the prediction falls into four categories, depending on the actual observed answer and the predicted answer, where true positives and negatives imply that the observed and the predicted class are the same, while false positives and negatives indicate that misclassification has occurred for the given observations.

Likely the most widely-accepted performance measurement metric for classification, the AUC – ROC curve represents the

ability of a model to distinguish between classes. The ROC curve is plotted with the True Positive Rate (TPR) on the y-axis and the False Positive Rate (FPR) on the x- axis. A higher AUC value indicates a model accurately predicting 0s as 0s, and 1s as 1s – effectively distinguishing between a customer that will churn vs one who will not.

While the goal of traditional binary classifiers revolves around predicting an observation's class label, certain problems require the additional nuance of measuring the likelihood that each example belongs to a given class. In the empirical work on probabilistic prediction in machine learning, the most standard loss functions are log loss, Brier loss, and spherical loss. In determining which loss function is likely to lead to better prediction algorithms, it was found that log loss is more selective and is likely to lead to better algorithms as a result – if a model is optimal under the log loss function, it will be optimal under the other two loss functions, but the opposite may not be true [9]. Log loss takes into account the uncertainty of a prediction based on how much it varies from the actual label by assigning a probability to each class rather than simply yielding the most likely class, quantifying accuracy of the prediction and then penalizing false classifications. By placing heavy penalties on classifiers that are confident about an incorrect classification, the model learns that it is better to be somewhat wrong rather than completely wrong [10].

For the purpose of our work, we have selected the log loss function as our evaluation metric because it incorporates probabilistic confidence as a measurement of an algorithm's prediction accuracy. In binary classification settings, as in our paper, the log loss for each observation equals:

$$H(p,q) = -\sum p_i \log q_i = -y \log \hat{y} - (1-y)\log(1-\hat{y}) \qquad (1)$$

Here, y refers to the actual class label, while $\hat{y}$ represents the predicted class label. By doing so, we will be able to see the incremental changes in accuracy and determine the impact of each individual optimization technique, clearly determining which technique can be applied to operationalized models already integrated into a company's production process.

## III. DATA

Utilizing proprietary data from our industry partner, the database consists of tables that capture business information regarding their customers and their geographic location, company projects, lead sources, employees, and their offered product selections. We used the entirety of the data for the initial analysis, excluding the tables irrelevant to our given problem. A master query allowed us to work only on the latest 5 years of data in order to capture the variation within the data and incorporate specific business constraints, leading to a final dataset with over 3,425,646 observations. We also maintained a strict coverage threshold of 50% for the model's attributes.

In table 3.1, we have grouped a total of 52 features into 7 categories, outlined below:

| Variable Categories | Description |
|---|---|
| Lead ID | Primary key for the master table, unique identifier for a lead |
| Homeowner Information | Zip Code, City, State |
| Product | A list of the product estimates for the lead |
| Source Information | Description of the which category or group the lead sources comes under |
| Datetime Information | Date and time of lead call, date and time of the impending appointment, Date and time when the lead was sourced |
| User Performance | Prior performance of the salespersons involved in the lead process, based on the zip code and time periods |
| Conversion Rates | Prior conversion rates in the area, prior leads in the area |

**Table 3.1:** Data used in study

## IV. METHODOLOGY

### A. Data Exploration and Pre-Processing

#### 1) Data Cleaning and Health

The initial stage of any machine learning project involves ensuring the relevance of your data to the current problem, identifying and treating outliers as well as any missing values. Our raw data contained junk values across tables that needed to be removed before the analysis. We filtered and excluded datapoints older than 5 years and removed features with less than 50% coverage to ensure results applicable to our industry partners and their problems during feature importance experiments.

#### 2) Exploratory Data Analysis

Once we filtered our data to meet our requirements, we explored the data to understand the following:

- Distribution and fill-rate of features
- Correlations between predictors
- Associations between predictors and the target variable

### B. Feature Engineering and Data Partitioning

#### 1) Hypothesis Development

Once we developed our master dataset and have gained an understanding of the business impact associated with the features, we developed hypotheses to understand the internal and external properties that directly impact the relationship between our predictors and the target.

- $H_1$: Impact of *intrinsic* predictors on the conversion rate
  - Past employee performance
  - Employee experience
  - Appointment details
  - Past neighborhood performance
- $H_2$: Impact of *extrinsic* predictors on the conversion rate
  - Lead source information
  - Customer details
  - Preferred service details

*2) Custom Metric Creation*

During this phase of the project, we generated several features that could be directly inputted into our model as a predictor. They were engineered to address our hypotheses regarding factors that impact our target variable. Some of the metrics we created are:

- **Lead Gap:** Difference in time when the lead was created to when the actual appointment was set
- **Previous Status:** identifying information on whether the lead was a prospect approached previously or a customer already once converted
- 6- and 12-month conversion rates split by geography
- 6- and 12-month records of employee performance handling that lead in that location

*3) Data Partition*

In order to perform tests on the model's predictive power, we split our data 80/20% into a training and testing set, respectively. To allow our model to learn better and given that the target class populated only 28% of the times in the original data, we knew we needed to downsample the data from our majority class to have the same number of observations of both the target and non-target class. Even after downsampling, we were still left with nearly 1.5 million observations for our training dataset.

### C. Model Building, Evaluation, and Ensemble Creation

*1) Base Model Training*

Given the properties of our data, we chose 7 initial classifiers to fit to our data:

1. Logistic Regression
2. Random Forest
3. Multilayer Perceptron (MLP) Artificial Neural Network
4. 4x Boosting Models (Gradient Boosting Machine, Light Gradient Boosting, CatBoost, and XGBoost)

*2) Probability Scoring*

We use the log loss model to estimate the performance of the classification model to gauge incremental improvements.

*3) Ensemble Creation*

Based on the algorithm's log loss score, we selected the four best-performing models to ensemble. In this case, the **XGBoost**, **Light Gradient Boosting**, **CatBoost**, and the **Artificial Neural Network** had both the lowest log loss values as well as the highest AUC.

## V. Models

As explored in the literature review section, there is no deterministic choice when it comes to selecting the right classifier. Rather, it is important to understand under which conditions an algorithm would outperform others, whether this is the quality and type of data or the type of application problem. The nature of a binary classification problem combined with the special attributes of our partner's proprietary data meant that we had several choices for technique, detailed below.

### A. Logistic Regression

A statistical method borrowed for machine learning problems, logistic regression is used when our target variable is categorical and is often considered the go-to method for binary classification problems. A logit model estimates the probability of the binary target using a linear combination of the predictor variables as arguments of the sigmoid function, outputting a number between 0 and 1 – given a threshold that establishes which values belong to class 1 and to class 0 [11].

$$y = e^{b_0+b_1x}/1 + e^{b_0+b_1s} \quad (2)$$

Similar to linear regressions, input values ($x$) are linearly combined using weights or coefficients (represented by $b$) to predict an output value ($y$). The main difference between them relates to their modeled output, with logistic regression being categorical rather than continuous [12].

### B. Random Forest

Random forests are a type of learning technique for classification that operates by constructing a multitude of decision trees with low correlation that operate as a group, with each tree classifying an observation into a class. The class which ultimately has the greatest number of votes across the forest is used to classify the observation [13]. It uses bagging and feature randomness when building each individual tree to create an uncorrelated forest of trees whose collective prediction is more accurate than of any individual tree. The basic premise behind their success is that as a collective group, the trees can protect each other from their individual errors [14].

There are many different evaluation metrics for tree-based models, but the most common and the one we have chosen to use for the purposes of our research is the Gini Impurity measure, which calculates the probability that a randomly chosen sample in a node would be incorrectly labeled if it was labeled by the distribution of the sample in that given node.

$$I_G(n) = 1 - \sum_{i=1}^{J}(p_i)^2 \quad (3)$$

At each node, the decision tree searches through the features for the value to split on that results in the greatest reduction in the Gini coefficient, repeating the splitting process in a greedy, recursive procedure until its pre-assigned maximum depth, or if each node contains only samples from a single class [15].

### C. Artificial Neural Network

A multilayer perceptron (MLP) is a class of artificial neural network that can be viewed as a logistic regression classifier. The input is first transformed using a learnt non-linear transformation and are composed of at least three layers [8]: the input layer (receives the signal), the output layer (to make a prediction about the input), and an arbitrary number of hidden layers in between them (the computational engine of the neural network). They train on a set of input-output pairs to model the correlations between the pairs, tuning the parameters by adjusting the weights and biases of the model to minimize errors [16].

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b) = \varphi(w^T x + b) \quad (4)$$

In the equation above, the perceptron produces a single output based on several real-valued inputs by forming a linear combination using its input weights (whose vectors are x and w, respectively) and sometimes passing the output through a nonlinear activation function – represented by $\varphi$. Our model in particular has four hidden layers with nodes sized 120, 60, 30, and 10, respectively. All nodes have the RELU (Rectified Linear Unit) activation function.

### D. Boosting Models

The term boosting in machine learning refers to a family of algorithms that focus on training a series of weak learners to convert them into strong learners [17], rather than traditional algorithms that focus on high quality predictions done by a single model. For the purposes of our study, we focused particularly on these models: Categorical Boosting (CatBoost), Extreme Gradient Boosting (XGBoost), Light Gradient Boosting (LGBoost), and Gradient Boosting Machine (GBM). Despite their similarities, each algorithm boasts their own pros and their own cons, including speed, accuracy, or ability to handle outliers and missing data, further reiterating the importance of validation to determine the appropriate – and the most effective – model for the given problem.

### E. Ensemble Model

Ensemble models – a collection of several models working together on a single dataset – tend to be reliable and accurate due to the diversity of the models that are contained within them. There are two main ways to determine the final prediction of the ensemble: hard vs. soft voting. Hard voting is when a model is selected by the ensemble to provide the final prediction by a simple majority vote for accuracy. On the other hand, soft voting can only be done when classifiers calculate probabilities for the outcomes, relevant to our probability scoring metric [18]. Soft voting arrives at the best result by averaging out the probabilities calculated by individual algorithms.

## VI. Results

### A. Feature Importance

Despite our data source coming from within the home improvement space, we paid particular attention to converting our results into industry-agnostic insights. To do this, we group our features into buckets that were more applicable outside our partner company, rather than highlight individual variables. In particular, we found **employee performance**, **neighborhood performance**, and **employee experience** to be the biggest contributors to a lead conversion.

It is interesting to note that amongst the top 5 features that

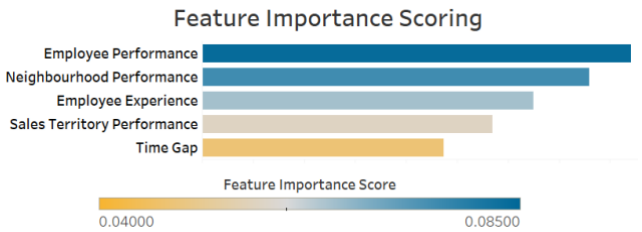


**Figure 6.3:** Feature Importance

contribute ~33% towards the probability of the final predicted class are **all extrinsic features**. This is particularly significant as these are the factors that companies have a higher level of control over, giving our partner company and any other businesses hoping to maximize their conversion rates a large scope for optimization and overall improvement.

### B. Model

Once we had our models (Light Gradient Boosting, XGBoost, CatBoost, the MLP, and the ensemble), we calculated log loss values to study the improvements across each tuning method. At each stage of the process, the values decrease, highlighting the value gained directly from each technique.

| | LGB | XGB | CB | ANN | E |
|---|---|---|---|---|---|
| Base Model | 0.8452 | 0.8524 | 0.8298 | 0.8269 | 0.8183 |
| Downsampling | 0.6531 | 0.6672 | 0.6547 | 0.6535 | 0.6525 |
| Feature Engineering | 0.5927 | 0.6045 | 0.5938 | 0.5832 | 0.5897 |
| Parameter Tuning | 0.5781 | 0.5871 | 0.5789 | 0.5785 | 0.5784 |

**Table 6.1:** Incremental log loss improvements across models

Throughout our research, our ensemble model consistently performed the best, with a **final log loss value of 0.5784**. However, one very surprising finding was the performance of our Artificial Neural Network, outperforming the ensemble after incorporating our engineered metrics into the model – with a log loss score of 0.5832 compared to the ensemble's score of 0.5897 – a relatively significant difference. However, after the final accuracy improvement technique of hyperparameter tuning, the neural network's final log loss was 0.5785, falling short of the best-performing log loss score by just one unit.

This further emphasizes the need to test various models before making a conclusive decision on which supervised learning algorithm should be used to address a given problem. In addition, it is clear that the marginal improvement from the base model towards the later stages of improving accuracy declines sharply, with an average decrease of just under 0.03 across them models. On the other hand, the decrease in log loss after performing feature engineering experts across models reached an overage of 0.3, over **10x more significant than the impact of hypertuning**. This is because the amount by which log loss can decrease is constrained, while increases in the log loss value are unbounded [10].

## VII. Conclusion

Throughout our research project, our objective was to build and improve upon supervised learning predictive models by comparing their probability score calculated through the log loss function. In doing so, we hope to provide insights to businesses who have already operationalized a base model and help to develop heuristics to improve those models already in production. Our focus for this research project was to gain industry-agnostic insights that can be applied to any binary classification problem that requires a conversion to occur.

Our exploratory analysis found that the top 5 most important features, accounting for approximately 33% of the probability of the prediction, are all **extrinsic factors** – which are the features that a company has more control over, including **employee performance and experience**, and **time gaps** (between touchpoints in the sales funnel). With clearer insight into the drivers of the prediction, our results illustrate that optimizing these features would likely lead to boosted conversion rates.

Further, by evaluating our classifiers with the log loss probability scoring method, we are able to see the incremental improvements in accuracy after each improvement technique is applied to the model. From our results, it is clear that **downsampling our data** and **feature engineering** both significantly improved the model's predictability, reducing log loss over **10x more** than through hyperparameter tuning. Across all the models, the ensemble came out as the clear winner, with a final log loss score of **0.5784**, followed very closely by our artificial neural network model – scoring **0.5785**.

These results tell us that while the ensemble has performed the best, there may be scenarios and cases where a neural network would make predictions quicker, and more accurately, further reiterating that there is no fixed model that will outperform others across all circumstances. Instead, finding the best solution depends on your data, as well as your model choice, what tuning methods have been applied, and what evaluation metric you are using to measure your success.

Although our research culminated in very relevant insights both for our partner company and other organizations looking to optimize their predictive classifiers, there is additional scope for improvement mainly relating to increasing the diversity of our data sources. Using more data on customer demographics, such as age or occupation, as well as data from competitors and other external sources, our partner company will be able to further improve the model as they operationalize it and integrate it into their current production process.

## Acknowledgment

We would like to thank both our industry partner and Professor Lanham for their continued support and guidance.

## References

[1] N. Prasasti and H. Ohwada, "Applicability of machine-learning techniques in predicting customer defection," 2014 International Symposium on Technology Management and Emerging Technologies, Bandung, 2014, pp. 157-162.

[2] A. Keramati, R. Jafari-Marandi, M. Aliannejadi, I. Ahmadian, M. Mozaffari, U. Abbasi, "Improved churn prediction in telecommunication industry using data mining techniques," Applied Soft Computing, Volume 24, 2014, Pages 994-1012, ISSN 1568-4946, doi: 10.1016/j.asoc.2014.08.041.

[3] J. Nielsen, "Conversion Rate: Definition as used in UX and web analytics," *Nielsen Norman Group*. [Online]. Available: https://www.nngroup.com/articles/conversion-rates/. [Accessed: 14-Mar-2020].

[4] T. Vafeiadis, K. I. Diamantaras, G. Sarigiannidis, K. Ch. Chatzisavvas, "A comparison of machine learning techniques for customer churn prediction," Simulation Modelling Practice and Theory, Volume 55, 2015, Pages 1-9, ISSN 1569-190X, doi: 10.1016/j.simpat.2015.03.003.

[5] J. Pamina, J. B. Raja, S. S. Peter, S. Soundarya, S. S. Bama, M. S. Sruthi, "Inferring Machine Learning Based Parameter Estimation for Telecom Churn Prediction," in *ICCVBIC*, S. Smys, J. Tavares, V. Balas, A. Iliyasu, Eds. 2019, Advances in Intelligent Systems and Computing, vol 1108. Springer, Cham.

[6] A. K. Ahmad, A. Jafar, K. Aljoumaa, "Customer churn prediction in telecom using machine learning in big data platform," Big Data 6, 28 2019, doi: 10.1186/s40537-019-0191-6.

[7] S. B. Kotsiantis, I. Zaharakis, P. Pintelas, "Supervised machine learning: A review of classification techniques," *Artificial Intelligence Review*, vol. 26, no. 3, pp. 159–190, 2006.

[8] V. Kotu and B. Deshpande, *Data science: concepts and practice*. Cambridge, MA: Morgan Kaufmann is an imprint of Elsevier, 2019.

[9] V. Vovk, "The Fundamental Nature of the Log Loss Function," *Fields of Logic and Computation II Lecture Notes in Computer Science*, pp. 307–318, 2015.

[10] A. B. Collier, "Making Sense of Logarithmic Loss," *datawookie*, 14-Dec-2015. [Online]. Available: https://datawookie.netlify.com/blog/2015/12/making-sense-of-logarithmic-loss/. [Accessed: 18-Mar-2020].

[11] A. Urso, A. Fiannaca, M. L. Rosa, V. Ravì, and R. Rizzo, "Data Mining: Prediction Methods," *Encyclopedia of Bioinformatics and Computational Biology*, pp. 413–430, 2019.

[12] R. Vasudev, "How are Logistic Regression & Ordinary Least Squares Regression (Linear Regression) Related?," *Medium*, 05-Jun-2018. [Online]. Available: https://towardsdatascience.com/how-are-logistic-regression-ordinary-least-squares-regression-related-1deab32d79f5. [Accessed: 25-Mar-2020].

[13] Breiman, L. Random forests. *Machine learning*, *45*(1), 5-32. 2001.

[14] T. Yiu, "Understanding Random Forest," *Medium*, 14-Aug-2019. [Online]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2. [Accessed: 22-Mar-2020].

[15] W. Koehrsen, "An Implementation and Explanation of the Random Forest in Python," *Medium*, 31-Aug-2018. [Online]. Available: https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76. [Accessed: 18-Mar-2020].

[16] C. Nicholson, "A Beginner's Guide to Multilayer Perceptrons (MLP)," *Pathmind*. [Online]. Available: https://pathmind.com/wiki/multilayer-perceptron. [Accessed: 14-Mar-2020].

[17] Brownlee, J. A gentle introduction to the gradient boosting algorithm for machine learning. *Machine Learning Mastery. Nov*, *9*. 2016.

[18] S. Howal, "Ensemble Learning in Machine Learning: Getting Started," *Medium*, 15-Dec-2018. [Online]. Available: https://towardsdatascience.com/ensemble-learning-in-machine-learning-getting-started-4ed85eb38e00. [Accessed: 19-Mar-2020].

[19] P. Gaspar, J. Carbonell, and J. L. Oliveira, "On the parameter optimization of Support Vector Machines for binary classification," *Journal of Integrative Bioinformatics*, vol. 9, no. 3, Jan. 2012.

[20] P. A. Flach and N. Lachiche, "Improving accuracy and cost of two-class and multi-class probabilistic classifiers using ROC curves," *Machine Learning*, vol. 42, no. 1/2, pp. 61–95, 2001.

[21] A. Rojas-Domínguez, L. C. Padierna, J. M. Carpio Valadez, H. J. Puga-Soberanes and H. J. Fraire, "Optimal Hyper-Parameter Tuning of SVM Classifiers With Application to Medical Diagnosis," in IEEE Access, vol. 6, pp. 7164-7176, 2018.