Dive into Machine Learning with the Tidyverse and Mlr

Machine learning (ML) is a rapidly growing field that has the potential to revolutionize many industries. The Tidyverse is a popular collection of R packages for data science that provides a consistent and powerful syntax for data wrangling, analysis, and visualization. The mlr package extends the Tidyverse with a comprehensive set of tools for machine learning.

This article will provide a gentle to machine learning with the Tidyverse and mlr. We will start by covering the basics of ML, including supervised and unsupervised learning, model evaluation, and feature engineering. We will then show how to use the Tidyverse and mlr to build and evaluate machine learning models. Finally, we will provide some resources for further learning.

To follow along with this article you will need to have basic experience with the R programming language and the Tidyverse. You will also need to install the mlr package. You can install mlr with the following command:

Machine Learning with R, the tidyverse, and mlr

by Frank Kane

4.4 out of 5

Language	: English
File size	: 16223 KB
Text-to-Speech	: Enabled
Screen Reader	: Supported
Enhanced typesetting	: Enabled
Print length	: 536 pages

```
install.packages("mlr")
```

Machine learning is a type of artificial intelligence (AI) that allows computers to learn without being explicitly programmed. ML algorithms are trained on data, and they can then use that training to make predictions or decisions.

There are two main types of ML: supervised learning and unsupervised learning. Supervised learning algorithms are trained on data that has been labeled with the correct answers. For example, a supervised learning algorithm could be trained to predict the price of a house based on its features, such as the number of bedrooms and bathrooms. Unsupervised learning algorithms are trained on data that has not been labeled. These algorithms can find patterns and structures in the data, such as clusters of similar data points.

The Tidyverse and mlr provide a powerful and consistent syntax for machine learning. The following steps provide a general overview of the ML workflow in R:

1. **Data preparation:** The first step is to prepare your data for ML. This includes cleaning the data, removing outliers, and transforming the data into a format that is suitable for ML algorithms. The Tidyverse provides a number of tools for data preparation, such as the `dplyr` and `tidyr` packages.

2. **Model training:** Once your data is prepared, you can start training ML models. The mlr package provides a number of functions for model training, such as the `train()` and `tune()` functions.
3. **Model evaluation:** Once you have trained a model, you need to evaluate its performance. The mlr package provides a number of functions for model evaluation, such as the `evaluate()` and `resample()` functions.
4. **Model deployment:** Once you have evaluated your model, you can deploy it to production. The mlr package provides a number of functions for model deployment, such as the `predict()` and `serialize()` functions.

Let's take a closer look at each of these steps.

Data preparation is an important part of the ML workflow. The Tidyverse provides a number of tools for data preparation, such as the `dplyr` and `tidyr` packages. These packages can be used to clean the data, remove outliers, and transform the data into a format that is suitable for ML algorithms.

The following code shows how to use the `dplyr` and `tidyr` packages to prepare data for ML:

Load the data

data % drop_na() %>% mutate(age = as.numeric(age))

Transform the data

data % mutate(gender = factor(gender))

This code reads a data set from a CSV file, drops any rows with missing values, converts the age column to a numeric data type, and converts the gender column to a factor variable.

Once your data is prepared, you can start training ML models. The mlr package provides a number of functions for model training, such as the `train()` and `tune()` functions. The `train()` function is used to train a model, and the `tune()` function is used to hyperparameter tune a model.

The following code shows how to use the `train()` function to train a linear regression model:

Create a linear regression model

model Model Evaluation

Once you have trained a model, you need to evaluate its performance. The mlr package provides a number of functions for model evaluation, such as the `evaluate()` and `resample()` functions. The `evaluate()` function is used to evaluate a model on a holdout data set, and the `resample()` function is used to evaluate a model using cross-validation.

The following code shows how to use the `evaluate()` function to evaluate a model:

```
# Evaluate the model evaluation Model Deployment Once you have evalu
```

Machine Learning with R, the tidyverse, and mlr

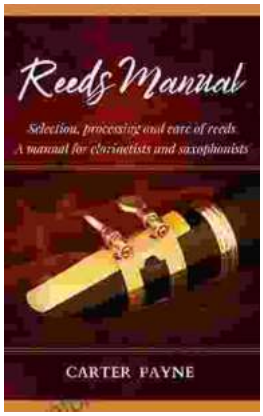
by Frank Kane

4.4 out of 5

Language	: English
File size	: 16223 KB
Text-to-Speech	: Enabled
Screen Reader	: Supported
Enhanced typesetting	: Enabled
Print length	: 536 pages

Unveiling the Urban Cheating Rich System: A Comprehensive Guide to Volume 1

In today's complex and ever-evolving urban landscape, cheating has become a rampant practice among the affluent elite. Fuelled by a desire for instant gratification, power,...



Selection, Processing, and Care of Reeds: A Comprehensive Manual for Clarinetists and Saxophonists

Reeds are essential components of clarinets and saxophones, and their quality and condition can significantly impact the instrument's sound and performance....