Statistic and Machine Learning Approaches for Marketing:

Individual project

Dilda Zhaksybek

March 2022

Contents

Introduction 3
Data preprocessing 3
k-Fold Cross-Validation 3
Feature Selection 4
Stepwise selection 4
Application 4
Machine learning algorithms 5
Linear Discriminant Analysis 5
Decision trees 6
K-Nearest Neighbors 6
Support Vector Machines 7
Random Forests 7
Evaluation of the models 8
References 10

Introduction

The aim of this project is to apply and explain 5 machine learning algorithms, as well as the full set of steps performed from data preprocessing to evaluation results. The experimental dataset is the data from a bank, where each observation is per client, and target variable represents whether client subscribed to the bank's services. Among the information retained per client, dataset include features such as age, marital status, education, housing along with other social and economic context attributes such as consumer price index or employment variation rate.

This project is a walk-through of a data science project from feature selection to explanation of the evaluation results, with the focus on explaining the machine learning algorithms applied in own words.

Data preprocessing

Before proceeding with any data preprocessing steps, dataset was randomly partitioned into train and test sets by 70% and 30% respectively.

As dataset contained missing values, both train and test were treated separately for them, but in the uniform manner.

For the variables "emp.var.rate" , "cons.price.idx","cons.conf.idx","euribor3m","nr.employed" – missing values were filled with the average values of the column.

For the variables "pdays" and "previous" – missing values were filled with 999 and 0 respectively, assuming there was no previous contacts, as most of the clients were not contacted.

For the variables "age" and "campaign" – missing values were dropped because the distance between min and max value are high.

All the categorical (character) variables were transferred into dummy variables using dummies package of R.

The dimensions of the train and test set are as following:

Train: (13713, 64)

Test: (5883, 63)

k-Fold Cross-Validation

To choose the optimal model among those computed by the stepwise selection method, test error can be estimated directly by applying k-fold cross-validation.

In k-Fold cross-validation, a dataset is divided into **k** sets, or folds. While one of the folds is held out as a validation test, the rest **k-1** folds are used to fit the chosen model. Mean Squared Error is computed on the validation test. The process is repeated k times, each time a random set of observations is treated as

a validation set. As at the end we have a k number of **MSE** values, k-fold cross-validation result value is then computed as the average of all **MSE** values. As in this project a few different machine learning algorithms are applied on a classification problem, our interest is in the minimum value of k-fold CV error rate across all models. Error rate is the number of misclassified observations.

Feature Selection

Stepwise selection

Not all the features are useful for the final model. To improve prediction accuracy and model predictability, several predictor selection methods are available. Among them is subset selection, which contains best subset selection procedure and stepwise model selection procedure. The later was used in this experimental setting.

The goal of stepwise selection is to identify a subset of predictors that are most relevant to the response, or prediction. We then fit the model on a subset of the variables only. Stepwise selection is computationally advantageous, especially for very large number of p predictors, comparing to best subset selection, as it does not consider every possible combination of variables ($\mathbf{2}^{\boldsymbol{p}}$ where p is the predictor). Stepwise selection can be either forward or backwards.

In forward stepwise selection begin with a model that contains no predictors at all, adding one predictor, that gives the best improvement, at each iteration. Thus, beginning at a Null Model (M0 – intercept), that contains no variables, **p-k** models are considered, and the model with a variable (set of variables) that yields the smallest **RSS**, or highest adjusted $\boldsymbol{R}^{\mathbf{2}}$, is added to the model. At next step (iteration), M1 model will again consider the best variable from **k-1** variables, among the **p-k** models. Stepwise selection considers $\boldsymbol{p}^{\mathbf{2}}$ possible models. So, at each step, every possible model that contains one more predictor that the step before is considered, giving a more restricted set of models to be considered. The process is also referred to as nested models. To choose among models, however, **RSS** or adjusted $\boldsymbol{R}^{\mathbf{2}}$ cannot be used, as models are of different sizes. In that case we use cross-validation to choose the model with the best set of p predictors.

In backwards stepwise selection, a model begins with full set of p predictors, or Full Model. At each step/iteration a least useful variable, in terms of its affect on **RSS** or $\boldsymbol{R}^{\mathbf{2}}$, is dropped. Then, the model that has the best cross-validated prediction error is selected. It is important to note that to apply backwards stepwise selection, the number of variables should be less than the number of observations.

Both models do not promise to yield a model with the best set of variables. But for large numbers of p predictors, it is preferred to best subset selection.

Application

In this experimental setting, backwards stepwise feature selection was applied.

To control computational nuances of the *train* function, first *trainControl* function of the CARET package was defined to perform 10-fold cross-validation resampling. The number of preferred features was set from 1 to the number of columns in the train set. To perform a backwards stepwise selection *method* parameter of the function was set to "leapBackward" – linear backward selection. The algorithm produces model with the best subset of variables for each of the number of variables. At the end, the model that has the lowest **RMSE** and highest $\boldsymbol{R^2}$ is the one that contains **21 variables.** The value of RMSE is 0.2857954 and adjusted $\boldsymbol{R^2}$ of 0.2014616.

Machine learning algorithms

5 machine learning algorithms were applied. Aim is to predict, whether a client will subscribe with the bank or not, on the basis of multiple predictors such as type of job, marital status, last contact with the client, number of contacts with the client and so on, as well as the predictors that affect socio-economic situation such as employment rate and consumer price index.

Linear Discriminant Analysis

Linear Discriminant Analysis, as the name suggest, is a linear model that is often used as a dimensionality reduction technique. It is a popular machine learning algorithm that uses Bayes theorem for classification, which proven efficient on multiclass problems.

LDA focuses on maximizing separability among the response categories. Using classes to create a new axis in a way that maximizes separation between categories. The new axis is created by maximizing the distance between two means of response classes/categories and minimizing the variation within each class. It is the squared difference between means of the classes divided by the sum of squared variances. Ideally the value large, meaning the distance between means of classes is large, ensuring great distance, separation between classes, while the variance within classes is small. Both separation distance between means and variance within classes matter. For example, if the focus was only directed towards maximizing the distance between means, the model would have a lot of overlap between classes, producing bad separation.

For dataset that have more than two classes, the approach changes a bit but not significantly. To measure the distances among the means, first a central point to all the observation needs to be found. After, the distance between that central point and the central points of each class is measured. Lastly, the goal is to maximize those distances, while minimizing the variance withing each class. Unlike classification problem with 2 classes, for 3 classes 2 axes will be created to make a separating plane. This is how LDA performs dimension reduction. For a big number of variables and classes, separating all the points and plotting the data in 2 or 3 dimensions.

LDA follows the assumptions that variables are Gaussian, meaning each predictor follows one-dimensional normal distribution, as well as that each data point has the same variance.

Decision trees

Decision trees are a supervised classification machine learning algorithm that splits the data at nodes according to a certain decision parameter into leaves. Decision tree can be pruned after it is "grown" in order to optimize the model and also remove the leaves that are irrelevant, insignificant or unnecessarily complexify the decision. One of the major advantages of the decision trees is their interpretability, however decision tree models yield high variance. The goal of the decision tree model is to predict class of the target variable using a set of simple rules (decisions).

Decision tree starts at the root node, which is the whole dataset that was not split yet. Each node represents a variable that is then split into two decision nodes depending on the decision boundary. Once the tree does not split any longer it arrives at the terminal leaf node. At each step, algorithm selects the split which will result in the most homogeneous resulting decision node or terminal leaf, but most importantly the split that leads to the greatest reduction of the RSS value. When a branch of the tree has an entropy of 0 – no further splitting is required. Entropy is the measure of randomness.

Unless a limit on splitting is set for a decision tree, it will keep splitting until there is a 100% accuracy on classification of the dataset. Thus, creating a model that is extremely overfit on the dataset. In order to prevent overfitting, pruning is used as one of the techniques. Simply put, some branches are cut from the bottom and up to improve the accuracy of the model on the validation set.

K-Nearest Neighbors

K-Nearest Neighbors, KNN, is a supervised machine learning algorithm that groups the classes of the variables based on their proximity/similarity. In other words, algorithm assumes that similar observations are in near proximity to each other. Same as decision trees, it can be used both for regression and classification, but is more often used for classification.

To begin, a user-specified number of k-neighbors is initialized. The distances between each observation of test data and each observation of training data is calculated. It can be calculated in various ways. One of the most popular methods of calculating a distance is Euclidean. After the distances are calculated nearest k-neighbors to the data point can be detected. In other words, of all the calculated distances, a class is defined by the number of k shortest distances. The most frequent label in the class is assigned as class label for classification problems. In case of regression problems, the class label is the average of the labels.

Choosing a value of K can be tricky. A value of K too low will lead to unstable predictions and overfitting the data. As the value of k increases the classification boundaries get smoother, but at the same time accuracy of classification is lost if the value of the k is too high. At k=1 the error rate would be 0 because any other closest value to the data point is the point itself. The optimal value of K can be found through cross-validation approach. KNN algorithm is also susceptible to dimensionality curse, meaning that it works best for the low number of predictor variables.

Support Vector Machines

Support Vector Machines is a linear model used both for classification and regression problems. SVM works in a way that it separates classes by creating a line or a hyperplane. SVM looks to find a separating the most generalized line/hyperplane. The way the algorithm performs that is by identifying ***support vectors***, the point closest to the line from both classes. After, the distance between those support vectors and the line is computed. The distance is referred to as a ***margin***. The goal of the model is to maximize the margin, creating the best, largest separation between the classes.

For the datasets that are not linearly separable, SVM projects datapoints on a higher dimension. In that case SVM would find a separating hyperplane in a higher-dimension, and then project it back to the original dimensions. Hyperplane would then be a n-1 dimensional subspace.

When tuning the SVM model a few things to consider would be a degree of tolerance, or penalty term, and kernel. As most of the models would probably not be linearly separable, a model must be assigned a ***penalty term***, specifying penalty for the misclassified variables. A higher value of the penalty term would apply more penalty to the misclassified data points, narrow the margin and depend on fewer support vectors when separating via hyperplane – hence trying to classify all the training points as correctly as possible. On the contrary, small values of penalty term would allow for more misclassification of the data points. Other thing to consider is a ***kernel trick***. It was stated earlier that for non-linearly separable datasets SVM projects datapoints on a higher dimension. But for large datasets this procedure would be extremely heavy in terms of computation. Thus, kernel trick is a way to produce datapoints as a set of coordinates on a higher dimension without actually applying the transformation. The most popular kernels are the polynomial and radial basis function kernels.

Random Forests

Random Forests is an ensemble algorithm of the decision trees, usually trained with the ***bagging*** method. Bagging is an ensemble method that creates different, random data subsets with replacement (individual observations can be chosen more than once – *bootstrap* sampling), generating several samples, on which a model is trained. A final output is then a combination of results based on the majority voting or averaging of those predictions.

Simply put, random forests combine multiple decision trees to make an improvement in accuracy and stability of prediction. Random forests, like decision trees, can be used both for classification and regression problems.

Unlike decision trees, which looks at the best feature to split upon, random forests splits on the best feature from a random subset of features. Thus, adding the randomness and diversity which generally results in a better model.

Evaluation of the models

Abovementioned algorithms were applied using CARET package. Once again, specifying *trainControl* to perform a 10-fold cross-validation when the models are fitted using the *train* function. The following models are specified in the **method** parameter: "lda", "rpart", "knn", svmRadial" and "rf".

Summarizing the accuracy of the 5 model the following result is printed:

```
Accuracy
          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda  0.8724490 0.8876937 0.8898614 0.8885007 0.8918673 0.8957726    0
cart 0.8826531 0.8936907 0.8967907 0.8950639 0.8982495 0.9008746    0
knn  0.8884840 0.8936907 0.8942378 0.8952089 0.8957536 0.9022611    0
svm  0.8927790 0.8937490 0.8964260 0.8964489 0.8986142 0.9022611    0
rf   0.8855685 0.8944779 0.8982495 0.8963764 0.8991612 0.9008023    0
```

The model that performs the best according to the accuracy score is the **support vector machines.**

Summary of the function yields:

```
Support Vector Machines with Radial Basis Function Kernel

13713 samples
   21 predictor
    2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 12342, 12342, 12342, 12342, 12341, 12342, ...
Resampling results across tuning parameters:

  C     Accuracy   Kappa
  0.25  0.8964489  0.2613638
  0.50  0.8961573  0.2686502
  1.00  0.8951365  0.2779010

Tuning parameter 'sigma' was held constant at a value of 0.04781915
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were sigma = 0.04781915 and C = 0.25.
```

At the end the penalty parameter used for the SVM model was set to 0.25. So it is on the smaller end, allowing for misclassifications but without hurting the accuracy score significantly. In fact, the model has accuracy score of ~0.9 on the training set, which is a good score at the end.

Confusion matrix:

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 5178  515
         1   58  132

               Accuracy : 0.9026
                 95% CI : (0.8947, 0.9101)
    No Information Rate : 0.89
    P-Value [Acc > NIR] : 0.0009327

                  Kappa : 0.2794

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9889
            Specificity : 0.2040
         Pos Pred Value : 0.9095
         Neg Pred Value : 0.6947
             Prevalence : 0.8900
         Detection Rate : 0.8802
   Detection Prevalence : 0.9677
      Balanced Accuracy : 0.5965

       'Positive' Class : 0
```

Now looking at the performance of the classification, our model performs with the 90% accuracy on the test set. It is important to mention that the model's **train and test accuracies are practically identical.** Meaning that our model does not overfit, and train and test sets are well distributed.

Sensitivity, or recall, is a score of how many of the positives were predicted correctly from all positives. The model correctly classified a client as "subscribed" in 99% of the cases.

However, our model is highly sensitive, without being specific. A specificity score of 0.2 says that a rate of false positives are quite high. So only around 20% of the client's who will not subscribe will be identified as non-subscribers by the model.

Let's discuss AUC value. AUC measures the ability of the model to distinguish between classes. SVM model yields AUC value of ~80%, meaning that in about 80% of the times the model distinguishes the class correctly.

To conclude, 10-fold cross validated support vector machine using Radial Basis Function kernel is highly sensitive towards identifying true positives, and not very specific identifying true negatives. At the same time, model performs uniformly between train and test sets, with the accuracy of ~0.9.

References

https://machinelearningmastery.com/machine-learning-in-r-step-by-step/

http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/154-stepwise-regression-essentials-in-r/

https://www.seldon.io/decision-trees-in-machine-learning

https://www.digitalvidya.com/blog/linear-discriminant-analysis/#:~:text=The%20linear%20Discriminant%20analysis%20estimates,Theorem%20to%20estimate%20the%20probabilities.

https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989#:~:text=SVM%20or%20Support%20Vector%20Machine,separates%20the%20data%20into%20classes.

https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f