Caret Versus Scikit-learn
A Comparison of Data Science Tools

Zhuoheng Xie, Zhenghao Ye, Simon Jones, Michael Roggenburg, Chris Root, Theerakorn Prasutchai, Matthew A. Lanham
Purdue University Krannert School of Management
xie176@purdue.edu, ye122@purdue.edu, jone1107@purdue.edu, mroggenb@purdue.edu, root2@purdue.edu, tprasutc@purdue.edu, lanhamm@purdue.edu

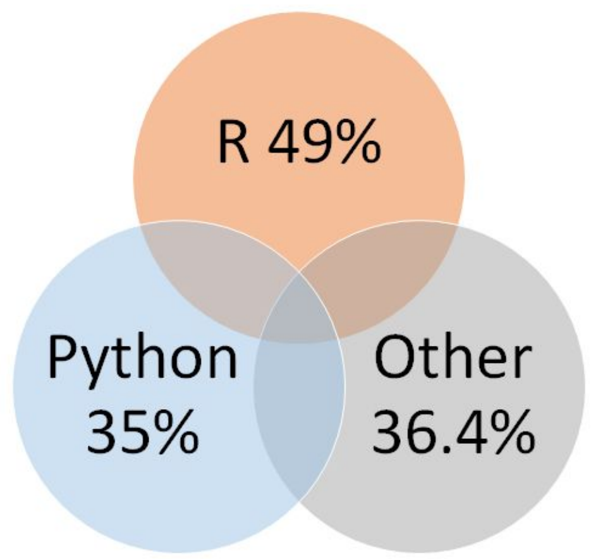
Abstract

The study focuses on comparing two popular packages in data mining and predictive analytics, caret in R and scikit-learn in Python. The criteria used in the comparison are accuracy, run time, as well as the limitations of the languages. The sample dataset was acquired from WSDM-KKBox's Churn Prediction Challenge from Kaggle competition. Dataset will be clean and run through ten different predictive models. These models will be trained separately using both caret and scikit-learn.

Introduction

Caret provides one of the most comprehensive wrappers for any set of R packages and can be solely used to define an entire workflow starting from data cleaning and preprocessing, all the way through model training, prediction, and performance analysis. Plus, it is free to use.

On the other hand, Scikit-learn provides the same functionalities in Python. It is also an open-source package which is free to use. Scikit-learn is designed for Data Mining and Machine Learning. Since Python is a widely-used language, it is more likely to be implemented in various applications.



R users, Python users, and R & Python users take account of 62% of the total number of users who perform Data Mining

Source: KDNuggets

We ran ten separate models using both R caret and Python scikit-learn, and described machine learning algorithms used in our study. In the Results section we show the comparison between R and Python on runtime and accuracy. Lastly, in the Conclusions we provide some key takeaways points for the reader to help them in their analytics journey.

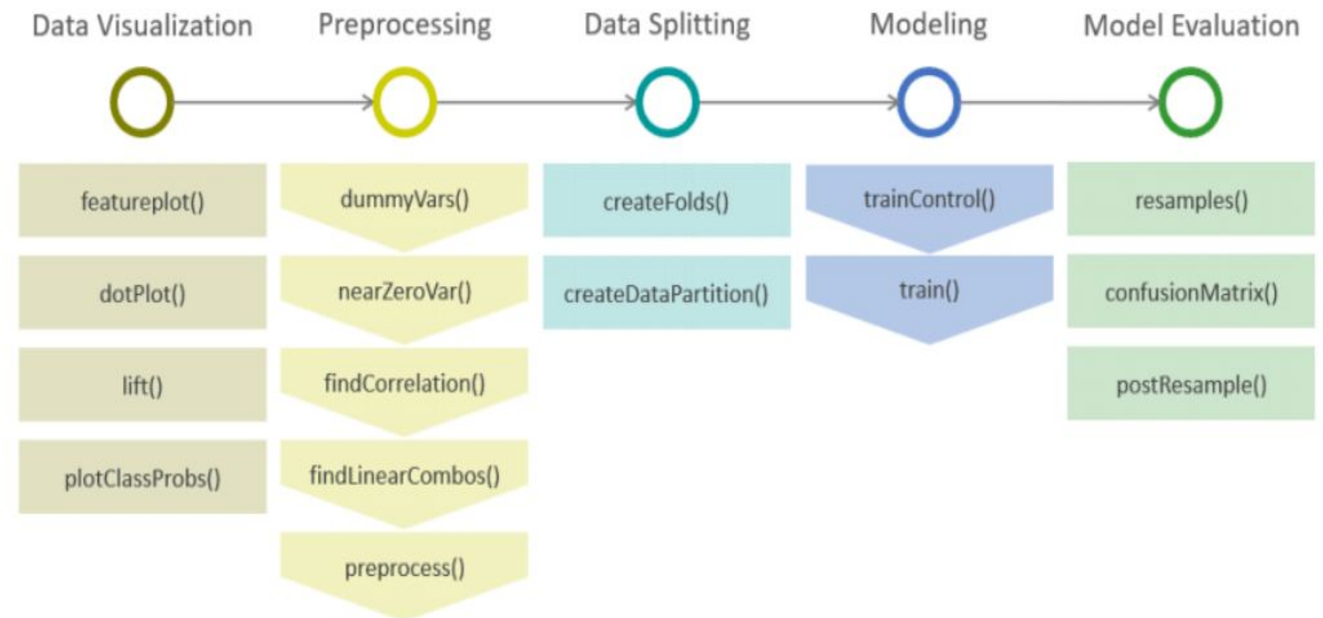
Literature Review

For new data scientists, the process of learning and applying these data mining methods can be daunting. For this reason, we conducted a literature review to understand what is known that has been published, and thus frame our proposed methodological workflow in this space.

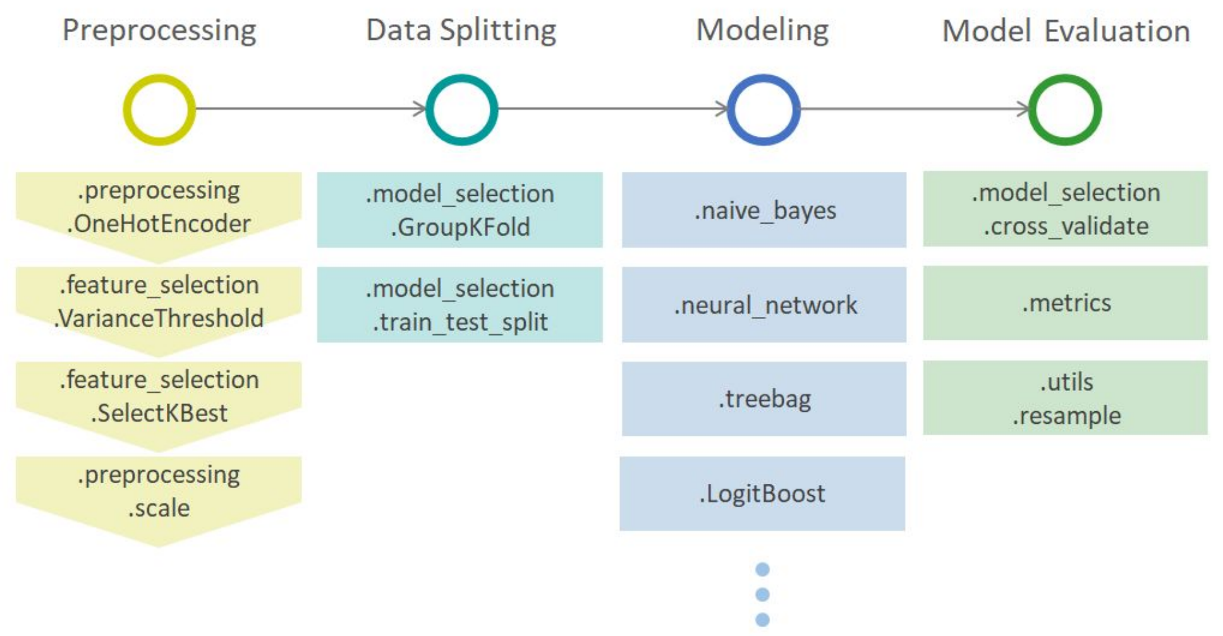
Author and Year	Python: Sci-kit Learn	R: Caret	Data Mining	Model Making	Performance Analysis
(Kuhn, 2008)		X		X	
(Batuwita, 2012)		X		X	
(Buitinck, et all 2013)	X		X		
(Fernandez-Delgado,et all 2014)		X		X	X
(Kuhn,2017)		X	X	X	X
(Scikit-learn Developers,2017)	X		X	X	X

We were more interested in understanding common workflows used in the studies, than the results of the studies themselves, and we use these studies as support for our proposed recommendation.

Methodology



Above, the data mining workflow using Caret functions is shown. Advantages of Caret are that it has Data Visualization functionality built-in and that it is more oriented to handle data mining tasks.


Preprocessing
Data Splitting
Modeling
Model Evaluation
.preprocessing .OneHotEncoder
.feature_selection .VarianceThreshold
.feature_selection .SelectKBest
.preprocessing .scale
.model_selection .GroupKFold
.model_selection .train_test_split
.naive_bayes
.neural_network
.treebag
.LogitBoost
.model_selection .cross_validate
.metrics
.utils .resample

Above, the data mining workflow using Scikit-learn functions is shown. Due to the way data is handled in Python, it is generally faster to train a model than caret with more robust data processing implementation, and potentially leading to better models.

Data
We used the data from WSDM-KKBox's Churn Prediction Challenge in Kaggle competition. There are 673,000 observations and 32 features including one that is an ID code. Out of these 32 features 16 are factors, 10 are numeric, and 6 are dates.

Preprocessing - If a record was missing any feature values it was dropped. Dummy variables were given to class features. Any features with a very low variance was dropped along with features that were highly correlated (90%) with each other. At the end, we centered and scaled our features with z-score standardization.

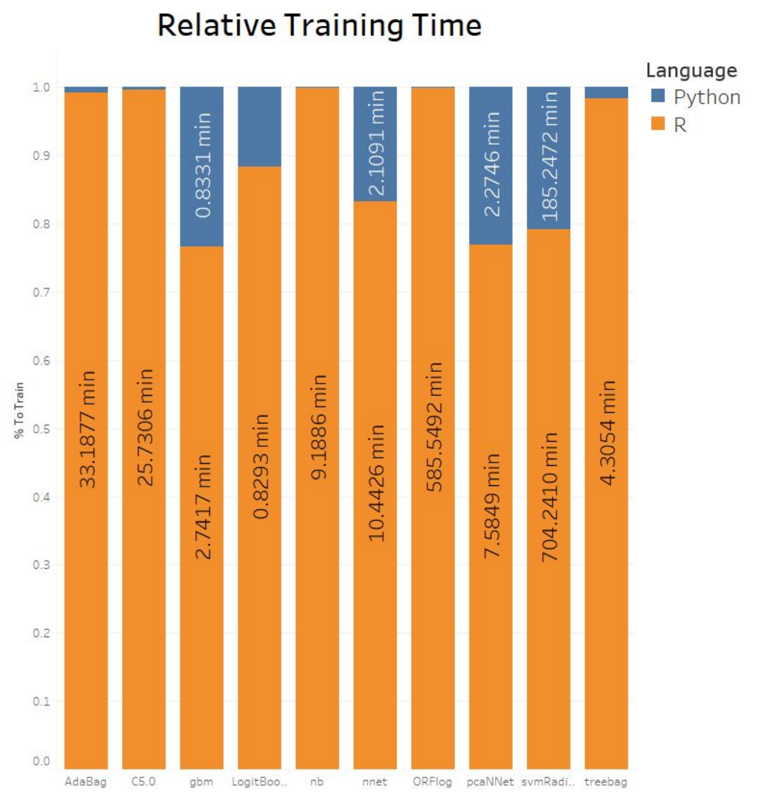
Data Splitting -The data was split 70-30 between the data set used for training the data and that used for testing the trained model, the split was based on a stratified random sample.

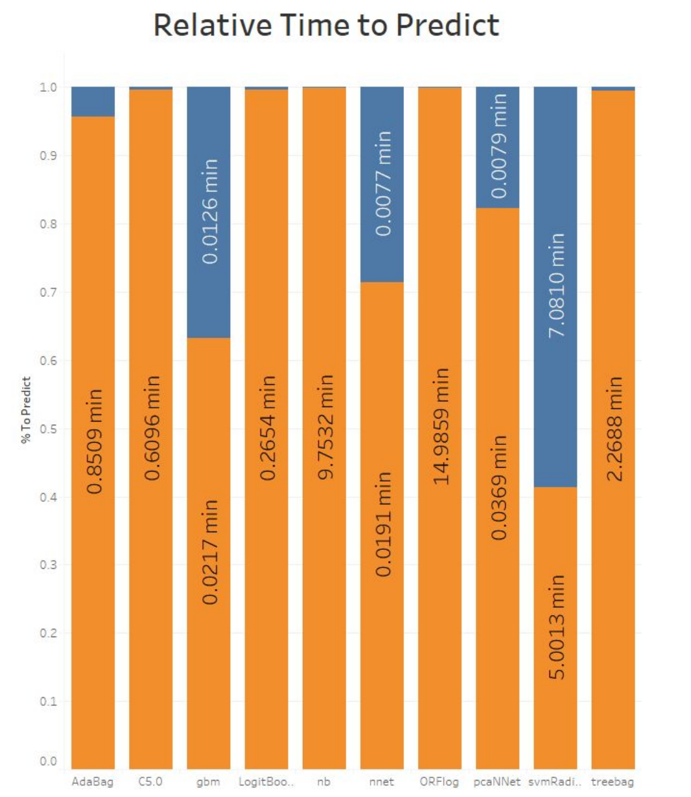
Modeling - The models we trained were Principal component Neural Network, Oblique Random Forest, Bagged Random Forest, Bagged ADAboost, Gradient Boosting Machine, Support Vector Machine w/ Radial Weights, Naive Bayes, Neural Net, Logistic Regression, and C5.0 Classification Tree.

Model Evaluation - We evaluated our models based primarily on accuracy.

Results

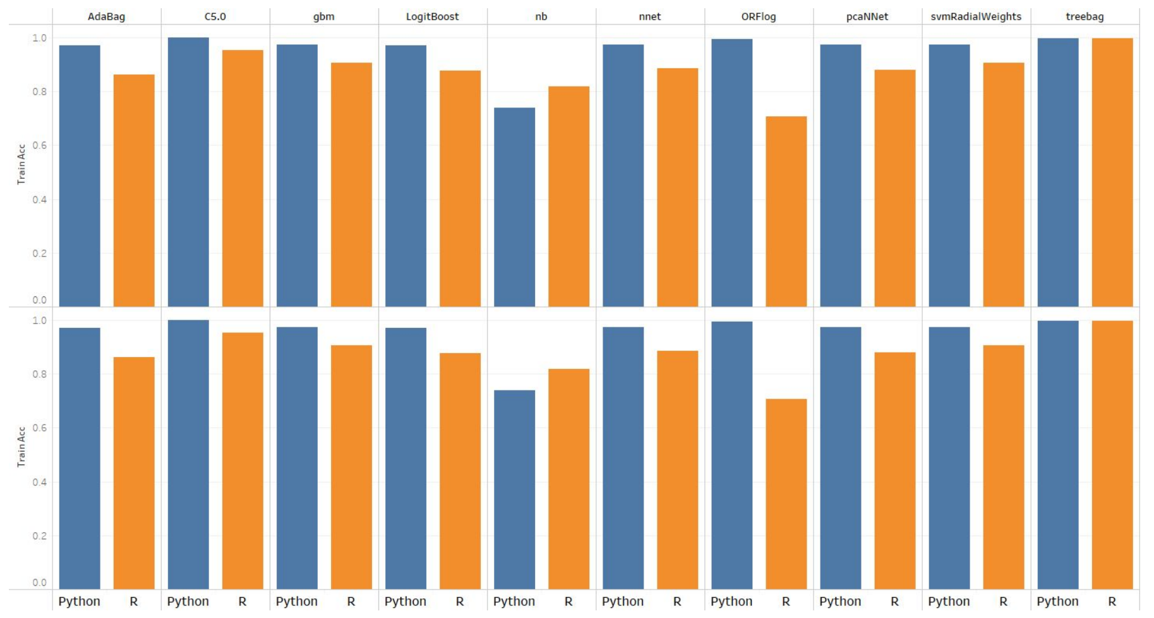
Runtime is one of the key factors to consider when choosing among a possible set of predictive solutions to support the business problem. Some detractors of R claim that it does not perform as well as other languages such as Python or SAS. Thus, we provide an idea of runtime for a large dataset such as the one this large company uses to understand if their customers will churn or not.





There is no concrete relation between runtime and accuracy of the predictive model. However, the more complex models will typically require more training and scoring time. Depending on the business problem at hand, a quicker, less accurate model or a slower, more accurate model may be preferred.

Accuracy Plot
If a model is well-trained we will expect to see the accuracy of the test set to be similar to the accuracy of the training set. The figure below shows the accuracy of each model on the train and test sets. Accuracy gives a percentage of the amount of overall correctly identified targeted variable.



What we do not want is to overfit the data to the training set, which means that the model we trained only gives us good results for the training set. If we put in the testing data into the overfitted model, we see that the results we get for the test set are not similar to the training set results.

Conclusions

When comparing caret with scikit-learn, the languages of the packages must be taken to account as well. Caret can perform various data mining functionalities easier and more in a more user-friendly way, thanks to the nature of the R language. On the other hand, scikit-learn trains models faster and sometimes more accurately due to how Python stores the data as matrices. Unlike R, there is no factor class of data in the Python which leads to some inconveniences for data mining process.

Acknowledgements

We thank Professor Matthew Lanham for constant guidance on this project.