DATA423 – Assignment 4

Q1

CARET Framework

Classification And REgression Training package streamlines model pre-processing and training steps for complex classification and regression problems. Currently, it is the best machine learning framework available in *R*. *Caret* assembles small pieces of codes with wrapper functions and assign a workflow which is available to nd user as high-level APIs. For example, the package performs cross-validation on all models, selects the best model, and fits on entire training data. (Semsch,2019). The framework also enables parallel processing options to effectively utilize all available computational power.

Pros

- Interface for modeling with more than 300 methods.
- Common pipeline for hyperparameter tuning.
- Facilities different data resampling and train/test split logics.
- Combined visualization for selected candidate model set.
- Improved speed and efficiency by using parallel processing.

Cons

- The computing and memory access is significantly slower for large size of data.
- Not suitable for *Deep learning, Image processing, Big data and, Audio/Video processing*.

Scikit-Learn Framework

Scikit-learn is a machine learning package initially developed by *David Cournapeau* for *Python* programming language. It is developed above some scientific computing and visualization libraries like *Numpy* and *SciPy*. It has efficient tools for performing classification, regression, and clustering algorithms. Since it is developed in *Python*, the package works well with popular data processing libraries like *Pandas*, *Matplotlib*, etc. It also supports *Natural Language Processing* algorithms text clustering, topic detection, sentiment analysis, document vectorization (Cesconi, 2019).

Pros

- Features included for data splitting and manipulations, performing statistics and metrics along with the ability for artificial data creation.
- High-level API to a large number of machine learning algorithms that do not require the user to call any more libraries.
- Consistent API across the models that should make the learning curve easier and well documented
- It works along with *Python* numerical and scientific libraries like *NumPy*, SciPy, Pandas, Keras, etc.

Cons

- It doesn't use hardware acceleration results in slow processing especially during the training models.
- The model API doesn't have much flexibility and some models such as *Random forest* are considered as a non-standard implementation due to the biasing of the feature importance.
- It is not the best choice for deep learning (Opala, 2018) but used internally for other deep learning libraries in *Python*.

mlr3

mlr3 is an improved rewritten version of *mlr* library both for *R* programming language with an object-oriented design using *R6* class system. Unlike other packages *mlr3* deeply focused on the computation part and doesn't come with visualization or related stuff (Schratz, 2019).

Pros

- Overcome the limitations of R's *S3* classes with the help of *R6*.
- Fast and convenient data frame computations.
- Safety type and defensive programming. All user input is checked with *checkmate*.
- Supports out-of-memory data-backends like databases.
- Objects are queryable for properties and capabilities and allow to program on them.

Cons

- Limited to basic building blocks for machine learning.
- Relatively new package with less user base.
- The reimplementation phase is still not completed.

Caret vs Scikit-Learn

Based on performance comparison in 2018 at the *Purdue University Krannert School of Management*, *Sckit-Learn* performs faster with a good amount of accuracy over *R's Caret* library. This is due to the internal implementations of data storage as matrices in *Python*. On the other side, *Caret* is more user friendly and easy to work with. *Caret* also facilitates great visualization platform. One drawback of *Scikit-Learn* is that it lacks the feature of *factor* datatype which is handy to deal with categorical data.

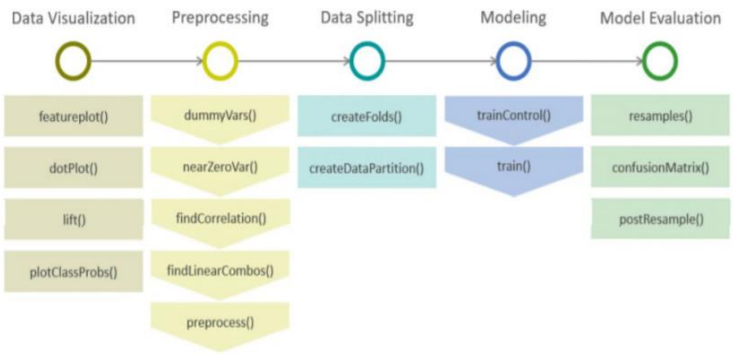

Fig. 1. Data mining flowchart for Caret

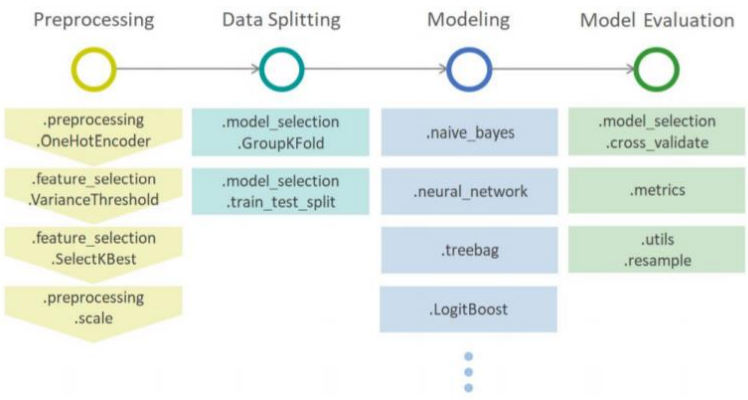

Fig. 2. Data mining flowchart for Scikit-Learn

Scikit-Learn has an upper hand over *Caret* on some areas like *Image processing, Deep learning, Big data, and Audio/Video processing* where computational complexity and large data processing matters. It's better when used with high-level libraries like *Keras* and *Tensorflow*. *Caret* provides more statistical features of models like p-values, standard errors, and variance inflation factors by default. *Caret* posses some inbuilt models which are commonly used but for others, the user has to manually install dependent packages while *Scikit-Learn* is itself a container for all available models (Satyabrata Pal, 2020).

Caret vs mlr3

Caret and *mlr3* are machine learning libraries for *R* serving the same purpose. Their difference lies in the implementation side, Caret uses *S3* class system while later uses *R6* class system. So, *mlr3* provides *R6* object for tasks learners, resampling, and measures.

The *R6* system in the *mlr3* supports an elegant way for object state-changes and reference semantics. This means that we don't need an assignment and we need to just call the function on the object. It will automatically update the object with the new content that the function creates (Berel,2019) and replaces the old traditional assignment which seems redundant, cluttered, and less elegant. Moreover, *mlr3* uses *data.table* for fast pre-processing and other manipulations and perform better performance than *Caret*.

A detailed comparison based on features are given below,

- **Pre-processing**
  Both packages have several methods for pre-processing of the dataset like a dummy, imputation, normalization, etc. The advantage of *mlr3* over caret is, all pre-processing methods of *Caret* are accessible in *mlr3* via wrapper methods (Probst, 2018).
- **Tasks**
  mlr3 is better than *Caret* in the creation of tasks. *mlr3* has more choices to create tasks: *Survival, Multilabel, Cost-sensitive, Classification, Regression, Imbalance Data, Spatial Data*. But *Caret* only has choices for *Classification, Regression and Cost-sensitive* (Probst, 2018).

- **Learners**
  Caret supports 271 learners and *mlr3* has 165 learners. It is easier to look up the learners in *mlr3* and we can access using *mlr3learners* package.
- **Measures**
  mlr3 has around 70 measures to evaluate the performance of the model and only one measure is given as default for each task(MSE in Regression, MMCE in Classification). Besides, custom measures can be added by the users. *Caret* has only two standard measures for each task (RMSE and R-squared for regression, accuracy, and Kappa for classification). We can change the measures using *trainControl* argument but it is more complicated than we do it in *mlr3* (Probst, 2018).
- **Benchmarking**
  mlr3 has functions for executing a benchmark of several tasks/datasets and several learners at once(parallelism also). No such functionality is available in *Caret* (Probst, 2018).
- **Parallelization**
  In *Caret* package, the parallelization only works internally when we applying a tuning strategy like a grid search using *doParallel* package (Probst, 2018). mlr3 is better than *Caret* in the case of parallelization. It has more options and we can parallelize on different levels: while doing resampling, tuning, training an ensemble model and selecting features.

Caret vs mlr3 vs Sckit-Learn

Comparison between various features of considered packages,

	Caret	mlr3	scikit-Learn
Language	R	R	Python
feature Selection	[model]+recipes	pkg:mlr3featsel	sklearn.feature_selection
pre-processing	✓	✓	✓
hyper-parameter tuning	grid search	SA	GridSearchCV,RandomizedSearchCV
ensemble/stacking	pkg:caretEnsemble	✓	sklearn.ensemble
parallelism	✓	✓	✓
custom optimized metric/loss functio	✓	✓	✓
extensibility	models	✓	models
feature engineering while learning	✓	✓	✓
resampling strategies	✓	✓	×
cost-sensitive learning	✓	✓	✓
models	>300	few	100 to 200
noticable additional packages	caretEnsemble	mlr3db	scikit-image
Model performance Evaluation	postResample,twoClassSummary, lift, confusionMatrix	performance, calculateROCMeasures	Estimator score method,Metric functions, Scoring parameter
class	S3	R6	Provides R6 with the help of mlapi package
Learning Curve	5	3	
Functionalities	4	5	
Modular API	✓	✓	✓
GUI	×	×	×
Model Validation and Comparison	×	✓	✓

Table. 1. Feature Comparison – Caret, mlr3, Sckit-Learn (obtained from references 11,12])

All three frameworks support most of the basic functionalities like feature selection, pre-processing, hyper-parameter tuning, resampling, etc. When we consider the number of models in each framework, *Caret* has the highest number of models (greater than 300) and *mlr3* supports the least number of models. Moreover, all these 3 frameworks have an extensive API library, but not support GUI. All frameworks are good, but their usage depends on the problem that we are trying to solve. *Caret* is good at solving complicated supervised and unsupervised in timely manner problems (Mousa, 2018). However, if we want to do image processing or need to integrate with a web application then *Scikit-Learn* becomes a better option than *Caret*. When comes to *mlr3*, it is good in core computational operations in R.

In conclusion, *Caret*, *Scikit-Learn*, and *mlr3* have their strength and weakness when different areas. According to the problem domain and choice of programming language, a framework should be chosen.

References

1. Semsch, K. (2019, August 6). Caret vs. tidymodels - comparing the old and new. Retrieved from https://konradsemsch.netlify.app/2019/08/caret-vs-tidymodels-comparing-the-old-and-new/

2. Cesconi, F. (2019, October 29). What are the advantages and disadvantages of using TensorFlow over Scikit-learn for unsupervised learning? Retrieved from https://www.linkedin.com/pulse/what-advantages-disadvantages-using-tensorflow-over-learning-cesconi/

3. Satyabrata Pal. (2020, April 21). Scikit-learn Tutorial: Machine Learning in Python. Retrieved from https://www.dataquest.io/blog/sci-kit-learn-tutorial/

4. Schratz, P. (2019, July 31). mlr3-0.1.0. Retrieved from https://mlr-org.com/docs/mlr3-0-1-0/

5. Berel, D. (2019, December 18). Meta Machine Learning aggregator packages in R: Round II. Retrieved from https://medium.com/analytics-vidhya/meta-machine-learning-aggregator-packages-in-r-round-ii-71ee1ff68642

6. Probst, P. (2018, November 9). mlr vs. caret. Retrieved from http://philipppro.github.io/2018-11-9-mlr_vs_caret/

7. Opala, M. (2018, September 5). Top Machine Learning Frameworks Compared: Scikit-Learn, Dlib, MLib, Tensorflow, and More. Retrieved from https://www.netguru.com/blog/top-machine-learning-frameworks-compared

8. Purdue University Krannert School of Management. (2018). Caret Versus Scikit-learn A Comparison of Data Science Tools. Retrieved from http://matthewalanham.com/Students/2018_PURC_caretvsscikit.pdf

9. Mousa, Saleh. (2018). Re: Which is the best tool, R or Phyton for Machine Learning? Retrieved from: https://www.researchgate.net/post/Which_is_the_best_tool_R_or_Phyton_for_Machine_Learning

10. Brownlee, Jason. (2019, August). A Gentle Introduction to Scikit-Learn: A Python Machine Learning Library. Retrieved from https://machinelearningmastery.com/a-gentle-introduction-to-scikit-learn-a-python-machine-learning-library/

11. WeLoveDataScience. (2019, July). Machine Learning do it with a framework. Retrieved from http://www.welovedatascience.com/user2019/#1

12. Farber, Rob. (2017, January). A Case for CPU-Only Approaches to HPC, Analytics, Machine Learning. Retrieved from https://www.nextplatform.com/2017/01/27/case-cpu-approaches-hpc-analytics-machine-learning/

Q2

Issues observed in the data science project for ***MyStyle® Clothing Corporation***,

1. Demographics and cultural differences between 12 different countries are not considered. This may yield undesired model performance.
2. Observations/Variables ratio was not discussed. The ratio should be greater than 10 for a strong dataset.
3. There is no information regarding the size of shops or the range of employees working. Hence, the median summary statistic will be more efficient than mean.
4. Decisions based on the classification of employees based on Nationalities, Sex, Religion, Physical appearance are not welcomed in many aspects.
5. The employees were not aware of monitoring as well as some data was extracted from personal files. This is unethical and illegal.
6. The variable *Start_time* contains shift start time on local time which needs to be converted into UTC or any suitable format since we are considering data from different time zones. Also, it might be a good idea to make it cyclic.
7. There is a data collection error for column *Number sick days* and *Age*. While all other data is for the year 2018, sick days entries are for 2019, and age entries are for 2020.
8. The nominal columns *Country*, *Branch* suffers from high cardinality hence it must be removed from consideration or reduce cardinality or use suitable encoding.
9. There is no mention about class imbalance in the categorical variables.
10. For other numeric variables, their distribution and range aren't available in the report. Need for correlation and Near Zero Variance check need to be done to familiar with the quality of outliers.
11. 4% present of data is found to be missing. The spread of missingness for the predictors should be visualized by a plot and different imputation strategies like *bag-impute, knn-impute, median-impute, mean-impute,* etc must be tried and best way should be chosen. Possible reasons for missingness must be investigated.
12. *Step 2* is missing from the report.
13. Before visualizing boxplots, density plots are advised to check the distribution and check if standardization by *centering* and *scaling* as well as normalization using Yeo Johnson transforms, etc are necessary before outlier detection. Classification of statistical and systematic outliers should be done.
14. Considered IQR multiplier range is not reported. Also, no mention of model-based and multivariate outliers.
15. For the newly added variable *BodyMassIndex* formula used is wrong. The correct equation for *BMI* is *Weight/Height*.
16. On adding variables *BodyMassIndex* and Net_*productivity*, colinearity between columns *Weight, Height, Sales_value, Returned_sales_value* and *shift length* must be checked and necessary actions to be done. *Weight* and *Height* may be removed along with *Sales_value* and *Return_sales_value*.
17. In *Company+branch+employee_id,* variable *Company* is not defined, and it may be a typo of *Country*.

18. To avoid data leakage pre-processing and imputation have to be performed after the train-test data split.
19. Instead of random splitting, a stratified split may give better performance.
20. On method selection, more diversity is expected. Few methods from categories Neural Networks, OLS, Tree-based, Kernel, and Robust methods are required for good estimations and try further methods like ones that perform well. Also, NULL performance to be taken into account.
21. Resampling and cross-validation on training set has to be considered to get optimum values for hyperparameters (from distribution of metrics) for each method. Random setting of parameters isn't welcomed.
22. A neural network is an opaque and expensive model and other transparent models with similar performance should be given preference.
23. There were no references to external sources to justify whether the Neural Network model is best for the purpose by the given criterion.
24. A visualization of all model performance is expected to justify the best model selection.
25. The best way to select model is through its performance in resampled data and not by performance in test data. Test results are for assessing the unseen performance of best model.
26. The report is not well structured in terms of tabular representations, lack of proper visualizations, language usage, and document formatting.