Assignment: Design and Application of a Machine Learning System for a Practical Problem
Set by: Prof. Luca Citi (lciti@essex.ac.uk)
Word Count: 972
Report (Task 4)

This report describes the way that different types of algorithms are used to address the mentioned problem in parts 2&3 and tries to provide the reason or set of reasons for the selection of a specific model as the best model for each part.

- As it has been mentioned in the pilot study, part 2 problem is a classification problem, and to tackle these kinds of problems, we need to apply classification algorithms. To avoid the overfitting issue, starting from simpler models and then using more sophisticated (if it is applicable), makes us more confident about the lower chance of overfitting, and lower computation cost in some cases.[1] at this part of the project, it has been tried to use Double Cross-Validation (a.k.a Nested Cross-Validation) method both to tune hyperparameters and evaluate the performance of the model in an efficient manner in terms of model performance (less bias) [2] and possibly computational cost. In order to handle missing values, I used an interpolate method which has been implemented in pandas library[10] to do imputation, and for the remaining null values (3 remaining), I replaced them with the mean of their column.

- Based on the suggestion in the assignment file, the first applied algorithm is a Decision-Tree Classifier. In short, Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. This method builds a model based on simple rules inferred from attributes of a dataset and then predicts the value of the target variable.[3] In this case, a procedure is required that allows models firstly, to select well-tuned hyperparameters for the dataset and secondly, select among a set of high-performance models on a dataset.[4] Hence, I used Nested Cross-Validation Method by using K-fold [5] with K = 10, shuffled, and random_state = 1 to train the model and using K-fold with K = 3, shuffled, and random_state = 1, to tune the hyperparameters of a model throughout every training process in the classification part to meet the mentioned demand. By doing so and having the following dictionary of the parameters to plug into GridSearchCV module in the case of a decision tree with fixed random_state=1, the output model would be a decision tree classifier with {max_depth = 5, criterion = 'gini', random_state = 1}. The figure of the tree as an outcome could be accessed via this link.

```
paras = {"max_depth": np.arange(1,10),
"criterion":np.array(["gini","entropy"]),
"splitter": np.array(["best", "random"])}
```

Accuracy Mean	Accuracy Std
0.859	0.025

Table1. Mean and Std of Accuracy metric for the selected decision tree classifier model during 10 iterations of K-fold cross-validation (K=10).

TN = 40	FP = 14
FN = 2	TP = 44

Table2. Confusion Matrix of the selected decision tree classifier model using scikit-learn notation for the last iteration.[6]

- Keeping the same evaluation method, I used Logistic Regression which is a statistics-based model that in its simple form uses a logistic function to model a binary dependent variable.[7] I used the default logistic regression model as a baseline to build the desired classifier. In this case, the parameter dictionary used for GridSearchCV module is as follows:

```
paras = {"penalty" : ['l1', 'l2', 'elasticnet','none'],
"C" : np.logspace(-4, 4, 20),
"solver" : ['lbfgs', 'liblinear', 'sag', 'saga']}
```

After training the model and tuning the mentioned parameters using Nested-Cross Validation process mentioned in the previous section, the outcome provides the following results:

Accuracy Mean	Accuracy Std
0.719	0.040

Table3. Mean and Std of Accuracy metric for the selected Logistic Regression classifier during 10 iterations of K-fold cross-validation (K=10).

TN = 45	FP = 9
FN = 19	TP = 27

Table4. Confusion Matrix of the selected logistic regression classifier model using scikit-learn notation for the last iteration.

- Keeping the same strategy and applying it on a Linear SVM as a baseline which is the algorithm that defines a line or a hyperplane which separates the data into classes[8], and the following dictionary as its parameter grid to that baseline, leads to the following result.

```
paras = {"C": np.logspace(-1, 3, 5),
"gamma": np.logspace(-7, -0, 2)}
```

Accuracy Mean	Accuracy Std
0.707	0.048

Table5. Mean and Std of Accuracy metric for the selected Linear SVM classifier during 10 iterations of K-fold cross-validation (K=10).

TN = 42	FP = 12
FN = 16	TP = 30

Table6. Confusion Matrix of the selected Linear SVM classifier model using scikit-learn notation for the last iteration.

- As the last algorithm on the classification part, trying Gaussian Naive Bayes, which is a statistical machine learning algorithm that assumes conditional independence of attributes with respect to class[9], was an option. Applying the same process, passing the var_smooth parameter through GridSearchCV as follows results in a mediocre model.

```
pars = {"var_smoothing": np.linspace(0,100, 200)}
```

Accuracy Mean	Accuracy Std
0.547	0.081

Table7. Mean and Std of Accuracy metric for the selected Naive Bayes classifier during 10 iterations of K-fold cross-validation (K=10).

TN = 0	FP = 54
FN = 0	TP = 46

Table8. Confusion Matrix of the selected Naive classifier model using scikit-learn notation for the last iteration.

- Before jumping to a conclusion, the discussed result was based on the imputed dataframe. I repeated the mentioned process on the dataframe with dropped null values, and the following table shows the mean accuracy and std of accuracy for each of the discussed models(except for Gaussian Naive Bayes) on the null-dropped dataframe.

Model	Mean Accuracy	Accuracy Std
Decision Tree	0.780	0.068
Linear SVM	0.754	0.070
Logistic Regression	0.780	0.043

Table9. Mean and Std of accuracy metric for trained models on the null-dropped dataframe.

  - Based on the information provided in the tables, we can conclude that Decision-Tree Classifier could be a good choice since it shows an acceptable performance especially on the imputed dataframe in comparison with other algorithms used on this dataset. As a result, I used the algorithm with its tuned parameter to produce the result of part B.

- The third part of the assignment is regarding calculating the amount of profit using Regression methods. To improve the R^2 score and then Root Mean Squared Error, I used QuantileTransformer module in sklearn.preprocessing, which transforms data features to follow the normal distribution, and using the transformed data in training process would be helpful to achieve better performance.[11]

- For the first algorithm, I used default linear regression model in sklearn and after transforming the data, and finishing training process, the outputted model performance is as follows:

R^2 Mean	RMSE Mean	R^2 Std	RMSE Mean
0.694	667.860	0.050	43.658

Table10. Mean and Std of RMSE and R^2 for the linear regression model.

- For the second algorithm, I used the default random forest regressor model in sklearn, which is an ensemble learning method that constructs a number of decision trees during training time.[12] After transforming the data using QuantileTransformer, and then carrying out training process, the outputted model performance is as follows:

R^2 Mean	RMSE Mean	R^2 Std	RMSE Mean
0.648	720.565	0.032	54.857

Table11. Mean and Std of RMSE and R^2 for the random forest model.

- For the third algorithm, I used Gradient Boosting technique using LightGBM, which is an ensemble algorithm that fits boosted decision trees by minimizing an error gradient[13], to build a regressor, and after transforming the data using QuantileTransformer, and then finishing the training process, the model performance is as follows:

R^2 Mean	RMSE Mean	R^2 Std	RMSE Mean
0.870	437.052	0.018	38.325

Table12. Mean and Std of RMSE and R^2 for LGBM regressor.

- For the fourth and the last algorithm, I used the default ridge model, which is a method of approximating the coefficients of multiple-regression models in problems where independent variables are highly correlated[14], to build a regressor, and after transforming the data using QuantileTransformer, and then finishing the training process, the model performance is as follows:

R^2 Mean	RMSE Mean	R^2 Std	RMSE Mean
0.694	667.863	0.049	43.570

Table12. Mean and Std of RMSE and R^2 for the ridge model.

- Based on the information provided in the tables, we can conclude that the LGBM regressor could be a good choice since it shows an acceptable performance during model

evaluation in terms of metrics like RMSE, and R^2. As a result, I used this method to produce the result for the next part of the assignment.

References

1. Tayo, B. O. (2021, December 12). Simplicity vs Complexity in Machine Learning — Finding the Right Balance. Medium. https://towardsdatascience.com/simplicity-vs-complexity-in-machine-learning-finding-the-right-balance-c9000d1726fb

2. Cawley, G. C., & Talbot, N. L. C. (2010). On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. Journal of Machine Learning Research, 11(70), 2079–2107. Opgehaal van http://jmlr.org/papers/v11/cawley10a.html

3. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

4. Brownlee, J. (2021, November 19). Nested Cross-Validation for Machine Learning with Python. Machine Learning Mastery. https://machinelearningmastery.com/nested-cross-validation-for-machine-learning-with-python/

5. Brownlee, J. (2020, August 2). A Gentle Introduction to k-fold Cross-Validation. Machine Learning Mastery. https://machinelearningmastery.com/k-fold-cross-validation/

6. sklearn.metrics.confusion_matrix. (n.d.). Scikit-Learn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

7. Tolles, J., & Meurer, W. J. (2016). Logistic Regression. JAMA, 316(5), 533. https://doi.org/10.1001/jama.2016.7653

8. Pupale, R. (2019, February 11). Support Vector Machines(SVM) — An Overview - Towards Data Science. Medium. https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989#:%7E:text=SVM%20or%20Support%20Vector%20Machine,separates%20the%20data%20into%20classes.

9. Sklearn Naive Bayes Classifier Python: Gaussian Naive Bayes Scikit-Learn Tutorial. (2018). DataCamp Community. https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn

10. GeeksforGeeks. (2018, November 19). Python | Pandas dataframe.interpolate(). https://www.geeksforgeeks.org/python-pandas-dataframe-interpolate/

11. sklearn.preprocessing.QuantileTransformer. (n.d.). Scikit-Learn. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html

12. Random decision forests. (1995). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/598994/

13. Brownlee, J. (2021, April 26). Gradient Boosting with Scikit-Learn, XGBoost, LightGBM, and CatBoost. Machine Learning Mastery. https://machinelearningmastery.com/gradient-boosting-with-scikit-learn-xgboost-lightgbm-and-catboost/

14. Ridge, a computer program for calculating ridge regression estimates. (1977). Google Books. https://books.google.co.uk/books?id=R4HSmWSNWEkC&pg=PP4&redir_esc=y#v=onepage&q&f=false