Final Assignment

JBI030: Data Mining 2019-2020

This is the final assignment of Data Mining. For this assignment, you are supposed to put together what you have learned during the course. This assignment is composed of two parts: a practical part, in which you will analyse a dataset trying to achieve a specific goal with the tools you have learned during the course, and a theoretical part, in which you will be asked some questions about the methodologies you have learned. As usual, you are going to work with your Assignment group.

You should write your answers on a Jupyter notebook. For the practical part, you must describe with a small report the steps you undertake for the analysis (data exploration, preprocessing, model selection, and so on). This report should be accompanied with the code you implement for the analysis, so that we can reproduce your results (**Don't forget to set the random seeds and random states of the scikit-learn estimators**). As explained in the next section, the analysis of the dataset consists of two parts: one in which you try and compare the performance of different models and preprocessing methods, and another one in which you test your results on a test set. For the theoretical questions, you can answer them in the Jupyter notebook, too. When you are ready for the submission, upload your notebook on Canvas, in the Assignment page (assignment: Final Assignment).

The total achievable score for this assignment is 10 points. The practical part consists of 6 maximum points; the theoretical part of 4 maximum points. For the practical part, how your work is evaluated is explained in the next section.

The deadline of this assignment is Thursday, April 28 at h.23.59. (Note: the true deadline is supposed to be April 19th, but we decided to extend it to the 28th, to give you some more time to work on the assignment and other exams.)

1 Part 1: Analysis of a Dataset

Files. The 9th of April, in the "File" section of the Canvas course page, you will find the file "churn_train.csv". In the first part of this assignment, you are going to work with this dataset, in order to find the best settings (preprocessing steps, models, hyperparameters) for the analysis goal (described in the sub-section "Dataset Description and Analysis Goal" below). In the second part of the assignment, a test dataset "churn_test.csv" will be released (see the sub-section "Plan of the Assignment" below); you will implement on the test data the two or three best models you found with the training dataset, and pick the one that performs best on the test dataset. All your steps must be carefully described, and the code you use to implement them must also be present in the Jupyter notebook.

Dataset Description and Goal. In marketing, the *churn rate* (or *customer churn*, or *customer attrition*) describes the proportion of customers, users, players, or subscribers that quit their business/subscription or, more in general, end the relationship with a company. In the file "churn_train.csv" you can find the Churn dataset, which collects data of a telephone company. The dataset relates the characteristics of telephony account (clients) features, their usage and whether or not the customer churned. In particular, the training dataset is composed of $n = 4000$ observations, and 20 features (described in Appendix A below). The target is a binary variable, denoting whether the customer left the company (class 1), or not (class 0). For the phone company, the characteristics of the accounts that churned are of particular interest, in order to prevent more clients from churning in the future. Therefore, for this problem Class 1 is the positive class.

Your goal for this challenge is to find a way to predict the class, in such a way that the AUC score is maximized. In the report, you are expected to frame the problem, as well as to give a short explanation of what the AUC score is, and how to interpret it.

Your Steps and Possible Challenges. As explained in the section "Grading" below, your work is not only going to be evaluated on the best AUC score you find on the test dataset, but also - and most importantly - on your understanding of the problem, and whether the steps you take for the analysis are sound and correct. Among other things, you will be evaluated also on the following aspects:

- dataset exploration
- choice of the preprocessing methods
- choice of the model selection method
- model comparison and selection
- plots (ROC curve, correlation matrix plots, ...)

Among the steps just listed above, you are expected to describe the ones you decide to undertake to achieve the final goal. You are also required to report possible challenges you encounter while exploring and mining the dataset. Suggestions of possible topics of discussion and steps to pursue in your project are:

- Problem framing: is this a supervised/unsupervised learning problem? What type of task is required?
- Is any ID feature present in the data? If so, how do you decide to treat it?
- Are 0-variance (or near-0 variance) features present in the dataset? If so, how do you treat them?
- Explore the data. What is the correlation structure among the features? What are the features that correlate the most with the class?
- Are missing values present in the dataset? If so, how do you behave w.r.t. this issue? (Note: **always** assume that missing values might be present also on the test dataset)
- Are the features of only one type (e.g., continuous), or of different types? If so, do the different set of features require different types of preprocessing tools?
- Do continuous feature need to be scaled? If so, what approach do you use?
- If there are categorical features in the data, how are you going to encode them?
- Is this an imbalance problem?
- Did you use any feature selection/dimensionality reduction technique? If so, which one? Which features where selected/discarded?
- What model selection method do you use to compare models and pipelines?
- What is your final best performing model on the test set? What AUC score does it lead to?
- Report the cross-validated ROC curve of the best two/three estimators, as well as on the test set (for the best estimator). Is a model uniformly better (=across all operating points) than others, or different models are better at different operating points?

Last, in the end of the report you are expected also to interpret the results; for example, if your final (best) estimator allows for interpretation/feature importance, what are the most relevant features for the prediction of churn?

Try to give a structure to your report. For example, an idea of possible structure is a division in the following sections: (1) Problem Framing; (2) Data Exploration; (3) Data Preprocessing and Model Tuning; (4) Results (here, you can also report the Precision/Recall curves); (5) Application to the test dataset.

Plan of the Assignment.

During the training stage, you should focus on finding the two or three best combinations of data preprocessing + algorithms (for example, which ones lead to the highest *AUC*

Release	Description	Phase	Deadline
07.04.2020	Publication of pdf file	/	/
09.04.2020	Upload training dataset	Training	
17.04.2020	Upload test dataset	Testing	28.04.2020 (h23.59)

score) among the ones you choose to implement. During the testing stage, you will apply these methods on the test data, and pick a winner. Notice that you will still have some time to work on your best model once the test data is released. However, you MUST NOT use the test dataset to tune your models (whether you do this will be clear from the code you show on your report).

Grading. The practical part of this assignment will be graded on a scale from 0 to 60; this final score will be dived by 10, and rounded to the nearest integer. The scoring system works as follows:

- Problem framing and understanding: max. 5 points
- Data exploration and description: max. 10 points
- Preprocessing methods (including feature selection) and motivation: max. 10 points
- Model selection method and motivation: max. 5 points
- Description of list of candidate models (and hyperparameters to tune) and motivation: max. 10 points
- Results interpretation: max. 10 points
- ROC Curve (and other plots) and interpretation: max. 5 points
- AUC score on test set: 5 points if $AUC^{(test)} \geq 0.84$; 4 points if $0.8 \leq AUC^{(test)} < 0.84$; 3 points if $0.75 \leq AUC^{(test)} < 0.8$; 2 points if $0.7 \leq AUC^{(test)} < 0.75$; 1 point if $0.65 \leq AUC^{(test)} < 0.70$. 0 points if $AUC^{(test)} < 0.65$.

Some useful functions.
`ColumnTransformers` (discussed during the Data Preprocessing lecture). Useful if you decide to write pipelines for the columns of categorical and continuous data separately, and then you implement them as estimator of a model selection method (such as GridSearchCV). If you are interested in how to specify the parameter grid of ColumnTransformer with GridSearch or RandomizedSearch, please see Appendix B below.

`make_scorer` and `roc_auc_score`. Both these functions are present in the `metrics` module of scikit-learn, and you should recall them from the Model Selection and Performance lectures.
`roc_curve`. Also present in the `metrics` module, it was discussed in the Model Performance lecture. It helps to retrieve the quantities for plotting the ROC curve.

cross_val_score, if you prefer tuning your pipeline/estimator manually or, for more automated methods, GridSearchCV or RandomizedSearchCV (remember to modify the 'scoring' argument of these functions according to the goal of the assignment).

Furthermore, you can use the `imblearn` library (discussed in the Data Preprocessing lecture) if you feel like it's needed. If you do so, remember to use the Pipeline provided by imblearn (it works exactly like a scikit-learn pipeline, but it allows including functions of the imblearn library).

Appendix A: The Churn data. Here we shall see a brief explanation of the variables present in the Churn data.

`State`. Column representing the states in the US.
`Account Length`. How long the account has been active.
`Area Code`. Area ID.
`Phone Number`. Surrogate for Customer ID.
`International Plan`. Whether present (1) or not (0).
`Voice Mail Plan`. Present (1) or not (0).
`Number of Voice Mail Messages`. Number of messages in voice mail.
`Total Day Minutes`. Minutes customer used service during the day.
`Total Day Calls`. Number of daily calls.
`Total Day Charge`. Daily cost of the customer.
`Total Eve Minutes`. Minutes customer used service during the evening.
`Total Eve Calls`. Number of evening calls.
`Total Eve Charge`. Evening cost of the customer.
`Total Night Minutes`. Minutes customer used service during the night.
`Total Night Calls`. Number of night calls.
`Total Night Charge`. Night cost of the customer.
`Total International Minutes`. Minutes customer used service to make international calls.
`Total International Calls`. Number of international calls.
`Total International Charge`. International service costs of the customer.
`Number Customer Service Calls`. Number of calls of the customer to the customer service.
`Churn` (class). 0 = No churn; 1 = Churn.

Appendix B: Tuning elements of ColumnTransformer. Let's suppose you have stored the name of your continuous feature in `numeric_feature_list` and your categorical ones in `categorical_feature_list`. Furthermore, you have prepared two pipelines (`num_pipe` and `cat_pipe`) to apply some transformations (`TransformerA1` and `TransformerB1`) to your numeric and categorical data. Subsequently you combine these operations with ColumnTransformer (in an object `all_pipe`), and write a pipeline (`new_pipe`) that chains the operations of such transformer with an estimator. Now, let's immagine you want to give this last pipeline to `GridSearchCV`, and tune the transformer of the continuous data (choosing between TransformerA1 and TransformerA2) ,

its hyperparameter (par1), along with the hyperparameter of the estimator (par3). The following snippet of code shows how to specify the hyperparameter grid for GridSearchCV (or RandomizedSearchCV).

```
num_pipe = Pipeline([
    ("opA", TransformerA1(par1)),
])

cat_pipe = Pipeline([
    ("opB", TransformerB1(par2))
])

all_pipe = ColumnTransformer([
    ("num", num_pipe, numeric_features_list),
    ("cat", cat_pipe, categorical_features_list)
])

new_pipe = Pipeline([
    ("preprocess", all_pipe),
    ("estimator", Estimator(par3))
])


param_grid = [
    {'preprocess__num__opA': [TransformerA1, Transformer A2],
    'preprocess__num__opA__par1': [grid_of_values],
    'estimator': [Estimator()],
    'estimator__par3' : [grid_of_values]
    }
]
```

2 Part 2: Theoretical Question

There is a concept which is often used in machine learning literature but simultaneously poorly understood: the concept of a *latent space*. Latent literally translates to hidden, so the latent space is a hidden space. In particular, it is a space which represents the data in a compressed way. Methods which are often mentioned in connection with latent spaces are matrix factorizations like SVD, PCA and k-means and deep neural networks. What the latent space is or how it can be visualized is most explicitely discussed for PCA. The principle which is used in PCA to derive the latent space is the matrix factorization, represented by a truncated SVD of the centered data matrix. The rank-r factorization of a matrix computes an r-dimensional latent space. In the scope of PCA, the latent space is the space spanned by the principal components.

Use the Olivetti faces dataset, introduced in Exercise 6 of Assignment 3 to visualize the latent space. From Exercise 6, you already got an idea about the latent space of PCA. Here, you are asked to apply these visualizations to PCA but also to the factorization methods of SVD (unconstrained matrix factorization) and k-means.

Provide for PCA, SVD and k-means clustering of the Olivetti dataset the following visualizations:

- A scatter plot of the $r = 2$-dimensional representation of the faces in the latent space, together with the faces which represent the two dimensions/features.
- A visualization of the faces which define the features of the latent space for $r = 5$.
- The reconstruction of the faces with indices $i \in \{0, 10, 20\}$ when using a rank of $r \in \{5, 25, 50, 100\}$

Point out differences and similarities of the three methods and describe how the differences in the matrix factorization objective functions are expressed in the obtained visualizations of the latent space.

3 Peer Review Paragraph (0 points)

Finally, each group member must write a single paragraph outlining their opinion on the work distribution within the group. Did every group member contribute equally? Did you split up tasks in a fair manner, or jointly worked through the exercises. Do you think that some members of your group deserve a different grade from others?