CSIS 3290 Fundamental of Machine Learning
Mini Project 02

Due date: Mar 08 17:00

Project Description:

In this project you will be working on a bank marketing dataset containing information of several thousands of direct phone marketing, performed by a banking institution in Portugal to offer their new bank term deposit product. The original data has more than 45000 with 20 features. You will be working on a 20% sample of the dataset with 17 features. Please see the last page of this document for the information about the dataset.

Project Submission Requirements

File/folder structure and naming convention

One single MiniProject2.ipynb file

A. Classification Modelling Requirement

Jupyter Notebook requirement

You are required to include one Jupyter file notebook in your submission. Your Jupyter notebook should include all the markdown texts signifying the steps with correct heading, python code, comments/analysis, and visualizations as stated in the following instruction. Note: You need to create the appropriate markdown headings for each section mentioned below. Codes should have some short comment describing the statement. Adding a markdown cell containing text before specific actions performed is appreciated.

1. **Title, Name and References**
   Include a title of your regression project. Include your name and student ID. Add information about any references you used to help complete the project

2. **Library import and data loading**
   Import all the required important library and load the provided unclean dataset.

3. **Data Analysis, Preparation and Wrangling**
   Before you start, please take a look at the original csv file to find out about the data. Then read the information in the last page of this project instruction about the dataset. Notice that **you should drop the 'duration' column** as explained in the dataset documentation. In addition to that, you should merge the two column 'day' and 'month' and change it into a numeric value, e.g., the number of days the last contact was performed until the end of the campaign. Note: the last date in the dataset is 17 November (2010). We will assume that we perform the analysis on 18 November 2010. In order to do that:
   - Make sure that the column duration is dropped

- Create a new column called “last_contact” as follow. Note that we need to use the datetime module in order to perform the manipulation

```
# drop duration as suggested by the documentation
df.drop('duration', axis=1, inplace=True)

# convert the two-column time to a column containing
# the number of days till the last day of campaign
from datetime import datetime as dt

last_date = "18 Nov, 2010" # assume the analysis date
dt_1 = dt.strptime(last_date, "%d %b, %Y") # change it to datetime

# the month needs to start with a capital letter
df['month'] = df['month'].apply(lambda x: x.capitalize())
df['date'] = df['day'].astype(str)+ " " + df['month'] + ", 2010"

# calculate the duration between the two dates
df['last_contact'] = df['date'].apply(lambda x: (dt_1 - dt.strptime(x, "%d %b, %Y")).days)

df.drop(columns={'day', 'date', 'month'}, axis=1, inplace=True)
```

After that, explore the dataset in your Jupyter notebook. Have a peek of the data and display the summary statistics of the data. Then you need to perform the following action:

- Create dummy variables for each categorical feature. You can find out all the categorical features by using df.info() or directly selecting the columns df.select_dtypes('object').columns
- Some of the categorical features will have the same value: 'yes', 'unknown', etc. In order to make sure that each dummy column unique, you need to use prefix related to the original column name. For example, for the column job, you should use get_dummies('job', prefix='job', drop_first=True)
- Make sure that each column has valid name, use lambda to rename the column name by replacing any appearance of hypen '-' with underscore '_'

There is a debate on theoretical machine learning on whether to use feature scaling before or after feature selection. Some ML techniques and algorithms need to use feature scaling to perform better. But at the end of the day, it all depends on the purpose and the ML pipeline used. In this project we will evaluate some new unseen data. In this regard, it will be easier for to do that if we perform feature selection before scaling.

4. Feature Selection and Scaling

Perform feature selection from the dataset. Use the following feature selection methods and choose the feature selection method that has the lowest number of features. If there is a tie, you should select the first method that has the lowest number of features.

- LogisticRegression
- Linear SVM (please use regularization hyperparameter value 0.001)
- SelectKBest with mutual_info_classif as the score metric, with the number of features to be selected equal to 10.
  Ref: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html

Select the method that gives you the least number of features. In order to get the columns selected for a specific model above, you can use model.get_support(indices=True). Make sure to save the dataframe of the selected features, or at least the column name of the selected feature for future use. In addition to that, save the feature dataframe into a csv file with the following naming convention: feature1.csv.

Use robust scaler to scale the feature selected. Make sure to separate the target column 'y_yes' from the dataframe before performing the scaling. Note: you can use pipeline to join the implementation of the feature selection and scaling.

5. Classifier Modelling

Divide the dataset into training and test sets with ratio 75:25. Use pipeline to implement the following classifier models with cross-validation and calculate the mean value of the cross-validation scores:

- The cross validation is applied on the training set only
- You need to use the scaled features, obtained from the previous step
- You need to use k-fold cross-validation (You can select the fold anywhere between 2 to 5)

You need to select the classifier that has the best mean accuracy score from the training dataset. If there is any tie between two classifiers, you should choose the first classifier. The classifier you need to use are as follow:

- Naïve Bayes classifier
- KNN
- Decision tree classifier
- Random forest classifier
- Ada BOOST classifier
- XGBOOST classifier

In order to implement the above models, you can use the same parameter as the one used in the lecture 8 demo. For the decision tree and random forest, you should limit the max_depth to be equal to 5, while the random forest can have estimator from 10 to 1000 trees.

When we use the ensemble method like random forest, boosting like ADABOOST or gradient boosting like XGBOOST, cross validation is not needed because of their inherent splitting of the dataset in their training. They will also work fine without feature scaling. But in this project I want you to use feature scaling and cross validation for those algorithms as well. Do some experiments yourself to check whether the performance will be better with scaling (and cv) or not.

6. Classifier Model – The Second Try

Implement a new feature selection method with random forest as the estimator. The number of features you get from using the random forest should be less than number of features obtained at step 4. Then, use robust scaler to scale the selected features. Make sure to separate the target column 'y_yes' from the dataframe before performing the scaling. Make sure to save the model for the robust scaler because we will need it later.

Then, save the feature dataframe into a csv file with the following naming convention: feature2.csv.

Use this new dataframe and implement all the classifiers stated at step 5.

7. Model Evaluation

Compare the accuracy of the best models obtained at step 5 and 6. You should analyze the difference in the accuracy values obtained between the one obtained at step 5 and 6. You should also comment on the amount of features selected between the two results. You should see that the accuracy of the best classifier obtained at step 6 is only slightly less than the best classifier obtained at step 5, even though we are using less number of features. We will use the classifier and features selected at step 6. You need to then evaluate the classifier model by finding out the accuracy of the model in the test set and by producing the following (along with your analysis):

- Confusion Matrix
- Classification Report

- ROC Curve

You should see that the number of false negative is quite big. It is easier for the classifier to determine the rejection (class 0), than to predict that the client will sign into the new banking product. Provide your insights on the possible cause or a way to improve our classifier performance.

8. Prediction

Predict whether a client with the features displayed in the table below will agree to join the new product being marketed by the banking institution. Make sure that the new dataframe from this unseen data consists only the columns that are used in the feature selection you used at step 6. In order to do that:

- You do not need to create the last_contact column since the new data has this information.
- You need to transform all categorical columns into numerical. However, you should not use drop_first. The performance of drop_first is a bit sketchy and you may drop the column that in fact is part of the features selected at step 6.
- You need to then compare the columns you have in the new dataframe. Make sure that the columns order and columns name are exactly similar to the ones included by the feature selection process at step 6. Failure to do that will affect your model performance since we will use a scaler's transform.

In order to use the robust scaler, you should not create a new robust scaler model and fit the new data. Instead, you need to transform the new data using the robust scaler model you used in step 6. Make a prediction based on the new scaled features. Discuss your finding by providing some suggestion to the management on how to increase the number of people to agree to sign into the new product.

The following shows the new data. If your chosen classifier needs additional feature(s), you should add the feature and add arbitrary values for the corresponding feature (see the allowable values for the dataset on the last page).

age	balance	job	marital	housing	contact	campaign	pdays	last_contact	poutcome
45	8000	blue-collar	married	yes	cellular	12	300	45	success
65	700	retired	divorced	yes	telephone	2	1	1	success
80	100	unknown	unknown	no	telephone	1	1	200	unknown
21	300	students	single	no	cellular	3	90	180	success

B. Project Report Requirement (within Jupyter notebook)

Based on your Jupyter notebook code and result, create a project report containing your findings. The document report should not exceed 5 pages. The report should contain summary of each step you did on the Jupyter notebook, such as:

- Introduction

- The best classifier model and features selected
- Analysis of the chosen model performance: accuracy, confusion matrix, classification report and ROC curve
- Discuss your result and their managerial implications to the banking institution
- Comment on the results on the expected outcome of the client mentioned at step 8 at the Jupyter notebook requirement and suggest some managerial strategies for the bank institution.

C. Project Grading Criteria

The project will be graded on a scale of 20 points.

Criteria		**Grading**
Project submitted, named properly with all files included in their corresponding folders to Blackboard.		1
Part	**Detail**	
Jupyter	Data wrangling and preparation	2
	Feature selection and scaling	3
	Classifier Models *The feature selection 1 csv file is included in submission	3
	Classifier Models – the second try *The feature selection 2 csv file is included in submission	3
	Model Evaluation	3
	Prediction	2
Report	The report was submitted following the stated requirements above	3