F20ML 2020-21: Coursework 1 [15%]

In this coursework, we will look at the Decision Trees and k-Nearest Neighbours algorithms in order to train models to perform a binary classification task. The coursework consists of four parts: (i) this document which contains a description of the problem and the data, (ii) the dataset splits in 3 separate files, (iii) a dataset loader class stored in a file called ***'BankDataset.py'*** and (iv) an accompanying Jupyter notebook called **'Coursework_1.ipynb'** file containing the necessary scaffolding code (with missing parts) needed to implement the algorithms to answer the questions below. In particular, you will have to **complete the cells therein in order to answer the various questions below**. **You will NOT have to submit a document report in .pdf or .doc but instead you will enter your responses (upload the completed .ipynb notebook and a couple of plot images) in the corresponding Test created on Vision. Read the instructions in the last Section 'Deliverables' below for more details.**

Note: We will use the existing implementations for the Decision Trees and k-Nearest Neighbours algorithms from the Python library **scikit-learn**.

Introductory Scenario

A retail bank wants to study the success of telemarketing calls for selling the bank's long-term deposits. They hire you to develop some data mining (and machine learning) tools for this purpose. They provide you with a large dataset comprising 11,755 instances with 16 + output = 17 attributes. The description of the 17 attributes are as follows:

Input variables consisting of background data from the bank's 11,755 clients:

1 - age
2 - job : type of job
3 - marital : marital status
4 - education
5 - default: has credit in default?
6 - balance: average yearly balance, in euros
7 - housing: has a housing loan?
8 - loan: has personal loan?

The following attributes are related with the last contact of the current telemarketing campaign:

9 - contact: contact communication type
10 - day: last contact day of the month
11 - month: last contact month of year
12 - duration: last contact duration, in seconds

Other recorded attributes are:

13 - campaign: number of contacts performed during this campaign and for this client
14 - pdays: number of days that passed by after the client was last contacted from a previous campaign (-1 means client was not previously contacted)
15 - previous: number of contacts performed before this campaign and for this client
16 - poutcome: outcome of the previous marketing campaign

Output variable (desired target):
17 - y - has the client subscribed to a term deposit?

The classification goal is to predict if the client will subscribe to a term deposit (variable no. 17: y) based on the remaining 16 attributes. The dataset has been nicely split into train/dev/test partitions and is available to you. You have already started up Anaconda navigator, Jupyter Notebook and have imported scikit-learn!

Part I - Dataset Analysis [5 marks]

You can find the dataset (train/dev/test splits) in the coursework assignment through Vision. All files should be in the same directory as the '**Coursework_1.ipynb**' file which has the scaffolding code.

- Import the **training set ONLY** using Pandas (**HINT**: the dataset is in csv format). We have given you the *`feature_names`* as an array already to help you construct the DataFrame properly.

Answer directly on the Vision Test (Coursework 1) the following questions:

1. What is the type of each feature? (Bulleted list with *<feature name, feature type>*)

2. Is the dataset balanced? Justify briefly (*one line*).

3. Print the **feature values** of every feature. In the case of **numerical** features print the **range** and **average** (with 3 decimal points) instead.

4. Using plots of **histograms**, **bar-graphs** or **heat-maps** *very briefly* (*i.e., with one line only*) comment on the following (don't forget to include the plots in your .ipynb notebook):

   - Does the `*age*` feature follow a **normal** distribution (just by eyeballing)?
   - Is the `*poutcome*` feature **unimodal** (just by eyeballing)?
   - Is the `*education*` feature **unimodal** (just by eyeballing)?
   - Taking into account **only** the **numerical** features do you notice any **correlation** between pairs of features?

Part II - Decision Trees [5 marks]

It's time to train some models. First stop; decision trees (*implement the code selection under the Decision Trees section)*.
Note: You can leave the criterion parameter to the default, i.e., Gini.

For this part (and the next one) we will be loading the dataset using a custom data loader we have created for you (no, we are not trying to be fancy!) that performs a couple of pre-processing steps (**1-of-k** or "one-hot" representation and **feature scaling**) in case you need them. The loader is implemented in a separate python file '**BankDataset.py**' that you can import similar to any other package (don't worry we have already done it for you!) You will be using the following command:

```
load_dataset(dataset_filename,preprocess_onehot=True/False,app
ly_scaling=True/False)
```

Remember that you will have to **load all three datasets** before training your decision trees.

Note: By default, the decision trees algorithm implemented in scikit-learn accepts only numerical values, so the *"one-hot"* parameter should be set to True when calling the load_dataset() function. This way, we automatically convert categorical features to a 1-of-k representation. For example, a feature "colour" which have three values, namely "Red", "Green", and "Blue", gets mapped to <1, 0, 0>, <0, 1, 0> and <0, 0, 1>, respectively depending on what is the actual value in the example. So you will have to preprocess your features that are categorical.

Ok, ready to fit a model. But how deep should the decision tree be? Perform hyper-parameter tuning for your Decision Tree algorithm.

Answer directly on the Vision Test (Coursework 1) the following questions:

1. What process did you follow for tuning the Decision Tree Algorithm? *Very briefly* (in a short paragraph or using bullet points) outline the steps.

2. (a) Plot the training and development accuracy curves (in a single plot) against your hyperparameter values (Just upload the file).
   (b) Briefly explain what is going on in this plot.

3. What is the accuracy on the test set? Explain *briefly* how you came up with this number (one line).

Part III - k-Nearest Neighbours [5 marks]

Ok, decision trees are cool, but maybe we can do better with k-Nearest Neighbours … let's find out!

Note: Remember that you will have to **load all three datasets** again before starting training. Be careful with your preprocessing parameters.

Answer directly on the Vision Test (Coursework 1) first the following questions:

1. Some of the features in our dataset are *categorical*. Explain *briefly* (not more than a paragraph) if there is an issue with the way we are currently representing them, i.e., in the case of k-Nearest neighbours.

2. Some other features are *numerical*; is there any issue with these in relation to k-Nearest neighbours? Explain *briefly* (with a few sentences).

Right, fitting models; round two. Perform hyper-parameter tuning for your k-Nearest Neighbour algorithm.

Answer directly on the Vision Test (Coursework 1) the following questions:

3. (a) Plot the training and development accuracy curves against your hyperparameter values (Just upload the file).
   (b) Briefly explain what is going on in this plot.

4. What is the accuracy on the test set? Explain briefly how you came up with this number (one line).

5. So ... the moment of truth! You are about to present your results to your project manager; which model are you going to choose for predicting the success of the telemarketing campaign based on the bank data?

Deliverables

You will have to **submit the Test in 'Assignments>Coursework 1 - Test' on Vision**. Over there, you should (i) upload the completed .ipynb Jupyter Notebook (**it should be named: Coursework_1_H0XXXXXXX.ipynb, by replacing the Xs with your Student ID**) , (ii) enter your answers to the questions outlined in the description above, (iii) upload a couple of plot images as part of your answers clearly indicated within the test.

It is advisable to properly comment your code. **Note**: Any 3rd party source used must be properly cited in the code (see also below).

Important Notes!

- **The assignment counts for 15% of the course assessment.**

- You are permitted to discuss the coursework with your classmates and of course with me and the teaching assistant. However, coursework reports must be written in your own words and the accompanied code must be your own. If some text or code in the coursework has been taken from other sources, these sources must be properly referenced. In particular, for pieces of code you get from the web (e.g., from StackOverflow), minimally you should provide the link where you found it as an inline comment in the jupyter notebook. Failure to reference work that has been obtained from other sources or to copy the words and/or code of another student is **plagiarism** and if detected, this will be reported to the School's Discipline Committee. If a student is found guilty of plagiarism, the penalty could involve voiding the course.
- You should **never give** hard or soft copies of your coursework report or code to another student. You must always refuse any request from another student for a copy of your report and/or code. Sharing a coursework report and/or code with another student is **collusion**, and if detected, this will be reported to the School's Discipline Committee. If found guilty of collusion, the penalty could involve voiding the course.
- **Pay special attention** to all the **Labs** as they should provide lots of insight as to how to tackle most of the questions in this coursework. The **Lab in Week 4** is specifically designed for you to ask any questions related to the coursework (there won't be any Lab Sheet for that Week). Also, **CHECK THE DOCUMENTATION OF scikit-learn** when you are in doubt.
- Your report and .ipynb notebook (exactly 2 files) should be submitted by **15:00 on Saturday 16 October**. No individual extensions are permitted under any circumstances. Students who submit after the deadline but within 5 working days of the deadline will be award 0.7x(awarded coursework mark). Submissions that are more than 5 days late will receive 0 marks.
- You will receive the final mark, answers, sample code solution and cohort-wide feedback no later than 15 working days.