15-488 Spring '20 - Project

A Query Answering Machine

Teacher: Gianni A. Di Caro

TA: Aliaa Essameldin <aeahmed@qatar.cmu.edu>

Introduction

The class project is an opportunity for you to apply the machine learning techniques learnt in class to the analysis of interesting real-world problems. You will build an ML-based *query answering machine* (QuAM): a software that provides useful answers to meaningful questions based on a trained model.

For the project, you will have to select an interesting scenario and envisage a stakeholder that would use your QuAM for asking purposeful questions. You should consider the usefulness of the question and the availability of relevant data. We leave it up to you whether you want to build a *classifier* or a *regressor*

It is up to you whether you want to work on this project *alone* or in a *team of two*. If you should choose to work alone, we would be expecting a less rigour and less amount of work from you, but you are still expected to submit all parts of the project

The project, which will be 30% of your final course grade, comprises 4 deliverables, D1, D2, D3, D4, for a total of 100 Points + 20 Bonus points:

1. D1 [Report]: **Initial Proposal and Dataset** (10 Points)
2. D2 [Dataset and Notebook]: **Final Dataset** (35 Points + 20 Bonus points)
3. D3 [Software and Notebook]: **Query Answering Machine** (40 Points)
4. D4 [Presentation]: **Final report** (15 Points)

The descriptions between square brackets define the type of the deliverable: *Report* is a PDF document, *Notebook* is a Jupyter notebook mixing up software and descriptions, *Dataset* is a file with data, *Software* is a .zip with your full python project, *Presentation* is a 15 minutes live presentation.

The next section details what you need to do for each deliverable. But before you jump into that, here are some important considerations that you need to keep in mind:

- Your data **CANNOT BE TEXTUAL**, since we did not really cover analysis and processing of text data in class. We recommend using numerical, temporal, or image data in your problem.
- You **MAY NOT** use techniques, models, or libraries that we did not cover in class. You may create something new by merging or tweaking discussed algorithms, but you absolutely cannot use any known techniques that we did not cover.
- Only exception for the previous rule is that you may use tools/libraries for the bonus part (collecting/creating data). Must get them approved by Teaching staff first, though.
- Do not try to jump to a good solution. In fact, a good part of your grade is on *process*. Find some simple and maybe naive answer to your question first and try it, then look into how you can improve on it, and iterate the process.
- We strongly recommend you read the entire handout and carefully plan your workflow before you start writing the proposal.

D1: Initial Proposal and Dataset (10 Points)

To begin the process, you need to come up with a problem that is interesting to you or to the world. Either a regression or a classification problem, depending on what you have been assigned.
Each team will first submit a *one-page proposal* that includes the following:

- Project Title
- Student(s) names, andrewID(s)
- Problem Specification: This should be one/two paragraphs briefly describing the problem and the questions of interest.
- Project Idea: This should be a short description of the QuAM that you will build, its type (classification or regression), and its interface.
- *(For Teams)* Divided spec: a brief description of how work will be divided.
- Dataset (preliminary): You have two options for the data that you can use in your project. You may find an existing dataset or *produce* your own data for an extra **20 bonus points** (see section D2.1). At this stage you have to make a first decision and provide a short description of how you would go about finding/producing the data.

D2: Final Dataset - Data Collection, Analysis, Wrangling, Feature Engineering (35 Points + 20 Bonus points)

Once you have set up your initial plan, you have to go through the steps that will take you from data generation or ingestion, to a set of engineered data features that can be effectively used for building the QuAM. In the ML pipeline, this is the part that prepares the data for feeding to ML algorithms. These preparatory steps, that can be seen as sub-deliverables of D2, are the following:

1. D2.1: Data Collection (5 Points + 20 Bonus points)
2. D2.2: Data Analysis (7 Points)
3. D2.3: Data Wrangling (9 Points)
4. D2.4: Feature Engineering (14 Points)

The actions that you will take during each step and the resulting outputs will have to be documented in a Notebook named `DataPreparation.ipynb`, consisting of four main sections named after the steps above.

At the end of the data preparation steps you should have a dataset with clean and well engineered features. This will be your *final dataset*, the one you will be using to train your models for the QuAM.

D2.1: Data collection (5 Points + 20 Bonus points)

Data are at the core of any ML-based QuAM. Therefore, finding or building the right dataset is the key to success. As pointed out in section D1, you have two options for the dataset: find an existing one or make your own custom data.

1. *Finding a dataset* (5 Points): You may choose to find a dataset online and be inspired by it for the problem. In this case, you must cite the source in your proposal and whether you will tweak it and how. Here are some websites that we recommend as potential starting points for your quest:
   - Kaggle, Just everything - `https://www.kaggle.com/`
   - World Bank data collections `https://datacatalog.worldbank.org/collections`
   - Data.Gov - `https://catalog.data.gov/dataset`
   - UCI Repository - `https://archive.ics.uci.edu/ml/index.php`
   - FiveThirtyEight - `https://data.fivethirtyeight.com/`
2. *Custom dataset:* (5 Points + 20 Bonus points): Alternatively, you may choose to collect or produce your own data. There are two ways to accomplish this.

- You can *collect* data by *scraping the internet*, this works best for image data. You will have to write a python crawler that you submit with your final project. A web crawler takes a 'seed' URL that it scrapes. You may use any libraries that you find helpful but you cannot use any existing web crawlers. You need to think of various aspects: how will the crawler label your data? Can you use unlabeled images? You need to think of all of that and discuss it with the teaching staff before you commit to this challenge. We may also help for designing the crawler.
- You can *create* data by writing a script that produces values, this works best for categorical and numerical data, while it may be a bit more complicated for image data. Notice however, the script cannot be as simple as a bunch of random data points printed to a file. Is there an underlying distribution for your data? Are the features correlated to one another? How would you simulate anomalies and noise in your data? If you're going to get the extra 20 bonus points this will depend on how meaningful your dataset will be.

D2.2: Data Analysis (7 Points)

It doesn't matter where your data comes from, you will need to explore your data first, in order to understand it better and gain useful information about its distributions, ranges, scales, and so on. Therefore, you need to produce some exploratory data analysis (EDA), that in turn will direct the next two phases of data wrangling and feature engineering. The EDA might consists of graphs, plots, summary statistics, and so on.

D2.3: Data Wrangling (9 Points)

Data are dirty, features might need scaling and normalization, entries might be missing, categorical data might need to be transformed into numeric data, and so on. All these operations of data cleaning and preparation, that we summarize with the term *wrangling*, need to performed on your dataset, if necessary, and thoroughly reported in the Notebook.

D2.4: Feature Engineering (14 Points)

Selecting or extracting the right features for the questions that the QuAM will have to answer is a critical step preparing the data. Features might be correlated or redundant, such that feature pruning or selection might be useful or even necessary. Some features are more relevant for target predictions than others, such that data complexity can be reduced by selecting only these more relevant features. New features can be generated or extracted from the existing ones. At this aim statistical indicators can be useful, as well as the use of more complex techniques for features extraction (e.g., HOG for image data), depending on the type of the data. Again, all the feature engineering steps that you will (have to) perform need to be added and documented in the Notebook. At the end of the feature engineering step your final dataset is built.

D3: Query Answering Machine (40 Points)

Now that you (hopefully) have a well-engineered dataset, it's time to design the QuAM! This means selecting the appropriate ML algorithm, setting the right inductive biases, selecting good values for the hyperparameters, understanding when/if the machine provides correct answers, evaluating the computational requirements for training and testing, revising the features if necessary, and so on. In addition, the design of the interface to the QuAM needs to be flexible enough to input the queries and set the values of regulatory parameters, if any. The output of all these operations will be your *final product*, the QuAM software that can be used to answer the questions of interest.

Your work on the QuAM will be graded accounting for the quality of what you achieve, as well as the processes that you follow in order to iteratively improve the quality of your solutions. The rationale behind the grading approach is described in the two sections that follow.

D3.1: Process Iterations [Notebook] (30 Points)

It is not expected, in general, that you find all the right solutions in one shot. Instead, a successful project starts simple, learns from mistakes and trends observed then builds up towards a sophisticated solution. This is why we expect you to have tried different things before finding a solution that will be translated into the final product. Namely, you will do your work in *iterations*. Each iteration is a working solution that is possibly different and has different complexity than the one preceding it.

We expect that you'll go through *three iterations in total. Each iteration is worth 10 points* (Solo projects can choose to only do 2 iterations worth 15 points each).

In the Notebook report, named `QuAM_report.ipynb`, you will describe reasonings and relevant data at each iteration according to the following structure:

- **Solution and Justification** (2 Points): Describe your current solution approach / algorithm in words, justify why you are choosing to try this specific approach then implement it. In your reasoning you can refer to the data analysis or to the results from the previous iteration, or to some other priors.
- **Solution Details** (4 Points): Justify all/any parameters/choices made in this experiment and explain any other relevant detail. Consider using plots to justify your choices.
- **Lessons Learnt** (4 Points): Visualize the trained model and measure its CV accuracy and report on it. What have you learnt from this experiment? How do these results help you move forward?

D3.2: Quality of the final QuAM [Software] (10 Points)

You have to submit a file, named `QuAM.zip`, including the the python code of your QuAM, any resources that it uses and a `README.md` file.

The *software interface* of the QuAM is up to you. We are not requesting anything more complex than a python module that can run in simple terminal interface and allows to pass inputs by file and/or by command line. However, you should feel free to build anything, for instance using `ipywidgets` (`https://ipywidgets.readthedocs.io/en/stable/`) to add buttons and nice graphics to a notebook, or using a simple TKinter interface, or building a full-fledged web-app.

You may divide your project across multiple modules. Just make sure you include all modules and any datasets or resources used by your QuAM. We should be able to seamlessly launch and use your QuaAM. A file `README.md` (in markdown) should be also included in the submission in order to describe all content in your submission and explain how to use the software.

Your final QuAM will be evaluated based on how good the answers of the QuAM are in relation to the complexity of the problem (i.e., if your problem is too easy, it won't be impressive that you reach high accuracy with your answers . . .). Additional aspects, such as the use of computational resources and the easiness of use of the interface will be accounted for.

D4: Final Report (15 Points)

During the last week of classes, you will *present your final product to the Teaching Staff Via Zoom.* You will have 10 minutes to present your work followed by 5 minutes for Questions. While your notebooks should be detailed enough to justify every decision you made trough the process of building your product, this is your opportunity to practice your oral presentation skills. We will leave it up to you to decide which parts of your process are worth including in your interview and which are not.

You should also point out what would you do to improve your results. In this part you may refer to algorithms or techniques that we did not cover in class.

After the presentations, you will have the option of using our feedback to make minor improvements in your final software and re-submit for grading.

Timeline

- 22 March - **D1: Initial Proposal and Dataset**. File to submit: `proposal.pdf`
- 01 April - **D2: Final Dataset**. Files to submit: `dataset.zip` including your `DataPreparation.ipynb` and any intial datasets it needs to run (or a URL if the dataset file is too big).
- 19 April - **D3: Query Answering Machine**. Files to submit: `QuAM_report.zip` including your `QuAM_report.ipynb` and any data it needs to run (or a URL if the dataset file is too big), `QuAM.zip`.
- 21-22 April - **D4: Final Report**. Live presentations on Zoom.
- 23 April - **D3: (Optional) an improved Query Answering Machine** `QuAM.zip`.