Project - Big-Scale Analytics

Professor: Michalis Vlachos

Deliverables due: Multiple milestones. Consult syllabus in Moodle.
Number of group participants: 3 (best to have one person in each team who is *not* a native French speaker. See later why!)

Foreword

You have decided to form a startup called “LingoRank” with two of your University friends and become a millionaire. You have until June to create a proof of concept for your investors. Your startup will revolutionize the way people learn and get better at a foreign language.

The Idea

You have noticed that, to improve one’s skills in a new foreign language, it is important to read texts in that language. These text have to be at the reader’s language level. However, it is difficult to find texts that are close to someone’s knowledge level (A1 to C2). You have decided to build a model for English speakers that predicts the difficulty of a French written text. This can be then used, e.g., in a recommendation system, to recommend texts (for example, recent news articles) that are appropriate for someone’s language level. If someone is at A1 French level, it is inappropriate to present a text at B2 level, as she won’t be able to understand it. Ideally, a text should have many known words and may have a few words that are unknown so that the person can improve.

To do this would require:

1. Thinking how to model the problem and gathering data (French texts, sentences, news articles etc.) and labelling them with the relevant level,
2. Building a model that predicts the difficulty/level of a new text,
3. Evaluating how good the model is.

This process is iterative. It may require several cycles.

The project will be conducted in multiple milestones that are due throughout the semester. We will give you feedback and a *tentative* grade for each milestone. The overall grade will be assigned at the end.

The **goal** of this project is to get acquainted with the data-driven problem solving methodology, and deal with the problem within a group, but without too much external help (i.e., we give you the problem, you have to make the decisions!).

The Team

Putting together the right team is the first skill you should master! Don't create a team where all the people have the same skillsets. Ideally, you should have at least one person good in coding, at least one person good in HTML and user interfaces, and at least one non-native French speaker.

Background

You may want to read **this** document first. This is not the solution that we need, but it will give you some idea of the problem, how to evaluate it, and also how to visualize your solution. Try also to spend the first week of your project getting acquainted with the problem, reading and brainstorming how you would solve it. Make sure you go over the *complete* project description (not only Milestone 1!) including the references. Read also both of these resources and search for related videos in YouTube:

- https://cloud.google.com/natural-language/automl/docs/beginners-guide
- https://cloud.google.com/automl-tables/docs/beginners-guide

Learning Goals

For this project, our goal is not to merely "write code" or "use the cloud". More important is to learn and understand the related concepts. For example, when you use the AutoML services, it will show you an evaluation of the model using precision, recall, AUC, F1-score, etc. Make sure you understand what those mean and how to interpret them. This will help you improve your model. It is also necessary to see where your model fails. If you see where it fails, it may mean that you need to provide more examples of that observed case. The machine learning process using data is not a one-off process. It is iterative, it requires thinking, it requires perseverance. So, don't be content with the first model. You have to go deeper and try to improve it. This is what we expect to see in this project.

Milestone 1 - Reading/Thinking & Gathering the data

First, you have to do some search on the topic, read some papers and think **how** to solve the problem. Will you solve it as a classification, as a regression problem or in some other way? Will you model the difficulty of each word separately or of a sentence as a whole? Once you have an idea how to model it, you go on to collect the data (note: here, we assume that you have attended the previous class on "Data Mining and Machine Learning"!). Consider spending one week brainstorming and reading about the problem.

Then you will start collecting the data and labelling them (if unlabelled). This is useful both for creating the predictive model and for doing the evaluation (remember the train/test split?).

Deliverable:
- A link to your GitHub. In the README, describe
  - ➡ Which papers did you read on the topic? (put down here only the useful ones);
  - ➡ How do you intend to solve the problem?
  - ➡ Links to the relevant datasets you found. Mention also the source of the texts, and **how many** annotated sentences you contributed for the following step.
- Go to this Google Sheets document and create a sheet with your group name. Copy *at least* 1000 examples (sentences) with their label (A1 to C2). Each label needs to have 3 verifications from each of the group member. It's OK to disagree. This may signal that certain examples are uncertain. If there is disagreement in some labels, you may either not use that sample or label it

using the average of the labels.

Of course, you may have collected much more data (if you can find them already annotated). In this case, provide a link to a SWITCHdrive or Google Drive folder.

Note, that one approach would be to model a sentence as the average difficulty of all the words in it. For this, you may want to look at how frequent a word is in a corpus (e.g. Wikipedia or other). Words that are frequent on the Internet are assumed to be easier. For a package related to that, look here. If you decide to follow this approach when you do the model, you would still need to contribute to this part with sentences and annotations.

In general, try to find datasets that are already annotated (see resources at the end). If you cannot find any, you have to do this process on your own (e.g., by collecting news titles from the Internet or extracting sentences from French text books you have used and then annotating them). I would recommend to start early! We expect to see at least 1000 good examples!

Remark: In the references at the end, we give you 2 sites that provide already labelled texts. These can be a starting point. Search also for more labelled datasets.

Note: Your final deliverable should also consider a way of dealing with cognates (words sharing the same etymology). Such words, even though their general frequency might be low, should be considered as easy, because, because it is straightforward for an English speaker to understand its meaning (liberté -> liberty, raison -> reason, nerveux -> nervous, etc).

Milestone 2 - Creating/Evaluating the model

Create a predictive model that can be used to predict how easy or difficult a French text is. You may model the problem as a classification problem (e.g. mapping a level from A1 to C2) or as a regression problem, eg A1 = 1, A2 = 2, B1 = 3 and so on). For this, we recommend using the various cloud services from Microsoft or Google, such as Google AutoML, that can be used for:

- **Text classification and sentiment analysis:** https://cloud.google.com/natural-language/automl/docs/features
- **Regression:** https://cloud.google.com/automl-tables/docs/beginners-guide

Watch the video presentation of the previous class that describe those services:
1. Text classification with Google AutoML
2. Sentiment analysis with Google AutoML

Deliverable: A callable API point which, given a text in French, will return its difficulty. Part of the exercise here is to learn how to use those cloud services (and/or build a Flask/docker web service). We also expect an update in the README of your GitHub.

Note: In this part, you should also consider how difficult a French word is for an English speaker (the "cognativity"). If a word is very similar to its English counterpart, it is naturally easier to understand. For this part, you should build another service. Your final solution can be a combination of the outputs of different services/analytics.

Depending on how you model the problem, pick the right metric and annotated data to evaluate it. For example, if you model as regression, you may use an RMSE metric; if you model as classification, use F1-score.

Note: Even if you model it as a regression problem, in the end we need the output as class labels (A1-C2), so that we can compare the different solutions and find out which group had the best solution. In this case, you need to incorporate some logic to output labels.

Based on the data that everyone collected, we will create a train/test split. You will then train on the train part and test your model on the (unseen) test part. **Your TAs will pick the train and test data based on the data collected for Milestone 1, so that we have a uniform evaluation across all teams.**

Design choices: There are several design choices that you may have to make here. E.g. to determine the difficulty of each word, do you use the full word or the stemmed version? Perhaps you also choose a set of n-grams (e.g., 4-grams) to represent each word and each sentence and then using that you determine its difficulty. These are all important design choices. You should experiment with several designs and **argue with numbers and metrics** why you chose a particular one.

Milestone 3 - Iterate & Improve

By now you have gathered data and created a basic model. But you can still improve it! At this point, you may also realize that the data you collected was not correct or that there are other ways to improve the model. In addition to evaluating the model, describe in your deliverable (i.e. in the README of your GitHub) how you improved the model and the data collection when you saw the predictive quality of your model. Things that you can also try to improve your model are:

- **Error analysis** by checking the confusion matrix and particularly the cases where the classification is incorrect. Why was this the case? Can you observe any patterns in those error cases?
- **Visualization** can help in the error analysis. If you estimate the difficulty of a sentence as the average of the difficulty or cognativity of words, then visualizing those which are the difficult words or cognates in a sentence can be useful. See for example the following visualization (the more red, the easier the word; the more blue, the more cognate it is). Can you think of way to visualize your results?

je
suis
important
et
européen

- **Feature augmentation:** Perhaps adding structural features of the sentence, such as part of speech (POS) analysis, eg via Spacy, can add helpful features for the prediction.

Extensions & Packaging the solution

We strongly advise you also to deploy a simple UI on the web which people can use. In that UI, you can put a sentence and have its difficulty/level returned. You may also build a simple recommender system. One other idea would be to extract subtitles from YouTube videos, then classify the text as A1-C2 and recommend videos similar to someone's knowledge of French. As you can see, the sky is the limit.

Note: Because this is a Master's course, we expect that most teams will "go the extra mile" to differentiate themselves from the other teams.

Final Deliverables

I. **Github:** A project **GitHub page** which will be updated accordingly during the different milestones. The main README and the notebook should reflect your current progress (and the relevant code). Note that, as we progress through the semester, you may need to revisit the previous parts. This is ok. You should evaluate your final result. Some part of your grade will also be influenced by the accuracy of your solution on the test set **evaluation.** In your README, mention your score for the metric and test data prepared by your TAs.

II. **API Services:** Provide the **endpoints of all the services** that you use and give examples how to call them. Clearly state what is the URL of your main service. Your main service should accept a sentence in French and return back its level. You may also have created other services (this is common when you design using a micro-services architecture), e.g. one which returns what the "cognativity" of a French word is. Clearly state all the web-service APIs that you use.

III. Tools: Mention which technical tools did you use: Flask, docker, cloud service, etc.

IV. **UI:** A usable UI in the cloud where we can try your service. We expect to see (at least) a simple webpage with a UI where you enter a text and it returns back its easiness/difficulty level. Feel free to get creative here!

V. **Video:** Create a YouTube video of your solution and embed it in your notebook. Imagine you are giving a presentation or a tutorial. The video should explain:
   A. The general architecture (cloud provider, technology, APIs, etc.)
   B. The algorithm (how you determine the difficulty, how you tackle cognativity)
   C. Show the UI that you have created
   D. An evaluation of your solution (accuracy, precision, recall, F1-score, etc.)

   Post the video link to #project channel in Slack. All projects will also be **presented live** by the group during the last class.

What we expect is that you would have used as many as possible of the tools shown during the course, e.g. cloud, Flask, Docker, etc. Full points will be given to those projects that took advantage of those components. However, you may also use more standard Python, but then you may expect some deduction of points.

Grading Scheme

I. **75%** of grade: GitHub quality, code quality, solution (UI+services) and evaluation
II. **25%** of grade from video + live presentation

Logistics and Deadlines

1. Register your group on Moodle (three people) **(by Friday 12pm of week 1)**
2. Deadlines for the different milestones are found on Moodle.

Useful Resources

- Common European Framework of Reference for Languages - Global scale
- French texts of different levels (1): https://french.kwiziq.com/learn/reading
- French texts of different levels (2): https://lingua.com/french/reading/
- https://www.fluentu.com/blog/french/french-cognates/
- Wordstats: https://pypi.org/project/wordstats/
- Example of an application, FluentU: https://www.youtube.com/watch?v=jDodNuNCyZY
- A graded lexicon for French as a foreign language and also here
- Oxford-Hachette French dictionary (can be useful for detecting cognates)
- Predicting using Google AutoML: https://www.youtube.com/watch?v=Kfg62KTH9W0

Text Classification with (GCP)