Project plan

Project name: T23 - Classification of an Academic Success

Project link (Kaggle): https://www.kaggle.com/competitions/playground-series-s4e6/overview

Repo link (private): https://github.com/azakatov/ut-ml-project-2024

Project members:

- Andres Alumets, Anton Zakatov, Pihla Järv, Muhammad Sohaib Anwar
- Our team decided that everyone shares all roles in parallel (i.e. no clear division between roles for now)

Problem statement:
The problem is that many students drop out of universities. This is a problem for academia, the country and students themselves. If we find out a way to predict who has the greater risks of academic failure then we will be able to know who should be assisted to avoid dropping out.
At the same time, the dataset seems to be known to us (as students) and probably we will be able to find some patterns that non-students could not catch.
Also, this competition seems to be a good project for beginners in ML and is indeed suitable for learning purposes.

Objectives:
Looking at the Kaggle competitions results we should aim at about 83.5% accuracy but the minimum accuracy that we would be satisfied with is 80%.

Data:
The data is available in Kaggle.
Data requires preprocessing: outliers elimination, invalid data elimination, missing values elimination/correction. Depending on the model, we may need to apply such techniques as one-hot encoding, normalization and features selection.
We found out that the data is unbalanced so we will need to make a training set balanced.
We assume that we may need to do features engineering as well.
For better data understanding we need to prepare visualization.

Methodology:
Most likely, we start with training a model based on a random forest algorithm. If we find out that it works badly (for example, if we get less than 80% of accuracy) then we would like to try using some neural network algorithms as it seems to find out the features by itself and probably will be a good choice in case we run out of ideas on what to do. Maybe a neural network model could be especially good after fine-tuning some existing model.

Evaluation:
As this is a classification task, we will use accuracy to evaluate our outcome.
If we train a neural network model then we could use a sparse categorical cross-entropy (we can't use the binary version of it as we have 3 output classes)

Expected challenges:

- The semantics of some features are lost (e.g. marital status represented by values from 1 to 6) and this can make features engineering difficult.
  Solution: We cannot really solve this problem because the data is given to us in Kaggle.
- There are 37 features and we don't know which ones will be useful. To solve this, we can try combinations of different features to figure out the useful ones.
  Solution: We could also try to apply PCA or UMAP to extract the most important features.
- Code sharing between teammates may be difficult as .ipynb documents contain outputs that will be messed up during merge requests and will produce a lot of merge conflicts.
  Solution: After some googling it seems there is a pre-commit hook for Jupyter notebooks that clears the output before a commit.
- It is difficult to split the workload as data preprocessing, training a model and model evaluation are mostly sequential processes.
  Solution: We try to split the workload in such a way that there are no blockers but this requires good communication and following the internal deadlines.

Resources and tools:

- Data preprocessing, model training and evaluation: pandas, numpy, scikit-learn
- Visualization: plotnine, matplotlib.
- Team collaboration: Slack, git, Google Docs, team-work rooms in Delta.
- HPC's JupyterHub when we run out of GPU's quota in Google's Colab while training a neural network model.
- Fast.ai could be a good tool for getting pre-trained models that could be fine-tuned if we decide to go with a neural network solution.

Some other ideas:

- There is a thesis that addresses a similar problem, probably we should get some ideas from there: https://comserv.cs.ut.ee/ati_thesis/datasheet.php?id=77245&language=en.
- There are 3 classes: graduate, dropout and enrolled. We will look at the data to determine if there is an easy way to separate enrolled people from the people who have dropped out or graduated. Maybe people who are still enrolled have some data missing for example. If this is the case then we can use binary classification to differentiate between graduates and dropouts.

Questions for further guidance:

What would be the best option to choose the model? To try many models at the beginning and just choose one based on the results or try one model, do our best and switch to another if the model we chose does not seem to be good enough?

Milestones and timeline:

WEEK	**DATE**	**MILESTONES OR PLANNED ACTIVITIES**
Week 4	September 30	Team formed
Week 5	October 7 *(teams formed)*	Scheduling first meetup Setting up a Github repo Choosing a project's topic
Week 6	October 14	Look into data First meetup Plan done
Week 7	October 21 *(project plan deadline)*	Submit plan Prepare an .ipynb document's template Data exploration Read thesis on the same topic (link above)
Week 8	October 28	We have figured out and implemented solution (for example pre-commit hook) to avoid unnecessary merge conflicts Visualize data to know outliers, missing values or values that do not make sense - more deep data exploration Baseline submission (randomforest trained on unprocessed data)
Week 9	November 4	Preprocess data Train basic random forest classifier on preprocessed data and tune hyperparameters
Week 10	November 11	Do features engineering base Make first somewhat good submission Prepare slides for intermediate presentation
Week 11	November 18 - 20 *(intermediate presentations)*	Presentation slides created Do an intermediate presentation
Week 12	November 25	Evaluate if we should change model and use deep neural networks Try to apply some existing neural network architectures in our project
Week 13	December 2	*Buffer week*

		In case of neural network model: Evaluate how good the NN model is and try using fine-tuning if training from scratch does not give the expected results
Week 14	December 9	Final submission into Kaggle Results analysis and interpretation Code cleanup Prepare slides for a final presentation
Week 15	December 16 - 18 *(final presentations and team evaluation)*	Presentation slides created Do a final presentation Team evaluation