COSC 2670/2732 Practical Data Science with Python

Project Assignment 3, Semester 2, 2021

Marks : This assignment is worth 35% of the overall assessment for this course.

Due Date : Wed, October 27 2021, 11:59PM (Week 14), via `canvas`. Late penalties apply. A penalty of 10% of the total project score will be deducted per day. No submissions will be accepted 5 days beyond the due date.

Objective

The key objectives of this assignment are to learn how to train and evaluate a non-trivial machine learning model under tight time constraints. The task is estimating a user review score in a recommendation system. You will be given a set of training features that contain a label which is the reviewer score for a specific alcoholic beverage between 0 and 5, but on a real scale and not an integer scale. Apologies in advance if you are not a consumer of alcohol for many good reasons – it just happens to be a reasonable test collection to use for our assignment. Most movie reviews or item reviews work similarly to this problem, but may not have as many features available. Your task is to research this problem, find a suitable solution, train a model, and produce a result file from a training set that will be scored using standard evaluation measures in Recommendation Systems.

If you are unfamiliar with Recommendation Systems, you might want to review the lectures between week 9 to week 12, where it is covered from multiple perspectives. There is a great deal of data provided for you to build your experiments, and you may decide to only use parts of what is provided – this is entirely up to you.

Provided files

The following files are provided:

- **A3.pdf** : This specification file.
- **train.tsv** : A large file of labeled beverage reviews suitable for training.
- **header.tsv** : The names of the columns for the val, train, and test sets.
- **val.tsv** : A predefined validation set you should use to tune and test your model. This should make it a little easier to get your model setup and tested.
- **test.tsv** : The holdout set that you will be used by us
- **header.tsv** : The names of the columns for train, val, and test.
- **features.tsv** : This contains several useful features from the collection. Not all must be used, but you may use some of them depending on your solution. There is a `RowID` mapping you can use to find the features for any row in the files above, which also have the `RowID` as the first column.

Feature	Description
RowID	The descriptor you can use to map to addition features.
BeerID	The first key feature. This is the "item" in item recommendation.
ReviewerID	The second key feature. This is the "user" in user-based recommendations
BeerName	The name of the beer
BeerType	The type of beer
Label	This is really the third key feature. It is a rating from 0 to 5 and what you need to predict.
RowID	Same as above. Used to map feature rows to core data rows.
BrewerID	The maker of the beer.
ABV	Alcohol by volume of the beer.
DayofWeek	Day of the week the review was submitted.
Month	Month the review was submitted.
DayofMonth	Day of the month the review was submitted.
Year	The year the review was submitted.
TimeOfDay	The time of day the review was submitted.
Gender	Gender of the reviewer (if known).
Birthday	The birthday of the reviewer (if known).
Text	The clean parsed text of the review.
Lemmatized	The Lemma equivalent of the text.
POS_Tag	The part-of-speech tags for the text.

Table 1: Features and descriptions.

- **header-features.tsv** : The names of the columns for the feature file.

  The A3.pdf file is on canvas, and all of the data files you can download using the URL below called `A3data.zip`. *Be Warned* : The zip file is 516MB. Sorry about that but I wanted to give up as much data as I could, and you decide what looks like it might be valuable as a feature, or use it to create your own (always a good idea). In case you need it, the MD5 of the zip file on the server is: `806b771876cd366b847595f0aac740c7`. I will let you know on canvas if this changes for some reason.

The Features Provided

Rules of the game

You are allowed to use any python library you like to solve the problem. We have seen many examples of algorithms that can be used to solve this problem and you should be able to use the surprise library pretty easily now.

If you want to use a deep learning library and colab, contact me and I *may* be able to provide a small amount of credit Google did give me a few credit codes but it is nowhere close to enough to give to everyone – so you'll have to convince me you will use them wisely and really need them because of the algorithm being used. I only have enough for about 10% of you. Sorry. I did try to get as many as I could, but that is all Google would give me right now. You can still use a GPU / CPU for free on colab as we showed in the Lectorial last week, and you can use a local one if you happen to have an NVIDIA GPU at home for gaming, or use a CPU for many deep learning algorithms, it may be a bit slow if you you do. You may also be able to apply for your own through

Google or AWS if you look around – both companies often provide them to students if you look at their educational sites. You can absolutely solve the problem without them, but if you use something really state-of-the-art, they do help (if you know how to use them).

We will discuss a trick or two that can be used to combine "weak" learners to get high quality results. So come to the lectorials if you want to learn more.

Just as last time, we will need you to ensure your environment is reproducible, so the correct way to do this is to create an anaconda environment for a specific version of python (I strongly suggest it be 3.8, install any packages you need using pip (**not** anaconda unless you use conda-forge and you can clearly document these to reproduce them), and then generate a requirements.txt file to include with your submission. So, something like:

```
conda create -n SXXXXXX python=3.8
conda activate SXXXXXX
pip install pandas numpy scikit-learn
pip freeze > requirements.txt
```

In a README[.txt/.md] you may have:

```
conda install -c conda-forge scikit-surprise
```

Be very careful with conda installs. A few simple ones should work no problem but it is notoriously difficult to reliably use only conda package installations across platforms, which is not reproducible for me (on MacOS) or a marker on Ubuntu. So I strongly encourage you to use `pip` when you can because it is pretty reliable across systems – assuming you capture all the packages and versions in the requirements file you create.

As usual, create a new environment you can start in Anaconda using "`conda activate SXXXXXX-A3`" and exit from using "`conda deactivate`" and try to use Python 3.8. If you have to use an older version of python, clearly document it in the readme with instructions to create the environment, just like we do each week in any lectorial code released on canvas.

Rubric

The marking will be defined as follows:

- Making your environment reproducible (5/35 marks). So heed my suggestion above. If you do not submit a correct requirements file and we cannot reproduce your results, you will lose marks.
- A write-up of your methodology and model decisions. This should be no more that 5 pages single column, 11pt font. We will provide an exemplar template to help you know what you cover – it is basically just like A2. You can base your experiments on the pre-defined validation set which has labels. You should very carefully compare at least three algorithms and at least one should be high quality. (10/35 marks)

- The remaining marks (10/35) are based on effectiveness. Similar to A2, we will define "Leaderboard" like ordering of all of the runs submitted (up to five each). Then, based on the rubric, you will get a score between 0 and 10. Don't panic, we won't give 4 or less unless none of the runs you submit can beat a very simple baseline algorithm. The scores will be based on model complexity and effectiveness will be measured using Mean Average Error (MAE). It should be relatively straightforward to beat the simple baseline. If you cannot, something went terribly wrong in your modeling and prediction. The highest quartile will require a relatively sophisticated one (or really good methodology and technique). You can use the data provided to do some sort of feature engineering on your own (and this often helps). You can submit up to 5 runs, all of which will be scored, and the best one will be used. This is a very simple and well-known measure, so you should be able to put up a solution that is good pretty quickly. Then you can spend your remaining time trying to get at least one great model working on the training / validation split provided.

The test prediction file should be two tab separated rows, rowid and your prediction between 0 and 5. These will look something like this:

RowID	**Prediction**
T1	4.22
T2	1.50
T3	4.55
T4	3.11
T5	2.50
T6	5.00
T7	1.00
T8	2.75
T9	4.90
T10	3.50
T11	3.00

Table 2: Predictions between 0 and 5 by Row.

Test Collection

You can download the two data files from:
`http://wight.seg.rmit.edu.au/jsc/A3/A3data.zip`

Submission

All submissions must be made through Canvas. A link will be provided within canvas, under the Assignments tab for submissions. Assignments submitted through any other method will not be marked. To submit your files, create a top level directory which is your student number. Inside that folder, there should be exactly like the zip file provided to you. You **should not submit the data**. For example, if your student ID is S101010, when I unzip the file `S101010.zip`, I would see the following:

```
$ unzip S101010.zip
Archive:  S101010.zip
   creating: S101010/
  inflating: S101010/Readme.txt (or .md)
  inflating: S101010/A3.py
  inflating: S101010/A3.pdf
  inflating: S101010/requirements.txt
  inflating: S101010/A3-[1,2,3,4,5].tsv
```

In the above, each of these is:

NOTE 3 : There is a discussion group available in the course canvas. Please do not post code snippets in this forum. Do ask questions if you have them! We will do our best to answer them as quickly as we can.

Plagiarism Warning

University Policy on Academic Honesty and Plagiarism: It is University policy that cheating by students in any form is not permitted, and that work submitted for assessment purposes must be the independent work of the student concerned. Please see the RMIT policy for more details: `http://rmit.info/browse;ID=sg4yfqzod48g1`. **THIS IS NOT A GROUP PROJECT.** Students are reminded that this assignment is to be attempted individually. Plagiarism of any form will result in zero marks being given for this assessment, and can result in disciplinary action. We routinely use plagiarism software on projects! Please, please don't do it. Be aware that paying someone on a coding site to do it for you is a form of plagiarism. If you are submitting work that is not your own, regardless of how you got it – you are in breach of this policy.

Extension Policy

Individual extensions will **NOT** be considered or granted by the PDS team. We are not monsters, but doing things on time matters. We consider it a core component of the assessment. If we decide to extend the deadline, the only fair way we can do this is to extend it for *everyone*, and that may not be possible given the the size of the course and university policies. We have set deadlines as late as we can in order to meet the university timelines we cannot control. So be early and not late. If you procrastinate and suddenly have other priories, you will be in a real bind, and that is not a valid reason for an extension.
If you have suffered a personal tragedy or illness, there is a University process in place to grant extensions. Our preference is that you go this route if you must, as they have very clear criteria to grant exemptions. For more information about applying for Special Consideration, see the rules and regulations at `http://www.rmit.edu.au/browse;ID=b1wqvnwk8aui`.

Getting Help

Come talk to us. Email us. Use the discussion board. Ask a question in a lectorial or a practical. There is help available if you need it.