CPSC 383: Explorations in Artificial Intelligence and Machine Learning

Assignment 3: Machine Learning, Neural Networks, Tensorflow

Weight: 15%

Collaboration

Discussing the assignment requirements with others is a reasonable thing to do, and an excellent way to learn. However, the work you hand-in must ultimately be your work. This is essential for you to benefit from the learning experience, and for the instructors and TAs to grade you fairly. Handing in work that is not your original work, but is represented as such, is plagiarism and academic misconduct. Penalties for academic misconduct are outlined in the university calendar.

Here are some tips to avoid plagiarism in your programming assignments.

1. Cite all sources of code that you hand-in that are not your original work. You can put the citation into comments in your program. For example, if you find and use code found on a web site, include a comment that says, for example:

   ```
   # the following code is from
   https://www.quackit.com/python/tutorial/python_hello_world.cfm.
   ```

   Use the complete URL so that the marker can check the source.

2. Citing sources avoids accusations of plagiarism and penalties for academic misconduct. **However, you may still get a low grade if you submit code that is not primarily developed by yourself. Cited material should never be used to complete core assignment specifications. You can and should verify and code you are concerned with your instructor/TA before submission.**
3. Discuss and share ideas with other programmers as much as you like, but make sure that when you write your code that it is your own. A good rule of thumb is to wait 20 minutes after talking with somebody before writing your code. If you exchange code with another student, write code while discussing it with a fellow student, or copy code from another person's screen, then this code is not yours.
4. **Collaborative coding is strictly prohibited**. **Your assignment submission must be strictly your code**. Discussing anything beyond assignment requirements and ideas is a strictly forbidden form of collaboration. This includes sharing code, discussing code itself, or modelling code after another student's algorithm. **You can not use (even with citation) another student's code.**
5. Making your code available, even passively (e.g. online repository accessible to other students), for others to copy, or potentially copy, is also plagiarism.
6. We will be looking for plagiarism in all code submissions, possibly using automated software designed for the task. For example, see Measures of Software Similarity (MOSS - https://theory.stanford.edu/~aiken/moss/).
7. Remember, if you are having trouble with an assignment, it is always better to go to your TA and/or instructor to get help than it is to plagiarize. **The most common penalty is an F on a plagiarized assignment.**
8. For assignments **limited use of generative AI in writing assistance is acceptable. For example, grammar suggestion, or code suggestion tools for programming**. **Programming or text that is largely generative AI produced is not allowed**. Learners are ultimately accountable for the work they submit. Use of AI tools must

be documented in an appendix for the assignment. The documentation should include what tool(s) were used, how they were used, and how the results from the AI were incorporated into the submitted work. Failure to cite the use of AI generated content in an assignment will be considered a breach of academic integrity and subject to Academic Misconduct procedures.

Late Penalty

For late individual assignments, those submitted within 24 hours of the initial deadline will receive 10% off, and within 48 hours will receive 20% off. After 48 hours, no late assignments will be accepted. -10% of 20 marks is -2 marks. -20% is -4.

Goal

We're going to explore property values in the City of Calgary. We will be using a version of the dataset from https://data.calgary.ca/Government/Current-Year-Property-Assessments-Parcel-/4bsw-nn7w/about_data that I've already pre-processed to reduce down for you to use. We are interested in doing a regression to predict the property values of a land description.

You can find some explorations of a similar concept of dataset on California property values https://inria.github.io/scikit-learn-mooc/python_scripts/datasets_california_housing.html This description is not using a neural network or tensorflow like we will so you can ignore their code. However, the visualizations can be an interesting exercise to observe when contemplating your decisions for how to complete your regression. The City of Calgary data will have some similar conclusions that longitude and latitude are important in property values but so will additional categories in our dataset like year of construction, property type, community, and square footage.

For this assignment, you must write your neural nets using the Tensorflow + Keras libraries with Numpy. You may also use Matplotlib and Pandas libraries if you choose to do so (although not necessary), and anything in the Python standard library is fair game. All of these packages were set up in a previous tutorial worksheet. **Using any other libraries is not permitted and will result in a grade of 0 on the assignment.**

Technology

Python 3.13, Tensorflow 2.20

Submission Instructions

You must submit your assignment electronically using **D2L**. Use the Assignment 3 dropbox in **D2L** for a final codebase electronic submission. In **D2L**, you can submit multiple times over the top of a previous submission. Do not wait until the last minute to attempt to submit. You are responsible if you attempt this, and time runs out. Your assignment must be completed in **Python 3**.

Description

The dataset is available as **yyc_pv.csv** on the course website. I've reduced the original dataset provided by the city and preprocessed out string formatting challenges, and missing values so the dataset should be rather clean. The dataset is rather larger as well, almost 600k lines. You may want to consider making a smaller version of only a portion of those lines when you are initially working on your model setup so that the size of the dataset doesn't slow you down and then swapping in the full dataset later.

The spreadsheet contains the following columns:

- ADDRESS: Civic address (text)
- COMM_CODE: Community code(text,category)
- COMM_NAME: Community name (text)
- ASSESSMENT_CLASS: Predominant assessment class code (text,category)
- ASSESSMENT_CLASS_DESCRIPTION: Description of predominant assessment class (text)
- YEAR_OF_CONSTRUCTION: Account AYOC (Actual year of construction) (int)
- PROPERTY_TYPE: Account property type: (text,category)
    - LO = Land only;
    - LI = Land and Improvement
- LAT: Latitude of one corner of property polygon (float)
- LON: Longitude of one corner of property polygon (float)
- LAND_SIZE_SM: Account assessable land area square meters (float)
- LAND_SIZE_SF: Account assessable land area square feet (float)
- ASSESSED_VALUE: Total assessed value of the property (int)

The main part of your work for this assignment will be deciding how to encode these features as vectors. Your goal is to determine a predicted ASSESSED_VALUE based on the other factors being provided in the other columns.

You will have to decide for yourself how to handle the data, such as which features to include and how to encode them. As you do this, write up reasoning for your choices. For example, how did you choose to encode different features and why? If you are discounting certain columns, why did you choose to do so? This information will be the first section of your **report.pdf** file.

You will also be responsible for dividing your data into training and testing data according to a 5:1 ratio. Note that yyc_pv.csv was not ordered in any way but taking the provided City of Calgary ordering after removal of some entry rows, so you'll have to shuffle your dataset before you make this division. The built-in random.shuffle function in Python might be helpful for this purpose. Numpy also has a shuffle function if you prefer to use that.

This is Assignment three, so you will be creating three versions of your neural net.

The first version should demonstrate underfitting, the second version should demonstrate overfitting, and the third version should try to achieve the lowest **mean absolute error** that you can. There will be no exact value given because depending on how you split your data your test/train values will vary between students. I think sub-300k is a reasonable **mean absolute error** to aim for on the complete 600k dataset**.**

To do version 1 and 2. Please reduce your dataset to 1/1,000 the size. To do this make a version of dataset that is only 600 entries from the provided one. You'll submit this as yyc_pv_reduced.csv when you submit your assignment.

You can create your model versions by adjusting any combination of the following:

- Number of layers in your neural net
- Number of neurons in each layer
- Which features from yyc_pv.csv you are including in your data
- Number of epochs
- Regularization parameter (can be added at different layers)
- Learning rate
- Activation functions

For each version, you should have a short section in your report describing the structure of your neural net. Include which choices you made for each of the values above (if you are using the default values just indicate that). You should also include a screenshot of the console output showing training **mean absolute loss** for each training epoch, and then the final output showing testing **mean absolute loss**. After this console output, indicate how you can tell that your neural net is underfitting or overfitting in that example, or whether it achieves the desired performance.

Finally, you will submit the code for your last version (the one with the lowest **mean absolute error**) to D2L with your report. Make sure that your code is well-commented and clean with your name and student ID at the top, as there will be marks allocated for code quality. There is always variability on different training runs because of the random initializations, but this last version of your code should be able to achieve your described **mean absolute error** on the majority of its executions.

Notes:

- When reduced to 600 entries the dataset is quite small, so one of your biggest challenges will be overfitting. However, when you do your third model you are free to make arguments if these changes are required anymore when you 600k entries instead.
- If you are having time issues with the 600k dataset there are some things you can try. The first is batching data so that training updates are done after multiple inputs are seen. Another is that you should have no issues if you drop it to 60k for this assignment.

Just make sure to mention that in your report and indicate the step you took to make the 60k reasonably representative.

- Feel free to use code from tutorials as a starting point. You should be able to get some initial functions from there for reading in and processing data, although you'll have to adjust those to work with this dataset.

Grading

Grading will be based on Data Processing (4), Network Properties (3), Model Quality (9), and Presentation (4).

For Data Processing we are looking for you to handle discrete and numeric data reasonably.

For Network Properties we are looking for your choice of features to use to be reasonable and justified, and for the network design choices list as options to be reasonable choices and justified.

For Model Quality we are looking at each of the 3 models and your report to see a clear description of your model, that the results match what would be expected, and evidence of the desired outcome (underfit, overfit, final 95%)

For Presentation we are looking for good code and a quality report.

As a bonus if you can an image and the code that produces a map like visualization of the city of Calgary using matplotlib that shows property values based on latitude and longitude you will receive 1 bonus credit grade.

Submit the following using the Assignment 3 Dropbox in D2L

1. **main.py (or main.ipynb)**
2. **report.pdf**
3. **yyc_pv_reduced.csv**