Lab 4 - Cloud Data, Stat 215A, Fall 2024

These are the **lab4-specific** instructions. Please also see the general lab instructions in **lab-instructions.pdf**.

Contents

1 Submission

Push a folder called `lab4` to your `stat-215-a GitHub` repository by **23:59 on Friday November 15**. Although this is a group project, each group member must submit the group's report and code in their own repo. I will run a script that will pull from each of your GitHub repositories promptly at midnight so take care not to be late as late labs will not be accepted.

The 12-page limit is not in place for this lab. The page limit for this lab is 20 pages. The bibliography and academic honesty sections don't count towards this limit.

Follow the general lab instructions in stat-215-a-gsi/disc/week1/lab-instructions.pdf for more details. Please do not try to make your lab fit the requirements at the last minute!

I have provided a `.tex` template as a guideline for the writeup. Since the template is intended to make grading easier, please do not deviate from it significantly without good reason. (I have not provided a `.ipynb` template because I think collaboratively writing this report in a notebook is a bad idea). In your `lab4` folder, please provide everything (except the data) that is needed for someone else to be able to compile your report and receive the exact same pdf.

Special coding instructions

For any autoencoders you train, please save the trained model (as a model checkpoint) in your `results` directory. It should be a `.pt` file. If you use YAML files to configure training runs (hint: you should), please include them somewhere in your `code` directory.

2 Academic honesty and teamwork

Academic honesty statement

You can use your statement from lab1 or lab2 or lab3.

Collaboration policy

Within-group: In a file named `collaboration.txt` in your `report` directory, you must include a statement detailing the contributions of each student, which indicates who worked on what. After the labs are submitted, I will also ask you to privately review your group members' contributions.

With other people: You are welcome to discuss ideas with me. Please avoid discussing specific ideas with other groups, as I don't want to receive 5 of the same lab. If you do consult with students in other groups, you must acknowledge these students in your lab report.

LLM usage policy

You are allowed to use LLMs (ChatGPT, GitHub Copilot, etc.) to help with the **coding** for this lab. If you use an LLM to help with coding, you must do the following:

- At the end of the report in the appropriate section (see template), state which tool(s) you employed and for what parts of the coding.
- Describe the nature of your interactions with the LLM. e.g. did you ask ChatGPT to code things in English? Did you use Copilot by writing a comment then tab-completing individual lines? Did you get Copilot to write whole functions? Did you mostly use LLMs for debugging? For writing comments? etc.
- If you used a prompt-based LLM like ChatGPT, add a link to your ChatGPT session, e.g. `https://chatgpt.com/share/f7a9094a-9eda-47a0-9103-4aa6345284b0`.

You are also allowed to use an LLM for polishing the writing of your report at the sentence or paragraph level. As an example, you **may**, for example, plug in a sentence or paragraph to ChatGPT and ask it to be checked for grammar and general sentence flow. You **must not**, for example, ask ChatGPT to expand some bullet points into a paragraph, or have it completely rewrite a paragraph, or plug in your whole report, or make something "sound better," etc. The report should still be in your "voice" and not in the (very obvious) ChatGPT voice. And you **must not** submit any writing directly by an LLM without reviewing/editing it yourself. Please ask me if you are unsure if something is allowed.

Any LLM usage for writing must be reported in the LLM usage section of your report—e.g. "I used ChatGPT to revise the grammar of section 2.1." It must be clear to me how much you used LLM tools.

3 What you need to do

The goal of this lab is the exploration and modeling of cloud detection in the polar regions based on radiances recorded automatically by the MISR sensor aboard the NASA satellite Terra. You will attempt to build a prediction model to distinguish cloud from non-cloud using the available signals. Your dataset has "expert labels" that you can use to train your models. When you evaluate your results, imagine that your models will be used to distinguish clouds from non-clouds on a large number of images that won't have these "expert labels".

The data can be found in `image_data.zip` which contains three files: `image1.txt, image2.txt, image3.txt`. Each of these files contains one "picture" from the satellite. Each of these files contains 11 columns described below. NDAI, SD and CORR are features based on subject matter knowledge. They are described in the article `yu2008.pdf` in the `lab4/documents` folder. The sensor data is multi-angle and recorded in the red-band. More information on MISR is available at `http://www-misr.jpl.nasa.gov/`

0	y coordinate
1	x coordinate
2	expert label (+1: cloud, −1: not cloud, 0: unlabeled)
3	NDAI
4	SD
5	CORR
6	Radiance angle DF
7	Radiance angle CF
8	Radiance angle BF
9	Radiance angle AF
10	Radiance angle AN

The following are the instructions for the three main parts of the lab. As usual, please carefully document your analysis pipeline, justify the choices you make, and place your work within the domain context. You should still have an introduction and conclusion section in your report.

EDA

1. Plot the expert labels for the presence or absence of clouds, according to a map (i.e. use the X, Y coordinates).
2. Explore the relationships between the radiances of different angles, both visually and quantitatively. Do you notice differences between the two classes (cloud, no cloud) based on the radiances? Are there differences based on the features (CORR, NDAI, SD)?

Feature selection and engineering

1. Some of the features might be better predictors of the presence of clouds than others. Assuming the expert labels are the truth, suggest three of the best features, using quantitative and visual justification. Be sure to give this careful consideration. Note: you don't necessarily have to use this small subset of features in your classification models.
2. Based on this information, can you engineer any new features? Maybe some that use a patch of data around a point?
3. I have provided an autoencoder which takes small patches of points and encodes them into a low-dimensional space, as well as some code to use it to get embeddings of points. How good are the autoencoder's features? How do they compare to the original features?
4. Try to train an autencoder as good as or better than mine, with the same or a different architecture. Can you get an embedding which better separates the cloudy and non-cloudy pixels? (Note: the autoencoder should have the same input/output dimensions as the one I have provided).

Data splitting

Before you begin modeling, split your data into training, validation, and test sets. (Or training and test, with some cross-validation scheme over the training set). Justify your choice of split based on how it reflects the challenges with possible applications of your model.

Note: I (and maybe you) trained an autoencoder in the section above before doing the data split. If you use the autoencoder-derived features in the downstream models, either retrain it on an appropriate training set or discuss the limitations of using an autoencoder trained on the full dataset. For instance, will this make your assessment of your models' generalizability too optimistic? Is this worse than manually selecting or engineering features based on the full dataset?

Modeling

1. Develop several (i.e., at least three) 0-1 classifiers for the presence of clouds, using your best features, or others, as necessary. Provide a brief description of the classifier, state the assumptions of the classification models if any, and test if the model assumptions are reasonable.
2. Assess the fit for different classification models e.g. using cross-validation, AIC, and/or the ROC curve. Think carefully about how to choose folds in cross validation and/or your training/test splits.
3. Pick a good classifier. Show some diagnostic plots or information related to convergence, parameter estimation, and/or feature importance.
4. For your best classification model(s), perform some post-hoc EDA. Do you notice any patterns in the misclassification errors? Again, use quantitative and visual methods of analysis. Do you notice problems in particular regions, or in specific ranges of feature values?
5. How well do you think your model will work on future data without expert labels? Is there a stability analysis, or some test that might give you a good idea of this?