CIS 4190/5190: Applied Machine Learning Fall 2025

Final Project

Release Date: October 27, 2025

1 Introduction

Welcome to the final project for CIS 4190/5190 Applied Machine Learning! This project is designed to be both a structured learning experience and an opportunity for exploration. The project specifications outlined here are **not intended to be fully prescriptive**. Instead, there will be open-ended components both to how you will go about achieving the objectives specified in the description (i.e., **basic components**), and also to the objectives that you set for yourselves beyond what is in the document (i.e., **exploratory components**). We hope that this project offers you the chance to deepen your understanding of machine learning concepts while also challenging you to explore new directions.

Each team (2-3 students) will work on one of two project directions:

A: Image2GPS: Predicting the camera location based on an input image

B: News Source Classification: Classifying the source of a news headline

Once you form teams of 2-3 members, please go to the Gradescope assignment for submitting your team formations and group project track selection by **November 5, 2025**. By **December 15, 2025**, each team will submit their collected data, their trained model(s), and a project report (5 pages, format attached separately).

1.1 Team Formation and Project Track Selection

After you form your group of 2-3 members, your group will fill out a group submission on Gradescope to show:

1. Your team members.
2. Which of the two topics do you want to choose (or others, see section 1.6. exceptions below.

1.2 Basic Components

The basic components of this project is to ensure that you experience and execute each stage of a standard machine learning pipeline, from data preparation through model development and rigorous evaluation, mirroring best practices in practical ML workflows:

1. **Data Collection**: Collect, curate, clean, and split your dataset, ensuring it is ready for modeling and internal evaluation.
2. **Model Design**: Build and iteratively improve your model architecture, making design

choices based on empirical findings.

3. **Model Evaluation**: Evaluate your models using appropriate metrics, select the best one based on performance, and submit it to the leaderboard.

To ensure a shared starting point, the course team will:

1. Provide starter data and skeleton code for both project directions. You can use them as the baseline for your work if applicable.
2. A live leaderboard will be set up, each team must submit at least one entry to the leaderboard to have a sense of how other teams are performing.
3. A portion of your project grade (about 15%) will be awarded for achieving at least a minimum benchmark score on the main problem (specifics and metrics defined in the project documents) and for surpassing a minimum level of performance on the problem.

1.3 Exploratory Components

Here the intention is to try to accomplish something new and exciting that will become part of your "portfolio" on a resume. Lots of freedom, and points mainly for interestingness/creativity. Take risks! This is the part that's really yours! You should aim to create something of value to others, while also being of value for your own education e.g.

- **Code**: implementation of an existing algorithm in a new programming language or using new libraries or with higher efficiency, fixing bugs in an existing codebase
- **Technique**: e.g. improving a neural architecture
- **Analysis**: e.g. asking and exploring a new question to which the answer is not obvious
- **Teaching**: e.g. producing an explanatory blog post on a topic related to your project
- **Application**: e.g. applying a known algorithm to a new domain
- **Datasets**: e.g. collecting a new dataset for a topic

If in doubt, please post on Ed discussions or consult Office Hours!

1.4 Project Report

As part of the project, you will submit a to document your approach, the experiments you conducted, and the insights you gained. Your report must address the following **basic components**:

- Describe the procedure and protocol for **data collection**, what considerations went into these, how you curated or cleaned up your datasets, how you split the datasets for internal evaluations (if you did), and what your final dataset ended up looking like.
- Describe the design considerations for the model, the iterative process through which

you tried to improve those designs, and what your final **model design** ended up looking like.

- Describe the evaluation protocols and results (how you evaluated model iterations, what performance metrics you used, and how you chose models to submit to the leaderboard)

It must also address the **exploratory components**. Some directions (non-exhaustive) are listed below:

- **Code**: Summarize your motivation for code modifications, such as porting to a new language or improving efficiency, and describe your approach, focusing on major implementation choices. Discuss specific challenges encountered and solutions implemented during development. Present benchmarks or comparisons showing the impact of your contributions on performance or usability.

- **Technique**: State the modeling or algorithmic innovation explored, including its theoretical motivation or inspiration from previous literature. Outline the main steps for implementation and validation of the new technique, and highlight any key design decisions or parameter choices. Briefly summarize results from experiments or ablation studies, comparing them to baseline performance.

- **Analysis**: Describe the research question or hypothesis you investigated, and detail the experimental design or analytical methods used to address it. Present your primary findings, supported by visualizations or summary statistics as appropriate. Discuss the insights gained and any implications or limitations that surfaced during your analysis.

- **Teaching**: Specify the teaching resource or artifact you developed, such as a blog post, demo, or presentation, and identify the intended audience. Explain your approach to breaking down complex concepts and making them accessible, including any visual or interactive elements used. Summarize feedback, audience engagement, or lessons learned from the process of creating educational materials.

- **Application**: Identify the real-world scenario or dataset where you applied your model, and note any modifications necessary to adapt your approach for this context. Provide a concise evaluation of model performance or utility in the new domain, mentioning both strengths and challenges encountered. Reflect on the broader applicability and relevance of your solution in practical settings.

- **Datasets**: Explain the motivation for collecting or curating new data and describe your sourcing and cleaning procedures. Give a brief summary of dataset characteristics, such as size, balance, labeling method, or notable attributes, using figures or tables if helpful. Note any difficulties with data quality and the steps you took to ensure the integrity and reproducibility of the dataset.

- **Comparison**: Analyze the pros and cons of State of the Art methods and how your work benefits from them or addresses their limitations. Use empirical results to compare your work with other SOTA to justify your claim. You don't have to outperform the SOTA methods.

1.5 Deliverables and Dates

At the end of the course, by **December 15th**, you will submit the following:

- **Dataset:** The dataset you collected and used in your experiments.
- **Best-performing model:** The model that achieved the best performance based on your evaluation metrics.
- **Project report:** A report (5 pages) that details your approach and findings with the structure detailed in 1.4

1.6 Exceptions

You are strongly encouraged to follow the standard guidelines for team size, project topics, and deliverables. Exceptions are rare and granted only for special cases mainly intended for: If you're performing research and can get your research-advising professor to sign off on an ML-related piece of the problem that you will lead with no help outside the team. To apply for an exception:

- Your advisor must agree to the external or expanded project responsibilities.
- Post a clearly labeled request ("Standard Project Exception Request [Your Name]") on Ed, outlining your intended topic and supporting documentation.
- Be aware that exceptions undergo extra scrutiny receive less support from the teaching team.

2 Image2GPS

2.1 Problem Definition and Motivation

In this project, you will train a regression model to predict GPS coordinates from input images. The first step involves building a dataset that pairs images with their corresponding GPS labels. You will then use this data to train supervised computer vision models capable of estimating the location where each image was captured. This project will provide you with hands-on experience in standard computer vision pipelines and the opportunity to develop practical CV models for everyday use cases.

Problem Definition and Scope: The goal of this project is to build a computer vision model that could predict the GPS locations from any images taken from campus. To make it more feasible, we define the testing region to be on Penn's campus in between the rectangular region from 33rd and Walnut to 34th and Spruce st, as shown in Figure 1.

2.1.1 Performance metrics

In this section we give some details on how will the dataset and the model that you submit be tested on our end. The dataset testing is easier. Basically, whatever dataset you collect and submit should perform similar (i.e. provide non-trivial gains) on the baseline model to the toy dataset that we release. To rephrase, the baseline model should be good enough to provide non-trivial performance on your dataset. For the model that you submit, we will perform inference on the hidden test data and calculate the RMSE loss of your model. The RMSE loss will indicate the mean distance in meters that your model predictions are far from the actual GPS coordinates. This loss will be a factor that decides your position on the leaderboard. Our test data will include cases that try to test your model for robustness against distribution shifts. So, you should try to think and come up with cases that your model might perform bad on and how can you mitigate them. Of course we will not test on images pointing towards the sky ;) Since, you will have the baseline model with you, we expect that the data you will collect will be of high quality and with proper testing you should be able to score well in the data collection component :)

2.2 Representative Data Sample

You can find some examples of test data that will be used for evaluation here. You can use it as a resource for validation, but we do not recommend including it in your training set.

2.3 Code for Training a Baseline Model

You can find how we build a naive baseline method that just slightly outperforms trivial solutions (by only outputting the mean of latitude and longitude) in this notebook.

Essentially, we resize the image to 224 by 224 (**you should also resize the image to 224 × 224**) and normalize it. More importantly, we normalize the outputs by calculating

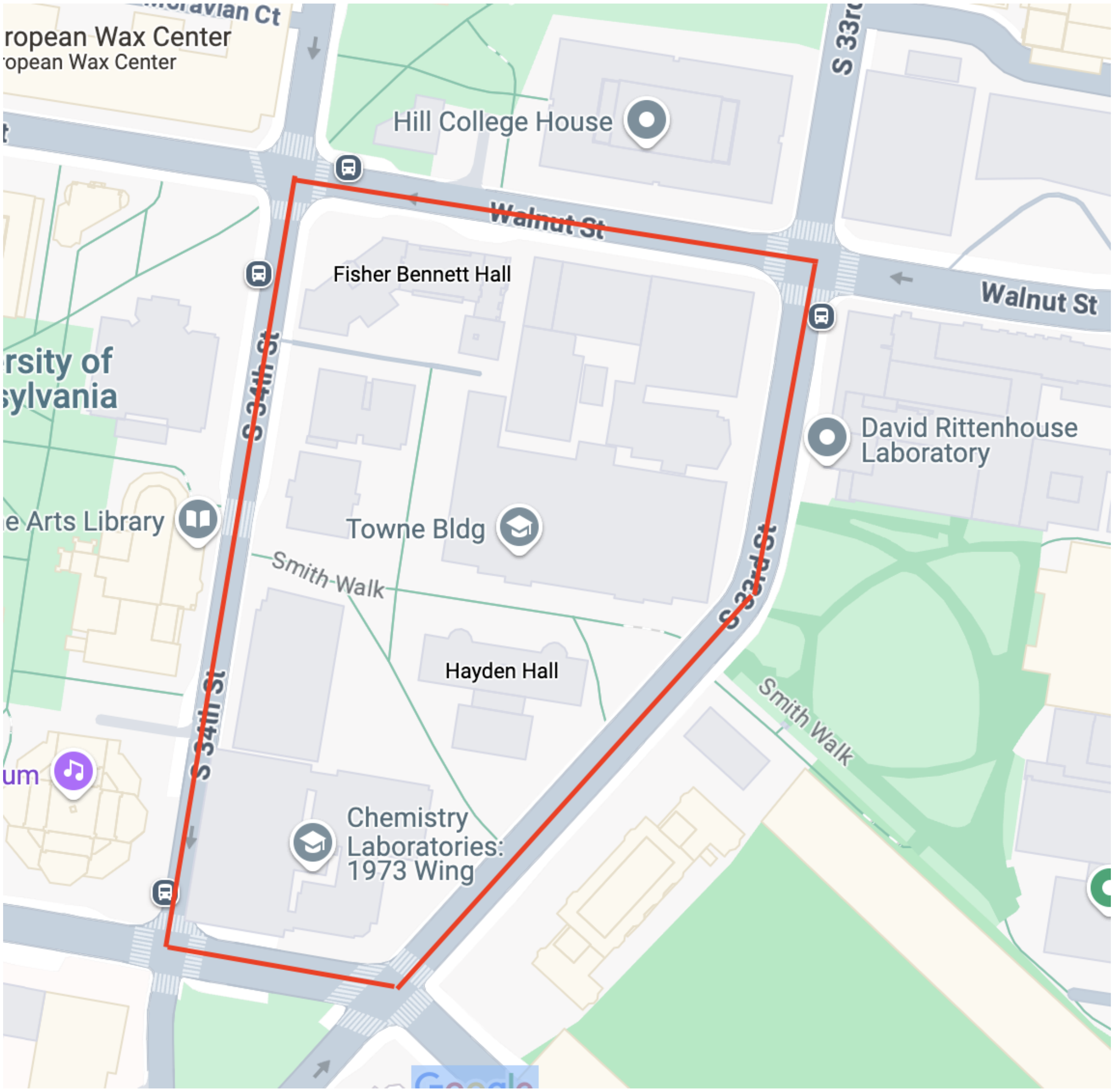


Figure 1: Test region for GPS prediction.

the mean and variance of the latitudes and longitudes (remember to scale it back after prediction).

We use a ResNet-18 as the backbone and modify the output of the last layer to have a dimension of two. We conduct full fine-tuning using MSE loss, the Adam optimizer with a learning rate of 0.001, and a scheduler that decreases the learning rate every few epochs. In total, we run 10-15 epochs to obtain baseline performance. Your model is expected to achieve performance at least as good as this baseline.

A reminder: make sure to set the Colab runtime to connect with GPUs. Note that you are also likely to run out of GPU allocation if you are using a free account.

2.4 Other Miscellaneous Resources

2.4.1 Data Collection

In modern computer vision tasks, models are often trained on large-scale image datasets available online, such as COCO and ImageNet. However, assembling a dataset that effectively supports your specific tasks can be more challenging than it appears. In the initial stage of this project, you will collect an image dataset with GPS labels. You will learn how to gather high-quality images suitable for this task, ensure consistency across different data collectors, and account for factors that may affect data distribution such as lighting and weather.

We will implement a standardized data collection procedure. First, the test dataset will consist exclusively of images taken along walkways, so you won't need to cover every inch of the campus. Second, it's important to note that each GPS location can correspond to multiple views. To ensure comprehensive coverage, you should capture several photos from different viewpoints at the same exact location. During data collection, we usually stand at a point and rotate to take eight pictures from various angles. When taking pictures, avoid zooming in or out, and always hold your phone upright.

2.4.2 Post Process and Extracting GPS Location

You can easily access the GPS location of the image that you captured through the EXIF data of the image. For this you will need to enable settings from your smartphone that allow you to capture/store the EXIF data of the picture that you take. We will give suggested approach for iPhone and Android (Samsung). Note that this may slightly differ for your smartphone but the general idea is the same. For iPhone, you will need to go to (Settings)—>(Privacy)—>(Location Services) and make sure that the application for Camera is given the access. For Android, the default camera application will have settings. You should open settings from inside the default camera app and make sure that the Location/Location tags option is selected. Note, you should always make sure that there is EXIF data collected for the picture that you capture first before starting to collect more images. To help you with this, please find the ipynb notebook. The notebook will extract the EXIF data (specifically, the latitude and longitude) from the image and store it in a CSV file. Feel free to create a copy of the notebook and change it to your requirements. This way you can create the training labels for the image that you capture.

2.4.3 Uploading Data and Model

TBD: details on uploading the data and model will be announced later

3 News Source Classification

3.1 Problem Definition and Motivation

In this topic, you will work to scrape the News Headlines dataset and train on a Binary Text Classification task using news headlines from two prominent news outlets: **Fox News** and **NBC News**. The goal is to create some machine-learning models that classify news based on their headlines. We will provide a baseline model with lower accuracies, 3815 URLs of 3815 news (so 2010 from FoxNews, 1805 from NBC) for you to start Headlines scraping, and a screenshot of the first few lines of hidden test data that we will use to test your models. Your task will be to collect a dataset, process it, and experiment with various models to improve classification performance as best as you can. To show and summarize your improvement, in the final report, you will submit one or several line charts of your models' metrics, including metrics of the baseline model.

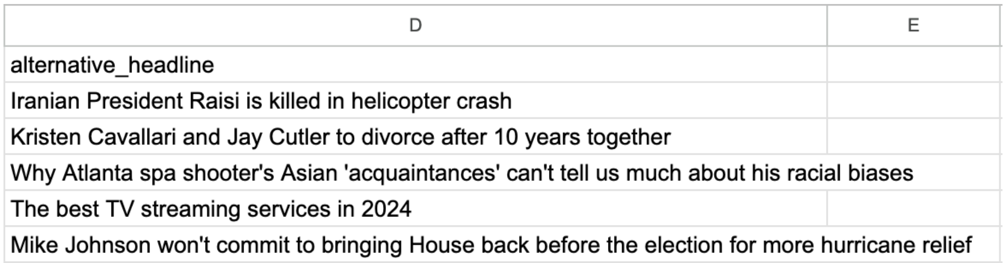
Here is the dataset of links to articles from NBC News and Fox News: Dataset.

3.1.1 Performance metrics

For your final model performance, we will evaluate your models on news headlines scrapped from NBC News and Fox News after the deadline submission. As such, your model will not have access to the testing set when you are training/validating your model. In the sections below, we describe what the data looks like and give you tips on how to clean the data, as well as how to do the webscrapping.

3.2 Representative Data Sample

Figures 2 and 3 are the examples of the test data for NBC and Fox News that we will use.

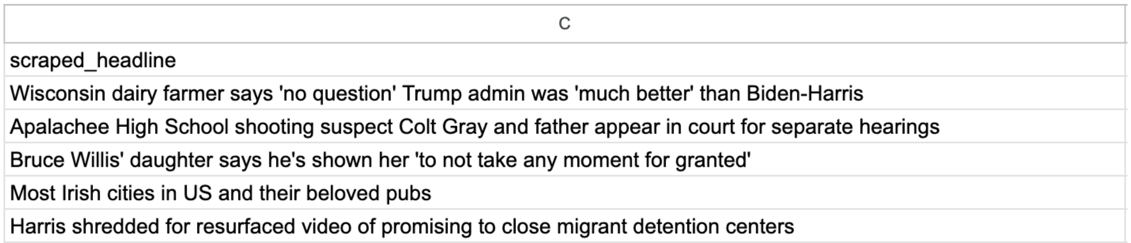


D	E
alternative_headline	
Iranian President Raisi is killed in helicopter crash	
Kristen Cavallari and Jay Cutler to divorce after 10 years together	
Why Atlanta spa shooter's Asian 'acquaintances' can't tell us much about his racial biases	
The best TV streaming services in 2024	
Mike Johnson won't commit to bringing House back before the election for more hurricane relief	

Figure 2: NBC Example Headlines

3.3 Code for Training a Baseline Model

In the following figures, please refer to the explanation and the code of the baseline model to reproduce this model. It should have an accuracy of 66.49% and all other metrics. You should build models than this baseline model.



C
scraped_headline
Wisconsin dairy farmer says 'no question' Trump admin was 'much better' than Biden-Harris
Apalachee High School shooting suspect Colt Gray and father appear in court for separate hearings
Bruce Willis' daughter says he's shown her 'to not take any moment for granted'
Most Irish cities in US and their beloved pubs
Harris shredded for resurfaced video of promising to close migrant detention centers

Figure 3: Fox News Example Headlines

```
# 1. Load the CSV files & Preprocess the data
# csv_file_path = '/merged_news_data.csv'
# news _df = pd.read_csv(csv_filePath)
# ... ...

# 2. Split the data into training and testing sets
# (80% train, 20% test)
# ... ...

# 3. Convert the labels to binary values (0 for 'FoxNews', 1 for 'NBC')
y_train = y_train.apply(lambda x: 1 if x == 'FoxNews' else 0)
y_test = y_test.apply(lambda x: 1 if x == 'FoxNews' else 0)

# 4. Convert the text data to TF-IDF features
vectorizer = TfidfVectorizer(stop_words='english', max_features=100)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# 5. Train a Logistic Regression model
model = LogisticRegression(max_iter=100)
model.fit(X_train_tfidf, y_train)

# 6. Make predictiosn on the test set
y_pred = model.predict(X_test_tfidf)

# 7. Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")
print("Classification Report:\n", classification_report(y_test, y_pred)
   )

## Result
# Accuracy: 0.6649
# Classification Report:
#               precision   recall  f1-score   support
#            0    0.69       0.54      0.60       358
#            1    0.65       0.78      0.71       400
#
```

```
#     accuracy                         0.66       758
#    macro avg     0.67       0.66     0.66       758
# weighted avg     0.67       0.66     0.66       758
```

3.4 Other Miscellaneous Resources

This dataset is a .csv file that contains links to Fox News at the beginning and NBC News at the end. To read this dataset in Python, you can use the `pandas` library. Your task is to web scrap the headlines from these links. For those new to web scrapping, please follow the steps below. The Python libraries that are most helpful are `BeautifulSoup` and `request`. To web scrape, you identify the HTML tabs that contain the text that you want. In the example below, we are scrapping the headline which is in a `<h1>` tag with the class name `"headline speakable."` Here is a sample Python code to web scrape:

```python
import requests
from bs4 import BeautifulSoup

url = https://www.foxnews.com/sports/juan-soto-sends-yankees-world-series-first-time-15-years # example website

response = requests.get(url)
if response.status_code != 200:
  raise Exception(f"Failed to load page:
  Status code {response.status_code}")

# Parse the HTML content with BeautifulSoup
soup = BeautifulSoup(response.text, "html.parser")

title = soup.find("h1", class_="headline speakable")

# title should be the following
# Juan Soto sends the Yankees to the World Series for the first time in 15 years
```

You are also encouraged to webscrap and collect your own headlines from Fox News and NBC News.

3.4.1 Cleaning Process Suggestions

After scraping the headlines, consider further cleaning to make sure your data is ready for training. Some possible operations are listed, but they are not required or no guarantee of performance improvement. You are highly encouraged to explore on your own and incorporate more advanced techniques:

- **Data Cleaning**: Remove unwanted elements such as HTML tags, scripts, and special characters. Address unnecessary spaces, tabs, and newline characters to ensure consistency. Decide how to handle punctuation marks within the headlines.

- **Normalization**: Standardize the text by converting it to lowercase or uppercase. Remove common stopwords that may not add significant value. Apply stemming or lemmatization to reduce words to their base forms.
- **Handling Missing or Incomplete Data**: Detect any missing or incomplete headlines in your dataset. Choose methods to handle these gaps, such as removing incomplete entries or imputing missing values.
- **Consistency Checks**: Ensure all headlines follow a consistent format. Identify and address duplicate headlines to maintain the integrity of your dataset.
- **Data Validation**: Manually inspect a subset of processed headlines for quality assurance. Calculate metrics such as average headline length or word frequency distributions to understand your dataset better.
- **Documentation and Tracking**: Maintain detailed notes on the preprocessing steps applied. Use version control systems like Git to manage changes in your preprocessing workflows.

3.4.2 Uploading Data and Model

TBD: details on uploading the data and model will be figured out later