COMPSCI 589 - Final Project - Spring 2025

Due May 9, 2025, 11:59 pm Eastern Time

1 Instructions

- **This final project should be done in groups of two students**. When making your submission on Gradescope, please do not forget to add all students who worked on this assignment. Do that both when submitting your code and your final report.
- We strongly recommend that you use LaTeX to prepare your submission. The assignment should be submitted on Gradescope as a PDF with marked answers via the Gradescope interface. The source code should be submitted via the Gradescope programming assignment as a .zip file. Include with your source code instructions for how to run your code.
- You may *not* use any machine learning-specific libraries in your code, e.g., TensorFlow, PyTorch, or any machine learning algorithms implemented in scikit-learn. You may use libraries like numpy and matplotlib. If you are not certain whether a specific library is allowed, do ask us.
- All submissions will be checked for plagiarism using two independent plagiarism-detection tools. Renaming variable or function names, moving code within a file, etc., are all strategies that *do not* fool the plagiarism-detection tools we use. If you get caught, all penalties mentioned in the syllabus *will* be applied—which may include directly failing the course with a letter grade of "F".
- The datasets for this project can be found on Canvas.
- The automated system will not accept assignments after 11:59 pm on May 9.
- Notice that you **cannot** use your "free late days" on the final project since May 9 (this project's deadline) is the official last day of classes.

2 Final Project — Overall Goals

- The goal of this final project is to compare the performance of different machine learning algorithms you implemented throughout the semester by testing them on four new, challenging datasets.
- By performing these comparisons, you will gain practical experience in selecting an algorithm (based on a given problem) and optimizing its hyper-parameters so that it works well. In addition, these analyses will give you insight into how to solve novel machine learning problems—problems where no one tells you which algorithms may work best nor how to adjust their hyper-parameters.
- **This project should be done in groups of two students**.
- **Notice that you may not use existing machine learning libraries for this assignment: you *must* use the implementations that you or your partner designed throughout the semester**.
- On each question below, please clearly indicate whose implementation was used; i.e., make sure that you explicitly mention the name of the student whose code was used to tackle each particular dataset.
- We expect the amount of time and effort you put into this assignment will be slightly *less* than that required to solve one homework. The reason is that this final project will not require implementing new algorithms, other than the ones you already implemented during the semester. Also, you will be working in groups and so it should be easier to split up tasks evenly.

- **Alternative: "Custom Project"**. You may replace the tasks mentioned above with a "custom project" that is more closely aligned with your interests. It has to be ML-related and use some of the techniques we discussed in class. For reasons of fairness, this project *cannot* be used as part of any other UMass activity for which you are given credit (e.g., other courses or independent studies). A custom project related to a research project that you are conducting with a UMass professor (or with anyone else) is acceptable if you are not already getting graded for it. If you would like to work on a custom project, please check with the instructor if the custom project you have in mind would be acceptable. If the project is deemed acceptable, you should let the instructor know, *before you start working on the project*, which tasks will be assigned to each student in the group.

3 Datasets

As part of this final project, you will be analyzing four datasets.

3.1 The Hand-Written Digits Recognition Dataset

The goal, here, is to analyze 8x8 pictures of hand-written digits (see, e.g., Fig. 1) and classify them as belonging to one of 10 possible classes. Each class is associated with one particular digit: $0, 1, \ldots, 9$. In this dataset, each instance is composed of $8 \times 8 = 64$ numerical attributes, each of which corresponding to the grayscale value of one pixel in the image being classified. This dataset is composed of 1797 instances.

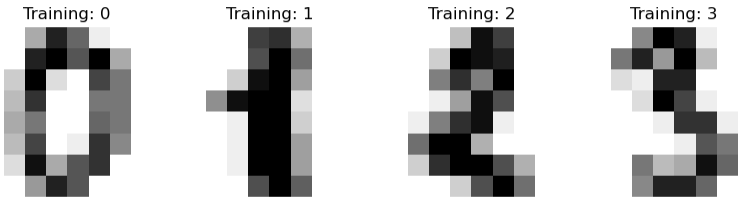

Figure 1: Examples of hand-written digits that you will be classifying.

To access this dataset, and for sample code describing how to load and pre-process it, please visit Scikit-learn's website. Further, please find below (in Fig. 2) a simple example of how to load this dataset; here, the instances are saved in *digits_dataset_X* and their corresponding classes are saved in *digits_dataset_y*.

```
from sklearn import datasets
import numpy as np
import matplotlib.pyplot as plt

digits = datasets.load_digits(return_X_y=True)
digits_dataset_X = digits[0]
digits_dataset_y = digits[1]
N = len(digits_dataset_X)

# Prints the 64 attributes of a random digit, its class,
# and then shows the digit on the screen
digit_to_show = np.random.choice(range(N),1)[0]
print("Attributes:", digits_dataset_X[digit_to_show])
print("Class:", digits_dataset_y[digit_to_show])

plt.imshow(np.reshape(digits_dataset_X[digit_to_show], (8,8)))
plt.show()
```

Figure 2: Example of how to load the Digits dataset from Scikit-learn, and how to show information about a random instance/digit in the dataset.

3.2 The Oxford Parkinson's Disease Detection Dataset

This dataset is composed of a range of biomedical voice measurements from 31 people, 23 of whom are patients with Parkinson's disease. Each row in the dataset corresponds to the voice recording from one of these individuals. Each attribute corresponds to the measurement of one specific property of the patient's voice—for example, their average vocal frequency or several measures of frequency variation. The goal, here, is to predict whether a particular person is healthy or whether it is a patient with Parkinson's. The binary target class to be predicted is *Diagnosis*: whether the patient is healthy (class 0) or a patient with Parkinson's (class 1). There are 195 instances in this dataset. All 22 attributes are numerical. The Parkinson's dataset can be found in this assignment's zip file on Canvas.

3.3 The Rice Grains Dataset

This dataset contains information about 3810 rice grain images. In particular, each rice grain instance is described by 7 numerical morphological features that were inferred from images of different rice grains (e.g., the longest line that can be drawn on the rice grain). The goal is to predict whether a given rice grain belongs to the Cammeo species or the Osmancik species. This dataset can be found in this assignment's zip file on Canvas.

3.4 The Credit Approval Dataset

This dataset contains information about 653 credit card applications. Attribute names and their values were mapped from their original representation to one that is not easily interpretable by humans to protect the confidentiality of the data; that said, they still do contain all necessary information to construct effective classifiers. The goal is to predict whether a given credit card application would be approved or not. Each credit card application is described by 6 numerical features and 9 categorical features. This dataset can be found in this assignment's zip file on Canvas.

⋆ Notice that some of the datasets above include both categorical and numerical attributes. Algorithms such as neural networks (as well as others) require numerical inputs. The standard way of converting categorical inputs to numerical inputs is by using the one-hot encoding technique. For a quick and high-level introduction to one-hot encoding, as well as examples on how to use Scikit's libraries to perform this type of conversion, please visit this website.

4 Experiments and Analyses

For each dataset, you should:

1. Evaluate the performance of at least two algorithms you studied and/or implemented during the semester (e.g., k-NN, Decision Trees, standard Naive Bayes, Random Forests, Neural Networks, etc).[1] You should discuss which algorithms you decided to test on each dataset and why.
2. You should evaluate the performance of each algorithm on a given dataset in terms of its accuracy and F1 score. Use stratified cross-validation with $k = 10$.
3. To obtain the best performance possible, you should carefully adjust the *hyper-parameters* of each algorithm when deployed on a dataset. For example, you may have to experiment with different values of k when optimizing the k-NN algorithm; different values of *ntree* when optimizing a Random Forest; different architectures and regularization parameters when optimizing a neural network; etc.
4. Given the observation above, you should first show, in a table, the performance of each algorithm (on a given dataset) under a few selected hyper-parameters. You should evaluate *at least* 6 hyper-parameter settings. After analyzing the performance of each algorithm under different hyper-parameters, identify the best hyper-parameter setting—that is, the set of hyper-parameters that resulted in the best performance for the corresponding algorithm on a particular dataset.
5. For each dataset, and considering the best hyper-parameter setting for each selected algorithm, construct relevant learning curves and/or graphs. These should be similar to the learning curves/graphs you constructed in the homework relevant to the particular algorithm you are evaluating. For example, if you choose to deploy a Random Forest to tackle one of the datasets, construct a graph relating its performance and the number of trees in the ensemble; if you choose to use a neural network, construct a learning curve showing the value of the cost function, J, as a function of the number of training instances presented to the network. Assuming that you evaluate two algorithms on each of the four datasets, these analyses should result in (at least) 8 graphs. Briefly discuss and interpret these graphs. For example: Why do you think a particular algorithm may be converging to a poor local optimum when deployed on a given dataset? Why do you think a specific algorithm performs better on datasets containing only numerical attributes? and so on.

After conducting the analyses above for each dataset, you should summarize your results in a table. In particular, you should create a table showing, for each dataset, the performance (accuracy and F1-score) of each of the algorithms. Your table should look like Table 1. You should highlight (in gray) the cells that indicate which algorithm had the highest performance—according to each of the metrics—on each dataset. In the example shown in Table 1, for instance, we can see that, when analyzing Dataset 1, Algorithm 1 had the highest accuracy but Algorithm 2 had the highest F1 score.

	Dataset 1		Dataset 2		Dataset 3		Dataset 4	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
Algorithm 1								
Algorithm 2								
Algorithm 3								
...								

Table 1: Example of how the table summarizing your final results look like.

To facilitate the task of creating this table on LaTeX, please visit this website. We have created a template for generating tables similar to the one shown above. This template can be found in the *table_results.tgn* file, provided to you as part of this assignment's zip file. You can load this template on the website mentioned above (File → Load Table) and complete the table with your results.

[1] Evaluating more algorithms will give you extra credit. See Section 5.

5 Extra Credit

There are four ways in which we may receive extra credit on this assignment. If you choose to work on any of the tasks below, please clearly indicate in your report, **in red**, which specific tasks you are working on for extra credit. Describe what you did as part of that task and discuss the corresponding results. Any source code you develop should be included in your Gradescope submission.

(Extra Points #1: 10 Points) You may earn up to 10 extra points if you analyze all four datasets using an additional algorithm. That is, in this case you would be evaluating more than just two algorithms on each dataset.

(Extra Points #2: 10 Points) You may earn up to 10 extra points if you select a new *challenging* dataset and evaluate the performance of different algorithms on it. A challenging dataset should necessarily include both categorical and numerical attributes and have more than two classes; or it should be a dataset where you have to classify images.

(Extra Points #3: 15 Points) You may earn up to 15 extra points if you *combine* different algorithms and construct an ensemble. For instance, you could create an ensemble composed of three neural networks (each with a different architecture) and a random forest. The process of training the ensemble would be as described in class—each algorithm would be trained based on its own bootstrap dataset, etc. The final predictions would then be determined via majority voting. Evaluate your ensemble algorithm on all four datasets discussed in Section 3.

(Extra Points #4: 15 Points) You may earn up to 15 extra points if you implement a new type of algorithm—a non-trivial variant of one of the algorithms studied in class. If you choose to do so, you should evaluate its performance on all four datasets discussed in Section 3.