Course Syllabus for SIADS 542: Supervised Learning Fall 2021

How to Get Help

If you have questions concerning the degree program, encounter a technical issue with Coursera, or issues using Slack, please submit a report to the ticketing system at umsimadshelp@umich.edu.

If you have an issue specific to the Coursera environment, you can also begin a live chat session with Coursera Technical Support (24/7) or view Coursera troubleshooting guides. (you may be asked to log in to your Coursera account).

For questions regarding course content, refer to the **Communications Expectations** section below.

Course Overview

Students will learn how to correctly apply, interpret results, and iteratively refine and tune supervised machine learning models to solve a diverse set of problems on real-world datasets. Application is emphasized over theoretical content. The supervised learning course is an essential part of the core MADS machine learning series: its concepts, algorithms, and evaluation methods are used heavily throughout the following machine learning courses that include: unsupervised learning, deep learning, and machine learning pipelines.

Prerequisites

In order to be successful in this course, you must know how to program in Python and be familiar with the numpy and pandas Python libraries for data manipulation, and matplotlib for plotting.

Instructor and Course Assistance

Instructor: Kevyn Collins-Thompson
Graduate Student Instructor: Yutong Xie

Course Communication Expectations

Slack is the preferred communication tool for this course. If you have questions about course content (e.g. lecture videos or assignments), please make sure to use Slack. Instructor and course assistant response time to Slack messages will aim to be within 24 hours, Monday-Friday.

Please try to monitor the Slack channels for the course regularly.

Personal communication that may involve sensitive information may be emailed directly to the instructor or course assistant. If you email the instructor or course assistant, please include SIADS542 in the email subject. Instructor and course assistant response time to email messages will be within 24 hours.

Office Hours are held on:

- Mondays at 12pm with Kevyn Collins-Thompson
- Fridays at 4pm with Yutong Xie

An additiona**l Enrichment Lecture** will be held on Mondays from 1-2pm EDT using the same Live Events Zoom link as Kevyn's preceding office hour.

Office hour sessions and Enrichment Lectures will be recorded for the benefit of students who are unable to join at these times.

If you would like to meet with Kevyn Collins-Thompson one-on-one, **Click here for Kevyn's appointment calendar link. Time slots are labeled *SI 542/543 ONLY* and are for 20 minutes from 9-10am on Tuesdays.**

Technology Requirements

The course programming will be based on Jupyter notebooks and Python 3.x.

Required Textbook

This course will use the following textbook as a reference and source of examples: Introduction to Machine Learning with Python, by Andreas C. Müller and Sarah Guido (O'Reilly Media)
This text is available free online via the University of Michigan Library:

1. On the Welcome! screen, choose "Select your institution" to open the menu and select the first option "Not listed? Click here."
2. In the Academic email box, enter your U-M email address (in the format: uniqname@umich.edu).

Users can also create an individual account using your U-M email, but don't have to. There is a more detailed description of access options here. (Unfortunately, to add to this there have been some users recently who have reported error messages when trying to login to this database. My general advice for this problem is to try using an incognito browser window and follow the steps above.)

This text is also available for purchase on the O'Reilly website.

Other Textbooks and Resources (Optional)

From time to time I may refer to examples or other content from the classic textbook **The Elements of Statistical Learning** (Second Ed.) by Trevor Hastie, Robert Tibshirani, and Jerome Friedman, published by Springer.

The entire textbook is free and available for online downloading.

For a very useful mathematical background, see the companion web page to the book "Mathematics for Machine Learning". Copyright 2020 by Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. Published by Cambridge University Press.

Learning Outcomes

Here's a summary of some key learning objectives we have (1) for the course overall, and (2) broken down by week.

Course-wide objectives

- Understand how to correctly prepare datasets for use, e.g. feature normalization, stratified train-validate-test splits, etc.
- Understand key families of supervised learning methods at an applied level (key parameters, decision boundary properties, etc)
- Understand how to evaluate and interpret results from scikit-learn estimators.
- Understand how to select an appropriate supervised machine learning method for a given scenario and dataset.
- Understand the tradeoffs inherent in different machine learning methods: speed, accuracy, complexity of hypothesis space, etc.
- Increase awareness of issues of algorithmic bias, transparency, fairness in supervised machine learning applications.

Week 1

- Understand basic concepts of supervised learning.
- Apply k-nearest neighbor classification as an example of supervised learning.
- Understand over- and under-fitting and how to detect and prevent these.

Week 2

- Be able to train and apply regression and classification objects (estimators) in scikit-learn.
- Understand and apply linear and logistic regression, linear and kernel support vector machines.
- Use model selection methods such as cross-validation to tune the choice of model and key parameters.

Week 3

- Understand and apply a wide variety of evaluation metrics to supervised learning scenarios.
- Be able to optimize a classifier for a variety of metrics.

Week 4

- Learn how to apply a Naive Bayes classifier.
- Learn how to apply decision trees and advanced tree-based classifiers like Gradient Boosted Decision Trees.
- Learn how to apply Neural Networks
- Understand what data leakage is, why it's critical to avoid, and how to detect it.

Schedule

Week 1: You'll be introduced to basic machine learning concepts, tasks, and workflow using an example classification problem based on the K-nearest neighbors method, and implemented using the scikit-learn library. This week's assignment has you work through the process of loading and examining a dataset, training a k-nearest neighbors classifier on the dataset, and then evaluating the accuracy of the classifier and using it to classify new data.

Week 2: We will delve into a wider variety of supervised learning methods for both classification and regression, learning about the connection between model complexity and generalization performance, the importance of proper feature scaling, and how to control model complexity by applying techniques like regularization to avoid overfitting. In addition to k-nearest neighbors, this week covers linear regression (least-squares, ridge, lasso, and polynomial regression), logistic regression, support vector machines, decision trees, and the use of cross-validation for model evaluation.

Week 3: We cover evaluation and model selection methods that you can use to help understand and optimize the performance of your machine learning models. For this week's assignment, you will train a classifier to detect spam email, analyze its performance with different evaluation metrics, and then optimize the classifier's performance based on different evaluation metrics, depending on the goals of the detection task (e.g. to minimize false positives vs false negatives).

Week 4: We will cover more advanced supervised learning methods that include ensembles of trees (random forests, gradient boosted trees), and neural networks (with an optional summary on deep learning). You will also learn about the critical problem of data leakage in machine learning and how to detect and avoid it. The final assignment brings everything together: you will design features for, and build your own classifier on, a prediction problem on a complex real-world dataset.

Assignments

Week 1: Review of important numpy operations on vectors and matrices. Classification with scikit-learn: basic concepts based on k-NN classifier. Identify likely under/overfitting scenarios.

Week 2: Regression with scikit-learn. Use and compare different regression methods.

Week 3: Train linear classifiers on a binary problem. Correctly normalize features and perform cross-validation. Create and interpret confusion matrices, ROC curves for a classifier. Perform grid search to find optimal parameters.

Week 4: Apply supervised learning techniques to a real-world dataset, including the methods introduced this week. Examples could include a data leakage scenario. Answer questions on interpreting the results.

Quizzes

Each week will also contain a short quiz to test your knowledge of material in the lectures and readings.

Grading and Course Checklist

I anticipate no major changes to this course grading scheme. However, as the course progresses, I reserve the right to offer additional bonus assessments or make minor adjustments/fixes as required, for any evaluation method in this course. If necessary, any such changes will always be done in a way that maximizes a student's grade across options.

You must complete all assignments and quizzes to get credit for this course.

Course Assignment	**Percentage of Final Grade**	**Passing Threshold**
Week 1 Quiz	5%	80%
Week 1 Jupyter Notebook Assignment	15%	
Week 2 Quiz	5%	80%
Week 2 Jupyter Notebook Assignment	20%	
Week 3 Quiz	5%	80%
Week 3 Jupyter Notebook Assignment	20%	

Week 4 Quiz	5%	80%
Week 4 Jupyter Notebook Assignment	25%	
Total	**100%**	

Late Submission Policy

Important! Please read and understand this section, and if anything is unclear, the instructors are happy to clarify. We realize that, now more than ever, the occasional crisis might mess up your schedule enough to require a bit of extra time in completing a course assignment. Thus, we have instituted the following late policy that gives you a limited number of flexible "late day" credits.

You have **a total of two (2) free late days** to use for any programming assignments and quizzes during the course. One late day equals exactly one 24-hour period after the due date of the assignment (including weekends). No fractional late days: they are all or nothing. As an example, suppose you had two course late days left. Submitting any time within 24hrs of the original due date counts as using the first late day. Beyond that time, submitting any time within the next 24h counts as using the 2nd late day. After that, each additional 24h period accrues a 25% per day penalty as follows:

Once you have used up your late days, there is a 25% penalty for each subsequent 24-hour period after the deadline that an assignment is late. For example, if the due date is 11:59pm Monday, and you have *no* late days left, penalties would be:

Submit before 11:59pm Tuesday:	25% deduction
Submit before 11:59pm Wednesday:	50% deduction
Submit before 11:59pm Thursday:	75% deduction
Submit after 11:59pm Thursday:	100% deduction

You don't need to explain or get permission to use late days, and we will track them for you. We will allocate any late days you have used at the end of the course, after all assignments are submitted, so that we can do the allocation in a way that maximizes your final grade. Note that resubmissions after the deadline will be counted as late submissions.

Please note: Submitting your work on time is very important in this course. The instructional team will periodically reach out to you and ask you about your progress; if you fall behind it may be difficult to catch up, and you will be at risk for not succeeding in the course.

Letter Grades

The grading scale for this course will be as follows:

A+	97%
A	93%
A-	90%
B+	87%
B	83%
B-	80%
C+	77%
C	73%
C-	70%
D+	67%
D	63%
D-	60%
F	0%

Program-wide Information

Help Desk(s): How to get Help

Need help? You may reach out to UMSI or Coursera depending on the type of question you have.

- Degree program questions or general help - umsimadshelp@umich.edu
- Coursera's Technical Support (24/7) - https://learner.coursera.help/

Academic Integrity/Code of Conduct

Refer to the Academic and Professional Integrity section of the UMSI Student Handbook. (access to Student Orientation course required).

Accommodations

Refer to the Accommodations for Students with Disabilities section of the UMSI Student Handbook (access to the Student Orientation course required). Use the Student Intake Form (requires U-M login) to begin the process of working with the University's Office of Services for Students with Disabilities.

Accessibility

Refer to the Screen reader configuration for Jupyter Notebook Content document to learn accessibility tips for Jupyter Notebooks.

Library Access

Refer to the U-M Library's information sheet on accessing library resources from off-campus. For more information regarding library support services, please refer to the U-M Library Resources section of the UMSI Student Handbook (access to the Student Orientation course required).

Student Mental Health

Refer to the University's Resources for Stress and Mental Health website for a listing of resources for students.

Student Services

Refer to the Introduction to UMSI Student Life section of the UMSI Student Handbook (access to the Student Orientation course required).

Technology Tips

We will be using Slack, Zoom, Google Docs, and Google Sheets to facilitate communication. Your own work on the project will be done in Jupyter.

We have created a Jupyter environment for you that is functionally equivalent to SIADS 516, which is a superset of the base MADS environment. You can access that environment via the "ungraded lab assignment" in Coursera. You can use that environment or choose to use any of the environments from courses you have already completed. Alternatively, you can use your own locally installed environment. Another possibility is to use Google Colaboratory, which may facilitate collaboration.

Working Offline

While the Coursera platform has an integrated Jupyter Notebook system, you can work offline on your own computer by installing Python 3.5+ and the Jupyter software packages. For more details, consult the Jupyter Notebook FAQ.