Math 342W / 690.4 / 742
Spring 2024 (6 credits)
Course Syllabus

Adam Kapelner

Queens College, City University of New York

document last updated Sunday 14th January, 2024 9:40pm

Course Homepage	`https://github.com/kapelner/QC_Math_342W_Spring_2024`
Slack Homepage	`https://QCMath342WSpring2024.slack.com/`
Contact	Write to @kapelner on slack in a public channel
Office	KY 604
Lecture Time and Loc	Mon and Wed 5-6:50PM / KY 283
Required Lab Time / Loc	Mon and Wed 7:10-8:05PM / KY061
Instructor Office Hours and Loc	see course homepage
TA / Office Hours	Josh Palter / see course homepage

Course Overview

MATH 342W. Data Science via Machine Learning and Statistical Modeling. 6 hr. lec./lab; 4 cr. Prereq.: MATH 231, MATH 241, CSCI 111 (or equivalent). Philosophy of modeling with data. Prediction via linear models and machine learning including support vector machines and random forests. Probability estimation and asymmetric costs. Underfitting vs. overfitting and model validation. Formal instruction of data manipulation, visualization and statistical computing in a modern language. Writing Intensive (W). Recommended corequisites include ECON 382, 387, MATH 341, MATH 343 or their equivalents.

You should be familiar with the following before entering the class:

- Basic Probability Theory: conditional probability, in/dependence, identical distributedness
- Modeling with discrete random variables: Bernoulli, Binomial
- Expectation and variance

- Linear algebra: Vectors, matrices, transpose, inverse, rank, nullity
- Fundamental programming concepts: basic data types, vectors, arrays, control flow (for, while, if, else), functions

We will review the above *throughout the semester* when needed and we will do so rapidly.

This is not your typical mathematics course. This course will do lots of modeling of real-world situations using data via the `R` statistical language.

Course Materials

We will be using many reference texts and three popular books which you will read portions from. However the main materials are the course notes. You should always supplement concepts from class by reading up on them online; wikipedia I find the best for this.

Theory Reference: It is not necessary to have these two books, but it is recommended. The first is "Learning from Data: A Short Course" by Abu-Mostafa, Magdon-Ismael and Lin which can be purchased used on Amazon. We will also be using portions from "Deep Learning" by Goodfellow, Bengio and Courville that can be purchased on Amazon and read for free at `http://www.deeplearningbook.org/`.

Popular Books: We will also be reading the non-fiction novel "The Signal and the Noise" by Nate Silver which can also be purchased on Amazon. This is *required* — you will have homework questions directly from this book. We will also be reading "Preditive Analytics, Data Mining and Big Data" by Steven Finlay that can be purchased on Amazon and it is also available online from the Queens College library system.

Computer Software: You need your own laptop and it is required to bring to class. We will be using either `R` and/or `Python` both which are free, open source programming languages available for all operating systems. I will be using `R` in class and Amir, the TA will be using `Python` in his office hours. You can hand in the lab assignments either in `R` or `Python` (more on lab assignments later.

- To download `R` go to `http://cran.mirrors.hoobly.com/`. I then recommend the IDE `RStudio` available for free at `https://www.rstudio.com/products/rstudio/download/` this way you can follow the demos in class.
- To download `Python` go to `https://www.python.org/downloads/`. I then recommend the IDE `Jupyter Notebook` available for free so you can follow Amir's Python translations of the class demos. To install Notebook, open up a command prompt and executing `pip install notebook`. After this is installed, you can run `jupyter notebook` in a command prompty which will open up a web browser with the IDE embedded. An alternative method is to install Anaconda using these instructions.

Source Control: You will be expected to use `git` and have a `github.com` account with a repository named `QC_MATH_342`. You will use this repository to submit coding homework assignments (and theory assignments if you use LaTeX).

Book on `R`: We will be making some use of "R for Data Science" by Wickham and Grolemund which can be purchased on Amazon or read online at `http://r4ds.had.co.nz/`.

Books on Python: The following books are recommended: Second Edition of Atomate the Boring Stuff With Python, The Official Style Guide for Python Code, Interactive Python Tutorial and Harvard Universities Quick Python Course.

The 650.4 Section

You are the students taking this course as part of a masters degree in mathematics. Thus, there will be extra homework problems for you and you will be graded on a separate curve.

Announcements

Course announcements will be made via slack in the `#general` channel (not on email). I am known to send a few slack messages per week on important issues. The Slack workspace `#general` channel is also connected to the course homepage via an integration. So every time I change the homepage (e.g. to release a new homework or upload notes or a video), you will get a notification.

I can't stress the following enough: if you are not on slack, you will miss all class announcements!!! Slack notifies you when there are messages. You may wish to mute all channels except for #general. See this article for how to do that.

The Use of Slack and Github as a Learning Management System

As the course homepage is updated (e.g. a new homework assignment is posted), you will hear about it in slack. Each assignment will have its own channel. You can feel free to discuss things with your fellow students there. If you are asking me a question, you must do so in the `#discussions` channel for a general questions or the assignment-specific channel (e.g. `#HW03`) so other students can see the question and benefit from the answer. If you pm me, I will not answer and just ask you to move it to the public channel. Do not be afraid to ask questions. There are many people who will have your same question!

Slack is a wildly successful company that recently got bought by Salesforce because businesses *really* use it. Pretend you are working at one of these businesses: no posting about random stuff; keep things professional!

We will not be using any features of github for learning management. Do not open "issues" on github!

Class Meetings

There are 56 scheduled meetings: 28 lecture periods of 110min followed and 28 lab periods of 55min (cunyfirst only lists it as 50min and that's a mistake). Of the 28 lecture periods, 24 will be lectures (see next two sections), 2 will be midterm exams which are in class and 2 will be review periods before the exams. Of the 28 lab periods, there will be 20 periods split into sets of 2 periods to cover 10 weekly lab assignments (see lab section further on). The other 8 lab periods will be devoted to either pure computing lectures, review labs (assignments that do not need to be handed in) or pure writing labs that help with the papers.

Lectures

Lectures will be further split into two modes: theory and practice. The first is a standard "chalkboard" lecture where we learn concepts and the second will be using the "computer/projector" to see the concepts in action in the `R` language.

Lecture Schedule

Below is a tentative schedule of the theory and practice topics covered by lecture number.

(1) **Theory:** Review of syllabus, introducing science and modeling, definition of phenomena / the response y, reality vs. approximation, measurement vs. prediction and simulation, definition of learning from data, model validation, heuristics, ambiguous models, mathematical models, causal inputs, response spaces $\mathcal{Y}$, definition of features $\boldsymbol{x}$ and its feature space $\mathcal{X}$.

Practice: Short history of `R`, introduction to RStudio, arithmetic, assignment of variables, mathematical functions, logical operations, numeric / integer / boolean data types, vectors, sequences, subsetting, sorting, taxonomy of illegal values.

(2) **Theory:** Feature spaces, binary, categorical and continuous data types, definition of metrics, ordinal codings, sample size n vs. number of features p, error due to ignorance of information δ, optimal response surface f, definition of training data $\mathbb{D}$, definition of candidate function set $\mathcal{H}$, definition of prediction function g, algorithms that produce prediction functions $\mathcal{A}$, definition of misspecification error, definition of optimal candidate function h^*, irreducible error $\mathcal{E}$, estimation error and residuals e.

Practice: Realizations of popular random variables, PDF / PMF / CDF / empirical CDF computations, quantile computations, factor-type variables, matrix data type and its critical functions, if / if-else / else / switch programmatic control, for / while / repeat loops, console printing, errors and warnings, try-catch control.

(3) **Theory:** Visualization of training data $\mathbb{D}$, threshold models for classification, null model for classification g_0, concept of a parameter θ and parameter space Θ, degrees of freedom, definition of objective function / error function, accuracy, sum of squared error (SSE), optimization within $\mathcal{A}$, linear threshold models, perception learning algorithm (PLA), introduction to neural networks.

Practice: Review of hashing and the list data type, the array data type for general tensors, naming for vectors / matrices / tensors, introduction to specifying functions, arguments, argument defaults, creating data matrices, tabling multiple features, the dataframe data type.

(4) **Theory:** Review of lines in multiple dimensions with Hesse Normal Form, derivation of the support vector machine (SVM) using maximum margin objective, hinge error, SVM using the Vapnik objective, defintion of hyperparameters λ, K-nearest neighbors algorithm (KNN).

Practice: Installing and loading libraries from CRAN and githuv, public vs private functions and scoping, loading datasets from libraries / files / URLs, creating threshold models, writing the PLA, matrix operations: arithmetic / transpose / inverse / rank / trace, optimization algorithms e.g. Nelder-Mead, writing the KNN algorithm, using the SVM library and setting the hyperparameter.

(5) **Theory:** Null model for regression g_0, linear models for continuous responses $\boldsymbol{\beta}$, minimization of SSE for $p = 1$ using basic calculus to arrive at the ordinary least squares (OLS) solution $\boldsymbol{b}$, review of covariance of two random variables $\sigma_{X,Y}$ and its estimate $s_{X,Y}$, review of correlation $\rho_{X,Y}$ and its estimate $r_{X,Y}$.

Practice: Computing sample covariance and correlation in the context of the OLS algorithm for $p = 1$, computing OLS error metrics, the formula object, using the `lm` function, visualizing the OLS line atop a scatterplot, computing predictions in OLS, OLS in the boston housing data

(6) **Theory:** Error metrics for regression: SSE, mean squared error (MSE) and root mean squared error (RMSE), approximate prediction confidence intervals using RMSE, sum of squares total (SST), concept of proportion of variance explained R^2, class of models with $R^2 < 0$, perfect fit models with $R^2 = 1$.

Practice: OLS on the Galton height data and the etymology of the word "regression" in statistics, computing OLS in the case of categorical variables (ANOVA), dummifying variables, computing model matrices.

(7) **Theory:** OLS estimates being the group averages with one binary feature, independence, dependence, association, correlation, OLS estimates with $p > 1$, design matrix X, vector derivative properties: constant scalars, multiples, multiplication, quadratic forms, general OLS solution, review of matrix inverses, transposes, rank and symmetric matrices.

Practice: Visualizing R^2 for a model using error density estimation, computing general OLS estimates in multiple dimensions from scratch, making predictions using the `predict` interface for modeling.

(8) **Theory:** OLS predictions as linear transformations, review of the linear algebra concepts of dimension, length, norm, subspace, linear in/dependence, column space, derivation of the orthogonal projection matrix for one dimension via law of cosines, outer products, idempotency.

Practice: Eigendecomposition, computing error metrics for general OLS, computing the null model.

(9) **Theory:** Derivation of the orthogonal projection in multiple dimensions, equivalence of the OLS algorithm with orthongal projection, definition of the hat matrix H, review of the linear algebra concepts of eigenvectors and eigenvalues, computing the eigenvectors and eigenvectors of H.

Practice: Computing the hat matrix H for the null model and in general, confirming its eigendecomposition and idempotency and rank, using H to find the OLS predictions and residuals and verifying their orthogonality.

(10) **Theory:** Verification of the symmetry and idempotency of H, proving that one multi-dimensional orthogonal projection is in general not the same as the sum of orthogonal projections in the component dimensions except if the component dimensions themselves are orthogonal, review of orthonormal matrices Q, proving the equivalence of $H = QQ^\top$, definition of $X = QR$ decomposition, the Gram-Schmidt algorithm, computing R, deriving the least squares estimate using Q and R.

Practice: Computing QR decomposition and confirming the orthonormality of Q, verifying the OLS predictions are the same with Q, writing the Gram-Schmidt algorithm, an overview of the piping / chaining concept in modern programming and in `R` with demos.

(11) **Theory:** Definition of sum of squares for the regression SSR, proving the sum of squares identity $SST = SSR + SSE$, showing that if p increases by one dimension, then SSR is obligated to increase forcing R^2 higher, definition of overfitting in modeling, definition of chance capitalization, demonstrating that full overfitting in OLS leads to $H = I$, a superficial introduction to regularization (lasso regression and ridge regression).

Practice: Demo showing the iterative addition of a feature and R^2 monotonically increasing and RMSE monotonically decreasing (overfitting), demonstrating that random vectors are never truly orthogonal and thus their projections are non-zero, a lasso fit and a ridge fit of a dataset with $p \geq n$.

(12) **Theory:** Definition of in-sample error metrics vs. out-of-sample (oos) error metrics, splitting $\mathbb{D}$ into $\mathbb{D}_{\text{train}}$ and $\mathbb{D}_{\text{test}}$ via split constant K, definition of "honest" validation via oos error metrics, definition of the final model g_{final} definition of underfitting, tracing the underfitting-overfitting complexity curve, definition of optimal-complexity models.

Practice: A more full demo of overfitting with visualizations, demo of consistency of OLS estimates, code to create train-test splits, demonstration that oos error is larger than in-sample error in the scenario of the model being overfit.

(13) **Theory:** Definition of raw features versus derives features, increasing complexity in $\mathcal{H}$ using polynomial functions of raw features, interpretation of OLS coefficients $\boldsymbol{b}$, Weierstrauss Approximation Theorem, OLS with polynomial features, definition of the full rank Vandermonde matrix, definitions of interpolation vs. extrapolation.

Practice: Square, cube and higher order polynomial fitting both raw and orthogonal with visualization, overfitting with high-degree polynomials, demonstration of extrapolation in models of many different polynomial degrees, prediction with polynomial models, extrapolation in the Galton height data.

(14) **Theory:** OLS using the log transformation on both features and response and interpretation of $\boldsymbol{b}$, derivation the log change is approximately percentage change, definition of first-order interactions in OLS, interpretations of coefficients in interaction models.

Practice: Log-linear model fitting and log-log linear model fitting, logging response to reduce the effect of influential observations, the grammar of graphics and the `ggplot` package to create histograms, scatterplots, box-whisker, violin plots, smoothing plots, overloading plots with many features, faceting, coloring, aesthetics and themes, using color illustrations and faceting to visualize potential first-order interactions in a linear model, fitting interaction models.

(15) **Theory:** oos error metrics as estimates of model generalization error, sources of variance in these estimates, mitigation by adjusting K, further mitigation by using cross-validation (CV), K-fold CV and its aggregated error estimates, approximation confidence intervals for generalization error, discussion of reasonable values of K in practice.

Practice: Simulating many different train-test splits to underscore that K trades bias vs. variance in the generalization estimate, writing code for K-fold CV, using the package `mlr3` to automate K-fold CV.

(16) **Theory:** Introduction of the fundamental problem of "model selection" of candidates $g_1, \ldots, g_M$, model selection with honest validation via splitting $\mathbb{D}$ into $\mathbb{D}_{\text{train}}$, $\mathbb{D}_{\text{select}}$ and $\mathbb{D}_{\text{test}}$ via split constants K_{select} and K_{test}, procedure to select best model among M candidates and validation of the best model.

Practice: Writing code for the model selection procedure, review of basic `C++`, optimizing `R` code via the `Rcpp` package, benchmarking routines that require heavy looping between `Rcpp` and base `R`, benchmarking routines that require heavy recursion between `Rcpp` and base `R`.

(17) **Theory:** Double-CV in the model selection procedure using inner folds and outer folds, discussion of reasonable values of K_{select} and K_{test} in practice, applying the model selection procedure to grid searching to locate the best value of hyperparameters λ in algorithms that require λ, definition of stepwise modeling using the model selection procedure via the underfitting-overfitting complexity curve concept, stepwise OLS with a large basis of candidate terms.

Practice: Using the package `mlr3` to automate the double-CV using inner and outer loops, using the package `mlr3` to automate the locating of optimal hyperparameters, demo of forward stepwise linear modeling and tracing the underfitting-overfitting complexity curve.

(18) **Theory:** Definition of hyperrectangle basis for $\mathcal{X}$ and its OLS solution, unfeasibility of this algorithm in high p, introduction of the regression tree algorithm.

Practice: Binning model demo and visualization for varying bin sizes, introduction to data wrangling using the packages `dplyr` and `data.table`: filtering, sorting, grouping, summarizing, feature derivation, dataframe joining (left, right, inner, full, between / overlap), benchmarking the two libraries.

(19) **Theory:** Full specification of regression tree algorithm: definition of a binary tree, definition of orthogonal-to-axes splits, nodes vs. leaves, left-right SSE weighting, leaf assignments, overfitting and tree-pruning.

Practice: Using the `YARF` package to produce regression trees, querying tree stats, visualizing trees and tree model predictions, tree differences by the pruning hyperparameter.

(20) **Theory:** MSE of g decomposition into bias and irreducible error for one $\mathbb{D}$, MSE of g decomposition into bias, irreducible error and variance for multiple $\mathbb{D}$'s, MSE decomposition of M different g's averaged, strategies to eliminate bias and variance, non-parametric bootstrap sampling, Breiman's concept of bootstrap aggregation (bagging), correlation ρ among the bootstrapped models, out-of-bag (oob) observations, validation in bagging via oob samples.

Practice: Visualizing M bagged trees, demonstrating near zero bias, demonstrating variance reduction as $M \to \infty$, a comparison to OLS and high degree polynomial models, demonstration of generalization error improvement, demonstration of validation in bagging.

(21) **Theory:** Demonstrating the bias for regression trees is near zero, reducing correlation among the bootstrapped models using feature sampling, introduction of random forests (RF) algorithm.

Practice: Demonstration that RF decreases ρ and demonstration that it outperforms bagging in both regression and classification.

(22) **Theory:** A basic discussion of "causality" from a philosophical perspective, directed causal graphs, correlation vs. causation, incidental effects, lurking variables, spurious correlation, causation is defined by manipulation, a quick definition of randomized experimentation, real-world causal diagrams, wrong interpretations of $\boldsymbol{b}$ in OLS, the highly limited but true / complete paragraph-long interpretation of $\boldsymbol{b}$ in OLS, a discussion of how OLS regression accomplishes estimation of single features with other features *ceteris paribus*.

Practice: Demos of whimsical spurious correlations, demonstration that spurious correlations are easy to find in simulation, a nice illustration of correlation without causation using an OLS model that reveals the true interpretation of $\boldsymbol{b}$.

(23) **Theory:** Introduction of classification tree algorithm, the gini metric, leaf assignments, the two errors: false negatives and false positives, the 2×2 confusion matrix and its metrics: precision, recall, accuracy, F_1 metric, false discovery rate, false omission rate.

Practice: Using the `YARF` package to produce classification trees, querying tree stats, visualizing trees and tree model predictions, tree differences by the pruning hyperparameter, measuring the two errors, computing confusion matrices and the other metrics.

(24) **Theory:** Missing data mechanisms (MDMs): missing completely at random, missing at random, not missing at random with examples, strategies to handle missingness: listwise deletion, imputation, multiple imputation, miss forests algorithm and its convergence, the concept of "retaining" missingness even after imputation, introduction of probability estimation using the independent bernoulli random variable model, optimal probability function, likelihood of $\mathbb{D}$, $\mathcal{H}$ for probability functions, generalized linear modeling, link functions: logistic / probit / complementary log-log, numerical approximations to the likelihood optimization.

Practice: An example of a dataset with missingness, assessing the different MDMs, listwise deletion, imputation and the `missForest` package for the recommended imputation, creating the missingness dummies as derives features in X.

(25) **Theory:** Definition of logistic regression (LR), log-odds interpretation of LR $\boldsymbol{b}$, prediction in LR, error metrics for probability estimation: Brier and Log scoring rules, classification modeling from probability regression, optimal asymmetric cost modeling, response-operator curves (ROC), detection-error tradeoff curves (DET).

Practice: Fitting LR models using the `glm` package, predicting with LR models, validating creating asymmetric cost classifiers in LR models, constructing ROC and DET plots using LR models, locating optimal models with minimal cost.

(26) **Theory:** Causal diagrams, confounding, correlation does not imply causation, correct interpretation of OLS estimates, asymmetric cost classification in tree models, introduction to boosting as a meta-algorithm, adaboost

Practice: demonstrating confounding and linear models by hiding the confounder and then revealing it, demonstration of asymmetric classification using trees, demonstration of boosting using package `xgboost`.

Labs

During labs will be in the computer-enabled classroom, the majority of this time will be your time. You will take turns "driving" the coding in front of the class, working on exercises that you will finish for homework. Thus we will spend most a lot of time talking through problem-solving skills in data science.

Homework

Homework will be split into *theory* and *practice* (called "labs"). This course will be the "writing in the major course" next year. Thus, a portion of each theory and practice homework will involve writing *English* and being graded on *English*.

Theory Assignments

There will be 4 or 5 theory homework assignments. Homeworks will be assigned and placed on the course homepage and will usually be due a week later in class. Homework will be

graded out of 100 with extra credit getting scores possibly > 100. I will be doing the grading and will grade an *arbitrary subset of the assignment* which is determined after the homework is handed in.

Homework must be handed in by emailing it to me as a PDF. You must do one of two things:

- **Print out the homework and handwrite your answers in the alotted space for each question. Then scan your homework as a PDF. There are a ton of good photo-PDF apps for both iPhone and droid.**
- **Open the PDF on your device and use a PDF-editing program to electronically handwrite your answers and save the PDF.**

I will NOT accept homework that is not atop the original rendered homework PDF file. Remember to write your name. There are no regrades during this pandemic semester. Homework must be at maximum 5MB.

You are encouraged to seek help from me if you have questions. After class and office hours are good times. You are highly recommended to work with each other and help each other. You must, however, submit your own solutions, *with your own write-up* and in *your own words*. There can be no collaboration on the actual *writing*. Failure to comply will result in severe penalties. The university honor code is something I take seriously and I send people to the Dean every semester for violations.

Lab Assignments

These will almost exclusively consist of short and medium coding exercises in `R`. Most of the assignment will be done for you and your peers during the Friday lab session.

Philosophy of Homework

Homework is the *most* important part of this course.[1] Success in Statistics and Mathematics courses comes from experience in working with and thinking about the concepts. It's kind of like weightlifting; you have to lift weights to build muscles. My job as an instructor is to provide assistance through your zone of proximal development. With me, you can grow more than you can alone. To this effect, homework problems are color coded green for easy, yellow for harder, red for challenging and purple for extra credit. You need to know how to do all the greens by yourself. If you've been to class and took notes, they are a joke. Yellows and reds: feel free to work with others. Only do extra credits if you have already finished the assignment. The "[Optional]" problems are for extra practice — highly recommended for exam study.

[1] In one student's observation, I give a "mind-blowing homework" every week.

Time Spent on Homework

This is a three credit course. Thus, the amount of work outside of the 4hr in-class time per week is 6-9 hours. I will aim for 7.5hr of homework per week on average. However, doing the homework well is your sole responsibility since I will make sure that by doing the homework you will study and understand the concepts in the lectures and you won't have all that much to do when the exams roll around.

Late Assignment Policy

Late homework will be penalized 10 points per business day (Monday–Friday save holidays) for a maximum of five days. *Do not ask for extensions*; just hand in the homework late. After five days, **you can hand it in whenever you want** until *the last scheduled class meeting according to the official academic calendar*. As far as I know, this is one of the most lenient and flexible homework policies in college. I realize things come up. Do not abuse this policy; you will fall far, far behind.

LaTeX Homework Bonus Points

Part of good mathematics is its beautiful presentation. Thus, there will be a 1–5 point bonus added to your theory homework grade for typing up your homework using the LaTeX typesetting system based on the elegance of your presentation. The bonus is arbitrarily determined by me.

I recommend using overleaf to write up your homeworks (make sure you upload both the hw#.tex and the preamble.tex file). This has the advantage of (a) not having to install anything on your computer and thus not having to maintain your LaTeX installation (b) allowing easy collaboration with others (c) alway having a backup of your work since it's always on the cloud. If you insist to have LaTeX running on your computer, you can download it for Windows here and for MAC here. For editing and producing PDF's, I recommend TeXworks which can be downloaded here. Please use the LaTeX code provided on the course homepage for each homework assignment.

If you are handing in homework this way, read the comments in the code; there are two lines to comment out and you should replace my name with yours and write your section. The easiest way to use overleaf is to copy the raw text from hwxx.tex and preamble.tex into two new overleaf tex files with the same name. If you are asked to make drawings, you can take a picture of your handwritten drawing and insert them as figures or leave space using the "\vspace" command and draw them in after printing or attach them stapled.

Since this is extra credit, do not ask me for help in setting up your computer with LaTeX in class or in office hours. Also, **never share your LaTeX code with other students** — it is cheating and if you are found I will take it seriously.

Homework Extra Credit

There will be many extra credit questions sprinkled throughout the homeworks. They will be worth a variable number of points arbitrarily assigned based on my perceived difficulty

of the exercise. Homework scores in the 140's are not unheard of. They are a good boost to your grade; I once had a student go from a B to and A- based on these bonuses.

TA Office Hours

This class has two TA's.

- Kennly will be running a 1hr/wk office hour session to help with the theory part of the course. Lab questions can be asked as well, but theory questions will be prioritized.
- Amir will be running a 1hr/wk office hour session to help with the lab part of the course and to answer any questions about `Python`. He translated all in-lecture `R` demos to `Python` and all labs assignments from `R` to `Python` as well. He is your guide to mastering Python.

Writing Assignments

There will be two writing assingments. (1) A "philosophy of modeling" essay. Here you will coalesce the non-mathematical material that is crucial to this class. The purpose is to make you truly understand the modeling process and its limitations from start to finish. (2) A final project. Here you will use build a predictive model using a dataset. This is the capstone project for the entire data science and statistics major and it is where you will tie everything together.

This class will soon be the writing in the major course. Thus, writing is a major part of the curriculum herein.

Examinations

Examinations are solely based on homeworks (which are rooted in the lectures)! If you can do all the green and yellow problems on the homeworks, the exams should not present any challenge. I will *never* give you exam problems on concepts which you have not seen at home on one of the weekly homework assignments. There will be three exams and the schedule is below.

Since the is the capstone course, there is no final exam, but a large final project. There will be two midterm exams and the schedule is below.

Exam and Major Assignments

- Midterm examination I will be on [see course homepage] in class with the first review session on the class meeting prior
- The philosophy of modeling paper's first draft is due [see course homepage]
- The philosophy of modeling paper's final draft is due [see course homepage]

- Midterm examination II will be on [see course homepage] in class with a review on the class meeting prior
- The final project is due [see course homepage]

Exam Policies and Materials

I allow you to bring any calculator you wish but it cannot be your phone. The only other items allowed are pencil and eraser. I do not recommend using pen but it is allowed. Food is not allowed during exams but beverages are allowed.

I also allow "cheat sheets" on examinations. For both midterms, you are allowed to bring two 8.5" × 11" sheet of paper (front and back). Four sheets single sided are not allowed. On this paper you can write anything you would like which you believe will help you on the exam.

Cheating on Exams

If I catch you cheating, you can either take a zero on the exam, or you can roll the dice before the University Honor Council who may choose to suspend you.

Missing Exams

There are no make-up exams. If you miss the exam, you get a zero. If you are sick, I need documentation of your visit to a hospital or doctor. Expect me to call the doctor or hospital to verify the legitimacy of your note.

Accommodations for Students with Disabilities

Candidates with disabilities needing academic accommodation should: 1) register with and provide documentation to the Special Services Office, Frese Hall, Room 111; 2) bring a letter indicating the need for accommodation and what type. This should be done during the first week of class. For more information about services available to Queens College candidates, contact: Special Service Office; Director, Miriam Detres-Hickey, Frese Hall, Room 111; 718-997-5870 (Monday – Thursday 8:00 a.m. to 5:00 p.m. & Friday 8:00 a.m. to 4 p.m.).

Class Participation

This portion of your grade is assessed based on your level of interaction during the course lectures e.g. asking / answering questions. Participation on slack counts towards this total.

Grading and Grading Policy

Your course grade will be calculated based on the percentages as follows:

Theory Homework	5%
Labs	10%
Midterm Examination I	26%
Midterm Examination II*	26%
Philosophy of Modeling Paper	13%
Final Project with Writeup	15%
Class participation	5%

*The second midterm is not cumulative. It only covers material *after* midterm I.

The Grade Distribution

As this is a small and advanced class, the class curve will be quite generous. Last year, it was approximately 40% A's and 40% B's. If you do your homework and demonstrate understanding on the exams, you should expect to be rewarded with an A or a B. C's are for those who "dropped out" somewhere mid-semester or who cannot demonstrate basic understanding. I have never given an F in this class. Don't give me a reason to change this tradition!

Checking your grade and class standing

You can always check your grades in real-time using `http://qc.gradesly.com`. You will enter in your QC ID number (or CUNYfirst email address). I will provide you with your password by email after the first assignment is completed.

Auditing

Auditors are welcome. They are encouraged to do all homework assignments. I will even grade them. Note that the university does not allow auditors to take examinations.