Module M1: Goals of Learning Machines
What is the Statistical Machine Learning Framework Considered Here?
*Readings***:** Chapter 1, Section 4.1
Estimated Lecture Time: 90 minutes

- Statistical Environment
  - What is the SML Framework? (Figure 1.1) [Model learns DGP represented as Pe …not Data Set]
  - Environment: Data Generating Process
    - Matrix Notation (Vectors, Matrices, Vector-valued functions, Vector-valued Matrices)
    - Random Vectors and Random Matrices
    - Feature Vectors for Representing Environmental Events
      - Numerical Coding Scheme: Photographic Image
      - Binary coding Scheme: True/False Features ("has cough","has yellow dots on skin"
      - Categorical coding scheme: variable with unordered values
        ACTION: move-forward, move-backward, do-not-move
        Traffic-Light-State: Red, Green, Yellow
        Reference Cell/One-Hot coding
      - Curse of Dimensionality (more features → more functions to choose from)
    - Sampling from a Hat/Environment with Replacement (Pe)
  - Model: Loss Function
  - Learning: Combining DGP and Loss Function into goal of minimizing Risk Function (expected loss)
  - Statistical Machine Learning Design Process (Recipe Box 1.1)
- Introduction to Supervised, Unsupervised, and Reinforcement Learning
  - Supervised Learning: Event has two components input vector and target vector (Environment/world is modeled as a Hat). Show as special case of Empirical Risk.
  - Unsupervised Learning: Event just consists of an input vector (Environment/world is modeled as a Hat). Show as special case of Empirical Risk.
  - Reinforcement Learning: Event is a sequence of interactions between learning machine and environment represented by a sequence of vectors which may include a reinforcement signal. (Environment/world is modeled as a collection of Hats). Show as special case
- Stock Market Prediction Example
  - ANN graphical notation nodes as processing units; activation levels; parameters/weights)
  - MATLAB Grader Example (illustrate MATLAB implementation and MATLAB grader). (Ch0Q1, Ch0Q2)
    Walk students through code; Doing the assignment. Submitting the assignment.

Module M2: Gradient Descent Introduction.
How Can We Minimize the Empirical Risk Function?
Readings: Section 1.3.4, Section 4.1, Section 5.2
Estimated Lecture Time: 90 minutes

- Function Minimization
  - Grid Search: one-dimensional versus multi-dimensional (note exponential growth)
  - Gradient Descent Method (Blind Skier Analogy; Calculus: Steepest Descent Direction is Derivative)
    - What is the Gradient Descent Method (one-dimensional Example)
    - Matrix Chain Rule (see Chapter 5)
    - Gradient Descent in Multi-dimensional Space
  - Application of Gradient Descent Method to Modeling Supervised Learning
    - Method 1 (Batch Learning): Sample from Hat to Form Empirical Risk Function. (1 hat-1 environment)

Use Batch Learning Gradient Descent to minimize Empirical Risk Function
by updating parameter estimates. Then repeat.
Give very brief overview of batch learning gradient descent.
Example 1.3.5 Batch Gradient Descent Stock Market Prediction
MATLAB GRADER Example (Illustrate MATLAB implementation in MATLAB grader)

- Method 2 (Mini-Batch Adaptive Learning):
  Sample from Hat to Estimate Gradient of Risk Function (expected loss)
- Method 3 (Adaptive Learning)
  Only 1 sample in Mini-Batch (1 hat-1 environment)
  Adaptive Learning Version of Example 1.3.5
  MATLAB Grader Example (illustrate MATLAB implementation and MATLAB grader)

- Reinforcement Learning (example: a robot control problem)[policy gradient reinforcement]
  Each parameter update results in a Hat Change (environment change)
  Risk = E{c(X|theta) pe(X|theta)}
  Gradient of Risk results in a two-stage sampling solution.
  Step 1: Sample initial state of an episode from the environment
  Step 2: Sample "action" from "initial state" and current parameters of robot
  Step 3: Update parameters of robot using adaptive gradient descent
  Step 4: Go To Step 1
  Note that the Environment is changing in Step 1.
  Robot's strategy for selecting an action is changing in Step 3.

Module M3: How Can We Design Linear Supervised Learning Machines?

Readings: Sections 1.5.1

Estimated Lecture Time: 90 minutes

- Begin with MATLAB grader example. Then Move into Module M3. Mention Maximum Likelihood section 13.2
- Concept of Supervised Learning; Discrepancy Function concept.
- Linear Regression Example 1.5.1 (Numerical Target)[predict conditional mean] (Draw diagram) (minimize residual error…which is variance of Conditional Gaussian..maximum likelihood estimation)
- Logistic Regression Example 1.5.2 (Binary Target)[predicted probability] (Draw diagram) (maximize predicted probability…maximum likelihood estimation)
- Multinomial Logit Regression Example 1.5.3. (Categorical Target)[predicted pmf] (Draw diagram) (maximize predicted probability…maximum likelihood estimation)

Module M4: How Can We Design Deep Nonlinear Supervised Learning Machines?

Readings: Section 1.5.2

Estimated Lecture Time: 90 minutes

- Logic Machine Approximators
  - Representation of Arbitrary Logical Function using Disjunctive Normal Form Neural Net
  - Approximating Logic Gates using Logistic Sigmoidal Functions (Example 1.5.5)
  - Shallow Neural Net Example 1.5.4 with Sigmoidal
- Function Approximators
  - Approximating Functions By Bump Superposition
  - Shallow Neural Net Example 1.5.5. with Radial Basis
- Approximation Theorems (Sigmoidal, Radial Basis Function, Softplus) (one-layer and two-layer theorems
- Residual Network Concept (Example 1.5.6)

Module M5: Automatic Feature and Hidden Unit Selection using Regularization.
Readings: Section 1.3.3
Estimated Lecture Time: 45 minutes

- Improving Generalization Performance by Preventing Overfitting
    - How to Choose Input Features? Related to Curse of Dimensionality?
    - How to Choose Number of Hidden Units?
        - Number of Hidden Units Too Large → Memorize
        - Number of Hidden Units Too Small→Inadequate Representation
    - IDEA: Have learning machine figure out which parameters to eliminate or parts of architecture to eliminate or decorrelate. Decorrelate=eliminate redundant features.
    - Regularization concepts: L1, L2, Elastic Net (L1-sparse, L2-decorrelate, Elastic Net-both) as Soft Constraints. (L2 decays all weights equally; while L1 decays same amount). Explain what they are and how they work.

Module M6: How Can We Learn About Sequences of Events? Recurrent Nets.
Readings: Section 1.5.3
Estimated Lecture Time: 90 minutes

- What are the RNN problems: prediction, classification, generation, translation
- Simple Recurrent Network
- Brief Comments about Gated Recurrent Network (don't go into details)

Module M7: How to Design an Unsupervised Learning Machine.
Readings: Section 1.6
Estimated Lecture Time: 90 minutes

- Examples of Unsupervised Learning?: filtering, reconstruction, compression, clustering
- Filtering/Reconstruction
    - Example 1.6.1—Reconstruction (LSI)
    - Example 1.6.2.-Reconstruction/Filtering
      (Linear Autoencoder Exercise 1.6.3, Nonlinear Autoencoder, Denoising Autoencoder)
- Clustering
    - K-Means Clustering;
    - General Clustering Example 1.6.4
    - Stochastic Neighborhood Embedding (Example 1.6.5; Exercise 1.6.2) Just sketch approach.

Module M8: How to Design a Reinforcement Learning Machine (Value Function Approach)
Readings: Section 1.7.1, 1.7.2
Estimated Lecture Time: 90 minutes

- What is Reinforcement Learning?
- Linear Machine Example 1.7.2
- Nonlinear Machine Example/Exercise 1.7.2

Module M9: How to Design a Reinforcement Learning Machine (Policy Gradient Approach)
Readings: Section 1.7.3
Estimated Lecture Time: 90 minutes

- What is a Reactive Versus Passive Learning Environment?
- Problem: How can we solve reinforcement learning in a reactive environment?
  Answer: Policy Gradient Reinforcement Learning
- Discuss Basic Theory with Constant Reinforcement

- Linear Machine Example (Simplified Lunar lander Problem)
- Nonlinear Machine Example (Simplified Lunar Lander Problem with Shallow Net Controller)

- Theory where Reinforcement Function is Learned
  - Model-Based Reinforcement Learning Example 1.7.3
    (Example where Reinforcement Function is Learned) [Example 1.7.3)

Module M10: Design of Unsupervised Learning Machines using SVD

Readings: Section 4.2

Estimated Lecture Time: 90 minutes

- How can we solve Compression Problems?
  - Draw Two Layer Linear Machine
  - Positive Definite, Eigendecomposition (Theorem 4.2.3)
    Example of Image Compression Problem for Symmetric Matrix
  - Singular Value Decomposition (Theorem 4.2.6) (interpreting as Neural Net see 4.3-4)
  - Example 4.2.3. Image Compression for General Matrix
    Exercise 1.6.1 Latent Semantic Indexing Problem
    (Document Retrieval, Document Similarity, Term Similarity, etc.)

Module M11: Design of Supervised Learning Machines using SVD

Readings: Section 4.3

Estimated Lecture Time: 90 minutes

- How can we Understand Response of the Linear Machine as a Music Equalizer (Fourier Analysis) Filter
  - Draw Linear Machine
  - Exercise 4.2.4. Linear Inference Machine Response (recognizes inputs in a subspace)

- How can we Obtain a Formula for Solving a linear least Squares Minimization Problem
  - Theorem 4.3.1. Least Squares Minimization using Pseudoinverse and SVD
  - Example 4.3.2. Linear Regression using SVD
  - Application to Multilayer Deep Neural Networks (re-estimating the final layer)

Module M12: What are Convergence Concepts and Continuous Functions

Readings: Section 5.1, Section 5.2

Estimated Lecture Time: 135 minutes

- Deterministic Convergence to a Point (Definition 5.1.1)
- Deterministic Convergence to a Set (Definition 5.1.8)
- Bounded Sequence (Definition 5.1.5)(Example 5.1.1, Example 5.1.2)
- Continuous Functions (trace a curve)
- Why are Continuous Functions Important?
  - Reason 1: Gradient Descent (doesn't work if the function isn't continuous…show on a diagram)
  - Reason 2: If parameter values converge, does that mean that a function of the parameter values converges? Continuous Function Convergence Theorem 5.1.4
  - Reason 3: Generalization Performance
    (similar responses for similar patterns even if they have never been seen)
  - Reason 4: Create Well-Behaved Measurement Instruments

- Continuous Function Composition Theorem 5.1.7
- Example 5.1.11
- Big O and little O notation

Module M13: How Can we Represent and Compute Derivatives for Functions with Many Variables?
Readings: 5.2.1, 5.2.2, 5.2.3
Estimated Lecture Time: 90

- Problem: We want to design gradient descent learning machines which have many parameters… possibly hundreds or thousands of parameters.
  We need a powerful and compact notation to take derivatives, represent them, and implement them.
  Vector Derivatives greatly simplify calculations and results.
    - Definition of a Vector Derivative (Definition 5.2.1)
    - Theorem 5.2.1. Sufficient Condition for Differentiability
    - Convention for Representing Matrix Derivative: Jacobian (Definition 5.2.3)
    - Derivative of scalar function with respect to parameter vector is a row vector of functions
    - Gradient is column vector of functions
    - Hessian is matrix of functions
    - Matrix Derivative (we will not use this much but its good to know)

- Problem: Show a multilayer feedforward network with error function (discuss challenges of deriving a gradient descent algorithm).
    - Rewrite the problem by breaking the problem into a collection of small computable functions.
    - Introduce an "expanded notation" which treats the "sigmodal nonlinearity" as an additional transformation explicitly on the graph.
    - Emphasize by this function decomposition method that if this was a series of scalar functions we could apply the scalar chain rule for calculus but for this problem we need a vector chain rule or possibly even a matrix chain rule.
        - Define the Vector Chain Rule (Theorem 5.2.2)
        - Define the Matrix Chain Rule (Theorem 5.2.3)
    - Example 5.2.10

Module M14: How to Design Linear Machine Gradient Descent Algorithms using Matrix Calculus.
Readings: Section 5.2.3
Estimated Lecture Time: 90 minutes
Problem: How can we apply Scalar-Vector Product Derivative Rule and Dot Product Derivative Rule to linear type networks (no hidden units)

- Gradient for Linear Regression Model (Example 5.3.2)
- Gradient Descent for Hebbian Learning (Example 5.3.1)
- Gradient for Logistic Regression Model (Exercise 5.3.1)
- Gradient for Softmax Regression Model (Exercise 5.3.3)

Module M15: How to Design Deep Net Gradient Descent Algorithms using Matrix Calculus.

Readings: Section 5.2.3, 5.2.4
Estimated Lecture Time: 90 minutes

- DIRECTED ACYCLIC GRAPH For representing complicated multiple layer deep learning networks and deriving learning rules where each arrow in the graph is a "layer" in the deep learning neural network
- Example 5.2.17 (discuss for different choices of transfer functions and error functions)
- Gradient Backpropagation for Deep Learning (Algorithm 5.2.1)

Module M16: Analysis of Gradient Descent using Taylor Series.
Why Does Gradient Descent Decrease Objective Function? Answer: Taylor Series Argument
Readings: Sections 5.3.1, 5.3.2
Estimated Lecture Time: 45 minutes

- Problem: Why does gradient descent work?
- Scalar Taylor Series Expansion
- Multivariate Taylor Series Expansion
- Derivation of Gradient Descent Algorithm using Multivariate Taylor Series (and limitations of analysis)

Module M17: Critical Point Test using Taylor Series.
When Should We Stop Learning? Answer: Critical Points and Flat Regions
Readings: Section 5.3.3.1, 5.3.3.2
Estimated Lecture Time: 45 minutes

- Problem: When should we stop the gradient descent algorithm? Looking for when change in state is small is not so great because it is algorithm-dependent. Looking when slope of function vanishes gives us an algorithm-independent measure of convergence.
- We would like to converge to a global minimizer but gradient descent just takes us to a critical point
- What is a critical point in multidimensional space? Definition 5.3.1
    - Numerically robust check for a critical point (Example 5.3.3)
- Flat Regions (as collections of connected critical points) [Definition 5.3.2)
    - Example 5.3.4 Flat regions in linear regression
    - Example 5.3.5 Flat regions in multilayer perceptron empirical risk functions

Module M18: Local Minimizer Test using Taylor Series.
When Has Gradient Descent Algorithm Arrived at a Good Suboptimal Solution?
Local Minimizers and Saddlepoints.
Readings: Section 5.3.3.3, 5.3.3.4
Estimated Lecture Time: 45 minutes

- Problem: We want to converge to a global minimizer but our gradient descent algorithm just converges to a critical point or a flat region. How do we get to a global minimizer?
- IDEAS: (1) Do we need a global minimizer or would a deep local minimizer be ok? No optimal way to pick up a coffee cup. (2) Convex function on convex set we can always end up at a global minimizers. (3) If non-convex, we can try multiple starting points and check where we end up.
- Local Minimizers
    - Definition of a Local Minimizer (Definition 5.3.3)
    - Strict Local Minimizer Test (Theorem 5.3.3) (Derivation of Test using Taylor Expansion)
    - Numerical Stopping Criteria for Strict Local Minimizer (Example 5.3.6)
- Prevalence of Saddlepoints in High-Dimensional Spaces

Module M19: Global Minimizer Test using Convex Functions.
When has Gradient Desent Algorithm Converged to a Globally Optimal Solution?
Readings: Section 5.3.3.5
Estimated Lecture Time: 45 minutes

- Global Minimizers over Entire Parameter Spaces or Small Regions of Space
    - Definition of a Global Minimizer (Definition 5.3.7)
    - Smooth Convex Function (positive semidefinite Hessian on convex subset)
    - Global Minimizer Test (Theorem 5.3.7)
    - Example 5.3.9 Linear Regression Global Minimizer Test

Module M20: Using Lagrange Multipliers to Design Objective Functions with Constraints
Readings: Section 5.3.4
Estimated Lecture Time: 90 minutes

- Hard Constraints Versus Soft Constraints
    - Give examples of L1, L2, and Elastic Net as "Soft Constraints"
    - But suppose we want to do the minimization using "Hard Constraints?"
    - This is where Lagrange Multipliers come in.
- Lagrange Multiplier Theorem (Theorem 5.3.8) Figure 5.4 (Explain "Lagrangrian")
- Example algorithm:
    - (1) Search for a critical point,
    - (2) Block descent on all variables but multiplier…increase on Multiplier since we are looking for a saddlepoint
- Cross Entropy Minimization (Example 5.3.10)

Module M21: Using Lagrange Multipliers to Design Machine Learning Algorithms.
Readings: Section 5.3.4
Estimated Lecture Time: 45 minutes

- Principal Component Analysis (Example 5.3.11) for Designing Recoding Transformations (Example 5.3.12)
- Multilayer Perceptron Gradient Descent (Example 5.3.15)
- L2 Regularization Term (use constraint |theta|^2 + eta^2 = 0) where "eta" is the slack variable (Example 5.3.13)
- Support Vector Machine (Example 5.3.14)

Module M22: Mixed Random Vectors and the Radon-Nikodym Density
Readings: 8.1, 8.2, 8.4
Estimated Lecture Time: 90 minutes

- *What is a Random Vector? Mapping events from Space-Time into subsets of a vector space Mapping events from Vector Space Space-Time Representation into subsets of a vector space. Borel-measurable function concept*
- *Review concept of a Discrete Random Vector*
- *Review concept of a continuous (absolutely continuous) random vector*
- *Discuss adding discrete and continuous. Discuss concatenating discrete and continuous random variables in same vector. Discuss distribution is not PMF or PDF but it is a Radon-Nikodym Density. More mixed random variable examples: volume control, amount of water in a cup placed in a rainstorm*
- *Discuss "support specification concept"*
- *Discuss concept of a random function in context of empirical risk minimization and adaptive gradient descent learning. Why it is useful to us.*
- *Discuss sufficient conditions for expectations to exist*

Module M23: Mixed Random Vector Applications

Readings: 8.4, 8.5

Estimated Lecture Time: 90 minutes

- Problem: How do we represent a DGP for discrete i.i.d. random vectors.
  Example Application of Probability Theory using a PMF DGP sampling with replacement from "hat".
- Problem: What does Empirical Risk for DGP discrete
  Converge to? Expectation of Empirical Risk (Example 8.4.1)
  - Show how this works because frequency of each pattern gets closer to true frequency
- Problem: How do we show adaptive learning works? First attempt.
  Adaptive Gradient Descent Learning Example (Example 8.4.6)
- Problem: Estimate Empirical Risk Approximation Bounds and Required Sample Sizes.
  Chebyshev Inequality (Example 8.5.1). Hoeffding Inequality (Example 8.5.2)

Module M24: Stochastic Sequences

Readings: Section 9.1, Section 9.2

- *Estimated Lecture Time: 90 minutes*
  Definition of a Stochastic Sequence (Definition 9.1.1)
- Definition of an I.I.D. Stochastic Sequence (Definition 9.1.6)
- Example 9.1.2. Sampling With Replacement is an Example of an I.I.D. Sequence Generator
- Missing Data Representations

Module M25: Convergence With Probability One.

Readings: Section 9.3

- Definition of Convergence with Probability One
- Important Applications
  - Strong Law of Large Numbers (Theorem 9.3.1)
  - Importance Sampling Algorithm (Example 9.3.1)
  - Uniform Law of Large Numbers (Theorem 9.3.2)

Module M26: Convergence in Probability/in Mean Square/Distribution with Applications

Readings: Section 9.3

Estimated Lecture Time: 90 minutes

- Convergence in Mean Square
  - Definition of Convergence in Mean-Square (Definition 9.3.3)
  - Weak Law of Large Numbers (Theorem 9.3.5)
- Convergence in Probability
  - Definition of Convergence in Probability (Definition 9.3.4)
  - MS Convergence Implies Convergence In Probability (Theorem 9.3.6)
- Convergence in Distribution
  - Definition of Convergence in Distribution (Definition 9.3.5)
  - Multivariate Central Limit Theorem (Theorem 9.3.7)
- Stochastic Convergence Relationships (Figure 9.4) (Section 9.3.5)

Module M27: Learnability of Probability Models in the Empirical Risk Framework

Readings: Section 1.5, Section 1.6, Section 10.1

Estimated Lecture Time: 120 minutes

- Problem: What is the Probability Model of a Machine Learning Algorithm and how does that probability model fit into the Empirical Risk Minimization Framework? Example 1.5.2, Example 1.5.3, Example 1.5.1, Example 1.6.6
- Problem: Under what conditions is perfect learning impossible? Correctly Specified and Misspecified Models (Section 10.1.1) Models are Imperfect. Problem: Is perfect learning required? Correctly specified versus Predictive Models.
- Smooth Parametric Probability Models (Section 10.1.2) Problem: Why is "smoothness property" important? (Example 10.1.3, Example 10.1.4, Example 10.1.5)
- Local Probability Models (Section 10.1.3)

Module M28: What are Gibbs and Exponential Family Models

Readings: Section 10.2

Estimated Lecture Time: 15 minutes

- Gibbs Probability models (Section 10.2)
- Linear Exponential Family with Examples

Module M29: Factoring Distributions using Bayesian Nets

Readings: Section 10.3

Estimated Lecture Time: 45 minutes

- WHY DO WE WANT TO FACTOR DISTRIBUTIONS? ANSWER: Simplifies Training Algorithm and Architecture Design for Extremely Complicated Distributions How to Factor ANY Joint Distribution in Different Ways.
- What is a First Order Markov Chain (Figure 10.4) (How to Factor and How to Construct from Local Probabilities]
- What is a Bayesian Network (Definition 10.3.2) (Figure 10.3) (Review Directed Acyclic Graph Defiintion 2.2.12) (1) Are conditional probabilities "consistent"? (2) Are they "sufficient" to specify a unique joint distribution? [How to Factor and How to Construct from Local Probabilities]
- Interpreting Linear or Nonlinear Regression as a Bayes Net (Figure 10.5)
- Hidden Markov Model. Hidden Markov Models as probabilistic dynamical systems with probabilistic measurement processes. Writing the Joint Distribution. Implementation of Emission and Transition Probabilities using Softmax Regression or Softmax Neural Networks. Explain why you would do that instead of a contingency table. Multivariate Gaussian Emission Probabilities.
- Bayesian Nets where the nodes are random vectors

Module M30: Markov Random Fields for Representing Probability Distributions

Readings: Section 10.4.1, Example 1.6.6

Estimated Lecture Time: 45 minutes

- Problem: What are the limitations of the Bayesian Network approach? (Beginning Section 10.4)
    - Requires Directed Acyclic Graph
    - Some problems are not naturally formulated as DAGs
    - Homogeneous versus Heterogeneous MRFs
- Examples of MRFs
    - Conditional Nets which cant be represented as Bayes Nets
    - Image Models (Homogeneous Net)

- Markov Logic Nets (Heterogenous Net)[Example 1.6.6]

- Markov Random Field Specification
  - Intuitive Definition (see Figure 10.7 and compare with Figure 10.3)
  - Neighborhood Functions and Neighborhood Systems. MRF Graph.

  Positivity Condition. Simplified Version: p(x) > 0 for all x in support of x where p is joint distribution.
- Positivity Condition: Examples 10.4.1, 10.4.2, 10.4.3
  Definition of a Markov Random Field (Definition 10.4.3)

Module M31: Factoring and Constructing MRFs
Readings: Section 10.4.2
Estimated Lecture Time: 90 minutes
(1) Are conditional probabilities "consistent"? (2) Are they "sufficient" to specify a unique joint distribution?

- Factoring Joint Distribution of a MRF (Section 10.4.2) [need to review concept of a "clique" from Chapter 2]
  - Recipe Box 10.1
  - Example 10.4.6
- Constructing Joint Distribution of MRF from Local Potential Functions (Section 10.4.2)
  - Recipe Box 10.2
  - Example 10.4.7, Example 10.4.8, Example 10.4.9

Conditional Random Fields (Example 10.4.10)

- Mixtures of Random Fields (Example 10.4.11)
- Hidden Markov Random Fields (Example ?:???)

Module M38: Descent Algorithms. What is a well-designed descent algorithm?
Readings: Section 7.1
Estimated Lecture Time: 90 minutes + 90 minutes

- What is a descent algorithm? ("search direction" and "stepsize")
- How do you analyze or design a descent direction?
  - Gradient Descent Algorithm
  - Example 7.1.1 Scaled Gradient Descent
  - Example 7.1.2 Gradient with Automatic Momentum
- How do you select a stepsize?
  - What is a "Downhill stepsize"?
  - What is an "Optimal stepsize"?
  - What is a "Wolfe stepsize"?

Module M39: Descent Algorithm Convergence.
Problem: When would we expect "batch learning" to converge to a solution. What is a solution?
Readings: Section 7.2
Why is convergence of a descent algorithm tricky?

  - Search direction converging to orthogonal direction to negative gradient
  - Stepsize converging too fast or too slow or constant

- Batch Learning Convergence Analysis (Recipe Box 7.1)
- Example 7.2.1: Convergence to Critical Points for Convex and Non-convex Cases
- Example 7.2.2: Convergence of Gradient Descent Batch Learning
- Example 7.2.3: Convergence of Gradient Descent with Regularization
- (also see Example 7.2.5)

Module M40: What are important classes of descent algorithms and search directions?
Readings: Section 7.3

- **Gradient Descent.**
  - See Algorithm 7.3.1 Batch Gradient Descent Algorithm
  - Example 7.3.1: Batch Gradient Descent Design for Shallow Neural Net
- **Newton-Raphson.**
  - Derivation of Newton-Raphson Algorithm (see Equation 7.37)
  - Computational Challenge, Memory Challenge, Convergence Challenge
- **Levenberg-Marquardt.**
  - First Version: Weighted sum of gradient descent and Newton
  - Second Version (more computationally efficient)
  - See Algorithm 7.3.2 Levenberg-Marquardt Algorithm
  - Example 7.3.2. Logistic Regression Parameter Estimation using Levenberg-Marquardt
- **L-BFGS and Conjugate Gradient Descent Methods.**
  - See Algorithm 7.3.3 L-BFGS Batch Learning Algorithm
  - Conjugate Gradient Descent Special Cases

Module M41: What is a MCMC Sampling Algorithm? How to Design MCMC Sampling Algorithms?
Readings: Section 11.1 (2 lectures)

- Applications
  - Finding the minimum of a potential function.
  - Evaluating a high-dimensional integral
  - Generating simulated data (checking model quality)
- When would we expect MCMC to work? [MCMC Convergence Theorem]
  - Irreducible Chain
  - Aperiodic Chain
  - Statement of the MCMC Theorem
- Hybrid MCMC
- Assessing Convergence for Expectation Approximation
  - MCMC Strong Law of large Numbers
  - MCMC simulation Error
- Challenges and Heuristics

Module M42: How to Design MCMC Metropolis-Hastings Sampling Algorithms?
Readings: Section 11.2

- Metropolis-Hastings Algorithm Design (Recipe Box 11.1)
- Metropolis-Hastings Algorithm Convergence Theorem (Theorem 11.2.1)
- Important Special Cases
  - Independence Sampler (Definition 11.2.3)
  - Metropolis Algorithm (Definition 11.2.4)
  - Random Scan Gibbs Sampler (Definition 11.2.5)
  - Deterministic Scan Gibbs Ampler (Definition 11.2.6)
- MCMC Applications
  - Example 11.2.2 MCMC Model Search
  - Example 11.2.3 MCMC Model Averaging
  - Example 11.2.5 Genetic Algorithms

Module 43: How to Design Adaptive Learning Algorithms?
Readings: Section 12.1 (worked through the receipe box illustrating steps with classical gradient descent example)

- Stochastic Gradient Descent Algorithm
- What are components of the stochastic descent algorithm? ("search direction" and "stepsize")
- How do you analyze or design a stochastic descent direction?
    - Average Downward Descent (See Equation 12.5)
    - Specific Condition is in Equation (12.14)
- How do you design an "annealing schedule"?
    - General Stepsize Annealing Conditions (Equations 12.
    - Example Annealing (see Step 3 of Recipe Box 12.1)
- Why would we expect stochastic descent to work? (Recipe Box 12.1)

Module 44: How to Design Passive Adaptive learning Algorithms
Readings: Section 12.2 (eliminate some examples so lecture is not rushed)

- ~~Example 12.2.2 Adaptive learning in logistic regression~~
- Example 12.2.1 Minibatch adaptive learning
- ~~Example 12.2.3 Adaptive Momentum adaptive learning~~
- Example 12.2.4 Random Block Gradient Coordinate Descent
- Example 12.2.6 Data Augmentation Models for Improving Generalization

Module 45 How to Design Reactive Adaptive Learning Algorithm (1 entire lecture on SAEM, then another entire lecture on SAEM and Contrastive-Divergence learning)
Readings: Section 12.3 (allow 3 lectures for this Module)

- Example 12.3.1 Robot Control Problem
- Example 12.3.2 Active Learning
- Example 12.3.3 Stochastic Approximation (Monte Carlo) EM

Module 46: Investigating Generalization Performance: What are Conditions for Convergence of Empirical Risk Minimizers as Sample Size Increases? Example Analyses.
Readings: Section 13.1

- Conditions for Ensuring Convergence of Risk Minimizers (Recipe Box 13.1)
- Maximum Likelihood Estimation
    - Real-Valued Targets
    - Binary Targets
    - Categorical Targets
- Cross-Entropy Minimization is Maximum Likelihood Estimation
- Conditions Ensuring Log-Likelihood Empirical Risk Functions are Convex
- Maximum A Posteriori Estimation
    - Parameter Priors and Hyperparameters

  - ML estimation is Special Case of MAP estimation
  - MAP estimation is Special Case of Bayes Risk estimation
- MAP risk function Design (Recipe Box 13.2)
- Constructing a Probabilistic Model for an Empirical Risk Function Minimization Learning Machine (Recipe Box 13.3)

Module 47: Investigating Generalization Performance: Sampling Error Analysis using Simulation Methods
How can we use simulation methods to estimate "confidence intervals" on predictions?
How can we use simulation methods to estimate "error bars" for our estimates of accuracy, precision, and recall?
How can we use simulation methods to compare models?

- K-Fold Cross Validation
- Estimating Statistic Distributions using Unlimited Data
- Estimating Statistic Distributions using Bootstrap Data
  - Nonparametric Bootstrap
  - Parametric Bootstrap
  - Mean Statistic as Minimum Mean Square Bayes Risk Estimator
  - Standard Error of Statistic
  - Simulation Error of Mean Statistics

Module 48: Sampling Error Analysis using Analytic Methods
How can we derive formulas to estimate "confidence intervals" on predictions?
How can we derive formulas to estimate "error bars" for our estimates of accuracy, precision, and recall?

- Assumptions for Asymptotic Analysis (Section 15.1)
- Theoretical Sampling Distribution Analysis (Recipe Box 15.1)
- Confidence Regions (Theorem 15.3.1 with example introducing theorem)
- Wald Tests (Recipe Box 15.3)
- Bayesian Hypothesis Testing

Module 49: Model Selection and Evaluation
How can derive formulas to compare models? What are important model selection criteria which are commonly used?

- Cross-Validation Model Selection Criterion (Recipe Box 16.1)
- BIC/XBIC Model Selection Criterion (Recipe Box 16.2)
- Model Misspecification Detection Model Selection Criterion (Recipe Box 16.3)