Midterm Review

Fall 2023

University of Toronto

Outline

How should I study?

- For each kind of learning model we have covered in isolation, make sure you are able to effectively understand (mathematically) the following concepts:
- **Model:** What assumption does the mode make about the relationship between data and outcomes? What is the hypothesis space of the classifier? What are the hyperparameters?
- **Optimization:** How do we learn the model given a labelled dataset? How might you characterize the optimization problem?
- **Generalization:** What practical strategies do we use in order to ensure that the predictive models we learn generalize well?

What kinds of questions can I expect?

The goal of the midterm is to test whether the knowledge you have gained in the first half of the course will *generalize*!

- True/False - with explanations,
- Deriving mathematical expressions (review homework exercises with your study groups!),
- Calculating probabilistic quantities (information gain, zero-one loss, sensitivity),
- Reasoning about the expected behavior of learning systems described in the question,
- Remember that you have a cheat sheet to leverage (Save the cheat sheet, and re-use for the final!).
- ...

Managing time

It may have been a while since you had a (in-person) midterm.

- Read the midterm end to end in one go,
- Answer the questions you are comfortable with *first*,
- Don't get nervous if you encounter a hard question:
  - Break up the question into smaller pieces,
  - Solve each piece in isolation to put together a solution.
- Good luck!

Types of learning

- Supervised learning: Learning a model using a collection of training examples labeled with the correct outputs,
- Unsupervised learning: Learning without labeled examples (more about this in the second half of the class).

We've seen two kinds of supervised learning:

1. **Regression**: Predict a scalar valued target (e.g. predict the stock price next year given the past history),
2. **Classification**: Predict discrete (binary or multi-class) class as target (e.g. predict whether a fruit is a lemon or an orange).

Nearest Neighbors/K-Nearest Neighbors

- **Idea:** Classify a new input x based on its k nearest neighbors in the training set,
- **Decision Boundary:** Boundary between regions of input space associated to different categories,
- **Hyperparameter** k controls the number of neighbors used in the method; need to understand how it controls and trades off underfitting/overfitting,
- Pitfalls:
  - Curse of dimensionality,
  - Sensitivity to data scales,
  - Memory/computational cost.

Decision Tree

- **Model:** Make predictions by splitting on features according to a tree,
- **Decision Boundary:** Axis aligned planes,
- **Learning algorithm:** Greedy algorithm based on picking features that maximize information gain:
  - Entropy $H(Y) = -\sum_{y \in \mathcal{Y}} p(y) \log p(y)$
  - Information Gain $IG(Y|X) = H(Y) - H(Y|X)$ measures the informativeness of a split,

Linear Regression

- **Model:** Linear function of features $y = w^T x + b$, The parameter b is often subsumed into w by padding x with 1,
- **Loss function:** $L(y, t) = \frac{1}{2}(y - t)^2$ for a single example. The *averaged loss function* (over the training examples) is what we use for learning,
- **Vectorization:** Rewriting loops as dot products to leverage efficient hardware implementations of matrix multiplication,
- **Learning algorithm:** Minimize averaged training error via Direct solution, Gradient Descent, Stochastic Gradient Descent, or Batch Gradient Descent,
- **Feature mapping:** Mapping features onto a different space where data is linearly separable,

Linear Classification

- **Binary Linear Classification:** Linear function of features $z = w^T x + b$ coupled with a decision rule $y = \mathbb{I}[z \geq 0][1] + \mathbb{I}[z < 0][0]$,
- **Linear separability:** When is data linearly separable, how can you identify scenarios
- **Geometry:** Input space, weight space,
- **Logistic Regression:** Linear function of features $z = w^T x + b, \;\; y = \sigma(z), \;\;$,
- **Loss function:** ~~$0-1$~~/~~Squared Error~~/Logistic~~+Squared Error~~
  Binary Cross Entropy Loss: $\mathcal{L}_{CE} = -t \log y - (1-t) \log(1-y)$,
- **Learning algorithm:** Minimize averaged training error via gradient based methods,

Multi-Class Classification

- Targets belong to a discrete set: $\{1, 2, \ldots, K\}$.
- Represented as a one-hot vector $t = (0, \ldots, 0, 1_k, 0, \ldots, 0) \in \mathbf{R}^K$ where exactly one entry is 1.
- **Model:** $z = Wx + b$, $y_k = \text{softmax}(z_1, \ldots, z_k)_k = \frac{e^{z_k}}{\sum_{k'} e^{z_{k'}}}$
- **Loss function:** $\mathcal{L}(\mathbf{y}, \mathbf{t}) = -\sum_k t_k \log y_k = -\mathbf{t}^T (\log \mathbf{y})$

Neural Networks

- Model: $y = f^L \circ \ldots \circ f^1(x)$
- Hidden unit, layer, weight, activation function, each layer is a feature detector,
- **Expressivity:** Universal function approximators (non-linear activation functions),
- **Regularization:** Early stopping,
- **Learning:** Backpropagation algorithm.

Model complexity and generalization

- **Underfitting:** too simplistic to describe the data,
- **Overfitting:** too complex, fits training data perfectly but does not generalize,
- **Hyperparameter:** can't include in training process itself, tuned using validation set,
- **Regularization:** Used to describe preferences of hypotheses to search over, $L1/L2$,
- **Bayes optimality:** Bayes error due to inherent unpredictability of targets, an algorithm that achieves the Bayes error is Bayes optimal,
- **Bias Variance Decomposition:** What is it, why is it useful, how do you calculate the bias/variance for different choices of loss functions,

Miscellany

- Comparisons between different kinds of classifiers,
- Contrasting decision boundaries between different models,
- Convexity, using convexity in proofs,
- Bagging and ensembles,
- Drawing computation graphs and using backpropagation.

2018 Midterm Version A Q7

7. **[2pts]** Consider the classification problem with the following dataset:

x_1	x_2	x_3	t
0	0	0	1
0	1	0	0
0	1	1	1
1	1	1	0

Your job is to find a linear classifier with weights w_1, w_2, w_3, and b which correctly classifies all of these training examples. None of the examples should lie on the decision boundary.

(a) **[1pt]** Give the set of linear inequalities the weights and bias must satisfy.

(b) **[1pt]** Give a setting of the weights and bias that correctly classifies all the training examples. You don't need to show your work, but it might help you get partial credit.