CS 480 Course Notes

Introduction to Machine Learning

Michael Socha

University of Waterloo

Winter 2019

Contents

1 Course Overview

This is an applied introductory machine learning course covering the basics of machine learning algorithms and data analysis. Topics covered include:

- Regression analysis
- Probabilistic modeling
- Support vector machines
- Supervised vs unsupervised learning
- Reinforcement learning
- Neural networks

2 Introduction - What is Machine Learning?

Machine learning is the field of study of how computers can improve their performance at tasks (i.e. learn) without being explicitly programmed to do so. Machine learning can be useful for tasks for which it is difficult to write a step-by-step imperative program. Sample applications include optical character recognition, computer vision, and game playing.

2.1 Learning Frameworks

- **Supervised Learning:** Goal is to learn a function based on its input and output (e.g. determining if an email is spam based on a set of emails labeled as spam or not spam).
- **Unsupervised Learning:** Goal is to learn a function based on its input alone (e.g. organizing data into clusters).
- **Reinforcement Learning:** Goal is to learn a sequence of actions that maximize some notion of reward (e.g. learning how to control a vehicle to perform some maneuver).

2.2 Challenges

Some of the challenges facing machine learning today are:

- Dealing with large amounts of data (algorithm complexity and distributed computing become very relevant)
- Generating reproducible results
- Challenges concerning real-world adoption of work (e.g. human computer interaction, robustness, ethical concerns)

3 Decision Trees

Decision trees contain questions as nodes and answers as edges, and serve to guide to an answer based on a set of observations.

3.1 Predictions Using Decision Trees

Consider a data set of employees and their job satisfaction, where we use the data set to predict whether an employee is satisfied with their job. An employee can have many features, such as age, salary, seniority, working hours. It is infeasible to build all possible decision trees when there are many features due to the exponential growth of the tree. Instead, for predictive purposes, it makes sense to focus on those questions that are informative to the prediction. For example, if there is no major difference in job satisfaction based on an employee's age, then it is unnecessary to include age in a decision tree, since the answer is not informative for the prediction.

In general, supervised learning problems have a set of possible inputs X, an unknown target function $f : X \to Y$, and a set of function hypotheses $\{h | h : X \to Y\}$. Supervised learning algorithms accept training examples (a pair (x, y) where $x \in X$, $y \in Y$) and output a function hypotheses that approximates f. When using decision trees for learning, each decision tree is a function hypothesis.

3.2 Encoding Functions in Decision Trees

Boolean functions can be fully expressed in decision trees, with one branch for true and another for false. Other function can be approximated as a boolean function. For example, instead of a node asking what an employee's salary is, it could ask whether the salary is above a certain amount.

3.3 Training and Testing

The key idea behind decision tree generation algorithms is to grow a tree until it correctly classifies all training examples. The rough procedure followed is:

1. If all training data has the same class, create a leaf node and return.
2. Else, create the best (i.e. most informative) node on which the split the data.
3. Split the training set over the above node.
4. Continue the procedure on each subset of training data generated.

3.3.1 Information Content

Criteria for finding the "best" split of data can be defined mathematically. If event E occurs with probability $P(E)$, then when E occurs, we receive $I(E) = \log_2 \frac{1}{P(E)}$ bits of information. This can be interpreted as that less likely events yield more information.

3.3.2 Entropy

An information source S which emits results $s_1, s_2, ..., s_i$ with probabilities $p_1, p_2, ..., p_i$ produces information at $H(S) = \sum_i^k p_i I(s_i)$. $H(S)$ is known as the information entropy of S. Information entropy can vary between 0, which indicates no uncertainty (i.e. all members of S in same class), and 1, which indicates high uncertainty (i.e. equal probability of all classes). The best split of data for classification is one that maximally decreases its entropy (a concept known as information gain).

3.4 Overfitting

A hypothesis $h_1 \in H$ is said to overfit training data if there is some alternative hypothesis $h_2 \in H$ that has a larger error over the training data but a smaller error over a larger set of inputs. Overfitting can occur due to errors in a data set or just due to coincidental irregularities (especially in a small dataset). Overfitting can be avoided by removing nodes with low information gain, either by stopping decision tree construction early or by pruning such nodes after the tree is constructed.

3.5 Inductive Bias

Inductive bias refers to the assumptions made about the target function to predict future outputs. Common inductive biases for decision trees are:

- Assumption that simplest hypothesis is the best (i.e. Occam's razor).
- Decision trees with information gain closer to the root are considered better.

These are examples of preference bias, which influence the ordering of the hypothesis space. This is distinct from restriction bias, which limits the hypothesis space.

3.6 Advantages and Disadvantages

Decision trees are good for:

- Ease of interpretation
- Speed of learning

Limitations of decision trees include:

- High sensitivity, with tree output changing significantly due to small changes in input.
- Not good for learning data sets without axis-orthogonal (i.e. constant in at least 1 dimension) decision boundaries.

4 Evaluation of Learning

4.1 Performance Formulation

Let $\hat{y}$ be an output generated by a function f approximating some target function. Let y be the corresponding output of the target function. A loss function $l(y, \hat{y})$ can be used to measure the accuracy of the approximation function f. Some common loss functions include:

- Squared Loss: $l(y, \hat{y}) = (y - \hat{y})^2$
- Absolute Loss: $l(y, \hat{y}) = |y - \hat{y}|$
- Zero/One Loss: $l(y, \hat{y}) = 1_{y \neq \hat{y}}$

We assume that the data coming from our target function comes from some probability distribution D, and that our training data is a random sample of (x, y) pairs from D. A Bayes Optimal Classifier is a classifier that for any input x, returns the y most likely to be generated by D.

Based on the available training data, the goal of supervised learning is to find a mapping f from x to y such that generalization error $\sum_{(x,y)} D(x, y) l(y, f(x))$ is minimized. However, since D is unknown, we instead estimate the error from the average error in our training or test data, which is $\frac{1}{N} \sum_{n=1}^{N} l(y_n, f(x_n))$.

4.2 Common Learning Challenges

Some common challenges in learning include:

- Inductive bias of the algorithm being distant from the concept actually being learned.
- The data itself having challenging characteristics, such as lots of noise, ambiguity, or missing information.

4.3 Training vs Validation vs Test Sets

Models are initially built based on a training dataset. Test sets (also known as holdout sets) are then used to estimate the generalization error. Validation sets are also used to measure the model's performance, but unlike test sets, validation sets can make changes to the model's parameters.

4.4 Bias-Variance Decomposition

Many machine learning algorithms are based on building a formal model based on the training data (e.g. a decision tree). Models have parameters, which are characteristics that can help in classification (e.g. a node in a decision tree). Models may also have hyper-parameters, which in turn control other parameters in a model (e.g. max height of decision tree).

Generalization errors result from a combination of noise, variance, and bias. Bias concerns how well the type of model fits the data. Models with high bias pay little attention to training data and suffer from underfitting, while models with low bias may pay too much attention to training data and become overfitted. Bias and variance tend to be at odds with one another (high bias typically leads to low variance, and vice versa). For example, a decision tree that makes the same prediction for all input has high bias and low variance, while a decision tree trained to return a correct prediction for each point of training data will have low bias and likely high variance.

4.5 Cross Validation

Cross validation is a technique for measuring how well a model generalizes. The idea behind it is to break up a training data set into K equally sized partitions, and use $K-1$ of the partitions as training data and the remaining partition for testing. This should be repeated K times, so that all points of data are at some point used for testing. Higher values of K lower the amount of variance of in the error estimation. To avoid training and testing data having a different probability distribution, the data should be shuffled before being split.

4.6 Bootstrapping

Bootstrapping is an alternative to cross validation where instead of dividing a training data set into partitions, a random sample of points (with possible duplicates) is used as training data. The remaining points are then used as testing data, with the goal being similar to that of cross validation.

4.7 Avoiding Overfitting

Overfitting often occurs when training data has many features, which allows for an approximation of the target function with many degrees of freedom. Overfitting can be avoided by only considering features with a strong correlation to the output. Cross-validation could be applied as follows:

1. Divide training data into K groups at random.
2. Within each group, find a small set of features with strong correlation to the output. Running this step for each group individually instead of for the entire training set further helps avoid overfitting.
3. Build a classifier using the features and examples from the $K-1$ groups.
4. Use the classifier to predict the examples and in group K and measure the error.

5. Repeat 2-4 to produce an overall cross-validation estimate of the error.

Bootstrapping can be applied in a similar way to avoid overfitting.

4.8 Performance Evaluation of Classifiers

Consider the following terminology for classification problems:

- True positive (TP) - Examples of class 1 predicted as class 1
- False positive (FP) - Examples of class 0 predicted as class 1 (Type 1 Error)
- True negative (TN) - Examples of class 0 predicted as class 0
- False negative (FN) - Examples of class 1 predicted as class 0 (Type 2 Error)

4.8.1 Accuracy and Error

The following formulas can be used to measure accuracy and error:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$ErrorRate = \frac{FP + FN}{TP + TN + FP + FN}$$

4.8.2 Precision and Recall

Precision and recall can be measured as follows:

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

Precision measures the ratio of positive predictions that were correct, while recall measures the ratio of total positive instances that were predicted. Similarly to how variance and bias are often at odds with one another, so are precision and recall.

4.8.3 F-Measure

An F-measure (also known as a F1 score) measures a model's accuracy by taking into account both precision and recall as follows:

$$F = \frac{2PR}{P + R}$$

To adjust the relative importance of precision vs recall, a weighted F-measure can be used, which is defined as follows:

$$F = \frac{(1 + \beta^2)PR}{\beta^2 P + R}$$

In a standard F-measure, $\beta = 1$, $\beta < 1$ means that precision is valued over recall, while $\beta > 1$ means recall is valued over precision.

4.8.4 Sensitivity and Specificity

Sensitivity is the same measure as recall. Specificity is a measure of how well a classifier avoids false positives, and is measured as:

$$Specificity = \frac{TN}{TN + FP}$$