Contents

1 Intro

1.1 Lecture 1

1.1.1 Introduction

What is this course? This is the big picture.

- Find patterns in data; using them to make predictions.
- Models and statistics help us understand patterns.
- Optimization algorithms "learn" the patterns.

The main pattern we will look at in machine learning is **classification**. Classification is often linked to decision-making–deciding the "class" that a certain datapoint belongs to. Binary classification is a special type of classification, where we only have to decide between two options.

We consider the following example.

Example 1.1
Suppose a credit card company wants to figure out if a customer is likely to default on their credit card debt. You would take the following steps.

1. Collect (labeled) training data: reliable debtors and defaulting debtors.
2. Evaluate new applicants (predictions).

The data collected will have certain "features." For example, features in this case it might be bank balance and income. The space of all possible values the features can take is called a feature space.

Definition 1.1 (Decision Boundary)
A decision boundary is a curve which divides the feature space into regions in a binary classification problem. On one side of the decision boundary, everything is predicted as one class, and on the other side, everything is predicted as the other class.

The simplest decision boundary is a line or a hyperplane (a line in higher dimensions), called a **linear classifier**. However, there are more types of classifiers that you can make.

A **nearest-neighbor classifier** predicts the class of a point p by finding its closest point q in the training data and assuming q's class is the same as p. This sounds great, since 100% of our training data will be correctly predicted! However, it is very jagged and random [TODO: Insert Figure]. This is evidence of **overfitting**. We believe that this will not lead to good predictions on a real.

Instead, an approach more robust to outliers is k-nearest neighbors, where you look at the k closest points to p and take a majority vote out of these k to decide the class of p.

1.1.2 Training, Testing, and Validation

Still, how do we know that we are not overfitting to our data? We go through the following process initially.

Note 1.1
Making a learning model:

1. Train a classifier: make it learn to distinguish one class from another.

2. Test the classifier on new data samples, preferably once as a final evaluation, after tuning hyperparameters.

Definition 1.2 (Errors)
Thus, we can talk about two types of useful measures of error.

- Training set error is the fraction of training images not classified correctly.
- Test set error is the fraction of misclassified new images, not seen during training.

Our example of a 1-nearest neighbor classifier has 0% training set error but is likely to have a sizable test set error! This is due to anomalies in the data, called **outliers**.

Definition 1.3 (Overfitting)
Overfitting is when the test error deteriorates because the classifier becomes too sensitive to outliers or other spurious patterns.

Most ML algorithms have a few **hyperparameters** that control over/underfitting, like k in the k-nearest neighbors. How do we find the optimal hyperparameter or even the right model? By using validation.

What we do is hold back a subset of the labeled training data, called the validation set. We train the classifier multiple times with different hyperparameter values. We then use the validation set to choose the setting that works the best. Now, we have three datasets that we want to use.

Generally we divide techniques into two broad categories.

- Supervised Learning - labeled training data
  - Classification - discrete class labels
  - Regression - estimating continuous parameters
- Unsupervised Learning - unlabeled training data
  - Clustering - looking for distinguishable groups in the data
  - Dimensionality Reduction - taking a large-dimensional feature space and extracting the key differences between the data