CS 578 Class Notes

Weiyi Zhou (zhou587)

January 14, 2019

1 Course Project

- Proposal due 1/27
- No page requirement, use ¿ 11 pt. font
- Submit proposal through CMT

1.1 Requirements/Goals

- Try Machine Learning: We learn ML by doing
- Demonstrate basic scientific training: important to know how to identify important questions, justify questions, explain thought process, etc.

1.2 Project Ideas

- Apply ML to fields other than CS
- Extend ML algos covered in class: Taking SVM from -1 or 1 output to a richer output space
- Exploring a specific "module" in ML: a sampling technique, or a specific type of ML like reinforcement learning

1.3 Evaluation metrics (NSF)

- Intellectual merit: How a proposal advances our understanding of ML (my personal understanding of ML for the purposes of this class)
- Broad impact: Ensure proposal/idea benefits others or makes the world a better place, otherwise difficult for other people to care
- Tractability: Be practical (Is there an available dataset? Simpler is better)

2 Inductive Reasoning (Review from 1/9)

Features	Predictions
x_i	$h(x)$ (-1 or 1)
$x = (x_{1_1}, x_{1_2}, ...)$	-1
x_2	+1
.	.
.	.
.	.
x_n	?

- Inductive reasoning - making predictions based on observations, i.e. attempting to generalize from specific cases
- Above table an example of inductive reasoning. We are given data in the form of pairs of data vectors (x_i, also called features) and the correct predictions for each data vector ($h(x)$). We then must make a prediction for the new data vector x_n based on the examples we have already observed
- ML = using machines to do inductive reasoning

3 The ML Pipeline

3.1 Basic Terminology

- Function Mapping - function that maps inputs to outputs: Ex. $h : x \rightarrow h(x) \in \{+1, -1\}$ for a function with binary labels
- Features, x - input, sometimes given as a vector of features, ex. $\{gender, age, height\}$
- Labels, $h(x)$ - output, predictions
- Hypothesis space - family of functions that the ML algorithm searches in to find the function that best maps x to $h(x)$
- Data - series of pairs consisting of a feature (vectors) and the corresponding label: $D = \{(x_1, y_1), (x_2, y_2), ...\}$
- Loss Function - determine how close a predicted label is to the true label. Example: $L(h(x), y) = \mathbb{1}(h(x) \neq y)$ would be a loss function that evaluates to 1 if the predicted label is equal to the true label or 0 otherwise

3.2 Testing

- Training set, D_{train} - dataset used to train and tweak model
- Using the example loss function of $L(h(x), y) = \mathbb{1}(h(x) \neq y)$, the total training loss can be represented by $\sum_{i \in D_{train}} L(h(x), y)$. We want

to minimize this loss value in order to find the function mapping h s.t. $\min_{h \in H} \sum_{i \in D_{train}} L(h(x), y)$, i.e. the function mapping that predicts labels closest to the true labels

- The problem with min expression in the previous example is overfitting. Consider the following:

$$h(x) = \begin{cases} y_i & \text{if } x \text{ is in train} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

  This function will have a very low loss on D_{train}, but will have very poor performance in reality, since it will always predict a 0 if the datapoint is not in the training set. This is known as overfitting.

- A better approach is to evaluate our function mapping h on a dataset of x, y that is drawn from the same underlying distribution as the training set, but does not contain the exact same points as the training set. This is known as the test set. We then seek to find is $\min_{h \in H} E_{x,y \in Pr(x,y)} L(h(x), y)$

- However, it is important that we do not make changes to the model based on our test set results. If we train the model on the training set, test the model on the test set, and get a poor result, we cannot then tweak the model in order to get a better test set result. This creates an implicit feedback loop that will create overfitting problems once again

- Validation Set, D_{val} - Separate set from testing and training set. We use this set to tweak/modify our model parameters. Once we are completely done changing our model, we can use the testing set to get our final performance. Note that once we run our model on the test set, we are no longer allowed to make any changes to the model, or we will cause overfitting.

- ML Recipe
  - For each parameter
    - Train on D_{train}
    - Evaluate on D_{val}
  - Select parameters with best performance on D_{val}
  - Find best model on the selected parameters using D_{train} and D_{val}
  - Test model on D_{test}