Lecture 3

by

Raghavendra Singh

IIT Delhi

Oct 10, 2016

Outline

1. Machine Learning
2. Docker

Basics

Definition:

- A machine learning algorithm is an algorithm that is able to learn from data.
- A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E
- Machine learning tasks process data where each data point is a set of *features*

Machine Learning Tasks

1. Classification
   - output which of k-categories input belongs to

$$f(x) : R^n \rightarrow \{1..k\}$$

   - Classification Task ILSVRC 2011
2. Regression
   - output a real value given input

$$f(x) : R^n \rightarrow R$$

   - Deep neural network for object detection
3. Structured Output
   - output is a vector, collection, with important relationships between the different elements.

Structured Output

- Transcription
  - OCR, Speech2Text
  - Multi digit recognition on Street View Imagery
- Machine Translation
  - Neural Machine Translation
- Pixel wise Segmentation
  - every pixel belongs to a category ? foreground, background
  - Deep Image Segmentation
- Image Captioning task
  - words in caption have a relationship amongst themselves and forms a valid sentence
  - Show and Tell

Other Tasks

1. Anomaly Detection
   - sifts through a set of events and flags some of them as atypical
     - credit card fraud detection – model your transactions and flag unlikely transactions
2. Sampling and Synthesis
   - generate new examples that are similar to given training data
     - Texture nets
3. Denoising and Compression
   - Missing data, corrupted data
     - Image compression

Other Tasks

1. Density estimation
   - Explicitly estimate the pdf (pmf) that the training samples are drawn from
   - Implicitly this happens in all above tasks
     - Generative Models

Performance Measure P

1. a quantitative measure of its performance.
2. specific to task
   - classification – accuracy: how many are predicted correctly
   - density estimation – average log probability of a set of examples
3. Measured on unseen test data – which is assumed to be from the same generative mechanism as training data

Experience E

1. Supervised experience
   - dataset containing features, and each example (x) is also associated with a label or a target (y)

$$\textit{Learn } p(y|x)$$

2. Unsupervised experience
   - dataset containing features

$$\textit{Learn } p(x)$$

3. Bayes Rule can convert conditional pdf into joint pdf and vice-versa; there is a thin dividing line between supervised and unsupervised experiences

Other Experiences

1. Multi instance learning – an entire collection of examples is labeled as containing or not containing an example of a class, but the individual members of the collection are not labeled.
   - Blue jeans
2. Semi supervised learning
3. Reinforcement learning – experience changes, and reward (label) at the *end* of experience
   - Alpha-Go

Example

X = $x_1 .....x_n$ where x_i is a m dim vector: number of features is m, number of datapoints is n
Y = $y_1 ....y_n$ y_i is label
Problem:

$$\hat{y} = \mathbf{w}x$$

$\hat{y}$ is prediction for x input
$\mathbf{w}$ is parameter matrix of dimension $mx1$
definition of our task T : to predict y from x by outputting $\hat{y}$
definition of our performance measure, P: for train set $T = (X^e, Y^e)$ minimize mean square error (MSE) between predicted labels $\hat{Y}^e$ and given labels Y^e

$$MSE(T) = \sum_i (y_i^e - \hat{y}_i^e)^2$$

Example(2)

Design an algorithm that will improve the weights **w** in a way that reduces MSE on test set T, when the algorithm is allowed to gain experience by observing a training set $R = (X^r, Y^r)$
To minimize MSE on R, we can simply solve for where its gradient is 0:

$$w = \frac{(X^r)^t Y^r}{(X^r)^t X_r}$$

Numerator is cross-correlation between data set and labels; denominator is normalizing for auto-correlation in data set

Generalization

The central challenge in machine learning is that we must perform well on new, previously unseen inputs, not just those on which our model was trained.

The ability to perform well on previously unobserved inputs is called generalization.

when training a machine learning model, we have access to a training set, we can compute some error measure on the training set called the training error, and we reduce this training error. So far, what we have described is simply an optimization problem.

Generalization (2)

Machine learning is optimizing the generalization error also.

the examples in each dataset are independent from each other, and that the train set and test set are identically distributed, drawn from the same probability distribution as each other.

One immediate connection we can observe between the training and test error is that the expected training error of a randomly selected model is equal to the expected test error of that model.

Underfitting occurs when the model is not able to obtain a sufficiently low error value on the training set. Overfitting occurs when the gap between the training error and test error is too large.

Capacity

Informally, a models capacity is its ability to fit a wide variety of functions. Models with low capacity may struggle to fit the training set. Models with high capacity can overfit by memorizing properties of the training set that do not serve them well on the test set.

while simpler functions are more likely to generalize (to have a small gap between training and test error) we must still choose a sufficiently complex hypothesis to achieve low training error.

our goal is to understand what kinds of distributions are relevant to the real world that an AI agent experiences, and what kinds of machine learning algorithms perform well on data drawn from the kinds of data generating distributions we care about.

Regularization

Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error. Preference for a kind of solution for example solution with low weight norm

Validation

we split the training data into two disjoint subsets. One of these subsets is used to learn the parameters. The other subset is our validation set, used to estimate the generalization error during or after training, allowing for the hyperparameters to be updated accordingly.

Typically, one uses about 80% of the training data for training and 20% for validation.