National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”
Institute of Physics and Technology

Lecture 17

Objects and Systems Identification Methods. Adaptive Approach

Dmytro Progonov,
PhD, Associate Professor,
Department Of Physics and Information Security Systems

Content

- Problem statement;
- Classification and regression trees;
- Random forests;
- Generalized additive models;
- Boosting:
  - Adaboost;
  - Gradient boost.

Problem statement

The kernel-based methods can work well, despite in implicitly relies on having a good kernel function to measure the similarity between data vectors. Often *coming up with a good kernel is quite difficult*.

One approach is to learn the parameters of kernel functions, by maximizing the marginal likelihood, e.g. ARD kernel. However, such methods ***can be computationally expensive***.

Another approach, known as multiple kernel learning, uses a convex combination of base kernels and then estimate the mixing weights. But this ***relies on having good base kernel***.

An ***alternative approach*** is to dispense with kernels altogether, and try to learn useful features $\phi(\mathbf{x})$ directly from the input data. That is, we will ***call an adaptive basis-function model (ABM)***:

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

Classification and regression trees (1/2)

We can write the model in the following form:

$$f(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}] = \sum_{m=1}^{M} w_m \mathbb{I}(\mathbf{x} \in \mathcal{R}_m) = \sum_{m=1}^{M} w_m \phi(\mathbf{x}, \mathbf{v}_m)$$

where $\mathcal{R}_m$ – mth region; w_m – mean response; $\mathbf{v}_m$ – encodes the choice of variable to split on, and the threshold value, on the path from the root to the mth leaf.

Recursive procedure to grow a classification/regression tree:

node.prediction = mean($y_i, i \in \mathcal{D}$) // or class label distribution

$(j^*, t^*, \mathcal{D}_L, \mathcal{D}_R) =$ split($\mathcal{D}$)

If (not worthSplitting(depth, cost, $\mathcal{D}_L, \mathcal{D}_R$))

    return node;

Else

    node.test = $\lambda \mathbf{x}. x_{j^*} < t^*$ // anonymous function

    node.left = fitTree(node, $\mathcal{D}_L$, depth+1)

    node.right = fitTree(node, $\mathcal{D}_R$, depth+1)

    return node;

Classification and regression trees (2/2)

Advantages of classification and regression trees (CART):

1. They are easy to interpret;
2. They can easily handle mixed discrete and continuous inputs;
3. They are insensitive to monotone transformations of the inputs;
4. They perform automatic variable selection;
5. They are relatively robust to outliers;
6. They scale well to large dataset;
7. They can be modified to handle missing data.

Limitations of CART:

1. They do net predict very accurately compared to other kinds of models;
2. They are unstable – small changes of inputted data can have large effects on the structure of the tree

Random forests

One way to reduce the variance of an estimate is to average together many estimates. For example, we can train M different trees on different subsets of the data, chosen randomly with replacement, and then compute the ensemble (***bootstrap aggregating*** – ***bagging***):

$$f(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} f_m(\mathbf{x})$$

where f_m – the mth tree.

The technique ***random forest*** tries to decorrelate the base learners by learning trees based on randomly chosen subset of inputted variables, as well as a randomly chosen subset of data cases.

Generalized additive models

A simple way to create a nonlinear model with multiple inputs is to use a ***generalized additive model (GAM)***:

$$f(\mathbf{x}) = \alpha + f_1(x_1) + f_2(x_2) + \cdots + f_D(x_D)$$

Here each f_i can be modeled by some scatterplot smoother, and $f(\mathbf{x})$ can be mapped to $p(y|\mathbf{x})$ using a link function, as in GLM.

Backfitting of GAM using MLE:

1. Estimate $\hat{\alpha} = \frac{1}{N}\sum_{i=1}^{N} y_i$;
2. Subtracting $\hat{\alpha}$ from responses and iteratively update each f_i:

$$\hat{f}_i = smoother\left(\left\{y_i - \sum_{k \neq i} \hat{f}_k(x_{ik})\right\}_{i=1}^{N}\right)$$

3. Ensure the output is zero mean – $\hat{f}_i = \hat{f}_i - \mathbb{E}\left[\hat{f}_i\right]$

Boosting (1/2)

Boosting is greedy algorithm for fitting adaptive basis-function models of the form

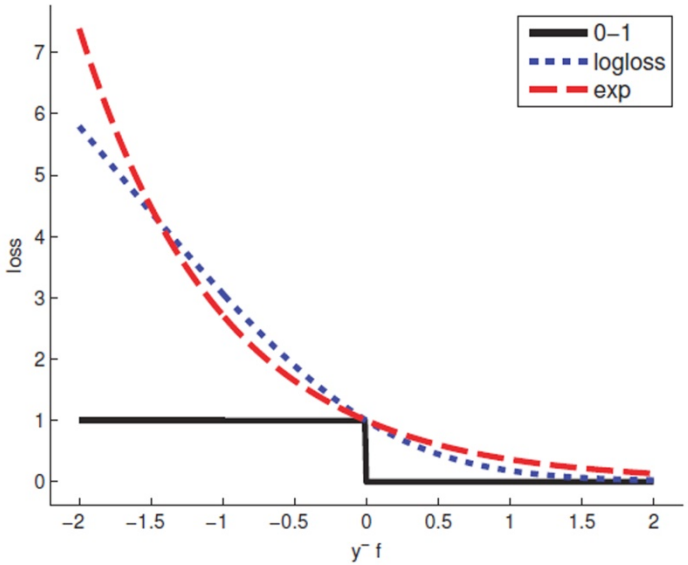
$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

where each ϕ_m are generated by algorithm called a ***weak learner*** or ***base learner***.

Boosting can be interpreted as a form of gradient descent in function space (Breiman, 1998).

Boosting (2/2)

Name	Loss	Derivative	f^*	Algorithm
Squared error	$\frac{1}{2}(y_i - f(\mathbf{x}_i))^2$	$y_i - f(\mathbf{x}_i)$	$\mathbb{E}[y\|\mathbf{x}_i]$	L2Boosting
Absolute error	$\|y_i - f(\mathbf{x}_i)\|$	$sgn\,[y_i - f(\mathbf{x}_i)]$	$median[y\|\mathbf{x}_i]$	Gradient boosting
Expon. loss	$\exp[-\hat{y}_i f(\mathbf{x}_i)]$	$-\hat{y}_i \exp[-\hat{y}_i f(\mathbf{x}_i)]$	$\frac{1}{2}\log\left[\frac{sigm(2f(\mathbf{x}_i))}{1-sigm(2f(\mathbf{x}_i))}\right]$	AdaBoost
Logloss	$\log\left[1 + e^{-\hat{y}_i f(\mathbf{x}_i)}\right]$	$\hat{y}_i - sigm(2f(\mathbf{x}_i))$	$\frac{1}{2}\log\left[\frac{sigm(2f(\mathbf{x}_i))}{1-sigm(2f(\mathbf{x}_i))}\right]$	LogitBoost



Adaboost

AdaBoost for binary classification with exponential loss:

1. Compute $w_i = 1/N$;
2. For $m = 1{:}\,N$ do:
   1. Fit classifier $\phi_m(\mathbf{x})$ to the training set using weights $\mathbf{w}$;
   2. Compute $err_m = \frac{\sum_{i=1}^{N} w_{i,m} \mathbb{I}(\tilde{y}_i \neq \phi_m(\mathbf{x}_i))}{\sum_{i=1}^{N} w_{i,m}}$;
   3. Compute $\alpha_m = \log[(1 - err_m)/err_m]$;
   4. Set $w_i = w_i \exp[\alpha_m \mathbb{I}(\tilde{y}_i \neq \phi_m(\mathbf{x}_i))]$;
3. Return $f(\mathbf{x}) = sgn[\sum_{m=1}^{M} \alpha_m \phi_m(\mathbf{x})]$.

Gradient boost

Gradient boosting for binary classification:

1. Initialize $f_0(\mathbf{x}) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{N} L(y_i, \phi(\mathbf{x}_i, \boldsymbol{\gamma}))$;
2. For $m = 1{:}\,N$ do:
   1. Compute the gradient residual using $r_{im} = -\left[\frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right]_{f(\mathbf{x}_i)=f_{m-1}(\mathbf{x}_i)}$;
   2. Use the weak learner to compute $\boldsymbol{\gamma}_m$ which minimizes $\sum_{i=1}^{N}(r_{im} - \phi(\mathbf{x}_i, \boldsymbol{\gamma}_m))^2$;
   3. Update $f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}_i) + \nu\phi(\mathbf{x}_i, \boldsymbol{\gamma}_m)$;
3. Return $f(\mathbf{x}) = f_M(\mathbf{x})$.

Conclusion

- Problem statement of adaptive approach to object and system identification was shown;
- Classification and regression trees were considered;
- Generalized additive models was presented
- Core principles of Adaboost and Gradient boost methods were described.