Machine Learning with MATLAB

Train Models in Learner Apps
- Import data from workspace or file
- Enable PCA for feature reduction
- Select among all common algorithms
- Export model to further optimize or integrate with other code
- Visually assess model performance

Algorithm
<table>
<thead>
<tr>
<th>Model</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision tree</td>
<td>c,r</td>
</tr>
<tr>
<td>Linear model</td>
<td>c,r</td>
</tr>
<tr>
<td>Support vectors</td>
<td>c,r</td>
</tr>
<tr>
<td>Gaussian kernel</td>
<td>c,r</td>
</tr>
<tr>
<td>Ensembles</td>
<td>c,r</td>
</tr>
<tr>
<td>K-nearest neighbor</td>
<td>c</td>
</tr>
<tr>
<td>Discrim. analysis</td>
<td>c</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>c</td>
</tr>
<tr>
<td>Gaussian process</td>
<td>r</td>
</tr>
<tr>
<td>(Gen.) Linear regression</td>
<td>(g)lm</td>
</tr>
<tr>
<td>Nonlin. regression</td>
<td>nlm</td>
</tr>
</tbody>
</table>

Machine Learning Functions

Namining Convention
fit + classification / rgression + model
e.g., for SVM classifier \( m = \text{fitcsvm}(X,Y) \)

Feature Selection
- Neighborhood Component Analysis
  - Automate identifying the features with predictive power.
  - `fscnca(X labels, 'Lambda',...);`
  - `find(mdl.FeatureWeights > 0.01)`
- Also available:
  - PCA
  - Sparse filtering
  - Matrix factorization
  - Stepwise regression
  - Reconstruction ICA
  - t-SNE

Hyperparameter Tuning
- Explore and change parameters in app:
- Automated Bayesian Optimization
  - Leverage Bayesian model to decide which points in the hyperparameter space to try next. Much faster than grid search.
  - `mdl = fit(X,labels, 'OptimizeHyperparameters','auto');`

Deploy
- Standalone, Web Apps, Spark
  - Share as standalone, MapReduce, and Apache Spark™ applications; web apps; and Microsoft® Excel® add-ins.
  - Integrate with Enterprise IT/OT
  - Convert into C/C++, Java®, .NET, or Python® library using MATLAB Compiler SDK™.
- C-Code Generation
  - Automatically convert to C/C++ code for embedded deployment using MATLAB Code™
  - 1. Train model \( Mdl = \text{fitcsvm}(X,Y); \)
  - 2. `saveCompactModel(Mdl,'mySVM');`
  - 3. Define entry-point function
    - `function label = predictSVM(x)`
    - `m = loadCompactModel('mySVM');`
    - `label = predict(m,x);`
  - 4. Generate C code
    - `codegen predictSVM -args {X}`

Learn more: mathworks.com/machine-learning

mathworks.com