Machine Learning with H2O on HAL
Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.
This is an Interdisciplinary Field

- **Domain Expertise**
- **Data Science**
- **Mathematics**
- **Computer Science**
- **Statistical Research**
- **Data Processing**
- **Machine Learning**
But Why Now?

- **Hardware**: high-performance GPUs, TPUs
- **Datasets**: large datasets collected from internet
- **Algorithmic advances**: activation functions, optimization schemes.
Introduction of H2O

• What is H2O.ai?
  • H2O.ai is the company behind open-source Machine Learning (ML) products like H2O, aimed to make ML easier for all.

• What is H2O?
  • An open source, Java-based, in-memory, distributed, ML and predictive analytics platform allowing you to build and productionize ML models.
  • Contains supervised and unsupervised models in R and Python, as well as a simple to use web-UI called Flow.
H2O Flow
Common Machine Learning Algorithms

• Machine Learning has 3 main functions: classification, prediction, clustering.

• Big 3 Basic Algorithms
  • K-Nearest Neighbor
  • Linear Regression
  • K-Mean Clustering

• Other Common Algorithms
  • Decision Tree / Random Forest / Naive Bayes / Support Vector Machine
• An object is classified by a majority vote of its neighbors
  • One of the simplest classification algorithm.
  • Often used in classification.
  • Computed from a simple majority vote of the nearest neighbors of each point.
  • K is constant specified by user.
  • KNN is computationally expensive.
K Nearest Neighbors

• How do we choose the factor K
  • Choosing K could be a challenge.
  • Boundary becomes smoother with increasing value of K.
Pros and Cons of KNN

Pros

- It is beautifully simple and logical

Cons

- It may be driven by the choice of K, which may be a bad choice.
- Generally, larger values of K reduce the effect of noise on the classification, but make boundaries between classes less distinct.
- The accuracy of the algorithm can be severely degraded by the presence of noisy or irrelevant features.
- It is important to review the sensitivity of the solution to different values of K.
Linear Regression

Example: Predict house's price using linear regression

Suppose we have a dataset giving the living areas and prices of 21,613 houses from House Sales in King County, USA. Given data like this, we can learn to predict the prices of other houses in King County.

Data set

<table>
<thead>
<tr>
<th>Living Area (Feet²)</th>
<th>Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1180</td>
<td>221,900</td>
</tr>
<tr>
<td>2570</td>
<td>538,000</td>
</tr>
<tr>
<td>770</td>
<td>180,000</td>
</tr>
<tr>
<td>1960</td>
<td>604,000</td>
</tr>
<tr>
<td>1680</td>
<td>510,000</td>
</tr>
<tr>
<td>5420</td>
<td>1,225,000</td>
</tr>
<tr>
<td>1715</td>
<td>257,500</td>
</tr>
<tr>
<td>1060</td>
<td>291,850</td>
</tr>
<tr>
<td>1780</td>
<td>229,500</td>
</tr>
<tr>
<td>1890</td>
<td>323,000</td>
</tr>
<tr>
<td>3560</td>
<td>662,500</td>
</tr>
<tr>
<td>1160</td>
<td>468,000</td>
</tr>
<tr>
<td>1430</td>
<td>310,000</td>
</tr>
<tr>
<td>1370</td>
<td>400,000</td>
</tr>
<tr>
<td>1810</td>
<td>530,000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Based on this regression line, we can predict the price for a house with a certain size of living area.

Regression line


Actually linear regression model guesses a hypothesis function $h(x)$ from training set. Given a training set, the model learns a function $h(x)$ which can be used for predicting the new/unseen data.

Training set

- $x$ is Input (feature)
- $y$ is Output (target variable)

Learning Algorithm

- input $x$ (living area of house)
- output $\hat{y}$ (predicted price of house)

Regression line

Size of living area = 4876 feet²

$\hat{y} = \beta_0 + \beta_1 x$

The regression line $h(x)$ is a hypothesis/function.
Linear Regression

• The goal of linear regression is to find the best fit line.
• minimizes the sum of the “squared differences” between the points and the regression line.

\[ h(x) = \theta_0 + \theta_1 x \]

How to find the appropriate parameter \( \theta_0 \) and \( \theta_1 \) in order to minimize the error – i.e. cost function \( J(\theta_0, \theta_1) \)

To minimize \( J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \)

Cost function (also known as Loss Function)

• \( m \) is number of training instances
• \( \hat{y}_i \) (y hat) is the predicted value
• \( y_i \) is the actual value

• Normal Equation (Closed Form)

It’s a method to solve for \( \theta \) analytically.

Using a direct “closed-form” equation that directly computes the model parameters that best fit the model to the training set (i.e., the model parameters that minimize the cost function over the training set).

It’s suitable for small feature set (e.g. < 1000 features).

• Gradient Descent

Using an iterative optimization approach, called Gradient Descent (GD), that gradually tweaks the model parameters to minimize the cost function over the training set, eventually converging to the same set of parameters as the first method.

Gradient Descent is better choice than Normal Equation when there are a large number of features, or too many training instances to fit in memory.
Linear Regression

• A gradient is the slope of a function at a specific point. The gradient of loss function/cost function is equal to the derivative (slope) of the curve.

Gradient Descent algorithm

The parameter are iteratively updated in the following equation:

\[ f_{\text{new}} = f_{\text{old}} - \alpha \cdot \nabla f \]

Learning rate (Step size)

Gradient \( \nabla f \) \( \cdot \) \( \alpha \) \( \cdot \) \( f \)

1. Pick a value for the learning rate \( \alpha \)
2. Start with a random point \( f \)
3. Calculate the gradient \( \nabla f \) at the point \( f \). Follow the opposite direction of gradient to get new parameter \( f_{\text{new}} \)
4. Repeat until the cost function converges to the minimum

In this example, initially the slope is large and positive. So, in the update equation, \( \alpha \) is reduced. As \( \alpha \) keeps getting reduced, notice that the gradient also reduces, and hence the updates become smaller and smaller and eventually, it converges to the minimum.
Linear Regression

• Find the best-fit line through Gradient Descent algorithm.

Iteratively find the minimum of cost function

Animation: a visual representation of gradient descent in action

https://alykhantejani.github.io/images/gradient_descent_line_graph.gif

Video: Cost Function Intuition #2 | Andrew Ng (8mins)
https://youtu.be/0kns1gXLYg4?t=2m8s
K-Mean Clustering

• Discover the structure within the un-labeled data.
• Clustering is a technique for finding similarity groups in a data, called clusters.
• It attempts to group individuals in a population together by similarity, but not driven by a specific purpose.
• Clustering is often called an unsupervised learning, as you don’t have prescribed labels in the data and no class values denoting a priori grouping of the data instances are given.
K-Mean Clustering

- A graphical view of K-means algorithm.

1. **Chooses the initial centroids**
   - $k$ initial "means" (in this case $k=3$) are randomly generated within the data domain (shown in color).

2. **Cluster assignment step**
   - $k$ clusters are created by associating every observation with the nearest mean.
   - In Cluster assignment step, the algorithm goes through each of the data points and depending on which cluster is closer, whether the red cluster centroid or the blue cluster centroid or the green; It assigns the data points to one of the three cluster centroids.

3. **Move centroid step**
   - The centroid of each of the $k$ clusters becomes the new mean.
   - In move centroid step, K-means moves the centroids to the average of the points in a cluster. In other words, the algorithm calculates the average of all the points in a cluster and moves the centroid to that average location.

4. **Steps 2 and 3 are repeated until convergence has been reached**
   - Steps 2 and 3 are repeated until the centroids do not move significantly.
K-Mean Clustering

• Weakness of K-means
  • The number of cluster “k” must be specified in advance.
  • Sensitive to initial centroids selection, which leads to unwanted solution.
  • k-means can only handle numerical data.
  • The algorithm may get stuck in the local optimum.
  • Sensitive to outliers and noise, which results in an inaccurate partition.
  • K-means cannot handle non-globular clusters or clusters of different sizes and densities.

Figure 1: K-means with clusters of different size
Figure 2: K-means with clusters of different density
Figure 3: K-means with non-globular clusters
Naïve Bayes

• Naïve Bayes is a simple but important probabilistic model
  • It based on applying Bayes’ theorem with the “naive” assumption of independence between the features.
  • It computes the conditional probability distribution of each feature given label, and then it applies Bayes’ theorem to compute the conditional probability distribution of label given an observation and use it for prediction.
  • It classifies new data based on the highest probability of its belonging to a particular class.
  • Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods.
Example: Can we play tennis today?

Here we have 4 attributes. What we need to do is to create “look-up tables” for each of these attributes, and write in the probability that a game of tennis will be played based on this attribute.

### Outlook

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Play = Yes</th>
<th>Play = No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>2/9</td>
<td>3/5</td>
<td>5/14</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>4/9</td>
<td>0/5</td>
<td>4/14</td>
</tr>
<tr>
<td>3</td>
<td>Overcast</td>
<td>3/9</td>
<td>2/5</td>
<td>5/14</td>
</tr>
</tbody>
</table>

### Temperature

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Play Tennis ?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

\[
P(A | B) = \frac{P(B | A) P(A)}{P(B)}
\]

### Temperature

<table>
<thead>
<tr>
<th>Temperature</th>
<th>Play = Yes</th>
<th>Play = No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hot</td>
<td>2/9</td>
<td>2/5</td>
<td>4/14</td>
</tr>
<tr>
<td>Mild</td>
<td>4/9</td>
<td>2/5</td>
<td>6/14</td>
</tr>
<tr>
<td>Cool</td>
<td>3/9</td>
<td>1/5</td>
<td>4/14</td>
</tr>
</tbody>
</table>

### Humidity

<table>
<thead>
<tr>
<th>Humidity</th>
<th>Play = Yes</th>
<th>Play = No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>3/9</td>
<td>4/5</td>
<td>7/14</td>
</tr>
<tr>
<td>Normal</td>
<td>6/9</td>
<td>1/5</td>
<td>7/14</td>
</tr>
</tbody>
</table>

### Wind

<table>
<thead>
<tr>
<th>Wind</th>
<th>Play = Yes</th>
<th>Play = No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong</td>
<td>3/9</td>
<td>3/5</td>
<td>6/14</td>
</tr>
<tr>
<td>Weak</td>
<td>6/9</td>
<td>2/5</td>
<td>8/14</td>
</tr>
</tbody>
</table>
Naïve Bayes

If \( X = (\text{Outlook} = \text{Sunny}, \text{Temperature} = \text{Cool}, \text{Humidity} = \text{High}, \text{Wind} = \text{Strong}) \), then

\[
P(\text{Play}=\text{Yes} | X) = \frac{P(\text{Outlook} = \text{Sunny}, \text{Temperature} = \text{Cool}, \text{Humidity} = \text{High}, \text{Wind} = \text{Strong} | \text{Play}=\text{Yes}) \cdot P(\text{Play}=\text{Yes})}{P(\text{Outlook} = \text{Sunny}, \text{Temperature} = \text{Cool}, \text{Humidity} = \text{High}, \text{Wind} = \text{Strong})}
\]

\[
= \frac{P(\text{Outlook} = \text{Sunny, Temperature} = \text{Cool, Humidity} = \text{High, Wind} = \text{Strong} | \text{Play}=\text{Yes}) \cdot P(\text{Temperature} = \text{Cool} | \text{Play}=\text{Yes}) \cdot P(\text{Humidity} = \text{High} | \text{Play}=\text{Yes}) \cdot P(\text{Wind} = \text{Strong} | \text{Play}=\text{Yes}) \cdot P(\text{Play}=\text{Yes})}{P(\text{Outlook} = \text{Sunny, Temperature} = \text{Cool, Humidity} = \text{High, Wind} = \text{Strong})}
\]

\[
= \frac{(2/9) \cdot (3/9) \cdot (3/9) \cdot (3/9) \cdot (9/14)}{(5/14) \cdot (4/14) \cdot (7/14) \cdot (6/14)}
\]

\[
= \frac{0.0053}{0.02186} = 0.2424
\]

\[
P(\text{Play}=\text{No} | X) = \frac{P(\text{Outlook} = \text{Sunny, Temperature} = \text{Cool, Humidity} = \text{High, Wind} = \text{Strong} | \text{Play}=\text{No}) \cdot P(\text{Play}=\text{No})}{P(\text{Outlook} = \text{Sunny, Temperature} = \text{Cool, Humidity} = \text{High, Wind} = \text{Strong})}
\]

\[
= \frac{P(\text{Outlook} = \text{Sunny, Temperature} = \text{Cool, Humidity} = \text{High, Wind} = \text{Strong} | \text{Play}=\text{No})}{P(\text{Outlook} = \text{Sunny, Temperature} = \text{Cool, Humidity} = \text{High, Wind} = \text{Strong})}
\]

\[
= \frac{0.0206}{0.02186} = 0.9421
\]

Since 0.9421 is greater than 0.2424 then the answer is ‘no’, we cannot play a game of tennis today.
Decision Tree

- Decision tree builds classification or regression models in the form of a tree structure.
  - predict the value of a target variable by following the decisions in the tree from the root (beginning) down to a leaf node.
  - A tree consists of branching conditions where the value of a predictor is compared to a trained weight.
  - Decision trees are prone to overfitting, additional modification, or pruning, may be used to simplify the model.
• Growing a tree involves deciding on which features to choose and what conditions to use for splitting, along with knowing when to stop.
• Generally entropy is a measure of disorder or uncertainty
  • Entropy is a concept used in physics, mathematics, computer science (information theory) and other fields of science. The concept of entropy originated in thermodynamics as a measure of molecular disorder: entropy approaches zero when molecules are still and well ordered.
Decision Tree

• Mathematical Definition of Entropy

\[ \text{Entropy} = - \sum_{i=1}^{n} p_i \log_2 p_i \]

Where \( p_i \) is the probability of getting the \( i^{th} \) value when randomly selecting one from the set.

In other words, where there are \( n \) classes, and \( p_i \) is the probability an object from the \( i^{th} \) class appearing.

Example:

- 16/30 are blue circles: \( \log_2 \left( \frac{16}{30} \right) = -0.9 \)
- 14/30 are orange crosses: \( \log_2 \left( \frac{14}{30} \right) = -1.1 \)

\[ \text{Entropy} = - \sum_{i=1}^{n} p_i \log_2 p_i = - \left( \frac{16}{30} \right) (-0.9) - \left( \frac{14}{30} \right) (-1.1) = 0.99 \]
Decision Tree

- Use entropy to measure the “purity” of the split

For example, consider the split on the outlook feature:

- **Sunny**: 2 Yes / 3 No
- **Overcast**: 4 Yes / 0 No
- **Rain**: 3 Yes / 2 No

For the pure set (4 Yes / 0 No):
- Entropy is 0, which means the sample is completely certain (100%).

For the impure set (3 Yes / 3 No):
- Entropy is 1, which means the sample is completely uncertain (50%).
Decision Tree with ID3 Algorithm

- Measures that can be used to capture the purity of split.
  - Information Gain

A reduction of entropy is often called an information gain. ID3 algorithm uses entropy to calculate the homogeneity of a sample.

\[
\text{Information Gain} = \text{Entropy}_{\text{before}} - \text{Entropy}_{\text{after}}
\]

Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches)

- A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous).
- The information gain is based on the decrease in entropy after a dataset is split on an attribute. 

[Diagram showing entropy calculations and decision tree branches]
Decision Tree with CART Algorithm

- Measures that can be used to capture the purity of split.
  - Gini impurity Index

- Equation for Gini impurity
  \[
  G_i = 1 - \sum_{k=1}^{n} (p_{ik})^2
  \]

  - \( p_{ik} \) is the ratio of class \( k \) instances among the training instances in the \( i \)th node.

- A node’s Gini attribute measures its impurity: a node is “pure” (gini=0) if all training instances it applies to belong to the same class. In other words, Gini Index would be zero if perfectly classified.

Gini impurity = 1 - (the probability of “yes”)² - (the probability of “no”)²

\[
= 1 - \left( \frac{105}{105 + 39} \right)^2 - \left( \frac{39}{105 + 39} \right)^2
\]
Decision Tree with CART Algorithm
Decision Tree

• Pros and Cons of Decision Tree
  • Pros
    • Simple to understand and to interpret.
    • To build decision tree requires little data preparation.
    • Handle both continuous and categorical variables.
    • Implicitly perform feature selection.
  • Cons
    • They are prone to over-fitting.
    • create biased trees in case of unbalanced data.
    • Instability.
    • Greedy approach used by Decision tree doesn’t guarantee best solution.
    • Standard decision trees are restricted by hard, axis-aligned splits of the input space.
Random Forest

• Random Forest is one of the most used algorithms.
• Random Forest = Bagging + Full-grown CART decision Tree
Random Forest

• Classification example: use Random Forest to classify data
  • After training, a tree set \( \{T\} \) can be obtained to predict the classes of the unseen samples by taking the majority vote from all individual classification trees.
Random Forest

• Pros and Cons of Random Forest
  • Pros
    • Random Forest algorithms can be grown in parallel.
    • Random Forest has higher classification accuracy.
    • Able to deal with the missing value and maintain accuracy in case of missing data.
    • Help data scientists save data preparation time.
  • Cons
    • Large number of decision trees in the random forest can slow down the algorithm.
    • Good job at classification but not as good as for regression.
    • like a black box approach, random forest is not easily interpretable.
• A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane.

- $H_1$ does not separate the classes.
- $H_2$ does, but only with a small margin.
- $H_3$ separates them with the maximum margin.

- Examples closet to the hyper-plane are **support vectors**
- Margin $\rho$ of the separator is the distance between support vectors.
Support Vector Machine

**Step 1:** Start with a random line of equation $ax + by + c = 0$.
Draw parallel lines with equations:
- $ax + by + c = 1$, and
- $ax + by + c = -1$

**Step 2:** Pick a large number. **1000** (number of repetitions, or epochs)

**Step 3:** Pick a learning rate. **0.01**

**Step 4:** Pick an expanding rate. **0.99**

**Step 5:** (repeat **1000** times)
- Pick random point $(p,q)$
  - If point is correctly classified
    - Do nothing
  - If point is blue, and $ap+bq+c > 0$
    - Subtract $0.01p$ to $a$
    - Subtract $0.01q$ to $b$
    - Subtract $0.01$ to $c$
  - If point is red and $ap+bq+c < 0$
    - Add $0.01p$ to $a$
    - Add $0.01q$ to $b$
    - Add $0.01$ to $c$
- Multiply $a$, $b$, $c$, by **0.99**
Support Vector Machine

• SVMs sometimes use a kernel transform to transform non-linearly separable data into higher dimensions where a linear decision boundary can be found, the kernel trick.

- apply transformation
- $z = x^2 + y^2$
- clear separation is visible
- transform back to original plane
Support Vector Machine

- SVM using a Non-Linear Kernel

Where do you build your fence?

Well if you’re a really data driven farmer one way you could do it would be to build a classifier based on the position of the cows and wolves in your pasture. Trying a few different types of classifiers, we see that SVM does a great job at separating your cows from the packs of wolves. I thought these plots also do a nice job of illustrating the benefits of using a non-linear classifiers.

You can see the the logistic and decision tree models both only make use of straight lines. ^1
Support Vector Machine

• Pros and Cons of Support Vector Machine
  • Pros
    • The training is relatively easy.
    • No local optimal, unlike in neural networks.
    • SVMs have a regularization parameter, which can help avoid over-fitting.
    • Effective in high dimensional spaces.
  • Cons
    • For classification, the SVM is only directly applicable for two-class tasks.
    • SVMs do not directly provide probability estimates.
    • Parameters of a solved model are difficult to interpret.
    • Long training time on large data sets.
    • Choosing a “good” kernel function can be tricky.
Hands-on time, but first let’s get familiar with H2O Flow.
(Recommended Web Browser : Firefox)
(ReservationName=uiuc_21)
H2O Flow

Assistance
- Description
- Load files into H2O
- Import SQL table into H2O
- Get a list of frames in H2O
- Split a frame into two or more frames
- Merge two frames into one
- Get a list of models in H2O
- Get a list of grid search results in H2O
- Get a list of predictions in H2O
- Get a list of jobs running in H2O
- Automatically train and tune many models
- Build a model
- Import a saved model
- Make a prediction

Using Flow for the first time?
- Quickstart Videos

Examples:
- Flow Web UI
- Importing Data
- Building Models
- Making Predictions
- Using Flows
- Troubleshooting Flow

Flow packs are a great way to explore and learn H2O. Try out these Flows and run them in your browser.
H2O Flow

- Admin
- Jobs / Cluster Status / Water Meter
H2O Flow

• Help
  • View example Flows
    • GBM_Example.flow
    • DeepLearning_MNIST.flow
    • GLM_Example.flow
    • DRF_Example.flow
    • K-Means_Example.flow
    • Million_Songs.flow
    • KDDCup2009_Churn.flow
    • QuickStartVideos.flow
    • Airlines_Delay.flow
    • GBM_Airlines_Classification.flow
    • GBM_GridSearch.flow
    • RandomData_Benchmark_Small.flow
    • GBM_TuningGuide.flow
    • XGBoost_Example.flow
H2O Flow

- Import Data
- Parse Data
- Split Data
- Build Model
- Predict
- Save Model
K-Mean Clustering with H2O

• Seeds Data Set
• Measurements of geometrical properties of kernels belonging to three different varieties of wheat.
  • area $A$
  • perimeter $P$
  • compactness $C = 4\pi A/P^2$
  • length of kernel
  • width of kernel
  • asymmetry coefficient
  • length of kernel groove
K-Mean Clustering with H2O

- Import Data:
  - importFiles [ "http://s3.amazonaws.com/h2o-public-test-data/smalldata/flow_examples/seeds_dataset.txt" ]

- Parse Data:
  - ["separator:9", "number_columns:8"]

- Build Model:
  - ["K-Means", "K:3", "Max_iterations:100"]
• Internet Advertisement Data Set
  • This dataset represents a set of possible advertisements on Internet pages.
  • The features encode the geometry of the image (if available) as well as phrases occurring in the URL, the image's URL and alt text, the anchor text, and words occurring near the anchor text.
  • The task is to predict whether an image is an advertisement ("ad") or not ("nonad").
Distributed Random Forest on H2O

• Import Data:
  • importFiles [ "https://s3.amazonaws.com/h2o-public-test-data/smalldata/flow_examples/ad.data.gz" ]

• Parse Data:
  • [destination_frame: "ad.hex", parse_type: "CSV", separator: 44, number_columns: 1559, single_quotes: false]

• Build Model:
  • buildModel 'drf', {"training_frame":"ad.hex", "response_column":"C1559", "ntrees":"10", "max_depth":20, "min_rows":10, "nbins":20, "mtries":"1000", "sample_rate":0.6666667, "build_tree_one_node":false, "balance_classes":false, "class_sampling_factors":[], "max_after_balance_size":5, "seed":0}
AutoML

• The term “AutoML” (Automatic Machine Learning) refers to automated methods for model selection and/or hyperparameter optimization.
  • To enable non-experts to train high quality machine learning models.
  • To improve the efficiency of finding optimal solutions to machine learning problems.
  • explore a variety of algorithms such as Gradient Boosting Machines (GBMs), Random Forests, GLMs, and Deep Neural Networks.
• No Free Lunch Here, AutoML is slow due to heavy workload.
AutoML

- parseFiles
  - [parse_type: "CSV", separator: 44, number_columns: 5]
- splitFrame
  - ["iris_data.hex", [0.75], ["frame_0.750","frame_0.250"], 174460]
- runAutoML
  - max_runtime_secs: 300
Thank You for Your Time!