

MSBA 5510
Dominican University of California

Week 3: Machine Learning
Modeling Using H2O.ai

Daqing Zhao
Fall 2022

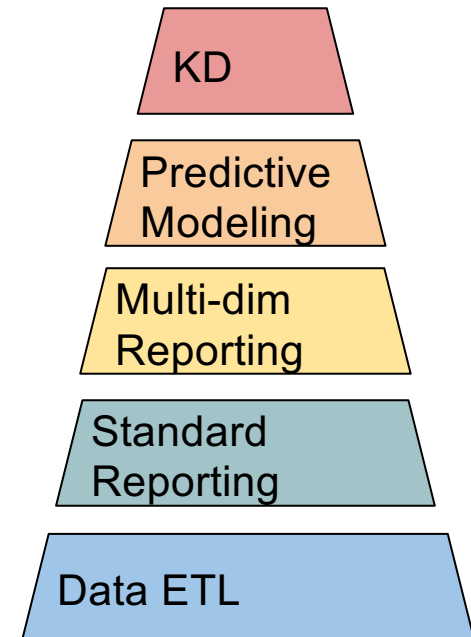
© Daqing Zhao

Learning Objectives

- Be able to build machine learning models using some opensource platform, and handle large data sets
- Knowing technical process of model building
- Knowing how to train models
- Knowing how to assess models
- Knowing how to optimize models
- Be able to explain how models make prediction
- Be able to optimize and select good models
- Be able to explain modeling concepts
- Be able to identify common problems in models

Business Intelligence

- BI Maturity process
- Data ETL (Extract/Transform/Load), Standard Reporting, Multi-dimensional Reporting, Segmentation/Predictive Modeling, Knowledge Discovery
- As businesses become more mature, more advanced services are developed and used
- BI maturity process roughly parallels the process of model building



Trend of ML Models

- Data sets are larger, more features and more rows
- Model optimizations take more computational resources
- Inclusion of unformatted data
- More model parameters, ensemble models
- Development of model algorithms and modeling methodologies, and software modeling platforms, data processing platforms
- Reduction of cost of data collection and storage, and computational resources
- Moore's Law and development of GPU and special purpose processors

Computer Servers

- CPU
- RAM
- Disk
- OS
- Applications



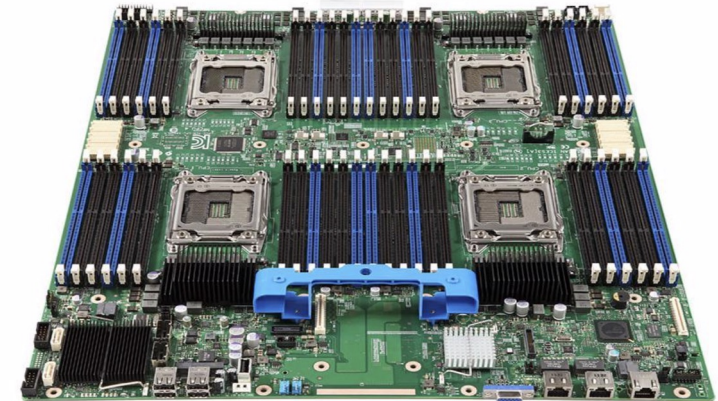
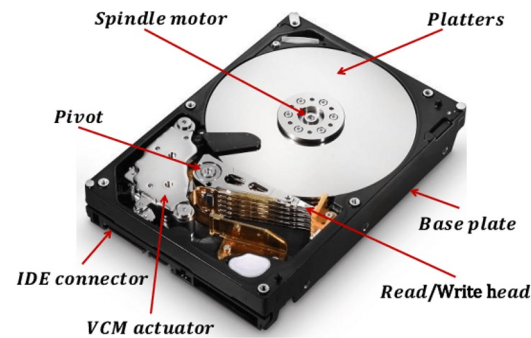
10-
20GB/sec



500MB/sec

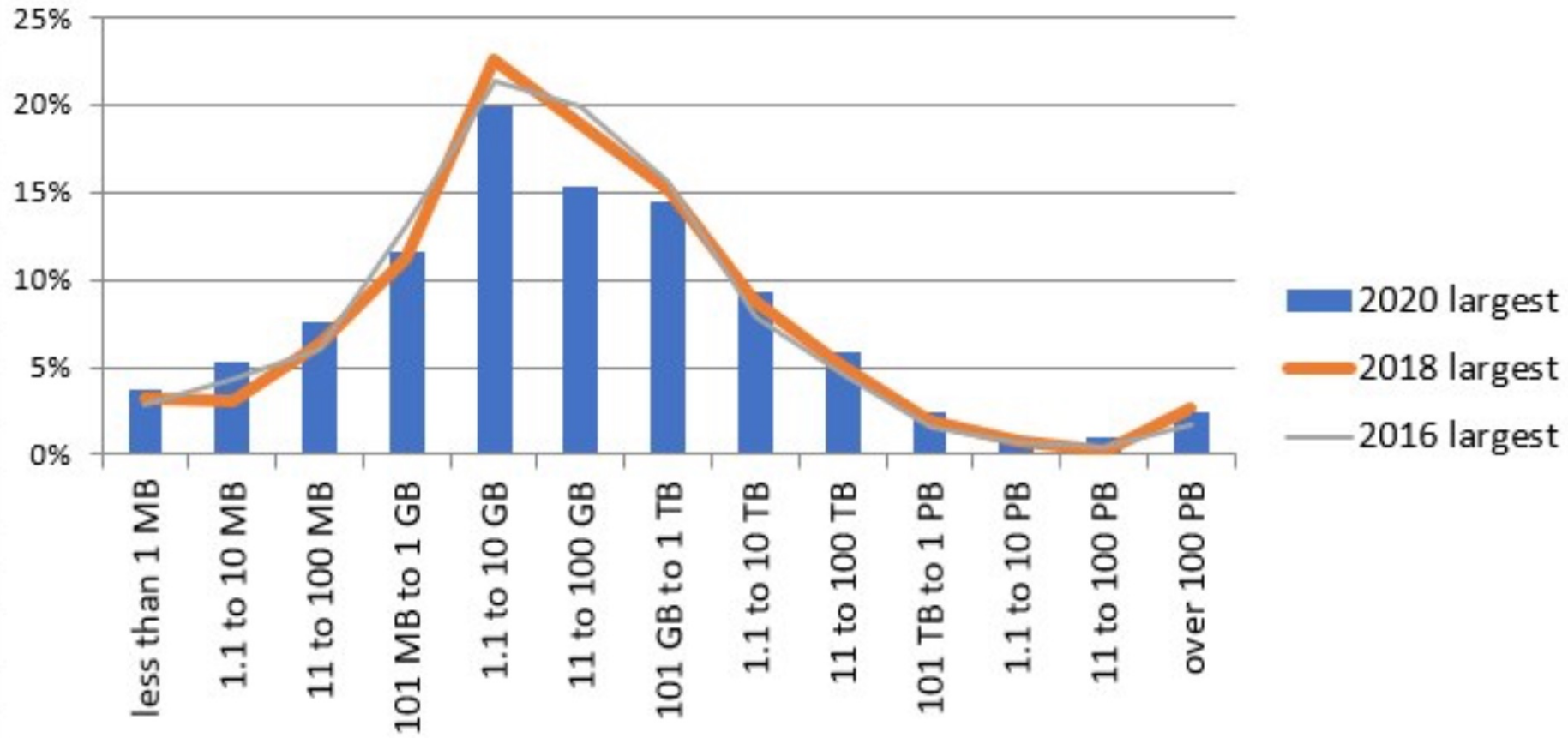


100MB/sec



It takes 17 minutes to read 100GB of data from a hard drive

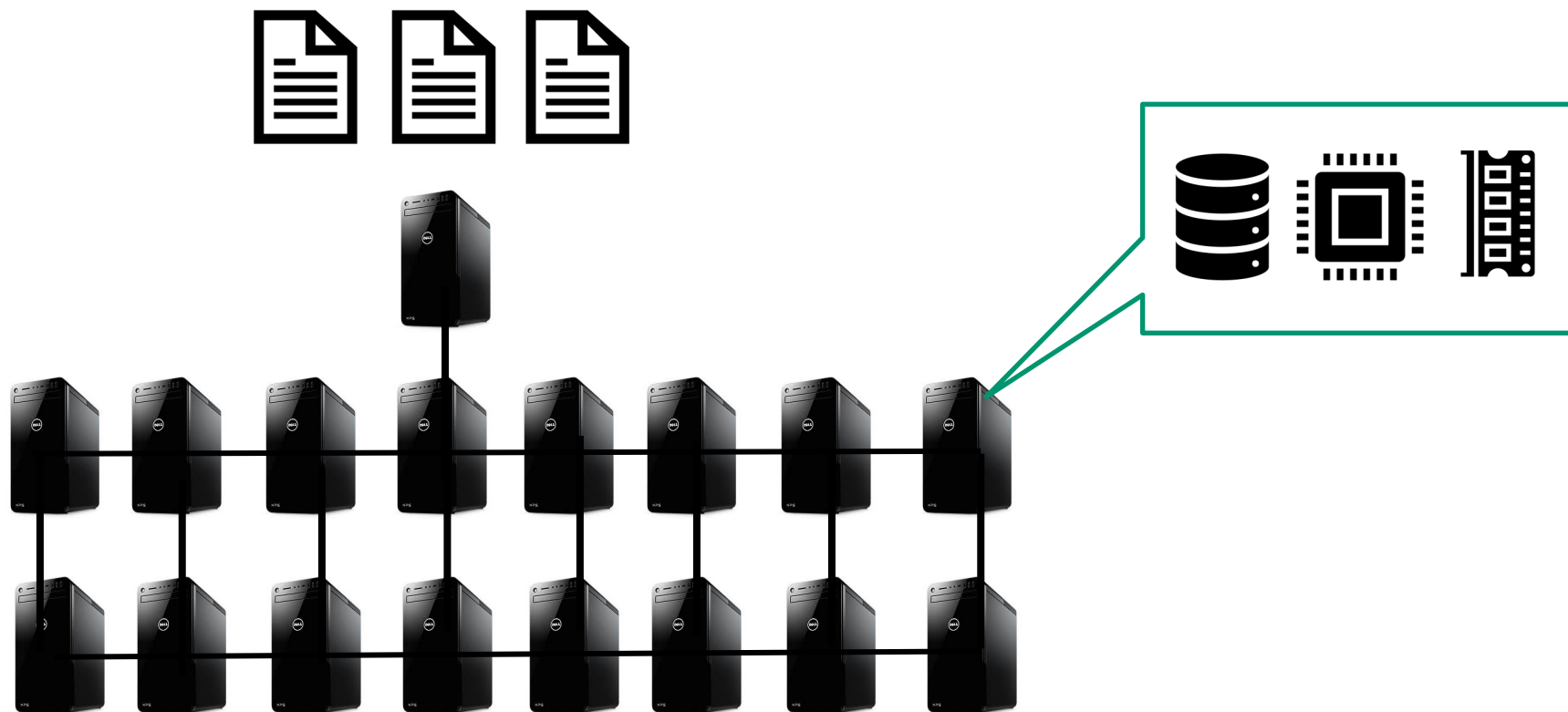
KDnuggets 2020 Poll: Largest Dataset Analyzed



Hadoop Cluster

Distributed
Storage and
Computing

Divide and Conquer
Split Data and Put in Many Servers



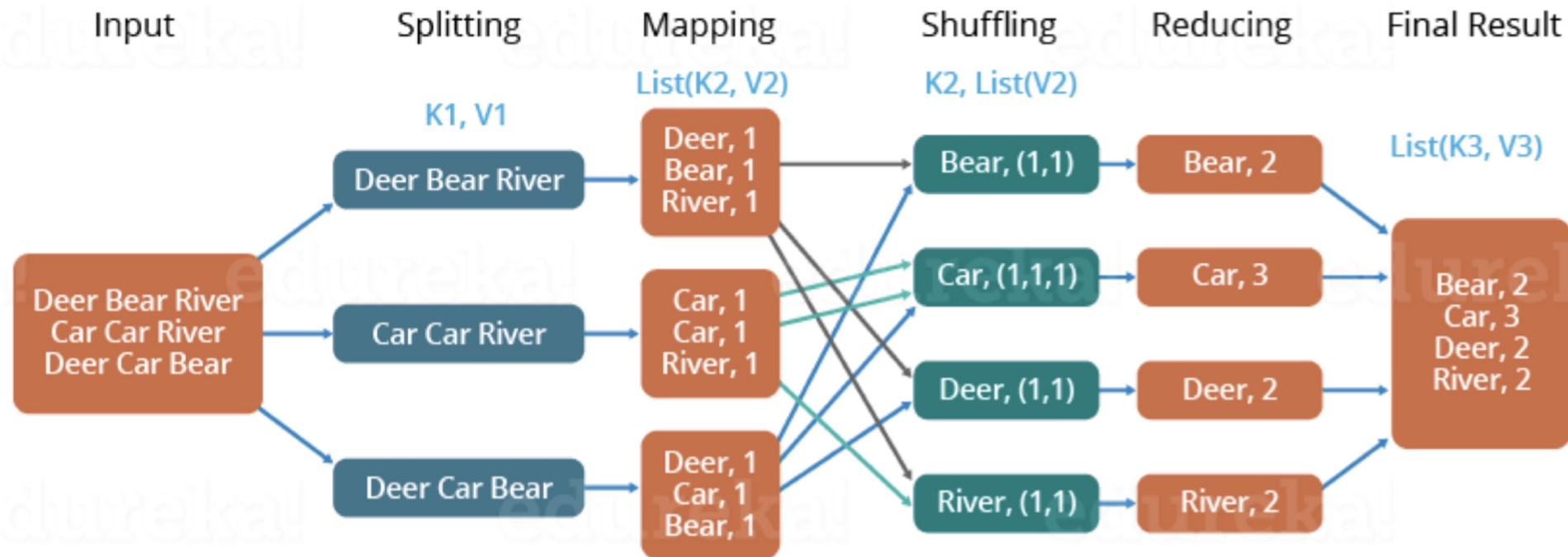
MapReduce

- MapReduce is the framework allowing a large cluster of servers to work as one computer for data processing
- Map is a set of jobs that runs locally on each server, loading data from disk and process to give some intermediate data
- Shuffle is to move intermediate data that need to be aggregated to the appropriate servers
- Reduce is a set of jobs that aggregates the intermediate data from Map jobs
- Map Reduce can be repeatedly used for complex tasks
- Map Reduce is the only guaranteed way to scale
- Writing Map Reduce by hand is very tedious, Hive translates SQL into Map Reduce jobs that can be executed

MapReduce Example

The Overall MapReduce Word Count Process

edureka!

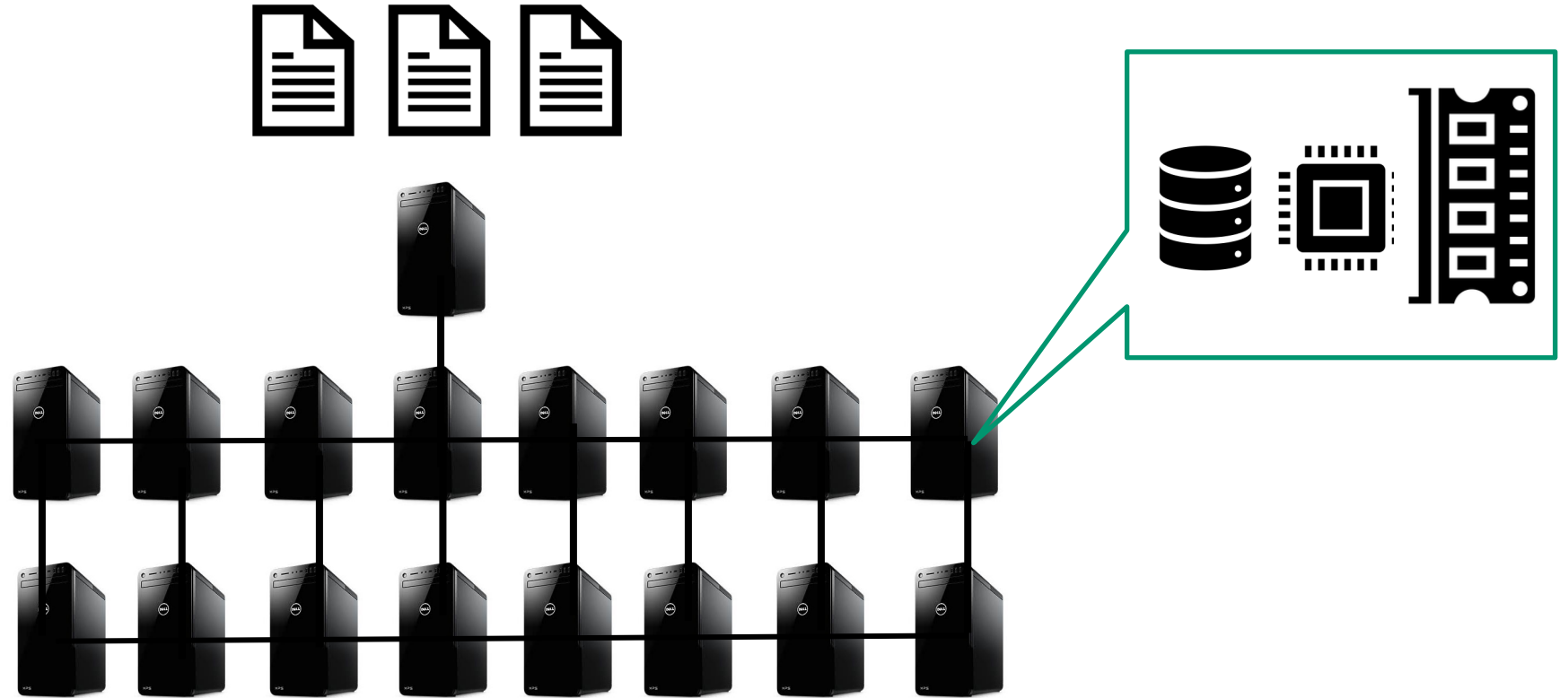


MapReduce Jobs

- Word count on a corpus of text documents on 26 servers, each document is stored on one server
- First step is that Map on each server count words on its own server
- Then shuffle partial results from all servers based on word's first letter, so one for each server,
 - ◆ “a” to server 1,
 - ◆ “b” to server 2,
 - ◆ ...
- Reduce on each server aggregates the partial results

Spark or H2O Cluster

Distributed
In-Memory
Storage
and Computing
By In-memory
MapReduce



Data Storage Trend

- Object Storage in Data Lake
 - Decouple Storage and Compute
 - Easier to scale data
 - Structured and Unstructured data, any data
- Block Storage in Warehouse
 - Structured data, Curated, Validated
 - Allow Compute, have interface of SQL
 - In cloud or on premise

Lakehouse Pattern

Lakehouse combines data lake and data warehouse

Lakehouse allows data science

Data exploration

Data curation

Streaming

ETL or ELT into structured data layer

While data warehouse supports BI and applications

Structured data

Databrick Lakehouse

Lakehouse on AWS, MS Azure and Google Cloud

Delta Lake, open source, providing structure and governance

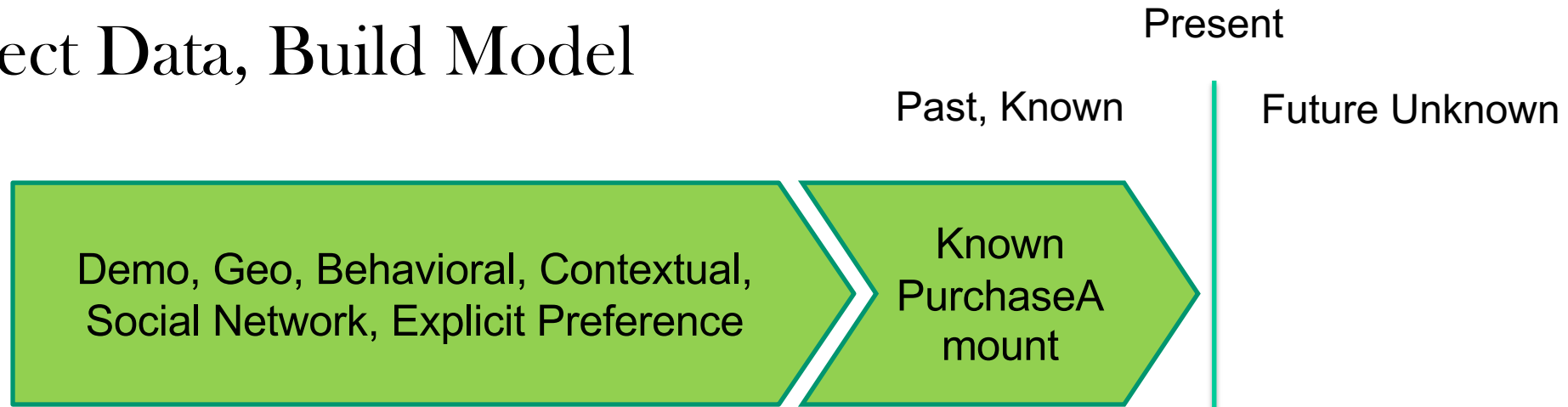
Bronze table, raw data ingested, as is

Silver table, refined view, query-able, clean, normalized, single version of truth

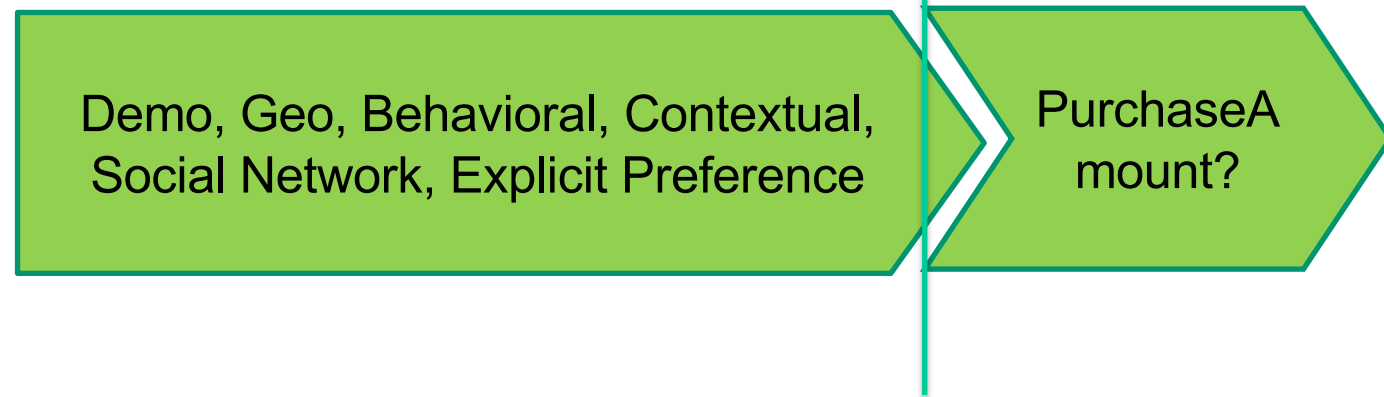
Gold table, data mart, aggregates, AI & Reporting, Streaming Analytics

Predictive Models

Collect Data, Build Model



Predict future cases



Targets and Predictors

- Unsupervised Learning
- Supervised Learning
- Binary Target
- Numeric Target
- Multiclass Classification
- One versus all
- One versus one

Data Formats

- Flat files, target and a tuple of predictors
 - A mixed list of numerical, categorical, ordinal vars, and a target
 - Format usually are in:
 - Text files
 - Database tables
 - R data frames
 - H2O data frames
 - JSON
 - on HDFS
 - on Spark RDD
- Pros and cons considerations:
 - Flexibility
 - Enforce data type
 - Compressed
 - Fast to access and write back
 - Easy to manipulate
 - Easy for ML algorithms to access

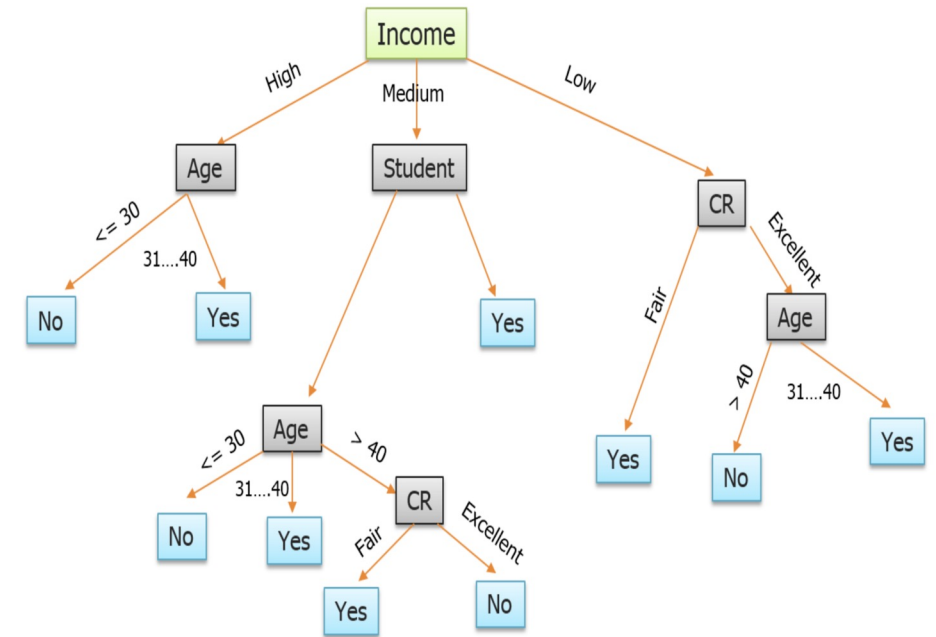
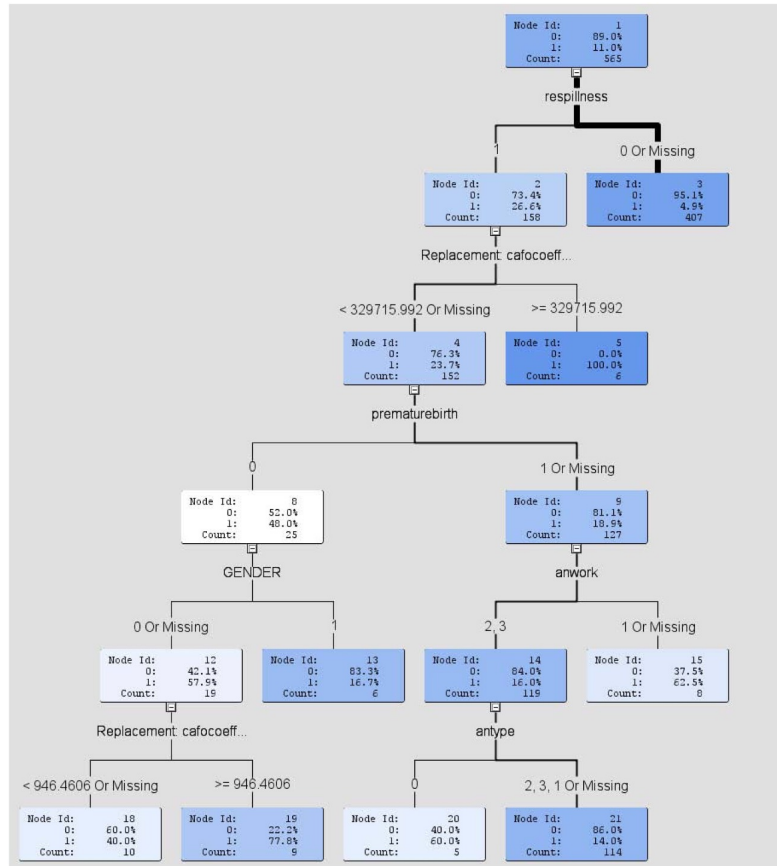
Modeling Technical Process

- Data Collection
- Data Cleaning
- Feature Engineering or Data Transformation
- Target Definition
- Model Optimization
- Model Validation
- Model Deployment
- Model Monitoring
- Feedback Data Collection
- Iterative Refinement

Decision Trees

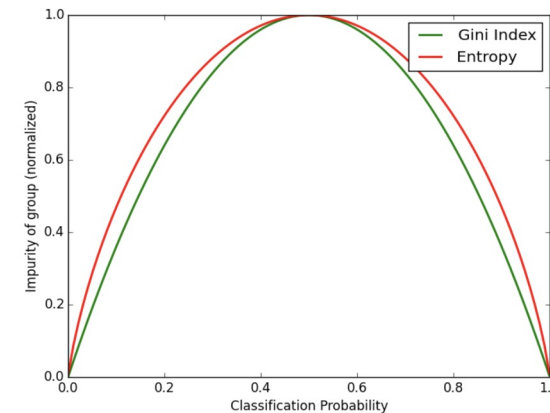
- **CART, CHAID, C4.5/C5**
- **CART, Classification and Regression Trees** uses **GINI**, binary trees
 - ♦ Regression tree using **Least Squared Deviation** or **Least Absolute Deviation**
- **CHAID, Chi-squared Automatic Interaction Detector** uses **Chi-squared** tests
 - ♦ Non-binary splits (into multiple branches)
 - ♦ Regression using **F-tests**
- **C4.5/C5** entropy difference to split into multiple branches
- Grow trees and prune trees

CART and CHAID

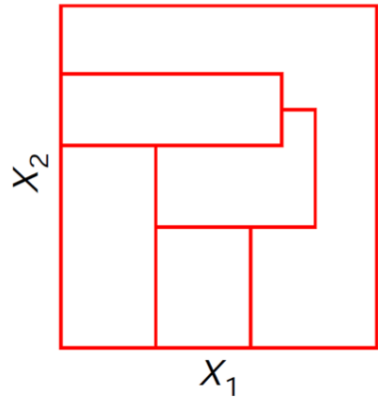


Entropy: $-\sum P(i) \log P(i)$

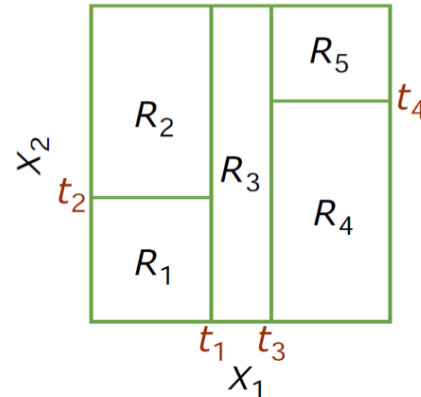
Gini: $1 - \sum P(i)^2$



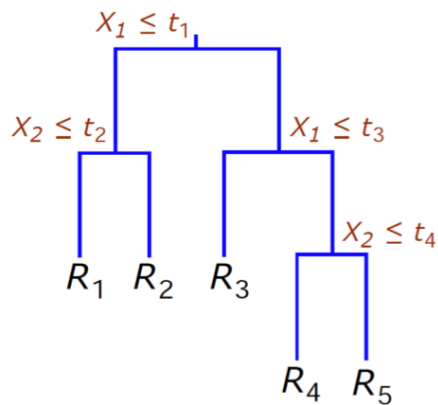
Partitions and CART



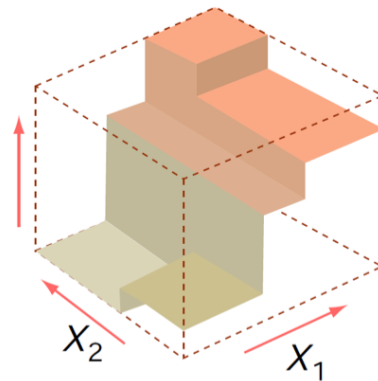
(a) General partition that cannot be obtained from recursive binary splitting.



(b) Partition of a two-dimensional feature space by recursive binary splitting, as used in CART, applied to some fake data.



(c) Tree corresponding to the partition in the top right panel.



(d) A perspective plot of the prediction surface.

Decision Trees

- Recursively split data to improve purity
- Find the best predictor and best place to cut
- To get child nodes of higher purity (entropy)
- Successively pick the best variable to split the current node (recursive)
- One variable can be used many times
- It does not assume any distribution
- Reproducibility is ensured by using validation and test data sets.

Decision Trees

- Splits by nature are nonlinear
- Splits to reduce entropy or Gini for purity
- For regressions to reduce least squares deviations
- Tree building processes are greedy
- Simple trees may not be accurate
- Tree building calculations are efficient
- For each split, we scan and search over all predictors
- At most 2^d splits, at maximum depth d

Decision Trees

- Simple trees are easy to interpret
- No need to do data transformations that are monotonic functions
- No need to assume any distribution
- Missing data do not break the prediction, may make prediction less accurate
- Tree structure unstable for sample variations
- Bad for linear patterns

DT Hyper Parameters

- Max tree depth
- Split criteria, Gini, entropy, twoing
 - Similar shapes
- Number of child nodes, binary is sufficient
- Min support of leaf nodes
- Cross validation
- Validation data file

Problems of Modeling

- Not Enough data, data quality issues
- Inaccuracy, not converging
- Overfitting
- Collinearity
- Data sparsity
- High dimensionality
- Model degrade quickly
- Leakage
- Heteroscedasticity

Leakage examples

- Leakage occurs when in training data information of predictors is present in target variables
- Include market index to predict stock price
- Include data fields available only for customers with behavior to be predicted
- Display ads appear on certain pages, then one can predict that clicks happen on pages of ads
- Customer intent prediction using data of part of the purchasing process
- Data transformation using information of entire data such as normalization or PCA and then use cross validation

Titanic Data Set

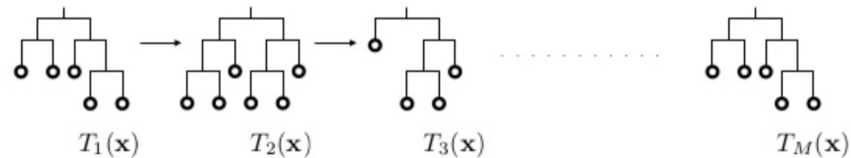
VARIABLE DESCRIPTIONS

Pclass	Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
survival	Survival (0 = No; 1 = Yes)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare (British pound)
cabin	Cabin
embarked	Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)
boat	Lifeboat
body	Body Identification Number
home.dest	Home/Destination

More Decision Trees

- Bagging and Ensemble models
 - Take bootstrap samples and build a set of trees, models
- Random Forests
 - Select a sample, and a subset of predictors
- Gradient Boosting Machines
 - Build a sequence of additive shallow trees
- Rulefit
 - Use a set of rules from a set of trees as predictors to fit a linear model

Gradient Boosting Machine (GBM)



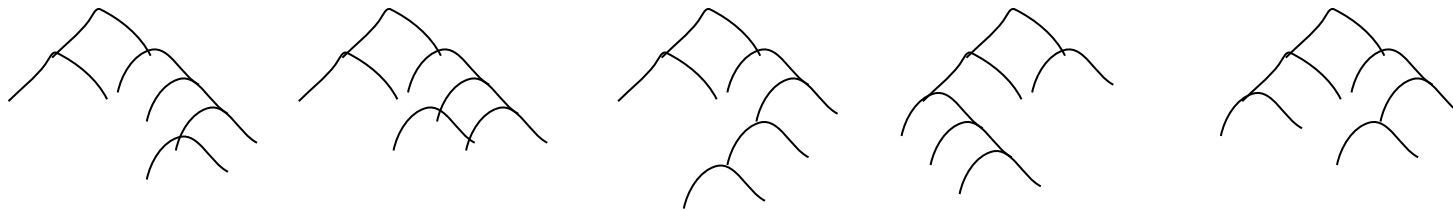
$$f_i(\mathbf{x}) = f_{i-1}(\mathbf{x}) + T_i(\mathbf{x}; \hat{\theta}_i)$$

GBM

- Gradient Boosting Machines
- Build a series of shallow CART trees added together
- Each tree stays once it is built
- Next tree will be built more on where the previous trees don't do well
- Each tree focuses somewhat different parts of the data set
- Weight at each step calculated to minimize loss function
- Each tree is multiplied by a learn rate
- Smaller learn rate limit each tree's contribution to allow slow learning
- Each tree is shallow, to reduce greedy learning

Random Forest

- Random Forest builds an ensemble of independent CART trees
- Each tree is built on a bootstrap sample from the training data
- Each tree only use a fraction of the p predictors, usually \sqrt{p}
- Results are averaged or obtained by voting, it is based on a committee of independent predictions
- Usually the number of trees are very large, 100s to 1000s
- Random Forests make estimation based on a principle of Crowdsourcing
- Random Forests are robust, stable and widely used



GBM Hyper Parameters

- Number of trees
- Learn rate
- Max tree depths
- Min support for leaf nodes
- Binomial (Bernoulli) or Gaussian loss function
- Cross validation
- Validation file
- Huber M outlier threshold
- Platt Scaling normalization of distribution

XGBoost

- Another GBM implementation
- Having more regularization to control overfitting
- Weighted Quantile Sketch for rank approximation
- Sparsity aware split for speed
- Utilization of GPU, much faster than CPU
- Work together with max tree size

RF Hyper Parameter

- Number of trees
- Max tree depths
- Min support for leaf nodes
- Binomial (Bernoulli) or Gaussian loss function
- Sampling rate
- Predictor sampling rate
- Cross validation
- Validation file

Parameters for LR

- L1 and L2 regularization, elastic net
- Indicator variables for categorical variables
- Putting continuous variables into some buckets
- Automatically finding interactions
- Neural network architecture
- Layers and number of neurons
- $Y = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + \dots$
- $\alpha * L1 + (1-\alpha) * L2$

L1 Regularization

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M |W_j|$$

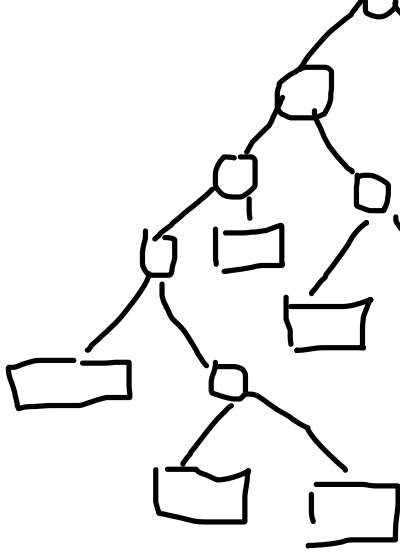
L2 Regularization

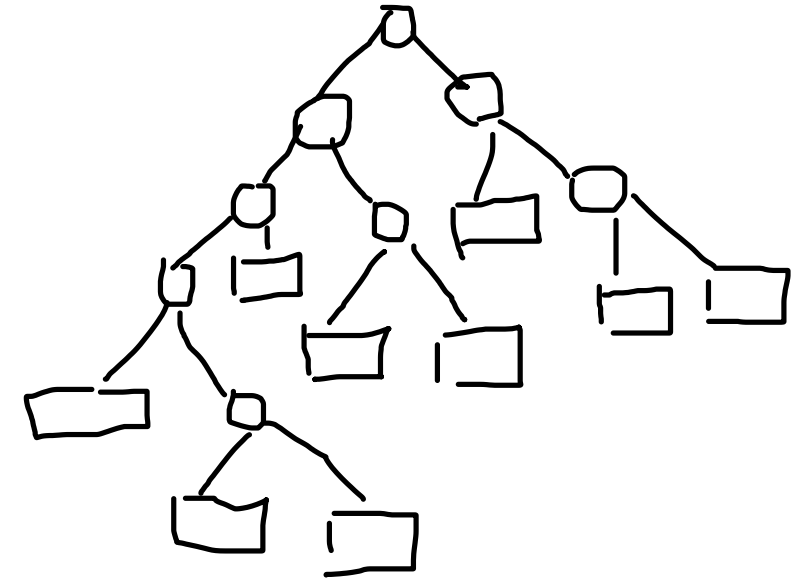
$$\text{Cost} = \underbrace{\sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2}_{\text{Loss function}} + \lambda \underbrace{\sum_{j=0}^M W_j^2}_{\text{Regularization Term}}$$

L2, L1 Regularizations

- L2 regularization adds to loss function a fraction of the sum of square of all parameters
- This tend to reduce the size of the parameters, and penalize unnecessarily complex models
- L1 regularization adds the sum of absolute values of coefficients
- A combination of L1 and L2 forms Elastic Net which works well
- Lambda strength, alpha weight of L1 vs L2

RuleFit

- Like GBM
 - Build a Sequence of Trees
 - But Add Nodes as Predictors
 - Both Interior and Leaf Nodes
 - Use as Input to Logistic Regression
 - Use L1 and L2 to Remove Unimportant Variables
- 



Biased Sampling

- In a typical marketing campaign, conversion rate is small
- How do you predict who is going to convert?
- For example, 1% customers converts, 10K out of 1 Million
- Typical model has accuracy of 80%, if we identify everyone as none converters, the error will be only 1%
- We want equal error rates for both converters and none converters
- For each percentage point of errors, 100 converters, 9.9K non converters
- We over sample converters so that model can predict converters better

H2O Modeling Platform

- H2O in-memory map reduce and custom compressions
- H2O Demo: 100 million rows and 50 columns, logistic regression on a 16 node cluster, finishes in 11 seconds
- A cluster of 20 machines
- Customer sample of 7 million, 4000+ columns
- Train 100 models, R interface, in two days
- Model accuracy by ROC AUC distributions, robustness
- Score all customers in about 1.5 hours
- Demo of H2O

H2O.ai's Architecture

- In Memory MapReduce
- Custom compression, columnar
- Use all CPUs and cores on a server and multiple servers in a cluster
- Some algorithms use GPU, if available
- Open source H2O 3
- Commercial product Driverless AI

AutoML in H2O

- The Automatic Machine Learning (AutoML)
- Automates the supervised machine learning model training process.
- The current version of AutoML trains and cross-validates
 - a Random Forest (DRF),
 - an Extremely-Randomized Forest (DRF/XRT),
 - a random grid of Generalized Linear Models (GLM)
 - a random grid of XGBoost (XGBoost),
 - a random grid of Gradient Boosting Machines (GBM),
 - a random grid of Deep Neural Nets (DeepLearning), and
 - 2 Stacked Ensembles
- one of all the models, and one of only the best models of each kind.

Data Sets

- Kaggle Competition Data Set
- <https://www.kaggle.com/ashydv/lead-scoring-logistic-regression>
- Freddie Mac Loan Delinquency Classification

Java Command Test

Open a Terminal on Mac (Lauchpad Other)
Open cmd on Windows (Windows Search)

```
[(base) daqingzhao@pa-dhcp-10-120-224-200 ~ % java
[Usage: java [-options] class [args...]
           (to execute a class)
 or  java [-options] -jar jarfile [args...]
           (to execute a jar file)
where options include:
    -d32          use a 32-bit data model if available
    -d64          use a 64-bit data model if available
    -server       to select the "server" VM
                  The default VM is server,
                  because you are running on a server-class machine.

    -cp <class search path of directories and zip/jar files>
    -classpath <class search path of directories and zip/jar files>
                  A : separated list of directories, JAR archives,
                  and ZIP archives to search for class files.
    -D<name>=<value>
                  set a system property
    -verbose:[class|gc|jni]
                  enable verbose output
```

Install Java

At Least One Per Study Group Installation of Java

Mac

https://www.java.com/en/download/help/mac_install.html

Windows

<https://www.oracle.com/java/technologies/downloads/#jdk19-windows>

Install Java

Use the Select Version of Java

Mac OS

```
/usr/libexec/java_home -V  
export JAVA_HOME=`/usr/libexec/java_home -v 17`  
echo $JAVA_HOME  
cd $JAVA_HOME  
java -version
```

H2O.ai Installation

- unzip downloaded
- `java -jar h2o.jar`
- `install.packages("/path/h2o-version/R/h2o_version.tar.gz",
repos = NULL, type = "source")`
- `demo(h2o.gbm)`

H2O.ai

H2O.ai has Free AWS Cloud Computing Sessions

<http://aquarium.h2o.ai>

Get Free Account

H2O-3 cluster

Driverless AI cluster

H2O.ai

H2O-3 cluster, Open Jupyter Notebook First, then R or Flow

Import file:

https://s3.amazonaws.com/data.h2o.ai/DAI-Tutorials/loan_level_500k.csv

ignore: "DELINQUENT", "PREPAID",
"PREPAYMENT_PENALTY_MORTGAGE_FLAG",
"PRODUCT_TYPE"

Try Models:

Gradient Boosting Machine; Distributed Random Forest; Generalized
Linear Models; RuleFit; XGBoost; Deep Learning

Optional Exercises

- Install h2o.ai opensource platform h2o 3
 - <https://h2o-release.s3.amazonaws.com/h2o/rel-zygmund/2/index.html>
- This requires installation of java 8
- Mac OS X
 - https://java.com/en/download/help/mac_install.xml
- Windows 10 64-bit
 - https://java.com/en/download/faq/java_win64bit.xml

Exercises: H2O.ai

On Loan Data or Leads Data or Own Data

Split Data Set into Two Data Set and Cross Validation

Change Model Hyper Parameters

Try on Same Set of Models:

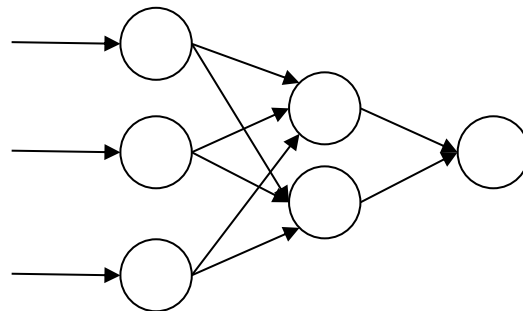
Gradient Boosting Machine; Distributed Random Forest; Generalized Linear Models; RuleFit; XGBoost; Deep Learning

Find Best Model in ROC AUC on Validation Data Set

Appendix

Improving LR

- Neural networks are a network of logistic regressions, each stage is fed to the next stage
- Each stage can be a layer of many logistic regressions
- Each logistic regression is a neuron
- A sequence of arrays of neurons, one layer feeding to another



Grid Search

Cartesian grid search

Random search

Bayesian search (Python Hyperopt)

Biased Sampling

- Customers form distributions in dimensions of age, income, house value
- If we sample from entire population, we make better predictions for typical customers
- Most valuable customers may not be typical, they are under represented in the sample
- We may build separate models for various segments based on demographics or lifetime value