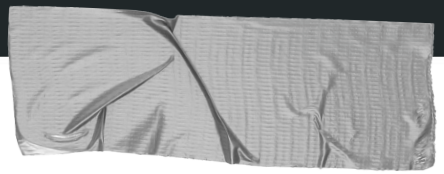
Fast and Scalable Machine Learning with GoLang

Vidyasagar N
@dumbyoda



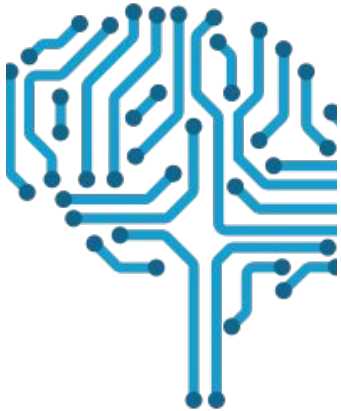
What

Discussion of Machine Learning, Go Libraries, Project Examples

- ➔ **Machine Learning**
  The basics of machine learning!
- ➔ **Golang in the architecture of machine learning systems**
  Our experience on using go along with machine learning systems
- ➔ **Go Libraries**
  Various go libraries solving specific puposes

Machine Learning

Machine learning is programming computers to optimize a performance criterion using example data or past experience.



Process

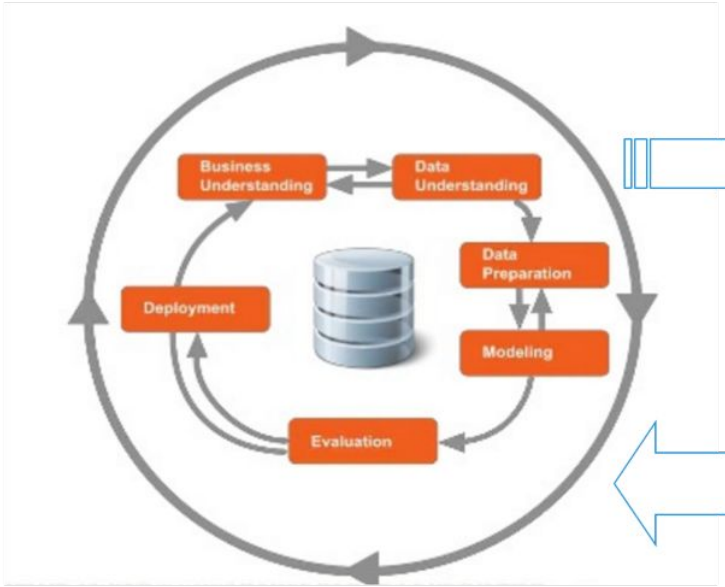
Data Reduction

Data Transformation

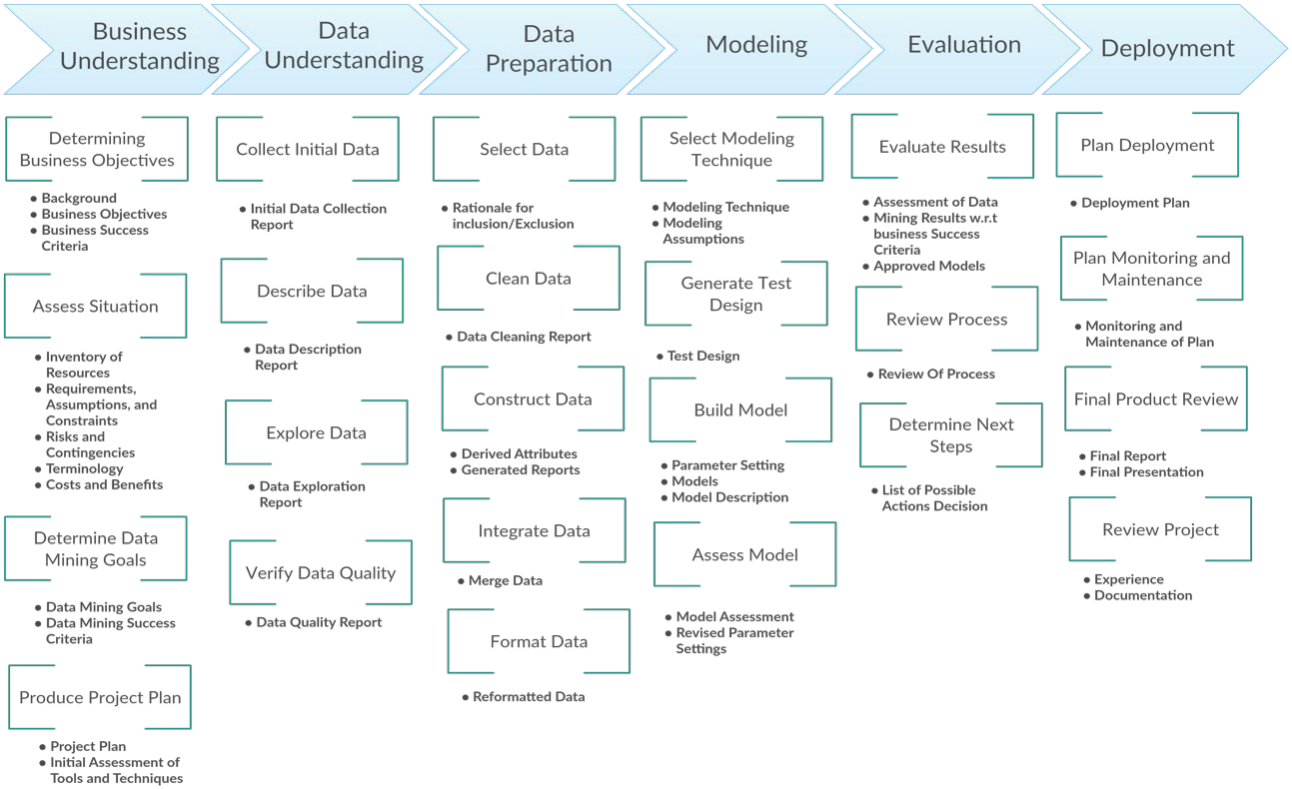
Data Cleaning

Data Consolidation

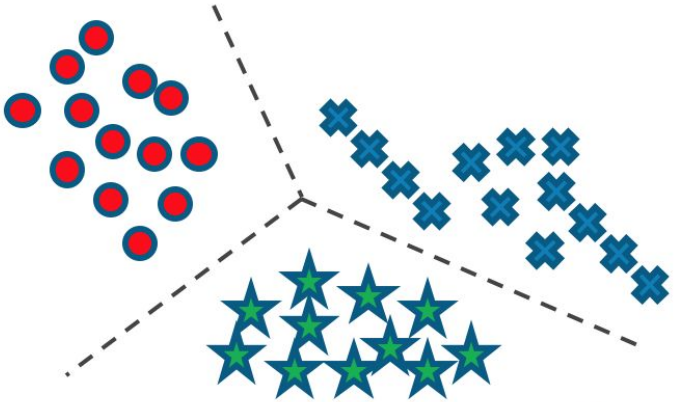
Modelling



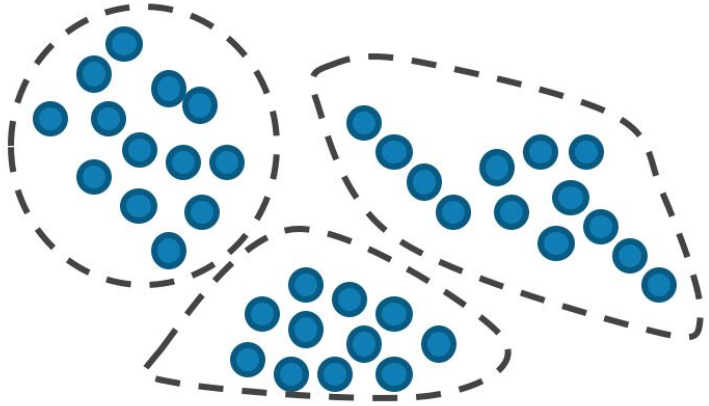
Life Cycle


Business Understanding
Data Understanding
Data Preparation
Modeling
Evaluation
Deployment
Determining Business Objectives
• Background
• Business Objectives
• Business Success Criteria
Assess Situation
• Inventory of Resources
• Requirements, Assumptions, and Constraints
• Risks and Contingencies
• Terminology
• Costs and Benefits
Determine Data Mining Goals
• Data Mining Goals
• Data Mining Success Criteria
Produce Project Plan
• Project Plan
• Initial Assessment of Tools and Techniques
Collect Initial Data
• Initial Data Collection Report
Describe Data
• Data Description Report
Explore Data
• Data Exploration Report
Verify Data Quality
• Data Quality Report
Select Data
• Rationale for inclusion/Exclusion
Clean Data
• Data Cleaning Report
Construct Data
• Derived Attributes
• Generated Reports
Integrate Data
• Merge Data
Format Data
• Reformatted Data
Select Modeling Technique
• Modeling Technique
• Modeling Assumptions
Generate Test Design
• Test Design
Build Model
• Parameter Setting
• Models
• Model Description
Assess Model
• Model Assessment
• Revised Parameter Settings
Evaluate Results
• Assessment of Data
• Mining Results w.r.t business Success Criteria
• Approved Models
Review Process
• Review Of Process
Determine Next Steps
• List of Possible Actions Decision
Plan Deployment
• Deployment Plan
Plan Monitoring and Maintenance
• Monitoring and Maintenance of Plan
Final Product Review
• Final Report
• Final Presentation
Review Project
• Experience
• Documentation

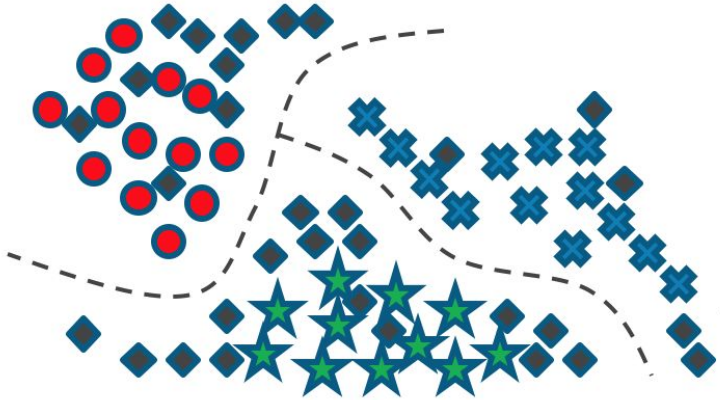
Different Techniques



Supervised Learning



Unsupervised Learning



Semi Supervised Learning

More

Reinforcement Learning

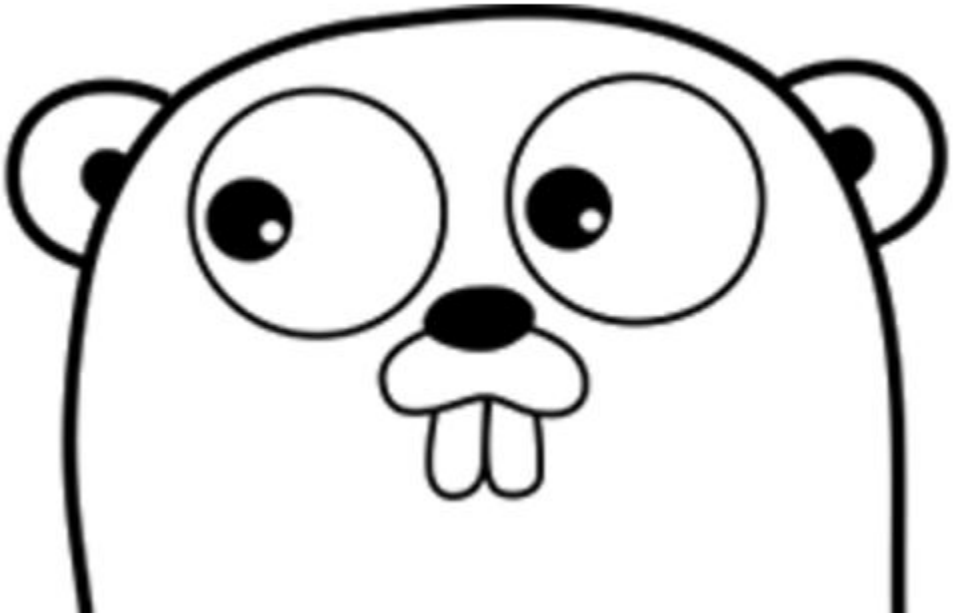
Forecasting

Optimization

Neural Network

Deep Neural Networks

Why golang?



Design Goals

Make managing concurrent/distributed systems easy

Improve collaboration with developers

Facilitate evolving codebases (refactoring etc.)

Very efficient and easy to build and deploy

Advantages of using golang for data science

Fun to write Go Code!

Very Fast in Runtime and Compilation

Easy Parallelization quite efficient compared to traditional languages like R(single threaded) and Python has Global interpreter lock

Portable and has Cross-compilation, also can call other languages from Go

Type System: safety of static typing, with a flexibility of dynamic and interfaces

Native Concurrency and Parallelism implemented (Routines, Channels, Events)

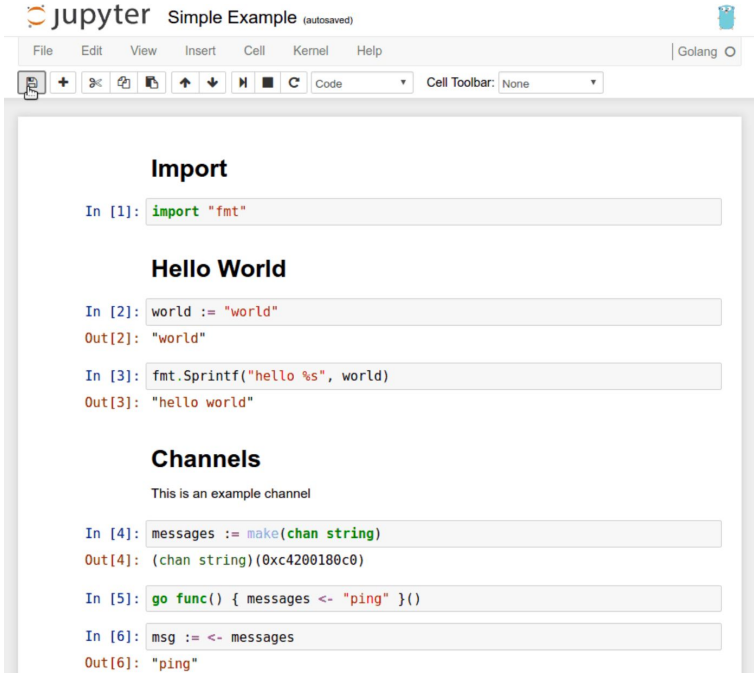
BUT, Just that Go is very new so there is lots of WIP!

Lot of libraries are existing however, some require heavy tuning

Go Notebooks

Jupyter notebook binding for Golang

https://github.com/gopherds/gophernotes



Data munging

https://github.com/kniren/gota

- Load/save CSV data
- Load/save XML data
- Load/save JSON data
- Parse loaded data to the given types (Currently supported: , , &)
- Row/Column subsetting (Indexing, column names, row numbers, range)
- Unique/Duplicate row subsetting
- Conditional subsetting (i.e.:)
- DataFrame combinations by rows and columns (cbind/rbind)
- DataFrame merging by keys (Inner, Outer, Left, Right, Cross)
- Function application over rows
- Function application over columns
- Statistics and summaries over the different features (Type dependant)
- Value counting (For histogram representations)
- Conversion between wide and long formats

Mathematical Operations

https://github.com/gonum

https://github.com/gonum/unit: Package for converting between scientific units

https://github.com/gonum/mathext: mathext implements basic elementary functions not included in the Go standard library

https://github.com/gonum/matrix: Matrix packages for the Go language

https://github.com/gonum/plot: A repository for plotting and visualizing data

https://github.com/gonum/blas: Basic Linear Algebra Sub Programs Implementation

https://github.com/gonum/graph: Graph packages for the Go language

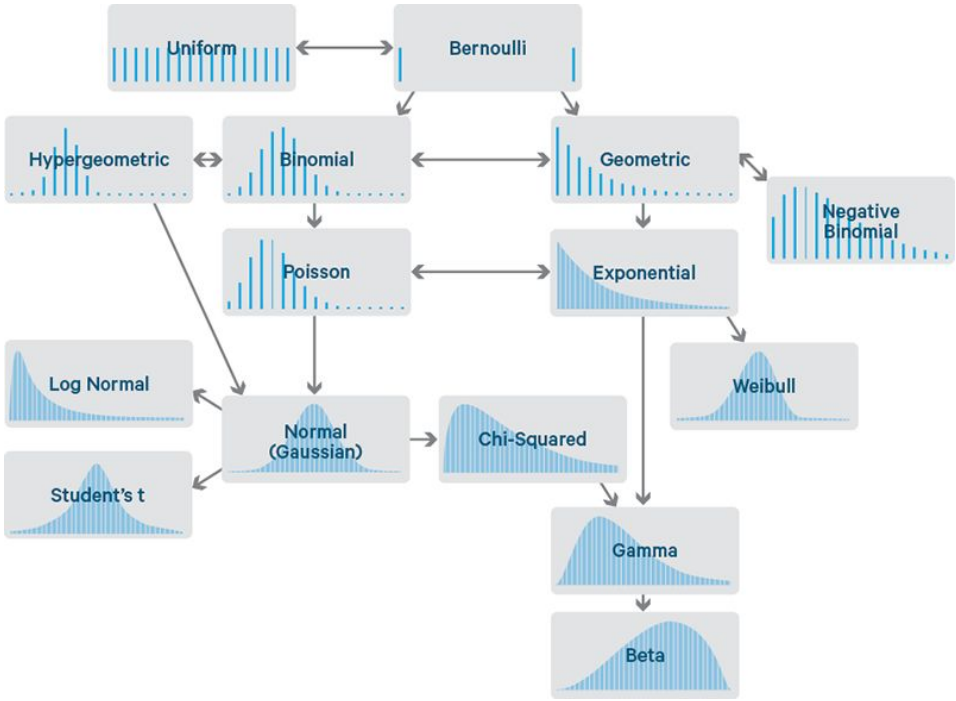
https://github.com/gonum/lapack: Linear Algebra Package

Probability Distributions

A probability function maps the possible values of x against their respective probabilities of occurrence, p(x)

p(x) is a number from 0 to 1.0.

The area under a probability function is always 1.



Probability Distribution in Go

https://github.com/e-dard/godist: Basic probability functions

https://github.com/chobie/go-gaussian: Gaussian (Normal Distribution)

Go Charting

gonum/plot – gonum/plot provides an API for building and drawing plots in Go.

goraph – A pure Go graph theory library(data structure, algorithm visualization).

SVGo: The Go Language library for SVG generation.

Text Extracting and Processing

Extracting

gocrawl: Polite, slim and concurrent web crawler.

Text Indexing

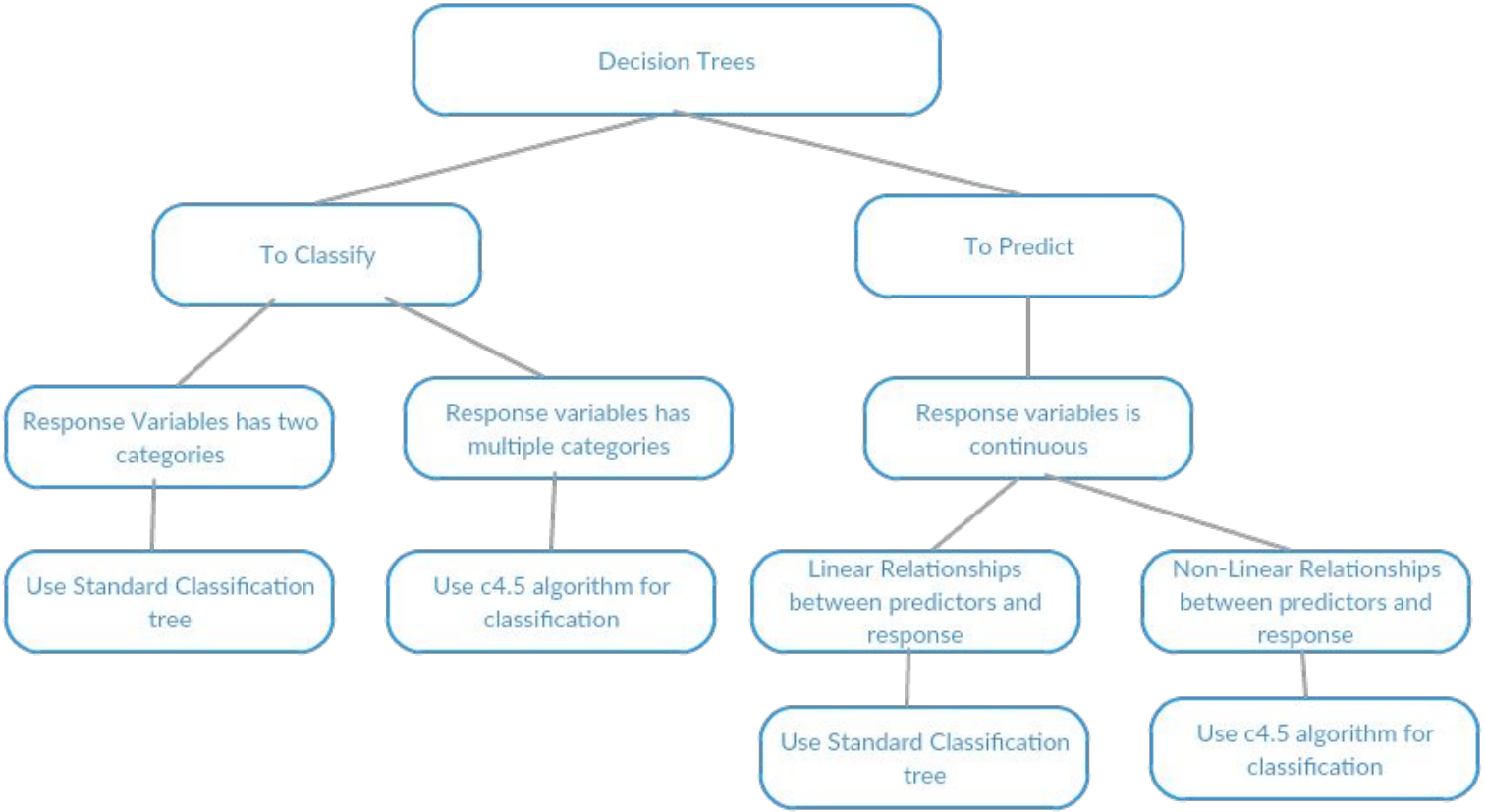
bleve: A modern text indexing library for go.

fulltext: Pure Go full text indexer and search library.

golucene: Go port of Apache Lucene.

golucy: Go bindings for the Apache Lucy full text search library.

Classification



Classification, Decision Trees in Go

Hector https://github.com/xlvector/hector - Golang machine learning lib. Currently, it can be used to solve binary classification problems.Logistic Regression , Factorized Machine , CART, Random Forest, Random Decision Tree, Gradient Boosting Decision Tree & Neural Network

Decision Trees in Go - https://github.com/ajtulloch/decisiontrees - Gradient Boosting, Random Forests, etc. implemented in Go

CloudForest - https://github.com/ryanbressler/CloudForest - Fast, flexible, multi-threaded ensembles of decision trees for machine learning in pure Go (golang). CloudForest allows for a number of related algorithms for classification, regression, feature selection and structure analysis on heterogeneous numerical / categorical data with missing values.

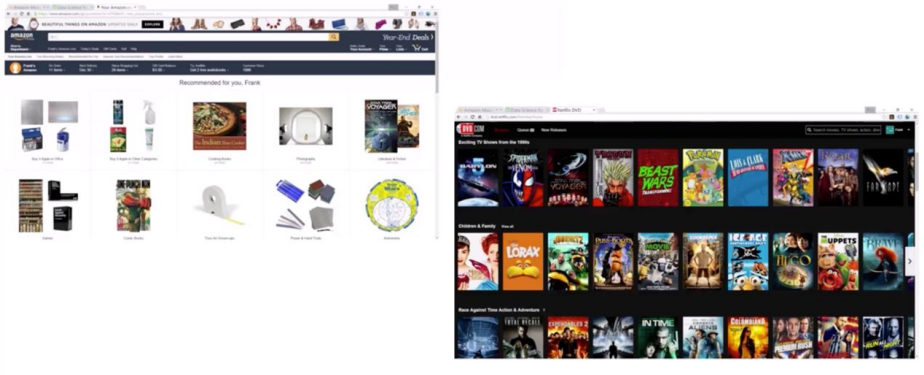
Random Forest Implementation: https://github.com/fxsjy/RF.go

Recommendation Engines: Collaborative Filtering

User - User based recommendation

Object - Object based recommendation

User - Object based recommendation



Recommendation Engines in Go

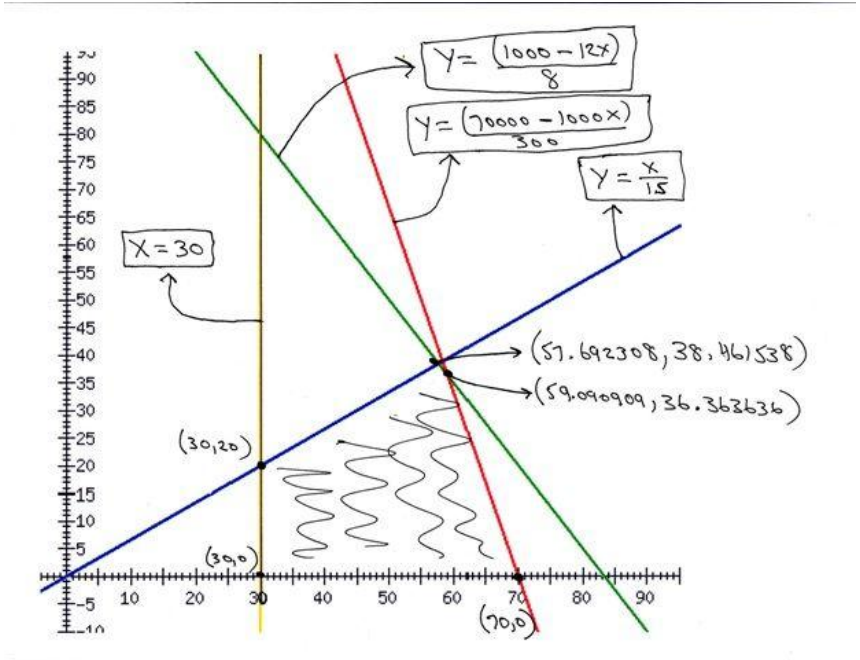
Collaborative Filtering (CF) Algorithms in Go - https://github.com/timkaye11/goRecommend

Recommendation engine for Go - https://github.com/muesli/regommend

Optimization and Linear Algebra

Each optimization problem consists of three elements:

- **decision variables**: describe our choices that are under our control;
- **objective function**: describes a criterion that we wish to minimize (e.g., cost) or maximize (e.g., profit);
- **constraints**: describe the limitations that restrict our choices for decision variables.



Sample Optimization Problem

A company produces copper cable of 5 and 10 mm of diameter on a single production line with the following constraints:

- The available copper allows to produces 21000 meters of cable of 5 mm diameter per week.
- A meter of 10 mm diameter copper consumes 4 times more copper than a meter of 5 mm diameter copper.

Due to demand, the weekly production of 5 mm cable is limited to 15000 meters and the production of 10 mm cable should not exceed 40% of the total production. Cable are respectively sold 50 and 200 euros the meter.

What should the company produce in order to maximize its weekly revenue?

Linear Algebra in Go

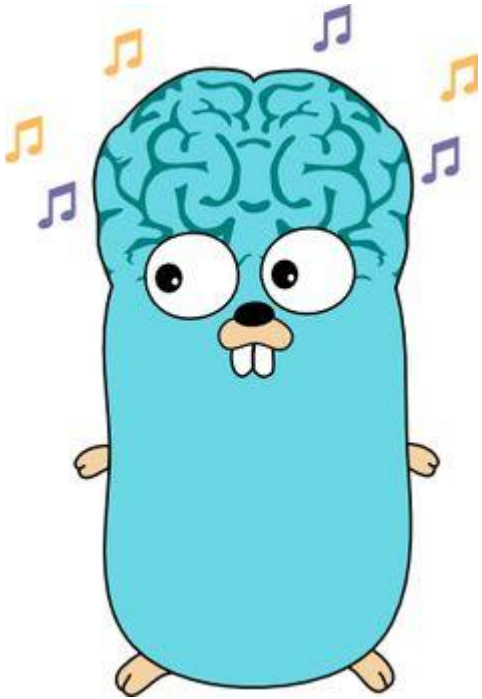
Linear Algebra for Go & Matrix Library: https://github.com/skelterjohn/go.matrix

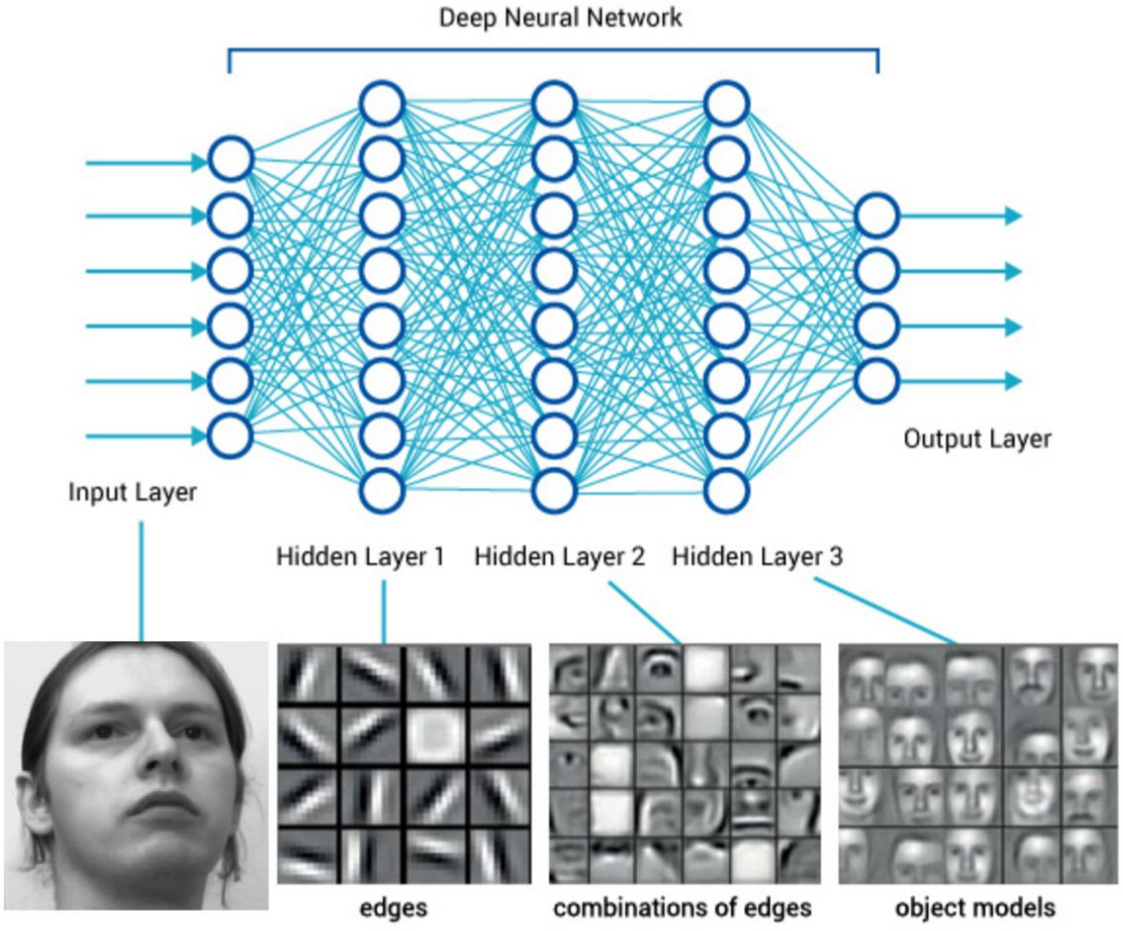
Mat64: Package mat64 provides basic linear algebra operations for float64 matrices.: https://godoc.org/github.com/gonum/matrix/mat64

BLAS Implementation for Go: https://github.com/gonum/blas

liblinear bindings for Go: https://github.com/danieldk/golinear

Neural Networks and Deep Learning




Deep Neural Network
Input Layer
Hidden Layer 1
Hidden Layer 2
Hidden Layer 3
Output Layer
edges
combinations of edges
object models

Neural Networks in Go

Neural Networks written in go : https://github.com/goml/gobrain

Go Fann - https://github.com/white-pony/go-fann

Multi-Layer Perceptron Neural Network - https://github.com/schuyler/neural-go

Genetic Algorithms library written in Go / golang - https://github.com/thoj/go-galib

Image Processing:

https://github.com/h2non/bimg: Small Go package for fast high-level image processing using libvips via C bindings

https://github.com/lazywei/go-opencv: Go Bindings for OpenCV

TensorFlow and Caffe support

Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by the Berkeley Vision and Learning Center (BVLC)

https://github.com/wmyaoyao/gocaffe

TensorFlow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them

https://github.com/tensorflow/tensorflow/issues/10

Gorgonia: https://github.com/chewxy/gorgonia: Similar to theano

Generic Machine Learning Libraries (More Stable)

GoLearn: https://github.com/sjwhitworth/golearn: One of the most prominent Go Machine Learning library, A very similar implementation as scikit-learn, most implemented in Go with some c++ bindings

GoML: https://github.com/cdipaolo/goml: Algorithms that learning, used for implementation of learning on the wire, running algorithms while the data is in the streams, channels, very well tested, extensive documentation.

Gorgonia: https://github.com/chewxy/gorgonia, very similar implementation to theano, allows us to define behavior about neural networks at a high level, but much much easier to deploy on various interfaces than theano

Machine Learning libraries for Go Lang: https://github.com/alonsovidales/go_ml:

MLGo: https://code.google.com/p/mlgo/

Algorithms implemented across various libraries

- Linear Regression

- Logistic Regression

- Neural Networks

- Collaborative Filtering

- Gaussian Multivariate Distribution for anomaly detection systems

- Gaussian mixture model clustering

- k-means, k-medians, k-medoids clustering

- single-linkage hierarchical clustering

- forecasting (https://github.com/datastream/holtwinters)

System Architectures

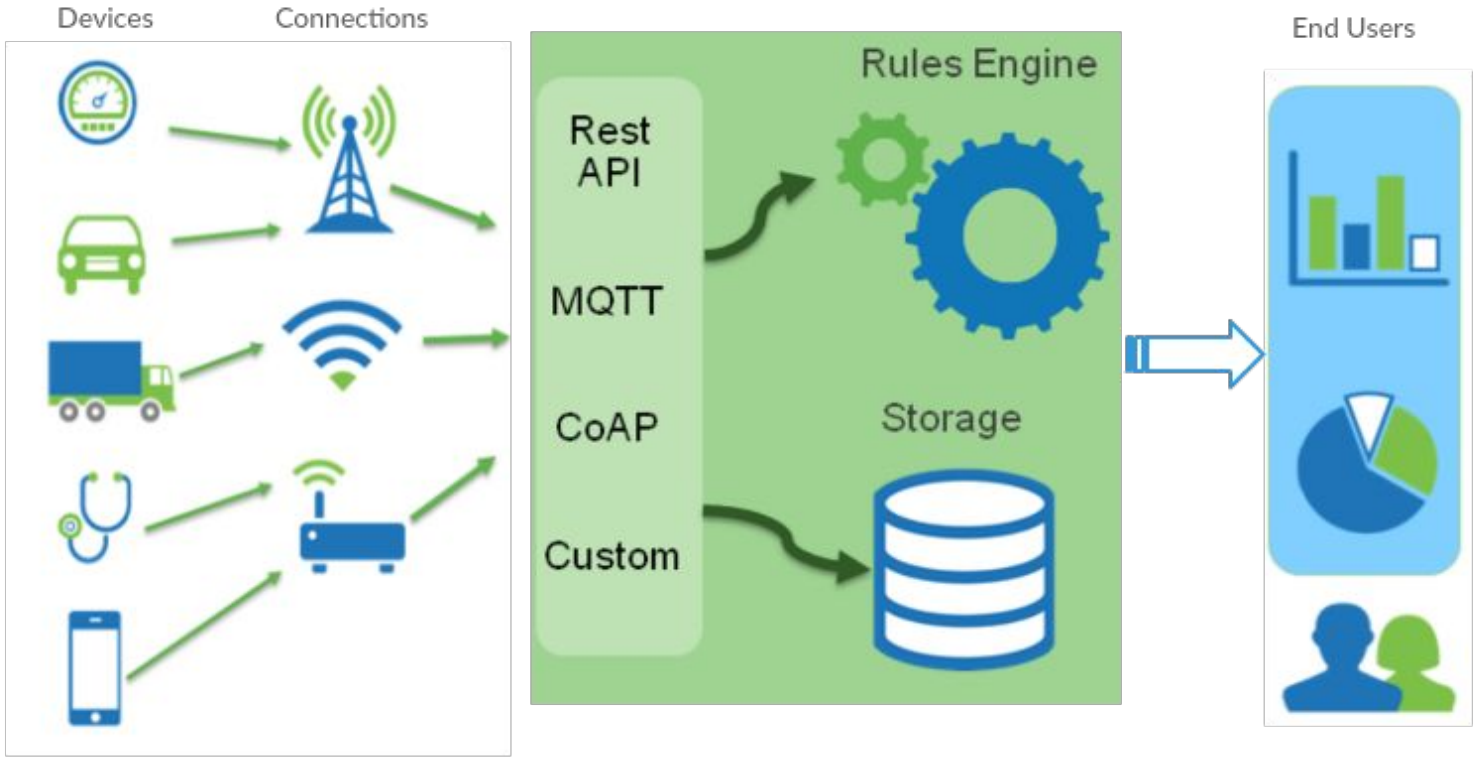
Use Cases

Energy Analytics

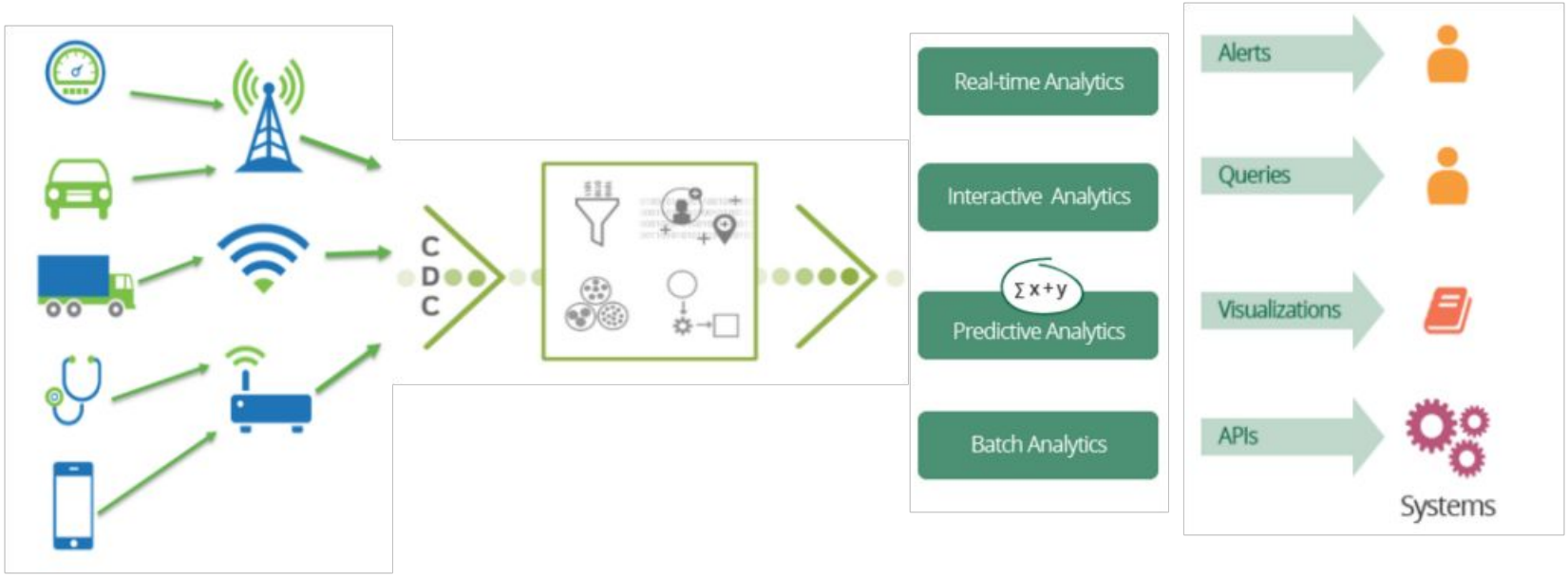
Transactional Frauds in Banking

Network Analytics

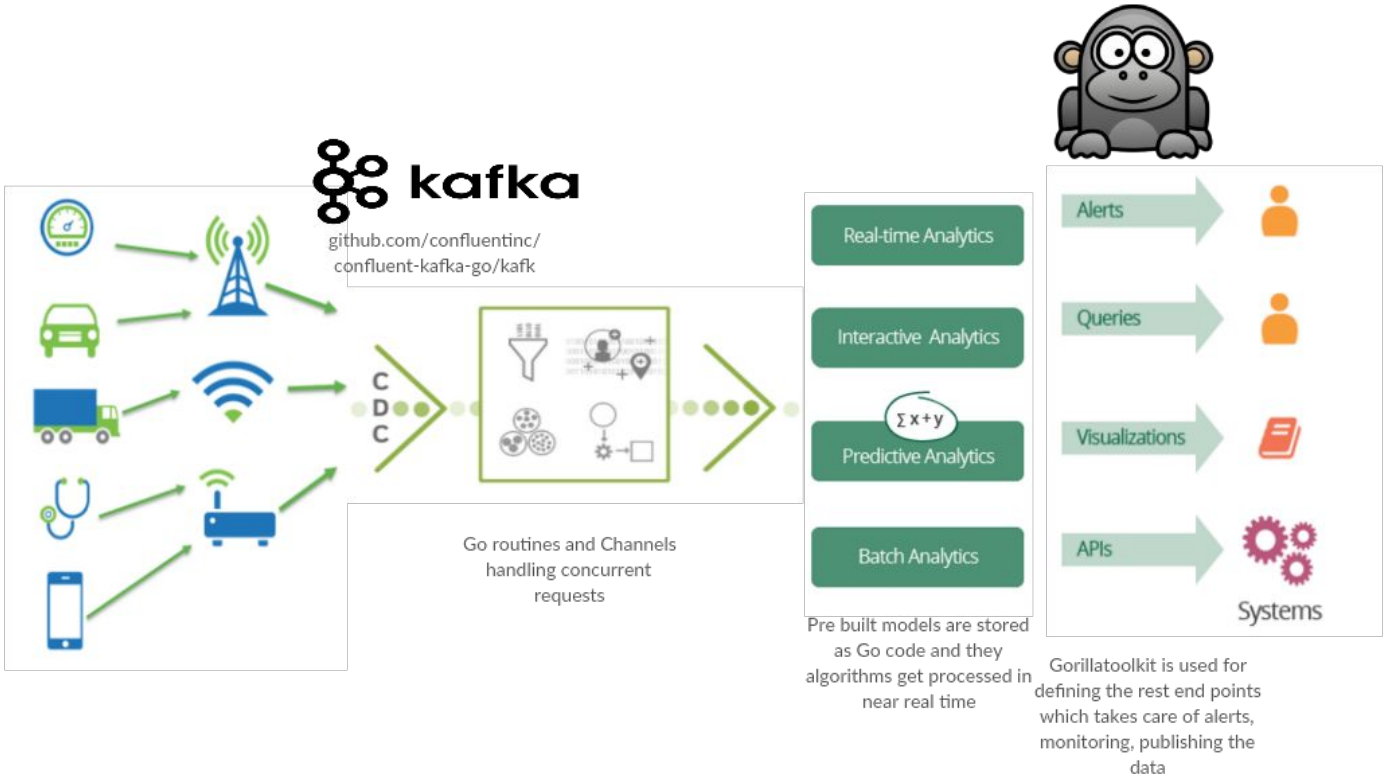
Energy Analytics



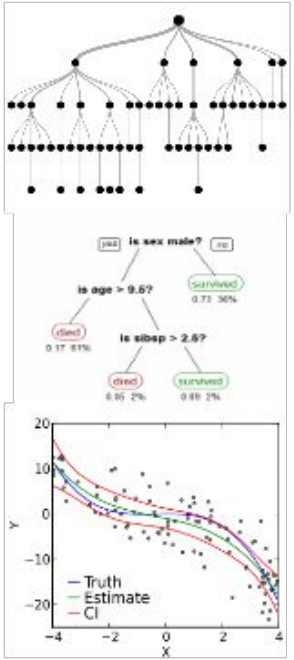
Architecture overview



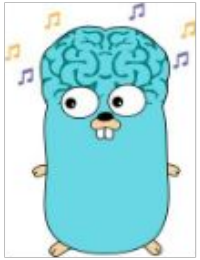
Architecture overview



Models



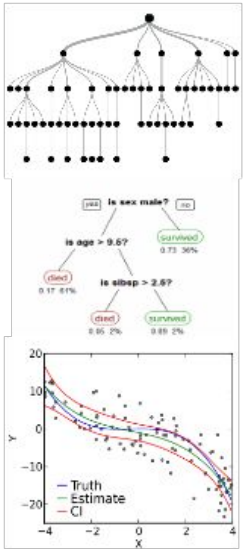
Using Go ML for running the algorithms in real time, the whole process would go through a single process chain



Native HTTP Apps

include traditional, batch learning interfaces, goml includes many models which let you learn in an online, reactive manner by passing data to streams held on channels.

Models



Rules

The rules engine, PMML, JSON rules is coverted to Go Land code

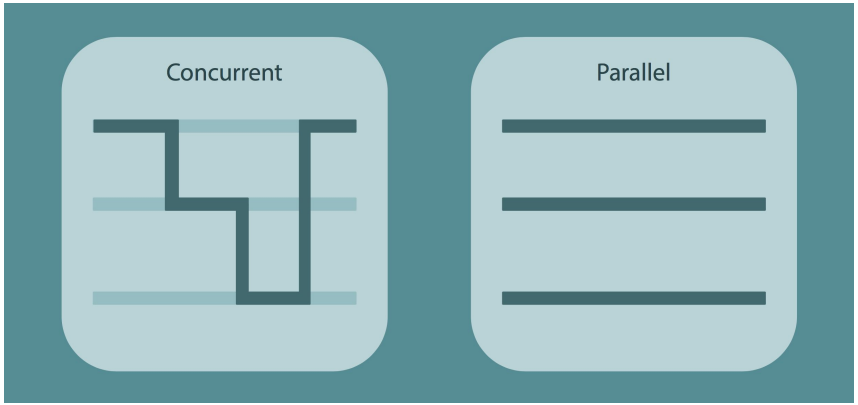
Native HTTP Apps

Concurrency

No Thread Primitives

Goroutines

Channels



Design Takeaways

Design decoupled, interface contracts enabled code

Write resilient batching, draining, stateless code

HTTP native apps for monitoring, alerting, processing was great

No tail-call optimization, some of the recursive algorithm implementation slower than Python based alternatives

Sufficient amount of tuning is required for optimizing performance

State of Go as a language for Machine Learning

A purely Go solution means fewer pieces from different languages that would have to be packaged and deployed together.

Great Community of developers

Using GO’s concurrency, fast runtime, and compilation capabilities very efficient codes can be written.

There are several open source libraries for various algorithms however, they are still in WIP, with specific tuning and customizations performs quite well in several scenarios

The ecosystem is still evolving, Let’s contribute in building an good ecosystem of machine learning with Go!

Good luck!
&
Thank You!