JOHN CLEM

https://www.linkedin.com/in/john-clem

Recent Projects

Character-based Deep Learning for Medical NLP

A character-based convolutional neural network (CNN) was experimented with traditional term frequency based approaches (ie: bags, ngram, TFIDF) in many cases for Natural Language Proceccing (NLP) of the provider notes within EHR data (used MIMIC III dataset). The novel CNN approach was proven competitive with traditional approaches where there is massive data available and especially where the data is noisy. This method could have many uses in healthcare such as treatment recommendation, helping doctors to quickly focus on more likely conditions of a patient via diagnosis prediction, etc..

The Python Scikit-learn library was used for traditional methods while the novel CNN approach was run on a server with GPUs and used Python's Keras library with a Tensor Flow backend all inside an NVIDIA-docker container.

Big Data Analytics for Healthcare – Various

Proper predictive modeling isn't a single algorithm, but evaluating and identifying the optimal over many pipelines. Using massive amounts of raw EHR data, multiple efforts from generating and evaluating pipelines that compute patient similarity for cohort construction and targeted treatment plans to predictive models for patient conditions or identifying data-backed treatment recommendations to doctors when a patient is not responding appropriately or quickly enough to the prescribed treatment. Naturally, this included exploratory data analysis; building pipelines (including ETL, feature engineering, rule-based as well as unsupervised phenotyping where appropriate, model implementation, and model validation); and constructing pipeline experiments and evaluation reports. Libraries were harnessed for basic ML legwork, but novel model and evaluation algorithms were custom engineered.

The technology stack utilized and deployed was Apache Hadoop (HDFS, map, reduce, Hive, & Pig), Spark (MLLib & GraphX), and Zeppelin in various containerized, virtual, and cloud environments using machine learning and data mining concepts in Scala, Python, Java, SQL, HQL, & PIG.

Analytics, Modeling, and Recommendations on Movie Data

Using the R language and approx 40k records of raw movie data with 39 attributes, analyze, visualize, and report on the interpretation offering recommendations and predictions. One of the main themes was to investigate the relationship between the attributes and a movie's box office success. This included making genre-based recommendations for release dates designed to maximize movie revenue. The recommendations were accompanied with supporting rationale and visualization that and bounded by the appropriate disclaimers. Provided insights toward goals through analysis and statistical evaluation of various features and movie ratings.

Another main theme was improving on predictive profit models using attribute featurization, feature transformation, and feature selection techniques. Part of improving prediction quality was graphing learning error, preventing overfitting, and determining the 'best' amount of training -vs- testing data.

Logistic Regression: Mathematical Origins through Model Evaluation

Beginning with a calculus-based detailed derivation of Logistic Regression (LR) from its roots in statistical learning, design and implement a system to train an LR model using gradient descent and evaluate the computational complexity. Using R, train and evaluate predictive capability using available breast cancer data. Analyze considering different convergence criteria and evaluate performance depending on varying the amount of training data. Demonstrate performance in terms of accuracy and the LR loss function.

Analytics and Telling the Story with Data

Using automotive vehicle data, identify, show, and communicate various relationships and patterns within the data such as: fuel efficiency by manufacturer, three-way relationship between model class / hwy mpg / city mpg, etc. Using data on over 50k diamonds, evaluate and articulate patterns with various attributes and attribute relationships. Draw and support conclusions textually and graphically using R.

Multi-Agent Learning in a Soccer Game

Using a simple two player soccer game as the context, implement, explore, and analyze competitive and cooperative learning in multi-agent systems. Learning algorithms implemented in Java and compared were Correlated-Q, Foe-Q, Friend-Q, and standard Q-learning.

Temporal Difference Learning Applied to the Random Walk Problem

Replicate the TD-Lambda set of algorithms as introduced in Richard Sutton's 1988 seminal paper on Temporal Difference (TD) learning. Developed in Python and applied to the bounded random walk problem where predictions are made on the outcome of a random walk. Within dynamical systems where observations are made over time, TD-Lambda is very pertinent and can have an advantage over supervised learning. This is essentially learning and adapting on the fly -vs- the typical supervised learning approach of only making inferences after all the data is available.

Reinforcement Learning and Decision Making – Various

Mini-projects requiring theory to be applied to real decision and game contexts such as:

- Optimal decision-making in a betting game based on rolling an N-sided die
- Find an optimal policy for any given each state in a continuous-domain MDP in which an agent is attempting to traverse a path consisting of different types of terrain. The agent has different actions it can take, and these actions have different effectiveness in different terrain types, meaning there is stochastic noise in the agent's inputs/outputs.
- Implement a Knows What It Knows (KWIK) agent that knows when it can accurately address a problem -vs- not and has a finite limitation on when it 'doesn't know'.

Applied KWIK to the bar brawl problem requiring a prediction whether a fight will break out among a subset of the overall patrons where there is a peacemaker and an instigator. If the instigator is present but not the peacemaker, then a fight will break out, but if the instigator is not present or if the peacemaker is present, then no fight will occur. The agent must predict without initially knowing the identities of the instigator and the peacemaker.

- Finding the value of acting optimally a two-armed bandit (slot machine) setting

InfoSec – Various

Projects included active malware analysis within a sandboxed environment, analyzing and adjusting firewall configuration, buffer overflow exploit, implementing DES encryption CBC mode encryption and decryption and attempt a class-wide brute-force DES encryption attack, and web security including exploiting via XSRF, XSS, and SQL injection.

Machine Learning for Trading – Various

Projects included assessing and optimizing portfolios, simulating the market, developing trading strategies using technical analysis, implementing and assessing K-nearest neighbor and Q-learning traders from scratch, and implementing and assessing other learning traders including linear regression and bagging. A lot of graphs were used to visualize portfolio performance compared to the market, trading strategies, and technical analysis. All projects were developed in Python and heavily used the Pandas, NumPy, and matplotlib libraries.

Predicting Robot Motion

Given ten one-minute videos (with location data) of a randomly-moving real-life robot within a confined space with obstacles, predict the motion occurring for the two seconds immediately following the provided samples. Pulled from the same set of techniques indicated below except using the Unscented Kalman Filter (UKF) because of the highly non-linear state transition and prediction models. Developed in Python, the UKF provided the necessary motion predictions and this prediction was fed into an obstacle response model which produced the final prediction adjusted to account for the known obstacles in this domain.

Robot to Catch another 'Runaway' Robot

The objective of this project was to build the robot software to chase, and capture, a 'runaway robot' that was trying to escape. To succeed, the AI robot agent had to perform within a continuous environment where there was noise in both its sensors as well as controls and is only provided with very noisy location sensor updates from the runaway robot. The agent needed to localize the other runaway, accurately predict real-time future locations of the runaway and intercept it. The agent could only move as fast as the runaway, so careful planning was required. The below topics were implemented in Python throughout the course, with most included in this project:

- Localization
- Histogram, Kalman, and Particle Filters

- Motion planning and the search problem using a variety of algorithms
- Data smoothing
- PID Control (and tuning)
- Simultaneous Localization And Mapping (SLAM)

Solving Human Intelligence Exam with Cognitive AI Agent

Four phased projects solving the acclaimed Raven's Progressive Matrices intelligence test problems (RPMs) at increasing levels of difficulty. RPMs have been used on human subjects and been deemed as a reliable measure of human cognition and are based on solving visual analogy problems. Please see the following link (or search) for examples and more details: http://www.iqmindware.com/iq-mindware/how-to-do-a-raven-matrices-test/. My agent performed at an adolescent to young adult level.

In the final project, the agent (Java) received image files for various matrices up to a 3x3 size with an objective of making the correct selection from among a pool of possible solution images using cognitive techniques and armed with computer vision to 'see' the images. A key distinction in the 'knowledge-based' set of approaches (not ML / data-based) that the agent employed is in harnessing domain knowledge rather than tremendous amounts of data and these approaches are analogous to how a human would approach and reason over the problem...think cognitive computing and IBM Watson.

Machine Learning – Reinforcement Learning in Markov Decision Processes (MDPs)

Designed an agent to act within two maze-like obstacle courses I constructed. This requires a more rich, and direct, interaction between the agent and the environment than the typical supervised and unsupervised learning approaches. Experiments were conducted over these two domains allowing the agent to have a deterministic interface with the environment as well as challenging it with non-determinism. These two problems were solved in Java using both policy and value iteration where the agent understands the world as well as using Q-learning where the agent begins with no model of the world and learns everything along the way directly from the environment.

Machine Learning – Unsupervised Learning and Dim' Reduction

Explore, test various implementations, experiment, and analyze clustering and dimensionality reduction algorithms to increase performance of the typical supervised learning approaches and to help understand and explain the data.

- Addressed following algorithms (Java):
  - Clustering:
    - K-means
    - Expectation Maximization
  - Dimensionality Reduction
    - PCA

- ICA
- Randomized Projections
- Correlated Feature Selection (genetic algorithms based)

Machine Learning – Randomized Optimization

Explore, test various implementations, experiment, and analyze randomized search and optimization techniques. Optimized neural network weights using the first three optimization methods below significantly improving on the customary backpropogation approach for ANN tuning. Coded and analyzed optimization techniques in the four peaks, continuous peaks, traveling salesman, and knapsack optimization problem domains.

- Addressed following algorithms (Java):
  - Randomized hill climbing
  - Simulated annealing
  - Genetic algorithms
  - MIMIC

Machine Learning – Supervised Learning

Explore, test various implementations, experiment, and analyze supervised learning techniques. In this open-ended project, I used the Boston Housing and the CMU Statlog Heart Disease data. The Boston housing data is being addressed as the regression problem of learning and predicting housing values while the heart data set is treated as the classification problem of learning if a patient has heart disease or not and predicting the appropriate label.

- Addressed following algorithms (Java and Weka):
  - Decision trees with pruning
  - Artificial neural networks
  - Boosting
  - Support Vector Machines
  - K-nearest neighbors
- Performed data pre-processing and cross-validation
- Tuned for best accuracy (RMSE) and analyzed and reported on the data and algorithms and their performance