Lecture 14: Link | Syllabus
**Class Week 14/15*

Database Systems by Coronel & Morris

*** 14.4-a** *Data Mining*

Data mining refers to analyzing massive amounts of data to uncover hidden trends, patterns, and relationships; to form computer models to simulate and explain the findings; and then to use such models to support business decision making. In other words, data mining focuses on the discovery and explanation stages of knowledge acquisition.

How is ML different from DM?

Machine learning is the process of TRAINING an algorithm on an EXISTING dataset in order to have it discover relationships (so as to create a model/pattern/trend), and USING the result to analyze NEW data.

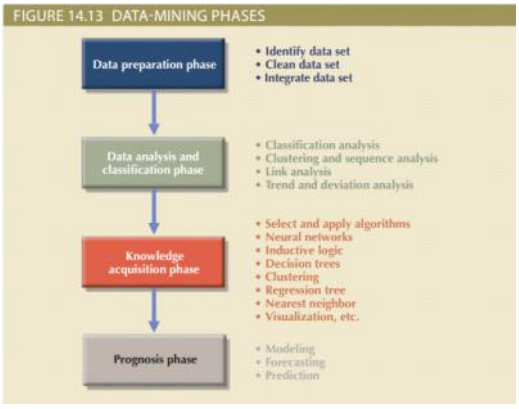
* Data mining consists of four general phases :

1. Data Preparation - In this phase the main data sets to be used by the data-mining operation are identified and cleansed of any data impurities. Because the data in the data warehouse is already integrated and filtered, the data warehouse usually is the target set for data-mining operations

2. Data Analysis & Classification - The data analysis and classification phase studies the data to identify common data characteristics or patterns. During this phase, the data-mining tool applies specific algorithms to find :
   - Data groupings, classifications, clusters, or sequences
   - Data dependencies, links, or relationships
   - Data patterns, trends, and deviations

3. Knowledge Acquisition - this phase uses the results of the data analysis and classification phase. During the knowledge acquisition phase, the data-mining tool (with possible intervention by the end user) selects the appropriate modeling or knowledge acquisition algorithms

4. Prognosis - Although many data-mining tools focus on the knowledge–discovery phase, others continue to the prognosis phase. In that phase the data-mining findings are used to predict future behavior and forecast business outcomes.



* *Data mining can be run in two modes:*

1. Guided. The end user guides the data-mining tool step by step to explore and explain known patterns or relationships. In this mode, the end user decides what techniques to apply to the data.
2. Automated. In this mode, the end user sets up the data-mining tool to run automatically and uncover hidden patterns, trends, and relationships. The data-mining tool applies multiple techniques to find significant relationships

Machine Learning (ML)

***** *Types of AI, Machine Learning ("ML"), types of ML*

Here is a good way [after Arend Hintze] to classify AI types (not just techniques!)...
Type I: Reactive machines - make optimal moves - no memory, no past 'experience'. Ex: game trees.
Type II: Limited memory - human-compiled/provided , one-shot 'past' 'experiences' are stored for lookup. Ex: expert systems, neural networks.
Type III: Theory of Mind - "the understanding that people, creatures and objects in the world can have thoughts and emotions that affect the AI programs' own behavior".
Type IV: Self-awareness - machines that have consciousness, that can form representations about themselves (and others).

Type I AI is simply, application of rules/logic (eg. chess-playing machines).
Type II AI is where we are, today - specifically, this is what we call 'machine learning' - it is "data-driven AI"! Within the last decade or so, spectacular progress has been made in this area, ending what was called the 'AI Winter'.

As of now, types III and IV are in the realm of speculation and science-fiction, but in the general public's mind, they appear to be certainty in the near term :)

Practically speaking, there are exactly three types of AI that have been pursued, in the quest for human-level AI:
- inference-based: 'symbolic'
- goals/rewards-based: 'reinforcement'
- connection-based: 'neuro'

ML is the ONE subset of AI that is revolutionizing the world.
"Machine learning focuses on the construction and study of systems that can learn from data to optimize a performance function, such as optimizing the expected reward or minimizing loss functions. The goal is to develop deep insights from data assets faster, extract knowledge from data with greater precision, improve the bottom line and reduce risk."
- Wayne Thompson, SAS

ML comes in several flavors - the key types of machine learning include:
- Supervised learning
- Unsupervised learning
- Semi-supervised learning
- Reinforcement learning

* *Classification*

Supervised learning algorithms are "trained" using examples (DATA!] where in addition to features [inputs], the desired output [label, aka target] is known. The goal is to LEARN the patterns inherent in the training dataset, and use the knowledge to PREDICT the labels for new data.

Unsupervised learning is a type of machine learning where the system operates on unlabeled examples. In this case, the system is not told the "right answer." The algorithm tries to find a hidden structure or manifold in unlabeled data. The goal of unsupervised learning is to explore the data to find intrinsic structures within it using methods like clustering or dimension reduction.

For Euclidian space data: k-means clustering, Gaussian mixtures and principal component analysis (PCA)

For non-Euclidian space data: ISOMAP, local linear embedding (LLE), Laplacian eigenmaps, kernel PCA.

Use matrix factorization, topic models/graphs for social media data.

Semi-supervised learning is used for the same applications as supervised learning. But this technique uses both labeled and unlabeled data for training - typically, a small amount of labeled data with a large amount of unlabeled data. The primary goal is unsupervised learning (clustering, for example), and labels are viewed as side information (cluster indicators in the case of clustering) to help the algorithm find the right intrinsic data structure.

DM/ML Tools

* *Theory => Practice*

In the two previous lectures, we looked at a variety of algorithms for DM and ML, eg. kNN, clustering, neural networks.

Now we'll examine how these have been/can be implemented - using tools/frameworks or APIs/languages/hardware.

Note that in 'industry' (ie. outside academia and gov't), tools/APIs... are heavily used to build RL products - so you need to be aware of, and be knowledgeable in, as many of them as possible.

* *APIs/frameworks: part 1*

These are the most heavily used:
- TensorFlow ('TF')
- Spark MLlib: https://spark.apache.org/mllib/ and https://spark.apache.org/docs/2.2.0/ml-pipeline.html
- Keras: https://keras.io/ [a higher level lib, compared to TF etc]; here are all the types of Keras layers
- Torch, PyTorch: https://pytorch.org, http://torch.ch/
- scikit-learn: https://scikit-learn.org/stable/
- Caffe: https://caffe2.ai/, http://caffe.berkeleyvision.org/ [→ Caffe2 → PyTorch]
- Apache mxnet: https://mxnet.apache.org/ [multi-language APIs, GPU and cloud support...]
- CNTK: https://docs.microsoft.com/en-us/cognitive-toolkit/
- TL;DR: simply learn Keras or PyTorch, and if necessary, TF.

* *APIs/frameworks: part 2*

Upcoming/lesser-used/'internal'/specific:
- FBLearner Flow - Facebook's version of TensorFlow :)
- Apache Mahout - a collection of ML algorithms, in Java/Scala
- .NET ML: https://dotnet.microsoft.com/apps/machinelearning-ai/ml-dotnet
- fastai [on top of PyTorch]: https://github.com/fastai/fastai
- OpenVINO: https://software.intel.com/en-us/openvino-toolkit and https://www.youtube.com/watch?v=rUwayTZKnmA&t=1s [a tutorial]
- Turi: an alternative to Apple's CreateML: https://github.com/apple/turicreate
- LibSVM: https://www.csie.ntu.edu.tw/~cjlin/libsvm/
- LightGBM: https://github.com/Microsoft/LightGBM
- XGBoost: https://xgboost.ai/ [and, look at Tianqi's slides and talk]
- CatBoost: https://tech.yandex.com/catboost/
- Google - SEED: https://ai.googleblog.com/2020/03/massively-scaling-reinforcement.html
- Uber's 'Fiber', for distributed ML training: https://venturebeat.com/2020/03/26/uber-details-fiber-a-framework-for-distributed-ai-model-training/

- LOTS of smaller efforts: https://github.com/EthicalML/awesome-production-machine-learning

* *Cloud*

The virtually unlimited computing power and storage that a cloud offers, make it an ideal platform for data-heavy and computation-heavy applications such as ML.

Amazon: https://aws.amazon.com/machine-learning/ Their latest offerings make it possible to 'plug in' data analysis anywhere.
Google: https://cloud.google.com/products/ai/ [in addition, Colab is an awesome resource!]
Microsoft: https://azure.microsoft.com/en-us/services/machine-learning-studio/ [and AutoML] [aside: alternatives to brute-force 'auto ML' include 'Neural Architecture Search' [incl. this], pruning, and better network design (eg using ODEs - see this).
IBM Cloud, Watson: https://www.ibm.com/cloud/ai [eg. look at https://www.ibm.com/cloud/watson-language-translator]

Others:
- h2o: https://www.h2o.ai/products/h2o/ [supports R, Python, Java, Scala, JSON, native Flow GUI [similar to Jupyter], REST...]
- BigML: https://bigml.com/features#platform
- FloydHub: https://www.floydhub.com/
- Paperspace: https://ml-showcase.paperspace.com/
- Algorithmia, eg. https://info.algorithmia.com/ and https://demos.algorithmia.com/

* *Pretrained ML models*

A pre-trained model includes an architecture, and weights obtained by training the architecture on specific data (eg. flowers, typical objects in a room, etc) - ready to be deployed.

Eg. this is simple object detection in the browser! You can even run this detector on a command line.

TinyMOT: https://venturebeat.com/2020/04/08/researchers-open-source-state-of-the-art-object-tracking-ai

Apple's CreateML is useful for creating a pre-trained model, which can then be deployed (eg. as an iPad app) using the companion CoreML product. NNEF and ONNX are other formats, for NN interchange.

Pre-trained models in language processing, include Transformer-based BERT and GPT-2. Try this demo (of GPT etc). There is GPT-3 currently available, GPT-4 in the works, Wu Dao 2.0, MT-NLG...

There are also, combined (bimodal) models, based on language+image data.

* *Tools*

Several end-to-end applications exist, for DM/ML. Here popular ones.

Weka is a Java-based collection of machine learning algorithms.
RapidMiner uses a dataflow ("blocks wiring") approach for building ML pipelines.
KNIME is another dataflow-based application.
TIBCO's 'Data Science' software is a similar (to WEKA etc) platform. Statistica [similar to Mathematica] is a flexible, powerful analytics software [with an old-fashioned UI].
bonsai is a newer platform.

To do ML at scale, a job scheduler such as from cnvrg.io can help.
SynapseML is a new ML library from Microsoft.

There are a variety of DATAFLOW ('connect the boxes') tools! This category is likely to become HUGE:
- Perceptilabs: https://www.perceptilabs.com/
- Lobe: https://insights.dice.com/2018/05/07/lobe-deep-learning-platform/
- https://www.producthunt.com/posts/datature
- smartpredict: https://smartpredict.ai/
- StackML: https://stackml.com/ [RIP]
- Baseet: https://baseet.ai/ [RIP]

* *Languages*

These languages are popular, for building ML applications (the APIs we saw earlier, are good examples):
- Python
- R
- Julia [Python 'replacement'?!]
- Wolfram
- JavaScript - this is a good list of JS-based libraries [look at ConvnetJS for nice demos]
- Scala - a functional+OO language - here is a roundup of libraries [these are in addition to Spark's MLlib Scala API]
- Java - another robust language for building ML libs [we already saw WEKA] and apps
- Jupyter [an environment, not a language] (eg. here is a collection of ML notebooks - as an exercise, run them all in Colab!) [also, here are notebooks for 'everything'!]

* *Hardware*

Because (supervised) ML is computationally intensive, and detection/inference needs to happen in real-time almost always, it makes sense to accelerate the calculations using hardware. Following are examples.

Google TPU: TF is in hardware! Google uses a specialized chip called a 'TPU', and documents TPUs' improved performance compared to GPUs. Here is a pop-sci writeup, and a Google blog post on it.

Amazon Inferentia: a chip, for accelerating inference (detection): https://aws.amazon.com/machine-learning/inferentia/
NVIDIA DGX-1: an 'ML supercomputer': https://www.nvidia.com/en-us/data-center/dgx-1/ [here is another writeup]
Intel's Movidius (VPU): https://www.movidius.com/ - on-device computer vision

In addition to chips and machines, there are also boards and devices:

- Pixy2: https://pixycam.com/ - camera + ML in a single board
- Coral: https://coral.withgoogle.com/
- Jetson Nano: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/
- Movidius NCS: https://software.intel.com/en-us/movidius-ncs

Overall, there's an explosion/resurgence in 'chip design', for accelerating AI training, inference. In April '21, NVIDIA announced its new A30 and A10 GPUs, at the annual [GTC] conference.

* *Summary*

We looked at a plethora of ways to 'do' ML. Pick a few, and master them - they complement your coursework-based (theoretical) knowledge, and, make you marketable to employers! Aside: LOTS of salaries etc., revealed here :)

Also, FYI - in industry (G-MAFIA/FAANG/MAMMA MIA, BAT, more!), ML is part of a bigger 'production pipeline':