Machine learning for algo trading

An introduction for non-mathematicians

Dr. Aly Kassam
Overview

• High level introduction to machine learning
• A machine learning bestiary
• What has all this got to do with trading?
• Some pitfalls to look out for
• What resources exist?
• What next?
ML – What is it??

“Machine learning is a scientific discipline that deals with the construction and study of algorithms that can learn from data. [1] Such algorithms operate by building a model based on inputs [2]:² and using that to make predictions or decisions, rather than following only explicitly programmed instructions.”

A ML bestiary

- Decision trees
- Naïve Bayes
- Bayesian Nets
- Neural Networks
- Genetic algorithms
- Logistic regression
- Hidden Markov Models
- Associated rule learning
- K-nearest neighbours
- K-means clustering
- Principal component analysis
- Support Vector Machines

Quantitative Support Services
**A ML bestiary**

**Supervised learning:**
- K-nearest neighbours
- Decision trees
- Logistic regression
- Support Vector Machines
- Bayesian networks
- Neural networks

**Unsupervised learning:**
- K-means clustering
- Hidden Markov Models
- Principal component analysis
- Associate rule learning
Supervised vs unsupervised

**Supervised learning**
- Start with a labelled “training” data set
- Used for producing predictive models
- Examples are:
  - Classification
  - Regression

**Unsupervised learning**
- No labelling on the data
- Used for producing descriptive models
- Examples are:
  - Clustering
  - Association learning
Supervised learning
Support Vector Machines

- Used to separate data into different classes
- It’s an example of a linear separator
- Works in multiple dimensions
- Kernels can be used to add nonlinearity

Source: www.stackoverflow.com
What is the support vector?

- The idea behind SVM is to find the hyperplane that results in the greatest margin between the support vectors.
Decision trees

- Classify data based on a sequence of questions
- Classification trees
  - Separate data into distinct classes
- Regression trees
  - Make real number predictions
- Usually trees are combined into ensemble models

Source: “Machine Learning”, Peter Flach, CUP
Decision tree - Data

From a spreadsheet to a decision node

Examples described by attribute values (Boolean, discrete, continuous, etc.)
E.g., situations where I will/won’t wait for a table:

<table>
<thead>
<tr>
<th>Example</th>
<th>Alt</th>
<th>Bar</th>
<th>Fri</th>
<th>Hun</th>
<th>Pat</th>
<th>Price</th>
<th>Rain</th>
<th>Res</th>
<th>Type</th>
<th>Est</th>
<th>WillWait</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Some</td>
<td>$$$</td>
<td>F</td>
<td>T</td>
<td>French</td>
<td>0–10</td>
<td>T</td>
</tr>
<tr>
<td>X2</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Full</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Thai</td>
<td>30–60</td>
<td>F</td>
</tr>
<tr>
<td>X3</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Some</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Burger</td>
<td>0–10</td>
<td>T</td>
</tr>
<tr>
<td>X4</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Full</td>
<td>$$$</td>
<td>F</td>
<td>F</td>
<td>Thai</td>
<td>10–30</td>
<td>T</td>
</tr>
<tr>
<td>X5</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>Full</td>
<td>$$$</td>
<td>F</td>
<td>T</td>
<td>French</td>
<td>&gt;60</td>
<td>T</td>
</tr>
<tr>
<td>X6</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Some</td>
<td>$$</td>
<td>T</td>
<td>T</td>
<td>Italian</td>
<td>0–10</td>
<td>T</td>
</tr>
<tr>
<td>X7</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>None</td>
<td>$</td>
<td>T</td>
<td>F</td>
<td>Burger</td>
<td>0–10</td>
<td>F</td>
</tr>
<tr>
<td>X8</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Some</td>
<td>$$</td>
<td>T</td>
<td>T</td>
<td>Thai</td>
<td>0–10</td>
<td>T</td>
</tr>
<tr>
<td>X9</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Full</td>
<td>$</td>
<td>T</td>
<td>F</td>
<td>Burger</td>
<td>&gt;60</td>
<td>F</td>
</tr>
<tr>
<td>X10</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Full</td>
<td>$$$</td>
<td>F</td>
<td>T</td>
<td>Italian</td>
<td>10–30</td>
<td>F</td>
</tr>
<tr>
<td>X11</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>None</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Thai</td>
<td>0–10</td>
<td>F</td>
</tr>
<tr>
<td>X12</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Full</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Burger</td>
<td>30–60</td>
<td>T</td>
</tr>
</tbody>
</table>

Classification of examples is positive (T) or negative (F)

Source: “Machine learning – Decision trees” by Nando de Freitas, on YouTube
**Decision trees – model**

![Decision tree diagram](image)

Source: “Machine learning – Decision trees” by Nando de Freitas, on YouTube
**K-nearest neighbours**

- “Similar” historical points forecast likely future behaviour
  
  - Can work on scalar values (find the last k similar values)
  - Can also work with vectors
  - Defining a pattern as a vector, forms the basis of pattern recognition
  - See:
    - “Machine Learning and Pattern Recognition for Algorithmic Forex and Stock Trading” (all 19 videos!) on YouTube for an example of this...
Unsupervised learning
Unsupervised learning

• As mentioned before, unsupervised learning is more concerned with **descriptive models**, e.g.
  – Clustering (hierarchical or k-means)
  – Association rule learning (“if this, then that”)
  – Dimensionality reduction (e.g. PCA)
K-means clustering

- Divide a universe of data into a collection of similar sets
- Find relationships in the data and group similar items together
• A system can exist in a number of states
• Each state can produce a number of measurable outcomes based on a probability distribution
• Transitions between states can occur with probabilities defined in a transition matrix
• Transitions to new states depend only on the current state (hence a Markov process)
What has this to do with trading?

- Questions:
  - What category of problem is trading?
  - What are good inputs? What are the outputs?

  - ML can apply to many aspects of the trading problem
  - Sorting and classifying inputs
  - Making predictions based on decisions
  - Estimating probabilities of movements or outcomes
**Feature selection**

- Features are important both for
  - Input data
  - “Predicted” response (outputs)
- Too many features → overfitting... Be careful!

- **Input features**
  - Technical indicators
  - Changes in Prices, Volumes, Ratios
  - External series
  - News feeds
  - Time of day
  - ...?

- **Output features**
  - Discrete moves (up/down/flat etc.)
  - True/false
  - Probabilities
  - ...?
Example 1 – Rao and Hong 2010

Rao and Hong try to predict future prices of 10 stocks and 1 index

They used:
- K-means clustering
- Hidden Markov model
- Support vector machine

Inputs were:
- EMA7, EMA50, EMA200
- MACD, RSI, ADX,
- High, Low, Close, Close>EMA200, lagged profits
**Example 1 – Rao and Hong 2010**

- **Methodology 1 – unsupervised (HMM)**
  - (1) use the K-means to identify 5 hidden states (clusters)  
    [big price up, small price up, no change, small price down, big price down]
  - (2) Use HMM and daily lag profits to determine:
    “What is the probability of seeing a big price drop tomorrow given today’s state and observations.”

- **Methodology 2 – supervised (SVM)**
  - Classify each training day as a **buy/sell** signal, and use the 10 inputs described above to train the SVM
  - Use a RBF Kernel to produce nonlinear decision boundary
  - Experimented with adding a new input – # of news stories
Example 1 – Rao and Hong 2010

Results – HMM

<table>
<thead>
<tr>
<th>ticker</th>
<th>accuracy prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>pot</td>
<td>0.5172</td>
</tr>
<tr>
<td>aapl</td>
<td>0.5172</td>
</tr>
<tr>
<td>gs</td>
<td>0.5517</td>
</tr>
<tr>
<td>mos</td>
<td>0.5517</td>
</tr>
<tr>
<td>ibm</td>
<td>0.5172</td>
</tr>
<tr>
<td>msft</td>
<td>0.7586</td>
</tr>
<tr>
<td>gg</td>
<td>0.5862</td>
</tr>
<tr>
<td>bac</td>
<td>0.5862</td>
</tr>
<tr>
<td>goog</td>
<td>0.5517</td>
</tr>
<tr>
<td>c</td>
<td>0.5172</td>
</tr>
<tr>
<td>sp_500</td>
<td>0.5172</td>
</tr>
</tbody>
</table>

Table 1: Results from HMM using K-means learner

Results – SVM

<table>
<thead>
<tr>
<th>ticker</th>
<th>No News</th>
<th>News</th>
</tr>
</thead>
<tbody>
<tr>
<td>pot</td>
<td>0.5667</td>
<td>0.7</td>
</tr>
<tr>
<td>aapl</td>
<td>0.5333</td>
<td>0.6667</td>
</tr>
<tr>
<td>gs</td>
<td>0.6</td>
<td>0.5333</td>
</tr>
<tr>
<td>mos</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>ibm</td>
<td>0.5667</td>
<td>0.5333</td>
</tr>
<tr>
<td>msft</td>
<td>0.8</td>
<td>0.6333</td>
</tr>
<tr>
<td>gg</td>
<td>0.5333</td>
<td>0.6333</td>
</tr>
<tr>
<td>bac</td>
<td>0.5667</td>
<td>0.6333</td>
</tr>
<tr>
<td>goog</td>
<td>0.5333</td>
<td>0.5667</td>
</tr>
<tr>
<td>c</td>
<td>0.5</td>
<td>0.5667</td>
</tr>
<tr>
<td>sp_500</td>
<td>0.5333</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Table 10: Results from SVM using both no news and with news with (RBF kernel 2)
Test data: Last 30 days. Without retraining
Example 2 – Random forests

• Two examples:

• Both teams use Random Forests (classification trees) to build classifiers
Example 2 – Random forests

• Lauretto et al. – methodology
  – Daily equities data (OHLCV)
  – Inputs are SMAs, EMAs, ROC, Stoch, MACD, RSI
  – Classes are: {Buy-Sell, Sell-Buy, Do-nothing}

• Lauretto et al. – results*:
  – 80% “successful devised operations”
  – 70% “seized opportunities”
  – Average return per operation: 4%

* WTF???
Theofilatos et al. – methodology

- Predicting one day ahead EUR/USD
- Only use autoregressive inputs, i.e. up to 10 days lagged data used as inputs
- Compared a bunch of ML algos, including SVM (with RBF kernel), RFs, NN, Naïve Bayes

Theofilatos et al. – results:

<table>
<thead>
<tr>
<th></th>
<th>Artificial Neural Networks (MLP)</th>
<th>SVM</th>
<th>Random Forests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Ratio (excluded costs)</td>
<td>0.22</td>
<td>0.43</td>
<td>0.72</td>
</tr>
<tr>
<td>Annualized Volatility (excluded costs)</td>
<td>11.46%</td>
<td>11.46%</td>
<td>11.45%</td>
</tr>
<tr>
<td>Annualized Return (excluding costs)</td>
<td>2.51%</td>
<td>4.90%</td>
<td>8.29%</td>
</tr>
<tr>
<td>Maximum Drawdown (excluding costs)</td>
<td>-18.89%</td>
<td>-14.74%</td>
<td>-9.94%</td>
</tr>
<tr>
<td>Correct Directional Prediction</td>
<td>50.12%</td>
<td>52.65%</td>
<td>53.50%</td>
</tr>
<tr>
<td>Transaction costs</td>
<td>0.92</td>
<td>0.92%</td>
<td>1.01%</td>
</tr>
<tr>
<td>Annualized Return (including costs)</td>
<td>1.59%</td>
<td>3.98%</td>
<td>7.28</td>
</tr>
</tbody>
</table>
Problems and gotchas

- Are YOU smarter than a machine? Don’t forget everything you already know...
- If you don’t believe it, it’s probably not real!

- How many datasets should you use?
  - 3! Training, validation, out of sample testing
- Input data needs pre-processing and scaling
- Over-fitting – regularisation, out of sample
- Computation speed, (online) (re-)training
- No peeping!
- GI/GO → SNAFU
Resources

• **Software**
  – Quantopian
  – Lucena
  – Azure
  – MATLAB
  – Python, R
  – WEKA
  – RapidMiner
  – JavaML
  – LibSVM

• **Online learning**
  – Coursera
  – Udacity
  – Nando de Freitas
    YouTube channel
  – Quantopian
  – Lucena

  – (wikipedia)
What next?

• DO NOT DESTROY MANKIND!
Questions?

Dr. Aly Kassam
aly@quantsupport.com