# Why is Machine Learning in finance so hard?

Hardik Patel
Machine Learning Engineer, qplum

www.qplum.co

Disclosures: qplum LLC is a registered investment adviser. Information presented is for educational purposes only and does not intend to make an offer or solicitation for the sale or purchase of any specific securities, investments, or investment strategies. Investments involve risk and are never guaranteed. Be sure to first consult with a qualified financial adviser and/or tax professional before implementing any strategy discussed herein. Past performance is not indicative of future performance.

# How does a typical quant pipeline look like?

- Start with a bunch of alphas
- Go long on the top 10
- Go short on the bottom 10

# But, how does Machine Learning come into the picture?

# 1. Changing Data Distributions

WOW, THOSE ALPHAS LOOK REALLY GOOD. LET ME ADD THEM TO MY PORTFOLIO.
1 YEAR LATER
I DON'T KNOW WHAT HAPPENED. THOSE ALPHAS WERE SUPPOSED TO MAKE ME RICH.

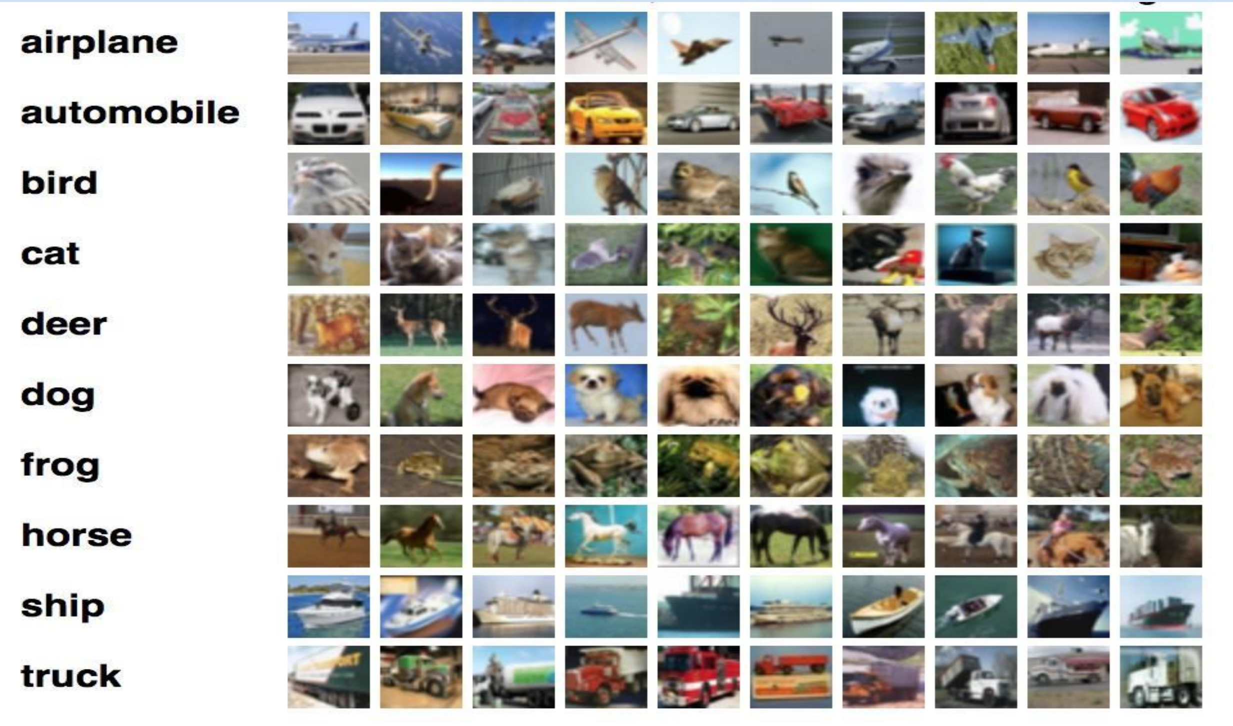
Source: www.hardikp.com

# Train/test data distributions are different in finance

- Primary assumption for almost all machine learning tasks is that the training dataset represents the kind of data you would see in the production (or test) dataset.
- But, this assumption doesn’t hold in financial time series datasets.
- It’s quite likely that the patterns of the past may never get manifested in future.
- Causal factors for tomorrow’s market are likely to be very different than what are training dataset contains.

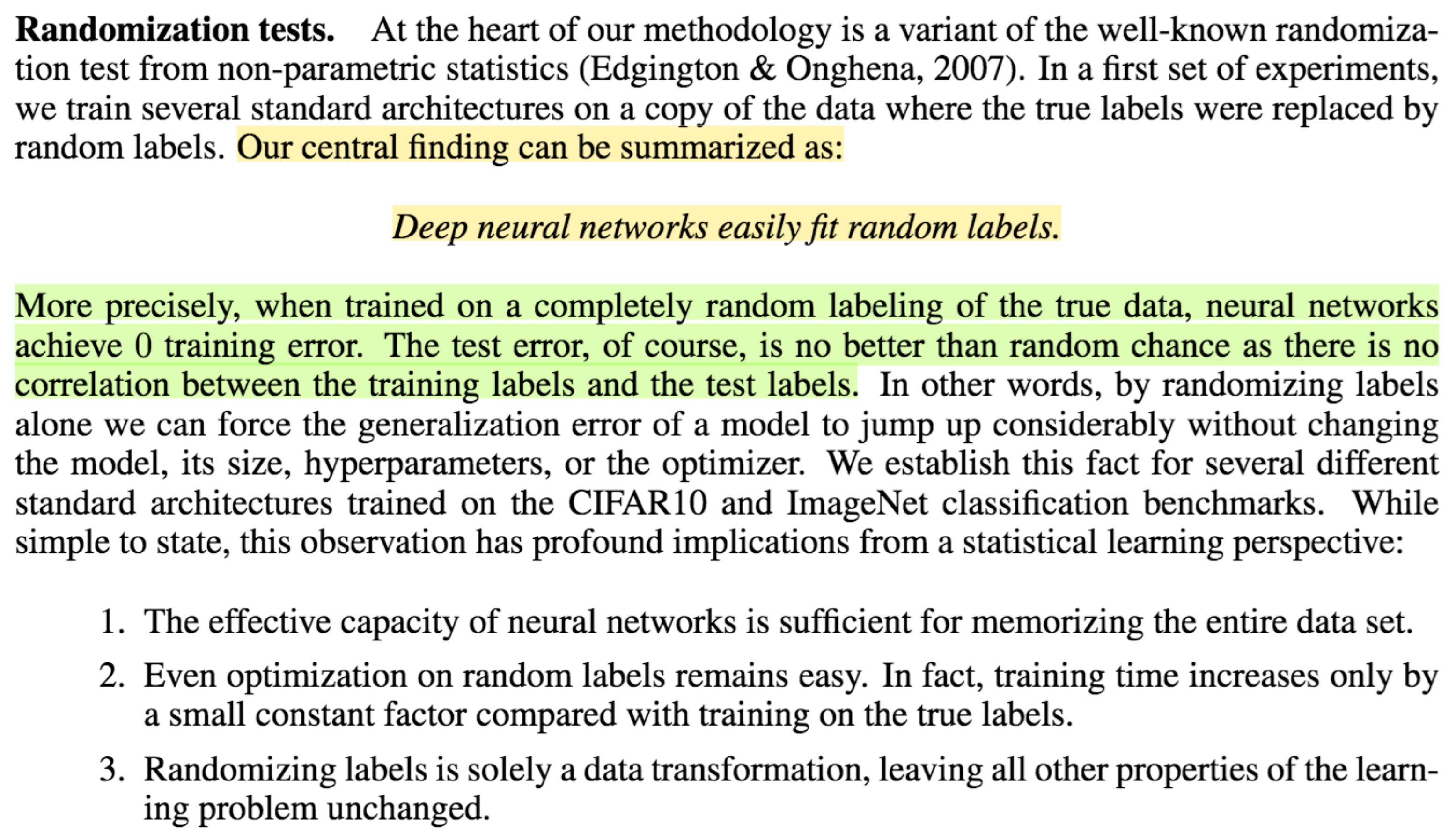
# Comparing against Computer Vision datasets

- CIFAR10 is an image classification dataset
- Dog images in train dataset would have similar distribution with dog images in the test dataset.



Source: Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, The CIFAR-10 dataset, www.cs.toronto.edu/~kriz/cifar.html

# "Understanding deep learning requires rethinking generalization" paper from Google Brain



**Randomization tests.** At the heart of our methodology is a variant of the well-known randomization test from non-parametric statistics (Edgington & Onghena, 2007). In a first set of experiments, we train several standard architectures on a copy of the data where the true labels were replaced by random labels. Our central finding can be summarized as:

*Deep neural networks easily fit random labels.*

More precisely, when trained on a completely random labeling of the true data, neural networks achieve 0 training error. The test error, of course, is no better than random chance as there is no correlation between the training labels and the test labels. In other words, by randomizing labels alone we can force the generalization error of a model to jump up considerably without changing the model, its size, hyperparameters, or the optimizer. We establish this fact for several different standard architectures trained on the CIFAR10 and ImageNet classification benchmarks. While simple to state, this observation has profound implications from a statistical learning perspective:

1. The effective capacity of neural networks is sufficient for memorizing the entire data set.
2. Even optimization on random labels remains easy. In fact, training time increases only by a small constant factor compared with training on the true labels.
3. Randomizing labels is solely a data transformation, leaving all other properties of the learning problem unchanged.

# Solving the Data Distribution problem

- **Walk-forward Optimization**
- Instead of generating a model, develop a system of generating models - each representing different time periods.
- Evaluate the system as a whole rather than an individual model.
- **Online learning**

# 2. Low Predictive Power

# Low Predictive Power of Financial Datasets

- Low accuracies compared to other domains. Getting to non-random accuracy numbers in classification problems is often challenging.
- “Partial Information” nature of the problem puts an implicit limit on the extent of what can be predicted.

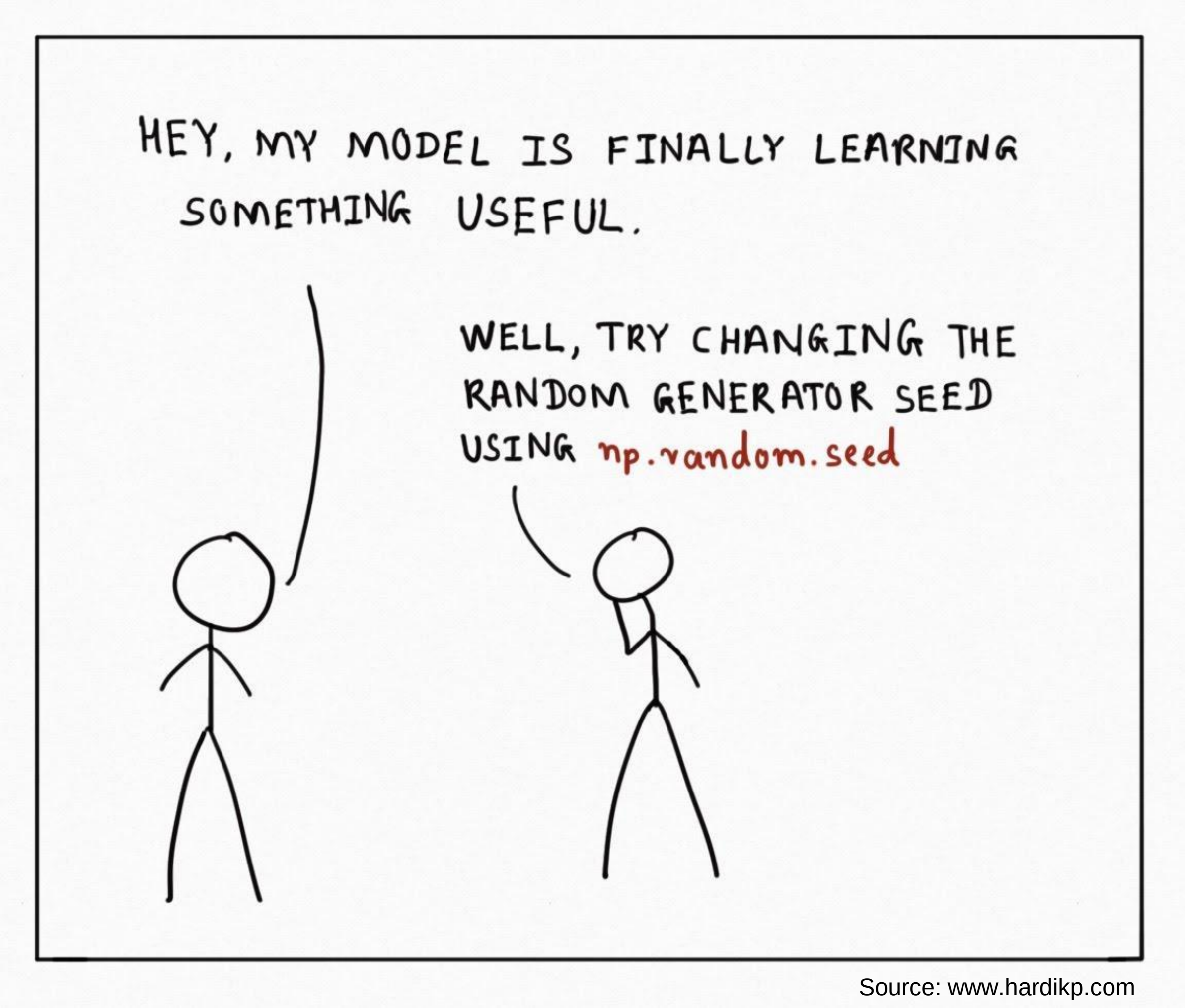
# Solving the Low Predictive Power issue

- While general prediction problems like return prediction are very hard, niche problems are easier to deal with.
- Focus on detecting regimes.

# 3. Low Signal to Noise Ratio

# Low Signal to Noise Ratio

If you see a pattern in the dataset, it's more likely to be noise than signal.


HEY, MY MODEL IS FINALLY LEARNING SOMETHING USEFUL.
WELL, TRY CHANGING THE RANDOM GENERATOR SEED USING np.random.seed

Source: www.hardikp.com

# Solving the Low Signal Problem

- Focus on **Model Interpretability**.
- There is almost always a chance of overfitting if you aren't able to answer interpretability questions.
- The challenge is to find the source of the model signal.

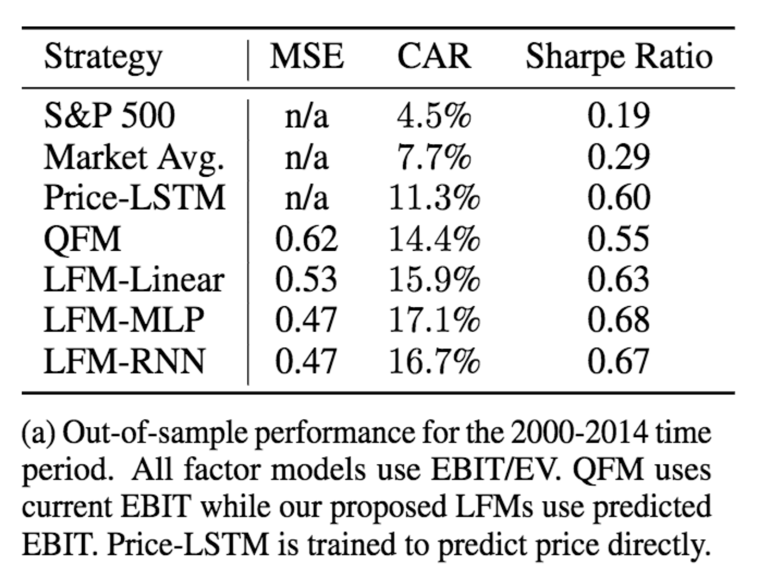
# 4. **Model accuracy** is not correlated with **utility function**

# Better Accuracy != Better Portfolio

- Better portfolio performance is the utility function.
- Model accuracy improvement might not lead to better portfolio returns.
- There is more portfolio management than predicting returns - trading costs, execution effects and other practical constraints are responsible for this divergence.

# Better Accuracy != Better Portfolio

- Evidence for this can be found at all scales in financial markets: from high-frequency trading to long-term investing.



| Strategy | MSE | CAR | Sharpe Ratio |
|---|---|---|---|
| S&P 500 | n/a | 4.5% | 0.19 |
| Market Avg. | n/a | 7.7% | 0.29 |
| Price-LSTM | n/a | 11.3% | 0.60 |
| QFM | 0.62 | 14.4% | 0.55 |
| LFM-Linear | 0.53 | 15.9% | 0.63 |
| LFM-MLP | 0.47 | 17.1% | 0.68 |
| LFM-RNN | 0.47 | 16.7% | 0.67 |

(a) Out-of-sample performance for the 2000-2014 time period. All factor models use EBIT/EV. QFM uses current EBIT while our proposed LFMs use predicted EBIT. Price-LSTM is trained to predict price directly.

Source: Alberg, John & Lipton, Zachary C. Improving Factor-Based Quantitative Investing by Forecasting Company Fundamentals

# Solution(1)

Directly optimize the utility function

1. Write down the utility function in terms of the raw data.
2. Use quadratic solvers or gradient descent to find the weights.

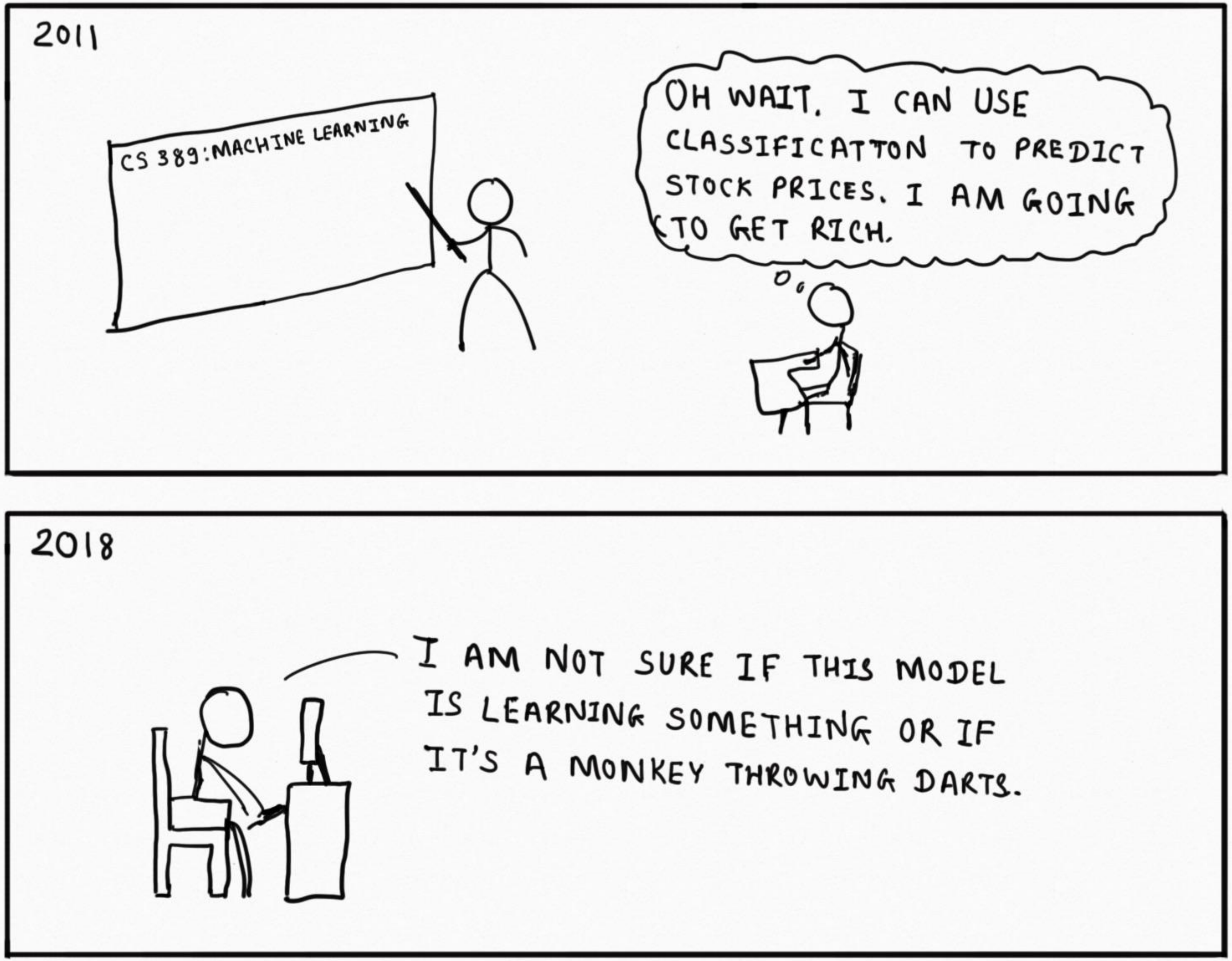
Notes:

- Describing each aspect of the utility function can be hard. This often leads to simplification of the utility function.
- The optimization process is often unstable and can be unreliable.

# Solution(2)

Reinforcement Learning is another way to directly optimize the utility function.

- Doesn't suffer from oversimplification issues.
- RL state can be made sufficiently complex.
- But the state doesn't have enough information to guide the agent in the right direction.
- Lack of enough predictive information often causes the agents to wander in random directions.


2011
CS 389: MACHINE LEARNING
OH WAIT. I CAN USE CLASSIFICATTON TO PREDICT STOCK PRICES. I AM GOING TO GET RICH.
2018
I AM NOT SURE IF THIS MODEL IS LEARNING SOMETHING OR IF IT'S A MONKEY THROWING DARTS.

Source: www.hardikp.com

Financial time-series is a partial information game (POMDP) that’s really hard even for humans - we shouldn’t expect machines and algorithms to suddenly surpass human ability there.

# Two recent developments in NN for finance

1. Use of Attention Mechanism in time series models. [https://arxiv.org/abs/1704.02971, http://homepages.inf.ed.ac.uk/scohen/acl18stock.pdf]
2. Using embeddings to represent and combine multiple sources of information.

# Questions?

**contact@qplum.co**

Disclosures: qplum LLC is a registered investment adviser. Information presented is for educational purposes only and does not intend to make an offer or solicitation for the sale or purchase of any specific securities, investments, or investment strategies. Investments involve risk and are never guaranteed. Be sure to first consult with a qualified financial adviser and/or tax professional before implementing any strategy discussed herein. Past performance is not indicative of future performance.