ANNUAL MEETING

November 9-12, 2020 • Online Event

AI in Actuarial Science

The State of the Art

Ronald Richman

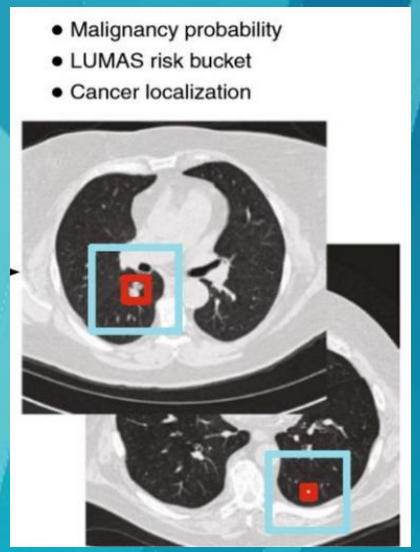
Associate Director - QED *Actuaries & Consultants*

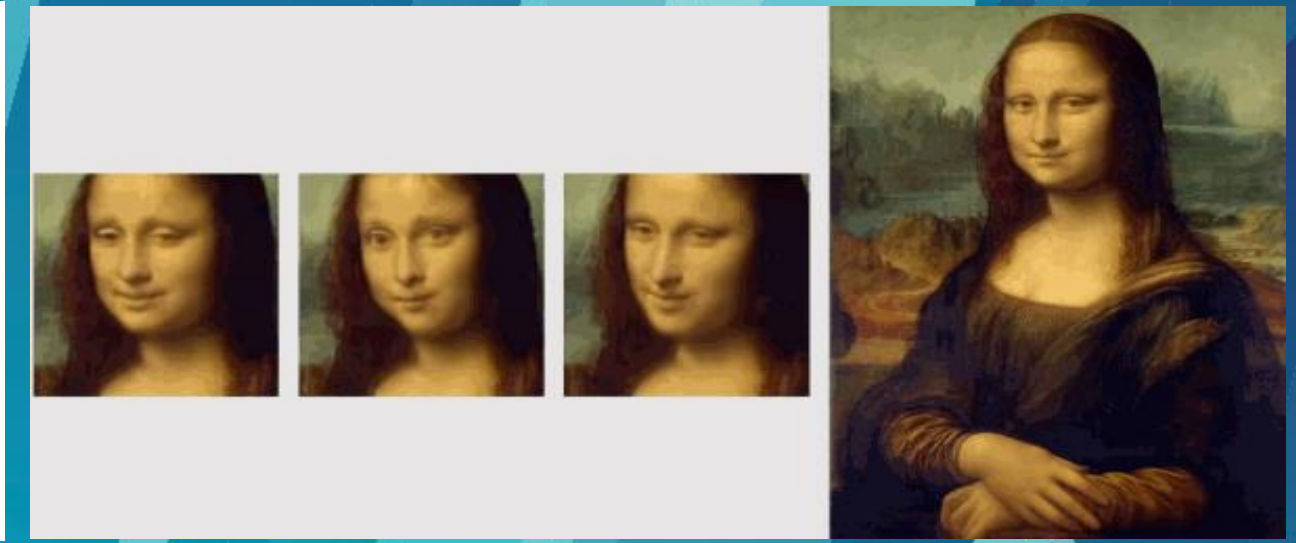
November 2020

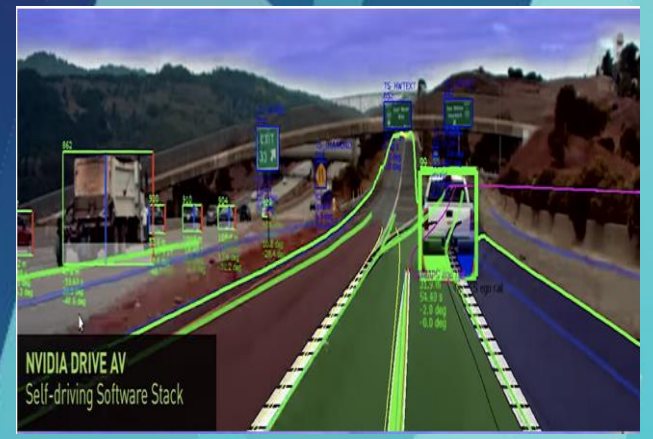
Goals of the talk

- What machine learning implies for actuarial science
- Understand the problems solved by deep learning
- Discuss the tools of the trade
- Discuss recent successes of deep learning in actuarial science
- Discuss emerging challenges and solutions

Deep Learning in the Wild







An exciting part of the world of finance is insurance

I think we all know that the insurance industry is exciting. I see it everywhere - the airlines, the cars, most all the businesses in the world. The insurance industry can really drive the economic innovation.

But one area of insurance that I really want to see develop more is financial advice. It might be a private sector service but insurance companies are not really there anymore. In general we are not allowed to talk to clients about financial solutions - we need to find a new solution. It would be fun to see what a private sector insurance can deliver.

- Man from www.thispersondoesnotexist.com/
- Mona Lisa from Samsung AI team
- Text from https://talktotransformer.com/
- Self- driving from NVIDIA blog
- Cancer detection from Nature Medicine

Actuarial Data Science

- **Traditionally, actuaries responsible for statistical and financial management of insurers**
  Today, actuaries, data scientists, machine learning engineers and others work alongside each other
- **Actuaries focused on specialized areas such as pricing/reserving**
  Many applications of ML/DL within insurance but outside of traditional areas
- **Actuarial science merges statistics, finance, demography and risk management**
  Currently evolving to include ML/DL
- **According to Data Science working group of the SAA:**
  Actuary of the fifth kind - job description is expanded further to include statistical and computer-science
  Actuarial data science - subset of mathematics/statistics, computer science and actuarial knowledge
- **Focus of talk: ML/DL within Actuarial Data Science – applications of machine learning and deep learning to traditional problems dealt with by actuaries**

Definitions and Diagram from Data Science working group of the Swiss Association of Actuaries (SAA)

Agenda

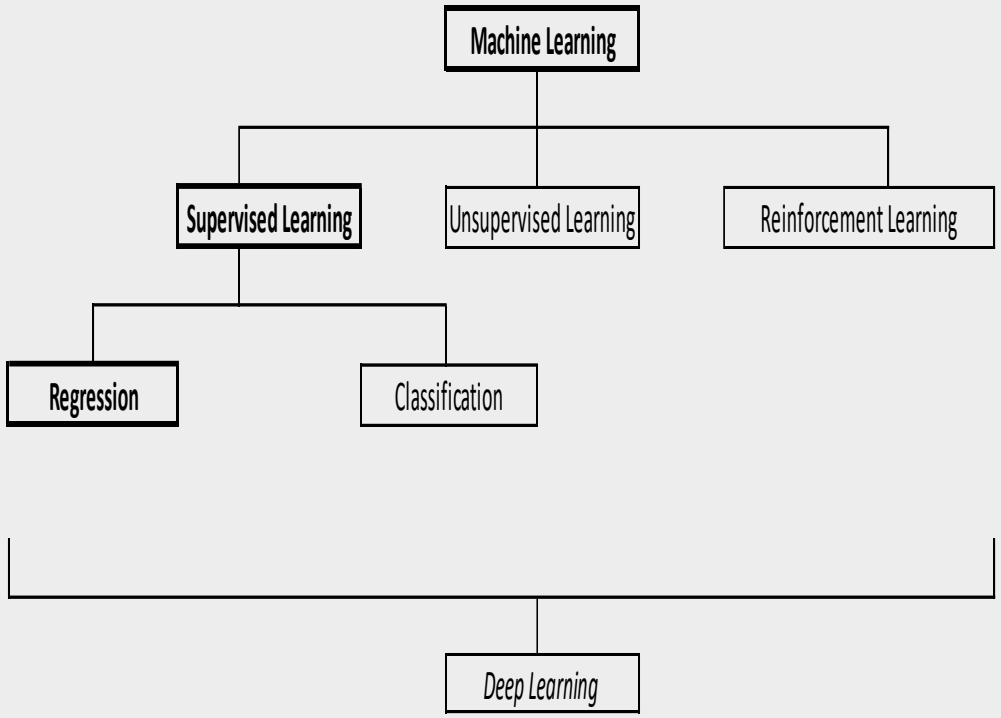
- From Machine Learning to Deep Learning
- Tools of the Trade
- Selected Applications
- Stability of Results
- Discrimination Free Pricing

Machine Learning

- **Machine Learning "the study of algorithms that allow computer programs to automatically improve through experience" (Mitchell 1997)**
- **Machine learning is an approach to the field of Artificial Intelligence**

  Systems trained to recognize patterns within data to acquire knowledge (Goodfellow, Bengio and Courville 2016).
- **Earlier attempts to build AI systems = hard code knowledge into knowledge bases … but doesn't work for highly complex tasks e.g. image recognition, scene understanding and inferring semantic concepts (Bengio 2009)**
- **ML Paradigm – feed data to the machine and let it figure out what is important from the data!**

  Deep Learning represents a specific approach to ML.

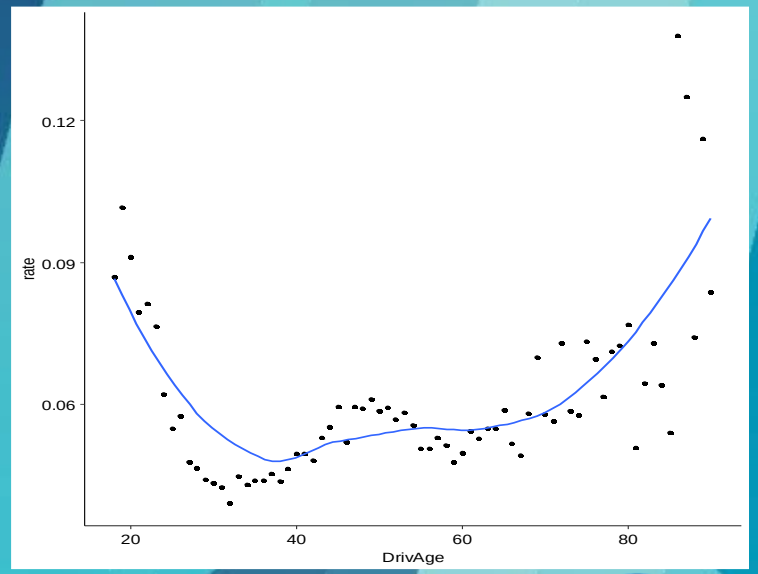


Supervised Learning

- **Supervised learning = application of machine learning to datasets that contain features and outputs with the goal of predicting the outputs from the features (Friedman, Hastie and Tibshirani 2009).**
- **Feature engineering - Suppose we realize that Claims depends on Age^2 => enlarge feature space by adding Age^2 to data. Other options – add interactions/basis functions e.g. splines**

y (*outputs*) **X (*features*)**

```
> freMTPL2freq
          IDpol ClaimNb     Exposure Area VehPower VehAge DrivAge BonusMalus VehBrand  VehGas Density Region DrivAge_2
     1:       1       1 0.100000000    D        5      0      55          50       B12 Regular    1217    R82       3025
     2:       3       1 0.770000000    D        5      0      55          50       B12 Regular    1217    R82       3025
     3:       5       1 0.750000000    B        6      2      52          50       B12  Diesel      54    R22       2704
     4:      10       1 0.090000000    B        7      0      46          50       B12  Diesel      76    R72       2116
     5:      11       1 0.840000000    B        7      0      46          50       B12  Diesel      76    R72       2116
    ---
678009: 6114326       0 0.002739726    E        4      0      54          50       B12 Regular    3317    R93       2916
678010: 6114327       0 0.002739726    E        4      0      41          95       B12 Regular    9850    R11       1681
678011: 6114328       0 0.002739726    D        6      2      45          50       B12  Diesel    1323    R82       2025
678012: 6114329       0 0.002739726    B        4      0      60          50       B12 Regular      95    R26       3600
678013: 6114330       0 0.002739726    B        7      6      29          54       B12  Diesel      65    R72        841
>
```



Goal: Explaining or Predicting?

- **Which of the following are an ML technique?**
  Linear regression and friends (GLM/GLMM)
  Generalized Additive model (GAM)
  Exponential Smoothing
  Chain-Ladder and Bornhuetter-Ferguson

- **It depends on the goal:**
  Are we building a causal understanding of the world (inferences from unbiased coefficients)?
  Or do we want to make predictions (bias-variance trade-off)?

- **Distinction between tasks of predicting and explaining, see Shmueli (2010). Focus on predictive performance leads to:**

  Building algorithms to predict responses instead of specifying a stochastic data generating model (Breiman 2001)…
  … favouring models with good predictive performance at expense of interpretability.
  Accepting bias in model coefficients if this is expected to reduce the overall prediction error.
  Quantifying predictive error (i.e. out-of-sample error)

- **ML relies on a different approach to building, parameterizing and testing statistical models, based on statistical learning theory, and focuses on predictive accuracy.**

Recipe for Actuarial Data Science

- **Actuarial problems are often supervised regressions =>**
- **If an actuarial problem can be expressed as a regression, then machine and deep learning can be applied.**
- **Obvious areas of application:**

  P&C pricing
  IBNR reserving
  Experience analysis
  Mortality modelling
  Lite valuation models

- **But don't forget about unsupervised learning either!**

Actuarial Modelling

- **Actuarial modelling tasks vary between:**

  Empirically/data driven
  - NL pricing
  - Approximation of nested Monte Carlo
  - Portfolio specific mortality

  Model Driven
  - IBNR reserving (Chain-Ladder)
  - Life experience analysis (AvE)
  - Capital modelling (Log-normal/Clayton copula)
  - Mortality forecasting (Lee-Carter)

Human input

Feature engineering

Model Specification

- **Feature engineering = data driven approach to enlarging a feature space using human ingenuity and expert domain knowledge**
  - Apply various techniques to the raw input data – PCA/splines
  - Enlarge features with other related data (economic/demographic)

- **Model specification = model driven approach where define structure and form of model (often statistical) and then find the data that can be used to fit it**

Issues with Traditional Approach

- **In many domains, including actuarial science, traditional approach to designing machine learning systems relies on human input for feature engineering or model specification.**

- **Three arguments against traditional approach:**

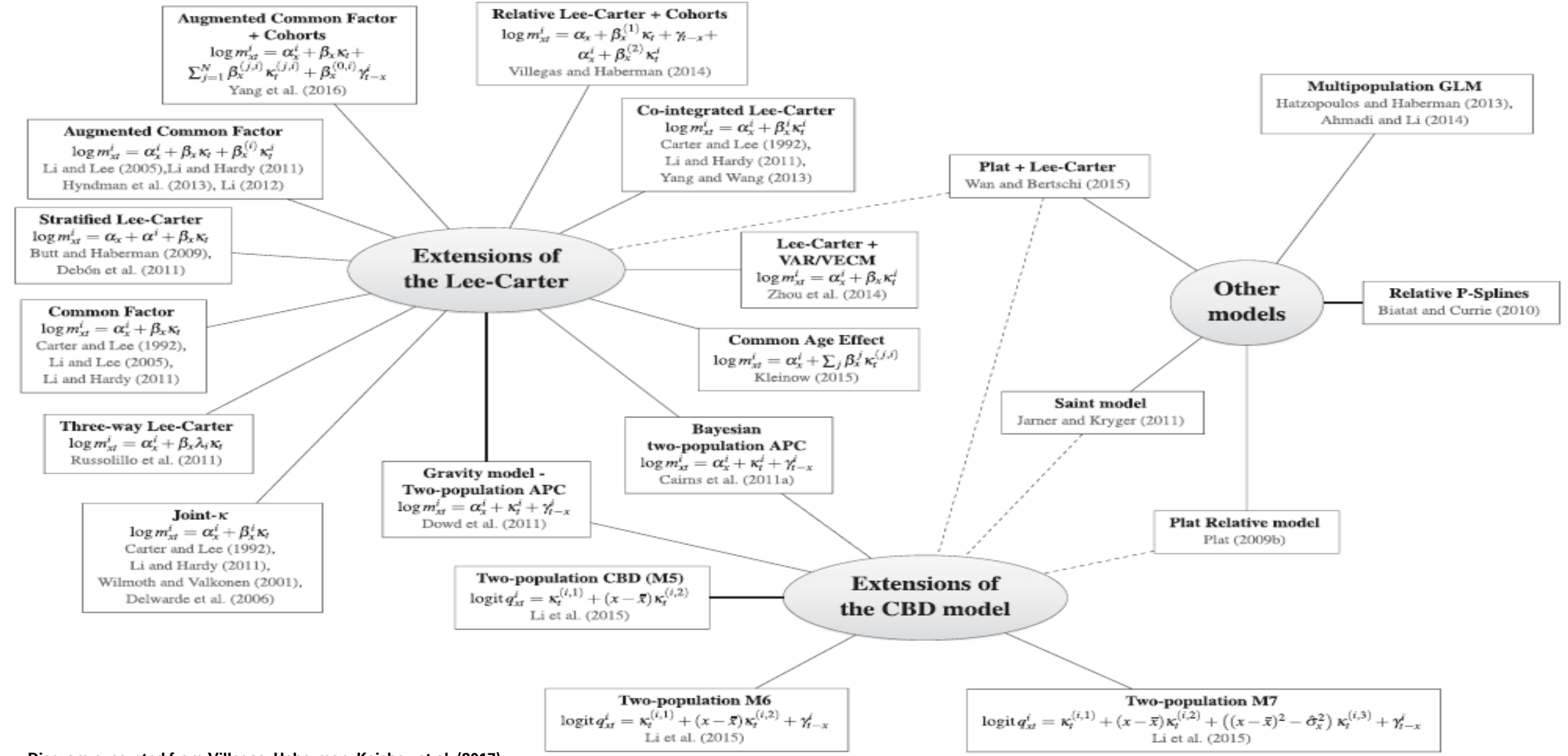
  Complexity – which are the relevant features to extract/what is the correct model specification? Difficult with very high dimensional, unstructured data such as images or text. (Bengio 2009; Goodfellow, Bengio and Courville 2016)

  Expert knowledge – requires suitable prior knowledge, which can take decades to build (and might not be transferable to a new domain) (LeCun, Bengio and Hinton 2015)

  Effort – designing features is time consuming/tedious => limits scope and applicability (Bengio, Courville and Vincent 2013; Goodfellow, Bengio and Courville 2016)

- **Within actuarial modelling, complexity is not only due to unstructured data. Many difficult problems of model specification arise when performing actuarial tasks at a large scale:**
  Multi-LoB IBNR reserving
  Mortality forecasting for multiple populations

Complexity: Multi-population Mortality Modelling



- **Diagram excerpted from Villegas, Haberman, Kaishev et al. (2017)**

Representation Learning

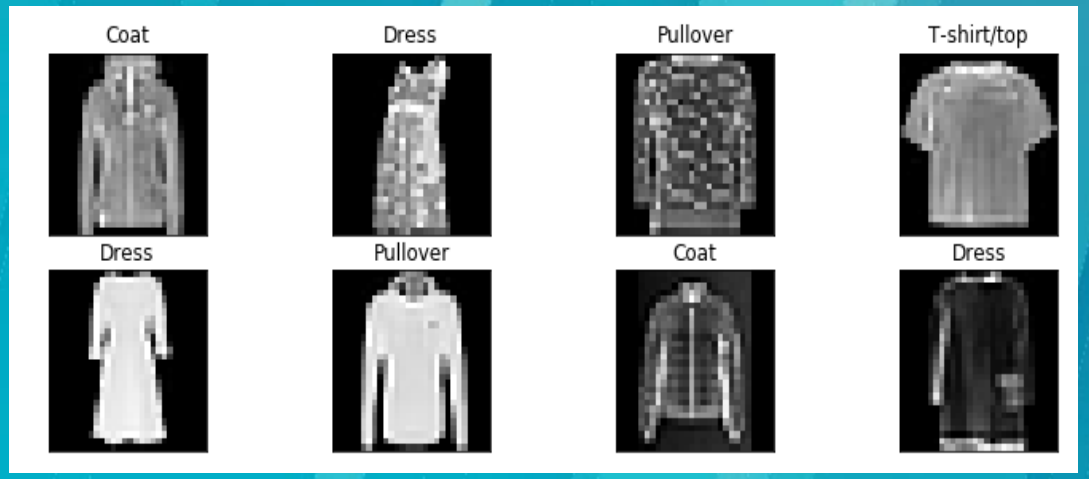
- **Representation Learning = ML technique where algorithms automatically design features that are optimal (in some sense) for a particular task**
- **Traditional examples are PCA (unsupervised) and PLS (supervised):**

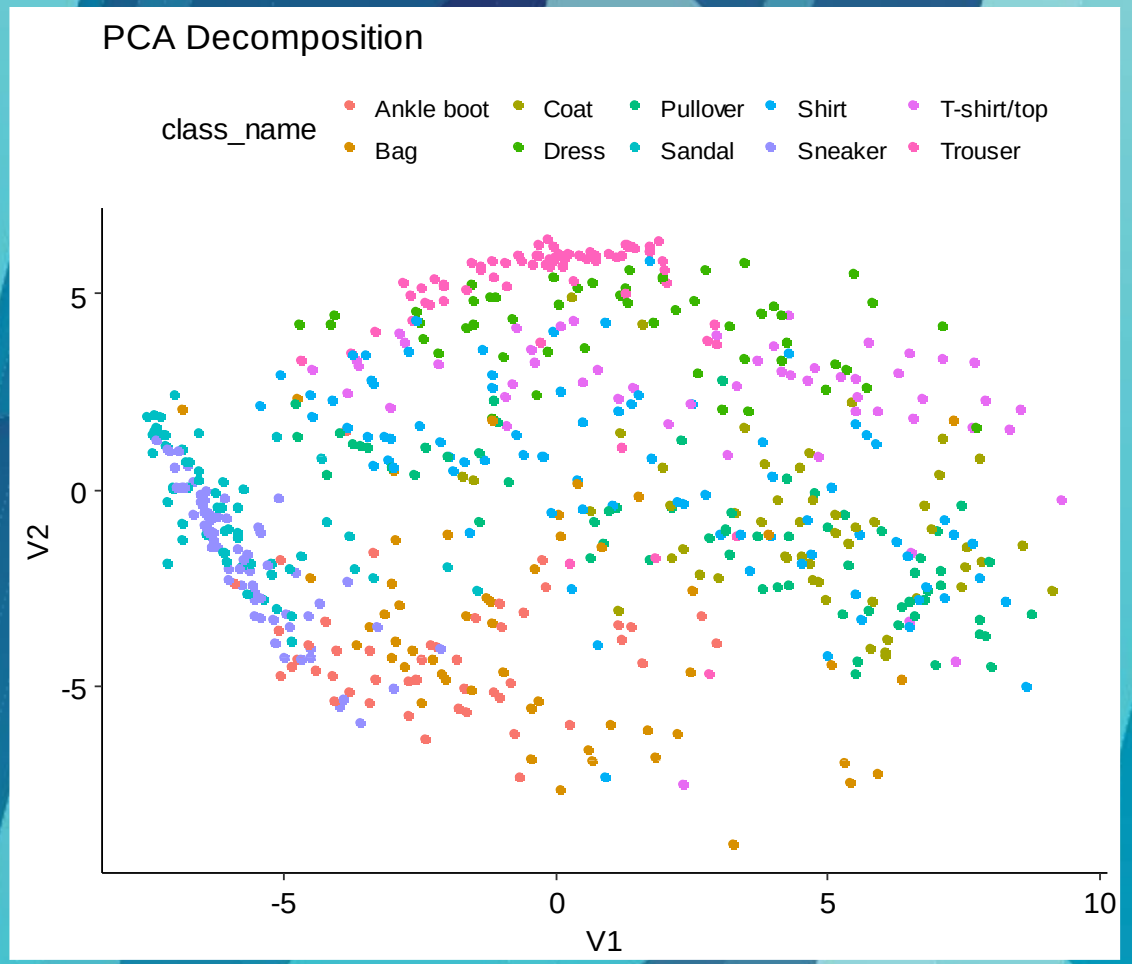
  PCA produces features that summarize directions of greatest variance in feature matrix

  PLS produces features that maximize covariance with response variable (Stone and Brooks 1990)
- **Feature space then comprised of learned features which can be fed into ML/DL model**
- **BUT: Simple/naive RL approaches often fail when applied to high dimensional data**

Example: Fashion-MNIST (1)

- **Inspired by Hinton and Salakhutdinov (2006)**
- **Fashion-MNIST –70 000 images from Zolando of common items of clothing**
- **Grayscale images of 28x28 pixels**
- **Classify the type of clothing**
- **Applied PCA directly to the images - results do not show much differentiation between classes**





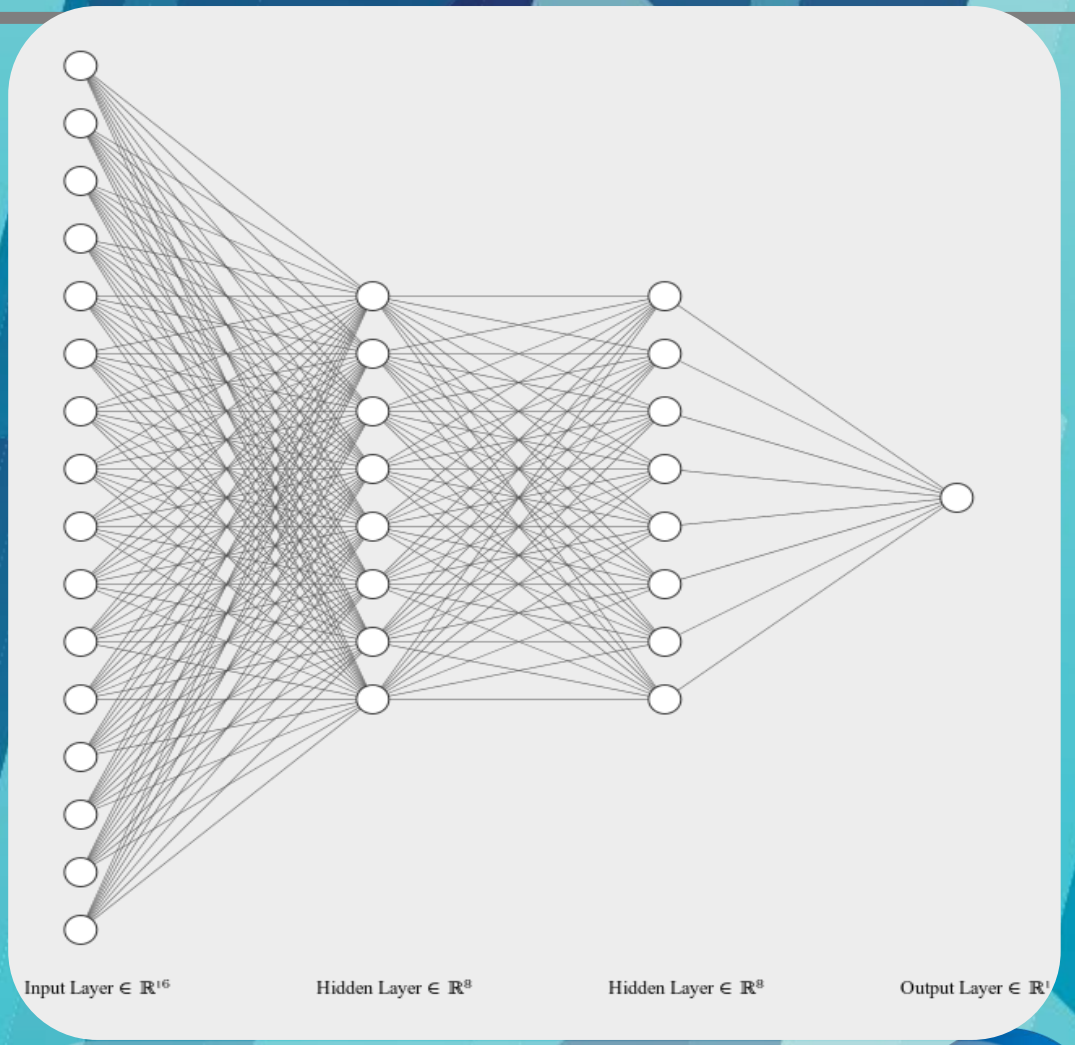
Deep Learning

- **Deep Learning = representation learning technique that automatically constructs hierarchies of complex features to represent abstract concepts**
  Features in lower layers composed of simpler features constructed at higher layers => complex concepts can be represented automatically

- **Typical example of deep learning is feed-forward neural networks, which are multi-layered machine learning models, where each layer learns a new representation of the features.**

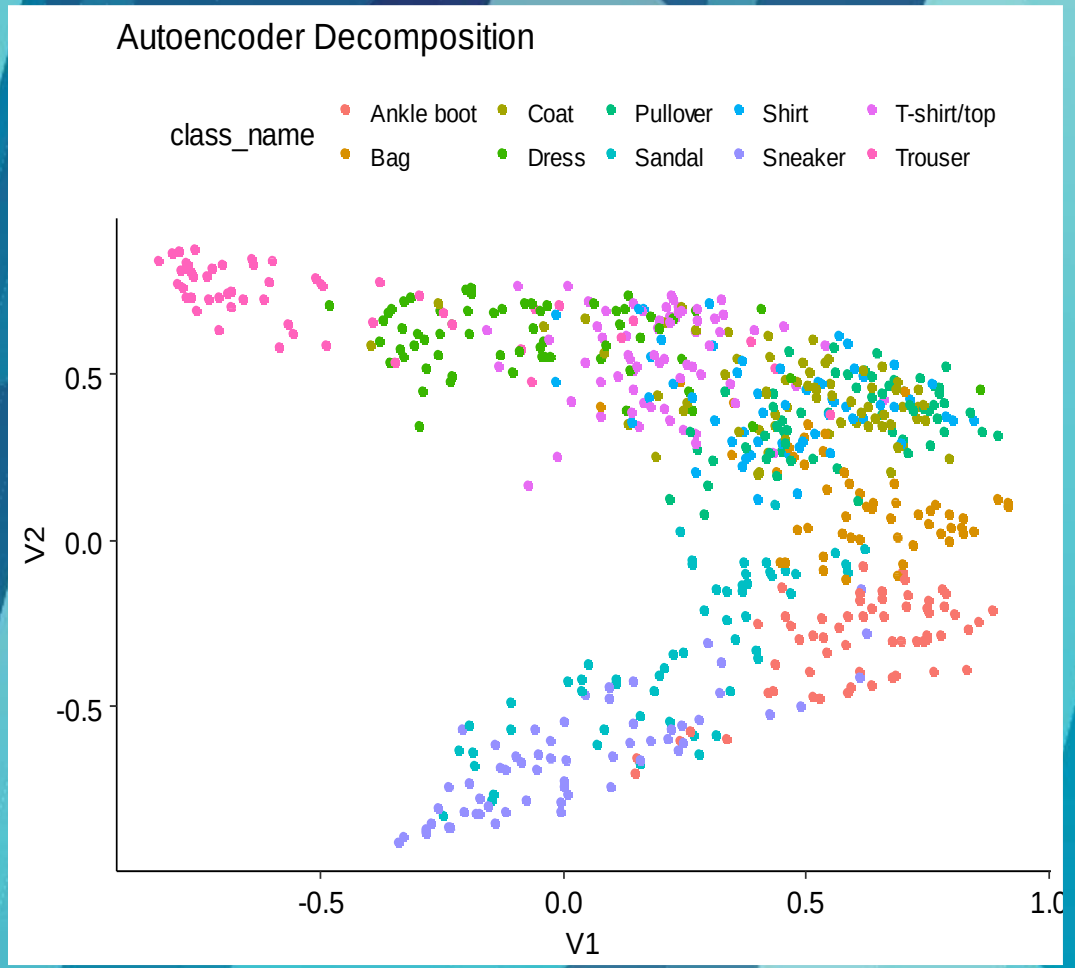
- **The principle: Provide raw data to the network and let it figure out what and how to learn.**

- **Desiderata for AI by Bengio (2009): "Ability to learn with little human input the low-level, intermediate, and high-level abstractions that would be useful to represent the kind of complex functions needed for AI tasks."**



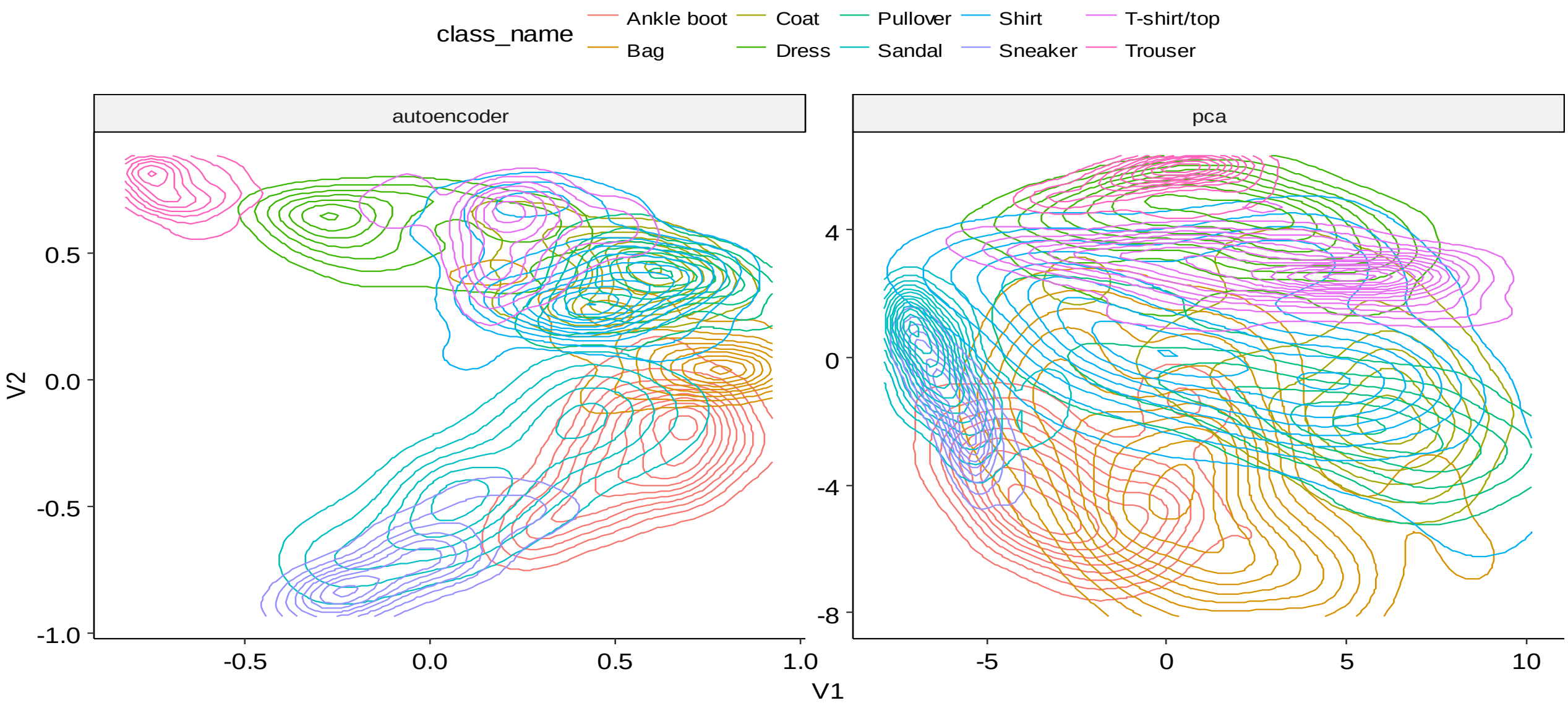
Example: Fashion-MNIST (2)

- **Applied a deep autoencoder to the same data (trained in unsupervised manner)**
  Type of non-linear PCA
- **Differences between some classes much more clearly emphasized**
- **Deep representation of data automatically captures meaningful differences between the images without (much) human input**
- **Automated feature/model specification**
- **Aside – feature captured in unsupervised learning might be useful for supervised learning too.**
- **Goodfellow, Bengio and Courville (2016) : "basic idea is features useful for the unsupervised task also be useful for the supervised learning task"**



Fashion-MNIST – Density Plot

Density in learned space



Deep Learning for Actuarial Modelling

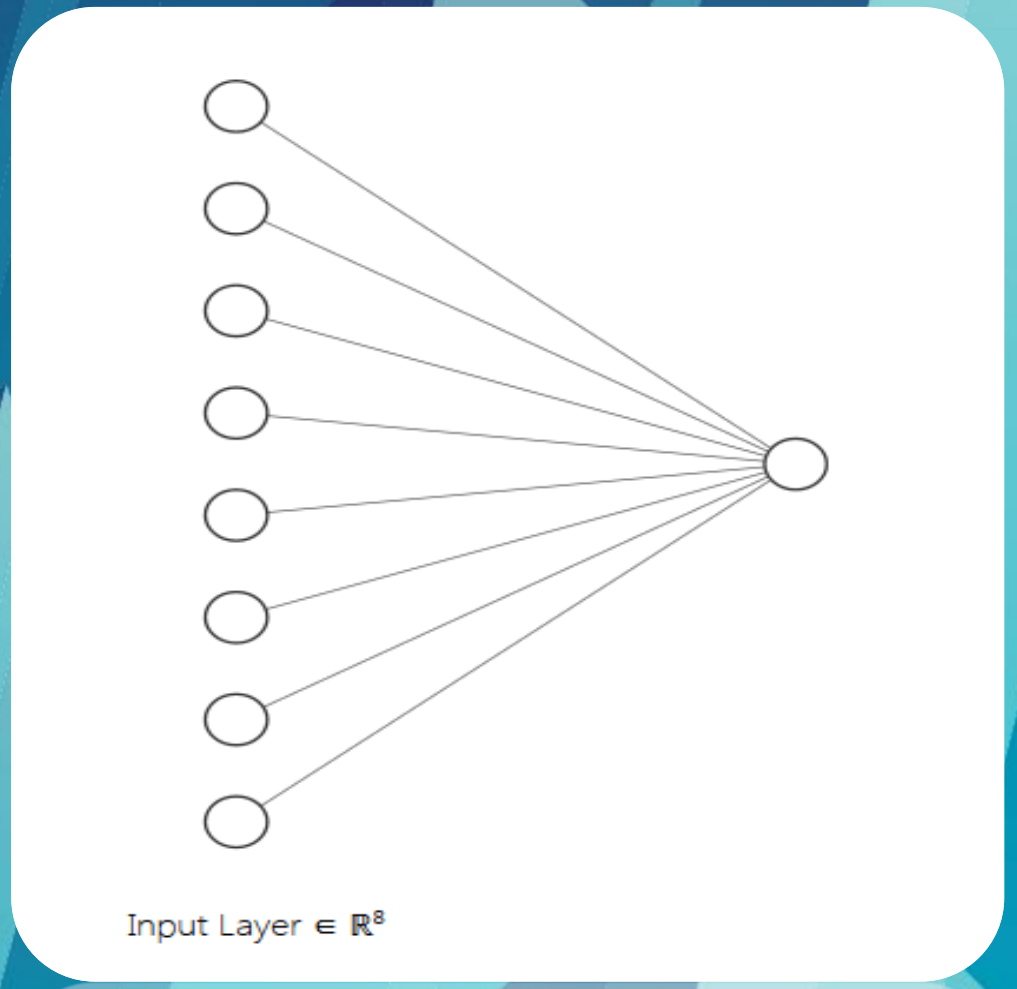
- Actuarial tasks vary between Empirically/data driven and Model Driven
- Both approaches traditionally rely on manual specification of features or models
- Deep learning offers an empirical solution to both types of modelling task – feed data into a suitably deep neural network => learn an optimal representation of input data for task
- Exchange of model specification for a new task => architecture specification
- Opportunity – improve best estimate modelling
- Deep learning comes at a (potential) cost – relying on a learned representation means less understanding of models, to some extent

Agenda

- From Machine Learning to Deep Learning
- Tools of the Trade
- Selected Applications
- Stability of Results
- Discrimination Free Pricing

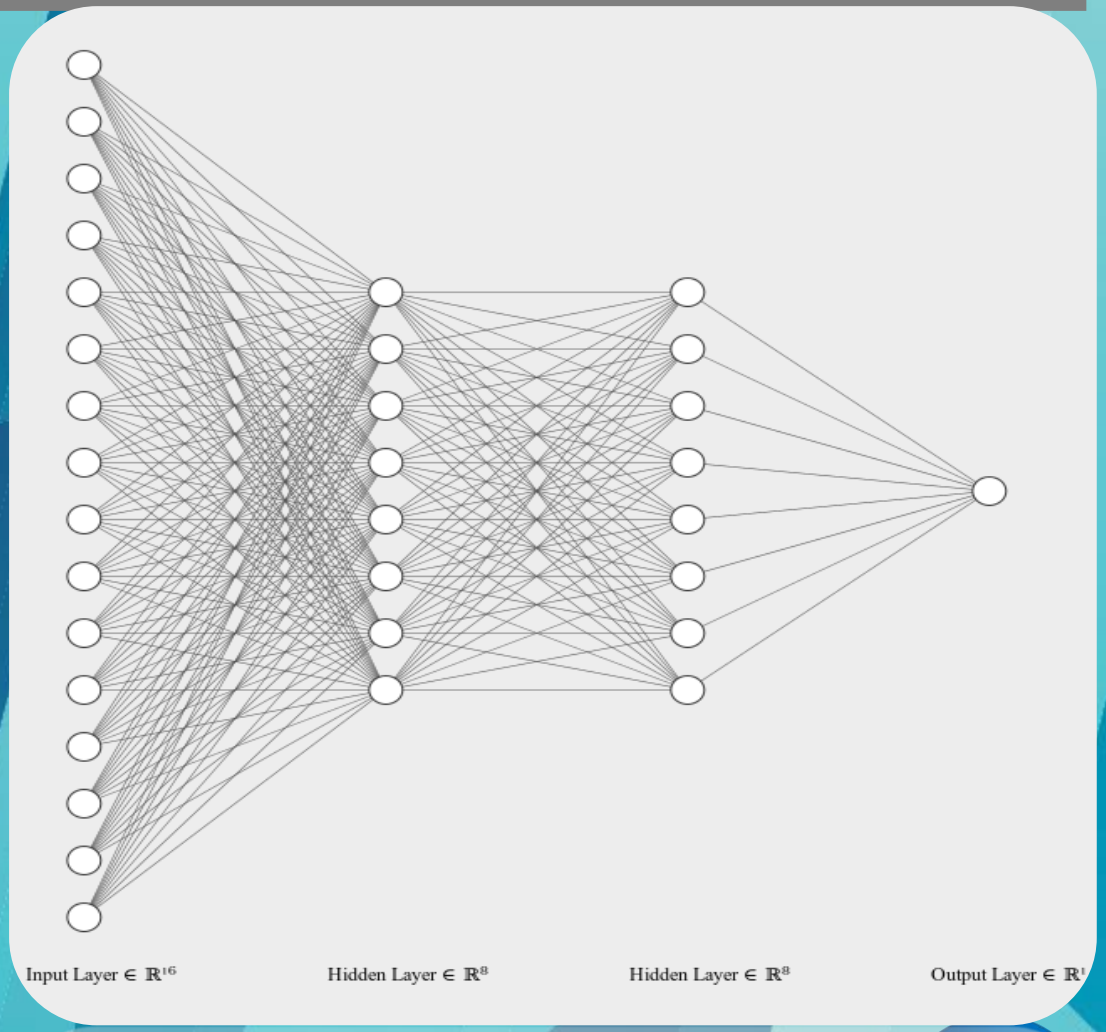
Single Layer NN = Linear Regression

- **Single layer neural network**
  Circles = variables
  Lines = connections between inputs and outputs
- **Input layer holds the variables that are input to the network…**
- **… multiplied by weights (coefficients) to get to result**
- **Single layer neural network is a GLM!**



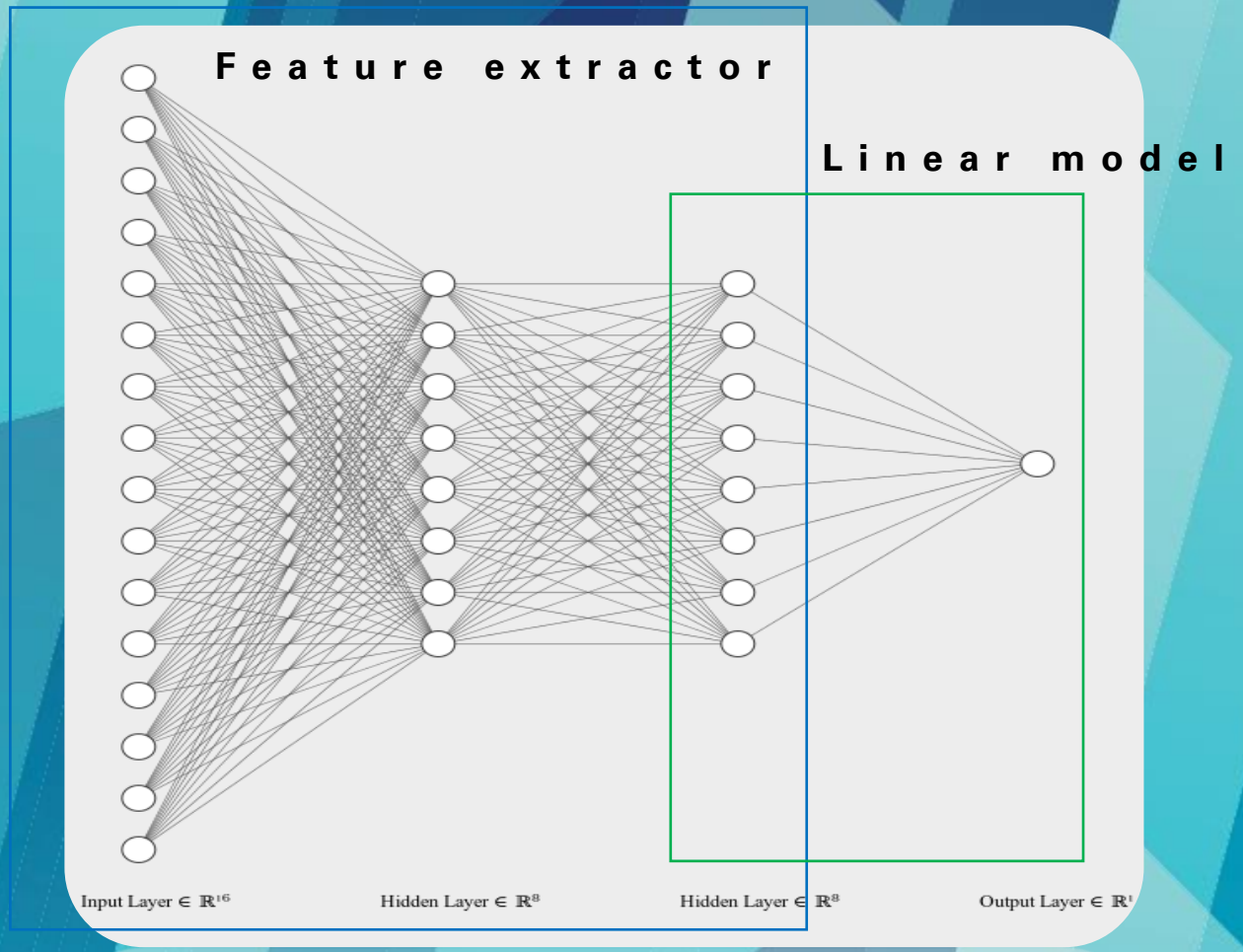
Deep Feedforward Net

- Deep = multiple layers
- Feedforward = data travels from left to right
- Fully connected network (FCN) = all neurons in layer connected to all neurons in previous layer
- More complicated representations of input data learned in hidden layers - subsequent layers represent regressions on the variables in hidden layers



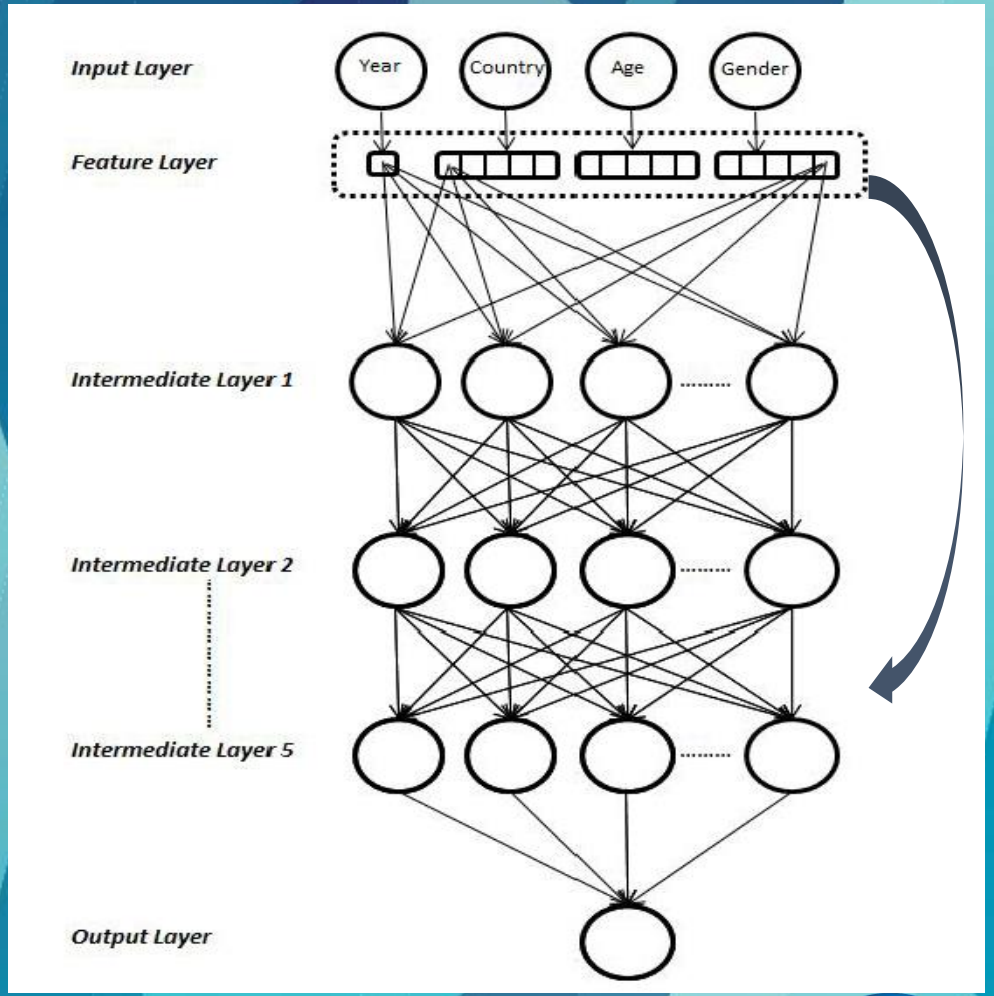
FCN generalizes GLM

- Intermediate layers = representation learning, guided by supervised objective.
- Last layer = (generalized) linear model, where input variables = new representation of data
- No need to use GLM – strip off last layer and use learned features in, for example, XGBoost
- Or mix with traditional method of fitting GLM



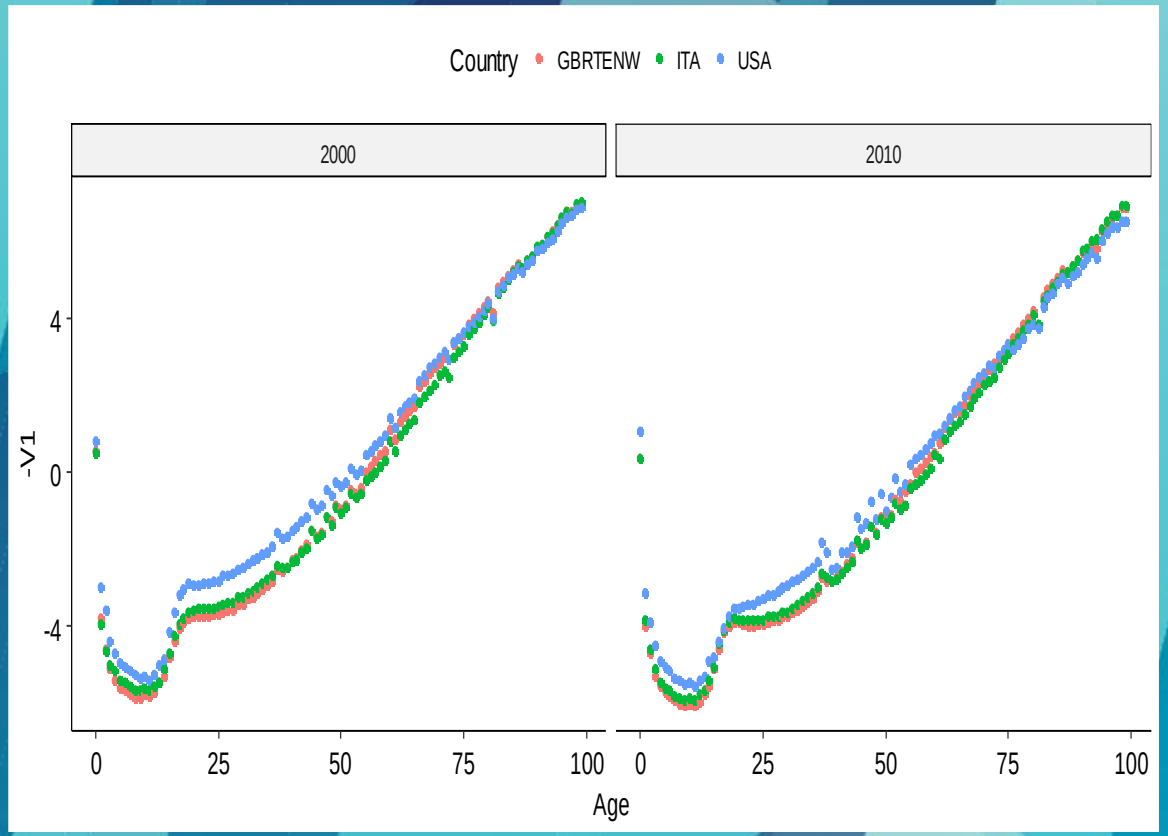
Example – Lee-Carter Neural Net

- Multi-population mortality forecasting model (Richman and Wüthrich 2018)
- Supervised regression on HMD data (inputs = Year, Country, Age; outputs = mx)
- 5 layer deep FCN
- Generalizes the LC model



Features in last layer of network

- **Representation = output of last layer (128 dimensions) with dimension reduced using PCA**
- **Can be interpreted as relativities of mortality rates estimated for each period**
- **Output shifted and scaled to produce final results**
- **Generalization of Brass Logit Transform where base table specified using NN (Brass 1964)**



$y_x = a + b * z_x^n$, where:

y_x = logit of mortality at age x
a,b = regression coefficients
z_x^n = logit of reference mortality

Specialized Architectures

- **Most modern examples of DL achieving state of the art results on tasks rely on using specialized architectures i.e. not simple fully connected networks**
- **Key principle - Use architecture that expresses useful priors (inductive bias) about the data => Achievement of major performance gains**

  Embedding layers – categorical data (or real values structured as categorical data)

  Autoencoder – unsupervised learning

  Convolutional neural network – data with spatial/temporal dimension e.g. images and time series

  Recurrent neural network – data with temporal structure

  Skip connections – makes training neural networks easier

- **Recently, specialized architectures have begun to be applied to actuarial problems**
- **Section ends with example of fine tuning a specialized architecture for a new task**

(Some) Actuarial Applications of DL

	Pricing	Reserving	Telematics	Mortality Forecasting	Quantitative Risk Management
Feed-forward Nets	• Ferrario, Noll and Wüthrich (2018) • Noll, Salzmann and Wüthrich (2018) • Wüthrich and Buser (2018)	• Castellani, Fiore, Marino et al. (2018) • Doyle and Groendyke (2018) • Gabrielli and Wüthrich (2018) • Hejazi and Jackson (2016, 2017) • Wüthrich (2018) • Zarkadoulas (2017)	• Gao and Wüthrich (2017) • Gao, Meng and Wüthrich (2018) • Gao, Wüthrich and Yang (2018)		• Castellani, Fiore, Marino et al. (2018) • Hejazi and Jackson (2016, 2017)
Convolutional Neural Nets			• Gao and Wüthrich (2019)		
Recurrent Neural Nets		• Kuo (2018a, 2018b)		• Nigri, Levantesi, Marino et al. (2019)	
Embedding Layers	• Richman (2018) • Schelldorfer and Wüthrich (2019) • Wüthrich and Merz (2019)	• Gabrielli, Richman and Wüthrich (2018) • Gabrielli (2019)		• Richman and Wüthrich (2018)	
Autoencoders			• Richman (2018)	• Hainaut (2018) • Richman (2018)	

Embedding Layer – Categorical Data

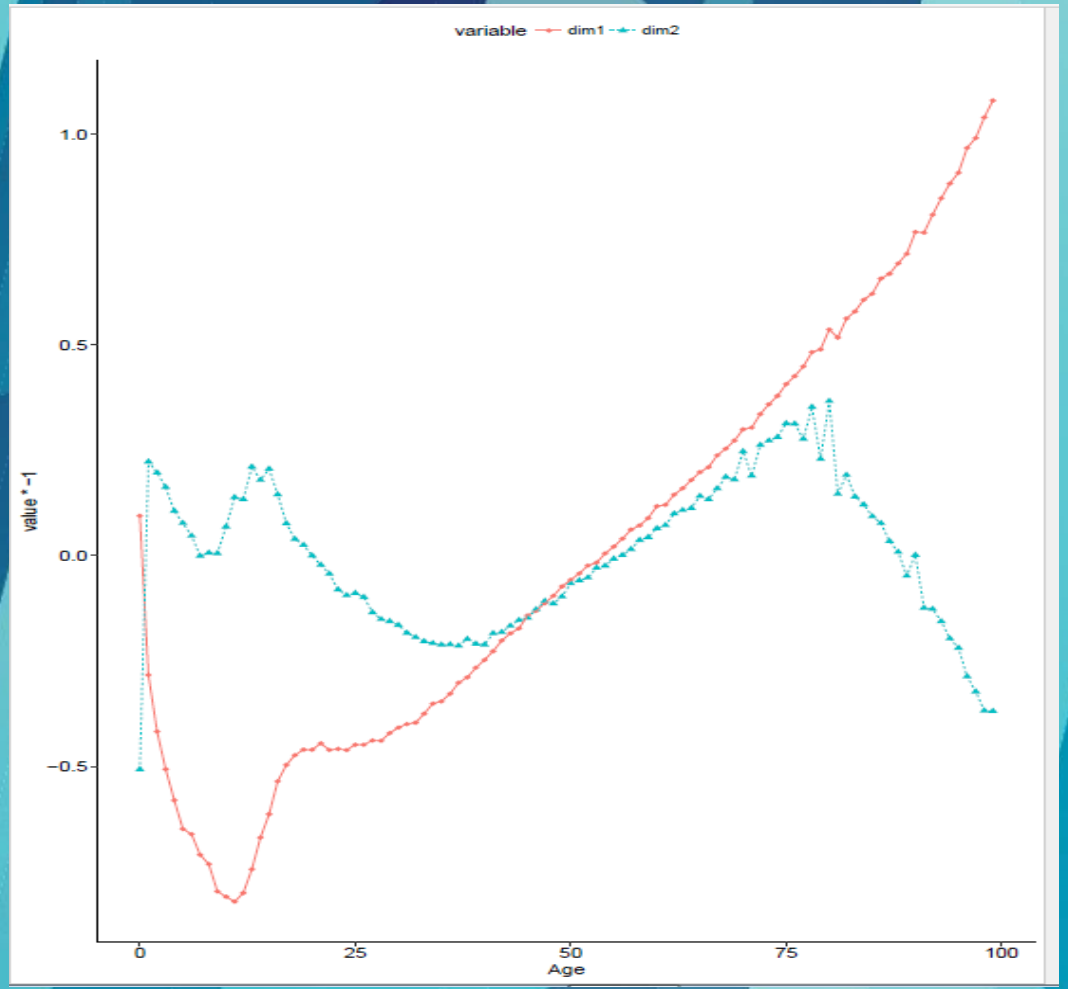
- **One hot encoding expresses the prior that categories are orthogonal => similar observations not categorized into groups**
- **Traditional actuarial solution – credibility**
- **Embedding layer prior – related categories should cluster together:**
  Learns dense vector transformation of sparse input vectors and clusters similar categories together
  Can pre-calibrate to MLE of GLM models, leading to CANN proposal of Wüthrich and Merz (2019

	Actuary	Accountant	Quant	Statistician	Economist	Underwriter
Actuary	1	0	0	0	0	0
Accountant	0	1	0	0	0	0
Quant	0	0	1	0	0	0
Statistician	0	0	0	1	0	0
Economist	0	0	0	0	1	0
Underwriter	0	0	0	0	0	1

	Finance	Math	Stastistics	Liabilities
Actuary	0.5	0.25	0.5	0.5
Accountant	0.5	0	0	0
Quant	0.75	0.25	0.25	0
Statistician	0	0.5	0.85	0
Economist	0.5	0.25	0.5	0
Underwriter	0	0.1	0.05	0.75

Learned embeddings

- Age embeddings extracted from LC NN model
- Five dimensions reduced using PCA
- Age relativities of mortality rates
- In deeper layers of network, combined with other inputs to produce representations specific to:
  - Country
  - Gender
  - Time
- First dimension of PCA is shape of lifetable
- Second dimension is shape of child, young and older adult mortality relative to middle age and oldest age mortality



Agenda

- From Machine Learning to Deep Learning
- Tools of the Trade
- Selected Applications
- Stability of Results
- Discrimination Free Pricing

Selected Applications

- **Following examples chosen to showcase ability of deep learning to solve the issues with the traditional actuarial (or ML) approaches.**
- **In most of these instances, deep learning solution outperforms the traditional actuarial or machine learning approach**
- **Complexity – which are the relevant features to extract/what is the correct model specification?**

  Multi-population mortality forecasting
  Multi LoB IBNR reserving
  Non-life pricing

- **Expert knowledge – requires suitable prior knowledge, which can take decades to build**

  Analysis of telematics data

- **Effort – designing relevant features is time consuming/tedious => limits scope and applicability**

  Lite valuation models

Multi-population mortality forecasting

- **Availability of multiple high quality series of mortality rates, but how to translate into better forecasts?**

- **Multi-population models (Kleinow 2015; Li and Lee 2005)**
  Many competing model specifications, without much theory to guide model selection
  Relatively disappointing performance of two models (CAE and ACF)

- **Richman and Wüthrich (2018) – deep neural net with embedding layers**

- **Outperforms both single and multiple populations models**

	Model	Average MSE	Median MSE	Best Performance
1	LC_SVD	5.50	2.48	19
2	ACF_SVD_region	3.46	2.50	36
3	ACF_SVD_country	7.30	4.77	9
4	ACF_BP	6.12	3.00	12

	Model	Average MSE	Median MSE	Best Performance
1	LC_SVD	5.50	2.48	33
2	CAE_SVD	4.76	2.35	13
3	CAE2_SVD	12.01	1.79	14
4	CAE2_BP	5.59	3.46	16

	Model	Average MSE	Median MSE	Best Performance
1	LC_SVD	5.50	2.48	7
2	LC_ACF_region	3.46	2.50	10
3	ACF_BP	6.12	3.00	4
4	CAE_BP	5.59	3.46	4
5	DEEP	2.68	1.38	51

Multi LoB IBNR reserving (1)

- **Even using triangles, most reserving exercises are more data rich than assumed by traditional (widely applied) methods (CL/BF/CC):**
  - Incurred/Paid/Outstanding
  - Amounts/Cost per Claim/Claim Counts
  - Multiple LoBs
  - Multiple Companies

- **Traditional solutions:**
  - Munich Chain Ladder (Quarg and Mack 2004) reconciles Incurred and Paid triangles (for single LoB) by adding a correction term to the Chain Ladder formula based on regression
  - Credibility Chain Ladder (Gisler and Wüthrich 2008) derives LDFs for sub-portfolios of a main LoB using credibility theory
  - Double Chain Ladder (Miranda, Nielsen and Verrall 2013) relates incurred claim count triangles to payment triangles

- **Would assume that multi-LoB methods have better predictive performance compared univariate methods, but no study (yet) comparing predictive performance of multi-LoB methods (Meyers (2015) compares several univariate reserving models)**

- **General statistical solution for leveraging multiple data sources not proposed**

Multi LoB IBNR reserving (2)

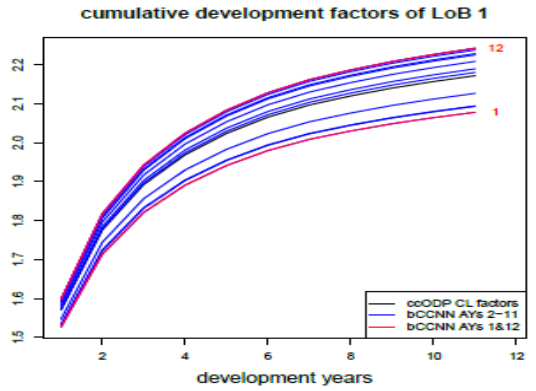
- **Recent work embedding the ODP CL model into a deep neural network (multi-LoB solution)**

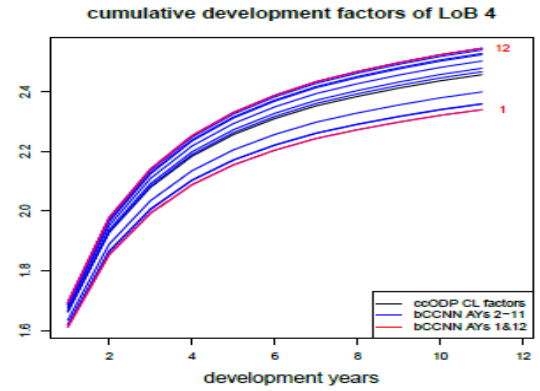
- **6 Paid triangles generated using the simulation machine of Gabrielli and Wüthrich (2018)**
  - Know true reserves
  - Relatively small data (12*12*6=478 data points)

- **Gabrielli, Richman and Wüthrich (2018) use classical ODP model plus neural boosting on 6 triangles simultaneously**
  - Dramatically reduced bias compared to ODP model
  - Model learns smooth development factors adjusting for accident year effects

- **Gabrielli (2019) extends model to include both paid and count data**
  - Further reduction in bias versus the previous model

	LoB 1	LoB 2	LoB 3	LoB 4	LoB 5	LoB 6
true reserves R_m^{true}	39,689	37,037	16,878	71,630	72,548	31,117
CL reserves R_m^{CL}	38,569	35,460	15,692	67,574	70,166	29,409
bCCNN reserves R_m^{LoB} (LoBs individually)	39,233	35,899	15,815	70,219	70,936	30,671
bCCNN reserves R_m^{+} (multiple LoBs)	40,271	37,027	16,400	70,563	73,314	30,730





		LoB 1	LoB 2	LoB 3	LoB 4	LoB 5	LoB 6
(i)	true claims reserves R_m^{true}	39'689	37'037	16'878	71'630	72'548	31'117
(ii)	CL reserves R_m^{CL}	38'569	35'460	15'692	67'574	70'166	29'409
(iii)	single NNDODP reserves R_m^{ind}	39'407	36'283	16'123	70'547	71'873	31'092
(iv)	multiple NNDODP reserves R_m^{joint}	40'403	37'172	16'434	70'727	73'513	30'770

Non-life pricing (1)

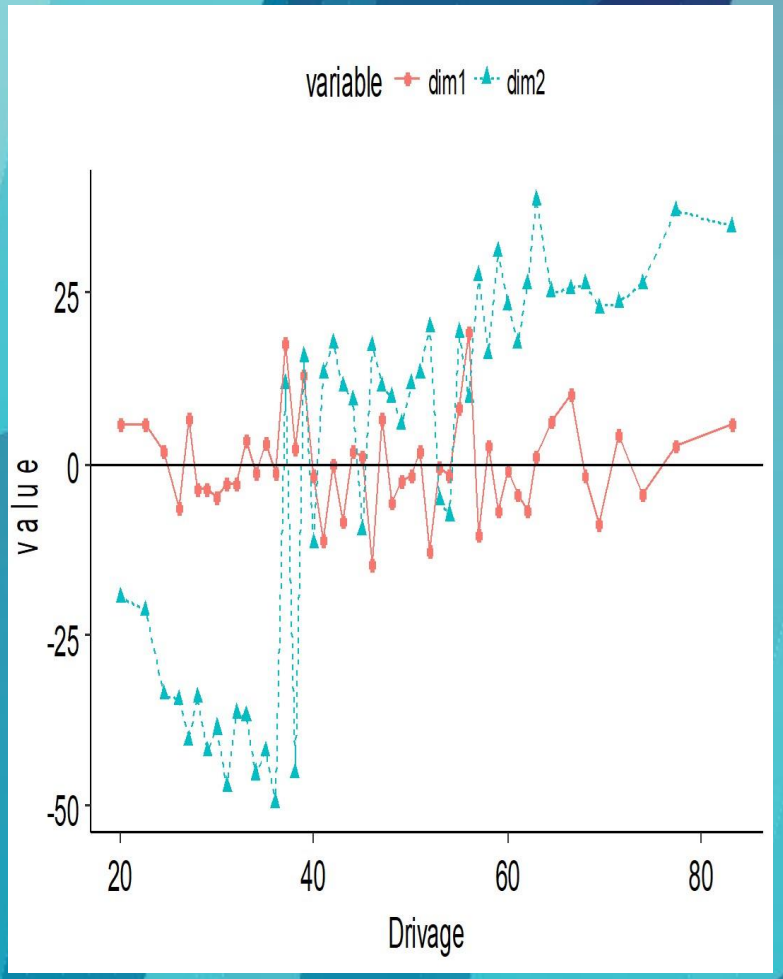
- **Non-life Pricing (tabular data fit with GLMs) seems like obvious application of ML/DL**

- **Noll, Salzmann and Wüthrich (2018) is tutorial paper (with code) in which apply GLMs, regression trees, boosting and (shallow) neural networks to French TPL dataset to model frequency**

ML approaches outperform GLM
Boosted tree performs about as well as neural network…
….mainly because ML approaches capture some interactions automatically
In own analysis, found that surprisingly, off the shelf approaches do not perform particularly well on frequency models
These include XGBoost and ‘vanilla’ deep networks

Non-life pricing (2)

- **Deep neural network applied to raw data (i.e. no feature engineering) did not perform well**
- **Embedding layers provide significant gain in performance over GLM and other NN architectures**
  - Beats performance of best non-deep model in Noll, Salzmann and Wüthrich (2018) (OOS Loss = 0.3141 using boosting)
- **Layers learn a (multi-dimensional) schedule of relativities at each age (shown after applying t-SNE)**
- **Transfer learning – use the embeddings learned on one partition of the data, for another unseen partition of data**
  - Boosts performance of GLM



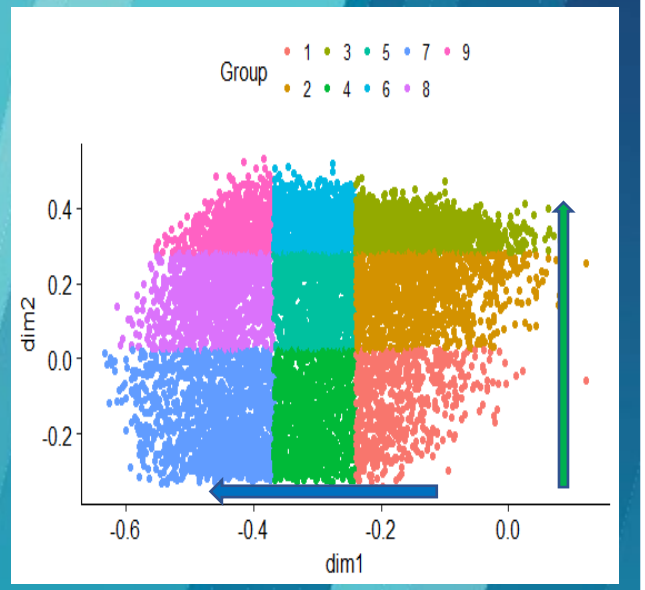
Model	OutOfSample
GLM	0.3217
GLM_Keras	0.3217
NN_shallow	0.3150
NN_no_FE	0.3258
NN_embed	0.3068
GLM_embed	0.3194
NN_learned_embed	0.2925

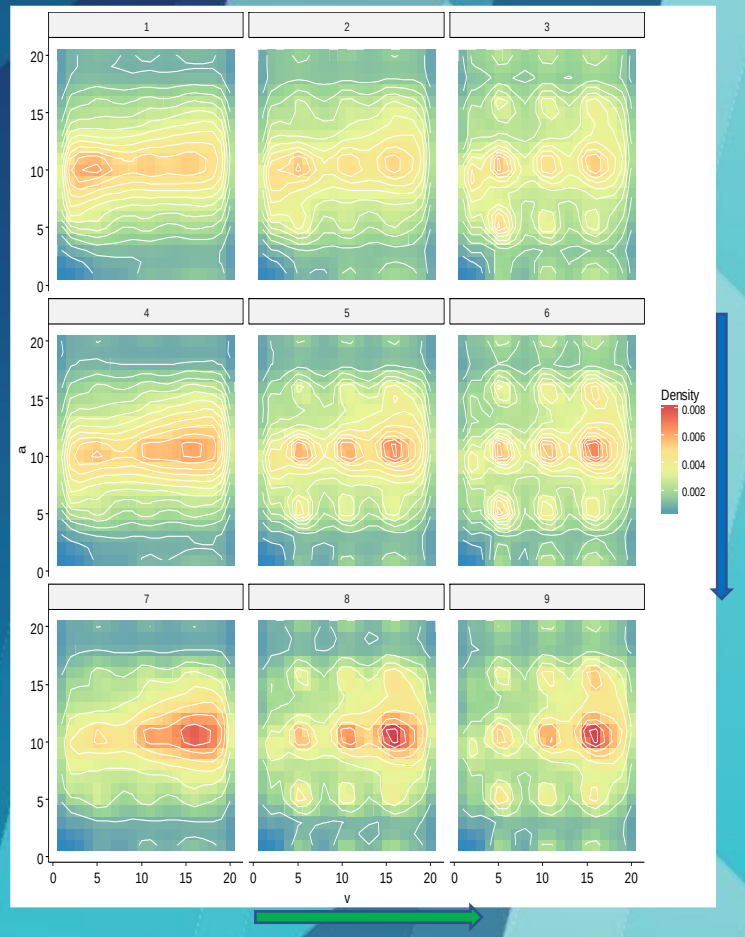
Telematics data (1)

- **Telematics produces high dimensional data (position, velocity, acceleration, road type, time of day) at high frequencies – new type of data for actuarial science!**
  - To develop "standard" models/approaches for incorporating into actuarial work might take many years => rely on deep learning
- **Most immediately obvious how to incorporate into pricing - most approaches look to summarize telematics data streams before analysis with deep learning**
- **From outside actuarial literature, feature matrices containing summary statistics of trips analysed using RNNs plus embedding layers such as Dong, Li, Yao et al. (2016), Dong, Yuan, Yang et al. (2017) and Wijnands, Thompson, Aschwanden et al. (2018)**
- **For pricing (within actuarial literature) series of papers by Wüthrich (2017), Gao and Wüthrich (2017) and Gao, Meng and Wüthrich (2018) discuss analysis of velocity and acceleration information from telematics data feed**
- **Focus on v-a density heatmaps which capture velocity and acceleration profile of driver but these are also high dimensional**
- **Wüthrich (2017) and Gao and Wüthrich (2017) apply unsupervised learning methods (K-means, PCA and shallow auto-encoders) to summarize v-a heat-maps - Stunning result = continuous features are highly predictive**
  - Unsupervised learning applied to high dimensional data produces useful features for supervised learning

Telematics data (2)

- **Analysis using deep convolutional autoencoder with 2 dimensions.**
- **Within these dimensions (left hand plot):**
  - Right to left = amount of density in high speed bucket
  - Lower to higher = "discreteness" of the density
- **Another application is to identify drivers for UBI at correct rate (and use resulting features for pricing). See Gao and Wüthrich (2019) who apply CNNs to identify drivers based on velocity/acceleration/angle**
  - 75% accuracy on 180s of data





Lite Valuation Models (1)

- **Major challenge in valuation of Life business with embedded options/guarantees or with-profits is run time of (nested) stochastic models**
- **In general, for Variable Annuity business, guarantees are priced and hedged using Monte Carlo simulations**
- **Under Solvency II, Life business with options/guarantees must be valued using nested Monte Carlo to derive the Solvency Capital Requirements (SCR)**
  - Outer loop - MC simulations to derive risk factors at t+1 under the real world measure
  - Inner loops - MC simulations to derive valuation given risk factors at t+1 under risk neutral measure
- **Running full MC valuation is time consuming; common solutions are:**
  - High performance computing
  - Replicating portfolios
  - Least Squares Monte Carlo (LSMC), where regression fit to results of inner loop conditional on outer loop
  - "Lite" valuation models, see work by Gan and Lin (2015)

Lite Valuation Models (2)

- **Recent work using neural networks to enhance this process**
- **Hejazi and Jackson (2016, 2017) provide novel approach based on matching prototype contracts**
- **For VA valuation and hedging, Doyle and Groendyke (2018) build a lite valuation model using a shallow neural network that takes key market and contract data and outputs contract value and hedging parameters.**

  Achieve highly accurate results versus full MC approach.
- **For modelling with-profits contracts in SII, Nigri, Levantesi, Marino et al. (2019) replace inner loop basis function regression of LSMC with SVM and a deep neural network, and compare results with full nested MC.**

  Find that DL beats the basis function regression and SVM, producing highly accurate evaluations of the SCR.

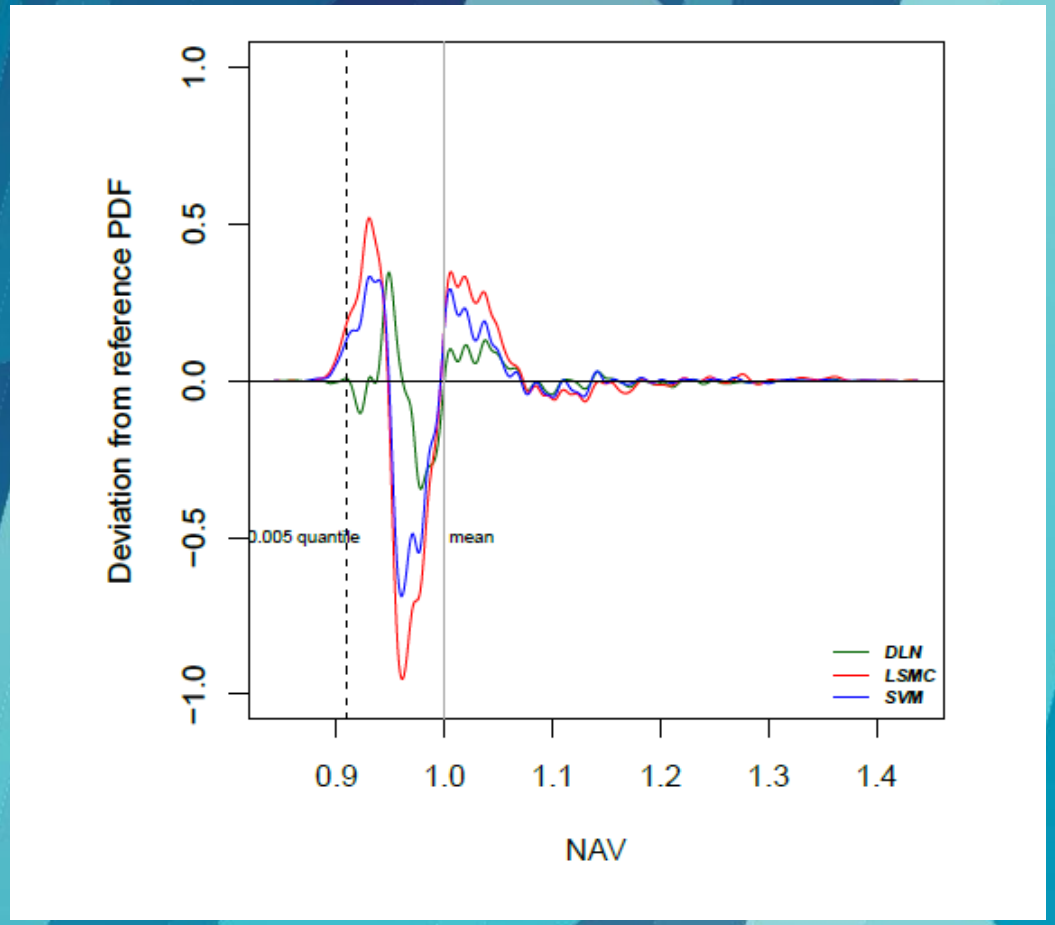


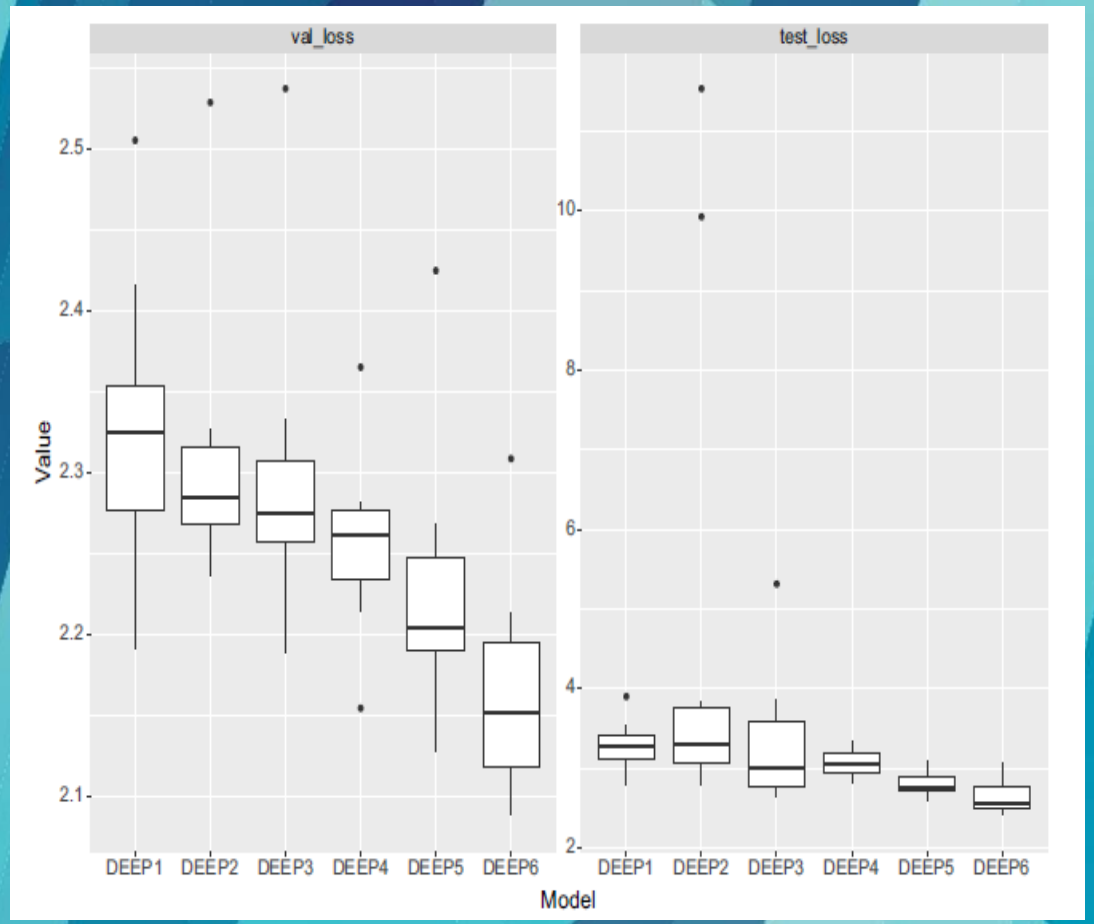
Diagram from Nigri, Levantesi, Marino et al. (2019

Agenda

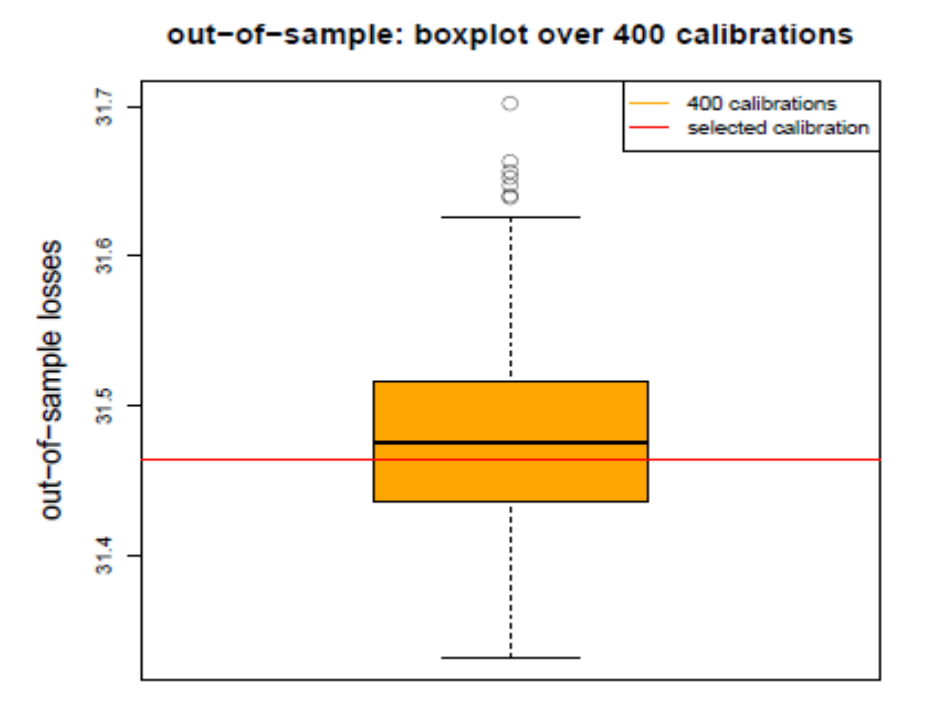
- From Machine Learning to Deep Learning
- Tools of the Trade
- Selected Applications
- Stability of Results
- Discrimination Free Pricing

Stability of results

- **The training of neural networks contains some randomness due to:**
  - Random initialization of parameters
  - Dropout
  - Shuffling of data
- **Leads to validation and test set results that can exhibit variability. Not a "new" problem; see Guo and Berkhahn (2016).**
- **Problem worse on small datasets (where other ML techniques are stable) and autoencoders**
- **Example – validation and test set results of 6 DL models run 10 times on LC NN model applied to full HMD dataset.**
- **Solutions - Average models over several runs or at several points in the training (see Gabrielli (2019))**
- **Results of network might not match portfolio average due to early stopping. See Wüthrich (2019) for analysis and solutions**

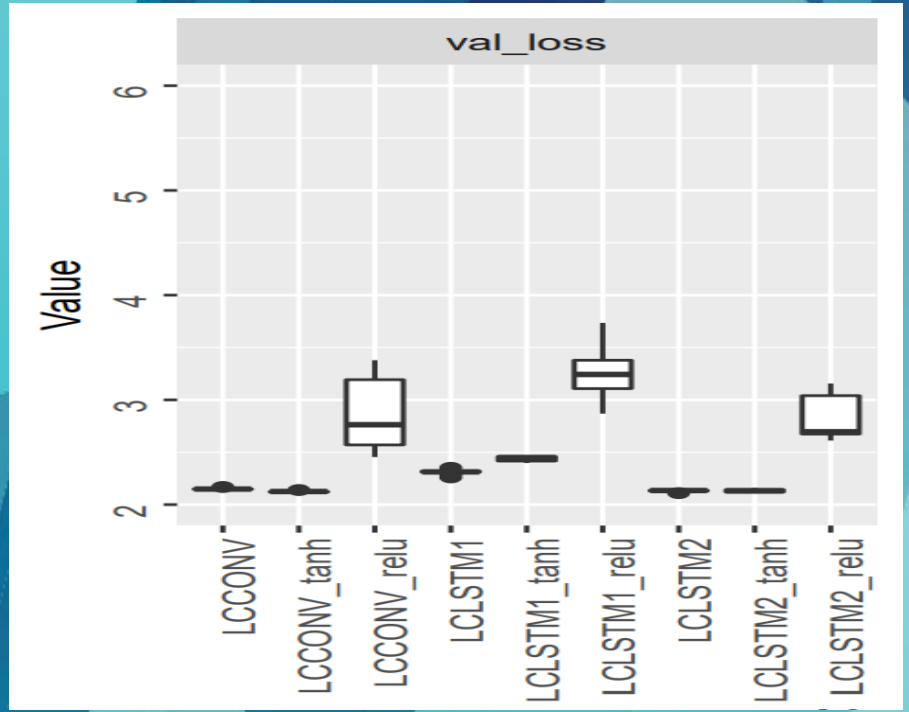


Recent Examples



Neural networks fit to French MTPL dataset

Richman and Wüthrich (2020)



Neural networks fit to HMD dataset

Perla, Richman, Scognamiglio and Wüthrich (2020)

Nagging Predictors

Richman, Ronald; Wüthrich, Mario V. 2020. "Nagging Predictors." Risks 8, no. 3: 83.

- Aggregating is a statistical technique that helps to reduce noise and uncertainty in predictors and is justified theoretically using the law of large numbers.
- An i.i.d. sequence of predictors is not always available thus, Breiman (1996) combined bootstrapping and aggregating, called bagging.
- This paper aims to combined networks and aggregating to receive the nagging predictor.
- Each run of the network training provides us with a new estimated network.
- Explore the statistical properties of the nagging predictors at a portfolio and at a policy level.

Crucial Difference between Bagging and Nagging

Bagging

Performs re-sampling on observations, thus, it tries to create new observations from the data D that follow a similar law as this original data. The re-sampling involves randomness and, therefore, bootstrapping is able to generate multiple random predictors $\hat{\mu}_i^{(j)}$.

Typically, these bootstrap predictors are i.i.d. by applying the same algorithm using i.i.d. seeds, but this i.i.d. property has to be understood conditionally on the given observations D.

Nagging

Not based on re-sampling data, but it always works on the same data set, and multiple predictors are obtained by exploring multiple models, or rather multiple parametrizations of the same model using gradient descent methods combined with early stopping.

Naturally, this involves less randomness compared to bootstrapping because the underlying data set for the different predictors is always the same.

$$\bar{\hat{\mu}}_t^{(M)} = \frac{1}{M}\sum_{j=1}^{M}\hat{\mu}_t^{(j)} = \frac{1}{M}\sum_{j=1}^{M}\mu(x_t^{\dagger}, \hat{\beta}^{(j)}), \qquad (13)$$

Regression Design for Predictive Modelling 1

Our goal is to build a predictive model for the claim counts N_i (`ClaimNb`) in Listing 1. We choose a Poisson regression model for data $(N_i, \boldsymbol{x}_i, v_i)$ satisfying

$$N_i \sim \text{Poi}\left(\mu(\boldsymbol{x}_i)v_i\right), \qquad \text{with expected frequency function } \boldsymbol{x}_i \mapsto \mu(\boldsymbol{x}_i).$$

The Poisson distribution belongs to the family of Tweedie's CP models with power variance parameter $p = 1$, effective domain $\Theta = \mathbb{R}$, cumulant function $\kappa_1(\theta) = \exp(\theta)$ and dispersion parameter $\phi = 1$. In fact, if we define $Y_i = N_i/v_i$, we have the following density w.r.t. the counting measure on $\mathbb{N}_0/v_i$

$$Y_i \sim f(y; \theta_i, v_i, p) = \exp\left\{v_i\left(y\theta_i - \exp(\theta_i)\right) + a_1(y; v_i)\right\}.$$

This provides the first two moments as

$$\mu_i = \mathbb{E}[Y_i] = \exp(\theta_i) \qquad \text{and} \qquad \text{Var}(Y_i) = \frac{1}{v_i}\exp(\theta_i) = \frac{1}{v_i}\mu_i.$$

The canonical link is given by the log-link, and we chose links $(K'^{-1}_P)(\cdot) = g(\cdot) = \log(\cdot)$.
These choices provide the network regression function on the canonical scale:

$$\boldsymbol{x}_i \mapsto \theta_i = \theta(\boldsymbol{x}_i) = \left(\kappa_1'\right)^{-1}\left(\mu(\boldsymbol{x}_i)\right) = \left\langle \boldsymbol{\beta}^{(d+1)}, \left(\boldsymbol{z}^{(d)} \circ \cdots \circ \boldsymbol{z}^{(1)}\right)(\boldsymbol{x}_i)\right\rangle, \tag{14}$$

The network predictor on the RHS gives the canonical parameter under the canonical link choice for $g(\cdot)$.

Regression Design for Predictive Modelling 2

Almost ready to fit the model to the data, i.e., to find a good network parameter $\boldsymbol{\beta} \in \mathbb{R}^r$ using the gradient descent algorithm.

As objective function for parameter estimation we choose the Poisson deviance loss function:

$$\begin{aligned}\mathcal{L}(\mathcal{D};\boldsymbol{\beta}) &= \frac{2}{n}\sum_{i=1}^{n} v_i \left[Y_i \log(Y_i) - Y_i - Y_i\theta_i + \exp(\theta_i)\right] \\ &= \frac{2}{n}\sum_{i=1}^{n} \mu(\boldsymbol{x}_i)v_i - N_i - N_i \log\left(\mu(\boldsymbol{x}_i)v_i/N_i\right),\end{aligned}$$

for regression function $\boldsymbol{x}_i \mapsto \mu_i = \mu(\boldsymbol{x}_i)$ defined through Equation (14), and where the terms under the summations are set equal to $2\mu(\boldsymbol{x}_i)v_i$ in case $N_i = 0$.

Author: Ronald Richman (FIA, FASSA, CERA), Associate Director, R&D & Special Projects at QED Actuaries & Consultants

Learning and Test Data

Features are pre-processed analogously to the example in Section 3.3.2 of Wüthrich (2019), i.e., use MinMaxScaler for continuous explanatory variables and two-dimensional embedding layers for categorical covariates.

Having this pre-processed data, we specify the choice of learning data D on which the model is learned, and the test data T on which we perform the out-of-sample analysis. To keep comparability with the results in Noll et al. (2018); Wüthrich (2019) we use exactly the same partition. Namely, 90% of all policies in Listing 1 are allocated to learning data D and the remaining 10% are allocated to test data T . This allocation is done at random, and we use the same seed as in Noll et al. (2018); Wüthrich (2019).

	Numbers of Observed Claims					Empirical	Size of
	0	1	2	3	4	Frequency	Data Sets
empirical probability on $\mathcal{D}$	94.99%	4.74%	0.36%	0.01%	0.002%	10.02%	$n = 610,212$
empirical probability on $\mathcal{T}$	94.83%	4.85%	0.31%	0.01%	0.003%	10.41%	$m = 67,801$

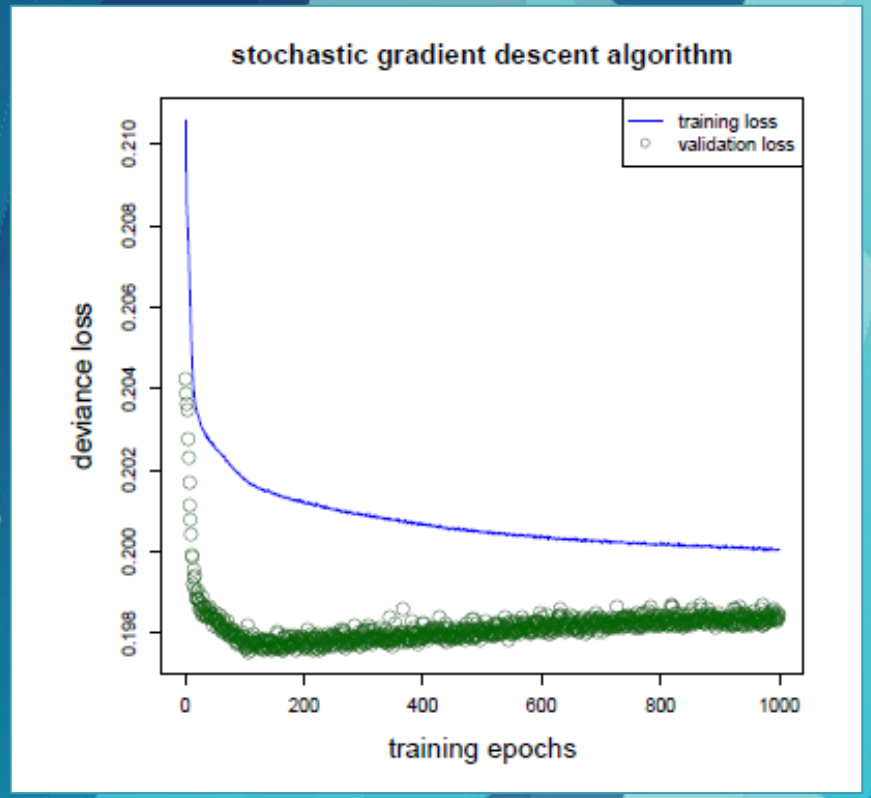
Gradient Descent Fitting 1

Need to ensure that the network does not over-fit to the learning data D. To ensure this, we partition at random 9:1 into a training data set $D^{(-)}$ and a validation set V.

Network parameter is learned on $D^{(-)}$ and over-fitting is tracked on V.

Run the nadam gradient descent algorithm over 1000 epochs on random mini-batches of size 5000 from $D^{(-)}$. Using a *callback* we retrieve the network parameter that has the smallest loss on V – stopping rule in place.

The fact that the resulting network model has been received by an early stopping of the gradient descent algorithm implies that this network has a bias w.r.t. the learning data D.



Gradient Descent Fitting 2

Additionally applied the bias regularization step proposed in Section 3.4 of Wüthrich (2019). This gives us an estimated network parameter $\widehat{\boldsymbol{\beta}}^{(1)}$ and corresponding mean estimates $\hat{\mu}_i^{(1)}$ for all insurance policies in D and T . This procedure leads to the results on line (d) in the table below:

	In-Sample Loss on $\mathcal{D}$		Out-of-Sample Loss on $\mathcal{T}$	
(a) homogeneous model	32.935		33.861	
(b) generalized linear model	31.267		32.171	
(c) boosting regression model	30.132		31.468	
(d) network regression model (seed $j = 1$)	30.184		31.464	
(e) average over 400 network calibrations	30.230	(0.089)	31.480	(0.061)
(f) nagging predictor for $M = 400$	30.060		31.272	

We compare the network results to the ones received in Table 11 of Noll et al. (2018), and we conclude that our network approach is competitive with these other methods (being a classical GLM and a boosting regression model), see the out-of-sample losses on lines (a)–(d).

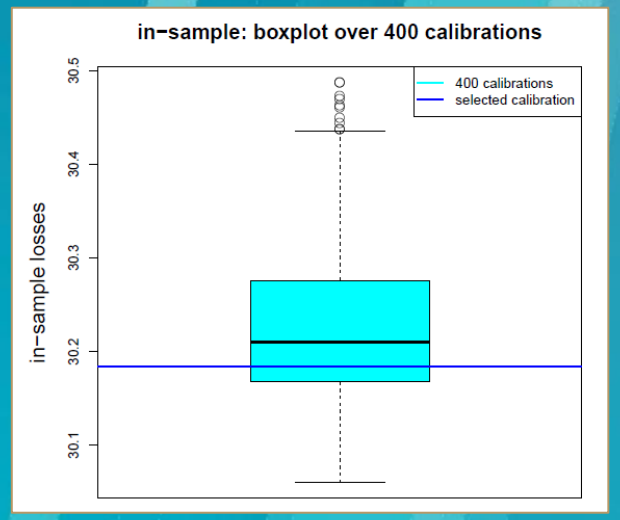
Comparison of Different Networks 1

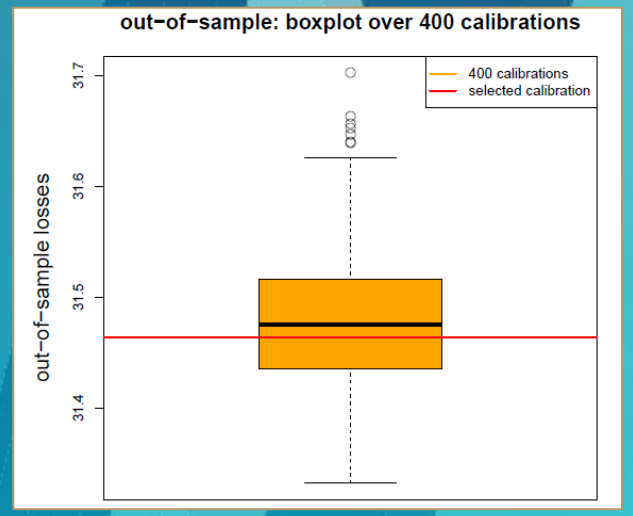
The issue with the network result now is that it involves quite some randomness.

We run the calibration procedure under identical choices of all hyper-parameters, but we choose different seeds for the random choices (R1)-(R3).

The boxplots below shows the in-samples losses L(D;$\hat{\beta}^{(j)}$) and out-of-samples losses L(T ; $\hat{\beta}^{(j)}$) over 400 network calibrations $\hat{\beta}^{(j)}$.

(R1) - randomly split the learning data into $D^{(-)}$ and V.
(R2) - randomly split $D^{(-)}$ into mini-batches of size 5000.
(R3) - randomly choose the starting point of the gradient descent algorithm.





Comparison of Different Networks 2

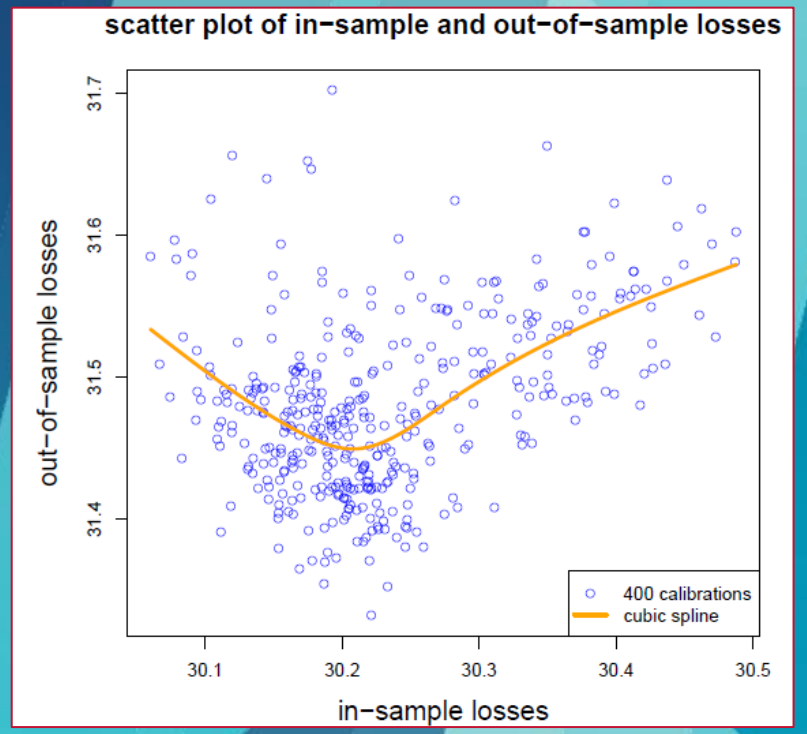
These losses have a rather large range which indicates that results of single network calibrations are not very robust.

We can calculate empirical mean and standard deviation for the 400 seeds j given by:

$$\overline{\mathcal{L}}(\mathcal{T};\widehat{\boldsymbol{\beta}}^{(1:400)}) = \frac{1}{400}\sum_{j=1}^{400}\mathcal{L}(\mathcal{T};\widehat{\boldsymbol{\beta}}^{(j)}) \qquad \text{and} \qquad \sqrt{\frac{1}{399}\sum_{j=1}^{400}\left(\overline{\mathcal{L}}(\mathcal{T};\widehat{\boldsymbol{\beta}}^{(1:400)}) - \mathcal{L}(\mathcal{T};\widehat{\boldsymbol{\beta}}^{(j)})\right)^2}.$$

The first gives an estimate for the expected generalization loss averaged over the corresponding portfolios. We emphasize in notation $\widehat{\beta}^{(1:400)}$ that we do not average over network parameters, but over deviance losses on individual network parameters $\widehat{\beta}^{(j)}$. The resulting numbers are given on line (e) from the table above. This shows that the early stopped network calibrations have quite significant differences, which motivates the study of the nagging predictor.

The scatter plot shows the in-sample and out-of-sample losses over the 400 different runs of the gradient descent fitting (complemented with a natural cubic spline).

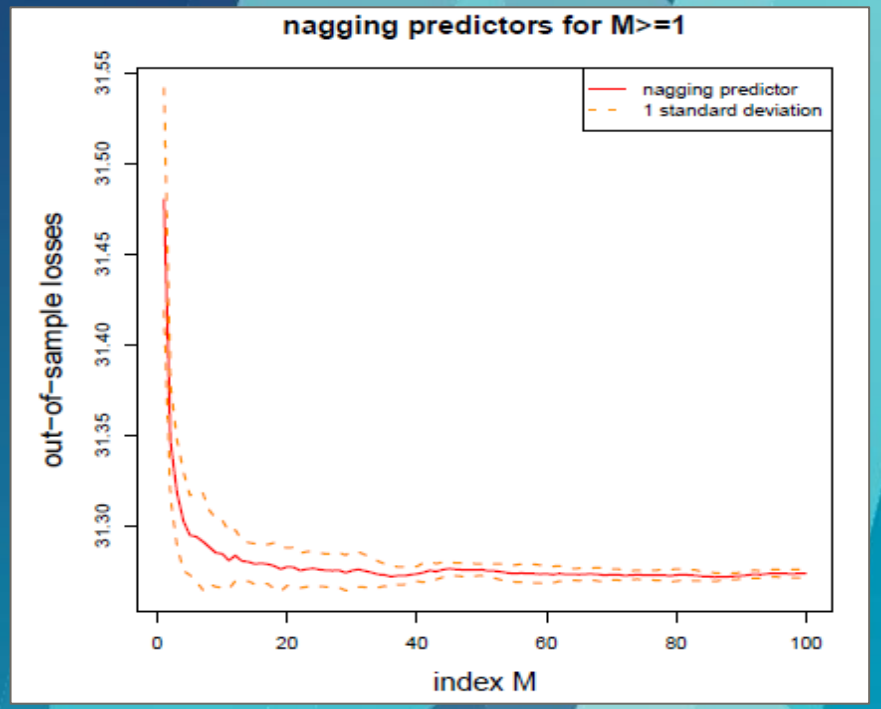


The Nagging Predictor

Able to calculate the nagging predictors $\bar{\bar{\mu}}_t^{(M)}$ over the test data set T . For M → ∞ this provides us with empirical counterparts of Propositions 3 and 4. We therefore consider for M ≥ 1 the sequence of out-of-sample losses:

$$\mathcal{L}(\mathcal{T}; \bar{\bar{\mu}}_{t=1,\ldots,m}^{(M)}) = \frac{1}{m} \sum_{t=1}^{m} \delta(Y_t^{\dagger}, \bar{\bar{\mu}}_t^{(M)}),$$

The figure gives the out-of-sample losses of the nagging predictors L(T ; $\bar{\bar{\mu}}_{t=1,\ldots,m}^{(M)}$) for M = 1, . . . , 100. Most noticeable is that nagging leads to a substantial improvement in out-of-sample losses, for M → ∞ the out-of-sample loss converges to 31.272. From this we conclude that nagging helps to improve the predictive model substantially.



Author: Ronald Richman (FIA, FASSA, CERA), Associate Director, R&D & Special Projects at QED Actuaries & Consultants

Pricing of Individual Insurance Policies

Should also ensure that we have robustness of prices on an individual insurance policy level. We analyze by how much individual insurance policy prices may differ if we select two different network calibrations $\widehat{\boldsymbol{\beta}}^{(j)}$ and $\widehat{\boldsymbol{\beta}}^{(j')}$. This will tell us whether aggregating over 20 or 40 network calibrations is sufficient.

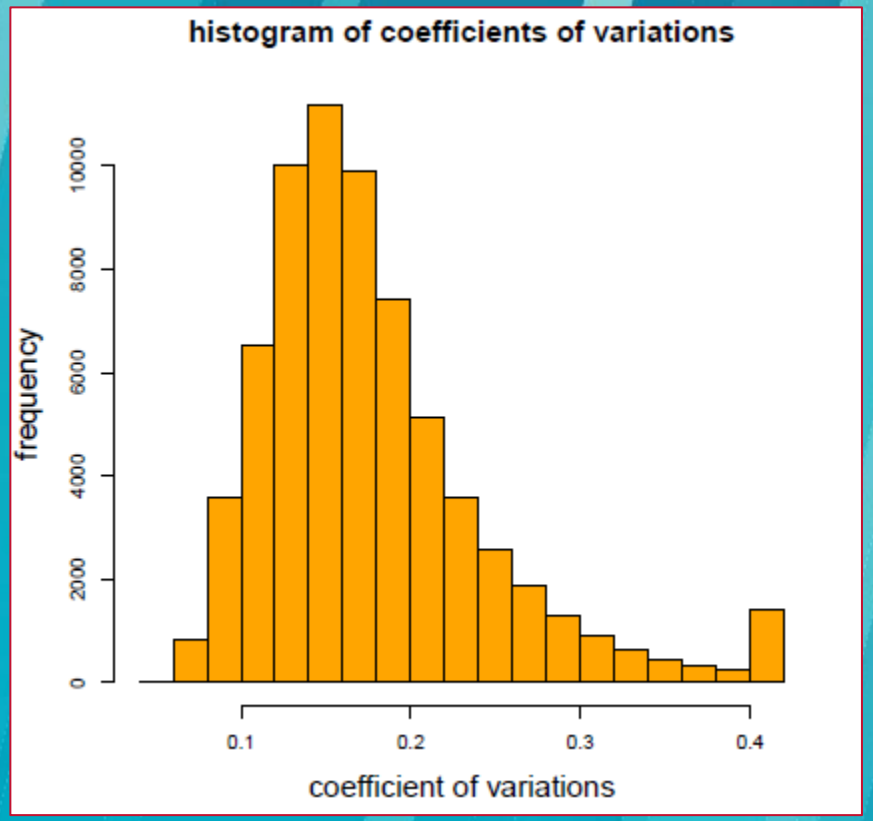
We expect that we need to average over more networks because the former statement includes an average over T , i.e., over m = 67, 801 insurance policies (though there is dependence between these policies because they simultaneously use the same network parameter estimate $\widehat{\boldsymbol{\beta}}^{(j)}$.

We calculate for each policy t = 1, . . . ,m of the test data T, $\bar{\mu}_t^{(M)}$ over M = 400 different network calibrations $\widehat{\boldsymbol{\beta}}^{(j)}$, j = 1, . . . ,M, and we calculate the empirical coefficients of variation in the individual network predictors given by:

$$\widehat{\mathrm{CoV}}_t = \frac{\widehat{\sigma}_t}{\bar{\mu}_t^{(M)}} = \frac{\sqrt{\frac{1}{M-1}\sum_{j=1}^{M}\left(\widehat{\mu}_t^{(j)} - \bar{\mu}_t^{(M)}\right)^2}}{\bar{\mu}_t^{(M)}}, \qquad (15)$$

Observations on Coefficients of Variation

We plot a histogram for the coefficients of variation against the nagging predictors for each single insurance policy t = 1,..., m (out-of-sample on T):



Observe most policies (73%) have a CoV of less than 0.2, however, on 11 of the m = 67, 801policies have a CoV bigger than 1. Thus, for the latter, if we average over 400 different network calibrations $\widehat{\boldsymbol{\beta}}^{(j)}$ we still have an uncertainty of $1/\sqrt{400}$, i.e., the prices have a precision of 5% to 10% in these latter cases (this is always conditional given D).

From this we conclude that on individual insurance policies we need to aggregate over a considerable number of networks to receive stable network regression prices.

Focus on Observations with CoV > 1

We list the 11 policies in the table below:

Listing 2. Policies with high coefficients of variation $\widehat{\text{CoV}}_t$.

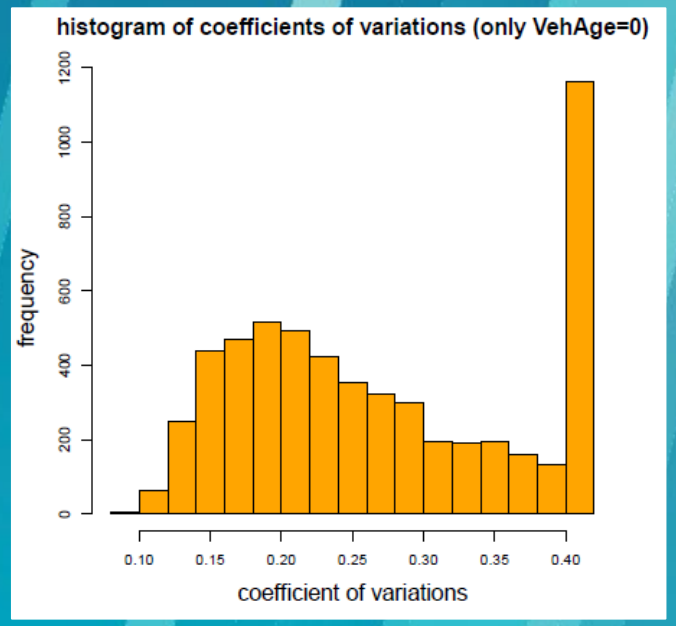
	Area	VehPower	VehAge	DrivAge	BonusMalus	VehBrand	VehGas	Density	Region
1	A	6	0	51	50	B3	Diesel	2.71	R21
2	A	6	0	51	50	B3	Diesel	2.71	R21
3	B	9	0	30	125	B3	Regular	4.32	R26
4	E	15	0	75	67	B14	Regular	8.38	R72
5	B	6	0	29	60	B3	Diesel	4.30	R21
6	A	10	0	29	60	B13	Regular	2.08	R24
7	E	9	0	31	125	B4	Diesel	8.35	R11
8	A	7	0	69	50	B14	Diesel	3.83	R82
9	A	10	0	59	50	B1	Diesel	3.33	R21
10	A	10	0	59	50	B1	Diesel	3.33	R21
11	A	10	0	59	50	B1	Diesel	3.33	R21

Striking is that all these policies have vehicle age **VehAge** = 0.

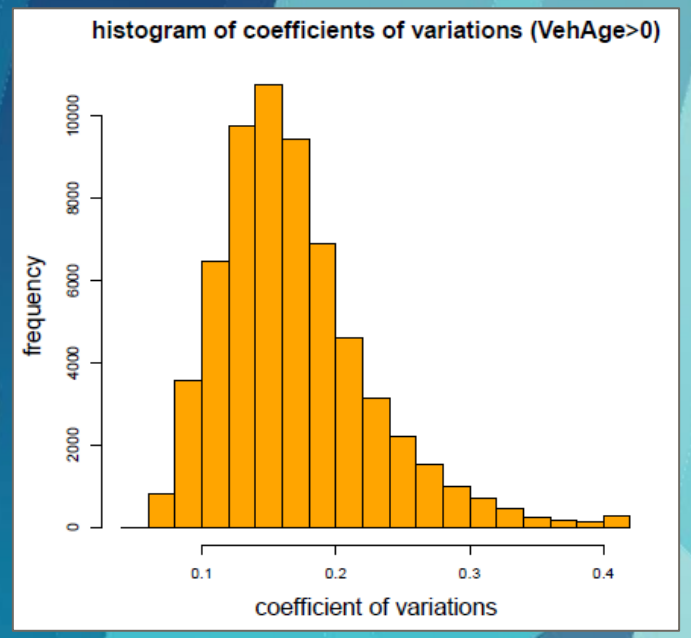
We will proceed to analyse policies with **VehAge** = 0 and **VehAge** > 0 separately.

Uncertainty in VehAge = 0

VehAge = 0 :



VehAge > 0 :



We indeed confirm that mainly policies with **VehAge** = 0 are difficult to price.

Meta Network Regression Model

Although the nagging predictor substantially improves the predictive model, it may not be fully satisfactory in practice. The difficulty is that it involves aggregating over M = 400 predictors $\hat{\mu}_i^{(j)}$ for each policy i.

For this reason we propose to build a meta model that fits a new network to the nagging predictors $\bar{\hat{\mu}}_i^{(M)}$, i = 1,..., n – "model distillation".

Since these nagging predictors are aggregated over M network models, and since the network regression functions are smooth functions in input variables (continuous features), the nagging predictors describe smooth surfaces.

Comparably simple to fit a network to the smooth surface described by nagging predictors $\bar{\hat{\mu}}_i^{(M)}$, i = 1,..., n , and over-fitting will not be an issue.

Optimal Model

The resulting in-sample and out-of-sample losses are in the table below:

	In-Sample Loss on $\mathcal{D}$		Out-of-Sample Loss on $\mathcal{T}$	
(d) network regression model (seed $j = 1$)	30.184		31.464	
(e) average over 400 network calibrations	30.230	(0.089)	31.480	(0.061)
(f) nagging predictor for $M = 400$	30.060		31.272	
(g1) meta network model (un-weighted)	30.260		31.342	
(g2) meta network model (weighted)	30.257		31.332	

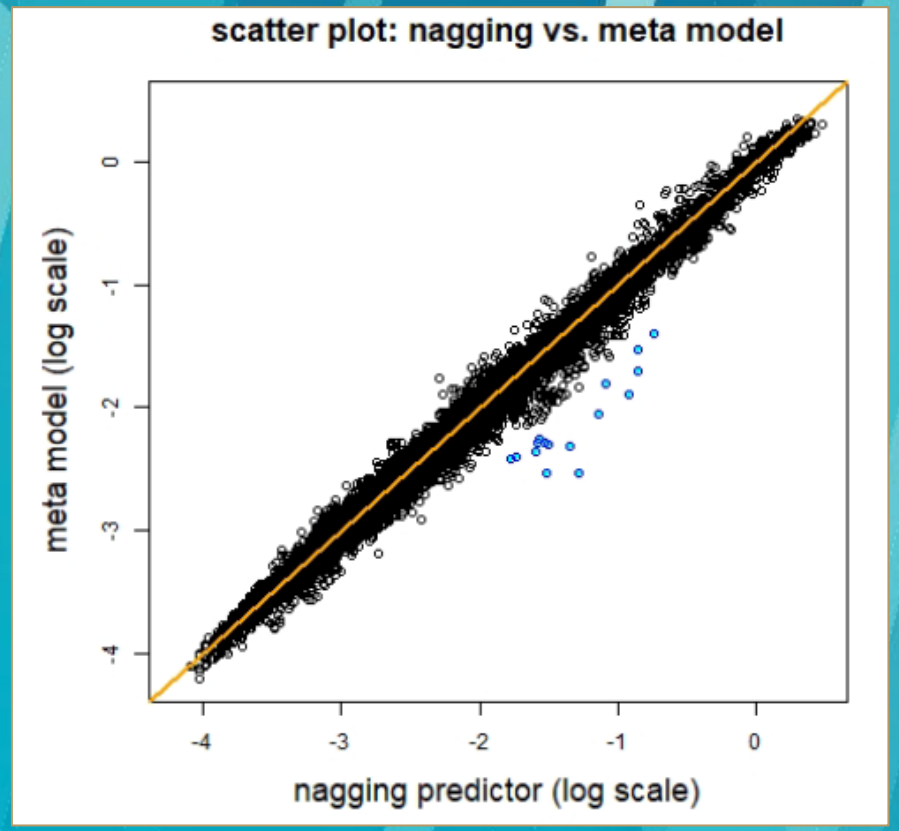
The weighted version (g2) has a better loss performance than the unweighted version.

It is slightly worse than the nagging predictor model, however substantially better than the individual network models and easier in handling than the nagging predictor.

For this reason, we are quite satisfied by the meta model, and we propose to hold on to this model for further analysis and insurance pricing.

Nagging Predictor vs Meta Model Predictor

The scatterplot below presents the two predictors:



The models are reasonably equal with the biggest differences highlighted in blue.

These refer to the policies with vehicle age 0 – the feature component within the data that is the most difficult to fit with the network model.

Agenda

- From Machine Learning to Deep Learning
- Tools of the Trade
- Selected Applications
- Stability of Results
- Discrimination Free Pricing

Discrimination Free Insurance Pricing

M. Lindholm, R. Richman, A. Tsanakas, and M. V. Wüthrich, "Discrimination-Free Insurance Pricing," SSRN Electron. J., Jan. 2020, doi: 10.2139/ssrn.3520676 Available: https://bit.ly/38huODw.

Current environment

- More advanced techniques becoming widely known and used
- Increasing scrutiny internationally on pricing practices (e.g. FCA review)

Legal/ethical requirements

- Legal (e.g. EU ban on gender based pricing) and ethical concerns (e.g. postal code ~= race in South Africa)
- How to ensure models are not influenced by discriminatory factors?

Naïve Solution = Unawareness Prices

- Ignore the problem by leaving out discriminatory rating factors
- Could advanced models figure out proxies for these factors?
- Actually, even simple models can do this!

Definitions

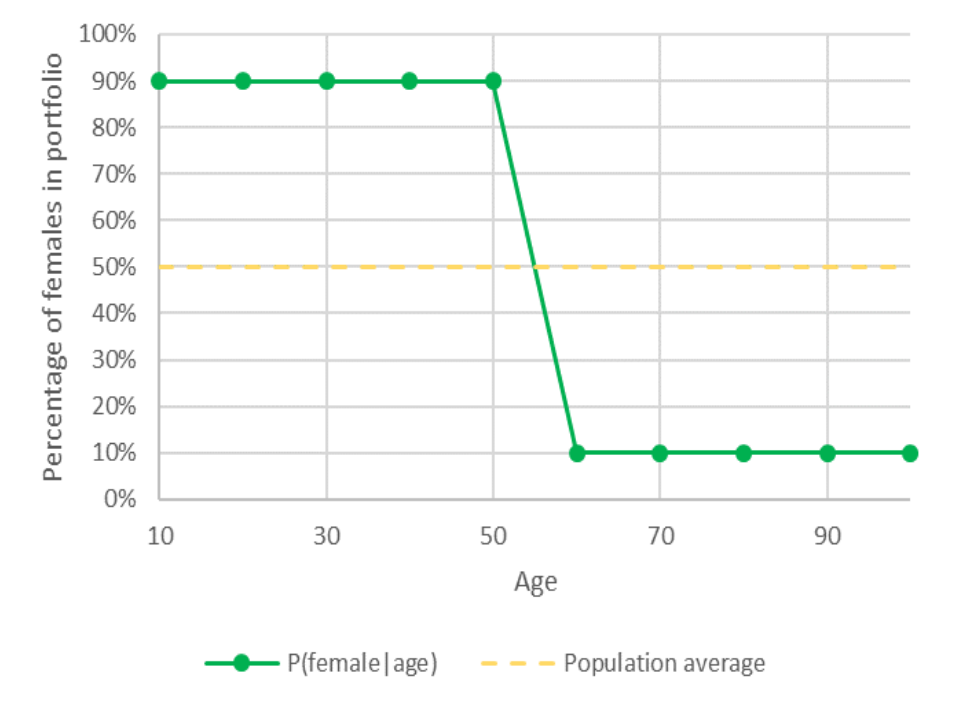
- **Insurance pricing models often take the form of best estimates plus a risk margin.**
- **Best estimates are usually defined as conditional expectations. Define:**
  - $Claims\ costs = Y$
  - $Non-discriminatory\ covariates = X$
  - $Discriminatory\ covariates = D$
- **Best estimate prices take account of both X and D:**

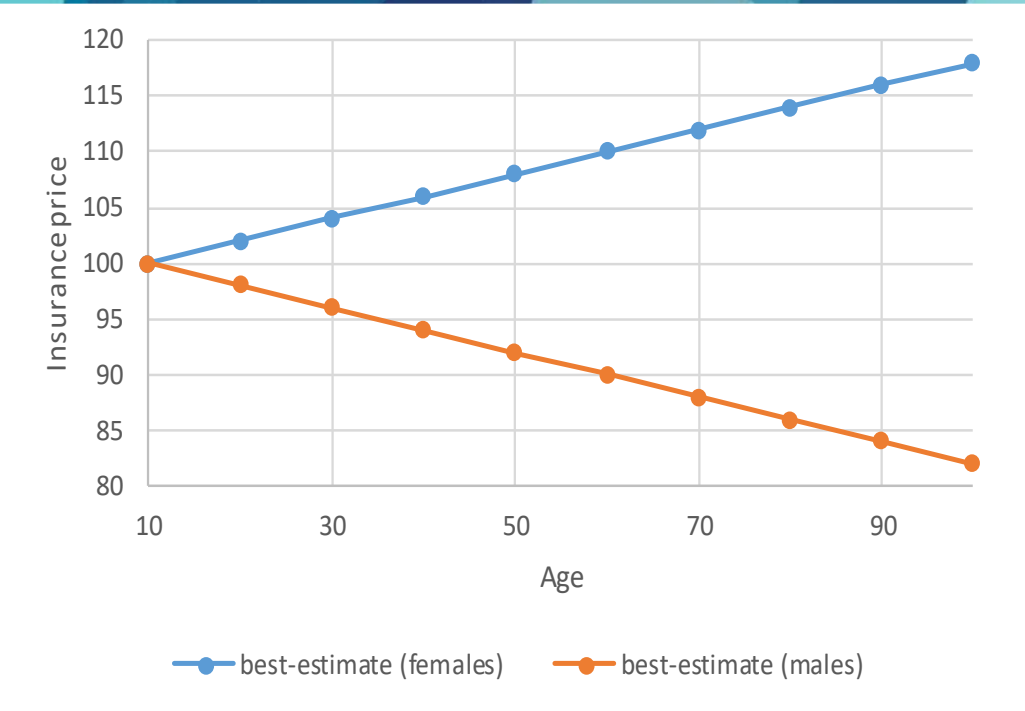
$$u(X,D) = E[Y|X,D]$$

- **For complex lines of business, we approximate $E[Y|X,D]$ using a regression model**
- **$u(X,D)$ discriminates based on D**
- **A naïve approach – unawareness prices - ignores D and hopes that X and D are uncorrelated:**

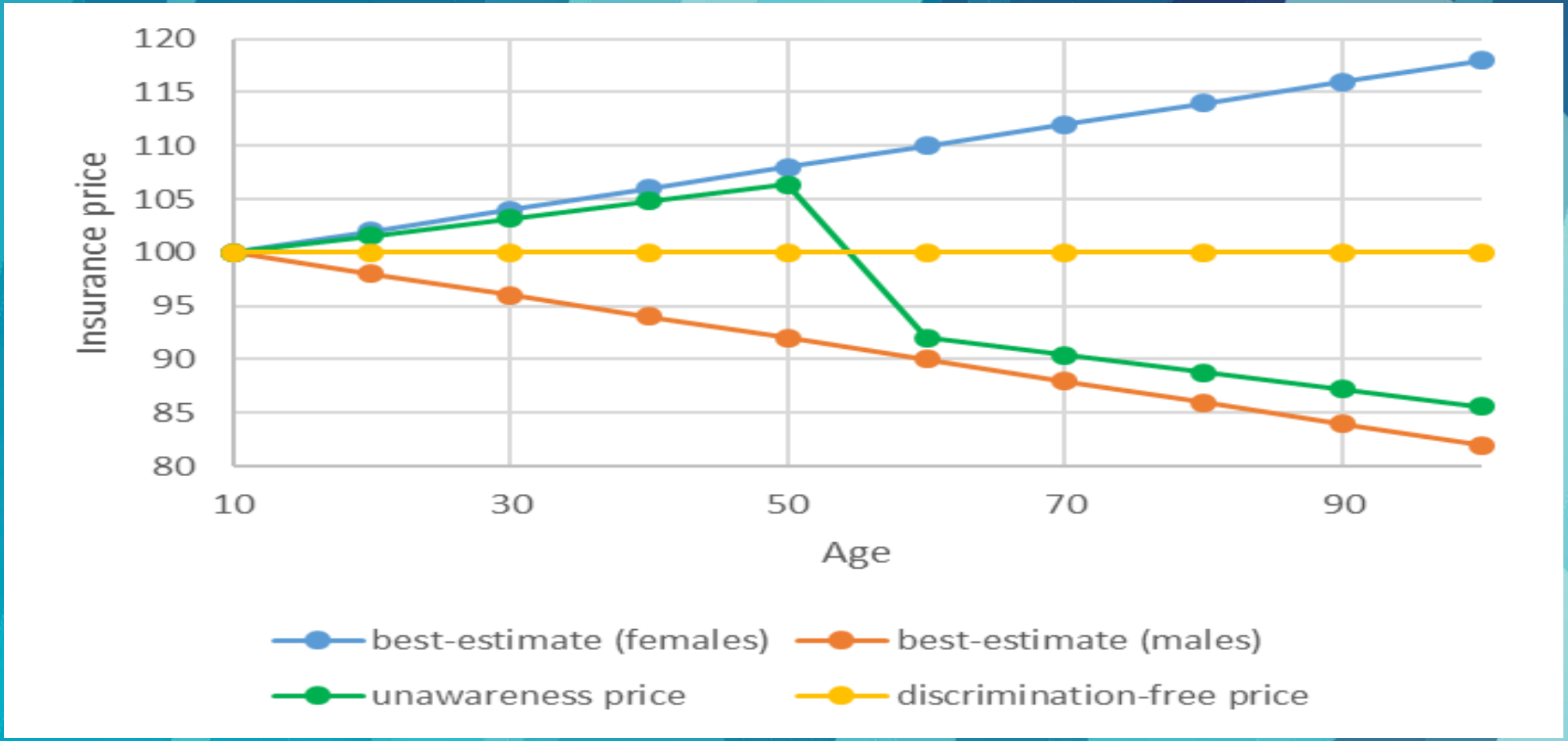
$$u(X) = E[Y|X]$$

What is the discrimination free price?





Discrimination free price?



Defining discrimination free prices

- **Intuition – we need to decouple X and D**
- **Propose a procedure whereby:**
  - Best-estimate prices (including D) are calculated using a model
  - Then take a weighted average of prices where the weights are independent of X
- **Formally:**

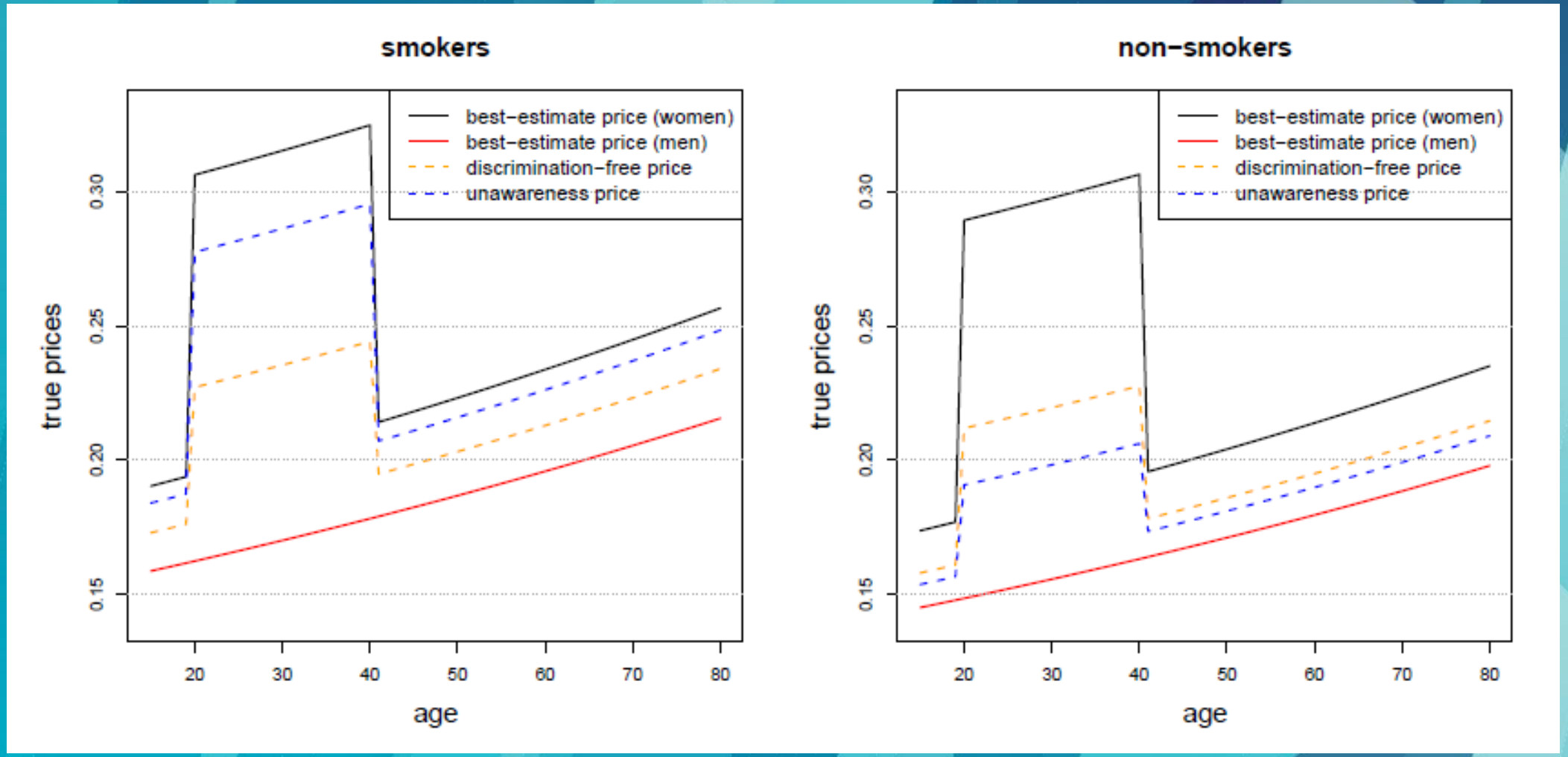
$$u^*(X) = \sum_d u(X, D = d)P(D = d)$$

- **It can be shown that:**

$$u(X, D) = \sum_d u(X, D = d)P(D = d|X)$$

- **Formal definition of $u^*(X)$ can be given using measure theory; see the paper for details**

Example: Health Insurance (Smoker ~= Woman)



$$P(D = woman|X = smoker) = 0.8$$

Conclusion

- **Deep learning can:**
  - Open new possibilities for actuarial modelling by solving difficult model specification problems, especially those involving large scale modelling problems
  - Allow new types of high frequency data to be analysed
  - Enhance the predictive power of models built by actuaries
- **To benefit fully from machine and deep learning, the goals of actuarial modelling, and implications for practice, need to be clarified**
- **The black box argument should be challenged:**
  - Learned representations from deep neural networks often have readily interpretable meaning
  - The process of learning a hierarchy of concepts can be illustrated – as shown for the LC NN model
  - Deep neural networks can be designed for interpretability (with other benefits as well)
- **More research is needed on several issues:**
  - Stability of results
  - Interpretability methods
  - Uncertainty intervals

Acknowledgements

- Mario Wüthrich
- Nicolai von Rummell
- Data Science working group of the SAA

Appendix – Other Techniques

- **Dropout (Srivastava, Hinton, Krizhevsky et al. 2014)**
  used to regularize NNs, can be combined with L1 or L2 regularizers
- **Batchnorm (Ioffe and Szegedy 2015)**
  technique used to make NNs easier to optimize and also provides a regularization effect
- **Attention (Bahdanau, Cho and Bengio 2014)**
  allows networks to choose most relevant parts of a representation
- **Generative Adversarial Models (GANs) (Goodfellow, Pouget-Abadie, Mirza et al. 2014)**
  Game between two NNs, whereby a generator network produces output that tries to trick a discriminator network.
  Useful for generative modelling, but other interesting applications such as BiGAN (Donahue, Krähenbühl and Darrell 2016)
- **Variational autoencoders (VAEs) (Kingma and Welling 2013)**
  Autoencoder with distributional assumptions made on codes
- **Neural Network Architecture Search (NNAS)**
  Techniques used to design NNs automatically
- **Pruning**
  New technique that takes a trained NN and tries to reduce redundancy (number of layers/parameters) while maintaining performance
  Part of Tensorflow 2 API

References

- **See** https://gist.github.com/RonRichman/655cca0dd79afcd20b33d3131c537414