

Statistical Computing

Michael Mayer

March 2023



Statistical Computing: What will we do?

Chapters

1. R in Action
2. Statistical Inference
3. Linear Models
4. Model Selection and Validation
5. Trees
6. Neural Nets

Remarks

- ▶ Chapters 3 to 6:
Statistical ML in Action
- ▶ Two weeks per chapter
- ▶ Exercises at end of chapter notes

Neural Nets

Outline

- ▶ Understanding Neural Nets
- ▶ Practical Considerations
- ▶ Extended examples

Neural Nets

- ▶ Around since the 1950ies
- ▶ Underwent different development steps, e.g.
 - ▶ use of backpropagation (Werbos, 1974)
 - ▶ GPUs (2009, ImageNet 2012)
- ▶ Black Box
- ▶ TensorFlow/Keras, PyTorch

“Swiss Army Knife” among ML Algorithms

Can fit linear models

**Learn interactions
and non-linear terms**

>1 Responses possible

**Flexible and mixed
in- and output dimensions**

Fit data larger than RAM

Non-linear **Learn «online»**
dimension reduction

**Sequential and spatial
in- and output**

Flexible loss functions

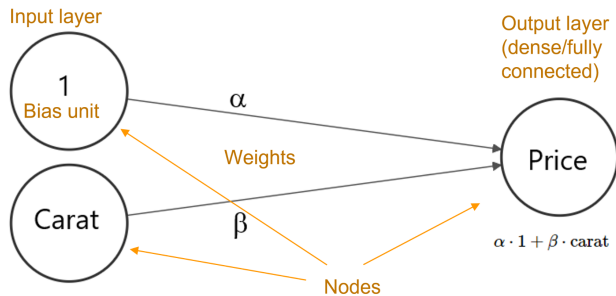
Understanding Neural Nets in three Steps

1. Linear regression as neural net
2. Hidden layers
3. Activation functions

Using **diamonds** data

Step 1: Linear Regression as Neural Net

- ▶ $\mathbb{E}(\text{price}) = \alpha + \beta \cdot \text{carat}$
- ▶ OLS
 $\hat{\alpha} \approx -2256, \hat{\beta} \approx 7756$
- ▶ Represented as neural network graph



Example

The Optimization Algorithm

Mini-batch gradient descent with backpropagation

Notation: Neural net f_β ; its total loss on data D and loss function L :

$$Q(f_\beta, D) = \sum_{(y_i, \mathbf{x}_i) \in D} L(y_i, f_\beta(\mathbf{x}_i))$$

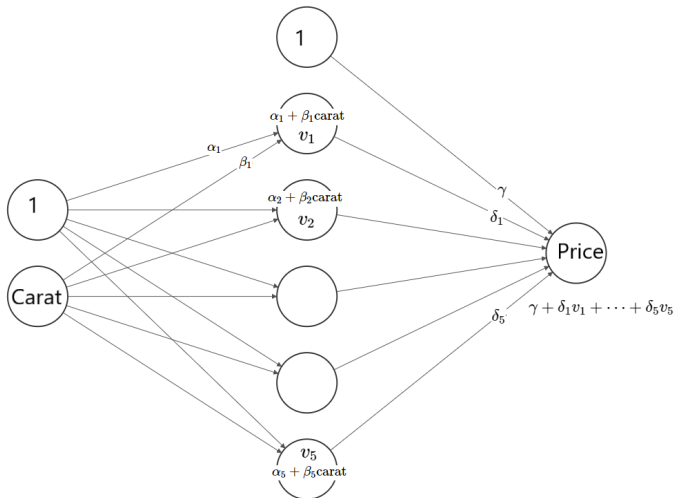
1. Init: Randomly initialize parameter vector β by $\hat{\beta}$
2. Forward: Calculate $Q(f_{\hat{\beta}}, D_{\text{batch}})$ on **batch**
3. Backprop: Modify $\hat{\beta}$ to improve $Q(f_{\hat{\beta}}, D_{\text{batch}})$
 - 3.1 Calculate partial derivatives $\nabla \hat{\beta} = \frac{\partial Q(f_\beta, D_{\text{batch}})}{\partial \beta} \big|_{\beta=\hat{\beta}}$ using backprop (=?)
 - 3.2 Gradient descent: Move slightly into right direction: $\hat{\beta} \leftarrow \hat{\beta} - \lambda \cdot \nabla \hat{\beta}$
4. Repeat Steps 2 and 3 until one **epoch** is over
5. Repeat Step 4 until some stopping criterion triggers

SGD? Local minima?

Step 2: Hidden Layers

- ▶ Add **hidden layers** for more parameters (= flexibility)
- ▶ Their nodes are latent/implicit variables
- ▶ Representational learning
- ▶ **Encoding?**
- ▶ **Deep** neural net?

Example



Step 3: Activation Functions

Non-linear transformations σ of node values necessary!

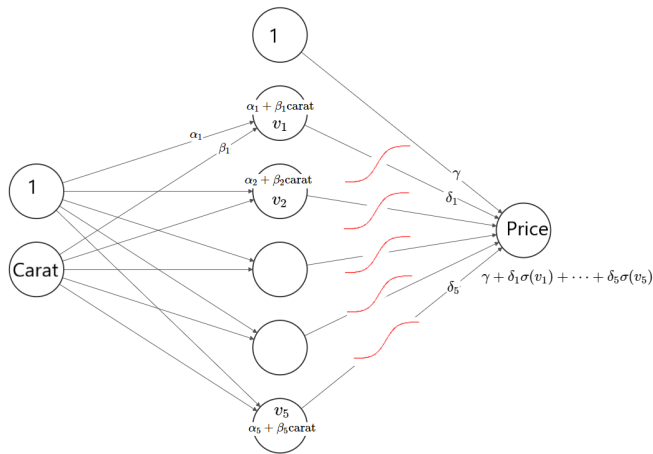
- ▶ tanh: $\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- ▶ ReLU: $\sigma(x) = \max(0, x)$



Two purposes

- ▶ Imply interactions and non-linear terms
- ▶ Inverse link as in GLMs

Example



Practical Considerations

**Validation and tuning
of main parameters**

Callbacks

**Overfitting and
regularization**

Input standardization

Missing values

Types of layers

Optimizer

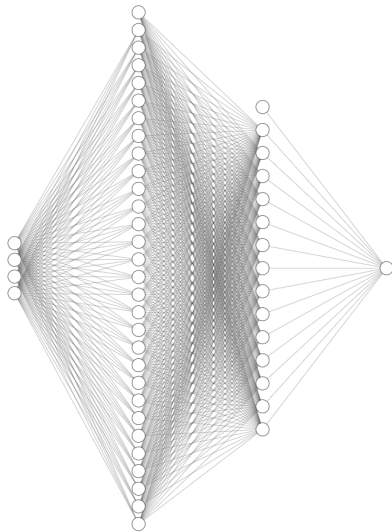
**Choosing the
architecture**

Categorical input

Interpretation

**Custom losses and
evaluation metrics**

Example: Diamonds



Excursion: Model-Agnostic Importance Measure

Permutation importance of feature $X^{(j)}$, data D , and performance measure S :

$$\text{PVI}(j, D) = S(\hat{f}, D^{(j)}) - S(\hat{f}, D)$$

- ▶ $D^{(j)}$ is version of D with randomly permuted values in j -th feature column
- ▶ Read: How much S worsens after shuffling column j ?
The larger, the more important. If 0, feature is unimportant
- ▶ Computationally cheap \rightarrow repeat m times
- ▶ Model is never refitted
- ▶ Training or test data?

Example

Embeddings

Represent unordered categorical X with K levels by $m \ll K$ numeric features

Embedding layer

- ▶ X integer encoded
- ▶ Dummy matrix \tilde{X} with K columns
- ▶ Multiply \tilde{X} with $(K \times m)$ matrix β
- ▶ Embedding matrix β estimated like other parameters
- ▶ Trick: $\tilde{X}\beta$ is calculated via index slicing from X and β
→ \tilde{X} is never materialized
- ▶ Think: $X_1 = j \rightarrow$ first row of $X\beta$ equals j -th row of β etc.

Example

Taxi trips

Excursion: Analysis Scheme X

$T(Y)$: quantity of interest

Steps

1. Calculate $T(Y)$ on the full data
2. Calculate $T(Y)$ stratified by covariates $X^{(j)} \rightarrow$ bivariate associations
3. Accompany Step 2 by ML model $T(Y | \mathbf{x}) \approx f(\mathbf{x}) \rightarrow$ multivariate associations
 - ▶ Model performance
 - ▶ Variable importance \rightarrow sort results of Step 2
 - ▶ PDP (or similar) for each $X^{(j)} \leftrightarrow$ Step 2

Example

Comparison of ML Algorithms

Aspect	GLM	Neural Net	Decision Tree	Boosting	Random Forest	k-Nearest Neighbour
Scalable	😍	😍	😊	😊	😐	😞
Easy to tune	😐	😐	😐	😐	😊	😐
Flexible losses	😊	😍	😊	😊	😐	😐
Regularization	✓	✓	✓	✓	✓	✓
Case weights	✓	✓	✓	✓	✓	✓
Missing input allowed	😞	😞	✓	✓	😞	😞
Interpretation	😍	😐	😍	😐	😐	😐
Space on disk	😍	😍	😍	😊	😞	😞
Birth date (approx.)	1972 (Nelder & Wedderburn)	1974 Backprop (Werbos)	1984 (Breiman et al.)	1990 (Schapire)	2001 (Breiman)	1951 (Fix & Hodges)