Introduction to Machine Learning

Michael Mayer

October 2022

Table of Contents

Data Science is 90% data preparation.
This lecture is about the remaining 10%.

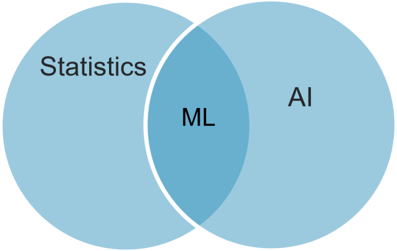
About Michael

- Pricing actuary at Swiss Mobiliar
- Lecturer at ETH Zurich
- PhD in statistics
- $(M + C)^2$ Blog: `https://lorentzen.ch/`
- `https://github.com/mayer79`

What is Machine Learning (ML)?

Collection of statistical algorithms used to

1. predict things (supervised ML) or to
2. investigate data structure (unsupervised ML).



This lecture is on supervised ML

- Regression
- Classification

Lecture Overview

Chapters

1. Basics and Linear Models
2. Model Selection and Validation
3. Trees
4. Neural Nets

Material

`https://github.com/mayer79/ml_lecture`

Table of Contents

Before Modeling

- Organization of data?
- Data types?
- Descriptive analysis

Structure of diamonds data

price	carat	color	cut	clarity
326	0.23	E	Ideal	SI2
326	0.21	E	Premium	SI1
327	0.23	E	Good	VS1
334	0.29	I	Premium	VS2
335	0.31	J	Good	SI2
336	0.24	J	Very Good	VVS2

Example

Statistical Models

Setting

- Approximate response variable Y by function f of covariates $X_1, \ldots, X_m$

$$Y \approx f(X_1, \ldots, X_m)$$

- Estimate unknown f from data by $\hat{f}$.
- Use $\hat{f}$ for prediction and to investigate relationships.
- Postulate model equation $\mathbb{E}(Y) = f(X_1, \ldots, X_m)$
  (we are often interested in means).

Remark

Other terms for response and covariates?

Linear Regression

- Postulate
$$\mathbb{E}(Y) = f(X_1, \ldots, X_m) = \beta_0 + \beta_1 X_1 + \cdots + \beta_m X_m$$
- Interpretation of coefficients β_j? Ceteris Paribus!
- Optimal $\hat{\beta}_j$? Minimize sum of squared errors/residuals
$$\sum_{i=1}^{n} (\underbrace{y_i - \hat{y}_i}_{\text{Residual}})^2$$
- $\hat{y} = \hat{f}(\ldots)$ are predictions

Example

Simple linear regression: $\mathbb{E}(Y) = \alpha = \beta X$

Aspects of Model Quality

Predictive performance

- Absolute performance
  - MSE $= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
  - Root-MSE (RMSE)
  - Loss and objective functions
- Relative performance
  - R-squared

Validity of assumptions

- Model equation is correct
- Normal linear model

$$Y = f(\ldots) + \varepsilon$$

$$\varepsilon \sim N(0, \sigma^2)$$

Example

Typical Problems

Missing values

Outliers

Overfitting

Collinearity

Categorical Covariates

- One-Hot-Encoding
- Dummy coding
- Integer encoding
- Interpretation?

Example

Example of One-Hot-Encoding

color	D	E	F	G	H	I	J
E	0	1	0	0	0	0	0
E	0	1	0	0	0	0	0
E	0	1	0	0	0	0	0
I	0	0	0	0	0	1	0
J	0	0	0	0	0	0	1
J	0	0	0	0	0	0	1
I	0	0	0	0	0	1	0
H	0	0	0	0	1	0	0
E	0	1	0	0	0	0	0
H	0	0	0	0	1	0	0

Linear Regression is Flexible

1. Non-linear terms
2. Interactions
3. Transformations like logarithms

These elements are essential but tricky!

Non-Linear Terms

Deal with non-linear associations to Y?

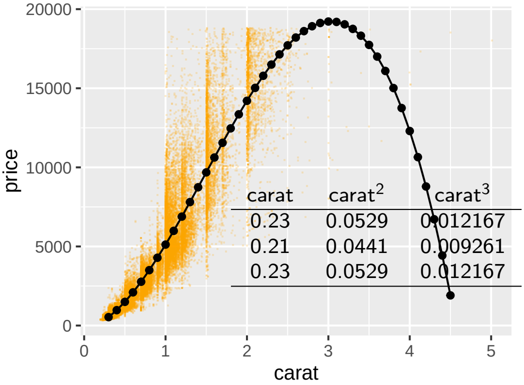
$\rightarrow$ invest more parameters

1. Polynomial terms
   - E.g., cubic regression

     $$\mathbb{E}(Y) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

   - Don't extrapolate!
2. Regression splines

Cubic terms for carat



Use systematic predictions

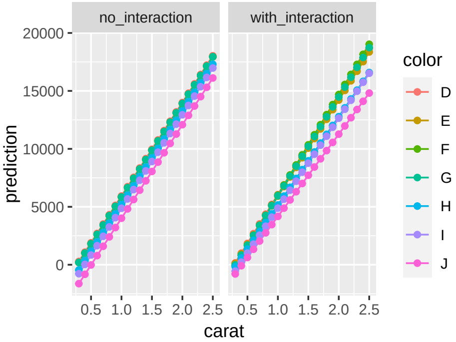
Interactions

- Additivity of effects not always realistic

$$\mathbb{E}(Y) = \beta + \beta_1 X_1 + \cdots + \beta_m X_m$$

- Adding interaction terms brings necessary flexibility $\rightarrow$ more parameters
- Interaction between features X and Z
  - For categorical Z, effects of X are calculated by level of Z
  - Like separate models per level of Z

Carat and color



Transformations of Covariates

Examples

- Dummy variables for categoricals
- Decorrelation
- Logarithms against outliers

Effects are interpreted for transformed covariates

Logarithmic Covariates

- $\mathbb{E}(Y) = \alpha + \beta \log(X)$
- Properties of logarithm allow interpretation for original covariate
- "A 1% increase in X leads to an increase in $\mathbb{E}(Y)$ of about $\beta/100$" ($\rightarrow$ lecture notes)

Example

Logarithmic Responses

What would happen for logarithmic response?

$$\mathbb{E}(\log(Y)) = \alpha + \beta X$$
$$\implies \log(\mathbb{E}(Y)) = \alpha + \beta X?$$

The implication is wrong (why?) $\rightarrow$ biased predictions and motivation for GLMs

- "A one-point increase in X is associated with a relative increase in $\mathbb{E}(Y)$ of $100\%(e^{\beta} - 1) \approx 100\%\beta$" $\rightarrow$ lecture notes
- Relative effects / multiplicative model

Examples

- Logarithmic response
- Response and covariate with log

Example: Realistic Model for Diamond Prices

- Response: log(price)
- Covariates: log(carat), color, cut and clarity

Exercise on Linear Regression

- Run last model without any logarithm
- Interpret the results
- Does model make sense from practical perspective?

Generalized Linear Model (GLM)

(One) extension of linear regression

Model equation

Two equivalent formulations

$$\begin{aligned} g(\mathbb{E}(Y)) &= \beta_0 + \beta_1 X_1 + \cdots + \beta_m X_m \\ \mathbb{E}(Y) &= g^{-1}(\beta_0 + \beta_1 X_1 + \cdots + \beta_m X_m) \end{aligned}$$

Components

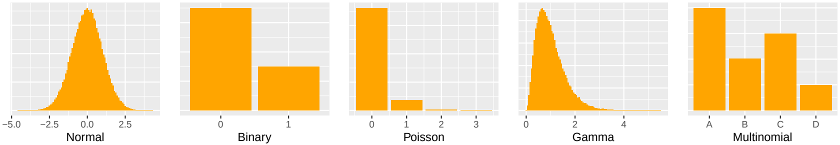
- Linear function/predictor
- Link function g to map $\mathbb{E}(Y)$ to linear scale
- Distribution of Y conditional on covariates

Typical GLMs

Regression	Distribution	Range of Y	Natural link	Typical link	Loss function
Linear	Normal	$(-\infty, \infty)$	Identity	Identity	Squared error: $(y-\hat{y})^2$
Logistic	Binary	$\{0,1\}$	logit	logit	Binary cross-entropy: $-(y\log(\hat{y}) + (1-y)\log(1-\hat{y}))$
Poisson	Poisson	$[0, \infty)$	log	log	Unit Poisson deviance: $2(y\log(y/\hat{y}) - (y-\hat{y}))$
Gamma	Gamma	$(0, \infty)$	1/x	log	Unit Gamma deviance: $2((y-\hat{y})/\hat{y} - \log(y/\hat{y}))$
Multinomial	Multinomial	$\{C_1, \ldots, C_m\}$	mlogit	mlogit	Categorical cross-entropy: $-\sum_{j=1}^m 1(y=C_j)\log(\hat{y}_j)$

- Predictions?
- Log-Link?
- For binary Y: $\mathbb{E}(Y) = P(Y=1) = p$
- Tweedie-GLM
- MSE $\rightarrow$ Deviance
- Same losses for ML

Empirical distributions of 100'000 values



Why GLM, not Linear Regression?

Linearity assumption not always realistic

1. Binary Y:
   Jump from 0.5 to 0.6 success probability less impressive than from 0.89 to 0.99
2. Count Y: Jump from $\mathbb{E}(Y)$ of 2 to 3 less impressive than from 0.1 to 1.1.
3. Right-skewed Y:
   Jump from 1 Mio to 1.1 Mio deemed larger than from 2 Mio to 2.1 Mio.

Logarithmic Y not possible in the first two cases

GLM solves problem by suitable link g

Further advantages?

Interpretation of Effects guided by Link

- Identity link: like linear regression
- Log link: like linear regression with log response
  - $\rightarrow$ multiplicative model for response
  - Now in mathematically sound way
- Logit link $\rightarrow$ additive model for logit(p)
  - $\text{logit}(p) = \log(\text{odds}(p)) = \log\left(\frac{p}{1-p}\right)$
  - Remember: $p = P(Y = 1) = \mathbb{E}(Y)$
  - $\rightarrow$ multiplicative model for odds(p)
  - Coefficients $e^{\beta} - 1 \approx 100\%\beta$ interpreted as odds ratios.

Examples with Insurance Claim Data

1. Poisson regression for claim counts
2. Binary logistic regression for claim (yes/no)

Exercise on GLM

- Fit Gamma regression with log-link to explain diamond prices by log(carat), color, cut and clarity.
- Compare the coefficients with corresponding linear regression for log(price).
- Evaluate the relative prediction bias on USD scale.

Table of Contents

Two Questions

- "How good is our model?"
- "Which model to choose among alternatives?"

Problem and solution

- "Insample" performance is biased
- Overfitting should not be rewarded
- Use data splitting to get fair results

Remarks

- Performance measure (evaluation metric) versus loss function?
- Confusion matrix?

Excursion: k-Nearest-Neighbour (k-NN)

- Alternative to linear model
- How does it work?
- Classification and Regression
- Standardization?

Example

Simple Validation

- Insample, 1-NN would win any comparison!?
- Split data into training and validation set, e.g., 80%/20%
- Use performance on validation set to make decisions
(choose models, choose parameters like k)
- Measure amount of overfitting ($=$ optimism)?

Example

Cross-Validation (CV)

- Simple validation is neither economic nor robust, except for large data
- Better: k-fold cross-validation

Algorithm

1. Split the data into k pieces $D = \{D_1, \ldots, D_k\}$ called "folds". Typical values for k are five or ten.
2. Set aside one of the pieces (D_j) for validation.
3. Fit the model on all other pieces, i.e., on $D \setminus D_j$.
4. Calculate performance on the validation data D_j.
5. Repeat Steps 2–4 until each piece was used for validation once.
6. The average of the k model performances yields the CV performance of the model.

Remarks

- How to choose and fit best/final model?
- What means «best»?
- Repeated CV?

Example

Hyperparameter Tuning

- Choosing k in k-NN is example of "hyperparameter tuning"
- Algorithms with more than 1 hyperparameter?
- Grid Search CV
- Randomized Search CV

Test Data and Final Workflow

Problematic consequence of model tuning?

- Overfitting on validation data or on CV!
- Performance of final model? $\rightarrow$ Test data

Workflow A

1. Split data into train/valid/test,
   e.g., by ratios 70%/20%/10%.
2. Train different models on training data and assess their performance on the validation data. Choose best model, retrain on combination of training and validation data and call it "final model".
3. Assess performance of final model on test data.

Workflow B

1. Split data into train/test,
   e.g., by ratios 90%/10%.
2. Evaluate and tune different models by k-fold CV on the training data. Choose best model, retrain on full training data.
3. Assess performance of final model on test data.

Example of Workflow B

When Test $=$ Validation?

Using Independent Partitions is Essential

Random splits

Grouped splits

Time-Series splits

Stratified splits

Exercises with Diamonds

1. As alternative to grouped splitting, deduplicate (why?) the diamonds data on "price" and all covariates and repeat the last example. Do the results change? Which results do you trust more?

2. Use CV to select the best polynomial degree of log(carat) in the Gamma GLM with log-link (with additional covariates color, cut, clarity). Evaluate on a test data.

Table of Contents

Trees

Decision Trees

Random Forests

Gradient Boosting

Decision Trees

- Simple
- Easy to interpret
- Decision trees are like wolves:
  Weak alone, strong together
- Around since 1984
  (Breiman, Friedman)

https://images.pexels.com/photos/3732527/pexels-photo-3732527.jpeg

How do Decision Trees Work?

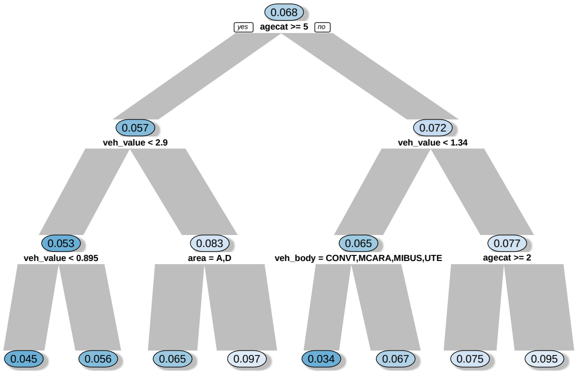
Algorithm

1. Split: find best "yes/no" question on best feature
2. Apply Step 1 recursively

Notes

- "best" regarding average loss improvement
- Typical losses: squared error, log loss/cross-entropy/information or Gini
- Predictions?

Example



The tree does a headstand

Properties of Decision Trees

Outliers

Greedy

Missing Values

Categorical covariates

Extrapolation

Interactions

Properties are inherited to groups/ensembles of decision trees

Random Forests

- Group of many decision trees
- Perform very well
- Black Box
- Around since 2001 (Breiman)

https://images.pexels.com/photos/1459534/pexels-photo-1459534.jpeg

How do Random Forests Work?

If we train 500 trees, how can we make sure they are different?

Idea: Introduce two sources of randomness

1. Each split considers random selection of features only.
2. Each tree is trained on bootstrap sample.

Predictions?

Comments on Random Forests

Number of trees?

Diversification!

Insample performance

Out-of-Bag (OOB)

Parameter tuning

Example

Interpreting a Black Box is Important

Minimum interpretation of model?

1. Performance
2. Variable importance
3. Effects $\rightarrow$ e.g. partial dependence plots

Example

Exercises on Random Forests

1. In our diamonds random forest, replace carat by log(carat). Do the results change?
2. Fit a (probability) random forest on the claims data to predict claim probability by `veh_value`, `veh_body`, `veh_age`, `gender`, `area`, and `agecat`.
   - Choose tree depth by OOB performance or CV.
   - Evaluate the model on an independent test dataset.
   - Interpret the results.

Gradient Boosted Trees

- Group of many decision trees
- Perform very well
- Black Box
- Around since 2001 (Friedman)
- XGBoost, LightGBM, CatBoost

Like random forests

https://www.gormanalysis.com/blog/gradient-boosting-explained/

How does Gradient Boosting Work?

Algorithm

1. Fit simple model
   (often a small decision tree).
2. Add another simple model to
   correct errors of first model.
3. Repeat Step 2 until stopping
   criterion triggers.

Remarks

- Completely different from random forest.
- Predictions are found by combining predictions of all simple models (like random forest).
- Flexible regarding loss function.

Example XGBoost

Parameter Tuning is Essential

1. Number of boosting rounds/trees
   $\rightarrow$ find by early stopping (validation/CV)
2. Learning rate
   $\rightarrow$ to get reasonable number of rounds
3. Regularization
   - Tree depth, number of leaves, loss penalties, etc.
   - $\rightarrow$ Grid/Randomized search and iterate

Why not one big grid search on all parameters?

Example XGBoost

Exercises on Gradient Boosting

1. Study online documentation of XGBoost to make the last model monotonically increasing in carat. Check the resulting partial dependence plot.

2. Develop a strong XGBoost model on the claims data to predict claim probability by `veh_value`, `veh_body`, `veh_age`, `gender`, `area`, and `agecat`.
   - Use `"binary:logistic"` as objective.
   - Use `"logloss"` as evaluation metric.
   - Use a clean CV/test strategy.
   - Interpret the results.

Table of Contents

Neural Nets

- Around since the 1950ies
- Underwent different development steps, e.g.
  - use of backpropagation
  - GPUs
- Black Box
- TensorFlow/Keras, PyTorch

“Swiss Army Knife” among ML Algorithms

Can fit linear models

Fit data larger than RAM

Learn interactions
and non-linear terms

Learn «online»

Non-linear
dimension reduction

>1 Responses possible

Sequential and spatial
in- and output

Flexible and mixed
in- and output dimensions

Flexible loss functions

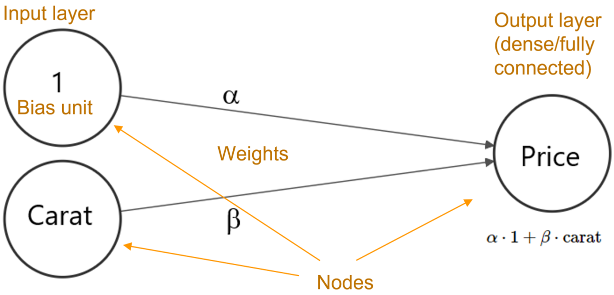
Understanding Neural Nets in three Steps

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

Using diamonds data

Step 1: Linear Regression as Neural Net

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



Example

The Optimization Algorithm

"Mini-batch gradient descent with backpropagation"

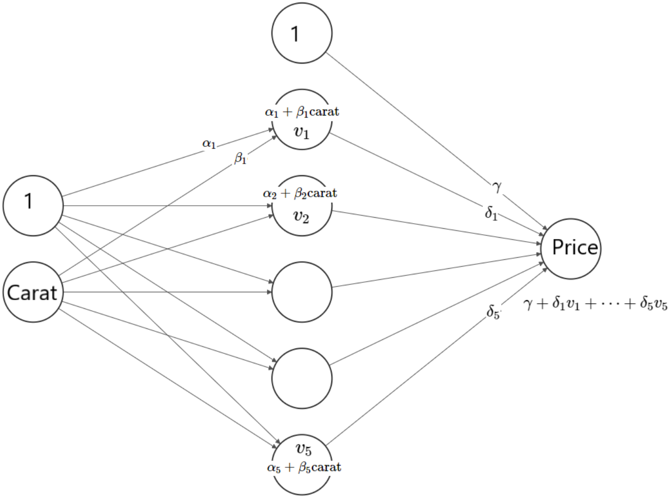
1. Init step: Randomly initiate parameters.
2. Forward step: Use parameters to calculate predictions of batch.
3. Backprop step: Evaluate average loss (e.g. MSE) of batch. Change parameters systematically to make it smaller.
4. Repeat Steps 2 and 3 until one epoch is over.
5. Repeat Step 4 until some stopping criterion triggers.

SGD? Gradients? Local minima?

Step 2: Hidden Layers

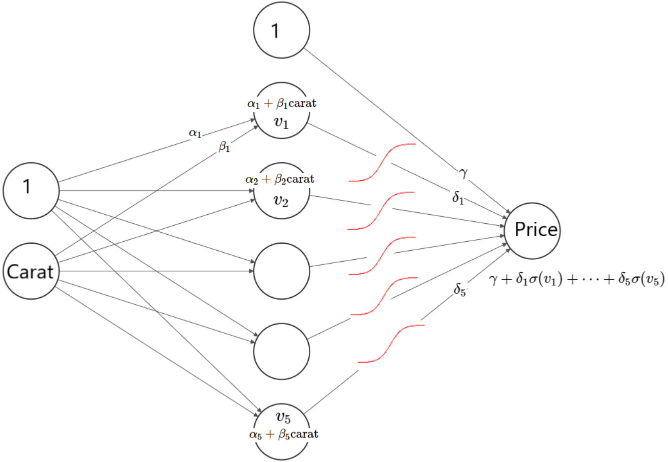
- Add hidden layers for more parameters ($=$ flexibility)
- Their nodes are latent/implicit variables
- Representational learning
- Encoding?
- Deep neural net?

Example



Step 3: Activation Functions

Non-linear transformations
σ of node values necessary!



Two purposes

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

Example

Practical Considerations

Validation and tuning of main parameters

Input standardization

Missing values

Categorical input

Callbacks

Types of layers

Interpretation

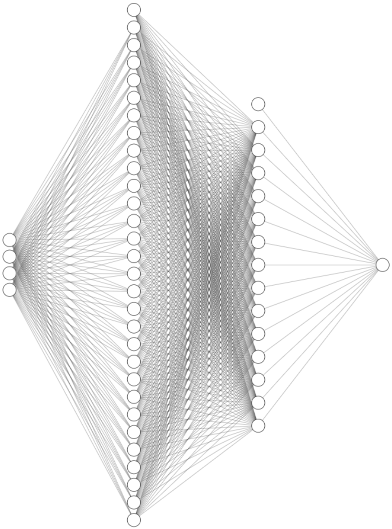
Optimizer

Overfitting and regularization

Choosing the architecture

Custom losses and evaluation metrics

Example: diamonds



Exercises on Neural Nets

1. Fit diamond prices by minimizing Gamma deviance with log-link
   (custom loss as in lecture notes; log-link means "exponential" output activation)
   - Tune model by simple validation.
   - Evaluate final model (for simplicity) on validation data.
   - Interpret final model.

   Hints: Use smaller learning rate and replace "relu" by "tanh". Furthermore, the response is to be transformed from int to float.

2. Study either the optional claims data example in the lecture notes or build your own neural net, predicting claim yes/no.

Table of Contents

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)

"Michael's Analysis Scheme X"

1. Take any property $T(Y)$ of key interest (churn rate, claims frequency, loss ratio, etc.) and calculate its value on the full dataset.
2. Do a descriptive analysis of $T(Y \mid X_j)$ for a couple of covariates to study the bivariate association between Y and each X_j separately.
3. Accompany Step 2 by ML model $T(Y \mid X_1, \dots, X_p) \approx f(X_1, \dots, X_p)$
   - Check its performance.
   - Study variable importance and use it to sort the results of Step 2.
   - Study partial (or SHAP) dependence plots for each X_j and compare them with the associations from Step 2.

What did you learn from Step 3?