Session 118: Machine Learning Theory and Real-World Considerations

Machine Learning Theory and Real-World Considerations

Presented at SOA - 2019

October 2, 2019

Disclaimers

PartnerRe

The thoughts are mine and no one wants to take credit.

The following presentation is for general information, education and discussion purposes only, in connection with the SOA Conference 2019. Any views or opinions expressed are those of the presenters alone. They do not constitute legal or professional advice; and do not necessarily reflect, in whole or in part, any corporate position, opinion or view of PartnerRe or its affiliates, or a corporate endorsement, position or preference with respect to any issue or area covered in the presentation.

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the participants individually and, unless expressly stated to the contrary, are not the opinion or position of the Society of Actuaries, its cosponsors or its committees. The Society of Actuaries does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented. Attendees should note that the sessions are audio-recorded and may be published in various media, including print, audio and video formats without further notice.

Presenters

PartnerRe

Who are we?

Thomas D. Fletcher, PhD, ChFC®
VP Data Analytics – North America
PartnerRe Analytics

- Background in Statistics and I/O Psychology
- Insurance industry since 2008
- P&C, Surety, Management Liability, Life/Health, Financial Services
- Projects span entire value chain (markets, customers, distribution, … risk assessment, … claims management)

Harrison Jones, ASA
Manager | Actuarial, Rewards & Analytics
Deloitte

- Held Data Scientist / Actuarial positions for past seven years
- Predictive modelling projects in P&C pricing, disability insurance, and life insurance experience studies
- Other areas of work include P&C valuation, IFRS 17, and insurance database architecture

Traditional Modeling vs. Machine Learning

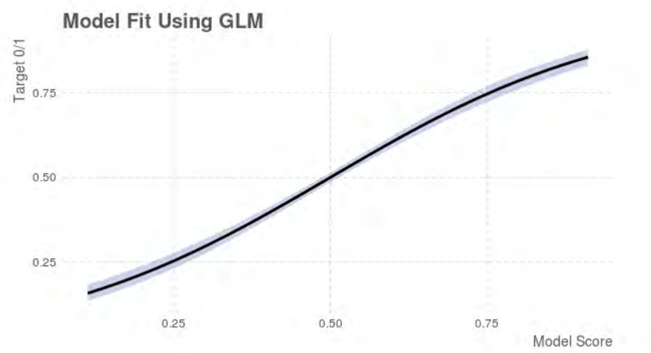
PartnerRe

Where are the fundamental differences?

Regression-based methods: (glm)

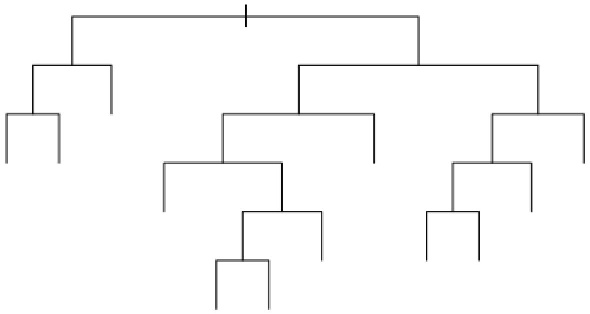
- Formula based with distributional assumptions (minimization of loss function via maximum likelihood)
- More manual lifting to prepare data, but simpler to decipher

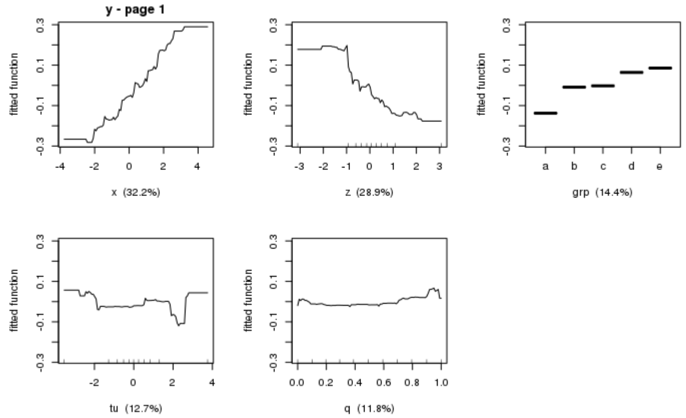
```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.071188   0.057127   1.246  0.21272
x            0.152938   0.021892   6.986 2.83e-12 ***
z           -0.167567   0.027284  -6.142 8.17e-10 ***
q            0.087370   0.070922   1.232  0.21798
grpb         0.202430   0.063983   3.164  0.00156 **
```



Tree-based methods: (cart/rpart, rf, gbm, xgboost)

- Algorithmic based with mostly non-parametric qualities (formula of a loss function ++)
- Requires more computer power – to address the cross-validation, bagging, boosting, etc. to ensure less variance due to sampling error – but, at a cost of instability across models (not every run yields identical results)
- Allegedly less effort to prep data (will *find* interesting effects in the data) …
- … but more difficult to interpret after the fact





Issues in Feature Engineering

PartnerRe

Creating your model variables with an eye towards scoring

	Traditional Modeling	Modern ML (algorithm dependent)
Missing Data	▪ Can not fit a model with NAs ▪ Can not score to model with NAs	▪ *Can* fit and score a model with NAs ▪ Often difficult to know how NA handled
Categorical Predictors	▪ Categories represented by columns (0/1) ▪ Numerous categories are problematic	▪ *Can* handle many categories (*depends*) ▪ May not observe all nominal differences
Non-linearities & Interactions	▪ Explicit specification of relationships ▪ Careful consideration of interpretation	▪ *Finds* non-linearities and interactions ▪ Difficult interpretation of relationships

MISSINGness

PartnerRe

What creates holes in your data (before and after modeling)?

No Entry Explicit NA

- Db does not have an entry – doesn't exist
- NULL may be 1, 0, or … (ask IT & users)

Variable Artefact

- During variable creation/calculation
- Division by 0 for a ratio
- NA in one component of a calculation

New Factor Level

- Factor level not present during training
- Particularly problematic in gbm in R

Out-of-range values

- Negative or large values set to 0 or NA
- Metric/unit inconsistencies (000s)

Considerations

- Properly addressed, each of the issues can be trivial
- Pattern of missingness – MCAR or systematic, %missing?
- CAUTION: !is.na could become post-dictor (and could mask other important insights) – UW asks for *test (credit check)* when something is suspicious

Addressing Missing Data

Traditional modeling (e.g., glm)

Variable Binning

- By binning into ‘buckets’ can add a ‘NA’ category
- Lose some precision, but gain flexibility
- Facilitates non-linearities as well as patterns of NA

binRS <chr>	GrpN <int>	countY <int>	PercY <dbl>
1	750	514	0.69
2	750	537	0.72
3	750	530	0.71
4	750	534	0.71
MISS	7000	3771	0.54

Imputation

- Many methods to impute NAs
- Mean, Mdn, Regression/Maximum likelihood based, …
- *Can* be controversial – depending …

y	grp	x	q	z	rs	tu
0	a	0.42408481	0.177988620	-1.3721478141	NA	-1.38659967
1	e	0.14781126	[illegible]	[illegible]	0.13031495	0.16703526
0	a	-0.59990287	0.686955123	0.0403619652	NA	0.977446

e.g. $\widehat{NA}$

e.g., μ

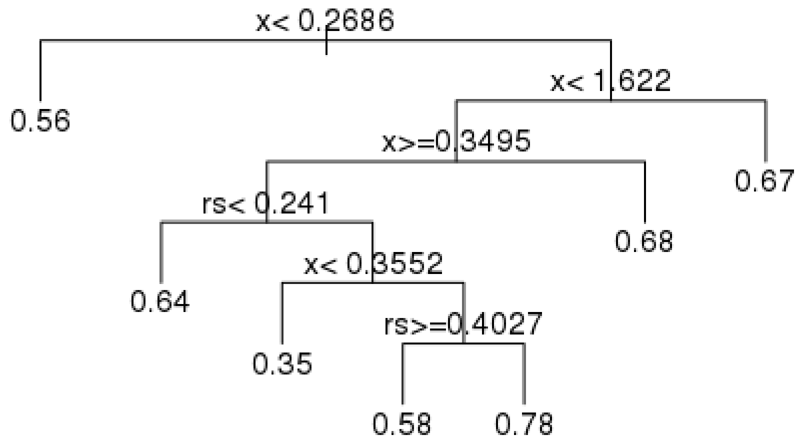
No Score

- May be ok for training models
- Can route NAs to human
- Impractical if a score is needed and %NA is large

Addressing Missing Data

PartnerRe

Tree-based methods (e.g., rpart & gbm)




```
Node number 3:
  mean=0.61533
  left son=6 (
  Primary spli
      x   < 1.622379   to the left,  improve=0.0027567700, (0 missing)
      rs  < 0.2417628  to the left,  improve=0.0009057351, (3216 missing)
```

Traditional Methods

- Methods previously described work here too
- Though, some *may* be unnecessary
- Must decide how much control you want

Missing Allowed

- Surrogate variables and majority observations
- Scores new data which may contain missing
- This does not hold for all platforms and algorithms

Careful Interpretation

- With a gbm all variables could be NAs and it scores
- May be unclear how arrived at the score given patterns of NAs
- May wish to set rules on which or how many permissible (e.g., no more than 3 or not if key variable)

	SplitVar <int>	SplitCodePred <dbl>	LeftNode <int>	RightNode <int>	MissingNode <int>
0	3	-4.303325e-01	1	5	15
1	0	0.000000e+00	2	3	4
2	-1	4.806977e-04	-1	-1	-1
3	-1	4.289089e-03	-1	-1	-1

Addressing Missing Data

Code Demo

PartnerRe

Code and sample data can be found at:

https://gitlab.com/HarrisonAtDeloitte/soa-2019

Items Covered:

- Finding missing values (Base R and `FindMissingValues()`)
- Missing value patterns (`visdat` and `naniar` packages)
- Decision trees – using surrogate splitting to avoid issues with missing values
- Ordinary Least Squares – no inherent mechanism to handling missing values (besides removing observations)
- Imputation (`simputation` package)

Categorical 'Predictors'

How to represent non-numeric data in a (*traditional*) model on the *RHS*?

Dummy/Effects Coding

- k-1 columns represent categories
- Type of coding allows for different purposes

Recode into Smaller Grps

- If hierarchical (SIC into 1,2 digits)
- Relationship to each other (clustering)
- Other relationships (e.g., regions)

Ordinal treated numerically

- First < Second < Third, but …
- Does not assume equal intervals

Multilevel Models

- Random coefficients (hierarchical linear models) can represent categories
- Out of scope for this discussion

Considerations

- Categories can be nominal (states, SIC codes, ICD codes) or ordinal (first, second, third), but are assumed to not have interval properties
- Number of levels can become unwieldy (50 states, 1000s of codes, etc.)
- CAUTION: New factor levels can create issues in scoring. Can lose information in coding into smaller groups and create ecological fallacy

Categorical: Examples and Implications

PartnerRe

Traditional modeling (e.g., glm)

Contrast Coding

- Intercept represents reference group; coefficient is difference in that level and the reference level
- Effects coding, coefficient is different in that level from overall average
- Ominibus interpretation requires model comparisons

Recoding

- Hierarchical can lose granularity quickly (ICD codes)
- Clustering can result in non-contiguous categories
- Regions may create greater heterogeneity within

Treat as Ordinal

- If not ordinal, nonsensical results (unless only 2 categories)
- Different (new) categories will be scored improperly

```
  (Intercept) grpb grpc grpd grpe
1           1    0    0    0    0
2           1    0    0    0    1
3           1    0    0    0    0
4           1    0    0    0    1
5           1    1    0    0    0
6           1    1    0    0    0
```

```
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.12441    0.04485   2.774 0.005542 **
grpb         0.21019    0.06371   3.299 0.000970 ***
grpc         0.22339    0.06407   3.487 0.000489 ***
grpd         0.33479    0.06383   5.245 1.56e-07 **
```

newCat	Avg. PopDensity	States
1	119.2	AK, AZ, CO, ID, MT, NE, NV, NM, ND, OR, SD, UT, WY
2	745.5	AL, AR, CA, GA, HI, IA, KS, KY, LA, ME, MN, MS, MO, NH, NC, OK, SC, TN, TX, VT, VA, WA, WV, WI
3	4243.8	CT, DE, FL, IL, IN, MD, MA, MI, NJ, NY, OH, PA, RI

Categorical: Examples and Implications

PartnerRe

Tree-based methods (e.g., rpart & gbm)

	GRP	numGRP
1	a	1
2	e	5
3	a	1
4	e	5
5	b	2

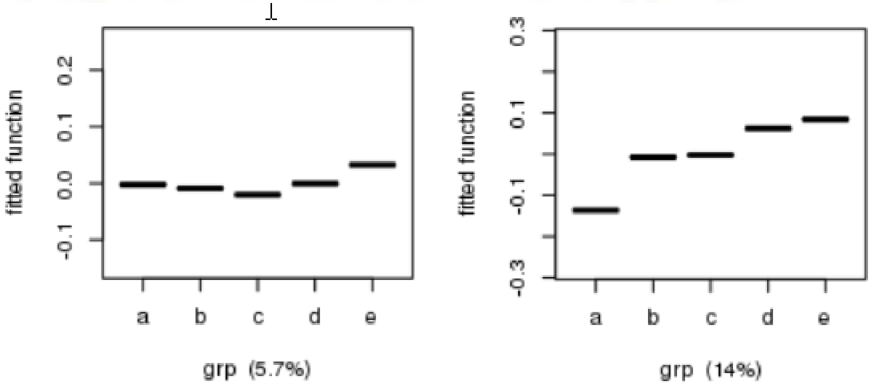
Traditional Methods

- Methods previously described work here too
- Though, some *may* be unnecessary
- Must decide how much control you want

```
if (is.ordered(x[, i])) {
  var.levels[[i]] <- levels(factor(x[, i]))
  x[, i] <- as.numeric(factor(x[, i])) - 1
  var.type[i] <- 0
}
else if (is.factor(x[, i])) {
  if (length(levels(x[, i])) > 1024)
```

Algorithm Dependent

- Implementation matters (e.g., R, Python)
- R gbm is not the same as python gbm
- xgboost not the same as gbm
- R gbm allows interpretation of importance of factor, not just levels within the factor



Careful Interpretation

- A benefit of R’s implementation of gbm is that one can interpret the factor as a whole; other algorithms often slit interpretation to the level of the factor
- Different variables’ inclusion (or hyperparameter tuning) can render different interpretations of the factor’s importance and how levels relate to target

Addressing Categorical Predictors

PartnerRe

Code Demo

Code and sample data can be found at:

https://gitlab.com/HarrisonAtDeloitte/soa-2019

Items Covered:

- Categorical variable treatment in common models
- Recoding into smaller groups
- Recoding into ordinal factors

Nonlinearities & Interactions

PartnerRe

Complexities modeling contingent relationships: “It depends …”

Nonlinearities in relationships

- X depends on itself (Age effect dampens or height, or accelerates on mortality)
- Often modeled as polynomial, but need not be ($X + X^2$)

Interactions in relationships

- X depends on some other variable
- Often modeled as a product (X*Z)
- Some interactions signal a cancelling of effect

Form of Interaction

- Not accounting for interaction may result in (directionally) incorrect model results
- Cross-over interactions can lead to Type II errors

Power & Type II errors

- Power (sample and effect size) often dampen ability to detect interactions
- Theorized interactions are rarely spurious

Considerations

- To understand Y~X relationship, explicitly modeling interactions is often necessary
- Complexities may or may not be noticeable in the data due to limitations
- CAUTION: Categoricals add another level of complexity in determining interactive relationships

Detecting Nonlinearities & Interactions

Traditional modeling (e.g., glm)

Polynomials as Representative

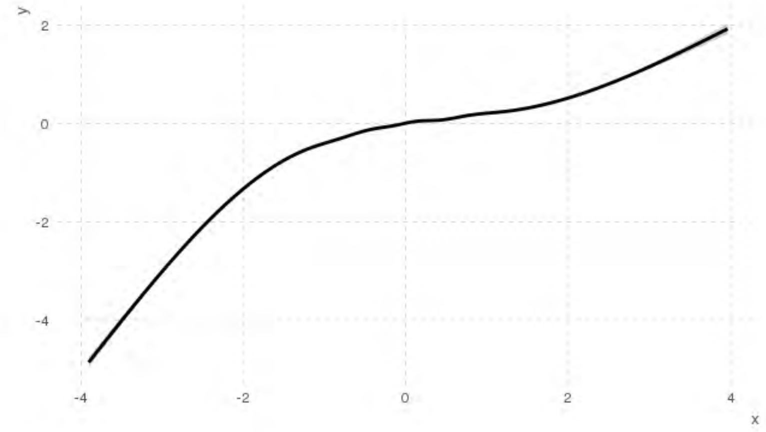
- True nonlinear relationships are rare in business, but one can model if need be (e.g., asymptotic)
- Polynomials are often effective at mimicking the effect
- Orthogonal polynomials add complexity but reduce concerns over multicollinearity (X, X^2, X^3)

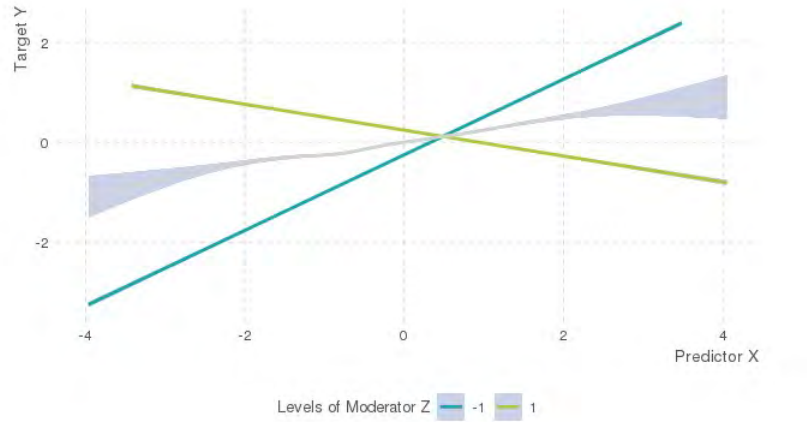
Multiplicative Variables

- Components MUST be present in model w/ interaction.
- Signs can be interpreted to understand form (+, +, -)
- Interpret graphically – always!

Categorical Interactions

- If the number of levels is small (i.e., 2), interpretation is greatly simplified (2x2 matrix of results)
- As number of levels increases, the complexity in interpretation of the output grows massively (1000s of ICD codes interacting with some contingency)





Detecting Nonlinearities & Interactions

PartnerRe

Tree-based methods (e.g., rpart & gbm)

Traditional Methods

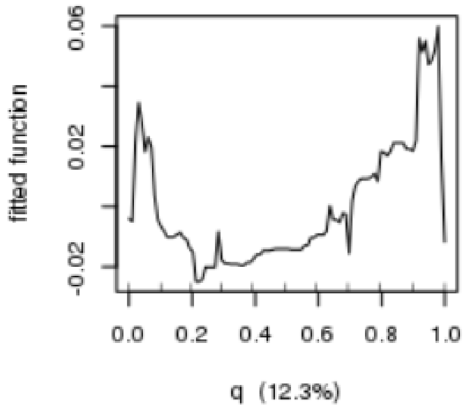
- Methods previously described work here too
- Though, some *may* be unnecessary
- Must decide how much control you want

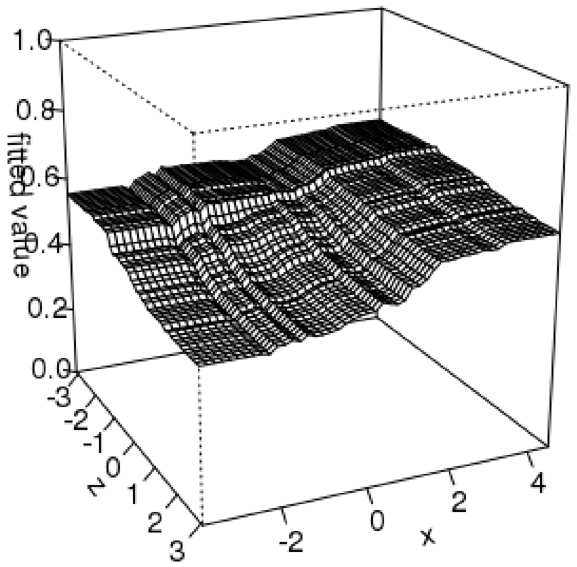
Algorithm Dependent

- Method matters (e.g., rf, gbm, rpart, xgboost)
- e.g., rf does not include all columns with each iteration
- Number and size of trees may matter (small trees may not allow for certain interactions to present)
- Interactions manifest by tree branching (x on different)

Careful Interpretation

- A benefit of R's implementation of gbm is that one can identify non-linearities via partial dependence plots and interrogate interactions with perspective plots
- Variables with key interactions tend to show higher levels of importance
- Not all interactions are detected – esp. if masking variable is present





Detecting Nonlinearities & Interactions

PartnerRe

Code Demo

Code and sample data can be found at:

https://gitlab.com/HarrisonAtDeloitte/soa-2019

Items Covered:

- Models that do / don’t automatically build non-linear predictors
- How to implement non-linear predictors in models that don’t automatically take care of it

PartnerRe