Preparing Data for Analysis Using R:
Basic through Advanced Techniques

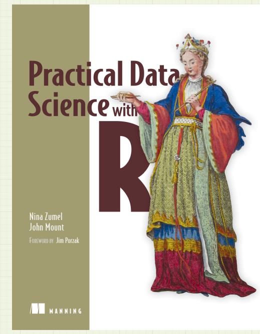
John Mount & Nina Zumel
Win-Vector, LLC

All materials: https://github.com/WinVector/PreparingDataWorkshop

1

Who I am

- Nina Zumel
- Principal Consultant at Win-Vector LLC
- One of the authors of Practical Data Science with R



2

Outline of Part 1

- Data Preparation
  - Typical data problems & possible solutions
- `vtreat`: Automating variable treatment in R
- Examples of automated variable treatment
- Conclusion

3

Throughout this workshop

- We will keep an idealized goal in mind: using machine learning to build a predictive model.
- We assume we can delegate the modeling or machine learning to a library, and take on the responsibility for data preparation and cleaning.
- Having a single ideal goal allows us to apply seemingly "ad-hoc" fixes in a principled manner.
  - We can check if our "fixes" are for good or bad.
  - We are not limited to mindlessly combining prior "name brand" procedures.

4

Data Preparation

5

Why Prepare Data at All?

- To facilitate modeling/analysis
  - Clean dirty data
  - Format data the way machine learning algorithms expect it
- Not a substitute for getting your hands dirty
  - But some issues show up again and again

6

Typical Data Problems

- “Bad” numerical values (NA, NaN, sentinel values)
- Categorical variables: missing values, missing levels
- Categorical variables: too many levels
- Invalid values
    - Out of range numerical values
    - Invalid category levels

7

First Example: Bad/missing Numeric Values

8

Bad Numerical Values

Miles driven	Gas Consumption
100	2
235	0
150	7.5
200	5.5
0	0
300	NA

MPG	
50	
Inf	Electric car/bad calculation
20	
36.4	
NaN	Non-numeric typo/ bad calculation
NA	Electric car

9

Whither Bad Values?

- “Faulty Sensor” — values are missing at random
    - Assume they come from the same distribution as the other values
    - The mean of the “good” values is a reasonable stand-in
- Systematically missing
    - Electric cars
    - They WILL behave differently from gas or hybrid cars
    - The mean of the good values is not a valid stand-in

10

A number of possible solutions

- Naive: skip rows with missing values
- Multiple models: build many models using incomplete subsets of the columns.
- Imputation: build additional models that guess values for missing variables based on other variables.
- Statistical: sum-out or integrate-out missing values.
- Pragmatic: replace with harmless stand-ins and add notation so the machine learning system is aware of the situation.

11

Missingness as signal

- In business analytics missing data is often an indicator of where the data came from and how it was processed.
- Consequently it is often one of your more informative signals when modeling!

12

One Pragmatic Solution

MPG
50
Inf
20
36.4
NaN
NA

MPG	MPG_isBad
50	FALSE
35.5	TRUE
20	FALSE
36.4	FALSE
35.5	TRUE
35.5	TRUE

13

Second Example: Unexpected or Novel Categorical Levels

14

Categorical Variables: Missing Values and Novel Levels

TrainingData

Residence
CA
NV
OR
CA
CA
NA
WA
OR
WA

NewData

Residence
NV
OR
NV
WY
CA
CA
NV
NA
OR

15

Novel Levels - Model Failure

```
> model = lm("premium~age+sex+residence",
data=TrainingData)

> predPremium = predict(model,
                        newdata=NewData)

Error in model.frame.default(Terms, newdata,
na.action = na.action, xlev = object$xlevels) :
factor residence has new levels WY
```

16

On the Way to the Solution: Indicator Variables

Residence
CA
NV
OR
CA
CA
NA
WA
OR
WA

Res_NA	Res_CA	Res_NV	Res_WA	Res_OR
0	1	0	0	0
0	0	1	0	0
0	0	0	0	1
0	1	0	0	0
0	1	0	0	0
1	0	0	0	0
0	0	0	1	0
0	0	0	0	1
0	0	0	1	0

17

Three Possible Solutions

Training Data Proportions

NA	CA	NV	WA	OR
1/9	1/3	1/9	2/9	2/9

1) A novel level is weighted proportional to known levels

Residence
WY

Res_NA	Res_CA	Res_NV	Res_WA	Res_OR
1/9	1/3	1/9	2/9	2/9

2) A novel level is treated as "no level"

Residence
WY

Res_NA	Res_CA	Res_NV	Res_WA	Res_OR
0	0	0	0	0

3) A novel level is treated as uncertainty among rare levels

Residence
WY

Res_NA	Res_CA	Res_NV	Res_WA	Res_OR
1/2	0	1/2	0	0

18

`vtreat` solution

Residence	# of occurrences
CA	2000
NV	1100
OR	1000
WA	1500
WY	18
ID	14
CO	8

Residence	# of occurrences
CA	2000
NV	1100
OR	1000
WA	1500
RARE	40

- Levels that appear fewer than N times (N user specified) : pooled to **`rare`**
- Levels (including `rare`) that don't achieve statistical significance code to **`zap`**
  - `zap` codes to "no level" (no model effect)
  - novel levels code to `rare` (if available), otherwise to `zap`

19

Third Example: Categorical Variables with Very Many Levels

20

Categorical variables: Too many levels

ZIP	SalePriceK
94127	725
94564	402
90011	386
94704	790
94127	1195
94109	903
94124	625
94124	439
94564	290

- Too many levels is a computational problem for some machine learning algorithms.

- You will inevitably have a novel level

21

The Best (but not always possible) Solution

Use as join key into domain knowledge.

San Francisco County ZIP codes	Avg. listing price Week ending Aug 13	Median sales price Date range: May-Aug '14
Name	Amount	Amount
94124	$571,667	$625,000
94134	$619,495	$640,000
94132	$713,583	$835,000
94102	$768,558	$605,000
94112	$771,234	$728,250
94111	$877,000	$959,000
94116	$904,071	$1,025,000
94107	$1,019,113	$908,500
94117	$1,057,000	$1,125,000
94131	$1,057,160	$1,200,000
94110	$1,128,511	$1,082,000
94122	$1,227,482	$930,000
94114	$1,405,793	$1,452,000
94103	$1,406,597	$850,000
94109	$1,408,431	$903,500
94105	$1,549,047	$1,107,500
94127	$1,569,846	$1,300,000

22

Pragmatic Solution: "Impact/Effects Coding"

ZIP	avgPriceK	ZIP_impact
90011	386	-253.4
94109	903	263.6
94124	532	-107.4
94127	960	320.6
94564	346	-293.4
94704	790	150.6
globalAvg	639.4	0

23

Impact-coding the ZIP variable

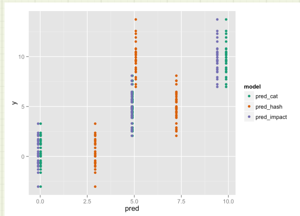
ZIP
94127
94564
90011
94704
94127
94109
94124
94124
93401

ZIP_impact
320.6
-293.4
-253.4
150.6
320.6
263.6
-107.4
-107.4
0

24

Sidebar:
Impact-Code; DON'T Hash!

- `Python/scikit-learn`: only takes numerical variables
- Hashing loses information!
- Impact-code, or convert to indicators: `OneHotEncoder()`
- If you must hash, use Random Forest



http://www.win-vector.com/blog/2014/12/a-comment-on-preparing-data-for-classifiers/

25

Automating Variable Treatment in R:
`vtreat`

26

Two-step Process

- Design the data treatment plans
  - Numeric outcome:
    `tPln = designTreatmentsN(train, xv, y)`
  - Binary class outcome
    `tPln = designTreatmentsC(train, xv, y, target)`
- Prepare the data sets
  - `train.treat = prepare(tPln, train, pruneSig=0.05)`
  - `test.treat = prepare(tPln, test, pruneSig=0.05)`

27

Designing the Treatment Plans:
Numeric Output

salePrice ~ ZIP + homeType + numBed + numBath + sqFt

treatPlan = designTreatmentsN(train,
c("ZIP", "homeType", "numBed", "numBath", "sqFt"),
"salePrice")

28

Example Input

```
  ZIP  homeType numBed numBath sqFt salePrice
94499     condo      4       4 1025    815678
94403     condo      2       3 1082    600635
94361 townhouse      1       3  751    444609
94115     condo      2       3 1093    349433
94217      <NA>     NA       3  914    692468
```

many-level categorical

categorical

numeric

treatPlan = designTreatmentsN(train,
c("ZIP", "homeType", "numBed", "numBath", "sqFt"),
"salePrice")

29

Using the treatment plan
to prepare data

`df.treat = prepare(treatPlan, df, pruneSig=0.2)`

df is any frame of appropriate format (training or test)

ZIP_catN	homeType_lev_NA	homeType_lev_x.condo	homeType_lev_x.loft
190033.174	0	1	0
-5320.826	0	1	0
35596.174	0	0	0
-119202.826	0	1	0
-94775.326	1	0	0

homeType_lev_x.single.family	homeType_lev_x.townhouse	numBed_clean	numBed_isBAD
0	0	4.000000	0
0	0	2.000000	0
0	1	1.000000	0
0	0	2.000000	0
0	0	2.456325	1

numBath_clean	numBath_isBAD	sqFt_clean	salePrice
4.000000	0	1025	815678
3.000000	0	1082	600635
3.000000	0	751	444609
3.000000	0	1093	349433
3.000000	0	914	692468

30

Designing the Treatment Plans: Binary Classification

loanApproved ~ ZIP + loanType + income + homePrice + FICO

```
treatPlan = designTreatmentsC(train,
  c("ZIP", "loanType", "income", "homePrice", "FICO"),
  "loanApproved", TRUE)
```

31

Conclusions

- There's no substitute for getting your hands in the data
- Nonetheless, some variable treatments are reusable again and again
- We've presented our go-to data treatments, and an R implementation for them: `vtreat`

32

Further References

- Impact Coding
  - http://www.win-vector.com/blog/2012/07/modeling-trick-impact-coding-of-categorical-variables-with-many-levels/
  - http://www.win-vector.com/blog/2012/08/a-bit-more-on-impact-coding/
- Converting Categorical Variables to Numerical (No Hashing)
  - http://www.win-vector.com/blog/2014/12/a-comment-on-preparing-data-for-classifiers/
- PRESS statistic
  - http://www.win-vector.com/blog/2014/09/estimating-generalization-error-with-the-press-statistic/

33

More references on `vtreat`

- vtreat on CRAN
  - https://cran.r-project.org/package=vtreat
- vtreat code on GitHub
  - https://github.com/WinVector/vtreat

34

Additional Issues: Overfitting and False Fitting

35

Issues

- Overfit from too many variables
  - Variable Selection
- False fit: upward biased model evaluations from nested models
  - Calibration sets
  - Data fuzzing (differential privacy techniques)

36

Switch Speakers

All materials: https://github.com/WinVector/PreparingDataWorkshop

37