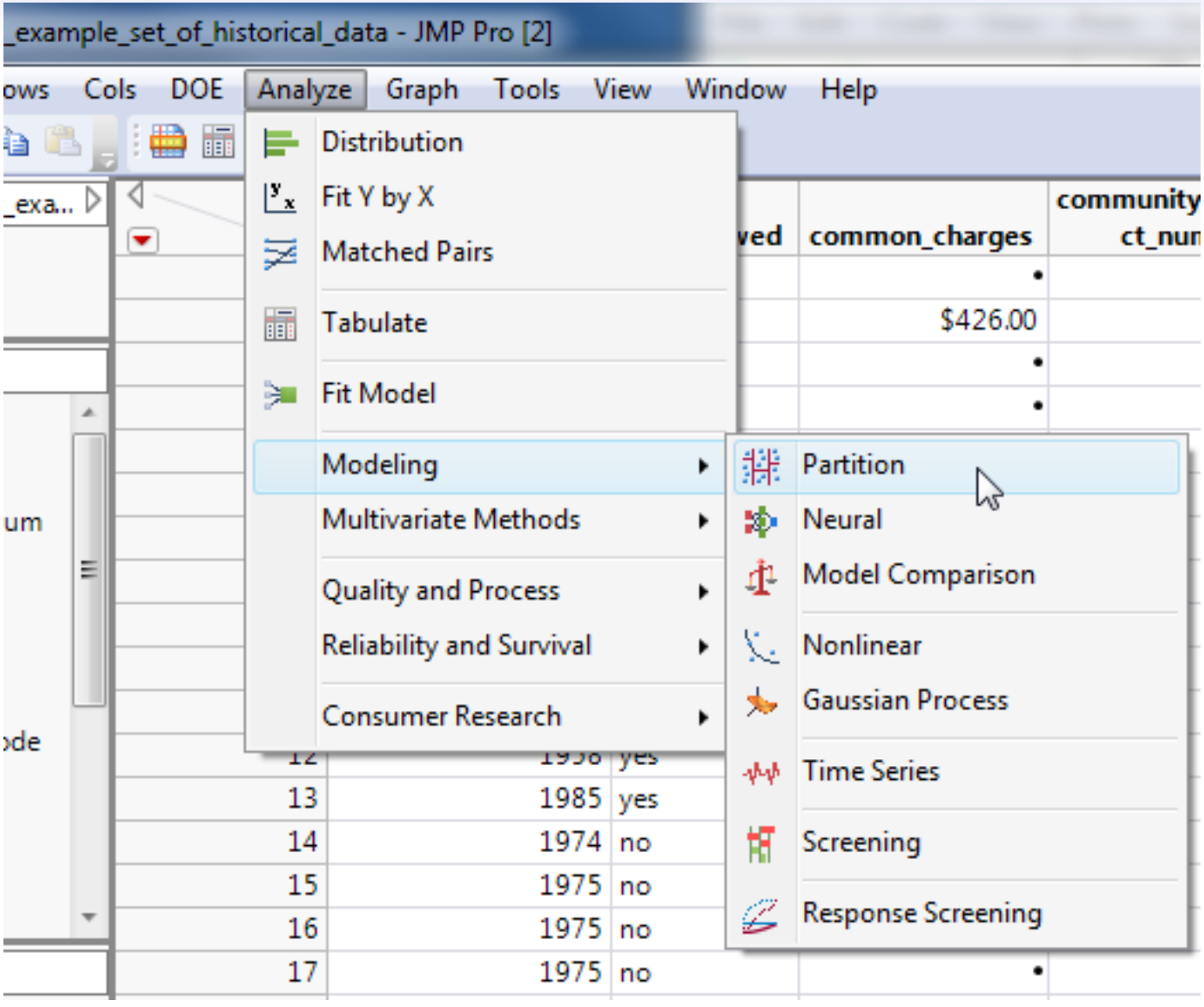
Random Forests in JMP and R

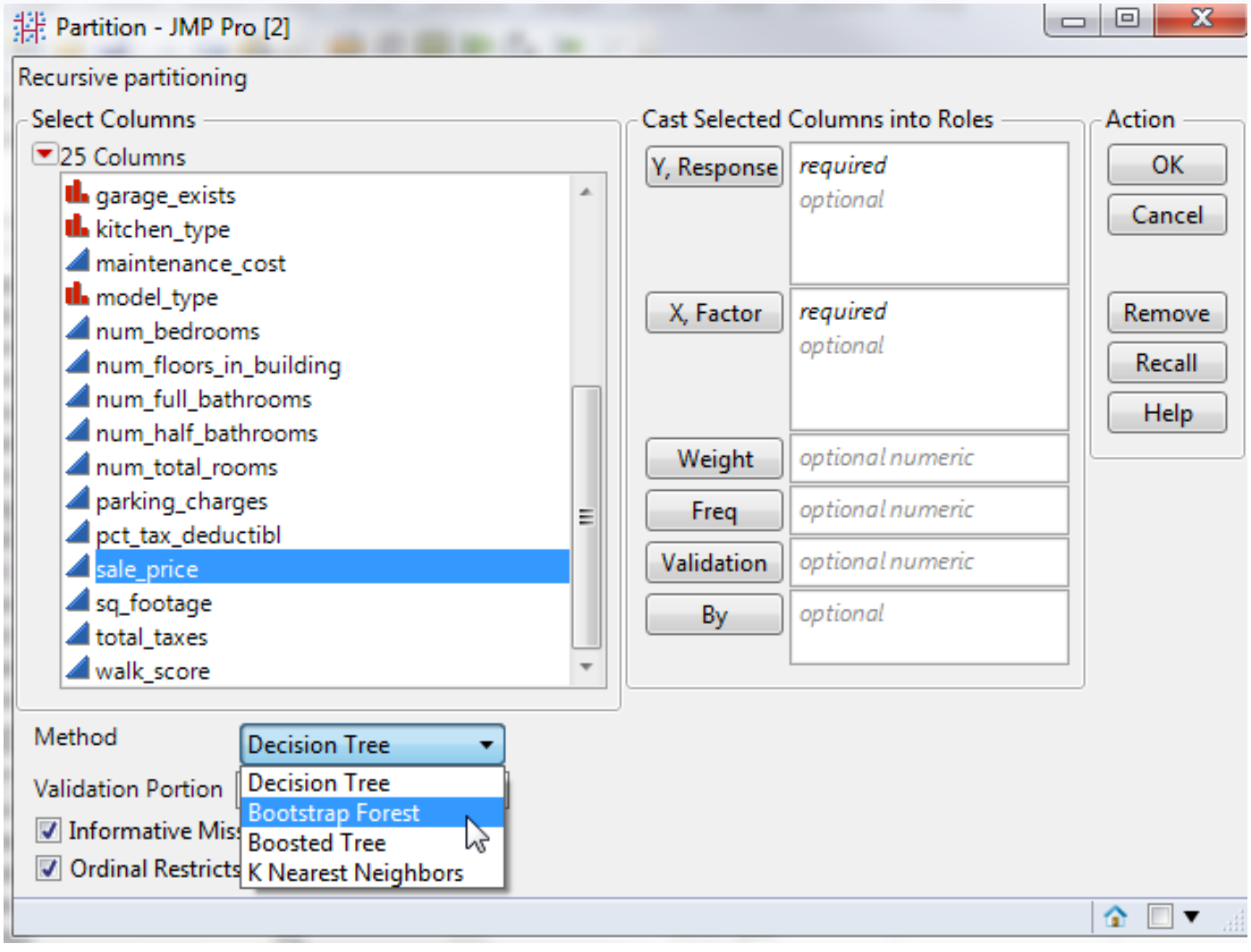
Adam Kapelner

Stat 422/722
at The Wharton School of the University of Pennsylvania

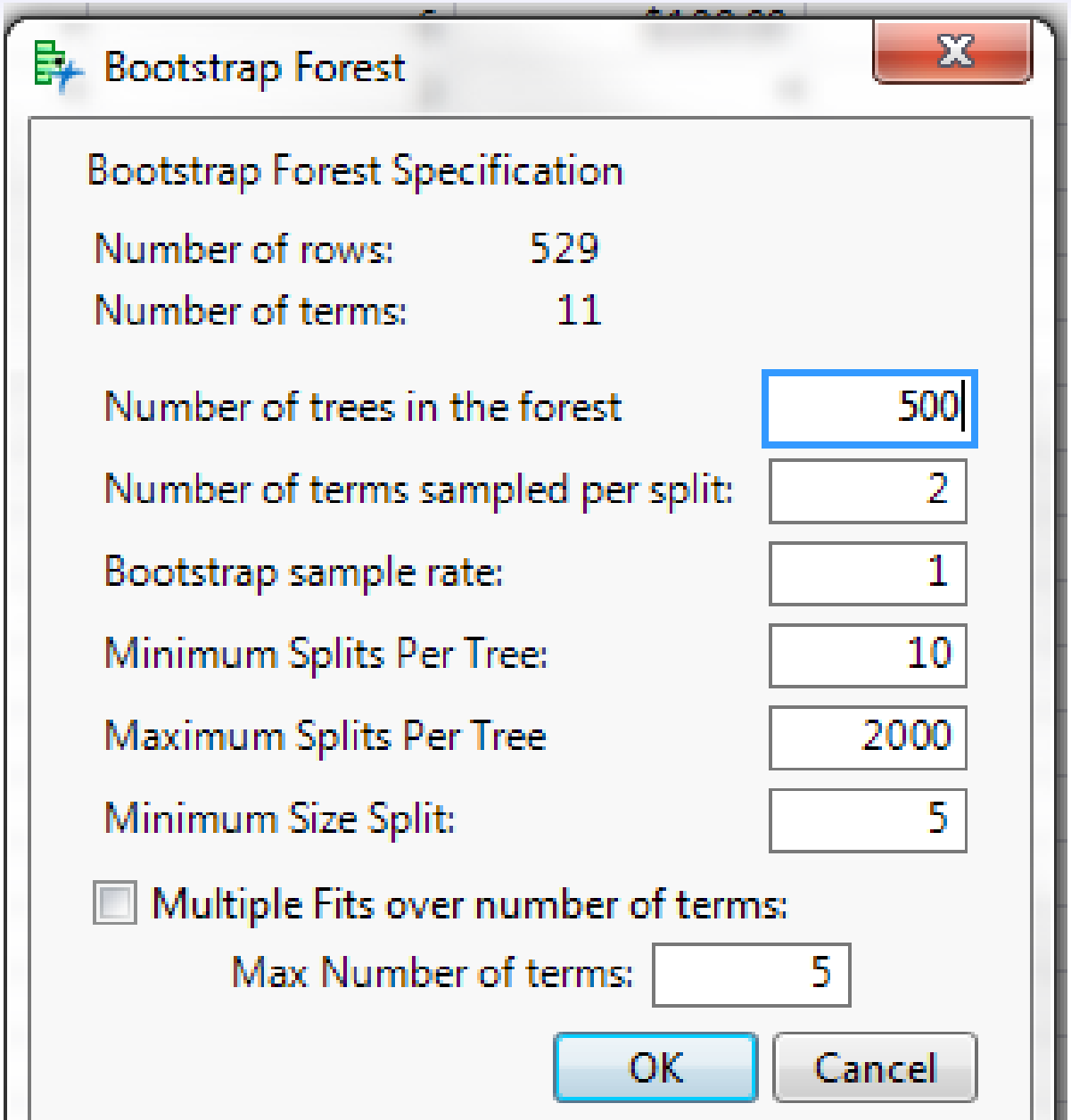
JMP 1/4


_example_set_of_historical_data - JMP Pro [2]
ows
Cols
DOE
Analyze
Graph
Tools
View
Window
Help
Distribution
Fit Y by X
Matched Pairs
Tabulate
Fit Model
Modeling
Multivariate Methods
Quality and Process
Reliability and Survival
Consumer Research
Partition
Neural
Model Comparison
Nonlinear
Gaussian Process
Time Series
Screening
Response Screening
common_charges
community
ct_nun
$426.00
13 1985 yes
14 1974 no
15 1975 no
16 1975 no
17 1975 no

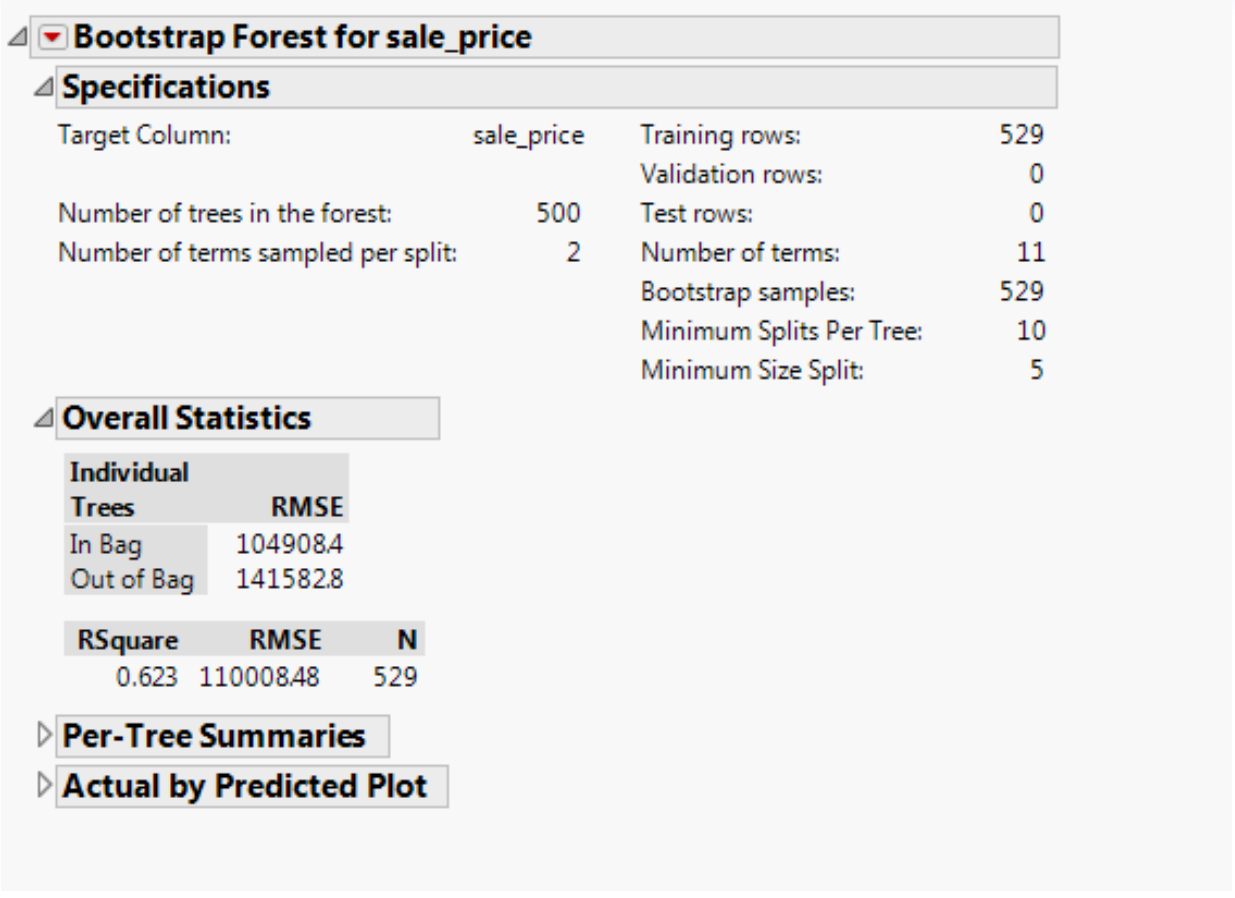
JMP 2/4


Partition - JMP Pro [2]
Recursive partitioning
Select Columns
25 Columns
garage_exists
kitchen_type
maintenance_cost
model_type
num_bedrooms
num_floors_in_building
num_full_bathrooms
num_half_bathrooms
num_total_rooms
parking_charges
pct_tax_deductibl
sale_price
sq_footage
total_taxes
walk_score
Cast Selected Columns into Roles
Y, Response
required
optional
X, Factor
required
optional
Weight
optional numeric
Freq
optional numeric
Validation
optional numeric
By
optional
Action
OK
Cancel
Remove
Recall
Help
Method
Decision Tree
Validation Portion
Informative Miss
Ordinal Restricts
Decision Tree
Bootstrap Forest
Boosted Tree
K Nearest Neighbors

JMP 3/4


Bootstrap Forest
Bootstrap Forest Specification
Number of rows: 529
Number of terms: 11
Number of trees in the forest 500
Number of terms sampled per split: 2
Bootstrap sample rate: 1
Minimum Splits Per Tree: 10
Maximum Splits Per Tree 2000
Minimum Size Split: 5
Multiple Fits over number of terms:
Max Number of terms: 5
OK
Cancel

JMP 4/4



Bootstrap Forest for sale_price

Specifications

Target Column:	sale_price	Training rows:	529
		Validation rows:	0
Number of trees in the forest:	500	Test rows:	0
Number of terms sampled per split:	2	Number of terms:	11
		Bootstrap samples:	529
		Minimum Splits Per Tree:	10
		Minimum Size Split:	5

Overall Statistics

Individual Trees	RMSE
In Bag	104908.4
Out of Bag	141582.8

RSquare	RMSE	N
0.623	110008.48	529

Per-Tree Summaries

Actual by Predicted Plot

You don't need to do oos validation on Random Forests as the "out of bag" is as good as oos.

R 1/2

```
#load the data
X = read.csv(
 "stat_422_722_project_example_set_of_historical_data.csv")
#recode sale_price as a number
X$sale_price = as.numeric(gsub('[$,]','',
  as.character(X$sale_price)))
#install and load up the RF package
install.packages("randomForest")
library(randomForest)
#run the RF
rf_mod = randomForest(sale_price ~
coop_condo + num_bedrooms +
num_full_bathrooms + walk_score, X)
rf_mod #print out its output
```

R 2/2

```
> rf_mod

Call:
 randomForest(formula = sale_price ~ coop_condo + num_bed
rooms +       num_full_bathrooms + walk_score, data = X)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 1

          Mean of squared residuals: 10620431253
                    % Var explained: 66.96
```

Again: you don't need to do oos validation on Random Forests as the "out of bag" is as good as oos.