

# Model Fitting II

Carolina A. de Lima Salge  
Assistant Professor  
Terry College of Business  
University of Georgia

*Business Intelligence  
Spring 2021*



Terry College of Business  
UNIVERSITY OF GEORGIA

A tibble: 1,338 x 7

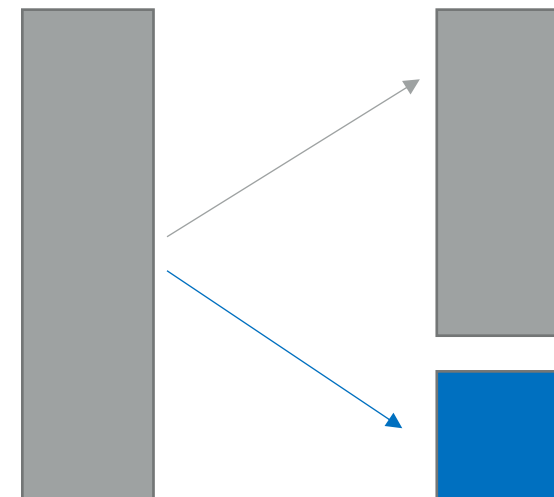
age <dbl>	sex <chr>	bmi <dbl>	children <dbl>	smoker <chr>	region <chr>	charges <dbl>
19	female	27.900	0	yes	southwest	16884.924
18	male	33.770	1	no	southeast	1725.552
28	male	33.000	3	no	southeast	4449.462
33	male	22.705	0	no	northwest	21984.471
32	male	28.880	0	no	northwest	3866.855
31	female	25.740	0	no	southeast	3756.622
46	female	33.440	1	no	southeast	8240.590
37	female	27.740	3	no	northwest	7281.506
37	male	29.830	2	no	northeast	6406.411
60	female	25.840	0	no	northwest	28923.137

1-10 of 1,338 rows

Previous 1 2 3 4 5 6 ... 100 Next

```
> postResample(pred = p, obs = charges_test$charges)
```

RMSE	Rsquared	MAE
5808.0045894	0.7989742	4184.9721150



Data

Training  
Data

Test  
Data

```
> summary(diff(resamps))
```

Call:  
summary.diff.resamples(object = diff(resamps))

p-value adjustment: bonferroni  
Upper diagonal: estimates of the difference  
Lower diagonal: p-value for H0: difference = 0

MAE	LM	RF
LM		1273
RF	< 2.2e-16	

RMSE	LM	RF
LM		968.9
RF	1.14e-12	

Rsquared	LM	RF
LM		-0.08711
RF	8.273e-14	

# Machine Learning Use

## Predictive modeling

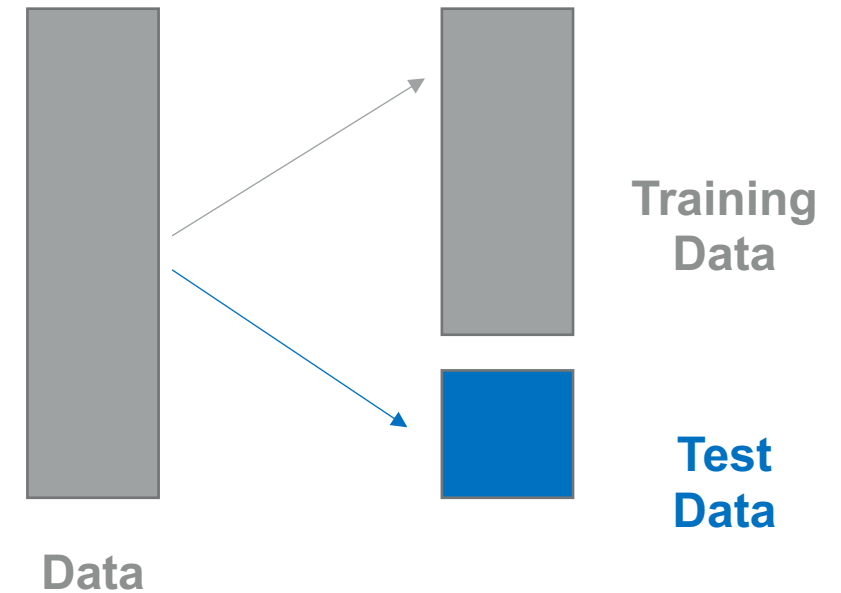
- The goal is to predict the target using a new dataset where we have values for predictors but not the target



# Machine Learning Use

Evaluate based on prediction error

- Build model using training data
- Assess performance on test (hold-out) data



# Model Evaluation

How well the model predicts new data (*not* how well it fits the data it was trained with)

- Key component of most measures is difference between actual outcome and predicted outcome (i.e., error)



# Model Evaluation

Error for data record = predicted (p) minus actual (a)

**RMSE: Root Mean Squared Error**

MAE: Mean Absolute Error

MAPE: Mean Absolute Percentage Error

Total SSE: Total Sum of Squared Errors



# RMSE

Error for data record = predicted ( $p$ ) minus actual ( $a$ )

RMSE = how much the  $p$ 's diverge from the  $a$ 's, on average

Assume the regression equation is  $y = 1.74x$ . What is the root mean squared error for the sample dataset?

$x$	$a$	$p$	$(p - a)^2$
1	2		
2	5		
-1	-2		

# RMSE

x	a	p	(p - a)^2
1	2	1.74	
2	5	3.48	
-1	-2	-1.74	

$$\begin{aligned}y &= 1.74x \\&= 1.74 * 1 = 1.74 \\&= 1.74 * 2 = 3.48 \\&= 1.74 * (-1) = -1.74\end{aligned}$$



# RMSE

x	a	p	(p - a)^2
1	2	1.74	0.0676
2	5	3.48	2.3104
-1	-2	-1.74	0.0676

$$= (1.74 - 2)^2 = 0.0676$$

$$= (3.48 - 5)^2 = 2.3104$$

$$= (-1.74 - -2)^2 = 0.0676$$





# RMSE

x	a	p	(p - a)^2
1	2	1.74	0.0676
2	5	3.48	2.3104
-1	-2	-1.74	0.0676

$$\begin{aligned}\text{RMSE} &= (0.0676 + 2.3104 + 0.0676) / 3 \\ &= \sqrt{0.8152} \\ &= .903\end{aligned}$$



# ML Regression in R


Use the the **caret** package

Insurance dataset – recall, *the goal is to predict the target using a new dataset as best as we can*

```
library(tidyverse)
library(caret)


insurance <- read_csv("insurance.csv")
```


# Insurance Data


 Dataset


## Medical Cost Personal Datasets


Insurance Forecast by using Linear Regression

 Miri Choi • updated 3 years ago (Version 1)

[Data](#) [Tasks \(2\)](#) [Code \(478\)](#) [Discussion \(11\)](#) [Activity](#) [Metadata](#) [Download \(54 KB\)](#) [New Notebook](#) 

 **Usability** 8.8

 **License** Database: Open Database, Contents: Database Contents

 **Tags** education, health, finance, insurance, healthcare

### Description

#### Context

Machine Learning with R by Brett Lantz is a book that provides an introduction to machine learning using R. As far as I can tell, Packt Publishing does not make its datasets available online unless you buy the book and create a user account which can be a problem if you are checking the book out from the library or borrowing the book from a friend. All of these datasets are in the public domain but simply needed some cleaning up and recoding to match the format in the book.

#### Content

Columns

- age: age of primary beneficiary
- sex: insurance contractor gender, female, male



# ML Regression in R

insurance

A tibble: 1,338 x 7

age <dbl>	sex <chr>	bmi <dbl>	children <dbl>	smoker <chr>	region <chr>	charges <dbl>
19	female	27.900	0	yes	southwest	16884.924
18	male	33.770	1	no	southeast	1725.552
28	male	33.000	3	no	southeast	4449.462
33	male	22.705	0	no	northwest	21984.471
32	male	28.880	0	no	northwest	3866.855
31	female	25.740	0	no	southeast	3756.622
46	female	33.440	1	no	southeast	8240.590
37	female	27.740	3	no	northwest	7281.506
37	male	29.830	2	no	northeast	6406.411
60	female	25.840	0	no	northwest	28923.137

1–10 of 1,338 rows

Previous **1** 2 3 4 5 6 ... 100 Next



# Selecting Predictors

The goal is to find a parsimonious model – i.e., a simple model that performs well

- Correlation between predictors
- Correlation between predictors and target



# Selecting Predictors

To compute the correlation, we need numeric values

```
# transform categories to numbers
insurance <- insurance %>%
  mutate(sexN = case_when(
    sex == "male" ~ 1,
    sex == "female" ~ 0
  )) %>%
  mutate(smokerN = case_when(
    smoker == "yes" ~ 1,
    smoker == "no" ~ 0
  )) %>%
  mutate(regionN = case_when(
    region == "southwest" ~ 1,
    region == "southeast" ~ 2,
    region == "northwest" ~ 3,
    region == "northeast" ~ 4
  ))
```



# Selecting Predictors

To compute the correlation, we need numeric values

```
# only select numeric variables
df <- insurance %>% dplyr::select(charges, age, sexN, bmi, children, smokerN, regionN)

# drop missing values NAs
df1 <- drop_na(df)

# compute correlation between predictors
cor(df1[,2:7])
```

```
> cor(df1[,2:7])
```

	age	sexN	bmi	children	smokerN	regionN
age	1.000000000	-0.020855872	0.109271882	0.04246900	-0.025018752	-0.002127313
sexN	-0.020855872	1.000000000	0.046371151	0.01716298	0.076184817	-0.004588385
bmi	0.109271882	0.046371151	1.000000000	0.01275890	0.003750426	-0.157565849
children	0.042468999	0.017162978	0.012758901	1.000000000	0.007673120	-0.016569446
smokerN	-0.025018752	0.076184817	0.003750426	0.00767312	1.000000000	0.002180682
regionN	-0.002127313	-0.004588385	-0.157565849	-0.01656945	0.002180682	1.000000000



# Selecting Predictors

To compute the correlation, we need numeric values

```
# compute correlation between predictors and the target  
cor(df1[,1:7])
```

```
> cor(df1[,1:7])
```

	charges	age	sexN	bmi	children	smokerN	regionN
charges	1.000000000	0.299008193	0.057292062	0.198340969	0.06799823	0.787251430	0.006208235
age	0.299008193	1.000000000	-0.020855872	0.109271882	0.04246900	-0.025018752	-0.002127313
sexN	0.057292062	-0.020855872	1.000000000	0.046371151	0.01716298	0.076184817	-0.004588385
bmi	0.198340969	0.109271882	0.046371151	1.000000000	0.01275890	0.003750426	-0.157565849
children	0.067998227	0.042468999	0.017162978	0.012758901	1.000000000	0.007673120	-0.016569446
smokerN	0.787251430	-0.025018752	0.076184817	0.003750426	0.00767312	1.000000000	0.002180682
regionN	0.006208235	-0.002127313	-0.004588385	-0.157565849	-0.01656945	0.002180682	1.000000000



# Model Induction

Set a starting value so that results are reproducible  
Split the data into training and testing

```
set.seed(12L) # set a starting seed to be able to get reproducible results

# partition data
trainIndex <- createDataPartition(df1$charges, # target variable
                                   p = 0.8, # percentage that goes to training
                                   list = FALSE, # results will not be in a list
                                   times = 1) # number of partitions to create

charges_train <- df1[trainIndex, ] # data frame for training
charges_test <- df1[-trainIndex, ] # data frame for testing
```



# Model Induction and Testing

Use training set to build model, then predict insurance cost using the test set

```
model <- train(charges ~ age + bmi + smokerN,  
               data = charges_train, # use training set  
               method = "lm") # linear regression  
  
# now predict outcomes in test set  
p <- predict(model, charges_test)
```



# Model Performance

Use training set to build model, then predict insurance cost using the test set

```
# how did we do? calculate performance across resamples  
# RMSE and R-squared  
postResample(pred = p, obs = charges_test$charges)  
# on average, our prediction is off by $5,808.00
```

```
> postResample(pred = p, obs = charges_test$charges)  
          RMSE      Rsquared      MAE  
5808.0045894  0.7989742 4184.9721150
```



# Model Performance

How to improve performance? One way is to try and specify a different method

```
model2 <- train(charges ~ age + bmi + smokerN,  
               data = charges_train, # use training set  
               method = "ranger") # random forest
```

```
# now predict outcomes in test set  
p1 <- predict(model2, charges_test)
```

```
# how did we do? calculate performance across resamples  
# RMSE and R-squared  
postResample(pred = p1, obs = charges_test$charges)  
# on average, our prediction is off by $4,209.91
```

```
> postResample(pred = p1, obs = charges_test$charges)
```

RMSE	Rsquared	MAE
4209.9144114	0.8931659	2451.1495949



# Which Model?

## So many choices!

### Linear Regression

```
method = 'lm'
```

Type: Regression

Tuning parameters:

- `intercept` (intercept)

A model-specific variable importance metric is available.

### Random Forest

```
method = 'ranger'
```

Type: Classification, Regression

Tuning parameters:

- `mtry` (#Randomly Selected Predictors)
- `splitrule` (Splitting Rule)
- `min.node.size` (Minimal Node Size)

Required packages: `e1071` , `ranger` , `dplyr`

A model-specific variable importance metric is available.

<http://topepo.github.io/caret/train-models-by-tag.html#model-tree>

## 7 train Models By Tag

The following is a basic list of model types or relevant characteristics. There entire in these lists are arguable. For example: random forests theoretically use feature selection but effectively may not, support vector machines use L2 regularization etc.

Contents

- Accepts Case Weights
- Bagging
- Bayesian Model
- Binary Predictors Only
- Boosting
- Categorical Predictors Only
- Cost Sensitive Learning
- Discriminant Analysis
- Distance Weighted Discrimination
- Ensemble Model
- Feature Extraction
- Feature Selection Wrapper
- Gaussian Process
- Generalized Additive Model
- Generalized Linear Model
- Handle Missing Predictor Data
- Implicit Feature Selection
- Kernel Method
- L1 Regularization
- L2 Regularization
- Linear Classifier
- Linear Regression
- Logic Regression
- Logistic Regression
- Mixture Model
- Model Tree
- Multivariate Adaptive Regression Splines
- Neural Network
- Oblique Tree
- Ordinal Outcomes
- Partial Least Squares
- Patient Rule Induction Method
- Polynomial Model
- Prototype Models
- Quantile Regression
- Radial Basis Function
- Random Forest
- Regularization
- Relevance Vector Machines



# Which Model?

## 6 Available Models

The models below are available in `train`. The code behind these protocols can be obtained using the function `getModelInfo` or by going to the [github repository](#).

Show 238 entries

Search:

Model	<i>method</i>	Value	Type	Libraries	Tuning Parameters
Adaptive- Network-Based Fuzzy Inference System	ANFIS		Regression	frbs	num.labels, max.i
Bayesian Regularized Neural Networks	brnn		Regression	brnn	neurons
Bayesian Ridge Regression	bridge		Regression	monomvn	None
Bayesian Ridge Regression (Model Averaged)	blassoAveraged		Regression	monomvn	None
Cubist	cubist		Regression	Cubist	committees, neighbors

<http://topepo.github.io/caret/available-models.html>



# Comparing Between Models

## Is one model statistically better than the other?

```
# first collect the resampling results of each model
resamps <- resamples(list(LM = model,
                          RF = model2))
```

resamps

```
# then use a simple t-test to evaluate the null
hypothesis that there is no difference
summary(diff(resamps))
```

```
> resamps
```

```
Call:
resamples.default(x = list(LM = model, RF = model2))
```

```
Models: LM, RF
Number of resamples: 25
Performance metrics: MAE, RMSE, Rsquared
Time estimates for: everything, final model fit
```

```
> summary(diff(resamps))
```

```
Call:
summary.diff.resamples(object = diff(resamps))
```

```
p-value adjustment: bonferroni
Upper diagonal: estimates of the difference
Lower diagonal: p-value for H0: difference = 0
```

```
MAE
    LM      RF
LM      1273
RF < 2.2e-16
```

```
RMSE
    LM      RF
LM      968.9
RF 1.14e-12
```

```
Rsquared
    LM      RF
LM      -0.08711
RF 8.273e-14
```



# Summary

- Regression with ML is different than regression with traditional OLS – one is focused on predictions while the other is focused on explanations
- When building a predictive ML model, split data into training and test sets (70-30 or 80-20)
- Always evaluate the performance of a model with the test data, and experiment with different methods to compare the performances of different models





***Thank You!***

