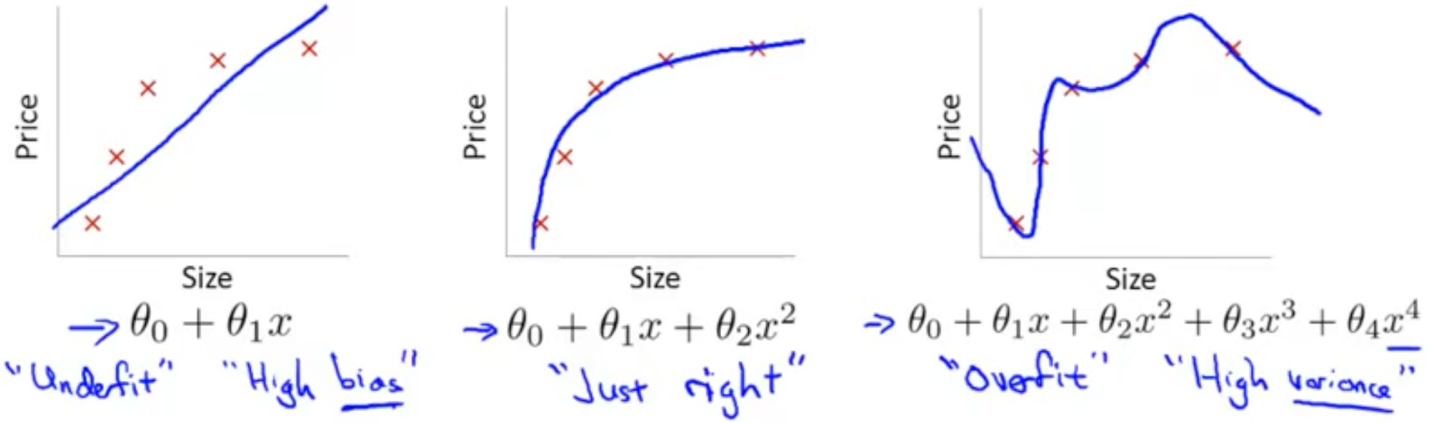
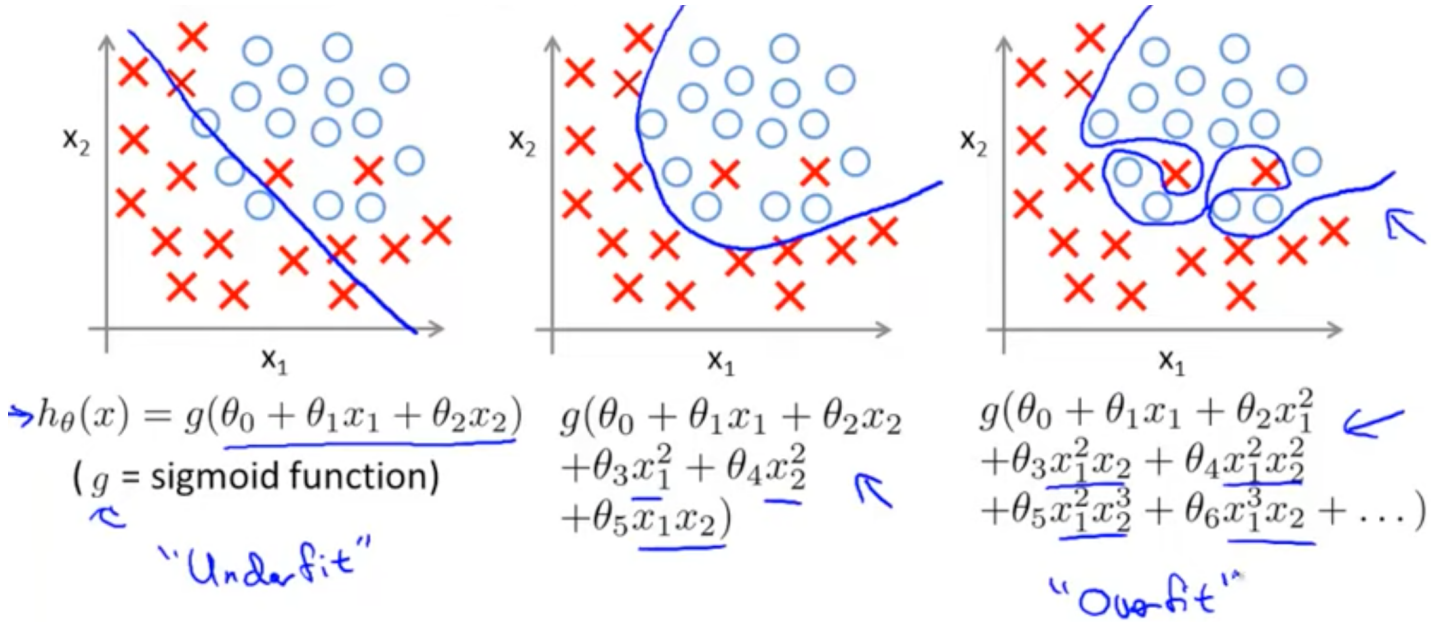
Overfitting

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

Overfitting

"One of the most important fundamental notions of data science is that of **overfitting** and **generalization**. If we allow ourselves enough flexibility in searching for patterns in a particular dataset, we will find patterns. Unfortunately, these "patterns" may be just chance occurrences in the data" (pg 111)

Generalization

We are interested in patterns that generalize

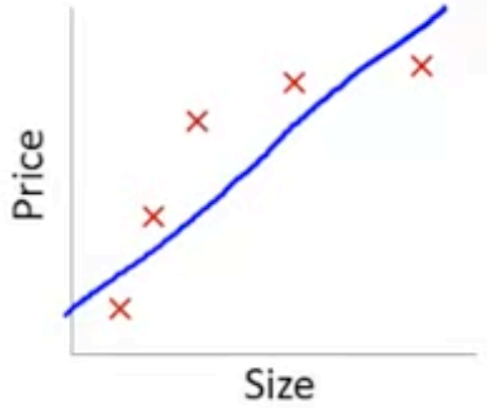
- i.e., we hope to create ML models that predict well for instances that we have not yet observed

Overfitting Defined

Finding chance occurrences in the training dataset that look like interesting patterns, but that do not generalize to the population of interest, is called ***overfitting*** the data

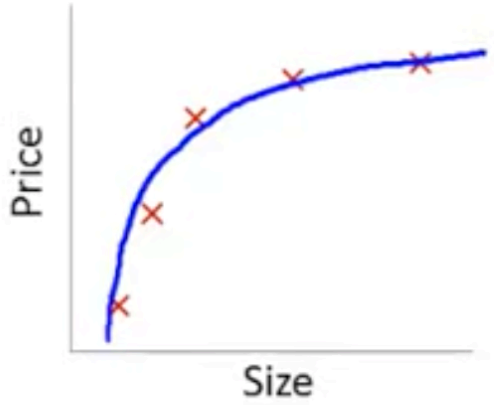
Overfitting is therefore the tendency of tailoring models to the training dataset, ***at the expense of generalization to previously unseen data points***

Example: Linear regression (housing prices)



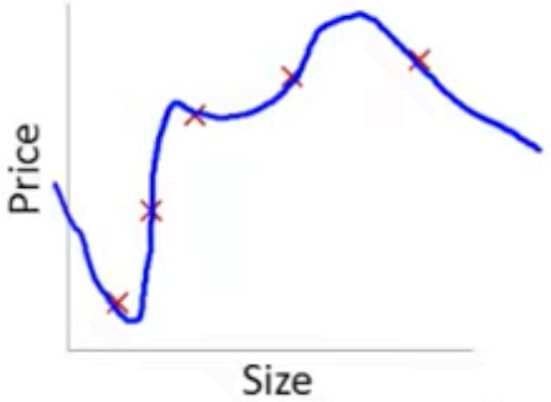
$\rightarrow \theta_0 + \theta_1 x$

"Underfit" "High bias"



$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2$

"Just right"

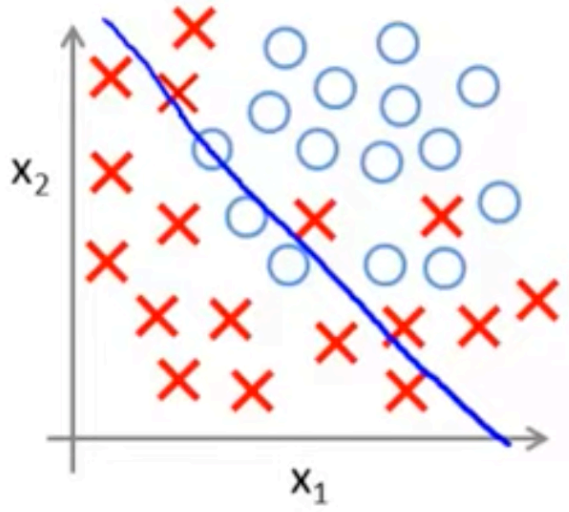


$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

"Overfit" "High variance"

Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

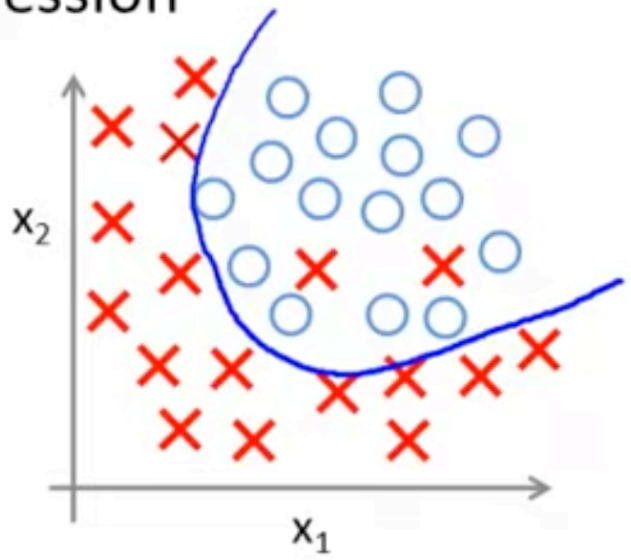
Example: Logistic regression



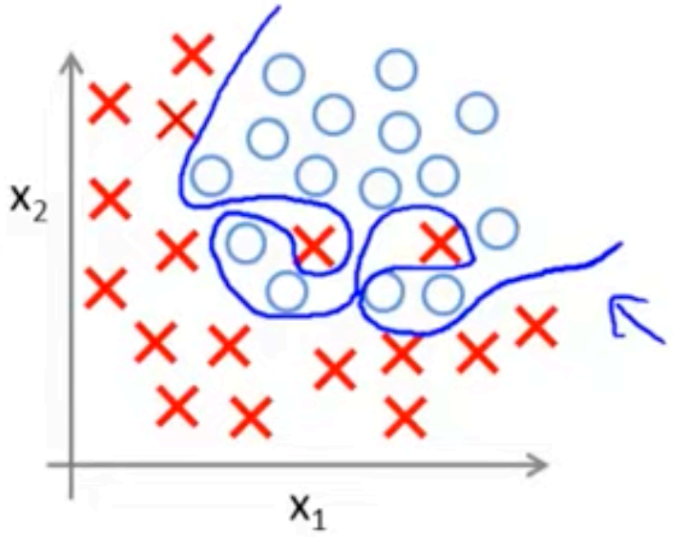
$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

"Underfit"



$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$



$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots)$

"Overfit"

No Single Solution

There is **no single choice or procedure that will eliminate overfitting**, and all procedures themselves have the tendency to over-fit to some extent

What to do?

Addressing Overfitting

Reduce number of features

- Manually select which features to keep
- Choose algorithms that automatically select features for you

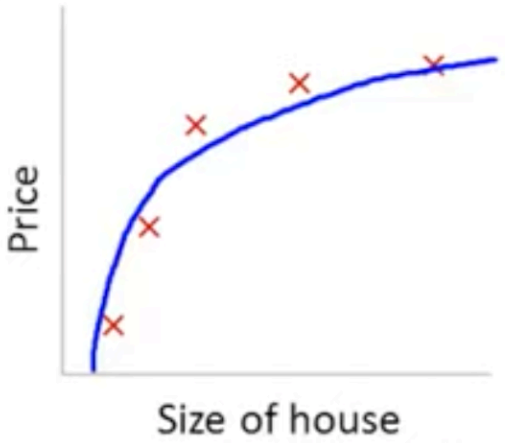
Addressing Overfitting

Regularization

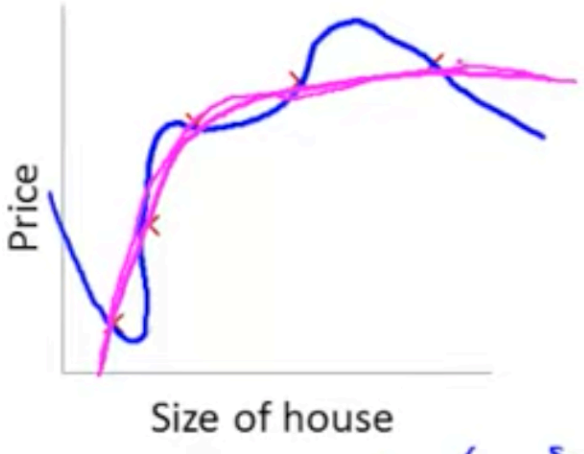
- Keep all the features, but reduce magnitude/values of parameters

Works well when we have a lot of features, each of which contributes a bit to predicting the target

Intuition



$$\theta_0 + \theta_1 x + \theta_2 x^2$$



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

Suppose we penalize and make θ_3, θ_4 really small.

$$\min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2$$

$\theta_3 \approx 0$ $\theta_4 \approx 0$

Regularization

Provides some combination of fit and simplicity

- Models will be better if they fit the data better, but they also will be better if they are simpler

Regularization

L1-norm penalty (sum of the absolute values of the weights)

- L1-norm + standard OLS = **Lasso regression**
- *Automatic feature selection*

Regularization

L2-norm penalty (sum of the squares of the weights)

- L2-norm + standard OLS = **Ridge regression**

Ridge Regression

```
method = 'ridge'
```

Type: Regression

Tuning parameters:

- `lambda` (Weight Decay)

Required packages: `elasticnet`

Avoiding Overfitting in R

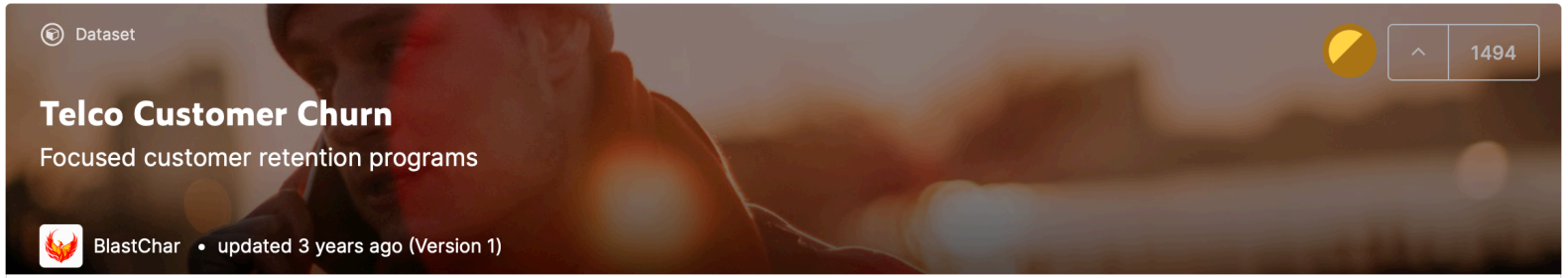
Use the the **`glmnet`** package

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

```
library(tidyverse)
library(caret)
library(glmnet)

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

Churn Data



Usability 8.8 **License** Data files © Original Authors **Tags** business

Description

Context

"Predict behavior to retain customers. You can analyze all relevant customer data and develop focused customer retention programs." [IBM Sample Data Sets]

Content

Each row represents a customer, each column contains customer's attributes described on the column Metadata.

The data set includes information about:

- Customers who left within the last month – the column is called Churn
- Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies

https://www.kaggle.com/blastchar/telco-customer-churn

Avoiding Overfitting in R

Filter

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
1	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
2	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	No	Yes	No	No	No	One year	No	Mailed check	56.95	1889.50	No
3	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
4	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	No	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
5	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	No	No	No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	Yes
6	9305-CDSKC	Female	0	No	No	8	Yes	Yes	Fiber optic	No	No	Yes	No	Yes	Yes	Month-to-month	Yes	Electronic check	99.65	820.50	Yes
7	1452-KIOVK	Male	0	No	Yes	22	Yes	Yes	Fiber optic	No	Yes	No	No	Yes	No	Month-to-month	Yes	Credit card (automatic)	89.10	1949.40	No
8	6713-OKOMC	Female	0	No	No	10	No	No phone service	DSL	Yes	No	No	No	No	No	Month-to-month	No	Mailed check	29.75	301.90	No
9	7892-POOKP	Female	0	Yes	No	28	Yes	Yes	Fiber optic	No	No	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	104.80	3046.05	Yes
10	6388-TABGU	Male	0	No	Yes	62	Yes	No	DSL	Yes	Yes	No	No	No	No	One year	No	Bank transfer (automatic)	56.15	3487.95	No
11	9763-GRSKD	Male	0	Yes	Yes	13	Yes	No	DSL	Yes	No	No	No	No	No	Month-to-month	Yes	Mailed check	49.95	587.45	No
12	7469-LKBCI	Male	0	No	No	16	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service	Two year	No	Credit card (automatic)	18.95	326.80	No
13	8091-TTVAX	Male	0	Yes	No	58	Yes	Yes	Fiber optic	No	No	Yes	No	Yes	Yes	One year	No	Credit card (automatic)	100.35	5681.10	No
14	0280-XJGEX	Male	0	No	No	49	Yes	Yes	Fiber optic	No	Yes	Yes	No	Yes	Yes	Month-to-month	Yes	Bank transfer (automatic)	103.70	5036.30	Yes
15	5129-JLPIS	Male	0	No	No	25	Yes	No	Fiber optic	Yes	No	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	105.50	2686.05	No
16	3655-SNQYZ	Female	0	Yes	Yes	69	Yes	Yes	Fiber optic	Yes	Yes	Yes	Yes	Yes	Yes	Two year	No	Credit card (automatic)	113.25	7895.15	No
17	8191-XWSZG	Female	0	No	No	52	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service	One year	No	Mailed check	20.65	1022.95	No
18	9959-WOFKT	Male	0	No	Yes	71	Yes	Yes	Fiber optic	Yes	No	Yes	No	Yes	Yes	Two year	No	Bank transfer (automatic)	106.70	7382.25	No
19	4190-MFLUW	Female	0	Yes	Yes	10	Yes	No	DSL	No	No	Yes	Yes	No	No	Month-to-month	No	Credit card (automatic)	55.20	528.35	Yes
20	4183-MYFRB	Female	0	No	No	21	Yes	No	Fiber optic	No	Yes	Yes	No	No	Yes	Month-to-month	Yes	Electronic check	90.05	1862.90	No
21	8779-QRDMV	Male	1	No	No	1	No	No phone service	DSL	No	No	Yes	No	No	Yes	Month-to-month	Yes	Electronic check	39.65	39.65	Yes
22	1680-VDCWW	Male	0	Yes	No	12	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service	One year	No	Bank transfer (automatic)	19.80	202.25	No
23	1066-JKSGK	Male	0	No	No	1	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service	Month-to-month	No	Mailed check	20.15	20.15	Yes
24	3638-WEABW	Female	0	Yes	No	58	Yes	Yes	DSL	No	Yes	No	Yes	No	No	Two year	Yes	Credit card (automatic)	59.90	3505.10	No
25	6322-HRPFA	Male	0	Yes	Yes	49	Yes	No	DSL	Yes	Yes	No	Yes	No	No	Month-to-month	No	Credit card (automatic)	59.60	2970.30	No
26	6865-JZNKO	Female	0	No	No	30	Yes	No	DSL	Yes	Yes	No	No	No	No	Month-to-month	Yes	Bank transfer (automatic)	55.30	1530.60	No
27	6467-CHFZW	Male	0	Yes	Yes	47	Yes	Yes	Fiber optic	No	Yes	No	No	Yes	Yes	Month-to-month	Yes	Electronic check	99.35	4749.15	Yes
28	8665-UTDHZ	Male	0	Yes	Yes	1	No	No phone service	DSL	No	Yes	No	No	No	No	Month-to-month	No	Electronic check	30.20	30.20	Yes
29	5248-YGIJN	Male	0	Yes	No	72	Yes	Yes	DSL	Yes	Yes	Yes	Yes	Yes	Yes	Two year	Yes	Credit card (automatic)	90.25	6369.45	No
30	8773-HHUOZ	Female	0	No	Yes	17	Yes	No	DSL	No	No	No	No	Yes	Yes	Month-to-month	Yes	Mailed check	64.70	1093.10	Yes
31	3841-NFECX	Female	1	Yes	No	71	Yes	Yes	Fiber optic	Yes	Yes	Yes	Yes	No	No	Two year	Yes	Credit card (automatic)	96.35	6766.95	No

Showing 1 to 31 of 7,043 entries, 21 total columns

Selecting Predictors

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

- **Lasso Regression** (will include all variables, automatic feature selection)

Selecting Predictors

Still transform variables

```
# transform categories to numbers
churn <- churn %>%
  mutate(genderN = case_when(
    gender == "Male" ~ 1,
    gender == "Female" ~ 0
    )) %>%
  mutate(PartnerN = case_when(
    Partner == "Yes" ~ 1,
    Partner == "No" ~ 0
    )) %>%
  mutate(DependentsN = case_when(
    Dependents == "Yes" ~ 1,
    Dependents == "No" ~ 0
    )) %>%
  mutate(PhoneServiceN = case_when(
    PhoneService == "Yes" ~ 1,
    PhoneService == "No" ~ 0
    )) %>%
  mutate(MultipleLinesN = case_when(
    MultipleLines == "Yes" ~ 1,
    MultipleLines == "No" ~ 0,
    MultipleLines == "No phone service" ~ 0
    )) %>%
  mutate(InternetServiceN = case_when(
    InternetService == "Fiber optic" ~ 2,
    InternetService == "DSL" ~ 1,
  mutate(DeviceProtectionN = case_when(
    DeviceProtection == "Yes" ~ 1,
    DeviceProtection == "No" ~ 0,
    DeviceProtection == "No internet service" ~ 0
    )) %>%
  mutate(TechSupportN = case_when(
    TechSupport == "Yes" ~ 1,
    TechSupport == "No" ~ 0,
    TechSupport == "No internet service" ~ 0
    )) %>%
  mutate(StreamingTVN = case_when(
    StreamingTV == "Yes" ~ 1,
    StreamingTV == "No" ~ 0,
    StreamingTV == "No internet service" ~ 0
    )) %>%
  mutate(StreamingMoviesN = case_when(
    StreamingMovies == "Yes" ~ 1,
    StreamingMovies == "No" ~ 0,
    StreamingMovies == "No internet service" ~ 0
    )) %>%
  mutate(ContractN = case_when(
    Contract == "Month-to-month" ~ 0,
    Contract == "One year" ~ 1,
    Contract == "Two year" ~ 1
    )) %>%
  mutate(PaperlessN = case_when(
    PaperlessBilling == "Yes" ~ 1,
```

Selecting Predictors

Still transform variables

```
# only select numeric variables
df <- churn %>% dplyr::select(Churn, SeniorCitizen, tenure,
                              MonthlyCharges, TotalCharges, genderN:PaymentN)

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

# transform target into a factor
df1$Churn <- as.factor(df1$Churn)
```

Splitting Data

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$Churn, # 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

churn_train <- df1[trainIndex, ] # data frame for training
churn_test <- df1[-trainIndex, ] # data frame for testing
```

Model Induction and Testing

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

```
# model induction
model <- train(Churn ~ .,
               data = churn_train, # use training set
               method = "glmnet",
                 # alpha and lambda paramters to try
                 tuneGrid = data.frame(alpha=1, # 1 is lasso 0 is ridge
                                       lambda=seq(0.0001,1))) # strength of penalty

# now predict outcomes in test set
p <- predict(model, churn_test, type = 'raw')

# add predictions to initial dataset
churn_test$pred_churn <- p
```

Model Performance

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

```
# how did we do? confusion matrix
confusionMatrix(data = churn_test$pred_churn,
                reference = churn_test$Churn,
                mode = "prec_recall",
                positive = "Yes")
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  924 171
       Yes 108 202

               Accuracy : 0.8014
                 95% CI : (0.7796, 0.822)
    No Information Rate : 0.7345
    P-Value [Acc > NIR] : 2.974e-09

                  Kappa : 0.4618

 Mcnemar's Test P-Value : 0.0002058

              Precision : 0.6516
                 Recall : 0.5416
                     F1 : 0.5915
             Prevalence : 0.2655
         Detection Rate : 0.1438
   Detection Prevalence : 0.2206
      Balanced Accuracy : 0.7185

       'Positive' Class : Yes
```

- Of all customers where we predicted churn, ~65% actually churned
- Of all customers that actually churned, we only correctly predicted about half (~54%)

Summary

- Overfitting occurs when the model fits the training data too well that it does not generalize to the population
- Regularization is a useful way to address overfitting for when we have a lot of features and lasso is especially useful because of automatic feature selection
- Results between lasso and logistic models for the churn dataset were nearly identical, with lasso doing slightly better in recall

Thank You!