Regression Analysis

Regression Analysis in Practice

Nicoleta Serban, Ph.D.
Professor
School of Industrial and Systems Engineering

Predicting Customer Churn

About This Lesson

Prediction

```
## Using the full model
pred.full = predict(full.model.red, newdata =
test.reduced, type = "response")
## Using the model from stepwise selection
## Variables not selected : Gender, Senior Citizen, Online
Backup and Device Protection
pred.step = predict(step.model, newdata =
test.reduced, type = "response")
## Using the model from LASSO
## Variables not selected : Online Backup and Payment
Method
pred.lasso = predict(lasso.retrained, newdata =
as.data.frame(new_test), type = "response")
## Using the model from Elastic Net (All selected)
pred.elnet = as.vector(predict(elnet.model, newx =
x.test, type = "response", s = cv.elnet$lambda.min))
```

Use classification threshold = 0.5

Predicted churn probability < 0.5 => Churn prediction = 0
Predicted churn probability > 0.5 => Churn prediction = 1

Prediction (cont'd)

```
## Using the full model
pred.full = predict(full.model.red, newdata =
test.reduced, type = "response")
## Using the model from stepwise selection
## Variables not selected : Gender, Senior Citizen, Online
Backup and Device Protection
pred.step = predict(step.model, newdata =
test.reduced, type = "response")
## Using the model from LASSO
## Variables not selected : Online Backup and Payment
Method
pred.lasso = predict(lasso.retrained, newdata =
as.data.frame(new_test), type = "response")
## Using the model from Elastic Net (All selected)
pred.elnet = as.vector(predict(elnet.model, newx =
x.test, type = "response", s = cv.elnet$lambda.min))
```

Prediction Output

Customers in Test Data	Actual Churn Value	predClass. full	predClass. step	predClas. lasso	predClass. elnet
6	0	0	0	0	0
122	1	0	0	0	0
139	0	0	0	0	0
257	0	0	0	0	0
522	0	0	0	0	0
594	1	1	1	1	1
733	0	0	0	0	0
951	0	0	0	0	0
982	1	0	0	0	0
1078	0	0	0	0	0
1091	0	0	0	0	0
1094	0	1	1	1	1
1123	0	0	0	0	0
1161	1	0	0	0	0
1249	0	0	0	0	0
1429	0	0	0	0	0

Use classification threshold = 0.5

Predicted churn probability < 0.5 => Churn prediction = 0
Predicted churn probability > 0.5 => Churn prediction = 1

Classification Accuracy

Classification Evaluation Metrics

- Accuracy:
  Proportion of response values Y_i (churn value) predicted correctly
- Sensitivity (True Positive Rate):
  Proportion of responses with $Y_i = 1$ (customers who left the company) predicted correctly
- Specificity (True Negative Rate):
  Proportion of responses with $Y_i = 0$ (customers who remained with the company) predicted correctly

	Person who left the company	Person who remained with the company
Classified as churned	True Positive	False Positive
Classified as not churned	False Negative	True Negative

Model Comparison via Classification Evaluation Metrics

```
## Calculate the Accuracy, the Sensitivity and the Specificity metrics to
evaluate these models at 0.5 threshold
pred_metrics = function(modelName, actualClass, predClass) {
  cat(modelName, '\n')
  conmat <- confusionMatrix(table(actualClass, predClass))
  c(conmat$overall["Accuracy"], conmat$byClass["Sensitivity"],
    conmat$byClass["Specificity"])
}
##Full model
pred_metrics("Full Model",test$Churn.Value, predClass.full)
##Stepwise selection model
pred_metrics("Stepwise Regression Model",test$Churn.Value, predClass.step)
##Lasso model
pred_metrics("Lasso Regression Model",test$Churn.Value, predClass.lasso)
##Elastic Net model
pred_metrics("Elastic Regression Model",test$Churn.Value, predClass.elnet)
```

Model Comparison via Classification Evaluation Metrics

Full Model

Accuracy	Sensitivity	Specificity
0.8180	0.8577	0.6832

Stepwise Regression Model

Accuracy	Sensitivity	Specificity
0.8174	0.8582	0.6807

Lasso Regression Model

Accuracy	Sensitivity	Specificity
0.8168	0.8576	0.6799

Threshold value: 0.5

All models have very similar prediction metrics. In this case, correctly identifying positives is more important for us. Therefore, we should choose a model with higher Sensitivity.

Classification Evaluation Metrics: Different Threshold

Full Model

Accuracy	Sensitivity	Specificity
0.7742	0.9116	0.5521

Stepwise Regression Model

Accuracy	Sensitivity	Specificity
0.7776	0.9136	0.5567

Lasso Regression Model

Accuracy	Sensitivity	Specificity
0.7759	0.9111	0.5547

Threshold value: 0.3

All models have very similar prediction metrics. Sensitivity has improved while the specificity has decreased as well as the overall accuracy.

Goodness of fit

Measure how well the Logistic Regression model (after variable selection through Stepwise Selection) fits on the training data

 # Removing variables not selected by stepwise regression

```
step.predictors <-  names(coef(full.model.red)[index.step])
x.train <- as.data.frame(x.train)
train.final <- x.train[, - which(colnames(x.train) %in% step.predictors)]
```

 # Aggregating the data

```
obdata.agg.n = aggregate(y.train ~ . , data = train.final, FUN = length)
obdata.agg.y = aggregate(y.train ~ . , data = train.final, FUN = sum)
dat.aggr <- cbind(obdata.agg.y, total = obdata.agg.n$y.train)
```

Fitting the model

```
mod.aggr = glm(y.train / total ~ . , data = dat.aggr, weight = total, family = binomial)
summary(mod.aggr)
```

Georgia Tech

Goodness of fit (cont'd)

 # Find the Chi-square test statistics and the corresponding p-value to test the given null hypothesis.

```
res = resid(mod.aggr, type="deviance")
cbind(statistic = sum(res^2), pvalue = 1-pchisq(sum(res^2), mod.aggr$df.resid))
```

$$Test\ statistic = X^2 = \sum_{i=1}^{p} r_i^2 \sim \chi^2 with\ dof = n - (p+1)$$

$$P\ value = P(\chi^2_{n-(p+1)} < X^2)$$

Chi-Square Test Statistics	P-value
4180.503	1

P-value is equal to 1, so our model reasonably fits the training data.

Georgia Tech

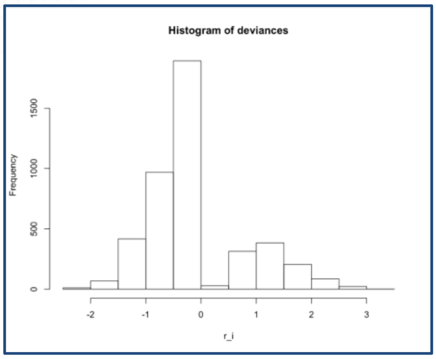
Goodness of fit (cont'd)

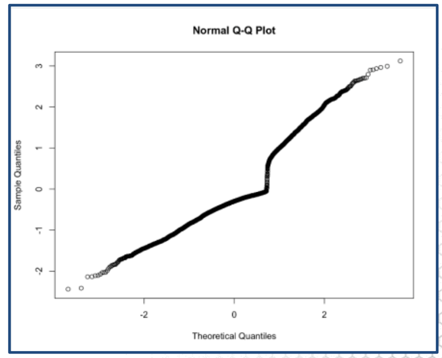
 # Checking the normality of deviance residuals assumption

hist(res, main="Histogram of deviances",breaks = 8,xlab = "r_i")

qqnorm(res)

Normality assumption seems to be violated due to bi-modality in the data.





Summary