+ Add article

Linear Regression

View
Edit
Changes

LAST MODIFIED:
June 15, 2015, 4:44 p.m.

See all children

The Method

Linear regression is used to determine how an outcome variable, called the dependent variable, linearly depends on a set of known variables, called the independent variables. The dependent variable is typically denoted by y and the independent variables are denoted by $x_1, x_2, \dots x_k$, where k is the number of different independent variables. We are interested in finding the best possible coefficients $\beta_0, \beta_1, \beta_2, \dots \beta_k$ such that our predicted values:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k$$

are as close as possible to the actual y values. This is achieved by minimizing the sum of the squared differences between the actual values, y, and the predictions $\hat{y}$. These differences, $(y - \hat{y})$, are often called error terms or residuals.

Once you have constructed a linear regression model, it is important to evaluate the model by going through the following steps:

- Check the significance of the coefficients, and remove insignificant independent variables if desired.
- Check the R^2 value of the model.
- Check the predictive ability of the model on out-of-sample data.
- Check for multicollinearity.

Linear Regression in R

Suppose your training data frame is called "TrainingData", your dependent variable is called "DependentVar", and you have two independent variables, called "IndependentVar1" and "IndependentVar2". Then you can build a linear regression model in R called "RegModel" with the following command:

```
RegModel = lm(DependentVar ~ IndependentVar1 + IndependentVar2, data = TrainingData)
```

To see the R^2 of the model, the coefficients, and the significance of the coefficients, you can use the summary function:

```
summary(RegModel)
```

To check for multicollinearity, correlations can be computed with the cor() function:

```
cor(TrainingData$IndependentVar1, TrainingData$IndependentVar2)
cor(TrainingData)
```

If your out-of-sample data, or test set, is called "TestData", you can compute test set predictions and the test set R^2 with the following commands:

```
TestPredictions = predict(RegModel, newdata=TestData)
SSE = sum((TestData$DependentVar - TestPredictions)^2)
SST = sum((TestData$DependentVar - mean(TrainingData$DependentVar))^2)
Rsquared = 1 - SSE/SST

In nutshell- Rsquared does three way comparision. SSE : Test data with respect to prediction from
model, SST :Test data with respect of training data.
```

Tips and Tricks

Quick tip on getting linear regression predictions in R posted by HamsterHuey (this post is about Unit 2 / Unit 2, Lecture 1, Video 4: Linear Regression in R)

Suppose you have a linear regression model in R as shown in the lectures:

```
RunsReg = lm(RS ~ OBP + SLG, data=moneyball)
```

Then, if you need to calculate the predicted Runs scored for a single entity with (for example) `OBP = 0.4`, `SLG = 0.5`, you can easily calculate it as follows:

```
predict(RunsReg, data.frame(OBP=0.4, SLG=0.5))
```

For a sequence of players/teams you can do the following:

```
predict(RunsReg, data.frame(OBP=c(0.4, 0.45, 0.5), SLG=c(0.5, 0.45, 0.4)))
```

Sure beats having to manually extract coefficients and then calculate the predicted value each time (although it is important to understand the underlying form of the linear regression equation.