BlackBox

By: Rohit Manjunath

New Columns | Encoding Categorical Columns

- Some machine learning models perform better with numerical columns. So let us convert our Sound and Switch columns into numerical.

Code:

```
decision <- rep(0,nrow(bb))

decision[bb$SWITCH == "Minimum"] <- 1

decision[bb$SWITCH == "Medium"] <- 2

decision[bb$SWITCH == "Maximum"] <- 3

bb$SwitchNum <- decision
```

Code:

```
decision2 <- rep(0,nrow(bb))

decision2[bb$SOUND == "Beep"] <- 1

decision2[bb$SOUND == "Gargle"] <- 2

decision2[bb$SOUND == "Hiss"] <- 3

decision2[bb$SOUND == "Kaboom"] <- 4

decision2[bb$SOUND == "Rumble"] <- 5

decision2[bb$SOUND == "Sizzle"] <- 6

bb$SoundNum <- decision2
```

- As you can see both columns range from 0 to 3(Switch) or 6(Sound).

Linear Regression

- Now we can use the linear regression model since we have the data completely in numerical values.

  Code:

  ```
  trainLm <- lm(SoundNum ~ INPUT1 + INPUT2 + INPUT3 + INPUT4 + SWITCH, data =
  train)

  predLm <-  predict(trainLm, newdata = test)

  decision3 <- rep(0, nrow(test))

  decision3 <- round(predLm)

  error <- mean(test$SoundNum != decision3)
  ```

- The 'error' rate is around 80%. We can get a better model than this. Hence, let's keep this as our base model.

LDA

- Using the LDA model let us pass in the parameters and the predicting column and see how it fares by calculating it's error.

  Code:

  ```
  trainLda <- lda(SOUND ~ INPUT1 + INPUT2 + INPUT3 + INPUT4 +
  SWITCH, data = train)

  predLda <- predict(trainLda, newdata = test)$class

  error <- mean(test$SOUND != predLda)
  ```

- The 'error' rate is around 44%. This error rate is much better than our Linear regression model's error rate. However, We can get a better model than this.

Neural Net

- Using the Neural Net model let us pass in the parameters and the predicting column and see how it fares by calculating it's error.

  Code:

  ```
  trainNn <- nnet(SoundNum / 6 ~ INPUT1 + INPUT2 + INPUT3 + INPUT4 + SWITCH, data =
  train, size = 5)

  predNn <- predict(trainNn, newdata = test) * 6

  decision4 <- rep(0, nrow(test))

  decision4 <- round(predNn)

  error <- mean(test$SoundNum != decision4)
  ```

- We divide 'SoundNum' by 6 because 'SoundNum' ranges from 0 to 6. We need to normalize to give us a range between 0 and 1.
- The 'error' rate is around 80%. This error rate is not better than our LDA model's error rate.

Decision Tree

- Using the Decision Tree model let us pass in the parameters and the predicting column and see how it fares by calculating it's error.

  Code:

  ```
  tree <- rpart(SOUND ~ INPUT1 + INPUT2 + INPUT3 + INPUT4 +
  SWITCH, data = train)

  rpart.plot(tree)

  predtree <- predict(tree, newdata = test,  type = "class")

  error <- mean(test$SOUND != predtree)
  ```

- The 'error' rate is around 34%. This error rate is much better than our all our other model's error rate.

Cross Validation

- Using different values of cp, minbucket, and minsplit I could come up with a better model, by using a cp value of 0.001.

  Code:

  ```
  tree <- rpart(SOUND ~ INPUT1 + INPUT2 +
  INPUT3 + INPUT4 + SWITCH, data = train,
  control = rpart.control(cp = .001))

  cross_validate(bb, tree, 5, 0.8)
  ```

	accuracy_subset	accuracy_all
1	0.6779778	0.6658587
2	0.6755540	0.6450831
3	0.6762465	0.6554709
4	0.6693213	0.6398892
5	0.6686288	0.6475069

- Cross Validating shows us that this control parameter gives us better results than the 'accuracy_all'

Conclusion

- We can conclude that out of the all the models tried, decision tree gives the best result as it has the lowest error rate.

Thank you!