Assignment_10.3_HillZach

Zach Hill

May 20, 2019

Fitting the Model

```
glm_data <- glm(formula = label ~ x + y, data = data, family = binomial)

glm_data
```

```
##
## Call:  glm(formula = label ~ x + y, family = binomial, data = data)
##
## Coefficients:
## (Intercept)            x            y
##    0.424809    -0.002571    -0.007956
##
## Degrees of Freedom: 1497 Total (i.e. Null);  1495 Residual
## Null Deviance:       2076
## Residual Deviance: 2052  AIC: 2058
```

```
summary(glm_data)
```

```
##
## Call:
## glm(formula = label ~ x + y, family = binomial, data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3728  -1.1697  -0.9575   1.1646   1.3989
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.424809   0.117224   3.624  0.00029 ***
## x           -0.002571   0.001823  -1.411  0.15836
## y           -0.007956   0.001869  -4.257 2.07e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2075.8  on 1497  degrees of freedom
## Residual deviance: 2052.1  on 1495  degrees of freedom
## AIC: 2058.1
##
## Number of Fisher Scoring iterations: 4
```

A: Accuracy using caret

```
train(as.factor(label) ~ x + y, data=data, trControl = trainControl(method = "cv"), method = "svmRadial
```

```
## Support Vector Machines with Radial Basis Function Kernel
```

```
##
## 1498 samples
##    2 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1348, 1347, 1349, 1349, 1349, 1348, ...
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##   0.25  0.8919140  0.7838055
##   0.50  0.9119142  0.8239470
##   1.00  0.9279144  0.8558235
##
## Tuning parameter 'sigma' was held constant at a value of 1.3056
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 1.3056 and C = 1.
```

B: Comparison to KNN

The model's accuracy is similar (but higher) to what we saw with KNN, according to caret's train function as shown above. This was not the case below when attempting to use probability to predict a label. I'm unsure where my mistake is being made.

```
data$model_prob <- predict(glm_data, data, type="response")

data_ratio <- sum(data$label == 1) / nrow(data)

data <- data %>% mutate(model_pred = 1*(model_prob > .48) + 0)
data <- data %>% mutate(accurate = 1*(model_pred == label))

sum(data$accurate)/nrow(data)
```

```
## [1] 0.5160214
```

C: Describing a difference

It could be that knn works better on a more linearly separable dataset where SVM (as was used in the train function above) is not as concerned with a linear relationship. These data might have had less linear correlation than would be required for a better fit from knn. This is just speculation as I could not reproduce the results manually.