KERNEL-BASED MACHINE LEARNING AND MULTIVARIATE MODELLING: Kernel PCA

Ricard Meyerhofer Parra

4/10/2019

Dataset

In this problem we are going to use a dataset from Kaggle which is named Social Network Ads. This dataset is a categorical dataset to determine whether a user purchased a particular product or not. The variables contained in the dataset, are the following:

Variable	Description	Attribute type
User ID	Id of the user	Numeric
Gender	Gender of the user either Male or Female	Categorical
Age	Age of the user	Numeric
Estimated Salary	Income of the user in dollars	Numeric
Purchased	Response variable	Categorical

The dataset is complete which means that does not have any missings. This does not imply that it does not have outliers but we are going to suppose that there are none. Because of it, we are not going to do any kind of imputations or modifications to our initial data.

Once we have read the dataset, we are going to split the dataset composed of 401 elements in train and test with a proportion of 75-25 respectively. Also we are going to scale the dataset both, test and train.

```
# Splitting into the Training and Test
library(caTools)
set.seed(123)
split <- sample.split(dataset$Purchased, SplitRatio <- 0.75)
training_set <- subset(dataset, split == TRUE)
test_set <- subset(dataset, split == FALSE)
# Scaling
training_set[, 1:2] <- scale(training_set[, 1:2])
test_set[, 1:2] <- scale(test_set[, 1:2])
```

Kernel PCA

Once we have read our dataset, now we are going to apply the Kernel PCA. We will choose $rbfdot$ kernel as is the one that suits best our data. We need to execute the following function which calculates the eigenvalues

```
library(kernlab)
kpca <- kpca(~., data = training_set[-3], kernel = 'rbfdot', features = 2)
training_set_pca <- as.data.frame(predict(kpca, training_set))
training_set_pca$Purchased <- training_set$Purchased
test_set_pca <- as.data.frame(predict(kpca, test_set))
test_set_pca$Purchased <- test_set$Purchased
```

Now we are going to fit logistic regression to the training set and later we are going to predict on the result set.

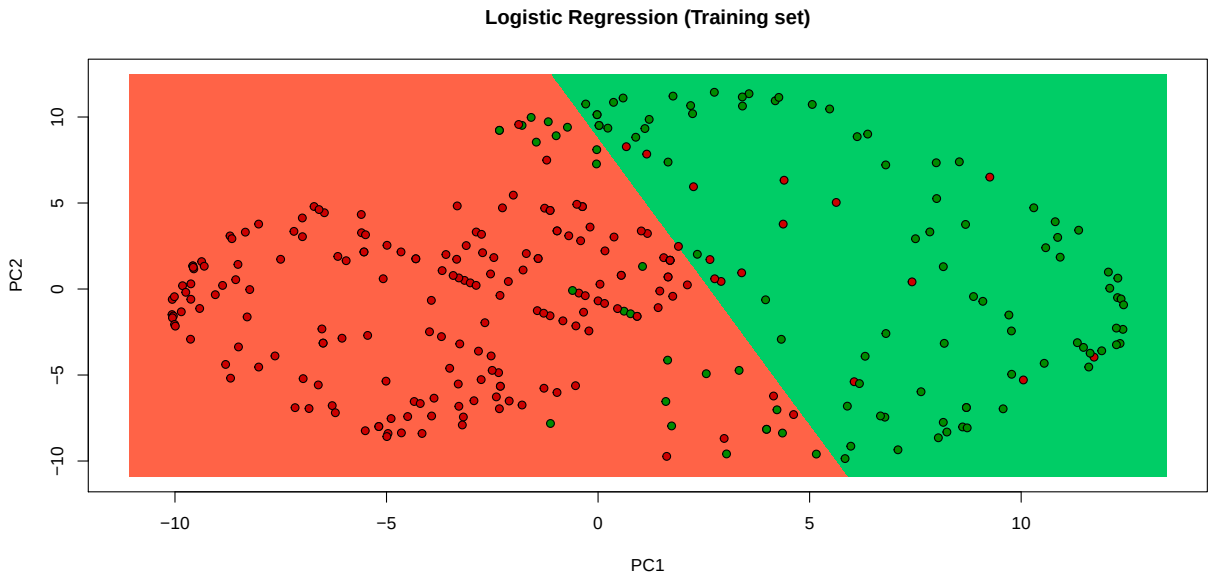
Below we can see that our accuraccy is the following is pretty good as we can see in the confusion matrix:

```r
# Predicting the Test set results
prob_pred <- predict(classifier, type = 'response', newdata = test_set_pca[-3])
y_pred <- ifelse(prob_pred > 0.5, 1, 0)
# Making the Confusion Matrix
cm <- table(test_set_pca[, 3], y_pred)
kable(cm)
```

	0	1
0	57	7
1	10	26

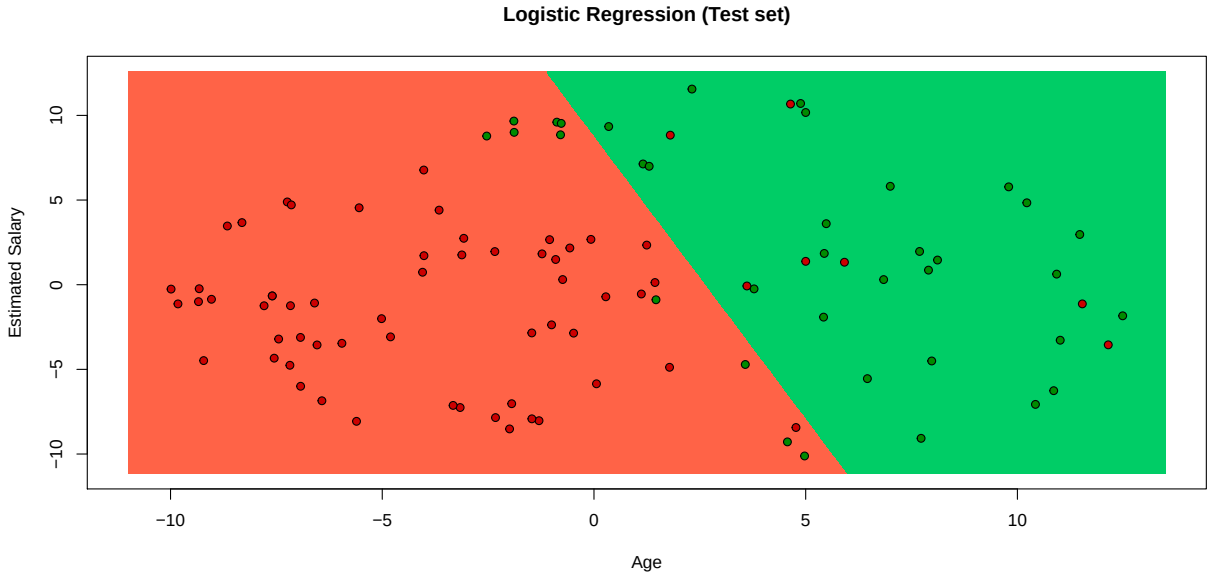
Training results

Here we can see our training results where we can see that we have not build a very specific model and it looks generic enough to perform good in the test subset.



Test results

As we can see in the output below, our model seems to perform quite well


Logistic Regression (Test set)
Estimated Salary
Age
−10
−5
0
5
10
−10
−5
0
5
10