Introduction to Machine Learning Project Report

Iira Häkkinen and Tuomo Artama

09 December, 2022

1. Introduction

In this report we present an R implementation of a multi-class classifier to predict if different types of new particle formation (NPF) events occur on a given day. An NPF event happens, when smaller particles start to form larger particles that affect cloud formation (Kerminen et al. 2018). There are three types of NPF events: Ia, Ib and II (Kerminen et al. 2018). There is also a possibility that an NPF event does not happen on a given day.

Our multi-class classifier uses logistic regression with RIDGE regularization. The classifier was trained on a dataset containing various atmospheric parameters (such as temperature, CO2 measurements, etc.) on multiple days throughout the years 2000-2011. Our estimate for the accuracy of the classifier, calculated on the training data, is approximately 0.9202586. It is notable, that this estimate may differ greatly from the actual testing accuracy, when our data is tested on a new test dataset.

2. Methods Used

Firstly, it is important to note that since the beginning of this project, we were aware of how limited time we have available for this project. This time limitation had a large effect on our choices throughout the project. This time limitation was kept in mind, so that we could perform as well as we could with the time that we had. Especially, it is the main reason behind our model selection. We chose a simple model, so that we would spend less time debugging the model, and more time on analyzing the results and writing a good report.

When choosing the classifier, we recollected what kind of machine learning classifiers we are aware of, and especially how well we know their implementations in R. We chose the linear regression model as our classifier, but we also talked over the Naive Bayes classification as well as the k-nearest neighbour classification. We came to the conclusion, that the Naive Bayes assumption is probably not valid with this dataset and that we do not know anything about implementing the kNN-classifier in R. We ended up choosing linear regression, because we were familiar with it's implementation in R and it is straightforward to use.

2.1 Implementation

Our multi-class classifier is implemented in R using logistic regression. The classifier is implemented using the glmnet-regressor provided by the library glmnet and the implementation of the model can be seen in the code block below (for full implementation, see the appendix of this report). We chose to use RIDGE regularization with our logistic regression model. Linear regression might not be the the optimal classifier for this task, but provides better values than just guessing the class of a given day.

```r
model <- glmnet(class4 ~ ., npf, family = "multinomial", alpha = 0, lambda = 0.01)
```

The choice of RIDGE as the regularization method was done by simply trying different values by hand for both RIDGE and LASSO regularization. By manually changing the values of the *alpha* and *lambda* parameters in the glmnet-model in R, we came to the conclusion, that the better values for the accuracy were achieved by using RIDGE regularization. We chose regularization over no regularization, because by testing both, we noticed that the accuracy was better with regularization.

3. Results

We have not yet had time to thoroughly analyze the results. From the *answers.csv* file we can see, that it appears that our classifier seems to be working at least better than a 50/50-classifier. We will do a division of the training dataset into training and validation sets, so that we may perform cross-validation for our classifier and get proper results to analyze. We will then at least analyze:

- cross-validation and cross-validation error
- accuracy and perplexity
- how the regularization (LASSO, RIDGE) affects the resulting accuracy and perplexity
- other types of models and how they perform in respect to each other (**if we have time**)
    - choose the best out of these as our classifier

4. Discussion

Our estimate for the accuracy of the classifier may not be reliable, since it is formed solely on the same dataset, that the classifier was trained on. We will improve our estimate for the accuracy by using cross-validation before the final deadline.

5. Conclusion

To be added.

6. Grading

To be added.

References

Kerminen, Veli-Matti, Xuemeng Chen, Ville Vakkari, Tuukka Petäjä, Markku Kulmala, and Federico Bianchi. 2018. “Atmospheric New Particle Formation and Growth: Review of Field Observations.” *Environmental Research Letters* 13 (10): 103003.

Appendix

The R implementation of our multi-class classifier is provided here.

```
options(error = function() traceback(3))

library(glmnet)
library(glmnetUtils)

npf <- read.csv("npf_train.csv")
npf_test <- read.csv("npf_test_hidden.csv")

# Remove columns "id", "date" and "partlybad" and set dates as rownames
rownames(npf) <- npf[, "date"]
npf <- npf[, -c((1:2), 4)]


# Some helpful functions
accuracy <- function(real, pred) {
    sum(as.numeric(real == pred)) / length(pred)
}

# Create a linear model
model <- glmnet(class4 ~ ., npf, family = "multinomial", alpha = 0, lambda = 0.01)

coefficients(model)

# Create variable class2 where the value is "event" if there was an NPF event
# and "nonevent" otherwise.
npf$class2 <- factor("event", levels = c("nonevent", "event"))
npf$class2[npf$class4 == "nonevent"] <- "nonevent"

# Predict class4 on training data
predicted <- predict(model, newdata = npf, type = "response")[,,1]

predicted_class4 <- c()
predicted_class2 <- c()
predicted_probs <- c()
for (i in 1:length(predicted[, 1])) {
    pred_class <- colnames(predicted)[which.max(predicted[i, ])]
    predicted_class4 <- c(predicted_class4, pred_class)
    event_prob <- 1 - predicted[i, 4]
    predicted_probs <- c(predicted_probs, event_prob)
    predicted_class2 <- c(predicted_class2, c("nonevent", "event")[1 + (event_prob > 0.5)])
}

acc <- accuracy(npf$class4, predicted_class4)

cat("Multiclass accuracy: ")
cat(acc)
cat("\n")


binary_acc <- accuracy(npf$class2, predicted_class2)
```

```r
cat("Binary accuracy: ")
cat(binary_acc)
cat("\n")

# Predict class4 on test data
predicted <- predict(model, newdata = npf_test, type = "response")[,,1]

predicted_class4 <- c()
predicted_class2 <- c()
predicted_probs <- c()
for (i in 1:length(predicted[, 1])) {
    pred_class <- colnames(predicted)[which.max(predicted[i, ])]
    predicted_class4 <- c(predicted_class4, pred_class)
    event_prob <- 1 - predicted[i, 4]
    predicted_probs <- c(predicted_probs, event_prob)
    predicted_class2 <- c(predicted_class2, c("nonevent", "event")[1 + (event_prob > 0.5)])
}


write.csv(data.frame(class4=predicted_class4, p=predicted_probs), "answers.csv", row.names=F, quote=F)
```