ML–DS I: Project II Report

Sebastijan Trojer
ML-DS I 2024/25 , FRI, UL
st5804@student.uni-lj.si

I. Introduction

In this project we focused on robust model evaluation techniques. We utilized a basketball shot dataset to compare 3 different models, and focused on robust parameter optimization of one of them. Experiments include regular cross validation (CV), nested CV and CV where we optimize training fold performance. Furthermore, we investigated the dependence of performance on different values of distance feature and discrepancies between supposed data generating process (DGP) competition type frequencies and sample competition type frequencies and the effect it has on the model performance. We omitted any preprocessing except encoding of the categorical values.

II. Methodology and results

A. Part 1: Baseline Classifier and Logistic Regression

In the first part of the experiment, we evaluated two models: a Dummy classifier that predicts class probabilities based on the relative frequencies of classes, and a logistic regression classifier. We evaluated their performance using 5-fold stratified cross-validation (CV) to ensure each fold reflects the overall class distribution, based on the assumption that our sample is representative of the data generating process (DGP). The performance metrics used were log score and classification accuracy.

B. Part 2: Support Vector Machine (SVM)

The third model evaluated was a support vector machine (SVM) classifier. For the entire evaluation process we fixed the number of iterations to 200 as it provided a good balance between speed and classification accuracy. In addition to the standard 5-fold stratified CV, we evaluated the SVM using two approaches: optimizing training fold performance and nested cross-validation. During evaluation we tuned the following parameter values:

- Kernel: We tested SVM with linear and radial basis function (RBF) kernel.
- Regularization parameter (C): We tested the following parameter values - 0.01, 0.1, and 1
- In case of RBF kernel we also tested different γ parameter values - 0.01, 0.1, 1 and ‘scale‘ value, which is computed as $\frac{1}{(n_{features} * \sigma(X))}$.

Optimizing Training Fold Performance: We first defined a parameter grid for hyperparameter tuning as described above. For each fold, the model was trained on $k-1$ folds and evaluated on those same training folds for each set of parameters. Then the model with best performance on the same train set was evaluated on the k^{th} fold. The best average performance was reported.

Nested Cross-Validation: Nested CV was used to obtain a more robust estimate of the model’s generalization performance. For each outer fold, we performed an inner CV on the $k-1$ training folds to select the best hyperparameters. Specifically, within each outer fold, the inner loop involved training on $k-2$ folds and validating on the remaining fold. The optimal parameter set from the inner loop was then used to train the model on the entire outer training set, and the model was subsequently evaluated on the outer test fold.

The results of the experiments are reported in Table I. All scores were bootstrapped across the folds.

As expected the baseline classifier achieved the worst performance. The next interesting thing is that while logistic regression achieved a better log score than SVM, the latter achieved better accuracy, indicating that SVM was overconfident in its predictions so the penalty on error was higher, which is also reflected by high standard error (SE) relative to that of logistic regression. A similar, but more extreme pattern appears if we compare the train fold performance optimization SVM and the nested CV SVM - the former achieved a better log score, however its accuracy is significantly lower, indicating severe overfitting. The nested CV version is also a lot more stable, accuracy wise, as indicated by the low SE.

C. Part 2: Error-Distance dependency and Competition type frequency discrepancy

In the second part, we were dealing with 2 problems: Firstly we suspected that the classification error depends on Distance. We explored that by plotting log score against the distance values, which is shown on Figure 1. We notice that the mean log score grows with distance, up to about 6 meters, but then the trend turns around. The reason for this might be that up to some distance the shots are easier to predict (more misses) but the model doesn’t get too confident in its predictions, however when the distance is even higher, the model makes very confident predictions and the log score is penalized more when the

Model	Evaluation Strategy	Mean log score	Mean accuracy [%]
Baseline classifier	CV	-20.043 ± 0.057	41.97 ± 0.17
Logistic regression	CV	-0.838 ± 0.010	70.19 ± 0.23
SVM	CV	-0.972 ± 0.103	71.18 ± 4.25
SVM	Train fold performance optimization	-1.197 ± 0.013	28.40 ± 1.78
SVM	Nested CV	-1.265 ± 0.109	66.58 ± 0.01

TABLE I
Model evaluation results.

prediction is wrong. To check this further, we bootstrapped the errors and computed the 95% confidence interval of the correlation, which came out to $[0.27, 0.29]$, with the mean of 0.28 ± 0.006, confirming a positive correlation between log score and distance.

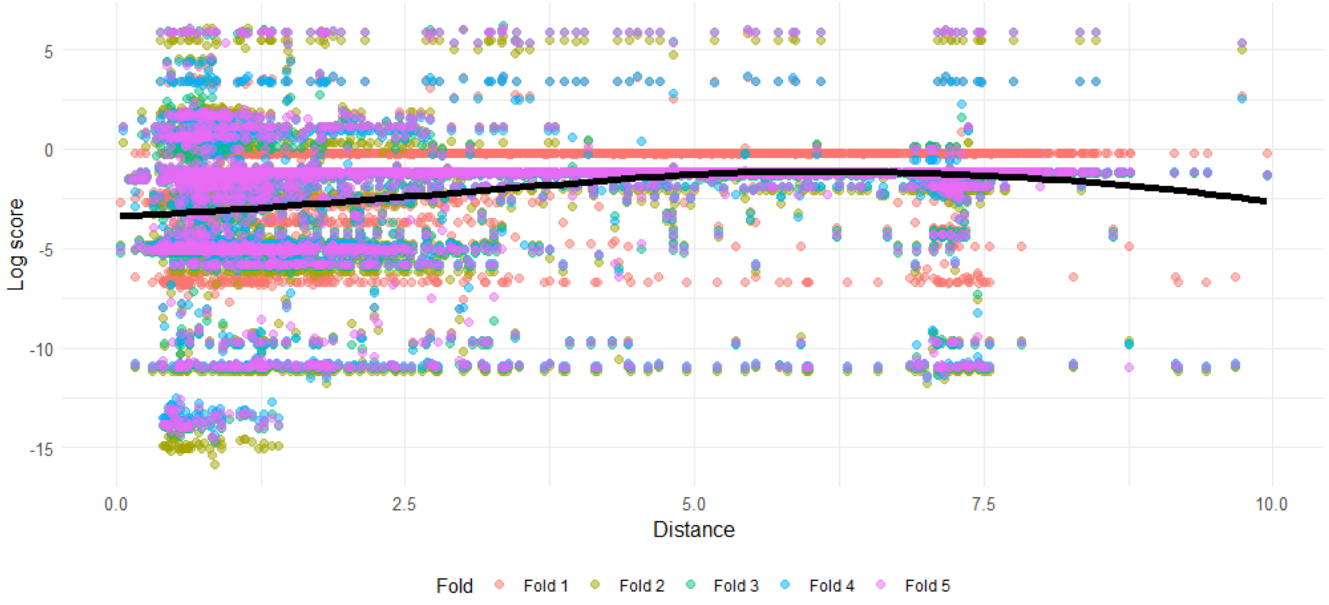

Fig. 1. Log score vs. Distance. Different colors indicate different folds. The black line indicates the mean log score.

Secondly, we learned that the dataset is not entirely representative of the DGP - the true relative frequencies of Competition types differed from the ones in our sample. Therefore we conducted an experiment where we computed the log score based on the empirical relative frequencies of the competition types and the assumed DGP frequencies. The mean unweighted log score was -2.10 ± 0.61 and the mean weighted log score was -2.08 ± 0.61, so slightly better. To get a more robust estimation we bootstrapped the scores and computed 95% confidence intervals again. For the unweighted data we got a confidence interval $[-2.14, -2.06]$ and for the weighted data $[-2.13 - 2.04]$, which is still slightly better than the unweighted. Given the results we confirmed that we slightly underestimated the performance, due to an unrepresentative sample.

III. Conclusion

In this project we tackled the challenge of robust model evaluation. We employed 3 different types of cross-validation and outlined the advantages and disadvantages of each on our specific use case. We also dealt with the problem of the sample being unrepresentative of DGP in 2 different ways and used bootstrapping to ensure a robust evaluation of the performance of our model.