Best Practices for Machine Learning with Small Datasets Using Empirical Evaluations

Ashwin Krishnamurthi

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

A dissertation presented in part fulfilment of the requirements of the Degree of Master of Science at The University of Glasgow

01/09/2023

Abstract

In real-world applications of machine learning, small datasets are commonly used due to the often time consuming and research intensive process of collecting and labeling large amounts of data. There are a host of challenges associated with tuning and evaluating models on such limited datasets. As a result, many studies have proposed various forms of cross validation as a means to build the best possible machine learning system under these conditions. This project aims to evaluate numerous prominent cross validation techniques in order to establish best practices when working with small datasets. The performance of two binary classification algorithms using these cross validation techniques were analyzed over repeated trials on a simulated dataset. Results suggest that variants of K-Fold cross validation and Monte Carlo cross validation can improve model performance compared to traditional cross validation. For evaluating models on small datasets, traditional cross validation and Monte Carlo cross validation demonstrated consistent performance across various dataset sizes. An analysis of the number of folds in K-Fold cross validation indicated that 10-Fold and 20-Fold cross validation underestimated model F1 scores significantly on smaller datasets, likely due to the high proportion of folds relative to the dataset size. The sizable underestimation of average F1 estimate differences when implementing Leave-One-Out cross validation further supported this observation. As a result of this effect, Leave-One-Out cross validation may not be a suitable alternative to traditional cross validation.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: ____________________ Signature: ____________________

Acknowledgements

I would like to express my deep gratitude to Dr. Jake Lever for his help over the course of these last few months. This project would not have been possible without his guidance and constructive advice.

Contents

Chapter 1: Introduction

1.1 Motivation

In machine learning, large datasets are often imperative to developing and training models that are both robust and accurate. However, collecting large volumes of data can be a very costly process that requires expert annotators and complex pipelines to ensure data integrity. As a consequence, the challenge that confronts many working in the field today is that practitioners are frequently required to work with very small datasets in the context of most real-world applications. This can make the process of developing machine learning models particularly laborious as there is a delicate balance that must be struck between training models on enough available data to facilitate learning while also ensuring that the model is not overfitted [20]. In these such cases where the size of the available dataset is limited, one of the most widely used evaluation methods for determining the effectiveness of a classification model is K-Fold cross validation [8, 17].

Since its rise in popularity, numerous variations of K-Fold cross validation have arisen along with other forms of empirical evaluation such as Monte Carlo cross validation. However, a wealth of available choices can also bring uncertainty. The tuning of hyperparameters such as number of folds and sometimes conflicting research related to the effectiveness and high computational expenses of these techniques can make choosing the ideal evaluation method a complicated task. This project investigates the efficacy of these different techniques in the hopes of streamlining this decision making process.

1.2 Aims and Objectives

The aim of this project is to identify best practices when working with small datasets amongst the plethora of prominent empirical evaluation techniques available today. While there is an abundance of literature regarding the benefits and drawbacks of these evaluation methods, much of it concerns large datasets in the context of Big Data problems.

Consequently, the objectives of this project are as follows:

- Develop numerous models using the leading cross validation approaches on simulated datasets of various sizes.
- Evaluate the effectiveness of these different evaluation techniques through repeated trials.
- Perform an analysis on the impact of the number of folds ('k') or splits in relation to the efficacy of K-Fold and Monte Carlo cross validation respectively.

1.3 Outline

There are five chapters in this report. Chapter 1 outlines the motivation as well as the aims and objectives of this project. Chapter 2 contains a background survey and analysis of cross validation techniques. Chapter 3 discusses the overall design of the project as well as an overview of how it was implemented. Chapter 4 examines the results of experiments as well as other findings. Lastly, Chapter 5 contains the conclusion of the report along with possible directions for future work.

Chapter 2: Analysis

2.1 Background Survey

The surveyed literature discusses a variety of evaluation techniques and their efficacy in different scenarios. The foundations of many of the key concepts are discussed here as they are the framework of this project. An analysis of any known limitations of these various methods are also included in order to understand the prevailing beliefs among researchers concerning this topic.

2.1.1 Supervised Learning

The fundamental idea behind supervised learning is to train models using existing labeled data in order to make predictions on unseen data. Classification is one form of supervised learning that works to understand underlying patterns in historical training data in order to accurately categorize new data. By learning the patterns instead of memorizing the data, an ideal classification model trained using supervised learning will be able to make reliable predictions by ignoring meaningless information (noise) in the training set. Accordingly, one of the biggest challenges in machine learning is ensuring that models adequately learn from training data without overfitting, which can result in poor performance on new data [3].

2.1.2 Traditional (Hold-out) Cross Validation

Traditional (also called hold-out) cross validation is one technique to assess how well a machine learning model is performing. Traditional cross validation involves splitting an available dataset into training and testing subsets as shown below in Figure 1. Models learn patterns only on the training data and are then evaluated on the test set. The logic behind this train-test split is to gauge how accurately the model will perform on unseen data by ”holding-out” some of the data explicitly for testing purposes. The performance of the model on the test set is then an estimate for how well the model will behave in real-world scenarios.

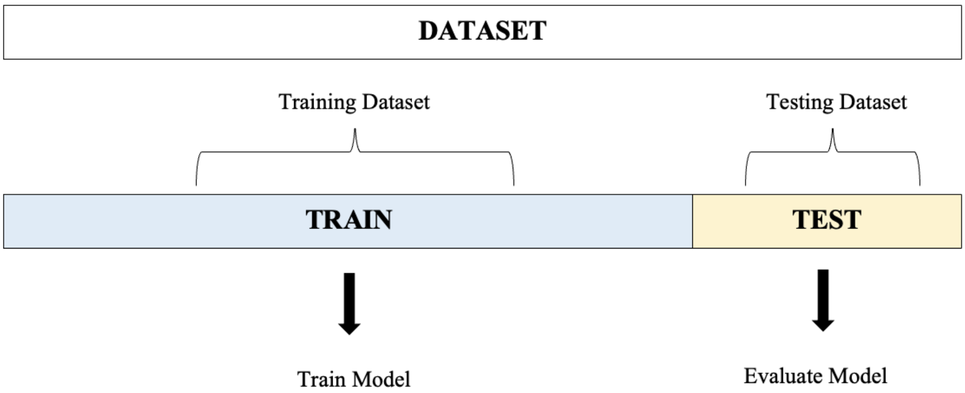


Figure 1: Train-Test Split in Hold-out Cross Validation

When using small datasets, however, there are some noted limitations of traditional cross validation. With limited data, dividing into training and test sets can lead to models that are not well trained or evaluated due to a scarcity of data that is used during both processes. There can be significant variability in the assessment of the performance of a model if the subset of test data does not accurately represent the dataset as a whole. Similarly, if the training set contains a disproportionate amount of outliers or noise, the model cannot learn the underlying patterns of the dataset as a result of the bias in the data split. Ultimately, by further dividing already small datasets, it can be difficult to develop robust models and accurately measure their effectiveness.

2.1.3 K-Fold Cross Validation

The process of K-Fold cross validation entails partitioning the dataset into multiple subsets (or "folds") in which each subset is used iteratively as both a training and validation set [2]. The benefit, especially when considering smaller datasets, is that compared to the traditional holdout method of a single train-test split of data, K-Fold cross validation can more accurately assess performance of a model on unseen data since every instance of the original dataset is used as part of both training and testing. This claim has been defended by Yadav and Shukla (2016) [20] and Blum et al. (1999) [4] which concluded that K-Fold cross validation was more effective than traditional hold-out validation for very large datasets. Generally speaking, the higher the number of folds ("k"), the higher the accuracy and risk of overfitting in cross validation [20, 5]. An example diagram of K-Fold cross validation with five partitions (k=5) is illustrated below in Figure 2.

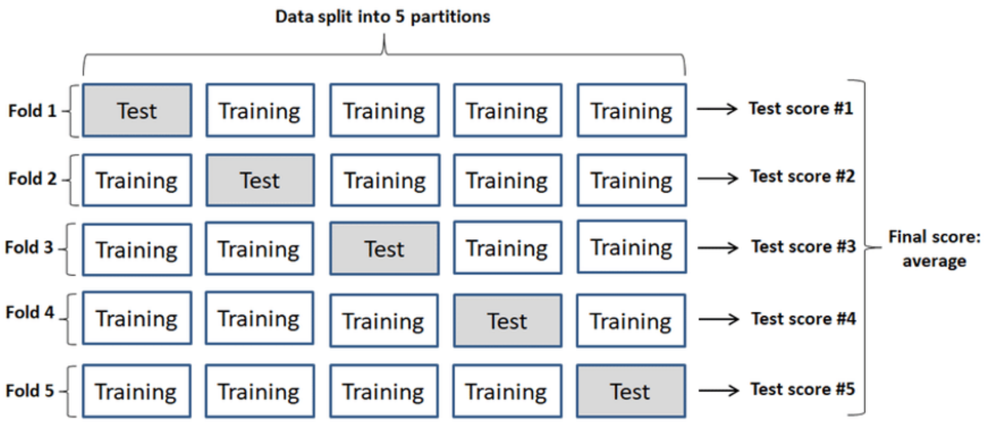


Figure 2: K-Fold Cross Validation Visualization [13]

2.1.4 Leave-One-Out Cross Validation (LOOCV)

Early literature in the 1970s established K-Fold cross validation as a viable means to estimate the prediction error of a model (see: [17, 1]) while more contemporary research has explored other variations of cross validation. Leave-One-Out cross validation (LOOCV) is one popular variant and an extreme version of K-Fold cross validation in which 'k' is equal to the total number of observations in the dataset. LOOCV is an attractive estimator for model selection because it can provide a nearly unbiased estimation of the generalization ability of a classifier [6].

However, the large number of folds required to carry out LOOCV makes it a very computationally expensive alternative to K-Fold cross validation that can also be susceptible to higher variation [6, 7]. Work by Rao and Fung (2008) [15] also indicates that LOOCV underestimates true error when the number of algorithms is large or there is high dimensionality in the dataset. Figure 3 below demonstrates how this effect is exacerbated when working with small datasets. Selection of unnecessary features and risk of overfitting due to this underestimation of true error is a known limitation of LOOCV that is difficult to guard against even when restricting the number of selected features [15].

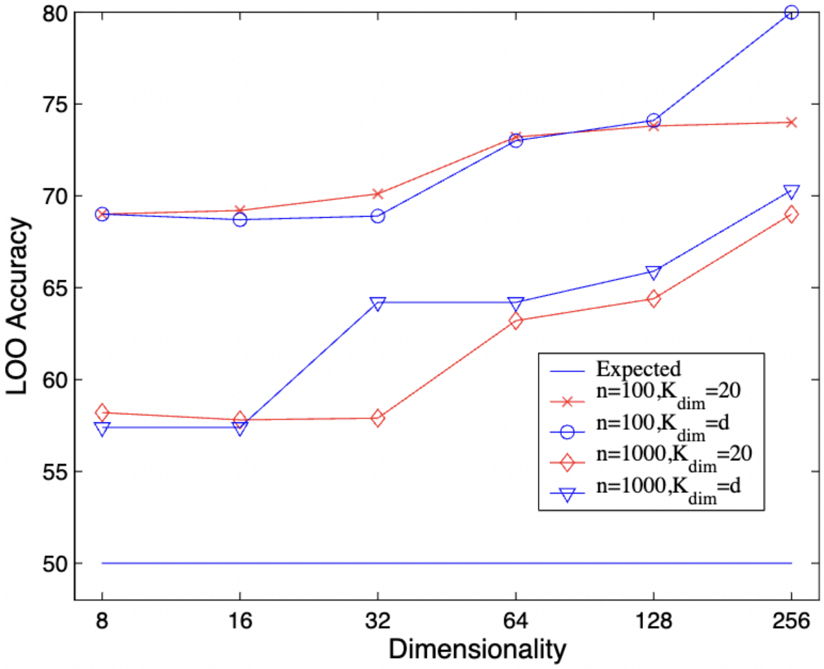


Figure 3: Overestimation of Accuracies for Small Datasets with High Dimensionality [15]

2.1.5 Monte Carlo Cross Validation (MCCV)

Other alternatives such as leave-k-out and Monte Carlo cross validation (MCCV) are investigated by Wang (2022) [18] and Shao (1993) [16] which conclude that both techniques can rectify some of the inconsistencies introduced by LOOCV. First appearing in a publication by Picard and Cook (1984) [14], MCCV involves randomly shuffling the dataset and generating multiple train-test splits in each iteration in the hopes of a more holistic estimate of model performance as shown below in Figure 4 [19]. Shao (1993) [16] asserts that when analyzing the probabilities of selecting the optimal model, Monte Carlo cross validation significantly outperformed LOOCV. Xu (2004) [19] reinforces these findings, while noting that some limitations exist when performing MCCV as an evaluation method. For example, on average, MCCV can overestimate the prediction error of a model due to the possibility of observations in the dataset overlapping in both training and testing sets across different iterations [19].

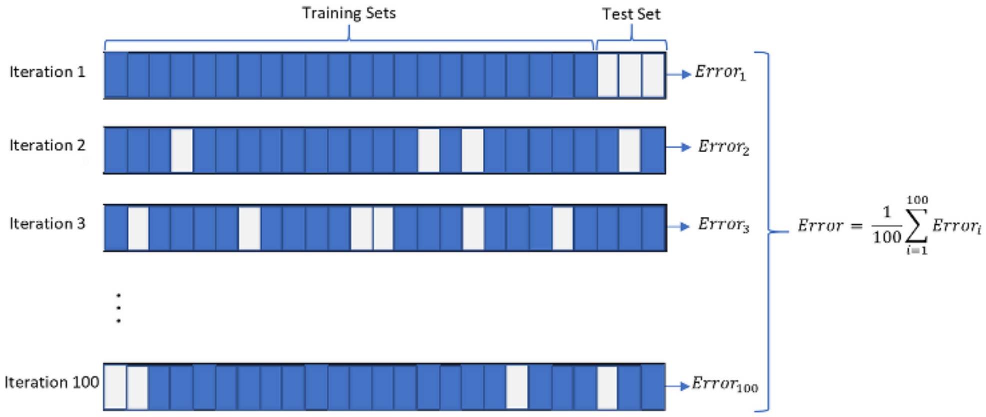


Figure 4: Monte Carlo CV, Iterations = 100 [12]

2.1.6 Optimal Number of K-Folds

There is also a plethora of existing literature that seeks to determine the optimal number of folds to use in K-Fold cross validation. Kohavi (1995) [9], for example, is a frequently cited publication that asserts that ten-fold cross validation is the best method to use for model selection when considering real-world datasets. The paper contends that for two and five folds, bias was high while when the number of folds were greater than ten, variance also increased [9]. Thus, a moderate 'k' value close to ten was most favorable in achieving a tradeoff between these errors [9]. More recent studies (see: [10, 11]) reinforced Kohavi's findings and determined that a higher number of folds (k $>$ 10) required greatly increased computational resources while providing no clear improvement to prediction performance as illustrated in Figure 5 below.

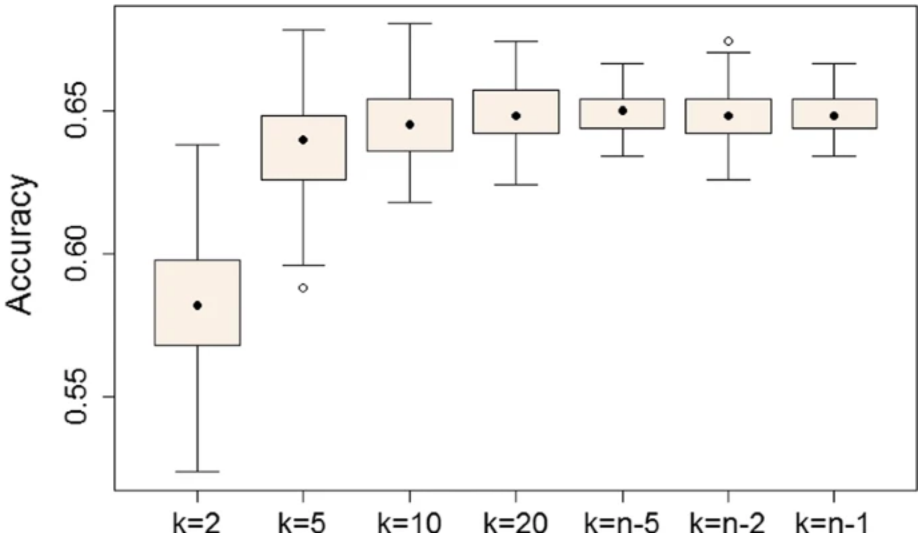


Figure 5: Overall Classification Accuracies At Different K-Folds [10]

2.2 Research Objectives

Related work in the literature survey conducted identified the key evaluation techniques when working with small datasets as:

- Traditional Cross Validation
- K-Fold Cross Validation
- Leave-One-Out Cross Validation
- Monte Carlo Cross Validation

This project will aim to evaluate these different methods across datasets of different sizes (particularly tiny datasets) with the goal of understanding under which scenarios the techniques prove useful. An effort will also be undertaken to learn the impact of the number of folds and splits in regards to the effectiveness of K-Fold cross validation and Monte Carlo cross validation respectively. Ultimately, by answering these questions, this paper aspires to identify best practices when evaluating classification models in machine learning.

Chapter 3: Design and Implementation

3.1 Framework

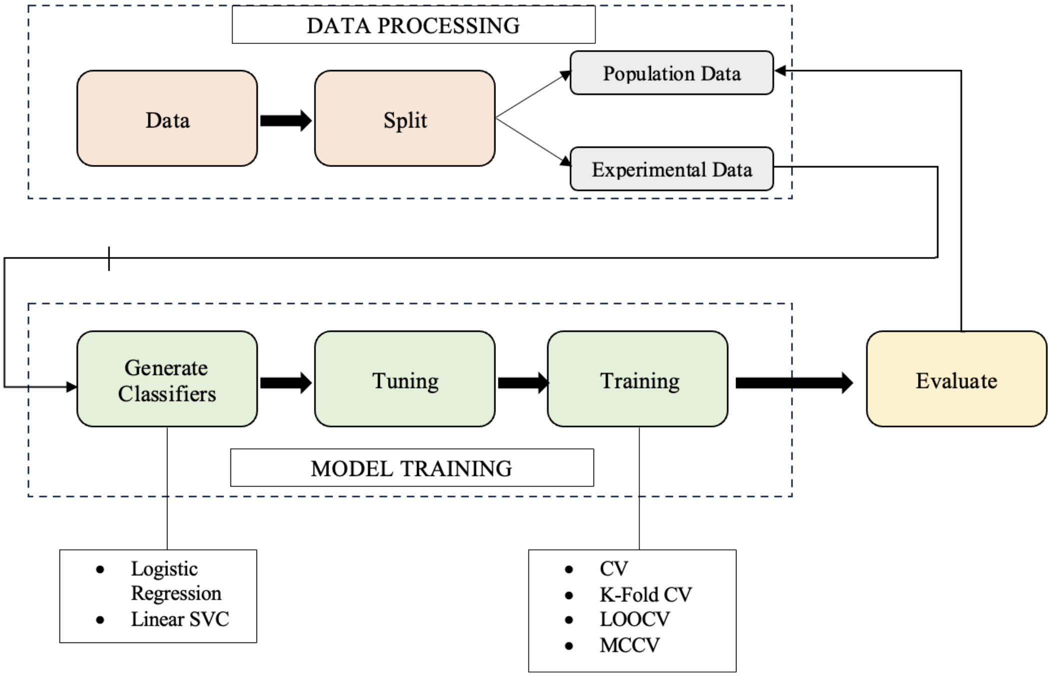


Figure 6: Model Pipeline

Figure 6 outlines the generic pipeline from data processing, to model training, and evaluation. Data processing entails creating a simulated classification dataset that is then subsampled for training and cross validation on the models. The models will then be evaluated on the population data in order to determine the effectiveness of the cross validation techniques. F1 score will be the primary evaluation metric and is discussed further in the following chapter.

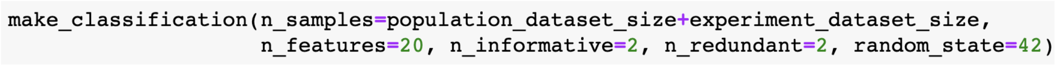
The following libraries were used to create the data and develop the models in Python 3.10.9:

- pandas 1.5.3 - For creating dataframes to hold data and results,
- numpy 1.23.5 - For simple vector math,
- scikit-learn 1.2.1 - For model creation, cross validation, hyperparameter tuning, and evaluation,
- matplotlib 3.7.0 - For data visualization
- seaborn 0.12.2 - For data visualization

3.2 Data Processing

3.2.1 Dataset Description

A simulated classification dataset of 11,000 rows was created using the scikit-learn library. The choice was made to use simulated data in order to run controlled experiments in which the different cross validation techniques can be evaluated on a large population dataset. Figure 7 below shows the dataset was created with 20 features (two informative features as well as two redundant features to add extra noise to the data). By including significant amounts of irrelevant data, the hope was to realistically simulate a real-world dataset (which can often be messy) and also determine if noise has an effect on the robustness and generalization ability of the models. Supervised learning also requires labeled data; consequently, there are two classes that each row of the data is categorized as (either 0 or 1).



```
make_classification(n_samples=population_dataset_size+experiment_dataset_size,
                    n_features=20, n_informative=2, n_redundant=2, random_state=42)
```

Figure 7: Code Used to Create Simulated Classification Dataset

3.2.2 Splitting Data

Of the 11,000 data points, 1,000 were allocated for experiments while the remaining 10,000 were reserved for the final evaluation of the developed models. Sub-samples of the experimental data at sizes of 30, 50, 100, and 200 were used for the training and validation of the models. The sub-samples are all of small sizes in order to determine how the different cross validation techniques perform specifically on tiny datasets, as well as to ascertain if dataset size has any affect. The population dataset is quite large, in comparison, in order to accurately evaluate the true performance of the trained models on unseen data.

3.3 Model Training

3.3.1 Classification Algorithms

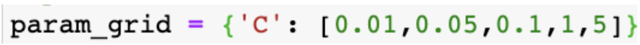
The following two classification algorithms were implemented during the course of the experiments:

1. Logistic Regression - One of the most common binary classification algorithms and a form of supervised learning that utilizes probabilistic values in order to label data.
2. Linear Support Vector Classification (SVC) - Another common machine learning algorithm that aims to create a hyperplane between data points in order to separate them into different classes.

The choice was made to use two different algorithms in order to leverage their unique strengths to accurately evaluate the cross validation techniques in cases where there is noise and outliers that can be especially consequential in small datasets.

3.3.2 Hyperparameter Tuning

Both classification models used L2 regularization in an effort to avoid overfitting. The hyperparameter ’C’, which is the inverse of regularization, was tuned using a parameter grid for all models. An exploratory analysis showed that a vast majority of the optimal ’C’ values in preliminary models laid between 0.01 and 5. As a result, the parameter grid for training throughout the course of experimentation was defined as follows:



```
param_grid = {'C': [0.01,0.05,0.1,1,5]}
```

Figure 8: Parameter Grid for Tuning Hyperparameter ’C’

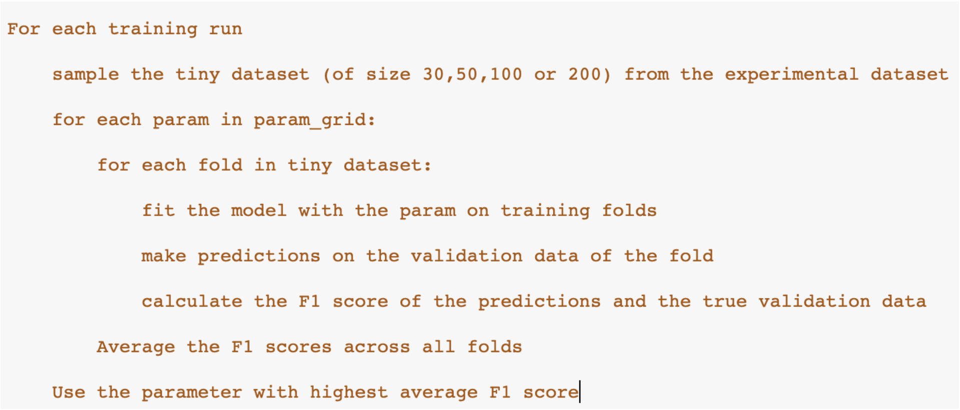
3.3.3 Application of Cross Validation Techniques

Traditional Cross Validation

The sub-sample data is divided into the conventional hold-out validation split (60% Training, 20% Validation, 20% Testing) and the models are trained, validated, and tested on these subsets accordingly. The traditional cross validation technique acts as a baseline on which we can compare the other evaluation methods against.

K-Fold Cross Validation

Over the course of experimentation, K-Fold cross validation was carried out using k = {2,5,10,20} folds. The average F1 score was then calculated of the model across each of the folds in a particular training loop as shown below in Figure 9.



```
For each training run
    sample the tiny dataset (of size 30,50,100 or 200) from the experimental dataset
    for each param in param_grid:
        for each fold in tiny dataset:
            fit the model with the param on training folds
            make predictions on the validation data of the fold
            calculate the F1 score of the predictions and the true validation data
        Average the F1 scores across all folds
    Use the parameter with highest average F1 score
```

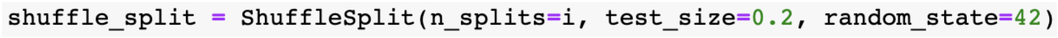
Figure 9: Pseudocode Representation of K-Fold CV Model Training

Leave-One-Out Cross Validation (LOOCV)

An extreme variant of K-Fold cross validation where the number of folds is equivalent to the number of rows in the dataset. Accordingly, the same methodology for K-Fold cross validation was reused from the above section with the number of folds parameter set to the dataset size.

Monte Carlo Cross Validation (MCCV)

Like K-Fold Cross Validation, MCCV requires splitting the data numerous times into training and testing sets. However, since the data splits are random in MCCV, the shufflesplit library from scikit-learn was used in implementation as shown below in Figure 10. The number of splits used during implementation were {2,5,10,20} and the average F1 score was calculated across the splits.



```
shuffle_split = ShuffleSplit(n_splits=i, test_size=0.2, random_state=42)
```

Figure 10: Code for Shufflesplit

Chapter 4: Evaluation

4.1 Overview

This chapter discusses the detailed evaluation of cross validation techniques in order to establish best practices when working with small datasets. Experiments were carried out with the aim of answering the following research questions in pursuit of this goal:

RQ1: Do alternative variations lead to better performing models than traditional cross validation?

RQ2: What impact does the number of folds (”k”) or splits have on the effectiveness of K-Fold cross validation and MCCV?

RQ3: Does a larger dataset lead to improved performance when using cross validation techniques?

RQ4: How closely related are model predictions on a sample compared to model predictions on a population?

4.2 Experimental Setup

For each classification algorithm type and each sample dataset size (30,50,100,200), 1000 models were each separately trained and tuned using different cross validation techniques. Hyperparameters of the model were tuned only using the data in the sample, and final predictions conducted on the population data were made without changing the parameters obtained. Random seeds were also set to facilitate reproducibility of experimental results.

4.3 Evaluation Metrics

4.3.1 F1 Score

One of the most widely used evaluation metrics in binary classification, F1 score is the harmonic mean of recall and precision. Because there is often a trade-off between recall and precision, F1 score can serve as a good indicator of the quality of a model since a balance between the two is required in order to achieve a high score. A higher F1 score often suggests that a model is more robust. For this reason, the metric is used as part of the optimization of hyperparameters during model training as well as evaluation throughout this project.

$$F_1\text{–score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$$

Figure 11: The Formula for F1 Score (TP = True Positive, FP = False Positive, and FN = False Negative)

4.3.2 Difference in F1 Estimate (Sample F1 Score - Population F1 Score)

Another important metric measured is the difference in the model F1 score from the training sample and the population dataset. If the F1 score of the sample is much larger than that of the population, it indicates that the model is overfitting on the training data and may not perform reliably when introduced to new data. On the other hand, a model that performs very similarly on the sample and the population is likely more trustworthy since its predictions are consistent. Therefore, in order to determine which is the 'best' cross validation technique, it is important to consider how accurately it estimates model performance.

4.4 Results and Analysis

4.4.1 Model Performances on Population Dataset

Table 1: Average F1 Scores for Compared Models

CV Technique	**Dataset Size**			
	30	50	100	200
CV	0.730	0.785	0.823	0.844
K-Fold - 2 folds	0.776	0.805	0.834	0.850
K-Fold - 5 folds	0.779	0.806	0.834	0.849
K-Fold - 10 folds	0.777	0.805	0.834	0.849
K-Fold - 20 folds	0.773	0.805	0.832	0.849
MCCV - 2 splits	0.770	0.803	0.832	0.849
MCCV - 5 splits	0.776	0.805	0.833	0.849
MCCV - 10 splits	0.777	0.809	0.834	0.850
MCCV - 20 splits	0.781	0.808	0.835	0.850
LOO	0.768	0.801	0.833	0.850

The average F1 scores across cross validation techniques for population datasets is displayed above in Table 1. The results between both classification algorithm types (Logistic Regression and Linear SVC) through 1000 model trials are averaged.

Across all dataset sizes, traditional cross validation produced the poorest results in terms of average population F1 score performance. This is especially true with the smallest dataset size (30) where average F1 score for traditional cross validation was more than 5% lower than the next worst performing technique (LOOCV). Monte Carlo cross validation with 20 splits performed the best, producing an average F1 score that was either the first or second highest in each dataset size. Following conventional wisdom, the average F1 score for each cross validation technique increased with dataset size, suggesting that more available data for training/validation improved model performance. It is also worth noting that as the dataset size increased, the disparities between average F1 scores across all cross validation techniques decreased. When the dataset size was 200, the difference between the lowest and highest average population F1 scores was just 0.006.

4.4.2 Cross Validation Techniques as Model Evaluators

Table 2: Average Difference in F1 Scores between Training Sample and Population Dataset

CV Technique	**Dataset Size**			
	30	50	100	200
CV	-0.028929	-0.010579	0.000214	0.005641
K-Fold - 2 folds	-0.025516	-0.013449	-0.005319	0.000439
K-Fold - 5 folds	-0.017102	0.000596	0.005123	0.008865
K-Fold - 10 folds	-0.087263	-0.024551	0.003781	0.007044
K-Fold - 20 folds	-0.264839	-0.139106	-0.027948	0.000233
MCCV - 2 splits	0.005825	0.018507	0.017067	0.013964
MCCV - 5 splits	-0.011588	0.002570	0.007336	0.009691
MCCV - 10 splits	-0.023083	-0.004323	0.001343	0.006068
MCCV - 20 splits	-0.028686	-0.009324	0.000911	0.004638
LOO	-0.356178	-0.373957	-0.402268	-0.416106

Table 2 displays the average difference in F1 scores between training sample predictions and population dataset predictions with results averaged between both classification algorithm types through 1000 model trials. Values closest to zero indicate that the estimation of model F1 performance through the cross validation technique were most accurate. Values below zero indicate that model F1 was underestimated (the sample F1 score of the model was lower than the population F1 score) while values above zero indicate overestimation.

As shown in the table above, across all dataset sizes, LOOCV was by far the worst evaluator of model performance. LOOCV consistently greatly underestimated model F1 performance by at least 0.35 on average, and this difference grew as the dataset size increased. 10-Fold cross validation and especially 20-Fold cross validation also considerably underestimated model performances for the smallest dataset sizes (30 and 50), but as the sample dataset size grew, the average difference in F1 score between the sample and population data dissipated, with 20-Fold cross validation even having the best estimation for dataset size 200. The effect of number of folds on model evaluation is analyzed more in depth in the next section of this chapter.

MCCV with a lower number of splits (2 and 5) along with traditional cross validation produced consistent estimations that were similar to the population F1 score on average. One trend worth noting is that smaller dataset sizes tended to produce underestimations of model performance while larger dataset sizes often produced the opposite. MCCV with 2 splits was the only the cross validation technique to overestimate model performance on average across all dataset sizes.

4.4.3 Effect of Number of Folds (”K”)

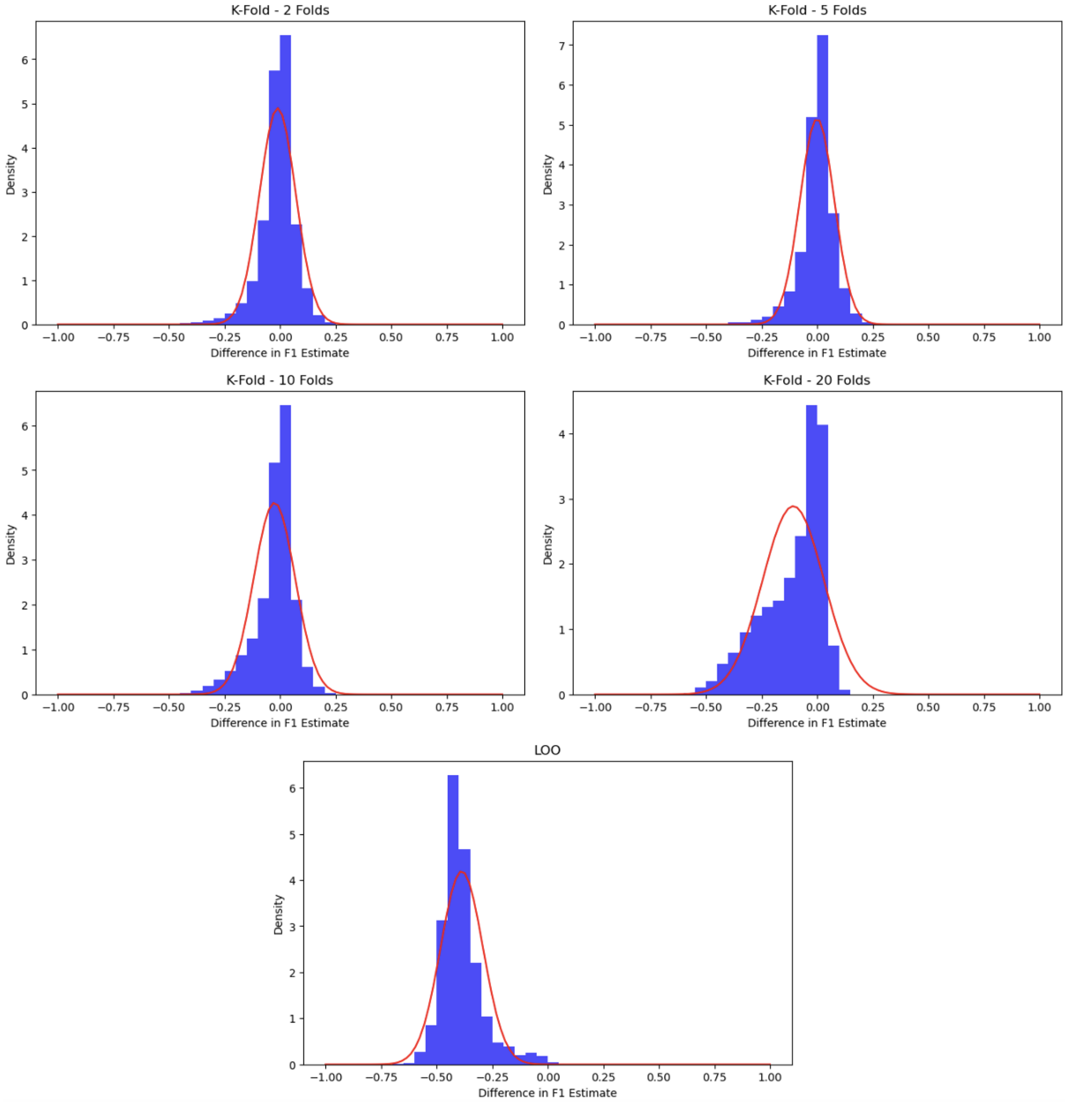


Figure 12: Histograms and Distribution Curves of the Difference in F1 Estimates for Number of Folds (”K”)

The figure above plots the distribution of the difference in F1 estimates across all dataset sizes and algorithm types for LOOCV and each variation of K-Fold cross validation. In an ideal distribution, a majority of the differences in F1 estimate would be centered around zero indicating that the cross validation technique often accurately estimates true model F1 performance. This is seen, more or less, in the plots for 2-Fold and 5-Fold cross validation where the highest concentration of differences in F1 estimates are right around zero and a majority of values are close to the center as illustrated by the narrow distribution curves.

However, as the number of folds increases, the distribution of the difference in F1 estimates starts to skew to the left of zero. Starting with 10-Fold cross validation, we see a higher and higher density of values falling below zero suggesting that there is an increase in the underestimation of model F1 performance as the number of folds increases. For LOOCV, we can see that nearly all differences in F1 estimates lie far below zero with the mean resting near -0.40.

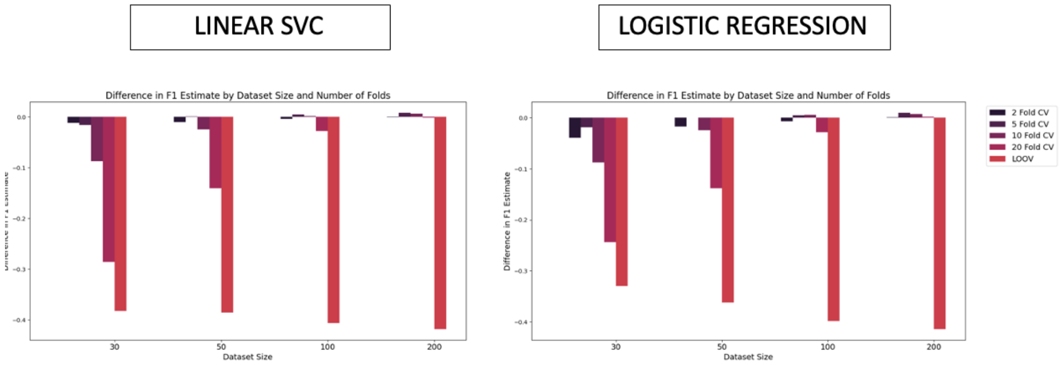


Figure 13: Average Difference in F1 Estimate by Dataset Size for K-Fold Cross Validation and LOOCV

The figure above displays the average difference in F1 estimate for Linear SVC and Logistic Regression over all dataset sizes. Across models using both classification algorithms, the average difference in F1 estimate was greatest for the smallest datasets for most forms of K-Fold cross validation. The K-Fold cross validation techniques also returned very similar results with both algorithms. With slightly larger datasets, models trained and evaluated with higher 'k' fold variations performed more accurate estimates of F1 scores on average and produced differences close to zero. Again, the outlier to these trends is LOOCV, which on average underestimated model performance more and more as the sample dataset size increased.

Ultimately, the data strongly suggests that LOOCV grossly underestimates a classification model's effectiveness when implemented with small datasets. While LOOCV performs reasonably well compared to other cross validation techniques in terms of tuning models that produce high F1 scores on population data, these experiments heavily indicate that it may not be an effective evaluator of model performance. The analysis also seems to suggest a larger number of folds can lead to underestimation of the F1 score when the number of folds is a higher proportion relative to the overall size of the dataset. This is evidenced by the large average difference in F1 estimate for 10-Fold and 20-Fold cross validation evaluated models of dataset size 30 and 50.

4.4.4 Effect of Number of Splits

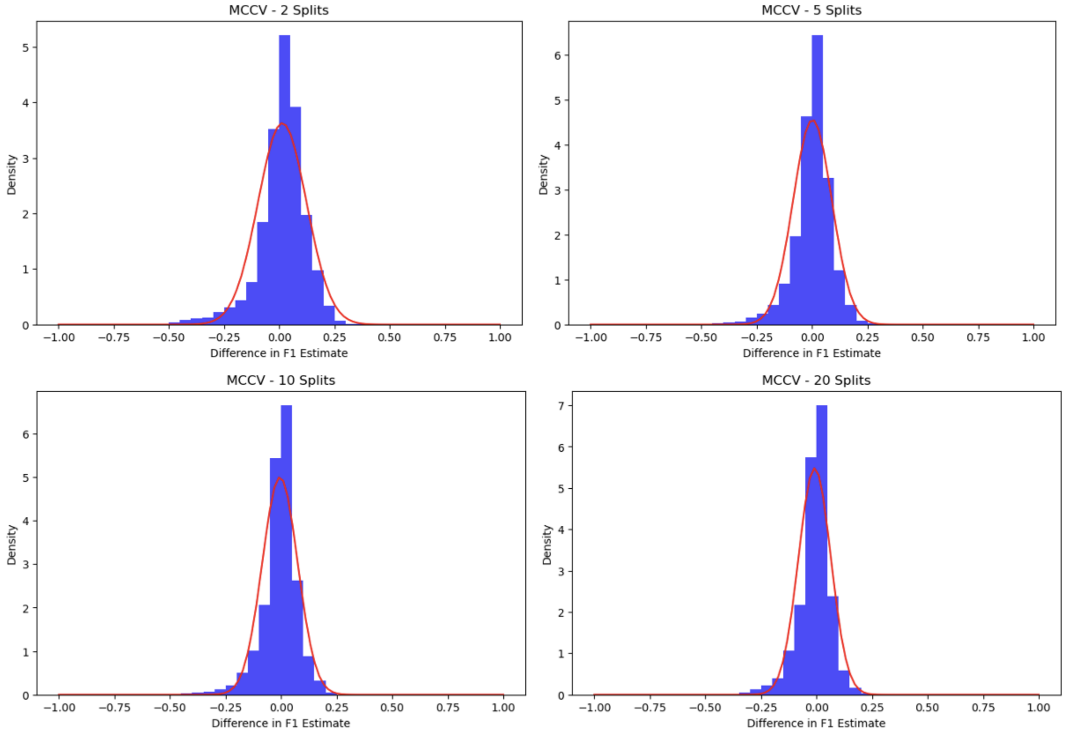


Figure 14: Histograms and Distribution Curves of the Difference in F1 Estimates for Number of Splits

The figure above plots the distribution of the difference in F1 estimates across all dataset sizes and algorithm types for all variations of MCCV. For all number of splits, the distributions follow a similar pattern where the highest concentration of difference in F1 estimates are near zero and the data follows a normal distribution. The distribution curves for MCCV with 2 splits and 5 splits are wider than those of the higher split variations, but the difference is slight. The plots indicate that regardless of the number of splits, MCCV performs reasonably well in regards to evaluating the true model F1 score.

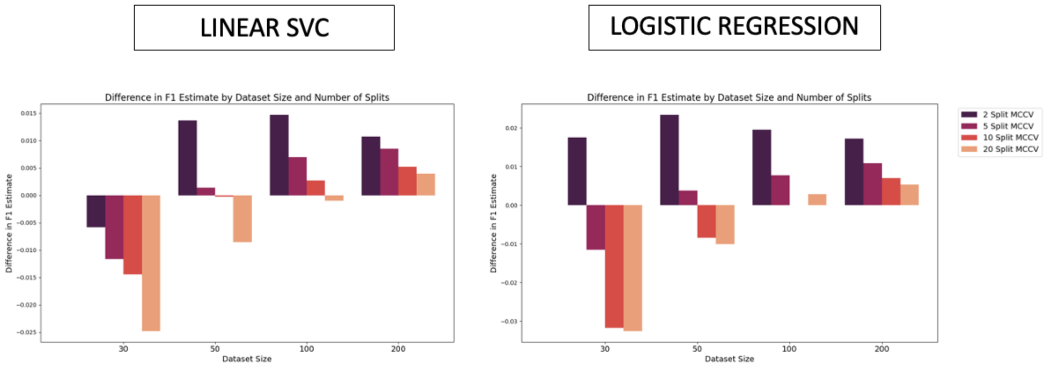


Figure 15: Average Difference in F1 Estimate by Dataset Size for MCCV

The figure above displays the average difference in F1 estimate for Linear SVC and Logistic Regression over all dataset sizes for MCCV. Generally speaking, MCCV appears to estimate F1 performance very similarly with both classification algorithms. With the smallest dataset size, MCCV with 2 splits and 5 splits appear to produce the closest F1 estimate on average with the difference increasing with the number of folds. However, with the larger datasets of size 100 and 200, the average difference in F1 estimate seems to decrease with a larger number of splits. Application of MCCV with the larger datasets also tends to result in slight overestimation while underestimation is more prevalent with the smaller dataset sizes.

Overall, there does seem to be some affect of the number of splits in MCCV in regards to the average difference in F1 estimate. For evaluating true model F1 performance, a lower number of splits appears to produce slighty more accurate results for smaller datasets (30 and 50) while a higher number of splits is more effective for larger dataset sizes (100 and 200). It is important to note, however, that this effect is not extremely pronounced with all variations of MCCV performing quite admirably with respect to training/tuning models with a high average F1 score (as seen in Table 1) as well as estimating model performance. In comparison to K-Fold cross validation, where an increased number of folds appeared to have a much larger impact on underestimating model performance, MCCV was much more consistent across a different number of splits and there was no skew in the average difference in F1 estimate across all dataset sizes.

4.4.5 Analysis of Computational Time

Table 3: Average Time (seconds) to Run 1000 Trials of Experiments

CV Technique	**Dataset Size**			
	30	50	100	200
CV - N/A folds	92.7	95.3	96.4	97.9
K-Fold - 2 folds	106.7	112.4	115.1	114.9
K-Fold - 5 folds	133.1	144.7	154.1	174.4
K-Fold - 10 folds	192.0	197.5	211.3	249.9
K-Fold - 20 folds	273.7	289.4	308.0	365.0
MCCV - 2 splits	103.2	113.8	118.5	120.2
MCCV - 5 splits	130.9	147.9	154.9	168.5
MCCV - 10 splits	179.9	199.9	214.1	239.8
MCCV - 20 splits	280.0	290.8	305.1	344.0
LOO	341.6	476.9	864.6	2122.4

The table above displays the average time it took to run 1000 trials of experiments with the different cross validation techniques in this project. Traditional cross validation was the quickest across all dataset sizes while computational time increased with more folds/splits. Unsurprisingly, LOOCV was the most time consuming cross validation technique due to the high number of folds and the computation required for each fold.

While computational expenses are generally not as notable an issue when working with small datasets, it is worth observing the increased time required for evaluating models with some of the variations of cross validation. This effect would be further exacerbated with a higher number of experimental trials or larger dataset sizes.

Chapter 5: Conclusion

5.1 Discussion

The goal of this project was to evaluate a variety of cross validation techniques in order to identify best practices when working with small datasets. In order to do so, numerous experiments were carried out on simulated data to understand what impact dataset size, type of binary classification algorithm, and tuning of hyperparameters such as number of folds or splits had on the effectiveness of these techniques. The cross validation methods were compared based on their abilities to train and tune models that produced high F1 scores on population data, as well as their performance in evaluating the efficacy of the models on sample datasets.

The results of these experiments suggest that implementations of K-Fold cross validation and MCCV can lead to improved model performance when compared to traditional cross validation. On average, F1 scores were higher for all versions of K-Fold cross validation and MCCV - considerably so for smaller dataset sizes. In terms of evaluating the effectiveness of models on small datasets, traditional cross validation and MCCV performed the most consistently across all dataset sizes.

An analysis on the effect of the number of folds in K-Fold cross validation suggested that 10-Fold and 20-Fold cross validation perform poorly as model evaluators on the smallest datasets (30 and 50) due to a sizable underestimation of model F1 score. This effect was not observed on larger datasets, perhaps since the number of folds was a smaller proportion of the total dataset size. Severe underestimation of the average difference in F1 estimate when using LOOCV seemed to support this argument that if the number of folds is a high proportion of the dataset size (100% in the case of LOOCV), the difference in F1 estimate is greater. The results indicate that due to this effect, LOOCV may not be a viable alternative to traditional cross validation.

5.2 Limitations and Future Work

Due to the timeline of this dissertation and computational limitations, there were some constraints in regards to the experimental setup of this project. During the hyperparameter tuning of the models, only one parameter ('C') was optimized, as tuning additional parameters like regularization would have significantly increased the trial run-time. Future studies would benefit from analyzing the effectiveness of these cross validation approaches on more complex models.

Only one simulated dataset was also used in the interest of ensuring results could be gathered in time. Consequently, it would be interesting to see if the results of this project would be reproducible on multiple different datasets with varying amounts of noise and bias. Particularly, using existing real-world classification datasets would be a direction that further research should explore.

Lastly, only two classification algorithms (Logistic Regression and Linear SVC) were explored in this research in an effort to limit the number of independent variables. Investigations involving additional classification algorithms such as Random Forests, Decision Trees, as well as other deep learning methods could provide further insights about these cross validation methods.

Appendix A: First appendix

The code for this project:
https://github.com/ashwink9812/MScDissertation

Bibliography

[1] David M. Allen. The relationship between variable selection and data agumentation and a method for prediction. *Technometrics*, 16(1):125–127, 1974.

[2] Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4(none):40 – 79, 2010.

[3] Daniel Berrar. *Cross-Validation*. 01 2018.

[4] Avrim Blum, Adam Tauman Kalai, and John Langford. Beating the hold-out: bounds for k-fold and progressive cross-validation. In *Annual Conference Computational Learning Theory*, 1999.

[5] PRABIR BURMAN. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76(3):503–514, 09 1989.

[6] Gavin C. Cawley and Nicola L.C. Talbot. Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers. *Pattern Recognition*, 36(11):2585–2592, 2003.

[7] Bradley Efron. Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78(382):316–331, 1983.

[8] Tadayoshi Fushiki. Estimation of prediction error by using k-fold cross-validation. *Statistics and Computing*, 21:137–146, 04 2011.

[9] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. 14, 03 2001.

[10] Bruce Marcot and Anca Hanea. What is an optimal value of k in k-fold cross-validation in discrete bayesian network analysis? *Computational Statistics*, 36, 09 2021.

[11] Isaac Nti, Owusu Nyarko-Boateng, and Justice Aning. Performance of machine learning algorithms with different k values in k-fold cross-validation. *International Journal of Information Technology and Computer Science*, 6:61–71, 12 2021.

[12] Rebecca Patro. Cross validation: K-fold vs. monte carlo, 2021. Accessed on: 29/08/2023.

[13] Phung and Rhee. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. *Applied Sciences*, 9:4500, 10 2019.

[14] Richard R. Picard and R. Dennis Cook. Cross-validation of regression models. *Journal of the American Statistical Association*, 79(387):575–583, 1984.

[15] R. Rao and Glenn Fung. On the dangers of cross-validation. an experimental evaluation. pages 588–596, 04 2008.

[16] Jun Shao. Linear model selection by cross-validation. *Journal of the American Statistical Association*, 88(422):486–494, 1993.

[17] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133, 1974.

[18] Lizhi Wang. The leave-worst-k-out criterion for cross validation. *Optimization Letters*, 17:1–16, 06 2022.

[19] Qing-Song Xu, Yi-Zeng Liang, and Yi-Ping Du. Monte carlo cross-validation for selecting a model and estimating the prediction error in multivariate calibration. *Journal of Chemometrics*, 18(2):112–120, 2004.

[20] Sanjay Yadav and Sanyam Shukla. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pages 78–83, 2016.