© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715172

Empirical Evaluation of Learning Curve Cross-Validation for Efficient Model Selection

DR. M. POMPAPATHI[1], N. SIVA PARVATHI[2], V. PAVANI[3], S. VARSHINI[4], V. MANIKANTA[5]
[1]*Associate Professor, Department of IT, R.V.R & J.C.C.E Guntur, India.*
[2, 3, 4, 5] *Final Year Students, Department of IT, R.V.R &J.C.C.E Guntur, India.*

Abstract- ***Model selection in machine learning is commonly performed using cross-validation, where candidate models are evaluated to estimate their generalization performance. Although reliable, this approach becomes computationally expensive when many models must be evaluated repeatedly on large datasets. Learning Curve Cross-Validation (LCCV) addresses this issue by evaluating models on progressively larger subsets of data and pruning weak candidates early. In this work, we implement the LCCV algorithm and evaluate it on several classification tasks. The implementation estimates performance at different training sizes using repeated cross-validation with confidence intervals. Models are pruned using optimistic learning curve extrapolation, while the Morgan–Mercer–Flodin (MMF) model is used to skip intermediate evaluation points. Experiments on real-world and synthetic datasets compare LCCV with traditional full cross-validation in terms of runtime, model selection agreement, and pruning behavior. Results show that LCCV can prune many candidate models and significantly reduce runtime on larger datasets, while introducing some overhead on smaller datasets.***

Keywords: Learning Curve Cross-Validation, Model Selection, Learning Curves, Early Pruning, Cross-Validation

I. INTRODUCTION

Model selection is an essential step in the machine learning workflow. Multiple candidate algorithms are typically compared to determine which performs best for a given dataset. Cross validation is widely used for this purpose because it provides reliable estimates of model performance on unseen data. However, evaluating many models with repeated cross validation can become computationally expensive, especially for large datasets.

Learning Curve Cross Validation (LCCV) [1] was proposed to address this problem. Instead of evaluating models on the full dataset, LCCV measures performance on progressively larger subsets of data, called anchors. At each anchor, cross validation provides a performance estimate and a confidence interval. Models that cannot outperform the current best candidate, even under optimistic assumptions, are pruned early, reducing unnecessary computation.

This idea relates to research on adaptive model evaluation. Jamieson and Talwalkar [2] studied non stochastic best arm identification, where weaker candidates are gradually eliminated to focus computation on promising ones. Similar ideas appear in work using learning curves to predict model performance. Domhan et al. [3] showed that learning curve extrapolation can accelerate hyperparameter optimization, while Van Rijn et al. [5] demonstrated that early learning curve observations can guide algorithm selection. Bergstra and Bengio [4] also showed that random search is an effective strategy for exploring hyperparameter spaces, emphasizing the need for efficient evaluation methods.

In this work, we implement the Learning Curve Cross Validation algorithm and examine its behavior across several classification tasks. The implementation evaluates models at multiple anchors using repeated cross validation with confidence intervals and prunes candidates based on optimistic learning curve extrapolation. Learning curves are fitted using the Morgan–Mercer–Flodin (MMF) model to allow skipping intermediate anchors when appropriate.

Experiments are conducted on several real and synthetic datasets using common classification algorithms. We compare LCCV with traditional full cross validation in terms of runtime, model selection agreement, and pruning behavior to better understand how learning curve-based model selection performs across datasets of different sizes.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715172

II. RELATED WORK

Efficient model selection has become increasingly important as the computational cost of training machine learning models continues to grow. Several methods have been proposed to reduce the cost of evaluating multiple candidate models or configurations.

Learning Curve Cross-Validation (LCCV) [1] speeds up model selection by using the structure of learning curves. Instead of evaluating every model on the full dataset, LCCV measures performance on progressively larger subsets of data. Cross-validation is performed at each stage to estimate performance and confidence intervals. Models that are unlikely to outperform the current best candidate are removed early, reducing unnecessary computation.

Related ideas appear in research on bandit-based resource allocation. Jamieson and Talwalkar [2] studied non-stochastic best-arm identification, where evaluation resources are focused on promising candidates while weaker ones are discarded. This principle of early elimination supports efficient model selection methods such as LCCV.

Learning curves have also been used to predict model performance. Domhan et al. [3] proposed techniques for extrapolating partial learning curves to estimate the final performance of neural networks during hyperparameter optimization. Similarly, Van Rijn et al. [5] showed that early learning curve observations can guide algorithm selection without requiring full evaluation of every candidate.

Work on hyperparameter search also emphasizes efficient evaluation strategies. Bergstra and Bengio [4] demonstrated that random search can be an effective approach for exploring large hyperparameter spaces, highlighting the need for methods that reduce evaluation cost.

Overall, these studies show that resource allocation, learning curve modeling, and efficient search strategies can significantly improve model evaluation. LCCV [1] combines these ideas by using learning curves and confidence estimates to guide early pruning during model selection.

III. PROPOSED METHODOLOGY

A. Overview of the LCCV Procedure

Learning Curve Cross-Validation (LCCV) reduces the computational cost of model selection by evaluating candidate models on progressively larger subsets of the training data instead of training every model on the full dataset.

Models are evaluated at predefined training sizes called anchors. At each anchor, repeated cross-validation estimates validation performance and a confidence interval. These estimates determine whether the model continues evaluation or is pruned.
Pruning is based on optimistic extrapolation of the learning curve. If the best possible projected performance of a model cannot exceed the current best competitor, the model is removed from further evaluation.

The implementation also fits a Morgan–Mercer–Flodin (MMF) learning curve model to observed points. This model is used only to skip intermediate anchors for promising models and is not used for pruning or final selection.

After all evaluations are completed, the final model is selected using the lower confidence bound of validation performance at the largest training size.

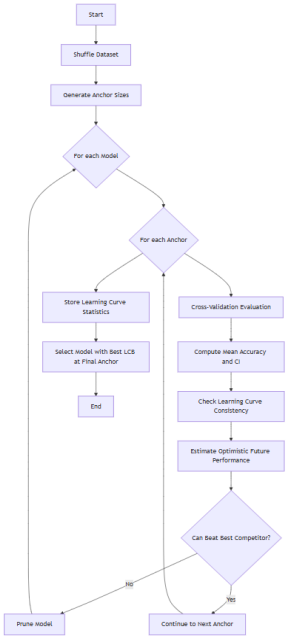

Figure 1 Flowchart of the LCCV Procedure

This flowchart summarizes the sequential evaluation, pruning, and selection process that forms the basis of the LCCV procedure.

B. Anchor-Based Learning Curve Construction

Model performance is evaluated at a sequence of dataset sizes called anchors.

The smallest anchor is defined as the larger of 32 samples or 2% of the dataset size. Each subsequent anchor doubles the training size until the full dataset is reached.

Training subsets follow a nested structure, where each anchor includes all samples from the previous anchor plus additional data. This ensures smooth learning curve growth.

Because cross-validation limits the available training samples in each fold, anchor sizes are constrained so they remain valid within the cross-validation scheme. The full dataset is evaluated separately as the final anchor.

C. Performance Estimation Using Repeated Cross-Validation

At each anchor, model performance is estimated using repeated cross-validation.

After each repetition, the algorithm updates the mean validation accuracy and computes a confidence interval using the Student's t-distribution. Repetitions continue until either:

the confidence interval becomes sufficiently small, or

- the maximum number of repetitions is reached.

A stricter confidence threshold is applied at the final anchor to obtain a more precise estimate. Training accuracy is also recorded during cross-validation but is not used for pruning.

D. Convexity Compatibility of Learning Curves

Learning curves typically follow a concave pattern, where performance gains decrease as training size increases.

The algorithm checks whether the observed learning curve is compatible with this pattern by examining slopes between anchors using confidence intervals. If a violation occurs, additional cross-validation repetitions are performed to reduce noise.

If instability remains after reaching the repetition limit, convexity-based pruning is disabled for that model. Convexity checks are also disabled for very small datasets to avoid unreliable estimates.

E. Optimistic Extrapolation and Model Pruning

Early pruning is the main mechanism that reduces computation.

After evaluating a model at an anchor, the algorithm estimates the most optimistic performance trajectory consistent with the observed learning curve. This projection represents the best performance the model could plausibly reach at the final dataset size.

The optimistic estimate is compared with the lower confidence bound of the best competing model. If the projected performance cannot exceed this bound by a defined margin, the model is pruned and no further anchors are evaluated.

Only models that complete evaluation without pruning are allowed to update the reference performance used for comparison.

F. Learning Curve Modeling for Anchor Skipping

To reduce computation for strong models, the algorithm fits a Morgan–Mercer–Flodin (MMF) function to the observed learning curve after at least four anchors.

The fitted model predicts performance at the full dataset size. If the predicted value is already competitive with the best observed model, intermediate anchors are skipped and the model is evaluated directly on the final dataset.

This step is used only for skipping anchors, not for pruning or final model selection.

G. Final Model Selection

After evaluating all candidate models, the final model is selected using the lower confidence bound of validation accuracy at the full dataset size.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715172

The lower bound is computed as the mean accuracy minus half the confidence interval width. The model with the highest lower bound is chosen.

If all models are pruned, a fallback mechanism returns the model with the best observed lower confidence bound.

H. Detailed Algorithm

The complete procedure of the proposed LCCV implementation is summarized in Algorithm 1.

```
Input:
  Dataset (X, y), Candidate models M, Cross-validation folds k
Output:
  Selected model m*
Shuffle dataset (X, y) using a fixed random seed
Generate anchor sizes S using geometric progression
Initialize table B storing best lower confidence bound per anchor
For each model m in M:
    Initialize storage for learning curve statistics
    For each anchor size s in S:
        Select nested subset of dataset with size s
        Perform repeated k-fold cross-validation on the subset
        Compute mean validation accuracy
        Compute confidence interval using t-distribution
        If confidence interval width ≤ tolerance:
            Stop repetitions
        Check convexity compatibility of the learning curve
        If convexity violation occurs:
            Increase repetitions or disable convexity check
        Compute optimistic extrapolation to the final dataset size
        Retrieve best competitor lower confidence bound from table B at anchor s (or closest
previous anchor)
        If optimistic performance < best competitor lower bound:
            Prune model m
            Stop evaluating larger anchors
        If number of observed anchors ≥ 4:
            Fit Morgan–Mercer–Flodin (MMF) learning curve model
            Predict performance at the final dataset size
            If predicted performance is competitive:
                Skip remaining intermediate anchors
                Continue evaluation at the final anchor
    Record anchor statistics for model m
    If model m was not pruned:
        Update table B with the best lower confidence bound observed at each anchor
Select model with the highest lower confidence bound at the final anchor
Return selected model m*
```

Figure 2 Algorithm 1

This algorithm optimizes model selection through learning-curve modeling and early pruning for fast, reliable results.

IV. EXPERIMENTAL SETUP

A. Datasets

The LCCV implementation was evaluated on a mix of real-world and synthetic classification datasets with different sizes. The datasets allow analysis of algorithm behavior on small, medium, and large learning problems.

Table 1 summarizes the main characteristics of all datasets used in the experiments.

Table 1

Dataset	Samples	Features	Classes
Iris	150	4	3
Wine	178	13	3
Ionosphere	351	34	2
Breast Cancer	569	30	2
Digits	1797	64	10
Synthetic12000	12000	30	2
Synthetic30000	30000	30	2

The first three datasets are small benchmarks used to observe learning curve behavior with limited training data. Breast Cancer and Digits represent medium-sized problems. Two synthetic datasets with 12,000 and 30,000 samples were generated to study the computational behavior of LCCV on larger datasets.

B. Candidate Models

The experiments include a diverse set of classification algorithms representing different learning approaches.

Table 2 Candidate Models

Category	Algorithms
Ensemble Methods	Random Forest, Gradient Boosting, Extra Trees
Tree-Based	Decision Tree
Instance-Based	k-Nearest Neighbors
Probabilistic	Naive Bayes
Linear Models	Logistic Regression, Stochastic Gradient Descent
Kernel Methods	Support Vector Machines

For large synthetic datasets, Support Vector Machines and Gradient Boosting were excluded due to their higher computational cost. The remaining models were evaluated to maintain manageable training times.

C. Evaluation Protocol

All datasets were randomly shuffled using a fixed random seed before evaluation to ensure consistent data ordering across models.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715172

Model evaluation follows the LCCV procedure using progressively increasing anchor sizes. Anchors start from a small subset of the dataset and roughly double until the full dataset size is reached.

At each anchor:

- Performance is estimated using k-fold cross-validation
- Most datasets use 5-fold cross-validation
- Smaller datasets may use fewer folds to maintain sufficient training samples

Repeated cross-validation is performed when necessary to obtain stable performance estimates. Mean validation accuracy and a confidence interval are computed using the Student's t-distribution.

Repetitions continue until:

- the confidence interval width falls below a tolerance threshold, or
- a maximum number of repetitions is reached.

A stricter threshold is applied at the final anchor to ensure accurate model comparison on the full dataset.

D. Baseline Comparison

To evaluate computational efficiency, LCCV was compared with a baseline model selection method using full cross-validation.

In the baseline approach, each candidate model is evaluated using cross-validation on the entire dataset without progressive evaluation or early pruning.

The comparison focuses on three metrics.

Table 3 Evaluation Metrics

Metric	Description
Runtime	Total time required to evaluate all models
Speedup	Ratio of baseline runtime to LCCV runtime
Model Selection Agreement	Whether LCCV selects the same model as full cross-validation

E. Repeated Experiments

Some datasets were evaluated using multiple random seeds to assess robustness. Each seed generates different dataset permutations and cross-validation splits.

Results from repeated runs were aggregated when reporting runtime and model selection outcomes. This reduces the influence of any single random data ordering.

V. RESULTS

This section evaluates the empirical performance of Learning Curve Cross-Validation (LCCV) across datasets of varying sizes. The experiments compare LCCV with standard full cross-validation in terms of model selection agreement, runtime efficiency, and pruning behavior. Results are summarized using aggregated tables and figures to highlight patterns across dataset scales

A. Dataset-Level Results

Table 4 summarizes the final experimental results for each dataset after aggregating repeated experiments across seeds. The table reports model selection agreement between LCCV and full cross-validation, the average runtime ratio between LCCV and full cross-validation, and the pruning rate.

Table 4

Dataset	Agreement	Avg Runtime Ratio	Median Runtime Ratio	Pruning Rate	Full CV Winner	LCCV Winner
Iris	1.0	0.156	0.116	0.778	RandomForest	RandomForest

Wine	1.0	0.155	0.107	0.667	ExtraTrees	ExtraTrees
Ionosphere	1.0	0.184	0.121	0.667	Extratrees	ExtraTrees
Breast Cancer	0.6	0.176	0.132	0.311	LogisticRegression	LogisticRegression
Digits	0.4	0.464	0.250	0.578	KNN	KNN
Synthetic12000	1.0	1.932	0.491	0.571	KNN	KNN
Synthetic30000	1.0	2.145	0.846	0.571	KNN	KNN

Across all datasets the overall model selection agreement between LCCV and full cross-validation was 0.67. Perfect agreement was observed for the small datasets and the large synthetic datasets, while disagreements occurred primarily for medium-sized datasets.

The pruning rates indicate that LCCV eliminated a substantial fraction of candidate models early in the evaluation process. The Iris dataset exhibits the highest pruning rate among the evaluated datasets (Table 4), indicating that weak models were quickly identified during early anchors.

B. Seed-Level Variability

To evaluate stability under different data splits, experiments were repeated using multiple random seeds for the Breast Cancer and Digits datasets. Table 5 reports the aggregated statistics.

Table 5

Dataset	Agreement Mean	Runtime Ratio Mean	Runtime Ratio Std	Pruning Rate Mean
Breast Cancer	0.60	0.176	0.043	0.311
Digits	0.40	0.464	0.121	0.578

For the Breast Cancer dataset, the most frequent models selected by full cross-validation were Logistic Regression and Support Vector Machines. These algorithms produced similar validation performance, which occasionally resulted in different selections across seeds.

For the Digits dataset, K-Nearest Neighbors was the dominant model, although ExtraTrees and SVM occasionally achieved comparable accuracy. The higher pruning rate for this dataset reflects the ability of learning curves to identify weaker models earlier as training size increases.

C. Dataset Size Comparison

To examine how LCCV behaves across different data scales, datasets were grouped into small, medium, and large categories. Table 6 summarizes the aggregated results.

Table 6

Dataset Size Group	Agreement Avg	Runtime Ratio	Pruning Rate
Small (Iris, Wine, Ionosphere)	1.0	0.165	0.704
Medium (Breast Cancer, Digits)	0.5	0.320	0.444
Large (Synthetic12000, Synthetic30000)	1.0	2.173	0.571

The results reveal a clear relationship between dataset size and the computational behavior of LCCV. For small datasets, the additional overhead associated with repeated cross-validation and confidence interval estimation results in runtime ratios below one, indicating that LCCV runs faster relative to full cross-validation.

For medium-sized datasets, pruning remains active but model performance differences become less pronounced, leading to occasional selection disagreements.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715172

For large datasets, LCCV maintains perfect agreement with full cross-validation while pruning more than half of the candidate models. The increased training cost at this scale makes early pruning particularly beneficial.

D. Runtime Scaling Analysis

Figure 3 illustrates the relationship between dataset size and runtime/speedup behavior.

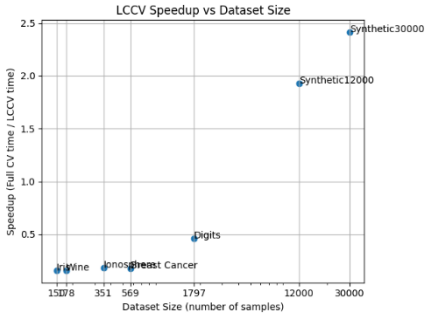

Figure 3 Runtime/speedup Ratio vs Dataset Size

The figure shows that the relative runtime of LCCV changes as dataset size increases. While the overhead of the algorithm dominates for smaller datasets, larger datasets benefit from early pruning because expensive model training can be avoided for clearly inferior candidates.

E. Pruning Behavior

Figure 4 illustrates the pruning rate observed across datasets.

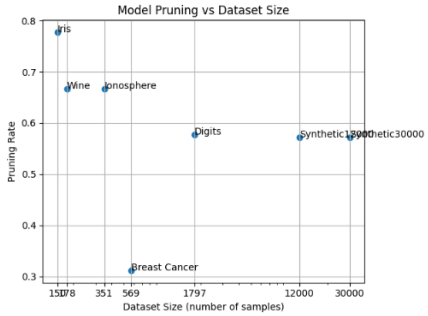

Figure 4 Pruning Rate vs Dataset Size

Pruning remains active across all dataset sizes, indicating that learning curves provide useful early signals about model performance. Small datasets show the highest pruning rates because performance differences between models become visible quickly during early training stages.

Medium datasets exhibit slightly lower pruning rates because multiple models achieve similar validation accuracy during early anchors.

F. Learning Curve Behavior

To illustrate the internal dynamics of LCCV, Figure 5 shows the learning curve of digits dataset winner during evaluation.

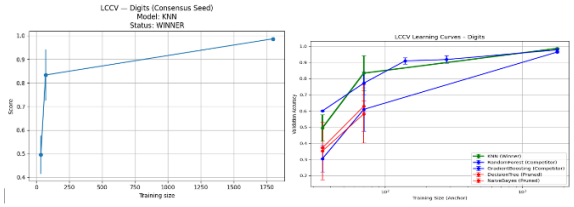

Figure 5 Example Learning Curve

The curves demonstrate how model performance evolves as the training set grows. Models with consistently lower validation accuracy are identified early and pruned before reaching the full dataset size. This behavior allows LCCV to reduce unnecessary training while preserving the final model selection.

G. Key Findings

The experimental evaluation highlights three primary observations.

First, LCCV consistently prunes a large fraction of candidate models during early anchor evaluations, reducing unnecessary computation.

Second, the computational behavior of LCCV varies with dataset size. While overhead dominates for small datasets, pruning becomes increasingly beneficial as dataset size grows.

Third, model selection agreement remains high across most datasets, with disagreements occurring primarily when multiple models achieve nearly identical validation performance.

Together, these results demonstrate that learning curve–based pruning provides a practical strategy for accelerating model selection while maintaining reliable predictive performance.

VI. DISCUSSIONS

The experimental results demonstrate that Learning Curve Cross-Validation (LCCV) can effectively reduce unnecessary model training through early pruning while maintaining reliable model selection. Across the evaluated datasets, LCCV consistently eliminated a substantial portion of candidate models during early training anchors, allowing weaker models to be discarded before full training. The results also show that the computational behavior of LCCV depends on dataset size. While the additional overhead of repeated cross-validation introduces limited benefits for small datasets, pruning becomes increasingly advantageous as dataset size grows and model training becomes more expensive. Occasional disagreements between LCCV and full cross-validation were observed mainly for medium-sized datasets where competing models achieved very similar validation performance. Overall, these findings suggest that learning curve–based pruning provides a practical strategy for accelerating model selection without significantly affecting predictive performance.

VII. CONCLUSION

This work presented an implementation and empirical evaluation of Learning Curve Cross-Validation (LCCV) for efficient model selection. By estimating learning curves and applying early pruning strategies, LCCV reduces the computational cost of evaluating multiple candidate models. Experimental results across datasets of varying sizes show that LCCV maintains high agreement with standard cross-validation while pruning many models early in the evaluation process. The benefits of this approach become particularly evident for large datasets, where significant reductions in training time can be achieved. These results demonstrate that learning curve–based model evaluation is a promising direction for scalable and efficient machine learning model selection.

REFERENCES

[1] A. Klein, S. Falkner, J. T. Springenberg, and F. Hutter, “Fast and Informative Model Selection using Learning Curve Cross-Validation,” Proceedings of the International Conference on Learning Representations (ICLR), 2017.

[2] K. Jamieson and A. Talwalkar, “Non-stochastic Best Arm Identification and Hyperparameter Optimization,” Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), 2016.

[3] T. Domhan, J. T. Springenberg, and F. Hutter, “Speeding Up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves,” Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2015.

[4] J. Bergstra and Y. Bengio, “Random Search for Hyper-Parameter Optimization,” Journal of Machine Learning Research, vol. 13, pp. 281–305, 2012.

[5] J. N. van Rijn, S. Bischl, and J. Vanschoren, “Fast Algorithm Selection Using Learning Curves,” Proceedings of the International Symposium on Intelligent Data Analysis, 2015.

[6] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian Optimization of Machine Learning Algorithms,” Advances in Neural Information Processing Systems (NeurIPS), 2012.

[7] F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Sequential Model-Based Optimization for General Algorithm Configuration,” Proceedings of the International Conference on Learning and Intelligent Optimization (LION), 2011.

[8] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization,” Journal of Machine Learning Research, vol. 18, no. 185, pp. 1–52, 2017.

[9] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms,” Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2013.

[10] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, “Auto-sklearn: Efficient and Robust Automated Machine Learning,” Advances in Neural

Information Processing Systems (NeurIPS), 2015.

[11] A. Klein, S. Falkner, N. Mansur, and F. Hutter, "Learning Curve Prediction with Bayesian Neural Networks," International Conference on Learning Representations (ICLR), 2017.

[12] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995.

[13] C. M. Bishop, Pattern Recognition and Machine Learning. New York: Springer, 2006.