Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

ModelFlow: A Machine Learning and MLOps Workflow

Hardik Kammal Sheth
M.Tech Data Science and Analytics
CSE, MIT World Peace University
Pune, India

Rushikesh Dilip Warade
M.Tech Data Science and Analytics
CSE, MIT World Peace University
Pune, India

Tanmay Atul Yerne
M.Tech Data Science and Analytics
CSE, MIT World Peace University
Pune, India

Yashraj Nitin Patil
M.Tech Data Science and Analytics
CSE, MIT World Peace University
Pune, India

Dr. Jayshree Ghorpade-Aher
DCET, SoCSE
MIT WPU
Pune, India

Abstract: ModelFlow is a safe workflow for binary classification used on tabular datasets and comes with data validation features, processing, model comparison, tuning and ensemble selection. The reference experiment performed on the IBM Telco Customer Churn dataset uses an 80/20 split for stratified data and 5-fold cross-validation for training. Six classifiers are compared using the same preprocessing and ranking process and the best candidates are tuned using optional feature selection before evaluating Soft Voting and Stacking. Among all models used, XGBoost had the highest mean cross-validation ROC-AUC score of 0.8469. Soft Voting was chosen as the final model as it was found to have an average ROC-AUC score of 0.8504 and a single hold-out ROC-AUC score of 0.8470. The implementation is flexible in making it possible to process any compatible CSV data provided by users. In addition to this, the paper also mentioned the future deployment framework using a web interface with FastAPI, Docker and Kubernetes.

Keywords - customer churn, machine learning workflow, model selection, ensemble learning, MLOps.

I. INTRODUCTION

Customer churn prediction helps to inform retention strategies by detecting customers who might discontinue using a service. However, achieving reliable prediction outcomes entails more than the use of one algorithm; it is necessary to validate data from the application, learn transformations with the test set unexposed, and compare model choices in accordance with the same protocol. ModelFlow sets all of these conditions clear in a single repeatable workflow.

The contribution is threefold: a generic validation and preprocessing layer for compatible binary-classification CSV files; a CV-only comparison of individual and ensemble models; and a serialized final pipeline that can be connected later to the deployed inference service. IBM Telco Customer Churn is only used as a reference experiment and does not constrain the platform.

II. RELATED WORK

Several methods like logistic regression, K-nearest neighbours, decision trees, random forests, support-vector machines and gradient boosting provide different approaches to binary prediction [1]-[6]. Combining more than one model is possible with methods like voting and stacking [7], [8]. There are also methods like cross-validation and random search that help in picking up configurations without only one validation split [9], [10]. When churn is the minority class, ROC-AUC and precision-recall metrics can supplement threshold-based metrics [12], [13].

Applied churn experiments run into the risk of optimistic performance reporting due to transformations, feature selection or parameter search that have observed data from final test. To mitigate this risk, ModelFlow clarifies preprocessing to be part of every fitted pipeline and to freeze the test partition before any candidate is ranked.

III. DATASET AND METHODOLOGY

The IBM Telco database consists of 7,043 records in conjunction with 21 columns of information. The target variable in the IBM dataset is churn. The dataset includes a total of 1,869 cases where there is churn, or churn is labelled Yes and 5,174 cases where there was no churn or churn is labelled No. Customer identifiable features are removed from the dataset. The split of the dataset was later done on a fixed stratified ratio of 80/20 with random seed 42 prior to the model training. For numerical features, median imputation, and scaling are applied , while most-frequent imputation and one-hot encoding are performed for categorical features. All transformations learnt are made available through the scikit-learn compliant pipelines [11].

Table I. Reference data profile

Property	Value
Records	7,043
Target	Churn: Yes / No
Positive class	1,869 (26.54%)
Input data	mixed numeric/categorical
Test policy	one final evaluation

Table II. Pipeline operations

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

Stage	Operation
Validation	Target and class checks
Numeric data	Median imputation; scaling
Categorical data	Mode imputation; one-hot encoding
Role inference	Numeric, categorical, ID, excluded
Artifacts	Metrics, plots and final pipeline

In Phase four, Logistic Regression, K-Nearest Neighbors, Decision Tree, Random Forest, Support Vector Machine and XGBoost are compared. The Dummy Classifier acts as the control model, which does not learn. The models are ranked based on average ROC AUC, average precision, average F1 score, average recall, variation in ROC AUC and overall training time. In Phase five, the best individual models are selected, the twelve randomized configurations of each model are created and compared, L1 feature-selection step is compared with pass-through preprocessing and then Soft Voting and Stacking models are developed.



Fig. 1. ModelFlow workflow from user data to a reusable final pipeline.

IV. VALUATION PROTOCOL

Five-fold stratified cross-validation on the training dataset identifies each rank. The 20% reserved test dataset is not involved in any preprocessing, feature selection, tuning or ranking steps. Once the winner from the final cross-validation stage is concluded, the model is trained using the entire training data and evaluated on the test dataset. The main evaluation metric used is ROC-AUC while other evaluation metrics like average precision, accuracy, precision, recall and F1 score serve as auxiliary statistics.
ROC-AUC indicates if the model assigns a better rank to instances of churn compared to those of no churn at all possible classification thresholds. Average precision focuses on the positive class, which is relevant in the case of imbalanced class distribution. The metric of accuracy alone may mask poor results for the positive class, therefore precision, recall and F1 score are calculated as well.

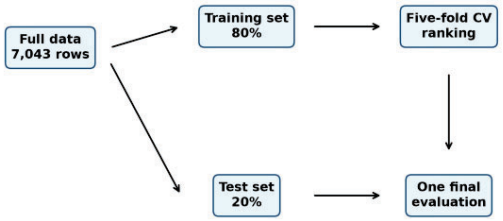

Fig. 2. Training-only cross-validation and one-time protected test evaluation.

Table III. Individual-model CV comparison

Model	ROC-AUC	Rank
Dummy	.5000	7
Logistic Regression	.8461	2
KNN	.7672	5
Decision Tree	.6513	6
Random Forest	.8197	3
SVM	.7989	4
XGBoost	.8469	1

V. RESULTS AND DISCUSSI

Among all the candidates, XGBoost is the one that performed best, with an average five-fold ROC-AUC value of 0.8469 and an average precision of 0.6660. The three shortlisted candidates during Phase 5 were the same candidates from Phase 4, which included XGBoost, Logistic Regression and Random Forest. There were a total of 23 transformed features retained by the selected XGBoost and Random Forest configurations with the help of the L1 selector while the Logistic Regression configuration was chosen on the basis of passthrough option.

Table IV. Tuned and ensemble CV comparison

Candidate	ROC-AUC	Rank
Tuned Logistic Regression	.8463	5
Tuned Random Forest	.8466	4

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

Tuned XGBoost	.8492	3
Soft Voting	.8504	1
Stacking	.8501	2

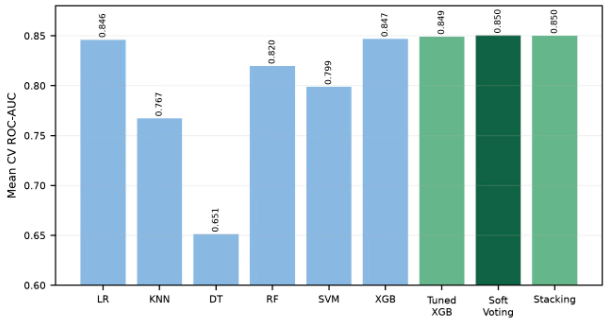

Fig. 3. Mean five-fold CV ROC-AUC for individual, tuned and ensemble candidates.

The improvement achieved when using Soft Voting instead of XGBoost is not significant due to the employment of the same training data. However, it provides useful information because it is acquired without looking at the final test sample. The same ROC-AUC was achieved when stacking was used, but the required fitting time is longer as it has to build a meta-model. Therefore, Soft Voting was chosen according to the specified rules of tie-breaking.

The selection of Soft Voting relied on predetermined CV rules only. The performance showed the protected hold-out ROC-AUC of 0.8470 with precision 0.6555, accuracy 0.8034, precision 0.6611, recall 0.5321 and F1-score 0.5896. This small difference in cross-validation and test ROC-AUC underlines the reliability of the result, but it is not true for every churn dataset.

Table V. Final Soft Voting hold-out result

Metric	Value
ROC-AUC	.8470
Average precision	.6555
Accuracy	.8034
Precision	.6611
Recall	.5321
F1-score	.5896

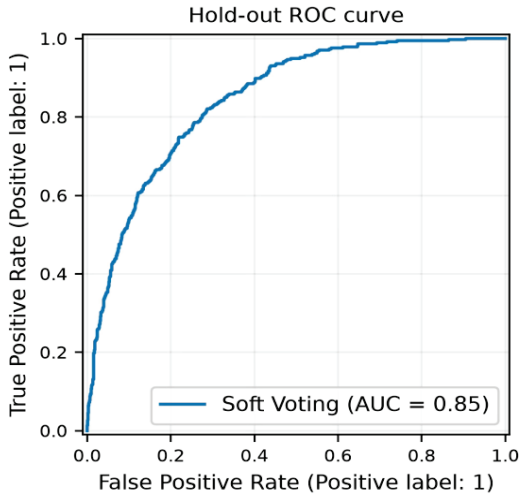

Fig. 4. ROC curve for the selected Soft Voting ensemble on the protected hold-out partition.

VI. DE YMENT EXTENSION

The work carried out allows the foundation of ML-development to be created together with storing the necessary pre-processing data acquired in a single serialization pipeline. At this stage, the project will open its workflow via a web interface and a FastAPI prediction service. The Docker system will provide deployment applications as Kubernetes and a Python operator will control the deployment of deployment packages.

This web UI will make it possible for users to upload supported CSV files, specify the target field, check data quality results and view the relevant leaderboard. A model registry will keep the chosen pipeline, schema of features and metrics. Moreover, the inference service will receive new inputs, compare their fields with the saved schema, and return the name with the probability of belonging to the class. The purpose of separating containerization from the statistical selection phase is for the deployed system to apply the same pipeline that passed the trial.

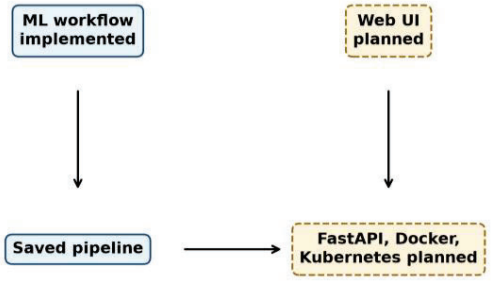

Fig. 5. Implemented ML core and planned deployment extension.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

VII. LIMITATIONS AND FUTURE WORK

The outcomes revealed are derived from one publicly available data set and one fixed hold-out split. Future endeavors will focus on evaluating additional data sets, implementing nested cross-validation, determining classification thresholds through training data-only out-of-fold predictions and analyzing serving delay and expansion performance after deploying the deployment layer.

VIII. CONCLUSION

The process of tabular binary classification has been thoroughly defined by ModelFlow. It checks the data, partitions the data into test data, performs the transformation process within the foldings, evaluates the models chosen for scoring, selects the final models, and checks the prediction results. Soft Voting obtained the highest CV ROC-AUC equal to 0.8504 in the reference experiment.

DECLARATIONS

Funding: No external funding was received. Conflict of interest: The authors declare no conflict of interest. Data availability: The IBM Telco Customer Churn sample data are available to the public [15]. Author contributions: All authors contributed to the conception of the paper, experimental review, and preparation of the manuscript and approved the submitted version. AI use: The Generative AI tool was utilized to refine the language and create the conceptual artwork; all authors checked the content and take responsibility for it.

REFERENCES

[1] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, Applied Logistic Regression, 3rd ed. Wiley, 2013.

[2] T. M. Cover and P. E. Hart, “Nearest neighbor pattern classification,” IEEE Trans. Inf. Theory, vol. 13, no. 1, pp. 21–27, 1967.

[3] L. Breiman et al., Classification and Regression Trees. Wadsworth, 1984.

[4] L. Breiman, “Random forests,” Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[5] C. Cortes and V. Vapnik, “Support-vector networks,” Mach. Learn., vol. 20, pp. 273–297, 1995.

[6] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proc. KDD, 2016, pp. 785–794.

[7] T. G. Dietterich, “Ensemble methods in machine learning,” in Multiple Classifier Systems, 2000, pp. 1–15.

[8] D. H. Wolpert, “Stacked generalization,” Neural Netw., vol. 5, no. 2, pp. 241–259, 1992.

[9] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in Proc. IJCAI, 1995, pp. 1137–1143.

[10] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” J. Mach. Learn. Res., vol. 13, pp. 281–305, 2012.

[11] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[12] T. Fawcett, “An introduction to ROC analysis,” Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.

[13] T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets,” PLOS ONE, vol. 10, no. 3, 2015.

[14] D. Kreuzberger, N. Kühl, and S. Hirschl, “Machine learning operations (MLOps): Overview, definition, and architecture,” IEEE Access, vol. 11, pp. 31866–31879, 2023.

[15] IBM, “Telco customer churn sample data,” IBM Community. [Online]. Available: https://community.ibm.com/community/user/blogs/steven-macko/2019/07/11/telco-customer-churn-1113.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)