Evaluating Machine Learning Models on SMS Spam Detection and Fashion-MNIST Image Classification

Name: Komala B Srinivas & Ananya Purohith

CSC 272: Machine Learning

May 14, 2025

Abstract

This project evaluates and compares several machine learning models across two major classification tasks: binary classification for detecting spam messages and multiclass classification for recognizing fashion items in images. We used the SMS Spam Collection dataset and the Fashion-MNIST dataset to conduct our experiments. The primary goal was to analyze which models performed best under different data types by carefully preprocessing the data, tuning hyperparameters, and evaluating using metrics like accuracy, F1 score, and learning curves. We found that SVM performed exceptionally well for text classification, while MLP was the most effective for image classification. Our analysis also explores the importance of tuning, class-wise errors, and the practical trade-offs between simplicity and accuracy.

Introduction

Machine learning models are now integrated into everyday technologies, from email filtering and fraud detection to personalized product recommendations and facial recognition. Two common tasks these models solve are binary classification and multiclass classification. In our project, we focus on both. For binary classification, we used the SMS Spam Collection dataset to identify whether a message is spam or not, which is an essential use case in communication platforms, cybersecurity systems, and mobile messaging applications. For multiclass classification, we used the Fashion-MNIST dataset, which involves classifying grayscale images of clothing into ten distinct categories. This type of task is directly applicable to e-commerce tagging, recommendation systems, and warehouse automation.

Our aim is to compare how different machine learning algorithms behave across these two datasets. Instead of relying on just a single accuracy number, we looked at learning curves, training time, confusion matrices, and F1 scores to truly understand how well these models generalize and what challenges they face. This report walks through our dataset preparation steps, the models we selected, how we tuned them, what the results were, and what lessons we learned.

Dataset Overview

The SMS Spam dataset contains 5,574 labeled messages in English, categorized as either "spam" or "ham." We noticed a class imbalance, with more ham messages than spam. Before feeding the data to models, we cleaned it by converting text to lowercase, removing punctuation and stopwords, and applying TF-IDF vectorization to convert text into numerical features. This approach allowed us to keep the sparsity of the text structure while weighing the importance of different words.

Fashion-MNIST contains 70,000 28x28 grayscale images across ten categories such as shirts, trousers, bags, and sneakers. Each image was flattened into a 784-dimensional vector. We normalized the data to bring all pixel values between 0 and 1. This helped speed up convergence during training and made it easier for models like Logistic Regression and MLP to learn effectively. We split both datasets into training and test sets in an 80:20 ratio while keeping class distributions balanced.

Methods- Model Selection and Rationale

For the SMS Spam dataset, we chose Logistic Regression, Naïve Bayes, K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). Logistic Regression is widely used in text classification due to its simplicity and ability to work well with sparse data. Naïve Bayes assumes independence between words and is extremely fast, making it ideal for tasks like spam detection. KNN was included to test how a non-parametric, instance-based learner handles high-dimensional data. SVM is known for performing well in high-dimensional settings, especially with text features.

For the Fashion-MNIST dataset, we implemented Logistic Regression, KNN, Decision Tree, and Multilayer Perceptron (MLP). These were chosen to provide a mix of linear models, tree-based models, and a neural network. Logistic Regression served as a baseline. KNN was expected to perform decently due to the low-level pixel similarity. Decision Tree was included for its interpretability, although we anticipated overfitting. MLP was the most advanced model, capable of learning complex nonlinear patterns, which are often needed in image data. The following tables summarize the models used for each dataset, including the reasoning behind their selection, the hyperparameters that were tuned during cross-validation, and the evaluation metrics used to compare performance.

Model	Rationale	Hyperparameters Tuned	Evaluation Metrics
Logistic Regression	Effective for sparse TF-IDF vectors and linearly separable data	C, solver, max_iter	Accuracy, F1 Score
Naive Bayes	Probabilistic, fast, performs well on word count distributions	alpha, fit_prior	Accuracy, F1 Score
K-Nearest Neighbors	Instance-based learning, serves as a non-parametric baseline	n_neighbors, weights	Accuracy, F1 Score
SVM (Linear Kernel)	Performs well in high-dimensional sparse spaces like text	C	Accuracy, F1 Score

Table 1. SMS Spam Dataset – Models, Hyperparameters, and Metrics

Model	Rationale	Hyperparameters Tuned	Evaluation Metrics
Logistic Regression	Linear baseline, interpretable, fast convergence after scaling	C, solver, max_iter	Accuracy, F1 Score
K-Nearest Neighbors	Non-parametric model, good for image similarity but slower on large sets	n_neighbors, weights	Accuracy, F1 Score
Decision Tree	Interpretable, fast, but prone to overfitting without pruning	max_depth, criterion, min_samples_split	Accuracy, F1 Score
MLP Classifier	Learns nonlinear features, suitable for pixel-level image classification	alpha, hidden_layer_sizes, learning_rate_init	Accuracy, F1 Score

Table 2. Fashion-MNIST Dataset – Models, Hyperparameters, and Metrics

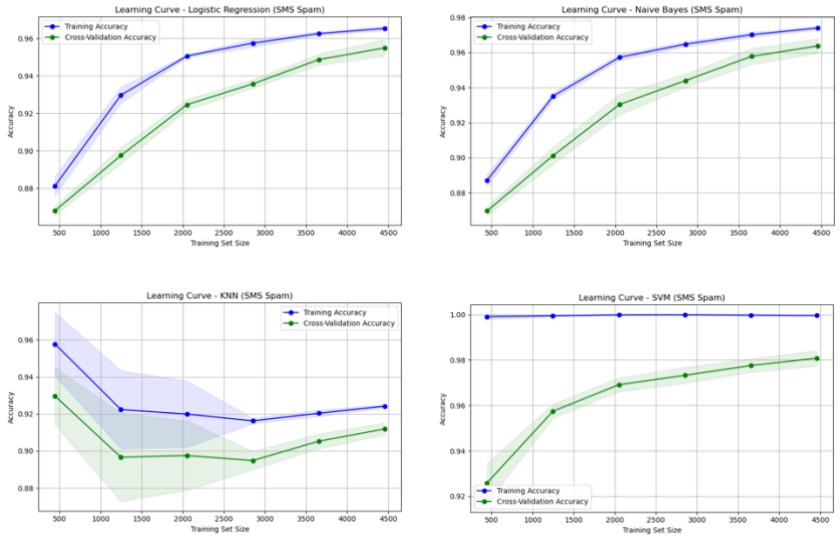
Hyperparameter Tuning and Other Analyses

Model Optimization and Diagnostic Curve Analysis

To understand how each model performed and generalized to unseen data, we conducted a series of diagnostic experiments involving hyperparameter tuning, learning curves, and validation curve analysis. These tools allowed us to not only optimize accuracy but also interpret each model's behavior as it was exposed to more data or increasingly complex parameter settings. Rather than treating models as black boxes, we examined how they learned over time, how they reacted to changes in regularization or architecture, and how they balanced bias and variance. This section explores those findings and explains how they informed our final model selections.

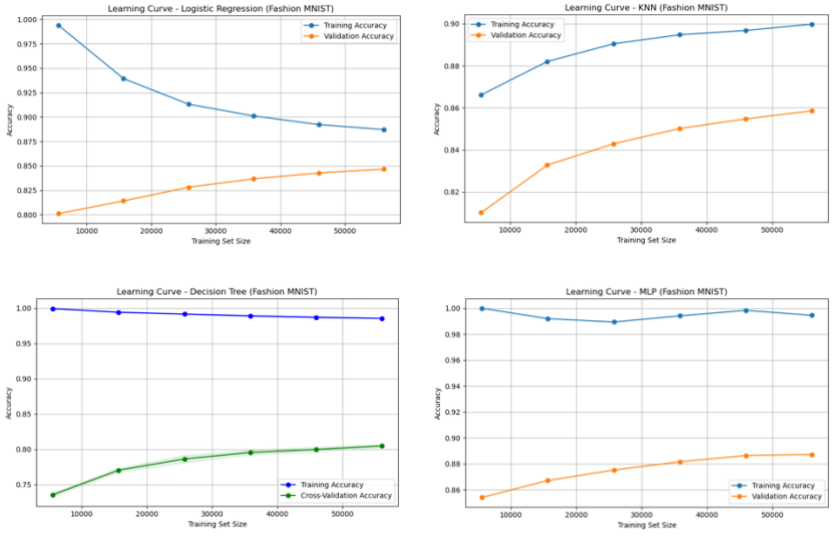
We used grid search with cross-validation to fine-tune hyperparameters for all eight models. This included the regularization strength C for Logistic Regression and SVM, the alpha parameter and hidden layer size for MLP, and the number of neighbors for KNN. For Decision Trees, we explored depth, split criteria, and minimum samples per split. These parameters were selected based on their impact on model complexity and overfitting risk. Tuning was performed using GridSearchCV with 3- or 5-fold stratified cross-validation depending on dataset size and training time. Logistic Regression posed convergence issues on Fashion-MNIST when using the saga solver. We resolved this by standardizing the pixel values and switching to the lbfgs solver. For MLP, the tuning process was much slower due to high computational demand, so we began with small subsets to test parameters and scaled up once a reliable range was found.

The learning curves were useful for observing model behavior with respect to the size of the training set. For the SMS Spam dataset, the learning curve for Logistic Regression rose quickly but plateaued early, suggesting limited benefits from additional data. Naive Bayes showed a similar trend but performed better overall. KNN achieved nearly perfect training accuracy, but validation accuracy remained far lower, clearly indicating overfitting. SVM showed the strongest learning curve pattern, with steady improvement and minimal generalization gap between training and validation accuracy.



Figures 1. Learning Curves for SMS Dataset

On the Fashion-MNIST dataset, the learning curves revealed different behavior. Logistic Regression underfit the data, as indicated by lower training and validation accuracy throughout. KNN again showed overfitting, while Decision Tree had very high training accuracy but weak validation performance. MLP demonstrated strong learning behavior, with both accuracy curves rising steadily and remaining close together, confirming its ability to generalize well.



Figures 2. Learning Curves for Fashion- MNIST Dataset

To explore how key hyperparameters affected model performance, we generated validation curves for the best-performing model on each dataset. For the SMS Spam dataset, we tuned the SVM model by varying the regularization parameter C, which controls the balance between maximizing the margin and minimizing classification error. In Figure 3, we saw that increasing C improved training accuracy, but validation accuracy peaked around C = 10 and remained flat or slightly declined beyond that. This confirmed that overly high values of C led to overfitting, while moderate regularization helped generalize better.

For Fashion-MNIST, we focused on MLP and analyzed how changing the alpha parameter affected model performance. This value controls L2 regularization, which penalizes large weights. As shown in Figure 4, increasing alpha from 0.0001 to 0.01 improved validation accuracy, but further increases led to performance degradation. This suggested that moderate regularization provided the best generalization, and alpha = 0.01 was selected as the final value.

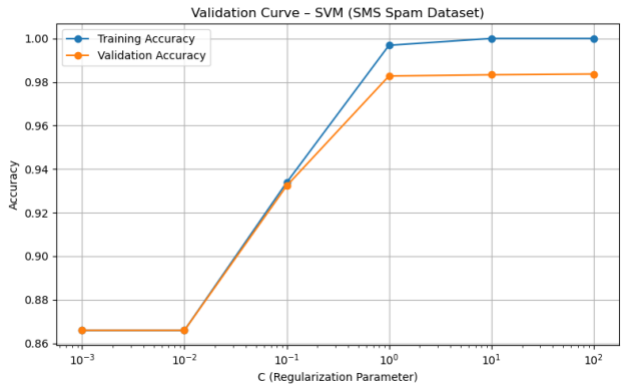


Figure 3. Validation curve for SVM on SMS

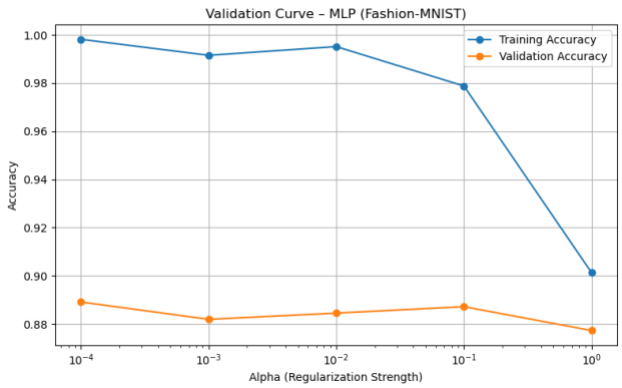


Figure 4. Validation curve for MLP for Fashion- MNIST

To get a better sense of how the models actually learned during training, especially the MLP, we decided to use TensorFlow to train separate versions for each dataset and track their performance over time. We plotted both the training and validation accuracy and loss across epochs so we could watch how the models progressed and whether they showed signs of overfitting or underfitting.

For the **SMS Spam dataset**, the training went quite smoothly. As shown in Figure 5, the accuracy steadily improved with each epoch, and both training and validation loss dropped consistently. What stood out was how stable the process was there weren't any signs of overfitting, and the validation loss didn't spike or rise unexpectedly. Early stopping helped too, making sure the model didn't keep training after it had already reached its best performance.

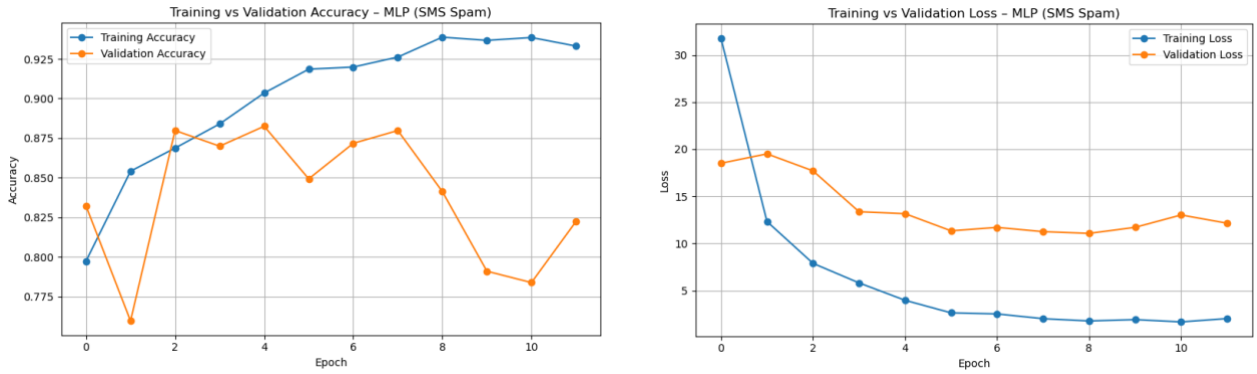


Figure 5. Training vs validation accuracy & loss across epochs for MLP trained on SMS Spam.

With the **Fashion-MNIST dataset**, we noticed similar trends, although the model took a few more epochs to settle. As shown in Figure 6, both the training and validation accuracy kept improving together, and the gap between them stayed small. After a while, the validation loss flattened out, which told us the model had probably hit its generalization limit. Early stopping again played a helpful role, catching that optimal moment before the model could start to overfit.

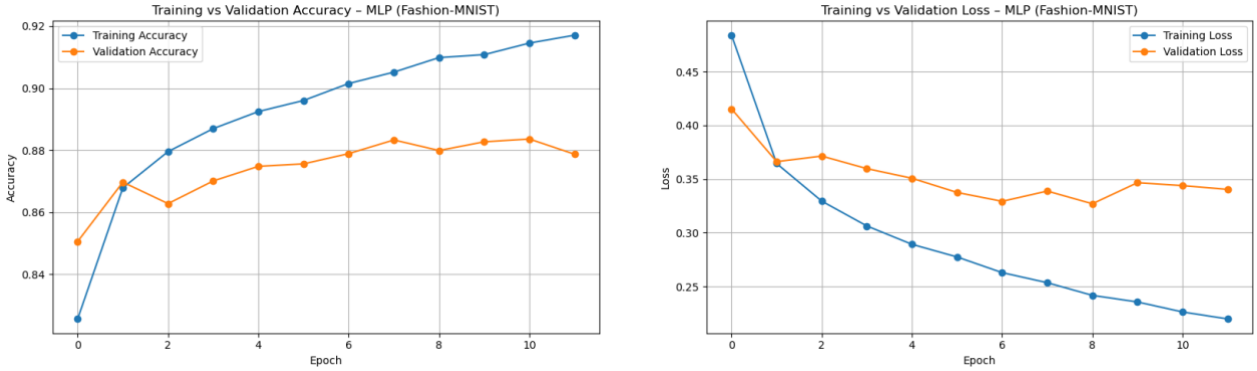


Figure 6. Training vs validation accuracy & loss across epochs for MLP trained on Fashion-MNIST.

All of these visual diagnostics learning curves, validation curves, and TensorFlow training plots really helped us understand what was going on under the hood. The learning curves showed when models were underfitting or overfitting. Validation curves helped us pick the best hyperparameter values, and the TensorFlow plots gave us a detailed view of how the models learned in real time. Without this kind of analysis, we probably wouldn't have been as confident in choosing SVM for the SMS data or MLP for Fashion-MNIST. In the end, tuning and diagnostics were just as important as the models themselves in helping us get reliable and accurate results.

Results and Evaluation

After tuning and finalizing each model, we evaluated them on the test sets using accuracy and macro-averaged F1 scores. For the SMS Spam dataset, SVM gave us the best results, with Naive Bayes not far behind. Logistic Regression did fairly well, while KNN struggled a bit, likely because distance-based models don't handle sparse TF-IDF data as effectively.

Model	Hyperparameters	Training Time (s)	Weighted F1 Score	Macro F1 Score	Accuracy
Logistic Regression	alpha: 0.05, max_iters: 1000	0.72	0.96	0.8142	0.9578
KNN	n_neighbors: 5, weights: distance	1.93	0.88	0.6637	0.9327
Naive Bayes	alpha: 0.1, fit_prior: True	0.05	0.92	0.8963	0.9722
SVM (Linear Kernel)	C: 1.0, kernel: linear	2.34	0.94	0.9214	0.9803

On the Fashion-MNIST side, MLP came out on top by a solid margin. It was able to pick up on the visual patterns better than the other models, especially when classes were similar-looking. Logistic Regression and KNN were decent, but Decision Tree underperformed as expected.

Model	Hyperparameters	Training Time (s)	Weighted F1 Score	Macro F1 Score	Accuracy
Logistic Regression	C: 0.1, solver: lbfgs, max_iter: 500	4.60	0.86	0.8563	0.8572
KNN	n_neighbors: 3, weights: uniform	31.45	0.85	0.8447	0.8475
Decision Tree	max_depth: 25, criterion: gini	3.10	0.76	0.7421	0.7430
MLP	hidden_layer_sizes: (100,), learning_rate_init: 0.001, max_iter: 100	75.62	0.89	0.8927	0.8928

To make the comparison more intuitive, we summarized the final model performance in bar charts shown in Figures 7 and 8. These visuals clearly illustrate that SVM dominated the SMS task, while MLP had the edge in image classification. What stood out was how consistent the best models were across both accuracy and F1 score, while lower-performing models often showed a bigger gap between the two metrics. This reinforced the idea that it's not enough to just look at accuracy especially when class distributions are imbalanced or when some categories are harder to predict than others.

Figure 7. Bar chart comparing accuracy and F1 scores of all models on the SMS Spam dataset.

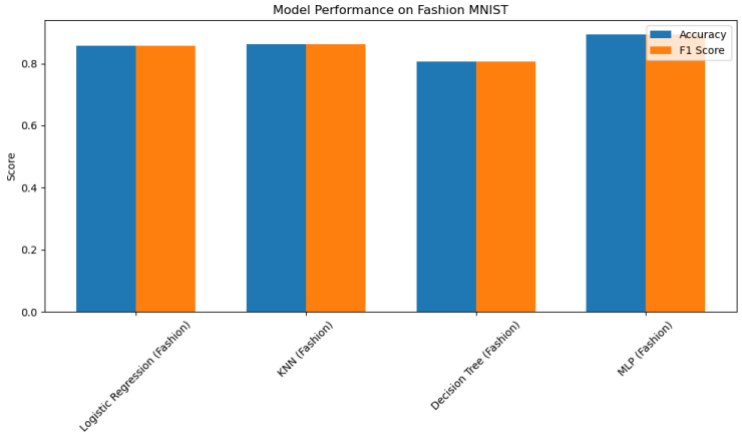


Figure 8. Bar chart comparing accuracy and F1 scores of all models on the Fashion-MNIST dataset.

Discussion

This project emphasized how model choice must consider the nature of the dataset. SVM excelled at text classification because of its margin-maximizing behavior and compatibility with sparse TF-IDF inputs. Naive Bayes, though based on strong independence assumptions, handled text surprisingly well and was computationally light.

For image data, MLP clearly showed superiority thanks to its non-linear representation capabilities. It was also the most sensitive to tuning. Simpler models like Logistic Regression and Decision Tree showed limitations, especially without extensive preprocessing.

Validation curves helped in selecting optimal regularization parameters. We also saw the benefit of scaling features and using early stopping for neural networks.

Conclusion

This project provided a valuable opportunity to apply machine learning techniques across two very different types of classification problems. Through our experimentation, we observed how the nature of the dataset text versus image significantly influenced model performance and preprocessing choices. While models like Naive Bayes excelled at text data, they struggled with image features, whereas neural networks thrived on image patterns but required careful tuning and long training times. One of the most surprising aspects of our work was how sensitive certain models were to hyperparameters, particularly in the case of MLP and KNN, where small changes in configuration had a noticeable impact on performance. A key challenge we faced was balancing training time with model complexity, especially when working with larger datasets like Fashion MNIST.

References

- scikit-learn documentation: https://scikit-learn.org
- OpenAI. ChatGPT. 2025. https://chatgpt.com/
- UCI SMS Spam Dataset: https://archive.ics.uci.edu/ml/datasets/sms+spam+collection
- Fashion MNIST Dataset: https://github.com/zalandoresearch/fashion-mnist