Image Classification with Random Forest

Mohina Ahmadi, Pavan Kaushik Aduri, and Pankaj Kumar Tiwari
Texas A&M University, mohina1729@tamu.edu, pkavu_1998@tamu.edu, contact_pankaj@tamu.edu

***Abstract*— This project aims to establish a robust methodology for classifying items within the FashionMNIST dataset. Employing a systematic approach, the study involves thorough data preparation and exploratory data analysis (EDA), ultimately selecting the Random Forest Classifier as the optimal model. The chosen Random Forest algorithm is subjected to a comprehensive development and fine-tuning process, optimizing hyperparameters for enhanced performance. Evaluation metrics are employed to provide a thorough assessment of the Random Forest Classifier's efficacy in classification. The resultant testing accuracy achieved an impressive 87.59%. This paper presents a detailed account of the methodology, showcasing the precision and effectiveness of the proposed approach in addressing the challenges of classifying items within the FashionMNIST dataset.**

Keywords—Image Classification, Random Forest Classifier, Exploratory Data Analysis

I. INTRODUCTION

In the field of machine learning, precise classification within datasets like FashionMNIST is crucial, particularly in industries like e-commerce and fashion. This paper establishes a robust methodology for FashionMNIST classification, starting with meticulous data preprocessing on a dataset of 60,000 training samples and 10,000 test samples across 10 classes. Advanced techniques like PCA and t-SNE are applied to unveil intricate patterns in the high-dimensional image data. The training data is strategically split for model development, with the Random Forest classifier chosen and hyperparameter tuning performed for optimization. The research concludes by fitting the test data to the tuned model, evaluating performance with relevant metrics, providing valuable insights for image classification in machine learning.

The study by B. Xu et al [1] introduces an enhanced random forest algorithm for image classification, addressing challenges in analyzing high-dimensional image data. The algorithm effectively reduces subspace size, improves classification accuracy, and minimizes error bounds. The paper by P. Mekha et al [2] discusses the application of image classification techniques to identify and classify rice leaf diseases. The research evaluates various classification algorithms, including Random Forest, Gradient Boosting, and Naïve Bayes. The paper by S.-H.Kim et al [3] introduces an innovative algorithm designed for the efficient classification of X-ray images, aiming to enhance accuracy and performance. The proposed method employs Local Binary Patterns (LBP) for texture feature extraction and utilizes Random Forests for fast and accurate classification. Drawing inspiration from these studies, our work leverages Random Forest as a central component. This decision is rooted in the algorithm's ability to handle high-dimensional data and showcase robust performance, particularly when enhanced with novel feature weighting techniques and tree selection methods.

II. METHODS

The methodology encompasses thorough analysis of the training dataset through cleaning, normalization, and exploratory data analysis (EDA), including PCA and t-SNE analysis. A Random Forest Classifier is then implemented for image classification, fine-tuned via hyperparameter tuning, and the best-fit model is rigorously tested on the test dataset for optimal performance.

A. Data Preparation

TensorFlow loads the balanced Fashion-MNIST dataset with 60,000 training samples and 10,000 test samples, containing diverse fashion accessories represented as 28x28 pixel grayscale images. A seamless data cleaning process reveals no missing values, and a normalization and transformation step scales pixel values to [0,1], flattening the dataset into 784 columns. The ten accessory categories, labeled 0 to 9, are prepared for analysis, and the dataset is split into training and validation subsets for robust model development.

Fig.1. Image Dataset

B. *Exploratory Data Analysis (EDA)*

1) Data Visualization

A count plot has been generated to visually represent the distribution of fashion accessory classes within the training dataset. Notably, each class exhibits a uniform distribution, with precisely 6000 instances allocated to each class. This visualization offers a clear and concise overview of the balanced distribution of fashion accessory categories across classes in the dataset.

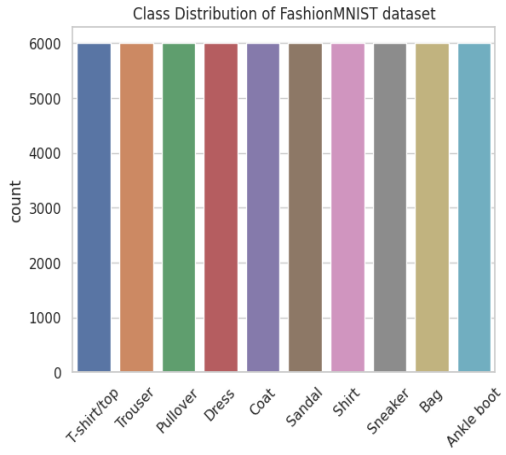

Fig. 2. Class Distribution

Descriptive statistics, including mean, median, and standard deviation, unveil essential insights into the image dataset. With a normalized mean of 0.286, grayscale images are indicated, while a low standard deviation of 0.35 suggests less variation and pixel values clustered around the mean. These statistical measures collectively illuminate the dataset's brightness, central tendency, and variability, crucial for interpreting visual characteristics. Calculating these metrics across columns provides a comprehensive understanding of pixel intensity variation.

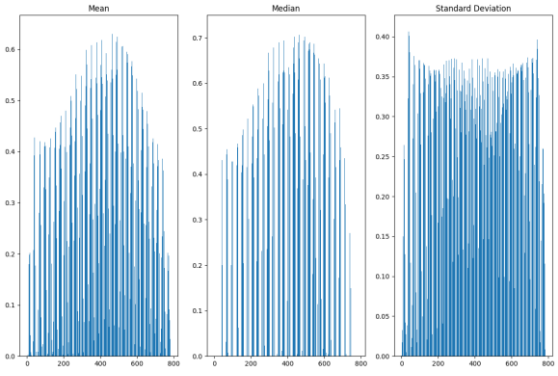

Fig. 3. Plots of Mean, Median and Standard Deviation

Analyzing the covariance matrix reveals feature relationships. In Fig. 4's heatmap, light blue hues indicate negative linear relationships, with dark blue highlighting strong negative correlations and vibrant yellow denoting robust positive relationships. The yellow diagonal represents feature covariance with itself, indicating feature variance. In image data, a bright yellow diagonal implies substantial pixel intensity variation, revealing pixels with diverse intensity values across the images.

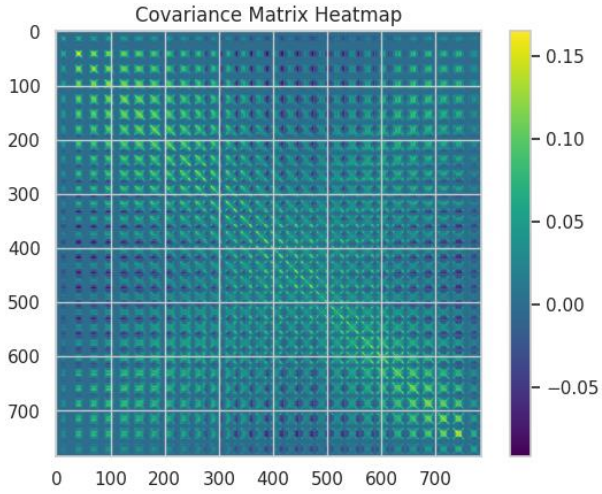

Fig. 4. Covariance Matrix Heatmap

As shown in Fig 5, including histograms on the covariance matrix diagonal offers crucial insights into the univariate distribution of each feature. These histograms visually showcase pixel intensity patterns, indicating a left-skewed distribution that signifies a dispersion of pixel values rather than concentration. Key takeaways involve peaks representing common intensity levels and the prevalence of specific intensity ranges, assisting machine learning algorithms in pattern recognition.

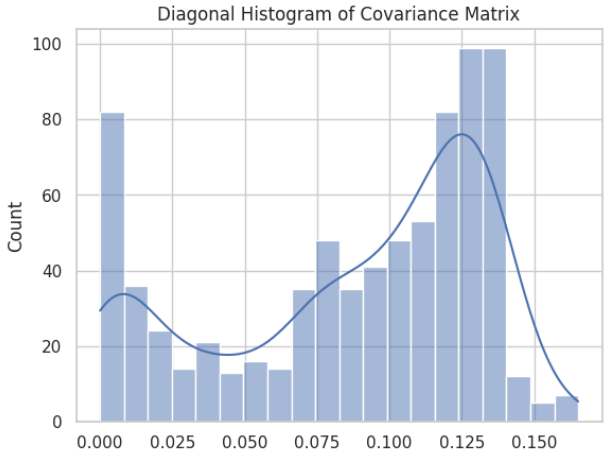

Fig. 5. Histogram of covariance matrix

2) PCA Analysis

Principal Component Analysis (PCA) stands as a powerful statistical method for dimensionality reduction in high-dimensional datasets. Scree plots aid in selecting the optimal number of components to retain, identifying the point where eigenvalues level off, ensuring a balance between data reduction and information retention. In the context of the FashionMNIST dataset with 784 features per image, PCA was applied to visualize and interpret the high-dimensional data. From the scree plot shown in Fig. 5, we can observe that there are more than 3 principal components before the elbow point and the maximum variance ratio is 0.3 (PCA assumes

linear relationship), suggesting that features have complex nonlinear relationships between them and PCA is unable to capture nonlinear relationships. As a rule of thumb, 2-3 principal components would be sufficient to retain more than 80 percent of the variance if there are linear relationships between features of the data. Plotting the first 2 principal components of PCA helps us visualize the clusters of classes and we can see that these clusters are tightly overlapped as PCA assumes the relationship between the features as linear. If the features have only linear relationships, then clusters formed by the first 2 principal components would be distinct from each other.

Recognizing the non-linear nature of the FashionMNIST dataset, t-SNE (t-distributed Stochastic Neighbor Embedding) was employed after PCA. This nonlinear technique excels in capturing intricate relationships between features, providing a comprehensive interpretation of the dataset's complexity. Observing the plot from the first 2 principal components of t-SNE, we see that clusters are comparably distinct from each other (PCA). Thus, the combined use of PCA and t-SNE ensures a good understanding of the FashionMNIST dataset, enhancing its suitability for advanced machine learning models capable of handling non-linear structures

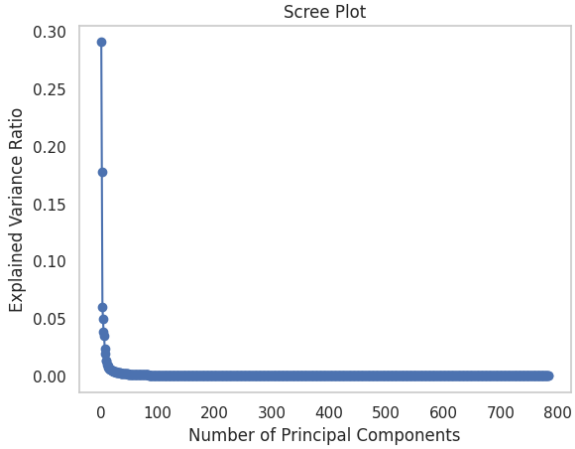

Fig. 6. Scree Plot

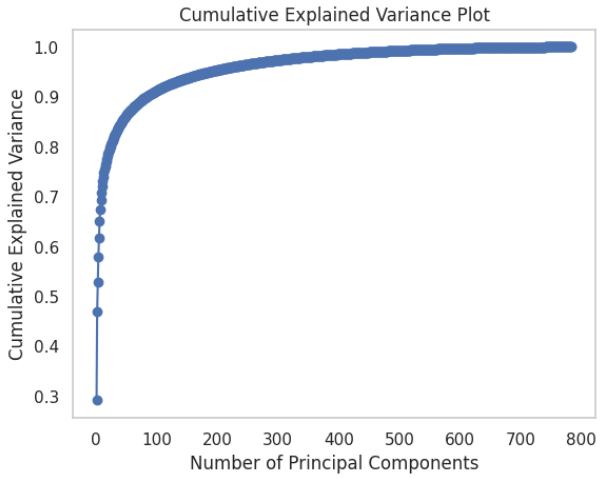

Fig. 7. Cumulative Explained Variance Plot

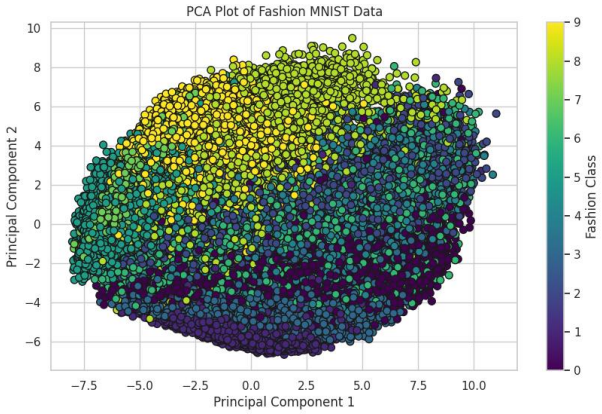

Fig. 8. PCA Plot

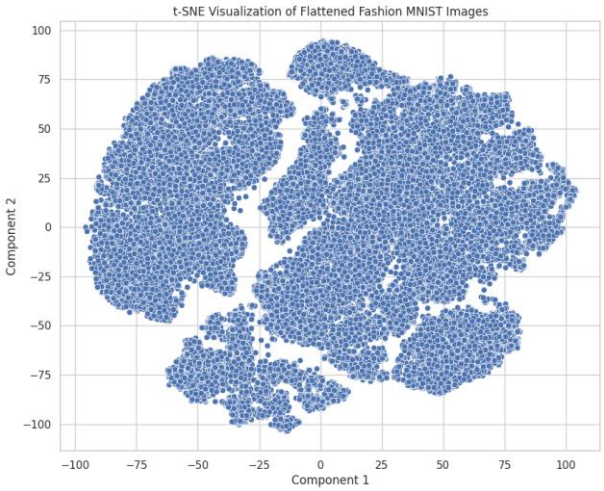

Fig. 9. t-SNE Plot

C. Random Forest Classifier

In recognizing the dataset's intricate non-linear relationships, we sought a model proficient in capturing such complexities, leading us to opt for the Random Forest Classifier, particularly fitting for our moderate-sized dataset. Random Forests, as an ensemble of decision trees, introduce randomness through bootstrapped sampling and feature randomness during training, enhancing generalization, accuracy, and robustness. Their ability to mitigate overfitting is particularly advantageous for handling noisy datasets, providing higher accuracy by combining diverse trees. The ensemble nature of Random Forest, adept at addressing challenges in image classification, makes it well-suited for high-dimensional datasets representing images. Moreover, Random Forest's capability to identify feature importance is valuable in image analysis, offering insights into the significant aspects of an image that contribute to classification decisions. Its ease of implementation and versatility make Random Forest a practical choice for image classification tasks, especially when computational resources may be limited or when interpretability is a priority.

Utilizing the Random Forest classifier from scikit-learn, we conducted effective hyperparameter tuning with a

validation dataset. The tuned model was then applied to the test dataset, and various metrics were evaluated.

1) Model Interpretability:

The Random Forest algorithm builds a collection of decision trees during training and outputs the mode of the classes of the individual trees for a given input as shown in Fig. 10. The steps are as follows:

a) Bootstrapped sampling: For each tree in the Random Forest, a bootstrapped sample is created by randomly selecting a subset of images with replacement from the training dataset.

b) Selecting random features: At each node of a decision tree, a random subset of features is considered for determining the best split. This introduces diversity among the trees and helps prevent overfitting.

c) Building decision trees: For each tree, the decision tree is constructed recursively. At each node, a subset of features is considered, and the algorithm chooses the feature and threshold that best splits the data based on a criterion such as Gini impurity.

This process is repeated until a stopping criterion is met, such as reaching a maximum depth or having a minimum number of samples in a leaf node. The above steps are repeated to create a predefined number of decision trees in the Random Forest. Each tree is constructed independently, resulting in a collection of diverse trees.

d) Prediction Phase: For a new image in the test set, each tree in the Random Forest predicts the class. Since we are doing classification, the mode or maximum votes of all tree predictions are taken as the final predicted class for the input image.

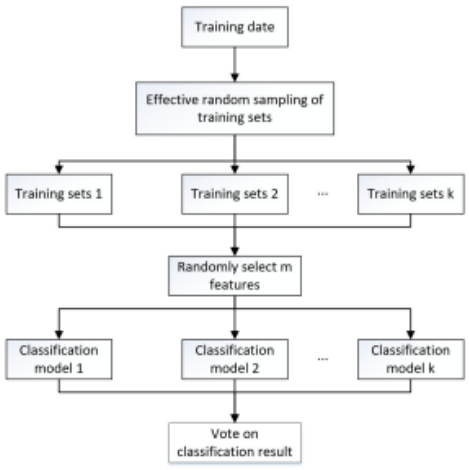

Fig. 10. Flow chart of Random Forest Algorithm

III. EXPERIMENTS

A. Hyperparameter Tuning for Optimal Model Selection

This section details the hyperparameter tuning process, employing a validation dataset with 10,000 samples to identify the optimal configuration for our Random Forest model. The objective is to boost accuracy on the validation set, subsequently improving overall performance on the test set. Notably, tuning n_estimators and max_depth hyperparameters has proven effective in enhancing model performance, while min_samples_split and min_samples_leaf is employed to mitigate overfitting. We have used criteria as gini impurity since we have a balanced dataset where samples are distributed equally across the classes.

We initiated our experiments with default hyperparameters, resulting in an initial accuracy of 88.43%. As shown in Table I, in the initial phase of hyperparameter tuning, we explored the impact of varying the max_depth parameter while keeping the number of trees (n_estimators) fixed at 50. In the second phase, we increased the number of trees to 100 while tuning the max_depth parameter. In the third phase, we further explored configurations with 200 trees, resulting in a slight improvement compared to previous iterations. The following phases with 300 trees and 500 trees exhibited no significant improvement, indicating that increasing the number of trees beyond a certain threshold may lead to computational complexity without substantial gains in performance. Therefore, achieving the optimal balance between computational efficiency and model accuracy requires careful consideration in hyperparameter tuning. Also, whenever max_depth is 10, there is reduction in accuracy (85%) for all n_estimator values.

We employed PCA to train the model on the training dataset, exploring three different values for n_components (2, 10, 100) and varying combinations of n_estimators and max_depth hyperparameters. Intriguingly, with n_components set to 2, the accuracy plateaued at 50% across multiple configurations of n_estimators and max_depth. However, when n_components increased to 10 or beyond, we observed that the accuracy mirrored that of the model where PCA was not applied.

TABLE I. HYPERPARAMETER TUNING

n_estimators	Max_depth	Validation Accuracy
50	10	85.52%
	50	88.22%
	100	88.24%
100	10	85.67%
	50	88.46%
	100	88.43%
200	10	85.65%
	50	88.51%
	100	88.49%
300	10	85.77%
	50	88.39%
	100	88.41%
500	10	85.71%
	50	88.50%
	100	88.53%

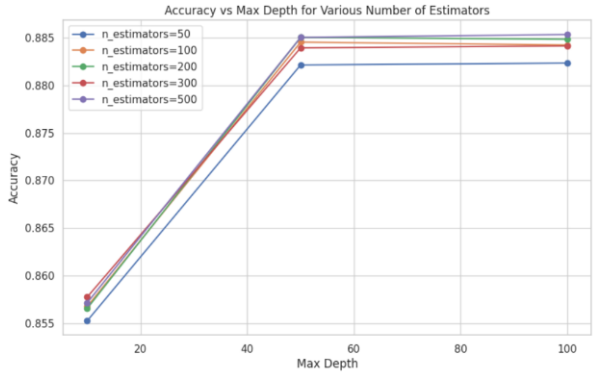

Fig. 11. Accuracy vs Max Depth for various number of estimators

B. *Testing on Test Data*

After thorough experimentation, the optimal model configuration was identified as estimators=200 and max_depth=50, considering computational complexity (time taken to execute) achieving an accuracy of 88.51% on validation dataset and accuracy of 87.57% on the test dataset. The results highlight the importance of tuning hyperparameters to achieve the best model performance. In assessing machine learning model performance, accuracy gauges overall correctness, precision assesses the accuracy of positive predictions, recall measures the model's ability to capture actual positives, and F1 score strikes a balance between precision and recall as shown in Table II. These metrics provide insights into different facets of model effectiveness, with F1 score being especially useful in handling imbalanced class distributions. As we have a balanced dataset, accuracy holds greater significance in assessing model performance.

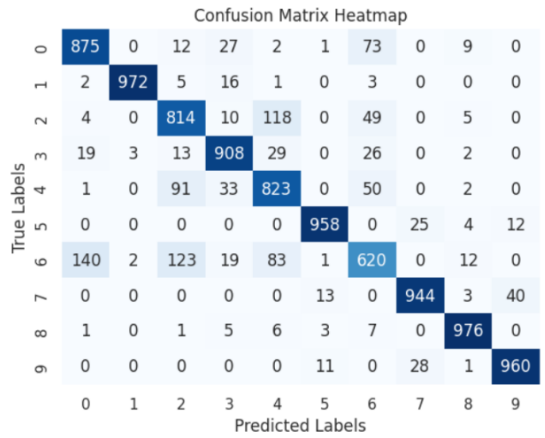

Fig. 12. Confusion Matrix for test data

TABLE II. EVALUATION ON TEST DATA

Best Fit Model with n_estimators=200, max_depth=50

Accuracy	87.57%
Precision	0.8776
Recall	0.8781
F1 Score	0.8766

IV. RESULTS AND CONCLUSION

The study concludes that Random Forest classifier with careful hyperparameter tuning contributes to accurate classification. The optimal model (n_estimators=200, max_depth=50) achieved an accuracy of 87.57% on the test dataset, showcasing the effectiveness of the proposed methodology. The results underscore the significance of balancing computational efficiency with model accuracy in hyperparameter tuning, providing valuable insights for image classification tasks in machine learning.

1) *Business Insights*

Random Forest's adaptability shines in its ability to handle diverse clothing styles within the dataset and efficiently process large volumes of images. The potential for real-time applications, such as online shopping platforms, underscores its quick and accurate categorization capabilities. Its versatility extends across various product categories, requiring minimal modifications for different classifications. Furthermore, Random Forest's adaptability allows for incremental updates as new data emerges, ensuring the model stays current with evolving fashion trends. In summary, Random Forest stands out as a versatile and effective solution for image classification in the dynamic realm of FashionMNIST, offering a robust balance between accuracy, interpretability, and adaptability.

REFERENCES

[1] B. Xu, Y. Ye and L. Nie, "An improved random forest classifier for image classification," 2012 IEEE International Conference on Information and Automation, Shenyang, China, 2012, pp. 795-800, doi: 10.1109/ICInfA.2012.6246927.

[2] P. Mekha and N. Teeyasuksaet, "Image Classification of Rice Leaf Diseases Using Random Forest Algorithm," 2021 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering, Cha-am, Thailand, 2021, pp. 165-169, doi: 10.1109/ECTIDAMTNCON51128.2021.9425696.

[3] S. -H. Kim, J. -H. Lee, B. Ko and J. -Y. Nam, "X-ray image classification using Random Forests with Local Binary Patterns," 2010 International Conference on Machine Learning and Cybernetics, Qingdao, China, 2010, pp. 3190-3194, doi: 10.1109/ICMLC.2010.5580711.