END-TO-END IMAGE CLASSIFICATION AND MODEL COMPARISON

A Comparative Study Using Transfer Learning and Classical Machine Learning Algorithms

Word Count: 1967

Table of Contents

1. Introduction 3
   - Project Overview and Problem Definition 3
   - Objectives and Scope 3
2. Dataset Description and EDA 4
   - Dataset Source and Structure 4
   - Exploratory Data Analysis (EDA) 4
3. Data Preprocessing and Feature Extraction 5
   - Preprocessing pipeline 5
     - Image loading and resizing 5
     - Label Encoding 5
   - Transfer learning approach 5
     - Selected CNN 5
     - Feature Extraction approach 5
4. Model Building, Training, and Hyperparameter Tuning 6
   - Rationale for model selection 6
   - Training Pipeline 6
   - Modeling CNNs 6
     - AlexNet 6
     - VGG16 6
     - MobileNetV2 7
   - Modeling Traditional Classifiers 7
     - Logistic Regression 7
     - Support Vector Machine (SVM) 7
     - Random Forest 7
     - XGBoost 7
     - Ensemble (Logistic Regression, SVM and Random Forest) 8
5. Results and Comparative Analysis 9
   - Performance metrics overview 9
   - Visualizations 9
   - Detailed Comparison 9
     - CNN performance: 9
     - Traditional Classifier performance: 10
     - Ensemble Model (Logistic Regression, SVM, and Random Forest): 10
   - Interpretation of Metrics 10
     - Feature Extractor Influence: 11
     - Traditional vs. Neural Network Performance: 11
     - The Role of Hyperparameter Tuning: 11
     - Computational Trade-offs: 11
6. Discussions and Insights 11
   - Error Analysis 11

Strengths and limitations 12
Feature Extractors 12
Classifiers 12
Ensemble Method 12
Impact of Data Preprocessing 12
Future Improvements: 12
7. Conclusion 13
8. References 14
9. Appendix 15
Code Notebooks 15
Confusion matrices and classifier reports 16
Confusion Matrix AlexNet 16
Classification Report AlexNet 17
Confusion Matrix VGG16 18
Classification Report VGG16: 19
Confusion Matrix MobileNetV2: 20
Classification Report MobileNetV2: 21
Confusion Matrix Logistic Regression: 21
Classification Report Logistic Regression: 22
Confusion Matrix SVM: 23
Classification Report SVM: 23
Confusion Matrix Random Forest: 24
Classification Reports Random Forest: 24
Confusion Matrix XGBoost: 25
Classification Report XGBoost: 26
Confusion Matrix Ensemble: 27
Classification Report Ensemble: 28
Visualizations 28
EDA 1: Visual Inspection of Sample Images 28
EDA 2: EDA - Class Distribution Analysis 30
EDA 3: Pixel Intensity Distribution Examination 31
EDA 4: File Size Analysis via Scatterplots 32
EDA 5: Metadata Inspection 33
EDA 6: Brightness Distribution Analysis Using Box Plots 34

1. Introduction

Project Overview and Problem Definition

Image classification is critical in computer vision, with applications from autonomous driving to medical diagnostics (LeCun, Bengio and Hinton, 2015). This project develops an end-to-end pipeline using the Intel Image Classification dataset from Kaggle (Kaggle, n.d.), comprising six classes (buildings, forest, glacier, mountain, sea, street) each with challenges of variability and high dimensionality. By leveraging transfer learning and classical classifiers, we aim to build a robust, reproducible system.

Objectives and Scope

Our objectives are twofold: first, to build a reproducible classification pipeline covering preprocessing, feature extraction, training, and evaluation; second, to systematically compare model combinations using metrics such as weighted F1 scores and confusion matrices. We leverage transfer learning with pre-trained CNNs (AlexNet, VGG16, MobileNetV2) for feature extraction and apply classical classifiers (Logistic Regression, SVM, Random Forest, XGBoost) and an ensemble approach (Géron, 2019; Tan et al., 2018).

2. Dataset Description and EDA

Dataset Source and Structure

We obtained the Intel Image Classification dataset from Kaggle via an API key and copied it to a separate directory to preserve the original files. The dataset is organized into training, test, with images distributed across six classes, and a validation set, enabling robust evaluation.

Exploratory Data Analysis (EDA)

The EDA phase is designed to provide an understanding of the dataset's structure, quality, and distribution. The following steps were performed:

1. Visual Inspection of Sample Images (EDA 1)
2. Class Distribution Analysis (EDA2)
3. Pixel Intensity Distribution Examination (EDA 3)
4. File Size Analysis via Scatterplots (EDA 4)
5. Metadata Inspection (EDA 5)
6. Brightness Distribution Analysis Using Box Plots (EDA 6)

By systematically carrying out these EDA steps, we develop a comprehensive understanding which lays a strong foundation for the subsequent feature extraction and modelling phases.

3. Data Preprocessing and Feature Extraction

Preprocessing pipeline

The initial stage of the project involves preparing the dataset through a systematic preprocessing pipeline.

Image loading and resizing

Images are loaded using TensorFlow's Keras API and resized to a uniform dimension (150×150 pixels). This resizing standardizes the input for the CNNs, balancing computational efficiency with the preservation of essential visual details (Géron, 2019).

Label Encoding

Textual labels such as 'buildings' and 'forest' are converted into numerical values using a LabelEncoder. This transformation ensures that the classifiers can efficiently process the categorical data, aligning with scikit-learn's modelling requirements (Scikit-learn, n.d.).

Transfer learning approach

Selected CNN

Pre-trained convolutional neural networks (CNNs) serve as the backbone for feature extraction. In this project, three CNN architectures are used:

- AlexNet
- VGG16
- MobileNetV2

Feature Extraction approach

For consistency, we use MobileNetV2 as the feature extractor. Its output, standardized via StandardScaler, feeds into all downstream classifiers. Although different CNNs yield varying representations, comparing their F1 scores allows us to identify the optimal architecture for this task.

4. Model Building, Training, and Hyperparameter Tuning

Rationale for model selection

We chose models to balance simplicity and performance. Classical models (Logistic Regression, SVM) are efficient and interpretable, while ensemble methods (Random Forest, XGBoost) capture complex patterns. Neural Networks provide benchmark comparisons.

Training Pipeline

A robust training pipeline was implemented using a Stratified K-Fold cross-validation strategy, ensuring that the class distribution is maintained in each fold. This method minimizes bias and provides a reliable estimate of model performance. For hyperparameter tuning, a grid search process was adopted, systematically exploring combinations of parameters such as regularization strength for Logistic Regression, kernel parameters for SVM, and tree depth and the number of estimators for Random Forest and XGBoost. The grid search optimizes the weighted F1 score, a key performance metric in this context (Scikit-learn, n.d.).

Modeling CNNs

AlexNet

We start with a simplified, non-pretrained AlexNet using 1000 images per class. It is trained with the Adam optimizer, cross-entropy loss, and early stopping to mitigate overfitting.

VGG16

We move on to the VGG16 model which has pre-trained weights. This model uses transfer learning by having the classifier layers unfrozen for feature extraction while its base remains frozen to retain learned features.
It uses stratified images up to 1000 images to prevent overloading. We also scale the pixels from 0 to 1. We have chosen to use an L2 regularizer with an Adam optimizer with cross-entropy as the loss function.

MobileNetV2

Since we got better results with VGG16, we can try a lighter model to increase the processing speed. MobileNetV2 is also pre-trained like the VGG16 and uses the same L2 regularizer, Adam optimizer and the cross-entropy loss function, and we use a stratified K-Fold validation. However, we also add an optional random erasing/masking function to see if the performance would improve.

Modeling Traditional Classifiers

In all traditional classifiers, to ensure we can input the image details into the model, we use transfer learning to extract image features from the pre-trained MobileNetV2 extractor as it is a quicker extractor than VGG16. GlobalAveragePooling2D stores the output in a fixed-length vector for the model to use. StandardScaler helps us scale the images to have a mean and variance of 0.

Logistic Regression

To view the performance of traditional classifiers, we start with the simplest logistic regression. We use Stratified K-Fold Validation to ensure each class has the same distribution, and we use hyperparameter C for regularization.

Support Vector Machine (SVM)

Similar to Logistic Regression, we use Stratified K-fold Validation for the training and hyperparameter C for regularization. Our goal here is just to see whether SVM would fare better than logistic regression for our task.

Random Forest

In this model, we still use Stratified K-Fold but instead of manually setting hyperparameters, we use GridSearchCV to tune hyperparameters by evaluates each combination using cross-validation to get the best performance.

XGBoost

In XGBoost, we change to use K-Fold Validation instead of Stratified K-Fold to see whether it would result in better performance. XGBoost has pre-trained weights and uses the multi-logarithmic loss function instead of cross-entropy or MSE. We want to explore whether these changes improves our performance over Random Forest.

Ensemble (Logistic Regression, SVM and Random Forest)

We want to explore whether another version of transfer learning would work any better than XGBoost. We keep to using Stratified K-Fold training but use ensemble methods with soft voting to make the final prediction.

5. Results and Comparative Analysis

Performance metrics overview

Comparison with Weighted F1 (rounded to 4 decimal places)	
AlexNet	0.7667
VGG16	0.8967
MobileNetV2	0.6552
Logistic Regression	0.8756
SVM	0.8718
Random Forest	0.8966
XGBoost	0.9054
Ensemble (Logistic Regression, SVM, RandomForest)	0.8900

The performance of the models was primarily evaluated using weighted F1 scores, which provide a balanced measure of precision and recall across classes. From the table, we see the model comparison revealed that XGBoost achieved best weighted F1 score, outperforming even CNNs like VGG16.

Visualizations

Visualizations, including confusion matrices and example predictions, provided insights into class-specific performance and common misclassifications.

Detailed Comparison

The comparative analysis of all models, both convolutional neural networks (CNNs) and traditional machine learning classifiers, provides a comprehensive understanding of their strengths and weaknesses. The following observations were made:

CNN performance

VGG16 emerged as the best CNN model, achieving a weighted F1 score of 0.8967, outperforming AlexNet (0.7667) and MobileNetV2 (0.6552). This reinforces the

effectiveness of deeper architectures with transfer learning, as VGG16 retains pre-trained features while allowing fine-tuning at the classification level.
AlexNet, trained from scratch, struggled to match the performance of transfer learning models. This indicates that when data is limited, training a deep model without pre-trained weights is less effective.
MobileNetV2 underperformed compared to the other CNNs, despite its efficiency in computational speed. This suggests that lightweight architectures optimized for mobile deployment may trade off accuracy for speed, making them less suitable for complex classification tasks without fine-tuning.

Traditional Classifier performance

XGBoost was the best-performing model overall, achieving the highest weighted F1 score of 0.9054. Its ability to handle structured data and optimize decision trees contributed to its superior performance over other classifiers and CNNs.
Random Forest (0.8966) performed competitively, closely matching VGG16, demonstrating the power of ensemble methods in capturing complex patterns. However, it slightly underperformed compared to XGBoost, likely due to XGBoost's superior boosting mechanism that optimally weighs misclassified instances.
Logistic Regression (0.8756) and SVM (0.8718) performed well, but were limited in capturing non-linear decision boundaries compared to tree-based models. While these models offer interpretability and efficiency, they struggle when high-dimensional features introduce complex relationships.

Ensemble Model (Logistic Regression, SVM, and Random Forest)

The ensemble approach, which leveraged soft voting from three different classifiers, achieved a weighted F1 score of 0.8900, higher than individual linear models but slightly lower than XGBoost.
The ensemble method successfully balanced biases from individual classifiers but did not exceed XGBoost's performance, indicating that boosting techniques are more effective in refining misclassifications.

Interpretation of Metrics

From the weighted F1 scores and confusion matrix insights, key takeaways include:

Feature Extractor Influence:

The choice of CNN feature extractor plays a crucial role in downstream classification performance. VGG16's superior performance highlights the advantage of deep networks trained on large-scale datasets, whereas AlexNet and MobileNetV2 did not generalize as well.

Traditional vs. Neural Network Performance:

Surprisingly, XGBoost outperformed all CNN-based approaches, demonstrating that for structured tabular representations of image features, decision-tree-based models can extract and refine patterns more effectively than deep learning models without full fine-tuning.

CNNs perform better when end-to-end training is possible, but when restricted to feature extraction, traditional classifiers can outperform them if the extracted features are sufficiently informative.

The Role of Hyperparameter Tuning:

Hyperparameter tuning significantly influenced model performance. XGBoost and Random Forest benefited from GridSearchCV-based hyperparameter optimization, while logistic regression and SVM saw moderate gains.

CNNs, particularly VGG16, leveraged early stopping and dropout, reducing overfitting and improving stability across validation sets.

Computational Trade-offs:

While XGBoost achieved the highest performance, CNN models required significantly longer training times. MobileNetV2, despite its lower accuracy, remains a viable option when computational efficiency is a priority.

6. Discussions and Insights

Error Analysis

Detailed examination of the confusion matrices revealed that certain classes, such as glaciers and mountains, were frequently misclassified. Some errors stemmed from overlapping visual features and inconsistent image quality across classes. A few images appear to have been incorrectly allocated to their respective classes, which could contribute to these misclassifications.

Strengths and limitations

Feature Extractors

VGG16 outperformed AlexNet and MobileNetV2, suggesting its deeper architecture extracts more discriminative features. However, even VGG16 is not immune to issues arising from low-quality or mislabelled images.

Classifiers

Classical models like Logistic Regression and SVM offer interpretability and speed, but struggle with the complexity inherent in image data. In contrast, ensemble methods (Random Forest and XGBoost) capture nonlinear relationships more effectively, though at a higher computational cost.

Ensemble Method

The ensemble approach, which integrates predictions from multiple classifiers, reduces individual model biases. Yet, it does not always guarantee superior performance compared to the best standalone model, as we see by the lower weighted F1 score compared to XGBoost.

Impact of Data Preprocessing

Consistent feature scaling and proper train-test splits were essential for reliable performance.

Future Improvements:

With additional time, further scrutiny of the dataset could be conducted to identify and reassign mislabelled images. Enhanced data augmentation and even fine-tuning of the pre-trained CNNs could lead to better feature extraction. Incorporating additional metadata, when available, might also enrich the model's ability to learn subtle distinctions. Specific changes can be tracked and compared so we may be able to better determine which factors led to XGBoost having the best F1 score.

7. Conclusion

This project developed an end-to-end image classification pipeline using the Intel Image Classification dataset from Kaggle. By combining rigorous preprocessing, transfer learning with CNNs (AlexNet, VGG16, MobileNetV2), and diverse classifiers (Logistic Regression, SVM, Random Forest, XGBoost, and an ensemble), we achieved competitive performance. Our results show that VGG16 provides superior features and that XGBoost, among traditional classifiers, attains the highest weighted F1 score. Future work will address data inconsistencies and explore advanced augmentation and fine-tuning techniques.

8. References

Géron, A. (2019) Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 2nd edn. Sebastopol: O'Reilly Media.

Kaggle (n.d.) Intel Image Classification dataset. Available at: https://www.kaggle.com/datasets/puneet6060/intel-image-classification (Accessed: 9 March 2025).

LeCun, Y., Bengio, Y. and Hinton, G. (2015) 'Deep learning', Nature, 521(7553), pp. 436–444.

Scikit-learn (n.d.) Scikit-Learn Documentation. Available at: https://scikit-learn.org/stable/documentation.html (Accessed: 9 March 2025).

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C. and Liu, C. (2018) 'A survey on deep transfer learning', in International Conference on Artificial Neural Networks. Springer, pp. 270–279.

TensorFlow (n.d.) TensorFlow Documentation. Available at: https://www.tensorflow.org/api_docs (Accessed: 9 March 2025).

XGBoost (n.d.) XGBoost Documentation. Available at: https://xgboost.readthedocs.io/en/latest/ (Accessed: 9 March 2025).

9. Appendix

Code Notebooks

Introduction.ipynb

Dataset setup and summary.ipynb

Exploratory Data Analysis.ipynb

- Visual Inspection of Sample Images (EDA 1)
- Class Distribution Analysis (EDA 2)
- Pixel Intensity Distribution Examination (EDA 3)
- File Size Analysis via Scatterplots (EDA 4)
- Metadata Inspection (EDA 5)
- Brightness Distribution Analysis Using Box Plots (EDA 6)

Data processing and cleaning.ipynb

Neural Networks.ipynb

1. AlexNet
2. VGG16
3. MobileNetV2

Logistic Regression.ipynb

SVM.ipynb

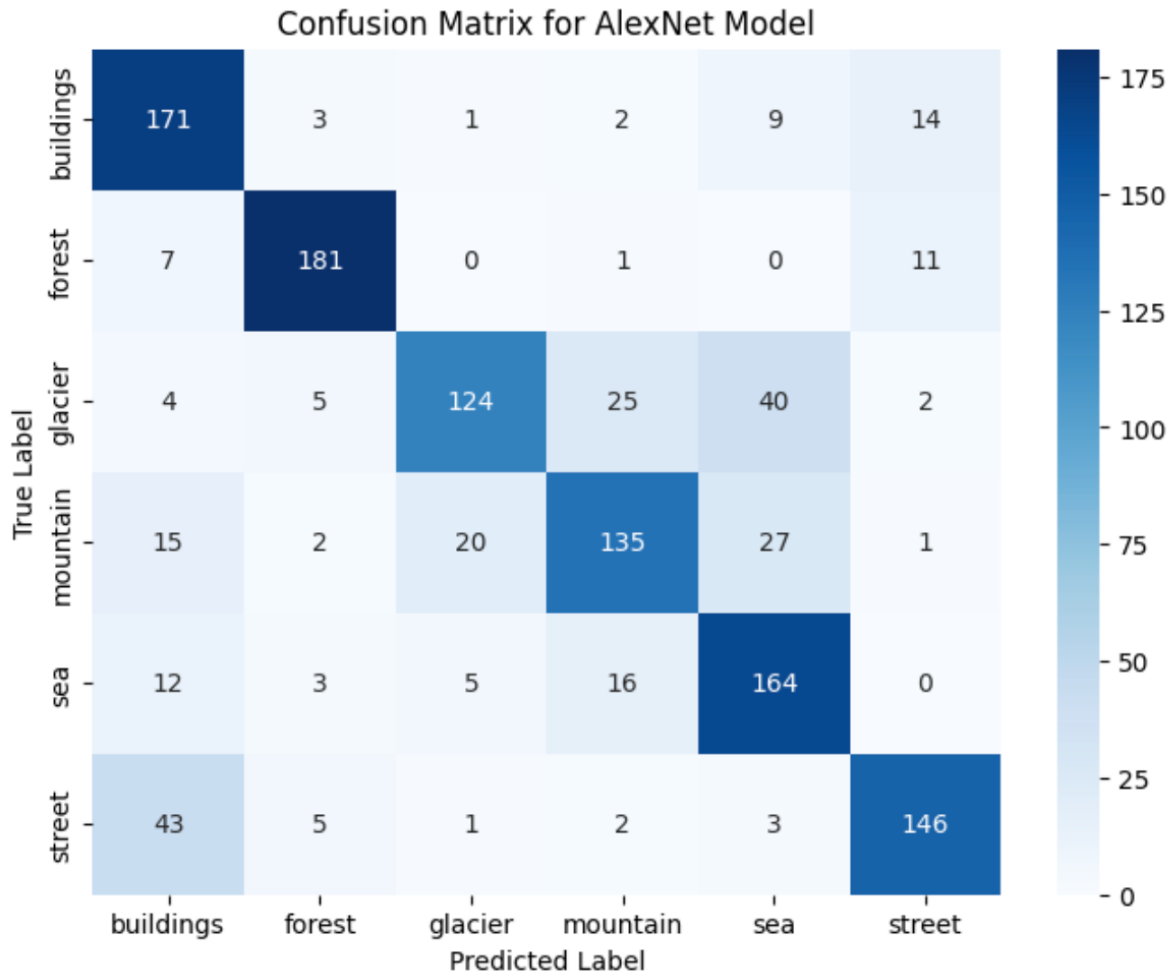
Random forest.ipynb

XGBoost.ipynb

Ensemble.ipynb

Confusion matrices and classifier reports

Confusion Matrix AlexNet



Classification Report AlexNet

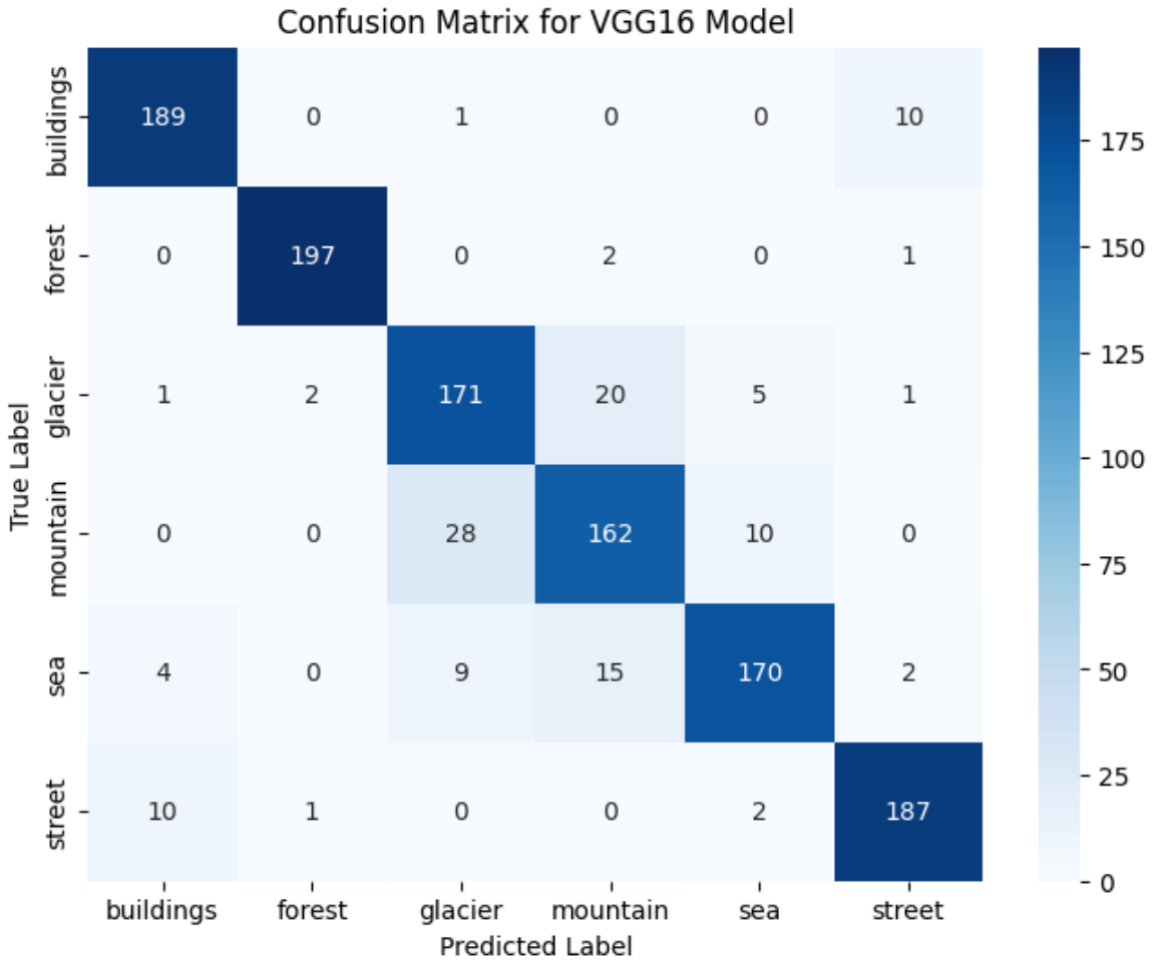
```
Classification Report:
              precision    recall  f1-score   support

   buildings       0.68      0.85      0.76       200
      forest       0.91      0.91      0.91       200
     glacier       0.82      0.62      0.71       200
    mountain       0.75      0.68      0.71       200
         sea       0.67      0.82      0.74       200
      street       0.84      0.73      0.78       200

    accuracy                           0.77      1200
   macro avg       0.78      0.77      0.77      1200
weighted avg       0.78      0.77      0.77      1200

Weighted F1 Score: 0.7667124076810939
```

Confusion Matrix VGG16



Classification Report VGG16:

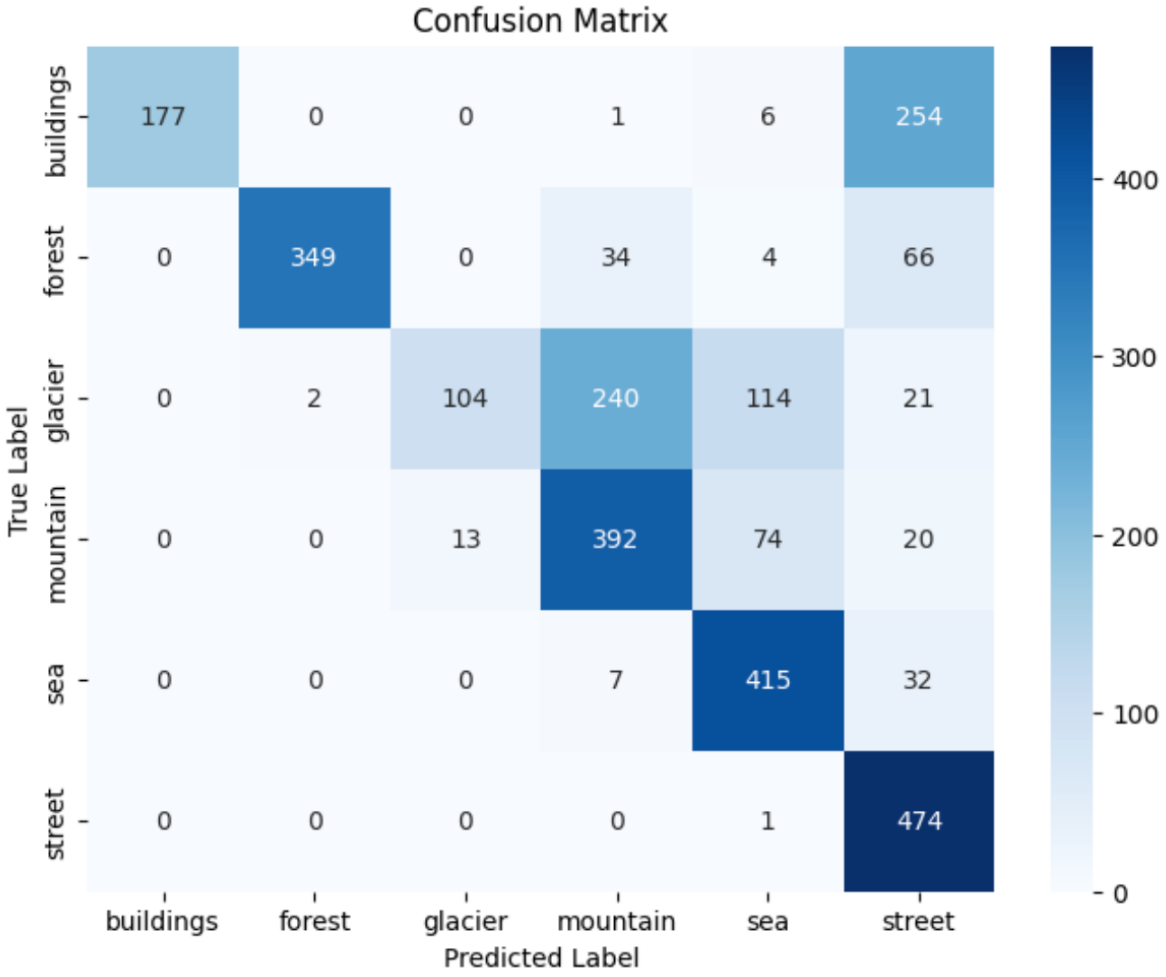
```
Classification Report:
              precision    recall  f1-score   support

   buildings       0.93      0.94      0.94       200
      forest       0.98      0.98      0.98       200
     glacier       0.82      0.85      0.84       200
    mountain       0.81      0.81      0.81       200
         sea       0.91      0.85      0.88       200
      street       0.93      0.94      0.93       200

    accuracy                           0.90      1200
   macro avg       0.90      0.90      0.90      1200
weighted avg       0.90      0.90      0.90      1200

Weighted F1 Score: 0.8966801265614324
```

Confusion Matrix MobileNetV2:



Classification Report MobileNetV2:

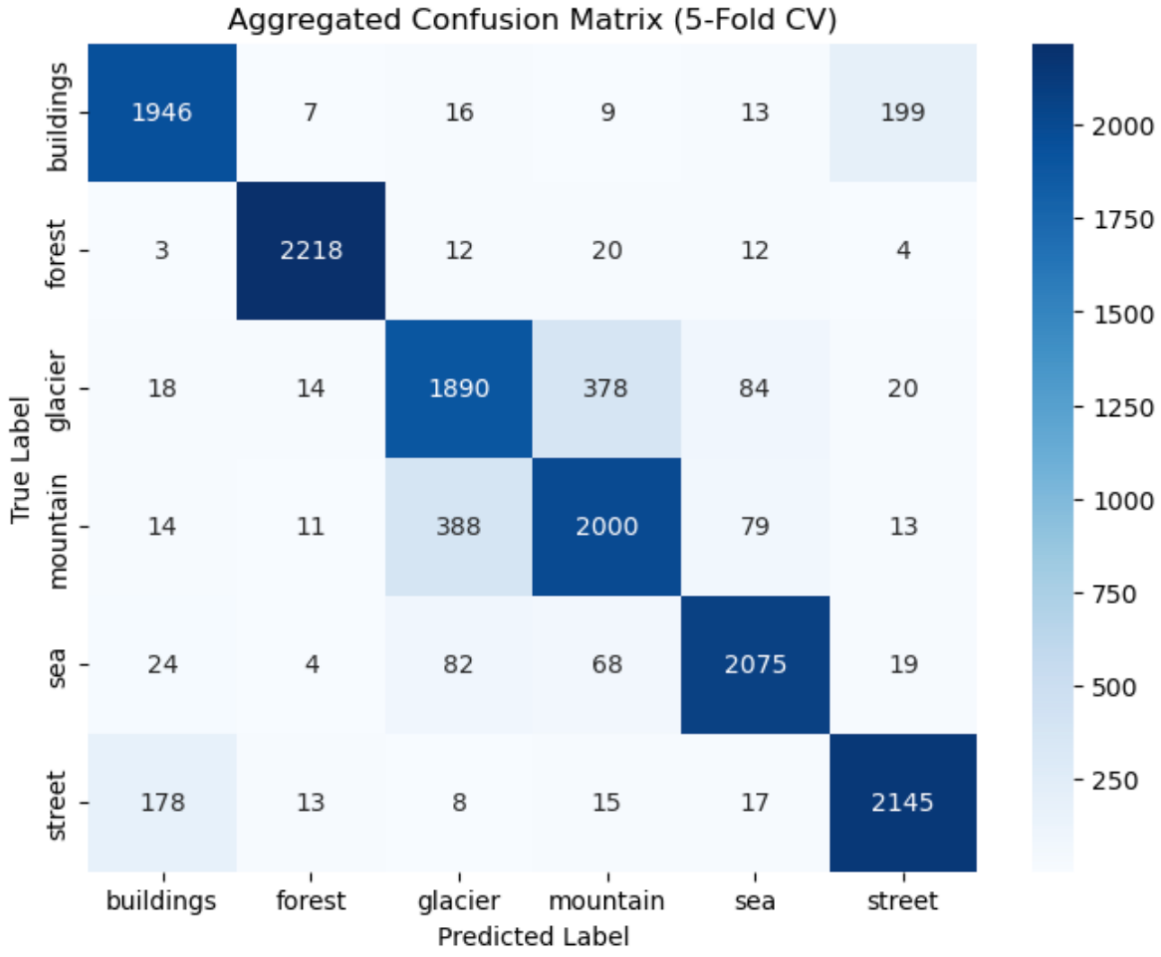
```
Classification Report:
              precision    recall  f1-score   support

   buildings       1.00      0.40      0.58       438
      forest       0.99      0.77      0.87       453
     glacier       0.89      0.22      0.35       481
    mountain       0.58      0.79      0.67       499
         sea       0.68      0.91      0.78       454
      street       0.55      1.00      0.71       475

    accuracy                           0.68      2800
   macro avg       0.78      0.68      0.66      2800
weighted avg       0.78      0.68      0.66      2800

Weighted F1 Score: 0.6552095344799777
```

Confusion Matrix Logistic Regression:



Classification Report Logistic Regression:

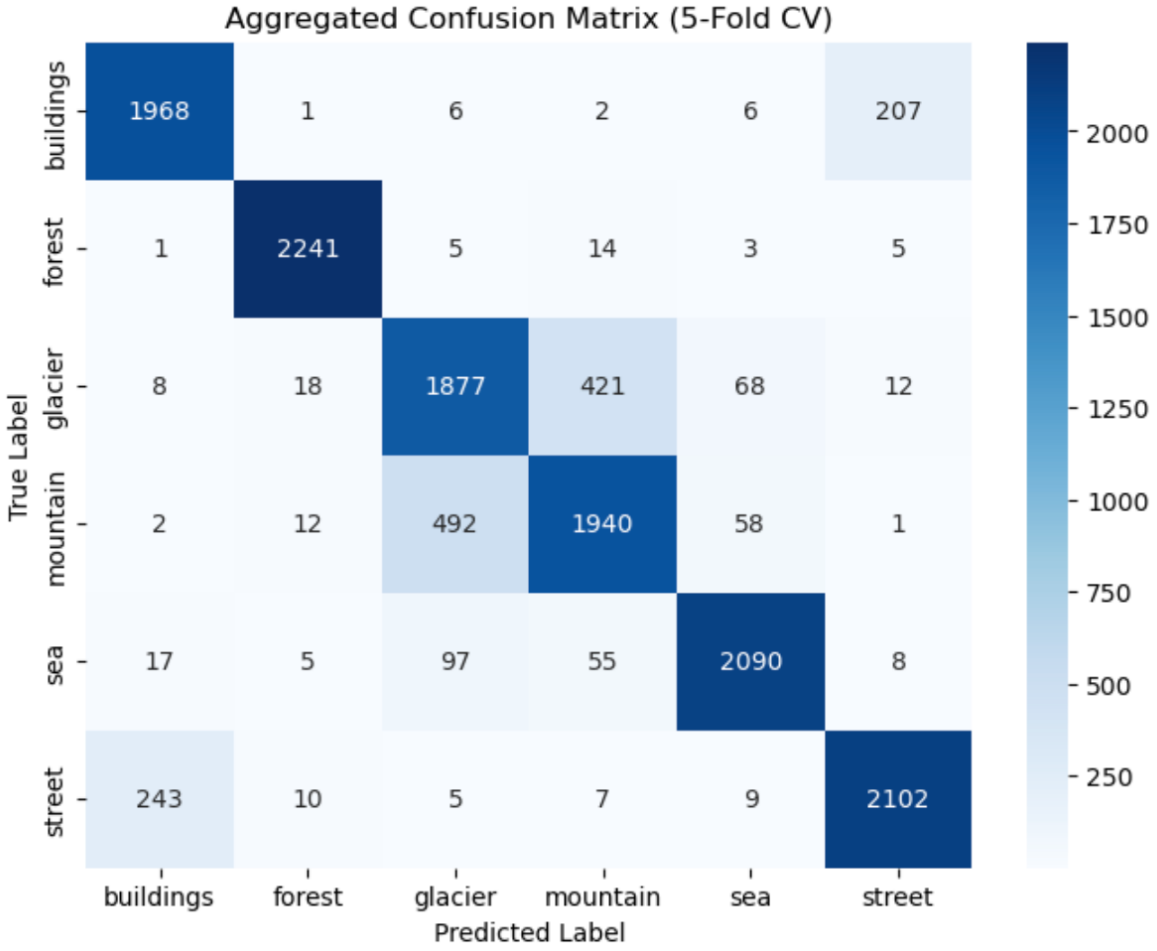
```
Aggregated Classification Report:
              precision    recall  f1-score   support

   buildings       0.89      0.89      0.89      2190
      forest       0.98      0.98      0.98      2269
     glacier       0.79      0.79      0.79      2404
    mountain       0.80      0.80      0.80      2505
         sea       0.91      0.91      0.91      2272
      street       0.89      0.90      0.90      2376

    accuracy                           0.88     14016
   macro avg       0.88      0.88      0.88     14016
weighted avg       0.88      0.88      0.88     14016

Aggregated Weighted F1 Score: 0.8756295191928755
```

Confusion Matrix SVM:



Classification Report SVM:

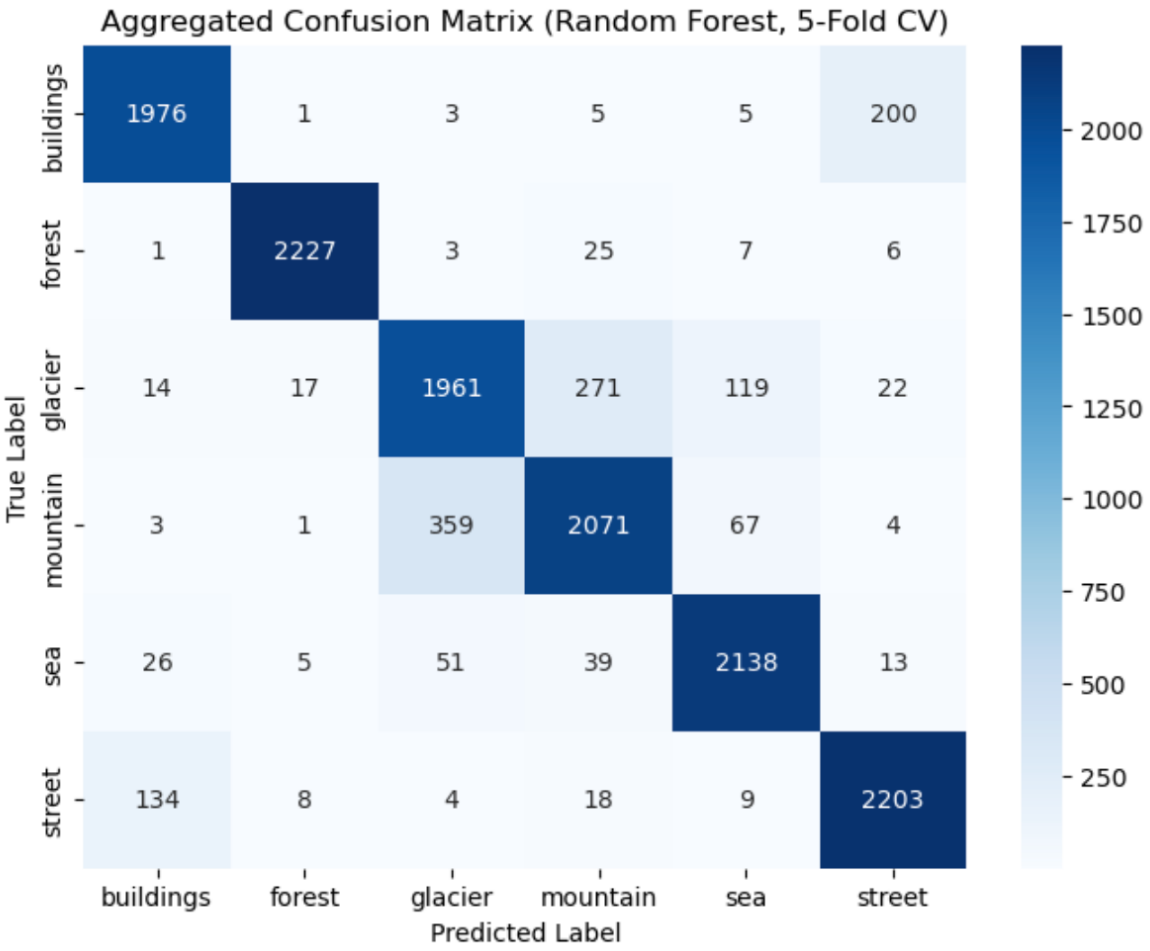
```
Aggregated Classification Report:
              precision    recall  f1-score   support

   buildings       0.88      0.90      0.89      2190
      forest       0.98      0.99      0.98      2269
     glacier       0.76      0.78      0.77      2404
    mountain       0.80      0.77      0.78      2505
         sea       0.94      0.92      0.93      2272
      street       0.90      0.88      0.89      2376

    accuracy                           0.87     14016
   macro avg       0.87      0.87      0.87     14016
weighted avg       0.87      0.87      0.87     14016

Aggregated Weighted F1 Score: 0.8718055193535389
```

Confusion Matrix Random Forest:



Classification Reports Random Forest:

Best fold from random forest:

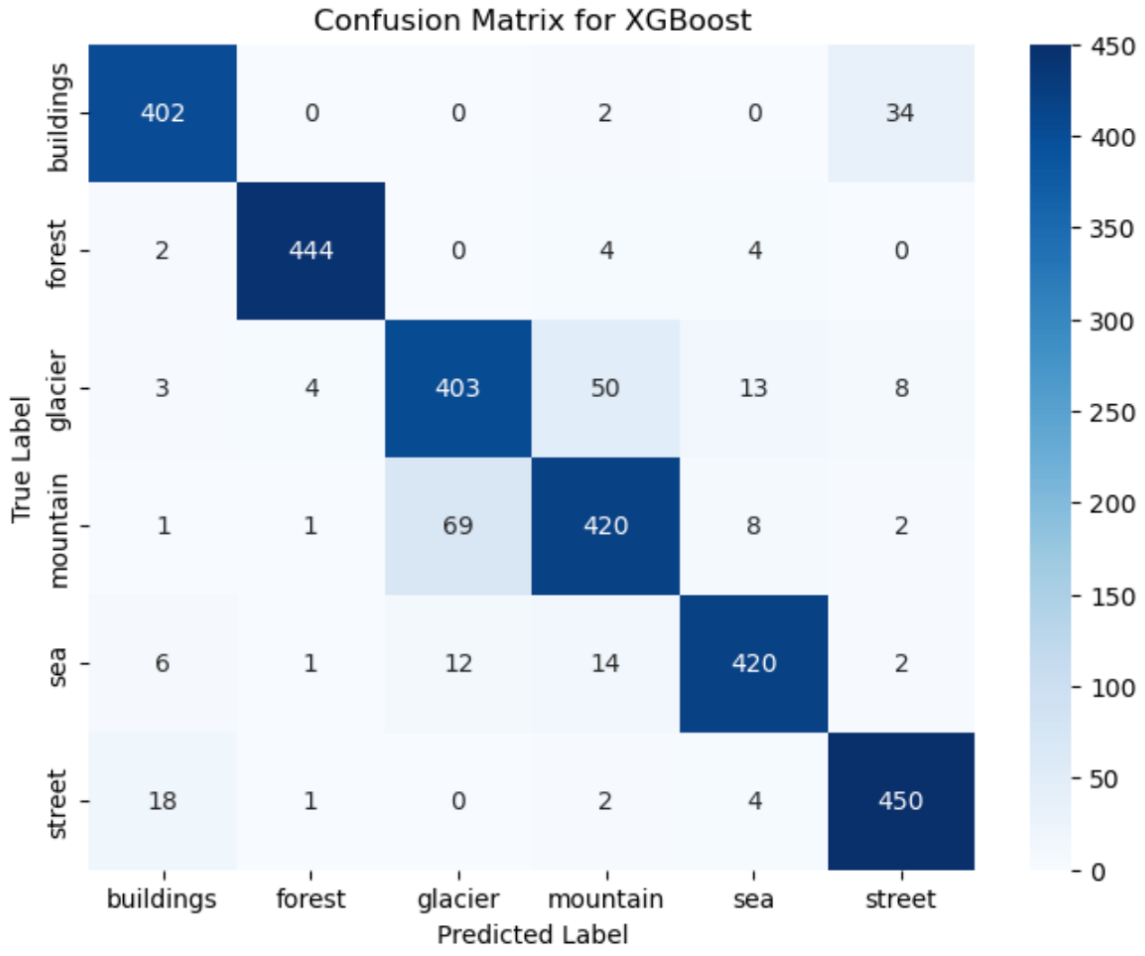
```
--- Fold 3 ---
Fitting 5 folds for each of 18 candidates, totalling 90 fits
Best parameters for fold 3 : {'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
Fold 3 Weighted F1 Score: 0.9056257192612409
Fold 3 Classification Report:
              precision    recall  f1-score   support

   buildings       0.90      0.91      0.91       438
      forest       0.99      0.99      0.99       454
     glacier       0.86      0.83      0.84       481
    mountain       0.85      0.86      0.85       501
         sea       0.93      0.94      0.93       454
      street       0.91      0.92      0.92       475

    accuracy                           0.91      2803
   macro avg       0.91      0.91      0.91      2803
weighted avg       0.91      0.91      0.91      2803
```

Average Weighted F1 Score across folds: 0.8966273681488419

Confusion Matrix XGBoost:



Classification Report XGBoost:

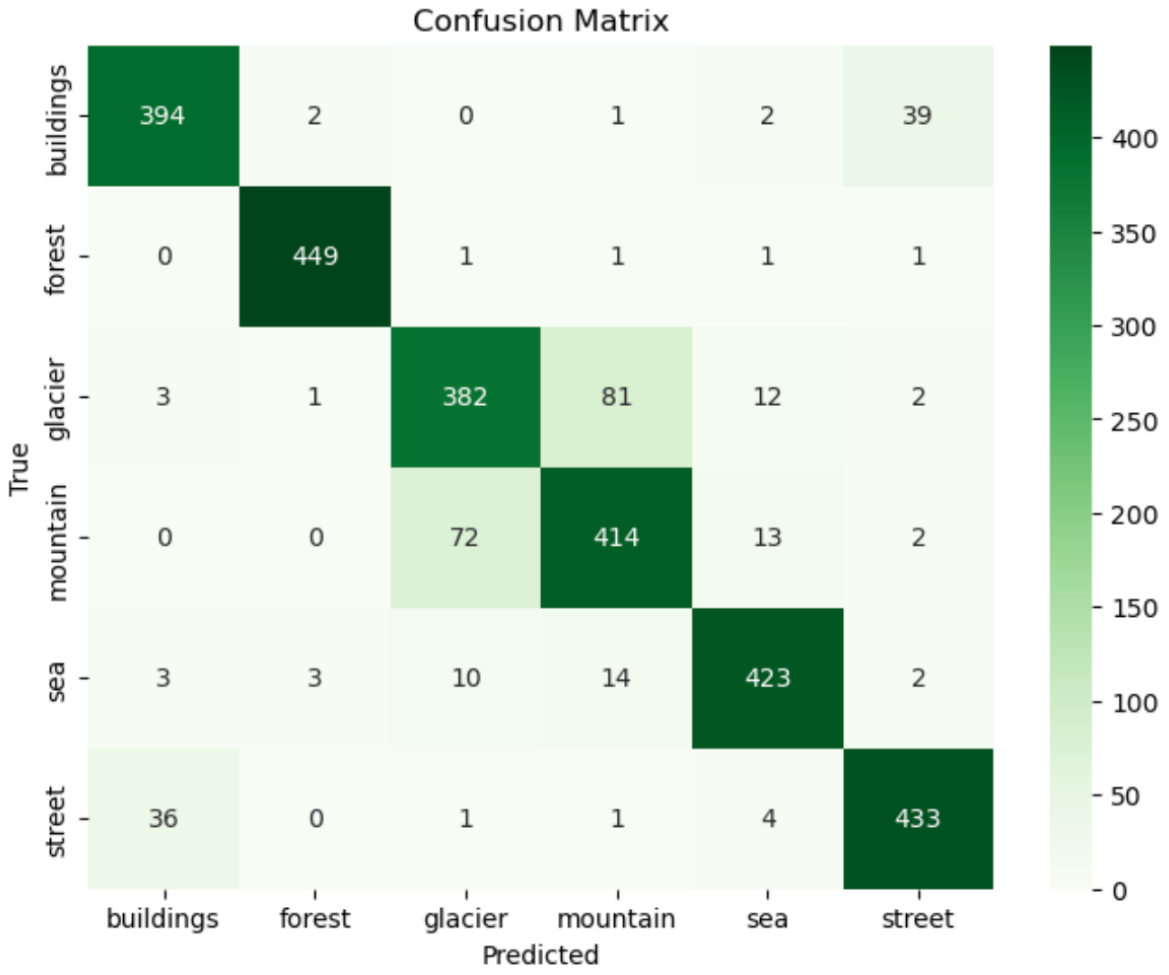
```
Classification Report for XGBoost:
              precision    recall  f1-score   support

   buildings       0.93      0.92      0.92       438
      forest       0.98      0.98      0.98       454
     glacier       0.83      0.84      0.84       481
    mountain       0.85      0.84      0.85       501
         sea       0.94      0.92      0.93       455
      street       0.91      0.95      0.93       475

    accuracy                           0.91      2804
   macro avg       0.91      0.91      0.91      2804
weighted avg       0.91      0.91      0.91      2804

Weighted F1 Score for XGBoost: 0.9054398750329047
```

Confusion Matrix Ensemble:



Classification Report Ensemble:

```
Classification Report:
              precision    recall  f1-score   support

   buildings       0.90      0.90      0.90       438
      forest       0.99      0.99      0.99       453
     glacier       0.82      0.79      0.81       481
    mountain       0.81      0.83      0.82       501
         sea       0.93      0.93      0.93       455
      street       0.90      0.91      0.91       475

    accuracy                           0.89      2803
   macro avg       0.89      0.89      0.89      2803
weighted avg       0.89      0.89      0.89      2803

Weighted F1 Score: 0.8899936603811862
```

Visualizations

EDA 1: Visual Inspection of Sample Images

As you can see, two example images from the training set and one from the test set for each class were displayed. This step allows for a qualitative assessment of the images, ensuring that the classes are visually distinct and that the dataset includes sufficient variability. Visual checks can quickly reveal any anomalies such as corrupt images or significant quality issues.

Sample Images for Forest

Forest - Train

Forest - Train

Forest - Test

Sample Images for Glacier

Glacier - Train

Glacier - Train

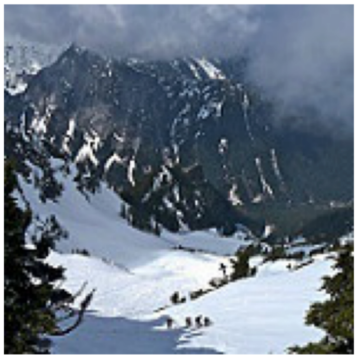
Glacier - Test

Sample Images for Mountain

Mountain - Train

Mountain - Train

Mountain - Test

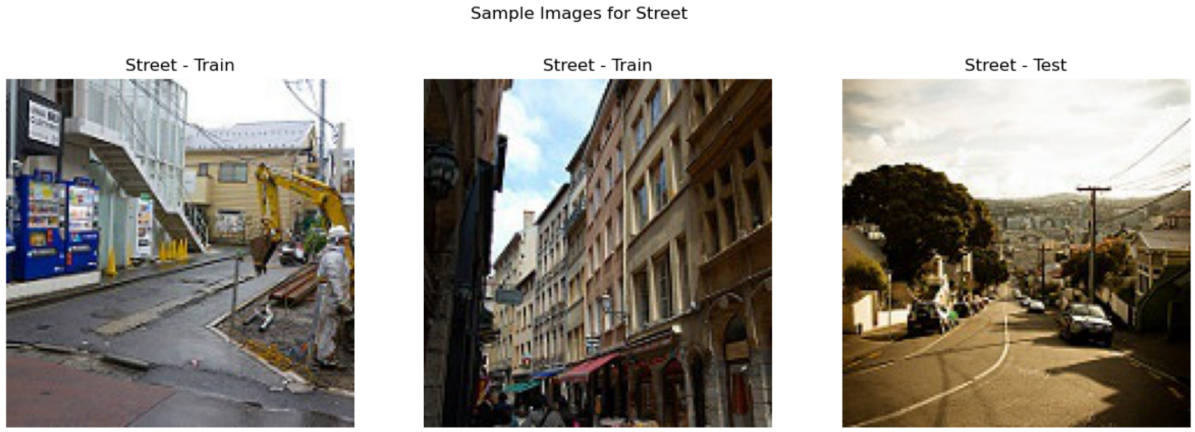


Sample Images for Sea

Sea - Train

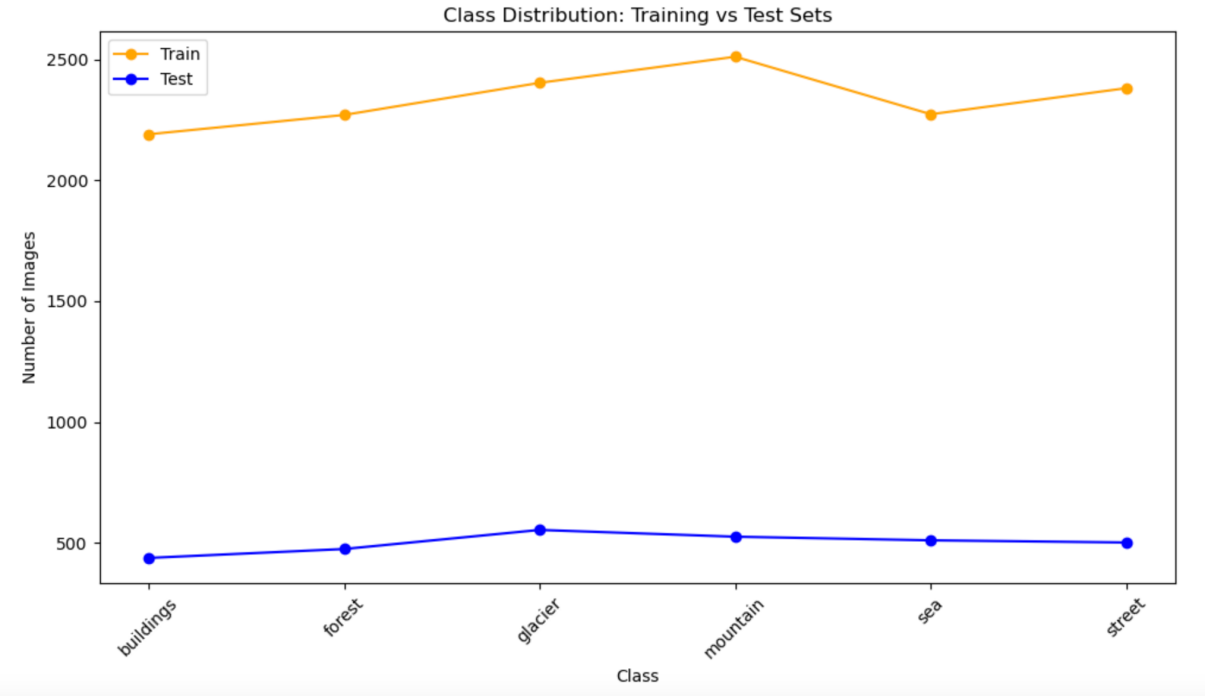
Sea - Train

Sea - Test



EDA 2: Class Distribution Analysis

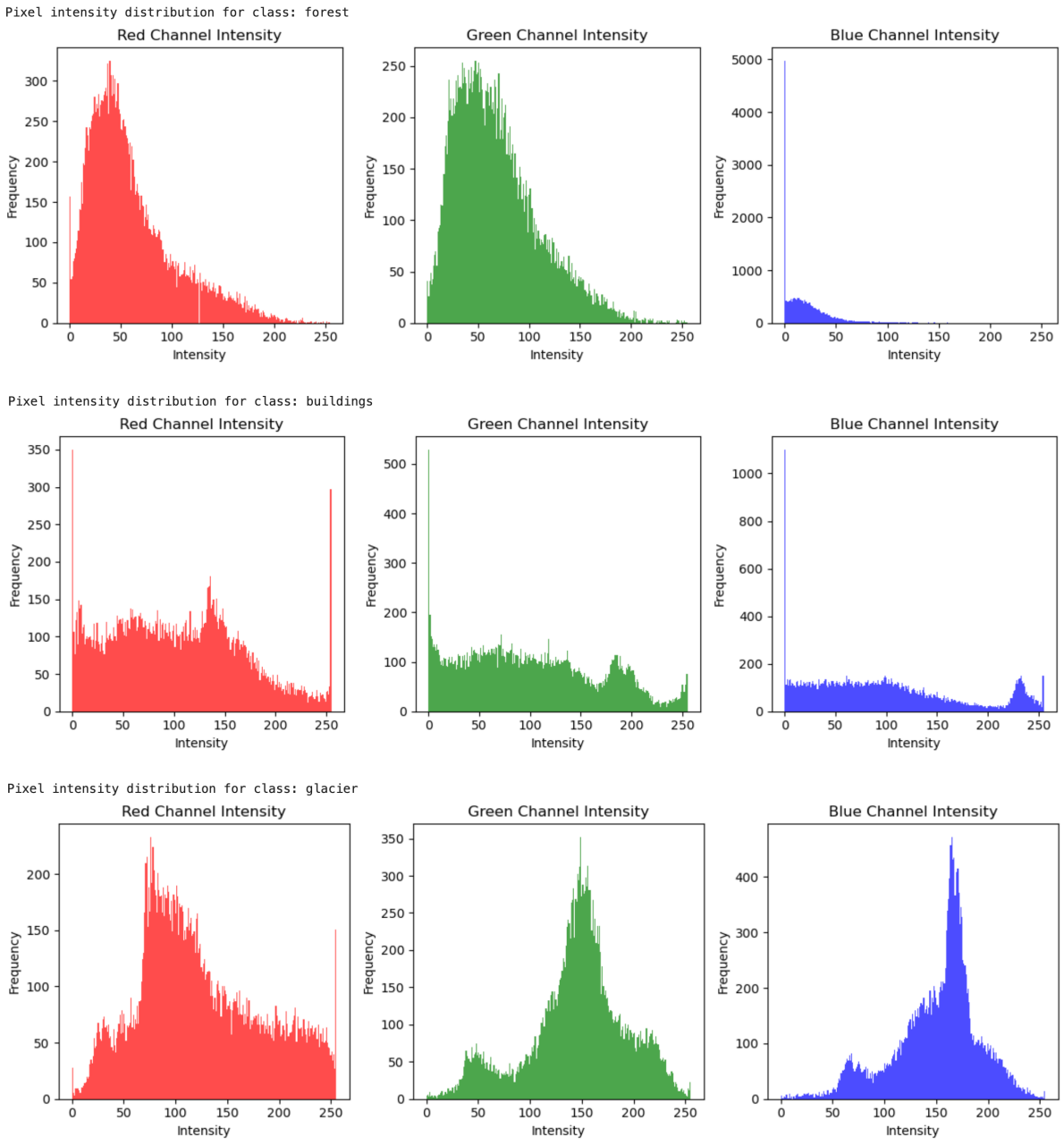
EDA 2 shows us a line graph was generated to display the number of images per class in both the training and test datasets. Assessing class distribution is critical to identify potential imbalances that could lead to biased models. Even slight variations, such as the predominance of forest images, must be acknowledged to understand how they might affect the training process.

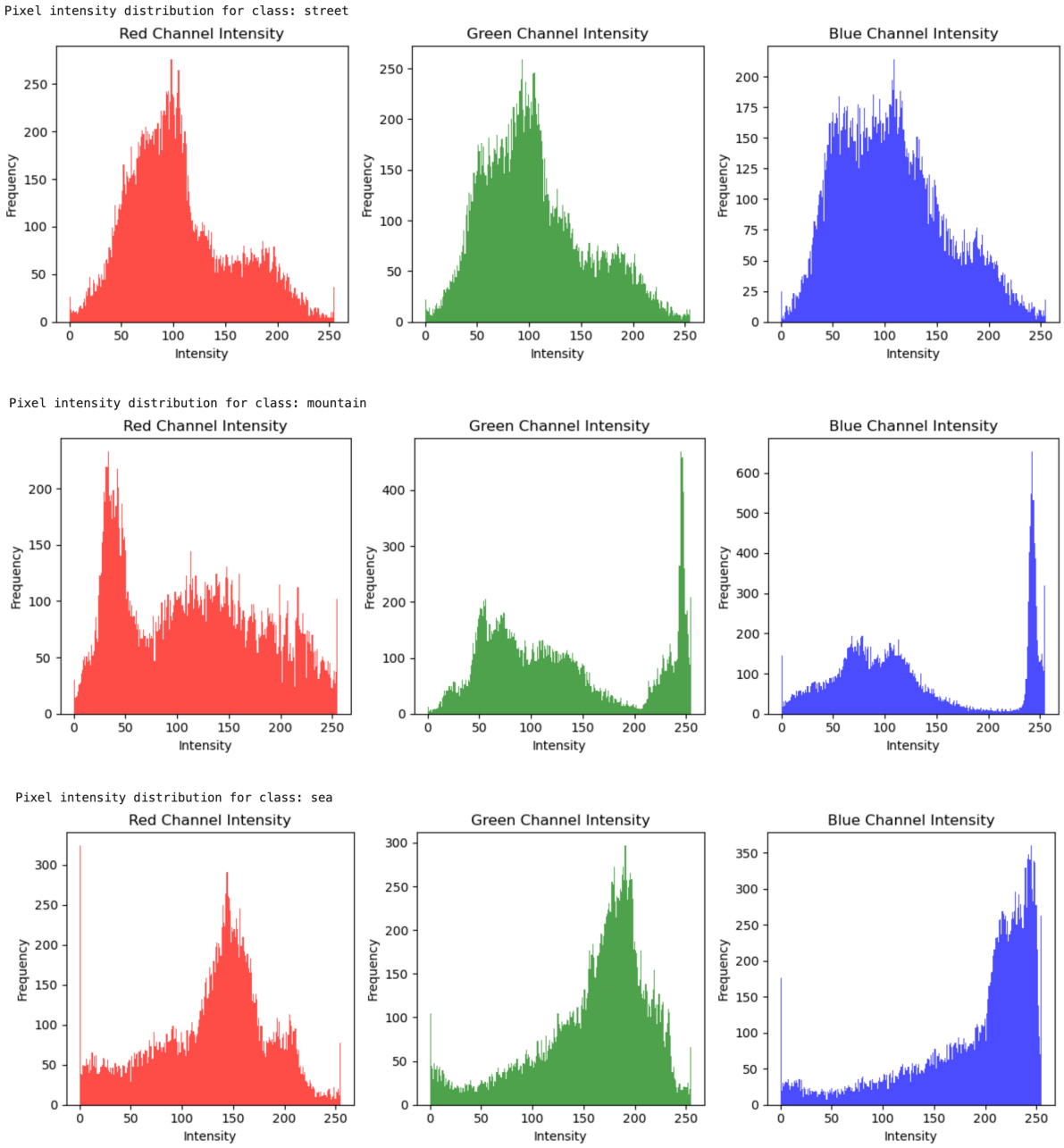


EDA 3: Pixel Intensity Distribution Examination

Graphs in EDA 3 show the distribution of pixel intensities for each RGB channel were plotted using sample images.

Evaluating the pixel intensity distributions helps verify the quality and consistency of the images. It provides insight into potential issues like overexposure or underexposure that could distort feature learning. Such distributions are pivotal for ensuring that the images do not introduce systematic bias into the feature extraction process.

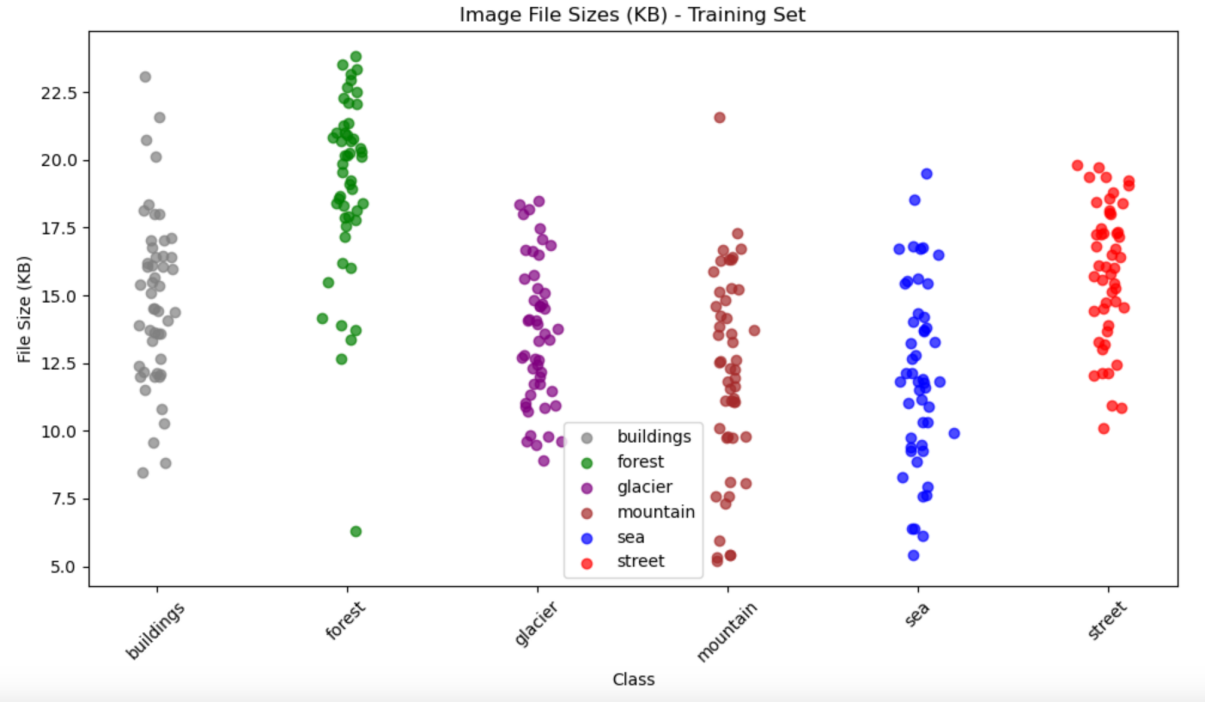


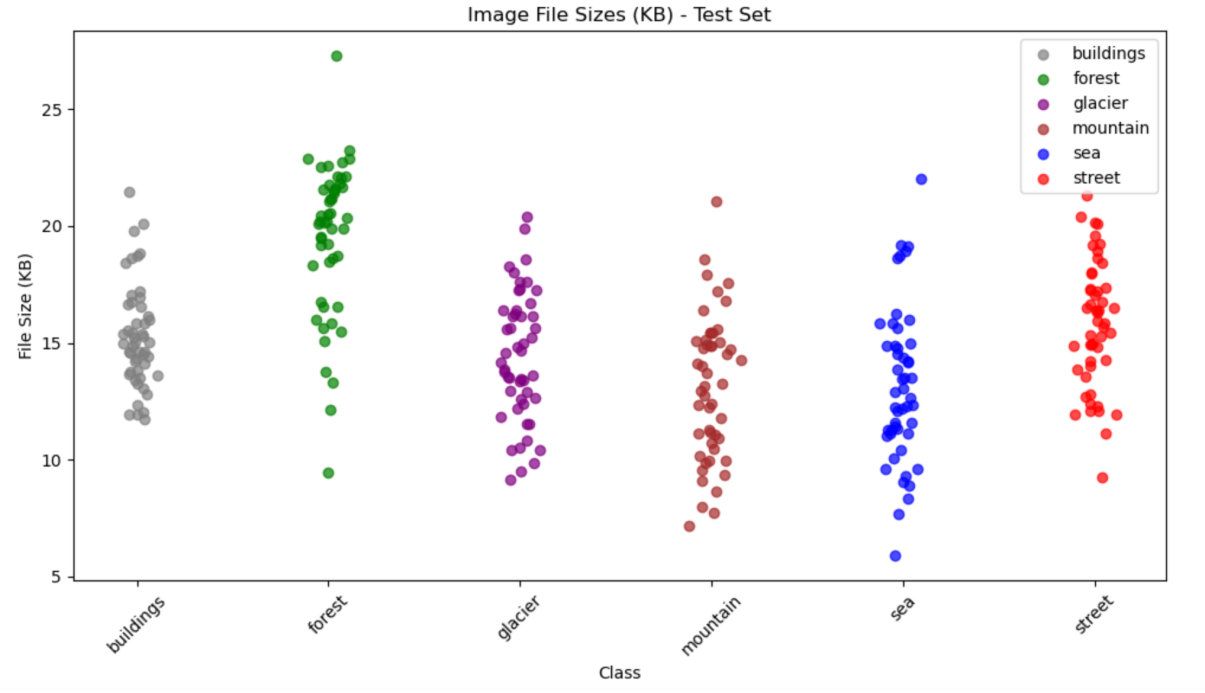


EDA 4: File Size Analysis via Scatterplots

In EDA 4, two scatterplots were created to illustrate the file size variations by class in the training and test datasets.

Although the dataset shows slight variations in file size across classes, with forest images tending to have larger file sizes, these differences were not significant enough to be considered outliers. Monitoring file size ensures that no class is disproportionately represented by unusually high or low-quality images.





EDA 5: Metadata Inspection

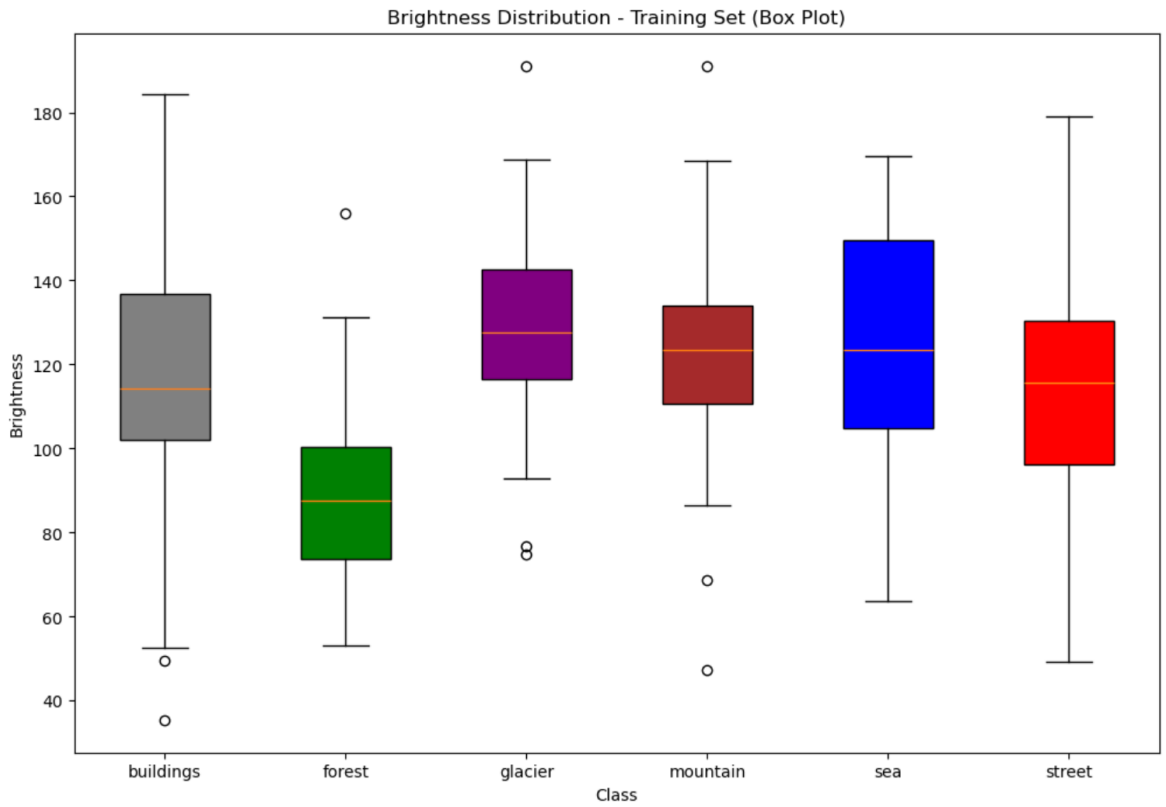
A scan of 100 random images was conducted to check for any embedded metadata. The results repeatedly showed no metadata, the absence indicates a standardized image format across the dataset, reducing the risk of hidden biases or additional preprocessing requirements.

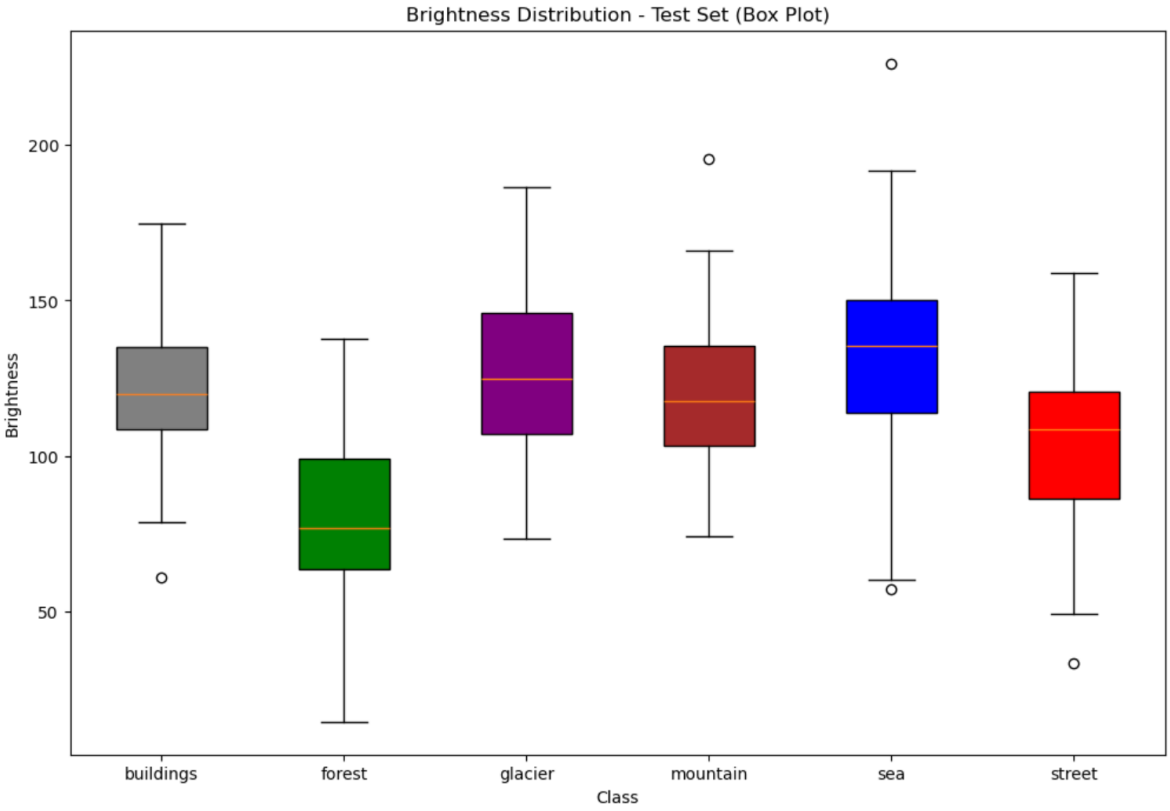
```
There is no metadata found in any of the sampled images.
```

EDA 6: Brightness Distribution Analysis Using Box Plots

Two box plots were generated to examine the brightness distribution of images across different classes in both the training and test sets as shown in EDA 6

Brightness distribution analysis is crucial for detecting inconsistencies in lighting conditions. For example, forest images generally exhibited a lower brightness distribution, likely due to tree shadows, while buildings and street images showed a wider brightness range in the training set compared to the test set. Such findings are essential for adjusting preprocessing steps or model expectations regarding illumination variances.




Brightness Distribution - Test Set (Box Plot)
Brightness
200
150
100
50
buildings
forest
glacier
mountain
sea
street
Class