CS5014 P2: Classification of Seal Images

220029176

University of St Andrews

March 2023

Word Count: 5251

Introduction

This report will critically evaluate the development of a classification model that can predict and categorise satellite images of seal environments. The model is created from a set of real experimental data which comprises two datasets: binary and multi-class. Within this project, several steps were taken to load and clean the data (Section 1A and 1B), analyse and visualise it (Section 1C) and, choose a suitable subset of features for the model (Section 1D). Suitable candidates for this classification problem were chosen (Section 2), trained and evaluated (Section 3). This document will explain each stage in detail, demonstrating an understanding of the consequences of each decision on the model's performance, and provide evidence showing how altering the decisions adjusts the model’s accuracy. The report will also contain a critical discussion of the results and ideas for further research (Section 4).

Part 1: Data Processing

A The Data

Understanding the data is a critical step in building an effective machine learning model. By gaining a deep understanding of the data, we can identify which type of model is most appropriate for the classification problem, and which pre-processing steps may be necessary. This could include steps such as removing missing values, scaling, normalizing features, or encoding categorical variables. This helps to avoid overfitting and ensure that the model accurately represents the underlying relationships in the data. Additionally, understanding the data is important in evaluating the performance of the model, by analysing metrics such as accuracy, precision, recall, and F1 score, and comparing the performance of different models.

The data provided consists of 964 columns, with the first 900 columns corresponding to a Histogram of Oriented Gradients (HoG) which are extracted from an image. HoG is a technique used to simplify the representation of an image by capturing important information. This involves computing the gradient and magnitude of each pixel within a 10x10 pixel cell and then proportioning the magnitudes into a histogram based on their orientation. These histograms are normalized within blocks of 20x20 pixels or 2x2 cells, the final product of HoG can be seen in Figure 1.

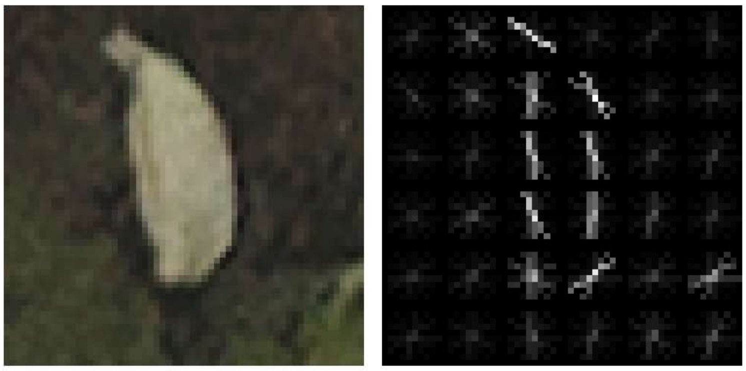


Figure 1: A whitecoat image to a Histogram of Oriented (HoG) Image based on 10x10px cells, 9 orientations and 2x2 Blocks.

Equation 1 combines these normalized histograms to generate a single feature vector, which represents the total number of features generated by a single image using the HoG technique.

$$B_{Row}B_{Column}N_{Orientations}N_{Cells} = N_{features}$$

Equation 1: Derivation of the Histogram of Oriented Gradients Count

B_{Row} = Number of blocks per cell row

B_{Col} = Number of blocks per cell column

$N_{Orientation}$ = Number of orientations

N_{Cells}= Number cells per block

$N_{features}$= Number of features

The subsequent 16 columns, from 900 to 916, are generated randomly from a normal distribution with a mean of 0.5 and a standard deviation of 2. Finally, the last 48 columns correspond to three colour histograms extracted from the same image for each of the three colour channels (red, green, blue). Each colour channel histogram is divided into 16 bins for values between 0 and 255. As these values are not proportioned between the relative bins, they must be whole numbers and nonnegative values. Finally, column names were produced based on all the sources of each feature.

B Loading and Cleaning the Data

The data sets were loaded into the Python script using the Pandas module functionality. The data sets provided were in comma-separated value (CSV) format. The datasets were read into two data frames to be used for further pre-processing.

Firstly, the data sets were checked and cleaned. This is an important step in machine learning models because the quality and completeness of the data used to train a model directly affects its accuracy and reliability. If the data contains null values, duplicate rows, or incorrect values, it can cause biases and errors in the model, which can lead to incorrect predictions and poor performance. By checking for null values, duplicate rows, and negative or invalid values, we can ensure that the data is clean and ready for use in a machine learning model. Additionally, by removing or correcting any problematic data, we can improve the quality of the training data and ultimately improve the performance of the model.

No null values or duplicate rows were detected when cleaning the data. Based on the knowledge gained by assessing the sources of the data within the dataset, the last 48 columns were checked for any negative values as these would be invalid values. Similarly, the first 900 columns were checked for negative values as well as any value greater than one. No invalid values were found, indicating the provided dataset was valid and ready for analysing and visualisation.

C Analysing and Visualising the Data

So far, we have analysed the sources of the data without looking at the data itself. Analysing and visualising the data helps us understand the characteristics and properties of the data. This includes understanding the distribution of the data, identifying outliers, and identifying any patterns or relationships between variables. Furthermore, analysing and visualising the data can help us identify which features are most important for the model. This can be done by identifying correlations between features and the target variable, or by identifying features that have a strong impact on the model's performance.

The binary data is highly skewed towards background images with a ratio of background to seal of 54431:7778 for the training data, see Figure 2. This raw data may cause a class imbalance issue where the algorithm is more biased towards the majority class. Although this ratio is likely due to the nature of the data and the proportion of background to seals in the wild, the class weights will need adjusting before any training can commence. Similarly, the multi-class occurrences in Figure 2 indicate a large proportion of images will contain background environments. Additionally, the populations of each seal classification in the multi-class data does not accurately represent the actual populations seen in nature (University of St Andrews, 2022). Within this report, it is summarised that juvenile seal numbers are greater than that of pups. Therefore, to accurately represent the populations seen in nature, the class weights will be manipulated within the training model and the results evaluated.

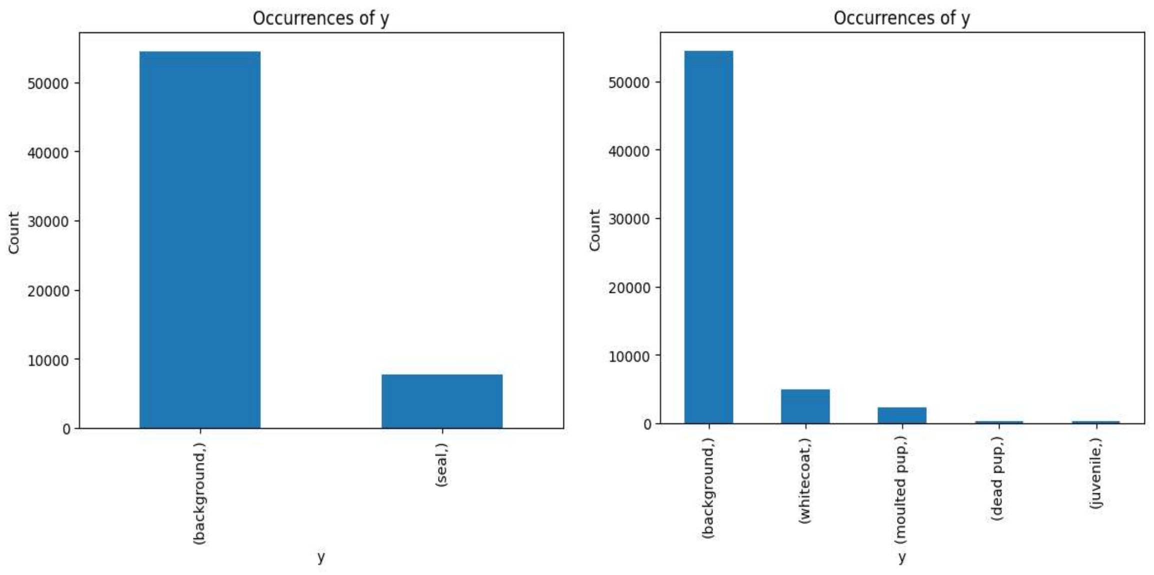


Figure 2: The Number of Each Class for Binary Data (left) and Multi-Class Data (right)

Analysing the 900 HoG features through visualisation may not be appropriate to determine the relationships between the different classifications. While it is possible to visualize the HoG features to get an understanding of the local gradient directions and magnitudes, analysing all 900 HoG features may not be appropriate as they are high-dimensional, and therefore it can be difficult to gain insights by visualizing them directly. Instead, dimensionality reduction techniques such as Principal Component Analysis (PCA) can be used to reduce the dimensionality of the HoG features and visualize them in a lower-dimensional space. These techniques can help to identify patterns or relationships between the classifications that may not be apparent when visualizing the high-dimensional HoG features directly.

The colour histogram values were analysed for both the binary and multi-class datasets. 3D scatter graphs were produced to visualize the relationship between the background and the seals, as well as the relationships between the individual seal classifications within the multi-class system. These graphs can be found in Appendix 1 and 2 for binary and multi-classification, respectively. From the binary graphs, the background and seal images are highly similar in the 47-143 range. However, there are clear patterns that indicate that these classifications can be distinguished. For instance, the 238.5-255 bin shows a clear correlation between the RGB colours for the background data, while the seal data does not show the same relationship. These and other distinctions may be essential to the classification problem. Similarly, the multi-class data in Appendix 2 suggests that distinguishing between the four classifications is challenging and may require using all the provided colour features to differentiate between the seal classes.

D Feature Extraction and Standardisation

Feature extraction is a crucial step in machine learning models as it helps to identify which features are relevant to the classification problem, and which features may need to be transformed or removed entirely. Now that we have a deeper understanding of the data, we can strategically decide which features to maintain and which to remove. Feature selection helps to avoid overfitting and ensures that the model accurately represents the underlying relationships in the data.

From the research conducted so far, the multi-dimensional HoG features seen within the first 900 features of the data are important points of data for the classifier and cannot be dropped from the dataset. However, the sheer number of features will need to be adjusted so the model can be built and trained within a reasonable timeframe. Therefore, the dimensionality reduction method PCA was chosen to reduce the number of HoG features within the model whilst maintaining as much critical information as possible. Principal Component Analysis (PCA) finds a new set of dimensions such that all the dimensions are orthogonal and ranked according to the variance of data along them, where variance is the distribution of the data, see Equation 2.

$$var(x) = \frac{\sum(x_i - \bar{x})^2}{N}$$

Equation 2: Variance Equation

Where,

x_i = The value of x in the i -th dimension

$\bar{x}$ = The mean value of x

N = The total number of variables

First, a covariance matrix is produced from the data points which are then used to calculate the eigen vectors and corresponding values, see Equations 3 and 4 for covariance and eigen vectors, respectively. These are then sorted according to the values in descending order. The first K eigen vectors are chosen, where K is the user input. The original matrix is transformed into a k dimensional matrix.

$$cov(x, y) = \frac{\sum(x_i - \bar{x})(y_i = \bar{y})}{N}$$

Equation 3: Co-variance Equation

Where,

x_i = The value of x in the i -th dimension

$\bar{x}$ = The mean score of x

y_i = The value of y in the i -th dimension

$\bar{y}$ = The mean score of y

$$Av = \lambda v$$

Equation 4: Eigen Vector Equation

Where,

A = a square matrix

v = vector

λ = a scalar

From Figure 3 we can see that the variance within the features plateaus at around the 400 features point. This indicates that the 400 features for the model is a reasonable threshold to consider when reducing the dimensionality of the dataset.

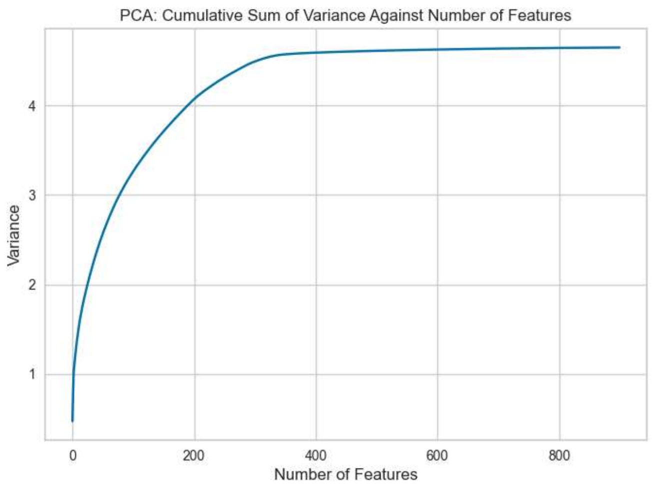


Figure 3: Cumulative Sum of Variance Against Number of Features

The relationships between the colour histograms mentioned in Part 1C suggest that all the colour histogram features are necessary for the final model. The requirement for normalization depends on the classification model used, as further described in Part 2B, where the random forest and support vector machine (SVM) algorithms were selected for this classification task. For the random forest algorithm, no normalization is required since each node in the forest is not comparing feature values. Normalization can even smooth out the non-linear nature of the model, causing these nonlinearities to not be reflected when y is back transformed. However, normalization is crucial for SVM as it assumes that the data is within a standard range. Therefore, when testing and evaluating the models, the features will be tailored to the needs of the algorithm to ensure optimal performance.

The normal distribution features appeared to have little to no relevance to the classification when researching the source of the provided data. Therefore, these should be removed to reduce the complexity and time taken to train the classification model.

Part 2: Modelling and Experimentation

The following models were built using a 75:25 split. A 75:25 train test split is often considered a good approach for large datasets because it allows for a large enough sample size in both the training and testing sets, while still ensuring that the model has sufficient training data to learn from (Baheti, 2023). The random state of 1 was chosen to keep the results consistent throughout the evaluation of the models and hyperparameters.

A Choosing the Models

Choosing the right model for the data is critical in achieving accurate classification. There are numerous models to choose from, each with its strengths and weaknesses, and selecting the appropriate one is crucial to ensuring the optimal performance of the algorithm. When dealing with a classification problem containing HoG features and colour histogram features, several models are particularly well-suited to the task. Support Vector Machines (SVMs) are a common choice for classification problems and have proven to be effective with HoG features (Bristow & Lucey, 2014). Additionally, Random Forests have shown to be well-suited to classification problems that involve multiple features, such as colour histograms (Gurung, 2020).

The Random Forest (RF) algorithm is a supervised classification algorithm that is commonly used for classification and regression problems. It is an ensemble learning method that combines multiple decision trees to create a more robust and accurate model. In a random forest, each decision tree is constructed using a random subset of the training data and a random subset of the features. This randomness helps to reduce overfitting and improve the model's ability to generalize to new data. During the training process, the random forest algorithm generates many decision trees and each of them provides a prediction. The final prediction of the algorithm is the majority vote of all the decision trees. Random forests are particularly well-suited for image classification tasks because they can handle many features. Random forests can also detect complex nonlinear relationships between features and image labels, which is important when the relationship between the features and labels is not straightforward such as in HoG features.

Support vector machines (SVMs) are a type of supervised machine learning algorithm that can be used for classification, regression, and outlier detection. SVMs are particularly well-suited for image classification problems because they can effectively handle high-dimensional data, such as the large number of features often present in images. SVMs can separate classes in a way that maximizes the margin between the decision boundary and the closest data points of each class. This margin maximization results in a more robust and accurate model that is less prone to overfitting. Additionally, SVMs can use a variety of kernel functions, such as radial basis functions, that allow them to model nonlinear relationships between the features and the target variable.

B Hyperparameter Optimisation

Hyperparameters are the parameters of a machine learning model that are not learned from the data, but rather set by the user before training the model. The choice of hyperparameters can significantly impact the performance of a model, and different choices of hyperparameters may lead to different levels of accuracy or generalization ability. We experiment with hyperparameters to find the optimal set of hyperparameters that will maximize the performance of a model on unseen data and avoid overfitting. Hyperparameter tuning is a crucial step in machine learning model development and can significantly impact the performance of the model.

Random Forest Classifier

There are many hyperparameters that can be implemented and evaluated in a decision tree classification model. While some hyperparameters, such as the number of trees and maximum depth, are important to consider, they may not have a significant impact on the model's performance (Terzic, 2023). However, certain hyperparameters are fundamental to decision trees and are critical for achieving optimal performance. One of the most important hyperparameters is the criteria used to split a branch. In scikit, there are three ways to measure the quality of the split: Gini Impurity, Entropy, and Log Loss. The choice of splitting criteria can significantly impact the accuracy and efficiency of the model.

Criterion

Gini impurity measures the probability of misclassification for a randomly chosen element in the dataset. It ranges from 0 to 1, with 0 indicating that all the elements belong to the same class and 1 indicating that the classes are equally distributed. The formula for calculating Gini impurity can be seen in Equation 5.

$$G = 1 - \sum_{c} P(c)^2$$

Equation 5: Gini Impurity Equation

Where,

$P(c)$ = Probability of a sample belonging to class c

Entropy measures the amount of disorder or randomness in the dataset. It ranges from 0 to 1, with 0 indicating that all the elements belong to the same class and 1 indicating that the classes are equally distributed. The formula for calculating entropy can be seen in Equation 6.

$$E = -\sum_{c} P(c)\log_2 P(c)$$

Equation 6: Entropy Equation

Where,

$P(c)$ = Probability of a sample belonging to class c

Log loss is another metric used to evaluate the quality of splits in decision trees, particularly in cases where the target variable is a probability distribution. Log loss measures the difference between the predicted probability distribution and the true probability distribution for a set of samples, see Equation 7.

$$L = -\frac{1}{N}\sum y_i log(c) + 1 - y_i \log(1 - c)$$

Equation 7: Binary Cross-Entropy / Log Loss Equation

Where,

y_i = Actual Output

c = Probability prediction

N = Number of observations

Support Vector Machines

SVMs also have several hyperparameters that can be adjusted to optimize model performance. One of the most important hyperparameters in SVM is the kernel function, which is used to transform the input data into a higher-dimensional space where it can be more easily separated. Common kernel functions include linear, polynomial, and radial basis function (RBF). However, due to the multi-dimensional nature of the data, only RBF and polynomial will be tested as the linear kernel is less effective against multi-dimensional features since it can only create linear decision boundaries.

To further refine these kernels, additional hyperparameters can be adjusted. For the polynomial kernel, the degree of the polynomial function can be increased to capture more complex nonlinear relationships in the data. Additionally, the coefficient of the polynomial can be modified to control the relative weight given to each polynomial term in the kernel function. For the RBF kernel, there are several hyperparameters that can be adjusted, including C and gamma. C determines the trade-off between the number of misclassified training examples and the simplicity of the decision surface.

Lower values of C lead to smoother decision surfaces, while higher values aim to classify all training examples correctly. On the other hand, gamma determines the influence of a single training example on the decision boundary. Higher values of gamma result in a decision boundary that is more sensitive to nearby training examples.

Kernal

The polynomial kernel works by transforming the input data into a higher-dimensional space using a polynomial function. It is useful when the data is not linearly separable in the original feature space but can be separated by a curved boundary in the higher-dimensional space, see Equation 8.

$$K(x, x') = (1 + \langle x, x' \rangle)^d$$

Equation 8: Polynomial Kernel

Where,

x = Input vector

x' = Input vector

d = Degree of the polynomial kernel

The RBF kernel works by transforming the input data into an infinite-dimensional space using a Gaussian function. This is useful when the data is not linearly separable and when the decision boundary needs to be nonlinear.

$$K(x, x') = \exp\left(-y\|x - x'\|^2\right)$$

Equation 8: Radial Basis Function Kernel

Where,

x = Input vector

x' = Input vector

y = Gamma value

Part 3: Model Evaluation and Comparison

Evaluating and comparing different classification models is crucial to ensure that we select the best model for our specific problem and dataset. This involves assessing the model's ability to accurately classify new data and generalise well to unseen examples. To do this, we use various evaluation metrics that provide insight into the model's performance. One of the most common metrics is accuracy_score(), which measures the proportion of correctly classified examples out of the total number of examples. While accuracy is an important metric, it can be misleading in imbalanced datasets where one class is much more frequent than the others. In such cases, we can use the balanced_accuracy_score() metric, which takes into account the distribution of examples across all classes. These metrics are used for a basic analysis of the model. Both metrics were only performed on the multi-classification model to gauge a basic understanding of the model’s accuracy for classification.

Two further metrics are used for a deep analysis of the multi and binary classification models. Firstly, the confusion_matrix() can be implemented to provide a detailed breakdown of the model's performance by showing the number of true positives, false positives, true negatives, and false negatives. This allows us to identify which classes are most frequently misclassified and gain insights into how the model is making errors. Finally, the classification_report() provides a summary of several important evaluation metrics, including precision, recall, f1-score, and support for each class. These metrics allow us to assess the model's performance in terms of its ability to correctly identify positive examples (precision), correctly capture all positive examples (recall), and the harmonic mean of both precision and recall (f1-score). Support refers to the number of examples in each class, providing insights into the dataset's class distribution.

By using a combination of these evaluation metrics, we can gain a comprehensive understanding of the model's performance and make informed decisions about model selection and further improvements. Two environmental parameters were set for all the following experimentation, the train-test split was defaulted to 75:25 and the class weight was set to balanced.

A Random Forest Classifier

The results of the RF classifier can be seen in Appendix 3. A base model was built for the random forest classifier where no hyperparameters were adjusted, using the default setting provided. Each criterion was tested against the base model, which is using Gini Impurity. From the table, we can see that the choice of criterion bared little to no effect on the overall or balanced accuracy. The only significant metric was time taken. However, the trade-off between a shorter time to train against the slight decrease in accuracy lead to the conclusion that Log Loss is the best criterion for this model, RF_LogLoss will be used for the final evaluation to determine the final model.

B Support Vector Machines

A base model was built for the support vector machines where no hyperparameters were adjusted, using the default setting provided. This was used as a simple model for comparison and basic evaluation.

The Radial Basis Function Kernel

The C hyperparameter was tested first with a default value of 1 to set a base model for basic comparison to different C values. The values of 0.5, 1.5, 2 and, 5 were chosen to capture the relationship between C and the accuracy, as seen in Appendix 4. From the data, we can see that as C increased, the balanced accuracy decreased, whilst the overall accuracy increased, indicating the model becoming more overfit to the data. This is known behaviour as a smaller value of C leads to a simpler decision boundary with more misclassified samples, while a larger value of C leads to a more complex decision boundary with fewer misclassified samples. From the observations, the base model where C = 1 was kept for further testing with gamma values.

The default value for the gamma value is $\frac{1}{N_{Features}Var(X)}$, other values tested include $\frac{1}{N_{Features}}$, 0.1, 1 and 10. These values were chosen to observe how the accuracy changes as the sensitivity to nearby training examples increases. From the data seen in Appendix 5, as the sensitivity increased, the time taken grew exponentially with over an hour to train the later model as well as the model becoming more overfit with every iteration. Based on these patterns and the results, the RBF_Gamma_0.1 model was chosen as this model provided a valuable trade-off between the total accuracy and the balanced accuracy.

The Polynomial Kernel

The polynomial kernel was trained first with experimentation on the polynomial degree hyperparameter. Six values were tested within the range 0 – 5, these were chosen to observe the effects of capturing different levels of complexity. In Appendix 6, we can observe that by reducing the levels of captured complexity, the time taken increases exponentially whilst negatively effecting the accuracy across the model. The value chosen for future modelling was a polynomial degree of 2. Whilst this degree did not have the highest balanced accuracy, it provided an adequate trade-off between time taken, overall accuracy and balanced accuracy.

The next phase of testing was the polynomial coefficient hyperparameter, the base value for the coefficient was 0. The testing took the range 0 – 4, these figures were chosen to observe how increasing the weight given to each polynomial term affects the accuracy of the model. From Appendix 7, we can see that as the coefficient increases, the accuracy metrics don't change dramatically, this indicates that this hyperparameter bares little influence over the classification model. However, the time taken to train the model changes throughout each coefficient, with a coefficient of 2 being the lowest. Therefore, a coefficient of 2 was chosen for the final polynomial model as the accuracy increased slightly across both metrics whilst significantly reducing the time taken when compared to the base model.

Therefore, the final models to thoroughly examine include a RF model using the log loss criterion, a SVM model with the RBF kernel using a C value of 1 and a gamma value of 0.1 and a SVM model with the polynomial kernel using a polynomial degree of 2 and a polynomial coefficient of 2.

C Model Comparison

The confusion matrices and the classification reports for the final three models can be seen for multi-class and binary in Appendices 8 to 13 and Appendices 14 to 19, respectively.

Multi-Classification Comparison

Looking at the confusion matrices, the SVM classifier with an RBF kernel has the highest accuracy of 0.94, followed by the random forest classifier with an accuracy of 0.9 and the SVM classifier with a polynomial kernel with an accuracy of 0.71. The random forest classifier misclassifies only a small number of instances, whereas the other two classifiers misclassify a relatively larger number of instances. The SVM classifier with a polynomial kernel has the highest number of misclassifications. Looking at the classification reports, the random forest classifier has the highest precision and F1-score for the class "Background." However, it has a 0 recall, precision, and F1-score for the other classes. This indicates that the model is only able to predict the "Background" class accurately. The SVM classifier with an RBF kernel has the highest precision, recall, and F1-score for the classes "Background" and "Whitecoat," whereas the SVM classifier with a polynomial kernel has the lowest precision, recall, and F1-score for all the classes. Therefore, based on the given data, the SVM classifier with an RBF kernel is the best model for the multi-class model.

Binary-Classification Comparison

Looking at the confusion matrices, we can see that all three models have a high accuracy rate, with the SVM Classifier with RBF Kernel and the SVM Classifier with Polynomial Kernel both having the highest accuracy rate of 0.98. The Random Forest Classifier has a slightly lower accuracy rate of 0.95. However, looking at the classification reports, we see that the SVM Classifier with RBF Kernel has the best performance in terms of precision, recall, and F1-score for both classes. It has a precision of 0.89 and a recall of 0.93 for the "Seal" class, which is the highest among the three models. Therefore, we can conclude that the SVM Classifier with RBF Kernel is the best model for binary and multi-classification and will be chosen for the final model.

In the final model, there will be SVM classifier with a RBF kernel with C = 1 and a gamma value of 0.1 to get the highest accuracy for seal classification.

Part 4: Discussion

A The Data

The 900 HoG pointers provided came from a 10x10 block of a 60x60px image. These metrics are not recommended from researchers and an 8x8 block of a 128x64 image approach may have yielded a more reliable model (Tyagi, 2021). Therefore, future work within this project will require the raw images to be processed in a more standardized way.

B The Approach

The approach taken throughout this report was always to build upon the fundamental elements into a final product, with an emphasis on utilising all knowledge gained from the lecture and lab materials. The initial development was established using the provided specification and dataset. From these documents, much information about the sources and nature of the model could be interpreted, this lead to further research into the industry standards when dealing with the sources as well as revision into discussed lecture topics. This cycle of adopting research, revising lecture materials, and applying the knowledge continued throughout the report until this discussion. This approach to developing a machine learning model is, to the best of my knowledge, a scientific attitude toward development and highlights the core values of this model. Despite this, the collaboration between peers to enhance ideas and creativity suggested within the specification was never truly achieved and could have provided critical insight into other approaches that may have enhanced this project. This ideology will be implemented within future projects.

C The Methods Used

Feature selection was a large aspect within this report, the choice of features highly dictates a model's accuracy when classifying. The choice to use PCA was based on prior research into dimensionality reduction algorithms. Additionally, with the dense multi-dimensional data it was deemed inappropriate to use algorithms such as Singular Value Decomposition or Linear Discriminant Analysis. However, Isomap Embedding may have also been an effective algorithm to utilise within this model, if the underlying structure of the data is non-linear and if the relevant features for classification are not well represented by the principal components. Isomap may also be better if there are complex interactions between the features that cannot be captured by a linear transformation. Further research should test this implementation.

SVM and Random Forest were chosen for the classification algorithms. These are two well-known algorithms for dealing with multi-dimensional data. However, alternative algorithms could have been equally as effective for image classification tasks involving HoG features and colour histogram features. One alternative algorithm that could have been effective is Convolutional Neural Networks (CNNs). CNNs are well-suited for image classification tasks and have been shown to outperform other machine learning algorithms on a variety of image classification tasks, including those involving HoG features and colour histograms. Therefore, this algorithm could be tested and compared in further research

The metric methods used within this report provide a deep insight into the performance of various models. However, these metrics may be misleading in imbalanced datasets where one class is much more frequent than the others. Therefore, alternative metrics like Area Under the Receiver Operating Characteristic Curve (AUC-ROC) or Area Under the Precision-Recall Curve (AUC-PR) could be used to evaluate the models' performance. Additionally, the report could have provided more insights into the dataset's class distribution to understand the model's ability to generalize to unseen data.

D The Results

There are results from this report that need to be analysed and adjusted within further research. However, this report has provided an insight into a scientific approach for handling a classification problem by evaluating a dataset and assessing each step thoroughly. In terms of the classification model, the results of the model will be tested independently by the moderator using an unknown output. However, due to the level of detail in each aspect of the development, the model is likely to perform successfully and close to that of the split test data (~0.97 for binary classification and ~0.94 for multi-classification for general accuracy).

Although this project could have been expanded further, within the limited timeframe, a reliable classification model has been successfully implemented utilising real experimental data to accurately predict and categorise satellite images of seal environments.

References

Baheti, P. (2023, March 2). *train-validation-test-set*. Retrieved from https://www.v7labs.com/: https://www.v7labs.com/blog/train-validation-test-set

Bristow, H., & Lucey, S. (2014). *Why do linear SVMs trained on HOG features perform so well?* arXiv preprint.

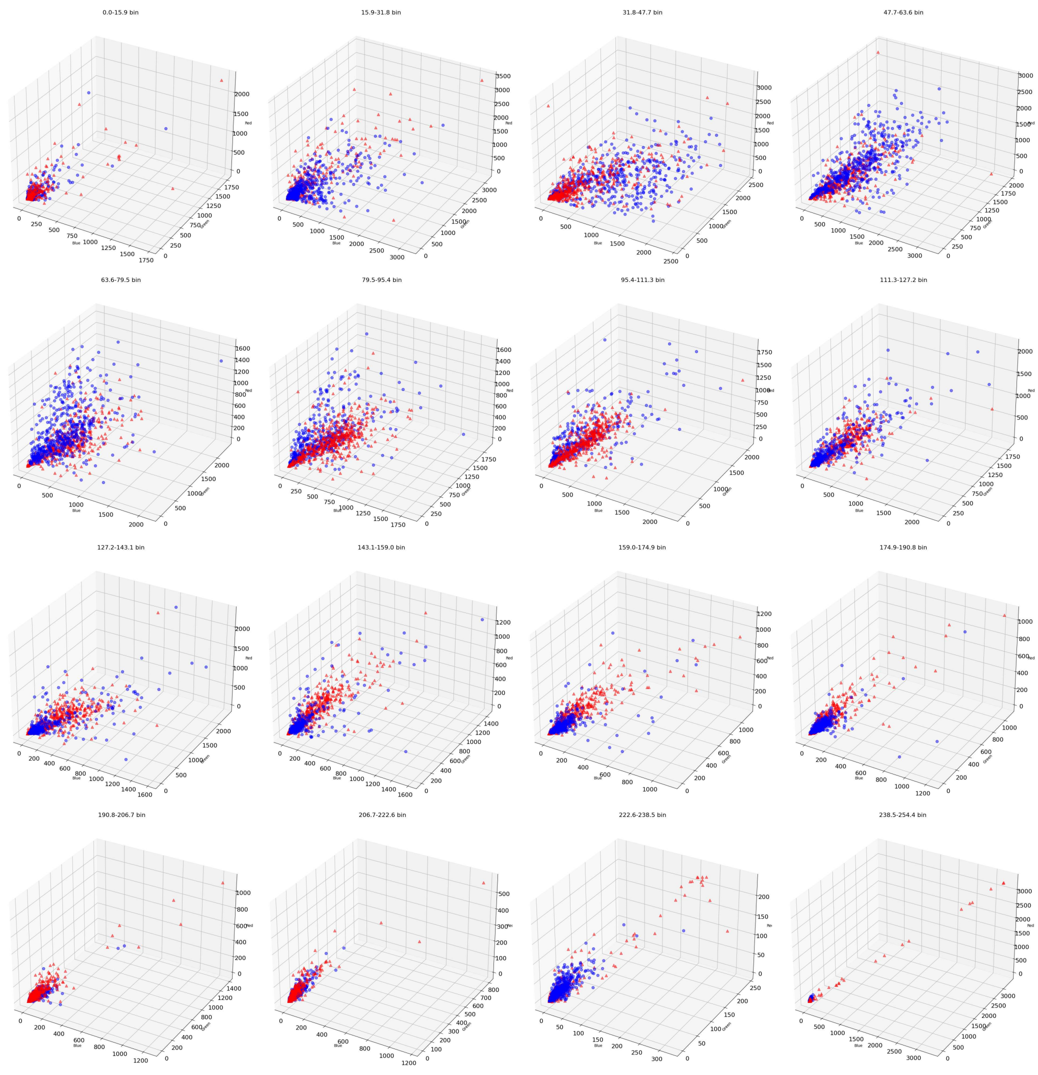
Gurung, R. B. (2020). *Random Forest for Histogram.* Stockholm: Stockholm University.

Terzic. (2023, Febuaray 20). Lecture 10: Trees and Forests . *CS5014 Machine Learning*. St Andrews, Fife, Scotland: University of St Andrews.

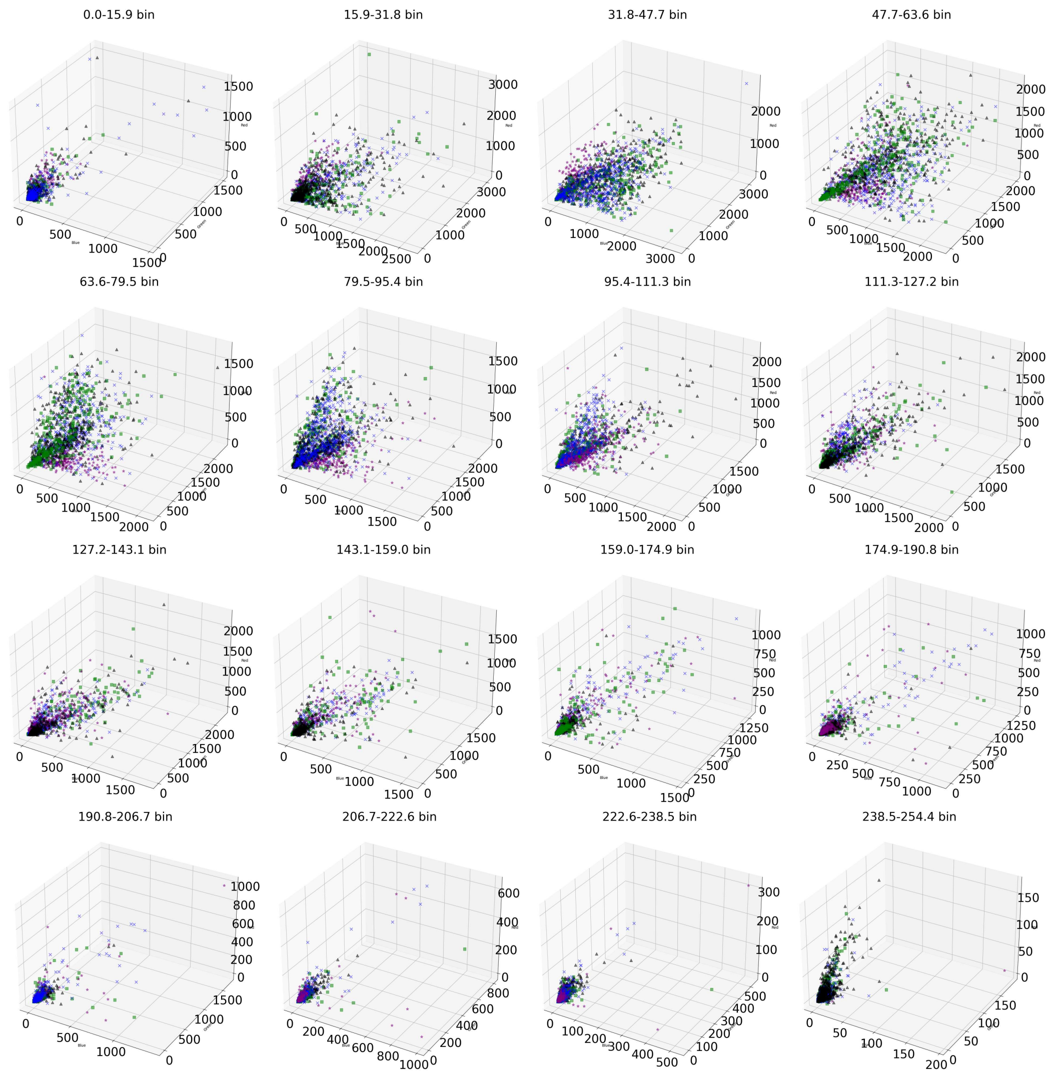
Tyagi, M. (2021, July 4). *HOG (Histogram of Oriented Gradients): An Overview*. Retrieved from https://towardsdatascience.com/: https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f

University of St Andrews. (2022). *Scientific Advice on Matters Related to the Management of Seal Populations: 2021.* St Andrews: University of St Andrews.

Appendices



Appendix 1: Scatter graphs of all the colour bins for Background (red) and Seals (blue)



Appendix 2: Scatter graphs of all the colour bins for Whitecoat (black), Moulting Pup (blue), Dead Pup (green) and Juvenile (purple)

Model Name	Criterion	Time Taken	Accuracy	Balanced Accuracy
RF_Base	Gini	01:30:04m	0.901	0.268
RF_Entropy	Entropy	02:11:08m	0.902	0.27
RF_LogLoss	Log Loss	02:10:05m	0.902	0.27

Appendix 3: Results from Random Forest Classification using Various Criterion for Mult-Classification

Model Name	C	Time Taken	Accuracy	Balanced Accuracy
RBF_Base_C	1	5:31:01m	0.954	0.58
RBF_C_0.5	0.5	6:01:03m	0.947	0.61
RBF_C_1.5	1.5	5:17:08m	0.955	0.57
RBF_C_2	2	5:19:05m	0.955	0.562
RBF_C_5	5	5:54:08m	0.954	0.541

Appendix 4: Results from Experimentation with the C Hyperparameter using the RBF Kernel on Multi-Classification Data

Model Name	Gamma	Time Taken	Accuracy	Balanced Accuracy
RBF_Base_Gamma	Scale	5:38:01m	0.954	0.58
RBF_Gamma_Auto	Auto	12:56:09m	0.841	0.65
RBF_ Gamma _0.1	0.1	04:00:02m	0.942	0.619
RBF_ Gamma _1	1	41:14:04	0.883	0.22
RBF_ Gamma _10	10	60:00:00	NA	NA

Appendix 5: Results from Experimentation with the Gamma Hyperparameter using the RBF Kernel on Multi-Classification Data

Model Name	Degree	Time Taken	Accuracy	Balanced Accuracy
Poly_Base_Degree	3	8:58:01m	0.953	0.515
Poly_Degree_0	0	18:10:3m	0.004	0.2
Poly_Degree_1	1	3:53:9m	0.882	0.628
Poly_Degree_2	2	3:39:2m	0.950	0.58
Poly_Degree_4	4	17:50:7m	0.946	0.447
Poly_Degree_5	5	36:50:3m	0.933	0.366

Appendix 6: Results from Experimentation with the Polynomial Degree Hyperparameter using the Polynomial Kernel on Multi-Classification Data

Model Name	Coefficient	Time Taken	Accuracy	Balanced Accuracy
Poly_Base_ Coefficient	0	3:44:09m	0.950	0.580
Poly_ Coefficient_1	1	2:51:5m	0.947	0.589
Poly_ Coefficient_2	2	2:46:05m	0.945	0.584
Poly_ Coefficient_3	3	2:52:05m	0.945	0.584
Poly_ Coefficient_4	4	2:58.07m	0.945	0.588

Appendix 7: Results from Experimentation with the Polynomial Coefficient Hyperparameter using the Polynomial Kernel on Multi-Classification Data

X	Background	Dead Pup	Juvenile	Moulted Pup	Whitecoat
Background	13597	0	0	0	12
Dead Pup	59	0	0	0	10
Juvenile	62	0	0	0	0
Moulted Pup	507	0	0	0	61
Whitecoat	805	0	0	2	438

Appendix 8: Confusion Matrix of the Final Random Forest Classifier for Multi-Classification

	Precision	Recall	F1-score
Background	0.9	1	0.95
Dead Pup	0	0	0
Juvenile	0	0	0
Moulted Pup	0	0	0
Whitecoat	0.84	0.35	0.5
Accuracy			0.9
Macro Average	0.35	0.27	0.29
Weighted Average	0.86	0.9	0.87

Appendix 9: The Classification Report of the Final Random Forest Classifier for Multi-Classification

X	Background	Dead Pup	Juvenile	Moulted Pup	Whitecoat
Background	13297	7	16	195	94
Dead Pup	7	10	0	21	31
Juvenile	9	0	39	8	6
Moulted Pup	76	6	2	308	176
Whitecoat	33	14	4	196	998

Appendix 10: Confusion Matrix of the Final SVM Classifier with RBF Kernel for Multi-Classification

	Precision	Recall	F1-score
Background	0.99	0.98	0.98
Dead Pup	0.27	0.14	0.19
Juvenile	0.64	0.63	0.63
Moulted Pup	0.42	0.54	0.48
Whitecoat	0.76	0.80	0.78
Accuracy			0.94
Macro Average	0.62	0.62	0.61
Weighted Average	0.95	0.94	0.94

Appendix 11: The Classification Report of the Final SVM Classifier with RBF Kernel for Multi-Classification

X	Background	Dead Pup	Juvenile	Moulted Pup	Whitecoat
Background	9888	612	407	1881	821
Dead Pup	6	36	0	16	11
Juvenile	6	0	51	4	1
Moulted Pup	109	80	22	281	76
Whitecoat	88	192	23	196	746

Appendix 12: Confusion Matrix of the Final SVM Classifier with Polynomial Kernel for Multi-Classification

	Precision	Recall	F1-score
Background	0.98	0.73	0.83
Dead Pup	0.04	0.52	0.07
Juvenile	0.10	0.82	0.18
Moulted Pup	0.12	0.49	0.19
Whitecoat	0.45	0.60	0.51
Accuracy			0.71
Macro Average	0.34	0.63	0.36
Weighted Average	0.90	0.71	0.78

Appendix 13: The Classification Report of the Final SVM Classifier with Polynomial Kernel for Multi-Classification

X	Background	Seals
Background	13579	29
Seals	822	1123

Appendix 14: Confusion Matrix of the Final Random Forest Classifier for Binary-Classification

	Precision	Recall	F1-score
Background	0.94	1.00	0.97
Seal	0.97	0.58	0.73
Accuracy			0.95
Macro Average	0.96	0.79	0.85
Weighted Average	0.95	0.95	0.94

Appendix 15: The Classification Report of the Final Random Forest Classifier for Binary-Classification

X	Background	Seals
Background	13385	223
Seals	131	1814

Appendix 16: Confusion Matrix of the Final SVM Classifier with RBF Kernel for Binary-Classification

	Precision	Recall	F1-score
Background	0.99	0.98	0.99
Seal	0.89	0.93	0.91
Accuracy			0.98
Macro Average	0.94	0.96	0.95
Weighted Average	0.98	0.98	0.98

Appendix 17: The Classification Report of the Final SVM Classifier with RBF Kernel for Binary-Classification

X	Background	Seals
Background	13429	179
Seals	158	1787

Appendix 18: Confusion Matrix of the Final SVM Classifier with Polynomial Kernel for Binary-Classification

	Precision	Recall	F1-score
Background	0.99	0.99	0.99
Seal	0.91	0.92	0.91
Accuracy			0.98
Macro Average	0.95	0.95	0.95
Weighted Average	0.98	0.98	0.98

Appendix 19: The Classification Report of the Final SVM Classifier with Polynomial Kernel for Binary-Classification