# Machine Learning Model for Classification of Various Cat Breeds

Gabriel A. Abeyie — Cameron V. Adkins — Purnachandra Anirudh Gajjala
December 5, 2022

**Abstract**

In this project, we attempted to create a classifier for determining the breed of a cat based on an image of the animal. In particular, we perform processing on an existing dataset to produce features for our classifier. This feature extraction mainly consists of various segmentation approaches and analysis of the color histograms. These segments are further analyzed to obtain location information for the ears. These collected features are then passed into the standard machine learning pipeline using the scikit-learn library. We make use of the RandomForestClassifier to obtain a prediction. Results demonstrate a moderately accurate model given the wide range of input data.

## 1 Introduction

Image classification is one of the fundamental problems in computer vision. It serves as a basis for many other computer vision tasks such as object recognition, image segmentation and object detection. The goal of categorizing images into one of several predefined classes is called image classification. Although the task of classifying images comes easy for human beings, there are challenges with automated systems. By using machine learning techniques, images can be classified.

The goal of this project is to help identify the different breeds of cats through image classification. Classifying images involves taking a set of features from an image and feeding it to a machine learning model. These features are often found at specific points within an image, where the hope is that they will reveal something about the class.

Developments in machine learning and deep learning provides a variety a ways in dealing with image classification. These methods range from building neural networks to adopting a supervised learning methods. In this project, we adopt the random forest classifier from classical machine learning to attempt to a classification.

Random forests or random decision forests is basically a learning method for performing classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. In the case of classification, the result of the random forest is the class selected by most trees. Random decision forests correct for decision trees' habit of over fitting to their training set and generally outperform decision trees.

## 2 Data

For this project, we synthesized a new dataset from the existing Oxford-IIIT Pet dataset. This dataset contains a number of high quality images of both dogs and cats. For our purposes however, we only select the included 12 cat breeds from this dataset. Furthermore we restrict our dataset to include only images with a corresponding head bounding box. This leaves us with 1187 image samples ranging across the 12 breed classes.

As mentioned, the dataset includes a head bounding box for roughly half of the images. Additionally, the dataset also includes a image matte for all included images as a trimap. Before we begin feature extraction, we apply these included metadata files to the input to create a set of 3 images for our model.

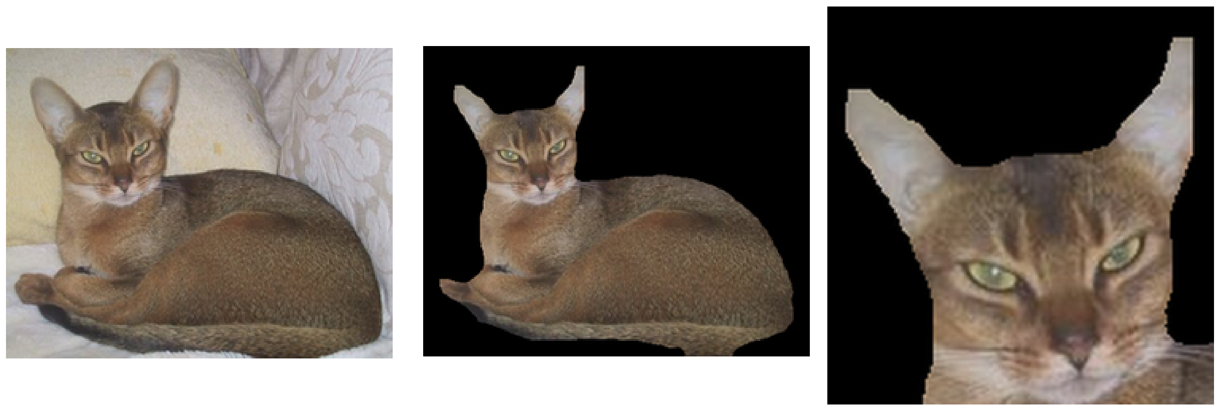

Figure 1: Example image being preprocessed by adding the included metadata from the dataset.

## 3 Feature Engineering

The main crux of our feature engineering falls into two camps: image segmentation and binned color histograms.

### 3.1 Segmentation

To describe image segmentation, it is a process where a digital image is divided into many subgroups known as segments. This technique helps to reduce the complexity of the image to make subsequent handling or examination of the image easier. A common mark is assigned to each image component or pixel that belongs to the same class.

For this project we implemented threshold, edge-based, and color-based segmentation.

#### 3.1.1 Threshold Segmentation

Thresholding is a simple way to get the basic structure of an object. This method takes a greyscale version of the image and scans each pixel. If the value is above a certain threshold, assign the pixel to a white value. Otherwise it becomes black. By doing this the overall form of the objects appears quite well. When combined with the provided image matte, our output displays the overall figure decently well.

We use this output to determine the positions of the ears. Our method simply walks the image until it finds the upper left/right most pixels that belong to the object and marks these as the ear positions. This does depend on the ears being the highest point in the image but observations suggest that this is often the case for this dataset.

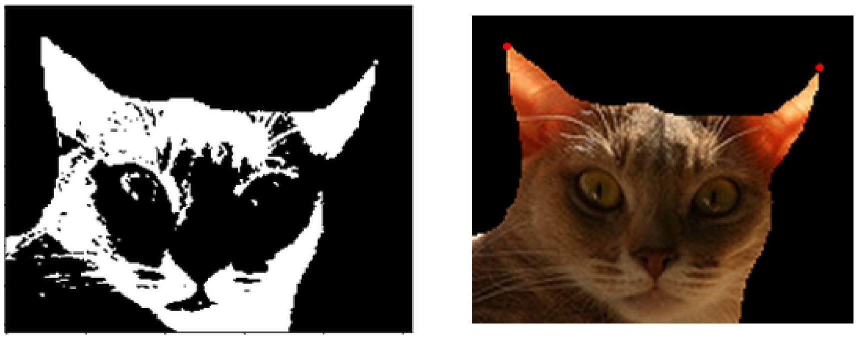

Figure 2: Example image after being processed by threshold segmentation. The red marks on the image denote the discovered position of the ears.

### 3.1.2 Color-based Segmentation

This is a more generalized case of the threshold segmentation. The main difference being that we now classify image segments into multiple parts based on the color rather than the greyscale value. In our implementation, we accomplish this via K-Means clustering.

This can be defined as finding subgroups in the data where data points in the same subgroup (cluster) are extremely similar. Clustering analysis can be carried out either on the basis of samples or on the basis of features, in which case we try to locate subgroups of samples based on features.

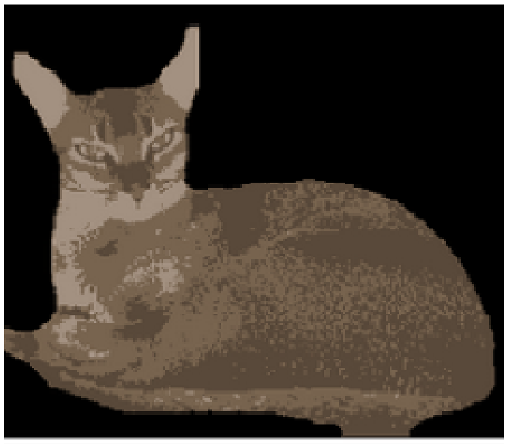

Figure 3: This image is segmented with a K value of 4. This corresponds to 4 unique colors being used to compose the image.

This feature was unfortunately unused in our final model, but demonstrates an area for future improvement.

### 3.1.3 Edge Detection

What divides two objects in an image? There is always an edge between two adjacent regions with different grayscale values (pixel values). The edges can be considered as the discontinuous local features of an image. We can make use of this discontinuity to detect edges and hence define a boundary of the object. This helps us detect the shapes of multiple objects in a given image. To detect edges, we use filters and convolutions.

Specifically, we make use of the Sobel and Laplace operators. This is a set of kernels that help detect image edges. By convolving the appropriate kernel with the input image's greyscale, we are able to obtain a fairly decent image of the edges.

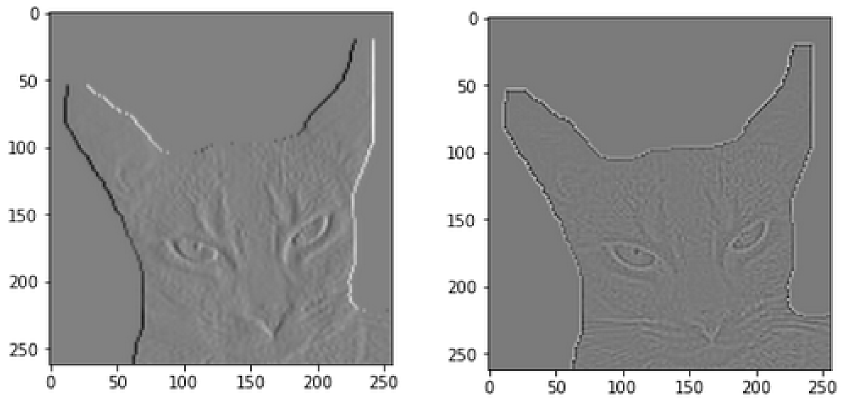

Figure 4: Left demonstrates the results of the Sobel operator while right corresponds to the Laplace operator. The different operators present features somewhat differently.

Again, this feature went unused but we have this available to us if we decide to pursue further development.

## 3.2 Color Histograms

Color histograms are a fairly simple way to introduce features like fur color and pattern into the feature set. A color histogram is simply a measure of how many times a particular color component appears. We represent this as a graph with separate axis for the individual components.

In our feature extraction, we make use of the corresponding OpenCV function to create 3 histograms with 5 separate bins to add to our feature vector. This gives us a decent bump in accuracy overall.

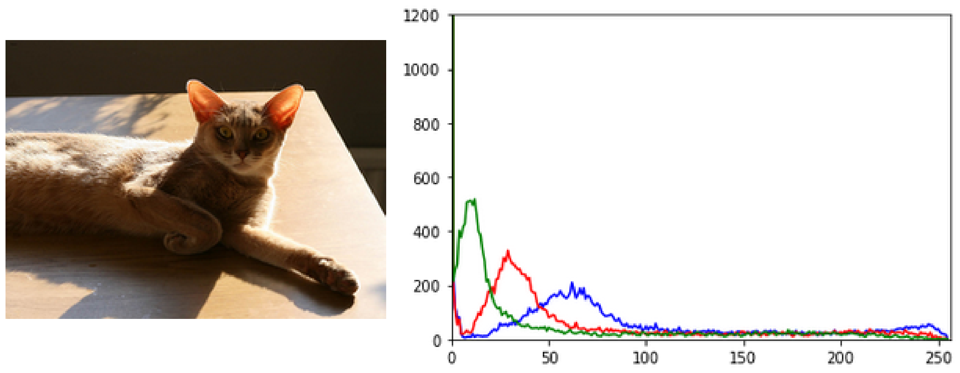

Figure 5: The image on the left gets turned into the histogram on the right during feature extraction.

## 3.3 Feature Selection

To assist in feature engineering, we created the following table to determine the most desirable features to include in our model:

| Breed | Size | Coat | Color |
|---|---|---|---|
| Abyssinian Cat | Small to medium,with males weighing 7 to 10 pounds and females weighing 6 to 8 pounds | Medium | Ruddy, red, blue, fawn |
| Bengal Cat | Medium to large with males: 10 to 18 pounds and females: 6 to 12 pounds | Medium | Bright orange to light brown, with dark spots or a distinctive marbling pattern |
| Birman Cat | Medium to Large, with males weighing 9 to 15 pounds and females weighing 6 to 10 pounds | Medium to Long | Seal point, blue point, chocolate point, lilac point |
| Bombay Cat | Medium, with males weighing 8 to 11 pounds and females weighing 6 to 9 pounds | short | black |
| British Shorthair Cat | Medium to large, with males weighing 12 to 18 pounds and females weighing 9 to 15 pounds | short, plush, dense | Blue, white, black, red, cream, smoke, silver and golden, plus a variety of patterns and shadings |
| Egyptian Mau Cat | Medium, with males weighing 10 to 14 pounds and females weighing 6 to 10 pounds | short to medium | silver, bronze, smoke |
| Maine Coon Cat | Large, with males weighing from 12 to 15 pounds and females weighing from 9 to 12 pounds | Shorter on theshoulders and longer on the stomach, smooth, shaggy | Most commonly brown tabby, but other colors and patterns are possible |
| Persian Cat | Medium to large, with males weighing 9 to 14 pounds and females weighing 7 to 11 pounds | Long, thick, glossy | White, blue, black, red, cream, chocolate, lilac |
| Ragdoll Cat | medium with 10 to 15 pounds | Semi-long, plush, silky | Seal, blue, chocolate, lilac, red and cream, plus various patterns and shadings, including bi-color, van, colorpoint and mitted |
| Russian Blue Cat | medium, 8 to 15 pounds | Short, dense, fine, plush | Blue with silver tips |
| Siamese Cat | Medium, with males weighing 11 to 15 pounds and females weighing 8 to 12 pounds | Short, fine, glossy | Seal point, chocolate point, blue point, lilac point |
| Sphynx Cat | Medium with Male: 8 to 12 pounds, Female: 6 to 9 pounds | Hairless, although some have a fine down | White, black, blue, red, cream, chocolate, lavender, cinnamon and fawn, plus various patterns and shadings |

## 4 Model Creation

Our software follows the standard machine learning pipeline for model creation. Our pipeline is fairly simple due to this being the secondary aspect of this project.

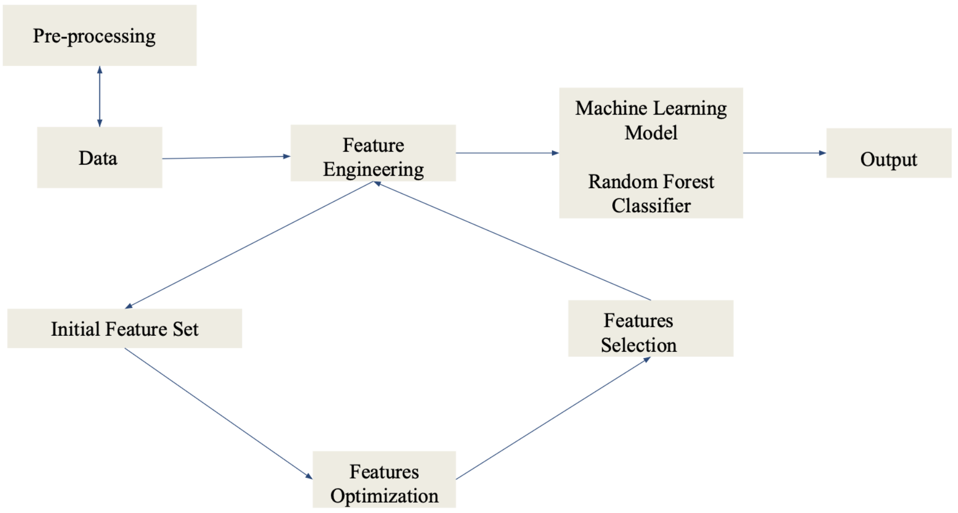

Figure 6: The machine learning pipeline that we employed in our project.

It should be noted that we employ the SMOTE rebalancer in addition to the scikit-learn robust scaler to obtain slightly better results.

As previously mentioned, the random forest classifier is used as our classifier model. This provides us with a fairly simple way to distinguish between multiple classes as many of the difficult aspect of model creation are taken care of by this classifier.

## 5 Results

Unfortunately, our results for model did not turn out as well as we would have hoped. We achieved a roughly 30% accuracy for our classifier.

```
Average Cross Validation Score from Training:
0.3547619047619047


Confusion Matrix:
[[10  6  1  1  1  1  3  1  0  2  1  3]
 [ 4 10  2  3  0  0  5  2  0  0  1  0]
 [ 0  1  6  1  2  4  1  2  6  1  5  1]
 [ 0  0  0 24  1  0  1  0  0  1  0  0]
 [ 0  2  1  1 15  1  5  3  3  4  0  0]
 [ 1  1  2  0  0  7  0  0  0  5  7  2]
 [ 5  7  2  1  0  0  4  4  2  1  3  1]
 [ 4  1  0  0  1  0  1  9  4  2  1  3]
 [ 3  0  4  0  1  4  0  5  6  0  5  3]
 [ 1  1  0  2  8  4  1  1  1 11  0  0]
 [ 0  5  7  1  0  3  7  1  3  0  5  3]
 [10  1  3  0  2  3  2  1  1  1  2  8]]
```

```



Missing classifications (if any): set()
Test Statistics:
                   precision    recall  f1-score   support

       Abyssinian       0.26      0.33      0.29        30
           Bengal       0.29      0.37      0.32        27
           Birman       0.21      0.20      0.21        30
           Bombay       0.71      0.89      0.79        27
British_Shorthair       0.48      0.43      0.45        35
     Egyptian_Mau       0.26      0.28      0.27        25
       Maine_Coon       0.13      0.13      0.13        30
          Persian       0.31      0.35      0.33        26
          Ragdoll       0.23      0.19      0.21        31
     Russian_Blue       0.39      0.37      0.38        30
          Siamese       0.17      0.14      0.15        35
           Sphynx       0.33      0.24      0.28        34

         accuracy                           0.32       360
        macro avg       0.31      0.33      0.32       360
     weighted avg       0.31      0.32      0.31       360



Testing Accuracy: 0.3194444444444444
```

A large part of the issue relates to iteration time, as it takes a significant period of time (roughly 30 min) to complete the pipeline. As a result, we were unable to iterate quickly enough to produce more than a mildly successful model.

## 6 Limitations & Future Work

Despite our best efforts, we got a moderate accuracy on our model, hence that being the main limitation. To improve upon the accuracy we have, we would need to work more on feature engineering. As mentioned, two of our three segmentation features ended up unused on this final model which leaves open the door for more improvement. We also have a number of other options to consider, for instance contour generation. Our original plan was to introduce not only ear positions but eyes and nose as well. Making use of these various tools we have to create better input features would lead to better results.

Another alternative would be to abandon this automated approach and approach this project from a data curation perspective. This would entail the creation of software to allow for easy annotation of images for our purposes. This would remove the image processing hurdle but introduce the rather large task of annotation.

## 7 Conclusion

Although the model that we created was less than ideal, we still were able to successfully complete the pipeline. With this experience of image processing under our belts, we have better equipped ourselves to handle this type of project in the future. Future work seems fairly straight for this project as our results were less than desirable, but we now have the resources to act on this.

## 8 References

## References

[1] Imad Dabbura. *K-Means Clustering.* URL: https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a.

[2] Dhaval Limbachiya. *Image-Segmentation using different approaches.* URL: https://github.com/thewall27/Image-Segmentation.

[3] Omkar M Parkhi et al. "Cats and dogs". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition.* 2012, pp. 3498–3505. DOI: 10.1109/CVPR.2012.6248092.

[4] Tony Yiu. *Understanding Random Forest.* URL: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.