Nate Kai Tsang, Jasper Kan, Thor Valentin Olesen

Visualizing the World: A Computer Vision Data Science Project

DATA C200 Final Project

Introduction:

This project consists of a large dataset with images belonging to 20 different categories. The goal is to classify the category of a given image using a machine learning model. We are given 1501 images to train our models, and 716 images to evaluate the model accuracies. We began by performing exploratory data analyses to give us insights into the feature selection process. Then, we trained 4 models: Logistic Regression, KNN, SVM and Random Forest. For each model, we used cross validation to select the best parameters, and a feature selection function to select the top 10 features. In the end, we identified Logistic Regression as the best model because it has one of the lowest validation errors, a fast computation time and is relatively easy to interpret.

Image recognition using machine learning techniques is important when dealing with a large database of images. While it helps some companies monetize their visual content through automatic classification of images, it also enhances cybersecurity by identifying and censoring inappropriate or offensive images on social media platforms. We hope to gain a better understanding of using machine learning models for image classification through this project.

Data Cleaning:

A majority of our efforts were spent extracting and cleaning the data into a format usable by the ML models. Our goal was to make this step as repeatable as possible for other users. To achieve this, we ensured that users didn't have to make any manual changes to the original zip files. The following steps were performed:

1. Extract train and test images from zip file
2. Convert grayscale images into representative RGB values
3. Remove corrupt files
4. For test data, remove spaces in file names (e.g. validation_pic (10).jpg)
5. Encode train data by extracting class from file name
6. Load cleaned data into data frames for use in the following notebooks

Exploratory Data Analysis:

To better understand the given data, we analyzed both the categories and dominant colors of all the images. Figure 1 shows the number of images in the training dataset in each of the 20 categories. Out of all images, gorilla is the most common category and kangaroo the least common. In Figure 2, we see the distribution of the most dominant colors across all images, with black being the most common and silver the least common. While this information is not particularly helpful in the feature selection process, it helps us understand the dataset better, which sometimes could lead to key insights.

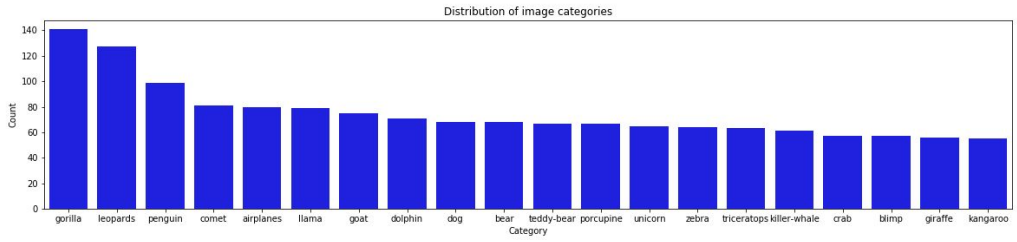

Figure 1: Distribution of image categories

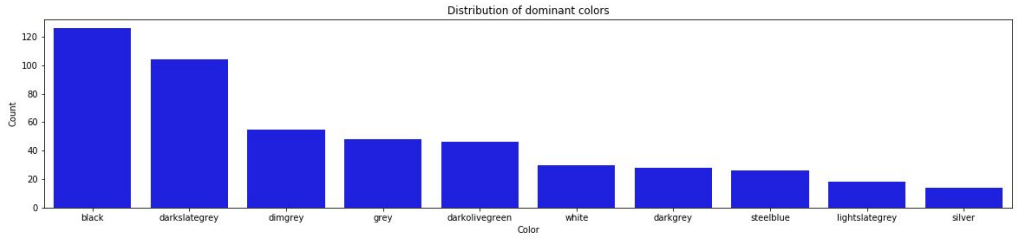

Figure 2: Distribution of dominant colors

We also decided to look at the distribution of the dominant colors in images by categories, shown in Figures 3-22 in Appendix A. As seen from these plots, almost all categories have a dark color as the most dominant color, with black being the most common. There are only 5 unique colors as the most dominant color across 20 categories. This tells us that the most dominant color in an image might not be the best distinguishing feature.

Feature Selection:

To select potential features for our models, we relied on our basic understanding of images, results from the EDA process and external sources. We knew that image size, aspect ratio and RGB value define an image. Therefore, we included these as potential features at the beginning. We also believed that the most dominant color in an image might help classify the image, even though the EDA results suggested otherwise. We decided to include the RGB value of the most dominant color as a potential feature, because we could still remove it later on when narrowing down the list of features.

Finally, to find more advanced potential features, we conducted research on feature detection and found that corners were helpful in identifying an image. We decided to use ORB (Oriented FAST and Rotated BRIEF) to detect keypoints in an image. We also discovered some useful functions in the scikit-image library that helped us quantify an image's luminance and peaks. In total, we found 21 potential features.

In the end, we used a tree-based feature selection function from the scikit-learn library to identify the top 10 features out of all the potential features. Most of the selected top features were expected, such as image size, aspect ratio and values related to RGB. The number of keypoints identified using ORB was one of the coolest features because we only learned about its importance after conducting extensive research on image feature detection. The average

luminance of a filtered image was another cool feature because the value was obtained by first using the Sobel transform.

Unsurprisingly, the most dominant color in an image did not work well. As discovered in the EDA process, most images have the same dominant color, making it hard to use the most dominant color to classify an image. The top 10 features can be found in the results section.

Model Training:

Four separate models were trained as part of this project: 1) logistic regression, 2) K-nearest neighbors, 3) random forest, and 4) support vector machine. While each of these are capable of doing multi-class classification, they differ in their underlying algorithms. Logistic regression is a method that utilizes the (linearly parameterized) sigmoid function to estimate the probability that an observation belongs to a particular class. KNN is a non-parametric method that clusters similar observations based on their features, and predicts their class based on similarity. Random forest is an ensemble of decision trees that introduces randomness when splitting on a node. The algorithm determines which split to make by finding the the feature and value pair that most significantly decreases error. SVM classifies observations by finding decision boundaries (i.e. hyperplanes) in n-dimensional space that best separate out classes.

We executed the following steps for each of the models.

1. Scale feature values appropriately for models
2. Instantiate model in sklearn
3. Identify hyperparameters to test using cross validation
    a. Logistic regression: regularization, solver
    b. KNN: number of clusters (i.e. “nearest neighbors”)
    c. Random forest: number of features to utilize in training process
    d. SVM: kernel type, regularizations
4. Use grid search with cross validation to select the most optimal hyperparameters
5. Using optimal hyperparameters, train a model on the training data
6. Calculate accuracy on the train and validation data, and generate the confusion matrix
7. Bootstrap confidence intervals for validation accuracy to get a better understand of the variation in results

Results:

In order to select the best features, we ran a tree-based feature selection method to narrow the list of all potential features down to 10, which are: image size, aspect ratio, average of the red-channel, average of the blue-channel, average of the green-channel, median of the red-channel, median of the blue-channel, standard deviation of the blue-channel, number of keypoints identified using ORB and the average luminance of a filtered image (used to approximate the number of edges). We ran our models with the top 10 features and all the features respectively. Since the model accuracies were very similar in both cases, we decided to use less features to avoid overfitting and to reduce computation time.

One of the most significant factors that affected model accuracy was the preprocessing of feature values. When normalizing the data, the model performed 10-50% *worse* compared to

models trained on the original data. In contrast, models trained on scaled feature values performed significantly better than the original data.

Our models achieved a validation accuracy of 40%-46%, with SVM performing the highest (46% accuracy) and KNN performing the lowest (40% accuracy). They also achieved a The 95% confidence interval for SVM was between 38-54%. We selected the Logistic Regression model to predict the test set because it had descent accuracy, is a simple model that's easy to interpret, and it didn't overfit the train data as much as some of the other models. We expect this model to generalize best to the test data.

Table 1: Summary of model performance

	Logistic Regression	**KNN**	**Random forest**	**SVM**
Validation accuracy	0.44	0.4	0.43	0.46
95% confidence intervals for validation accuracy	[0.36, 0.52]	[0.33, 0.48]	[0.35, 0.51]	[0.38, 0.54]

Conclusion:

In this project, we limited our features to numerical values only. However, it is possible to use a pixel by pixel matrix as a feature, which could be significantly improve the final model. Despite the feature constraints presented in this project, we were still able to achieve almost a 50% accuracy on the validation set by focusing our efforts on transforming the data and optimizing model parameters to increase model accuracy. It would be interesting to see how much we could improve the model accuracy by if we combine our data cleaning and parameter optimization efforts in this project with more powerful features.

In future research opportunities, we could spend more time on the EDA process in order to understand what values differ the most across images of different categories. By having better potential features initially, we could further improve the selected top features given by a feature selection method. We could also explore the difference between a tree-based feature selection method and a L1-based selection method.

Appendix A: Plots of the Distributions of Dominant Colors in Images by Categories

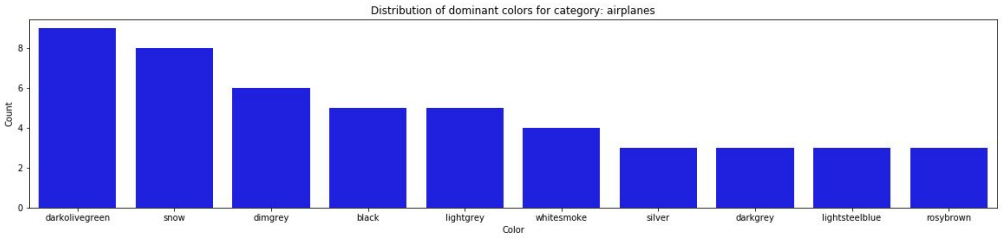

Figure 3: Distribution of dominant colors for airplanes

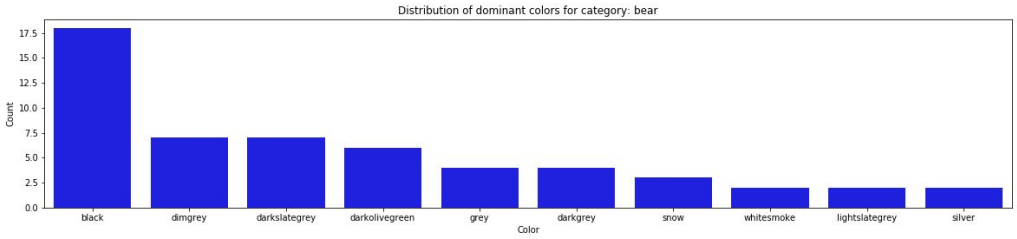

Figure 4: Distribution of dominant colors for bear

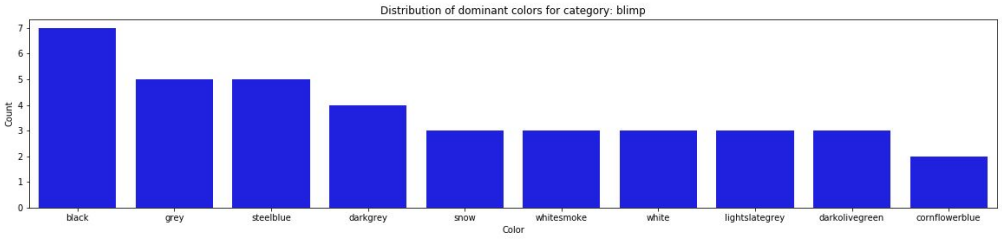

Figure 5: Distribution of dominant colors for blimp

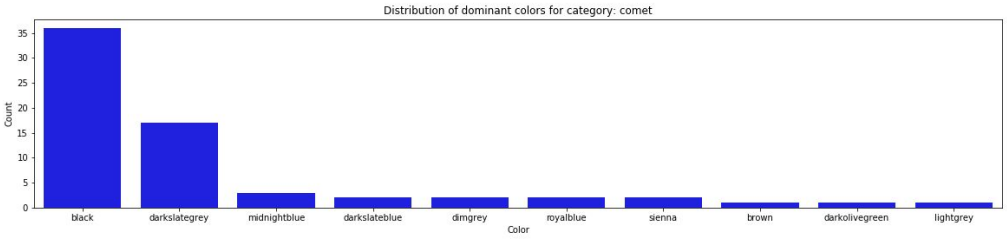

Figure 6: Distribution of dominant colors for comet

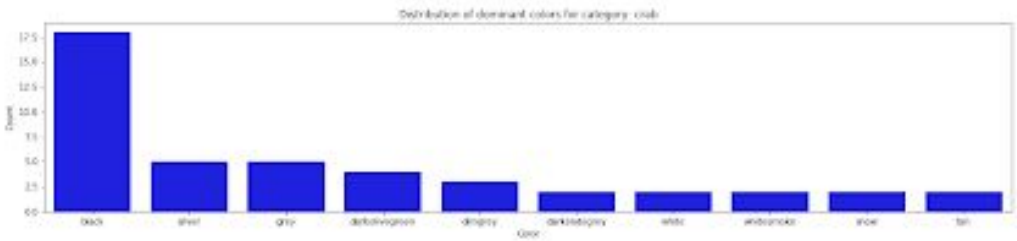

Figure 7: Distribution of dominant colors for crab

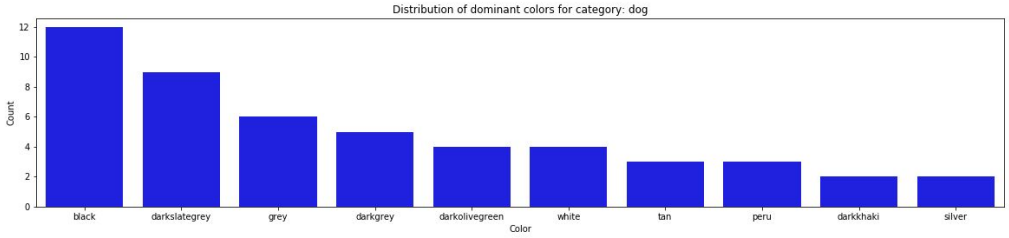

Figure 8: Distribution of dominant colors for dog

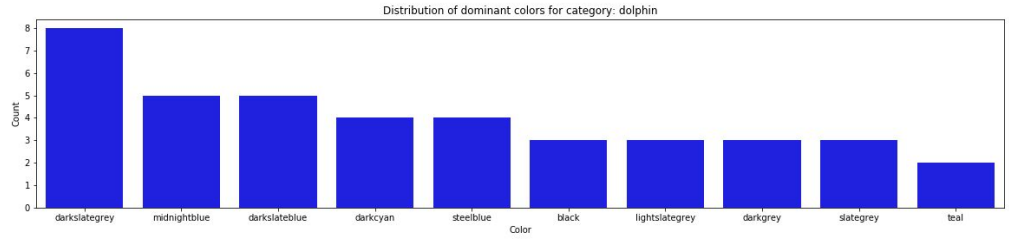

Figure 9: Distribution of dominant colors for dolphin

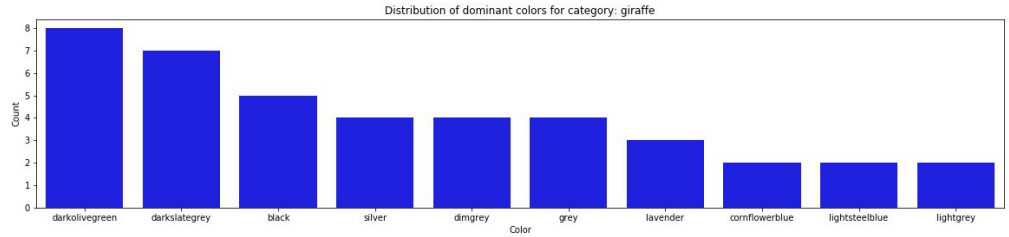

Figure 10: Distribution of dominant colors for giraffe

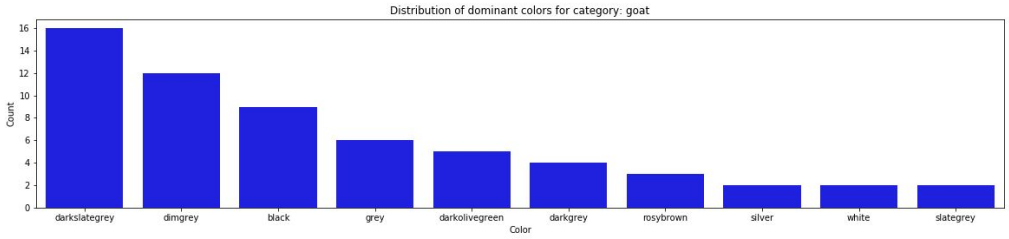

Figure 11: Distribution of dominant colors for goat

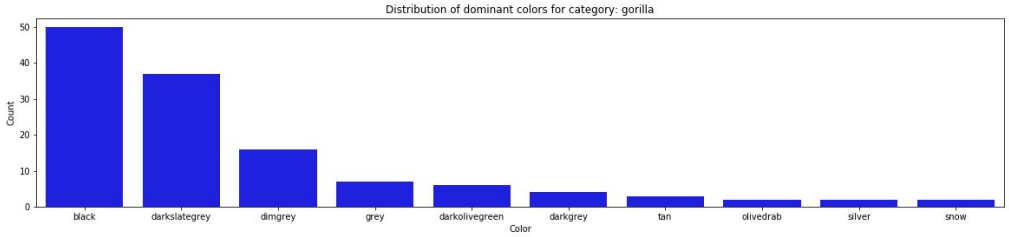

Figure 12: Distribution of dominant colors for gorilla

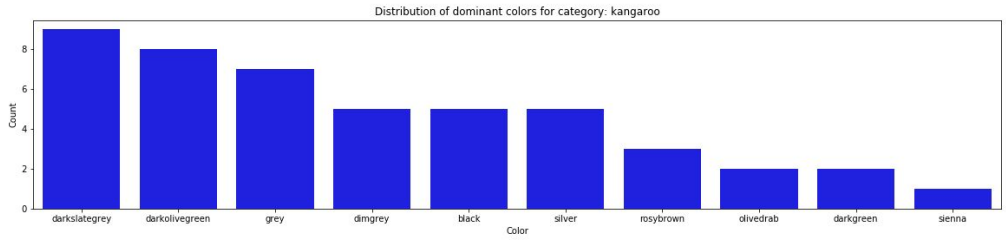

Figure 13: Distribution of dominant colors for kangaroo

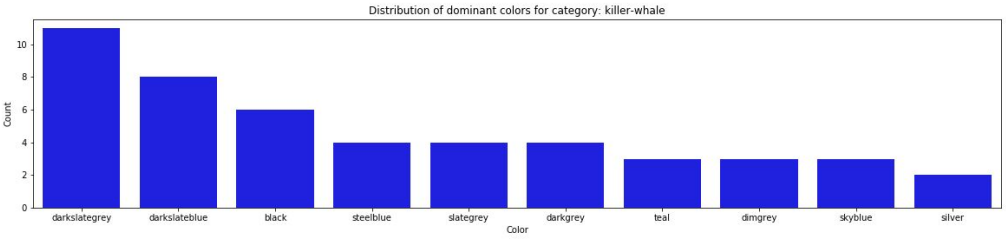

Figure 14: Distribution of dominant colors for killer-whale

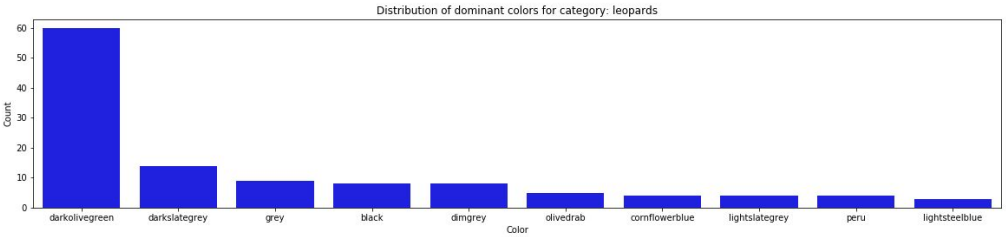

Figure 15: Distribution of dominant colors for leopards

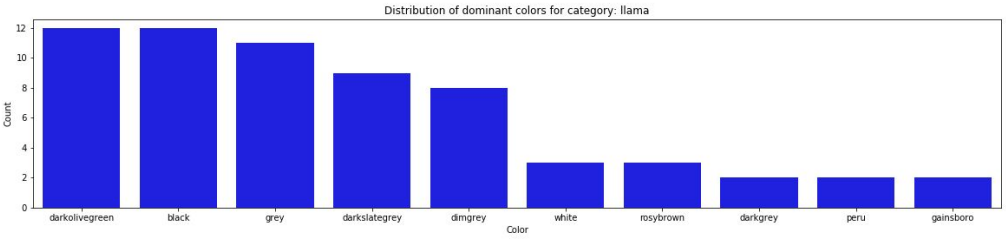

Figure 16: Distribution of dominant colors for llama

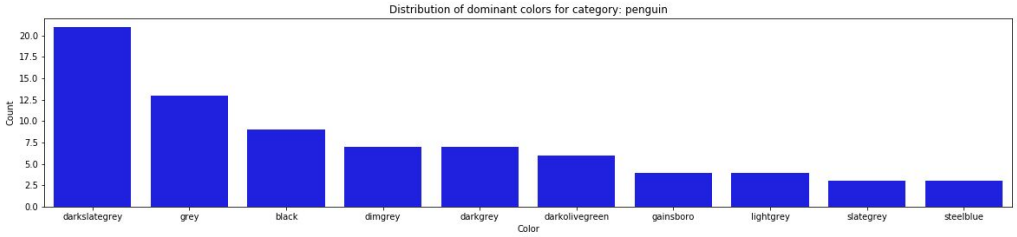

Figure 17: Distribution of dominant colors for penguin

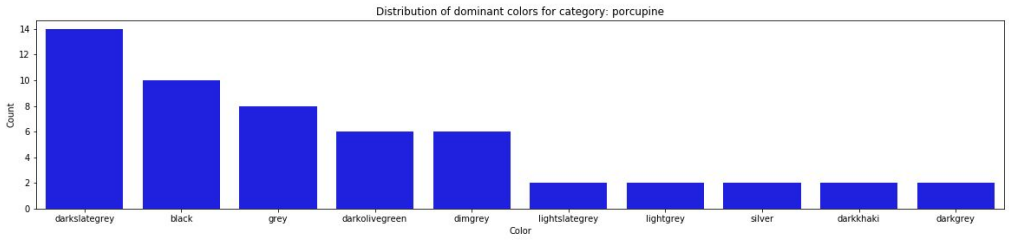

Figure 18: Distribution of dominant colors for porcupine

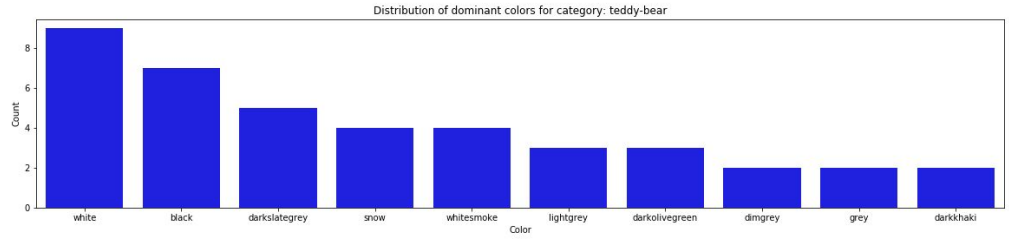

Figure 19: Distribution of dominant colors for teddy-bear

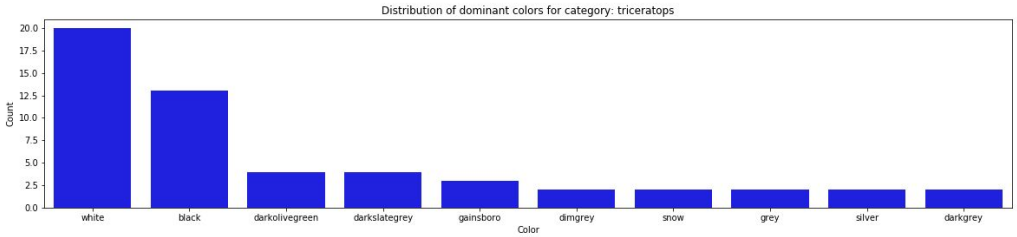

Figure 20: Distribution of dominant colors for triceratops

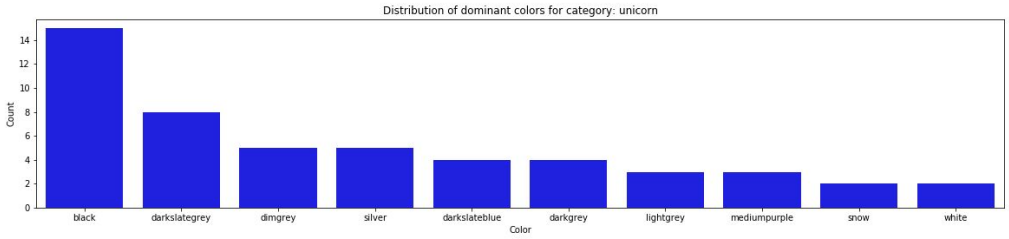

Figure 21: Distribution of dominant colors for unicorn

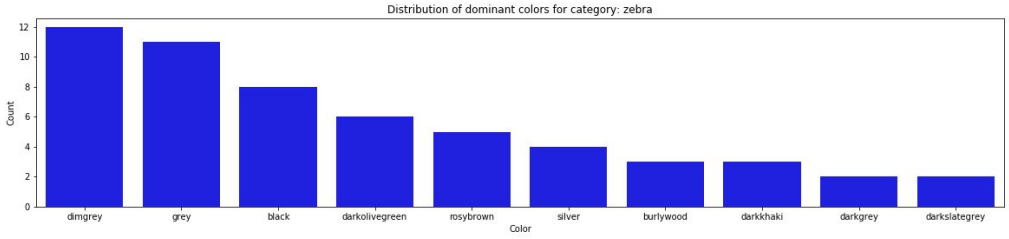

Figure 22: Distribution of dominant colors for zebra