AML Group 2 - Cats vs. Dogs

Team:

Ben Perkins
Lauren Madar
Mangesh Walimbe
Samin Barghan

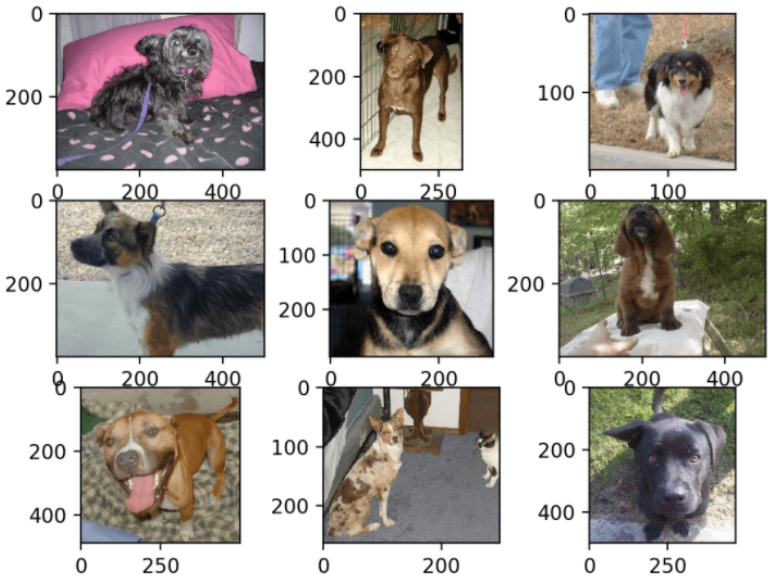
Abstract

The task of classifying images is currently fertile ground for research in the area of computer vision and deep learning. One of the fundamental tasks involved is that of object detection within the images. A so-called ‘bounding box’ is drawn which gives the algorithm used a rectangular area to work with, as opposed to the much more unbounded irregular areas of a natural object. In order to predict these bounding boxes, our team will implement a Linear Regression model with gradient descent to achieve convergence. The task of classifying the given images as ‘Cat’ or ‘Dog’ will be handled at first by a Logistic Regression model. This will arrive at predictions via stochastic gradient descent, primarily because of the large size of the data. This will allow for a very good fit, but in a much less resource-intensive manner than with standard gradient descent. Both models will be evaluated by Average Precision, which is a standard metric for image classification. Additionally, both CXE and MSE will be used as well. We also plan to implement a confusion matrix and the Area Under Curve based on the Precision and Recall metrics, as these are often used to estimate how well a model performs. Our pipeline will include both column and image transformers, feature engineering for numeric data, and a regression pipeline.

Data

The csv file includes 12,966 rows of cats and dogs images with 21 columns:

Identifying columns: ImageID, Source, LabelName, Confidence
Dimensional and positional columns: XMin, XMax, YMin,YMax, XClick1X, XClick2X, XClick3X, XClick4X, XClick1Y, XClick2Y, XClick3Y, XClick4Y
Bounding box and image descriptive columns: IsOccluded, IsTruncated, IsGroupOf, IsDepiction, IsInside



When we look at a few random images, we can see that the photos vary in color and have different shapes and sizes. Also, we can see a photo with both a cat and dog, with the cat being barely visible (bottom row middle) so this shows any classifier fit on this type of photos will have to be robust.

The first step to prepare data must be to standardize the images. Photos will have to be reshaped before modeling so that all images have the same shape and size. One approach we may use would be to load all photos and look at the distribution of the photo widths and heights then determine a new image size that fits the majority of the images. Smaller size allows a model to train more quickly. Another approach would be to start with a fixed size of 200x200 pixels. We can also filter color images to determine where the majority or highest density of each color pixel lies within the image.

The metadata contained in the csv file will need to be matched to each image file, and during Exploratory Data Analysis, we will determine relationships between any of the columns using pandas. For example, how many images contain more than one cat or dog (IsGroupOf)? How many of those images have IsOccluded, IsTruncated, IsInside? Can we determine if the bounding box of one object is larger than the other in order to guess the ‘main’ object? This will drive creation of additional features.

The code and project files are stored in a GitHub repository: i526Sp21Group2. We will impute missing data and document the strategy used, if needed (depending on the results of EDA). NumPy DataFrames embedded in our project Jupyter Notebook will track our exploration and transformation of data and engineering of any features ahead of training and fitting. Other Python libraries may be used for visualizations and will be documented.

Algorithm and Models

The team will use SciKit-Learn tools, along with NumPy and Pandas.

Linear Regression

We are planning to use linear regression with gradient descent to predict the bounding box.

- The dataset has the extreme end point coordinates of the bounding box as well as for the whole image. Based on these coordinates, we need to predict the bounding box in the image. Linear regression models will be best suited for this scenario. We can establish the relationship between these coordinates using linear regression and predict the bounding box.

- Here we need to iterate through each point to identify the boundary of the box. So we will use gradient descent.

Logistic Regression

Once we predict the bounding box, we will use logistic regression to classify the images.

- Logistic regression is the classification algorithm. It assigns observations to a discrete set of classes based on the probability. Using this regression method, we will be predicting the class of the 'main object' in the image (cat or dog). We will use binomial logistic regression for this problem.

- Since the dataset is large, we will use stochastic gradient descent. It chooses only one random datapoint while changing weights and has an early stop. It is faster than gradient descent. SGDClassifier implements linear models with stochastic gradient descent.

Metrics

To gauge the performance of the bounding box predictions, we can find the Average Precision for each class, and then the Mean Average Precision over both classes. Setting an appropriate threshold to evaluate detection choices, using the Intersection Over Union (IOU), will give us the correctness of the detections. The Area Under the Curve (AUC) of the Precision and Recall values based on the object detections will give the Mean Average Precision, which is a good metric by which to judge which model's specific parameter set performs the best.

Use of Logistic Regression in the first phase suggests using the Cross-Entropy Loss (CXE) function to measure the success of the model's predictions. We will then extend this measure to also include the Mean Squared Error from the Linear Regression task as a way of gauging the overall accuracy of these models.

Constructing a confusion matrix will also be important for evaluating the success of the models. It will calculate and display not only the true positives, but also false positives and will show how the classifier performs on each class (cat, dog). The visual representation makes it easy to see not only when the classifier errs, but also how it errs. This information will undoubtedly give us feedback as we tune the models.

It also may be worthwhile to create a Precision-Recall Curve based on the results of the predictions, as well as find the F-measure. The Cats and Dogs task can be seen as a binary classification task, where each image is either one or the other of the two choices. This makes Precision-Recall and F-measure relevant metrics. The F-measure will provide the 'harmonic mean' of the precision and recall figures, which summarizes this for easier analysis.

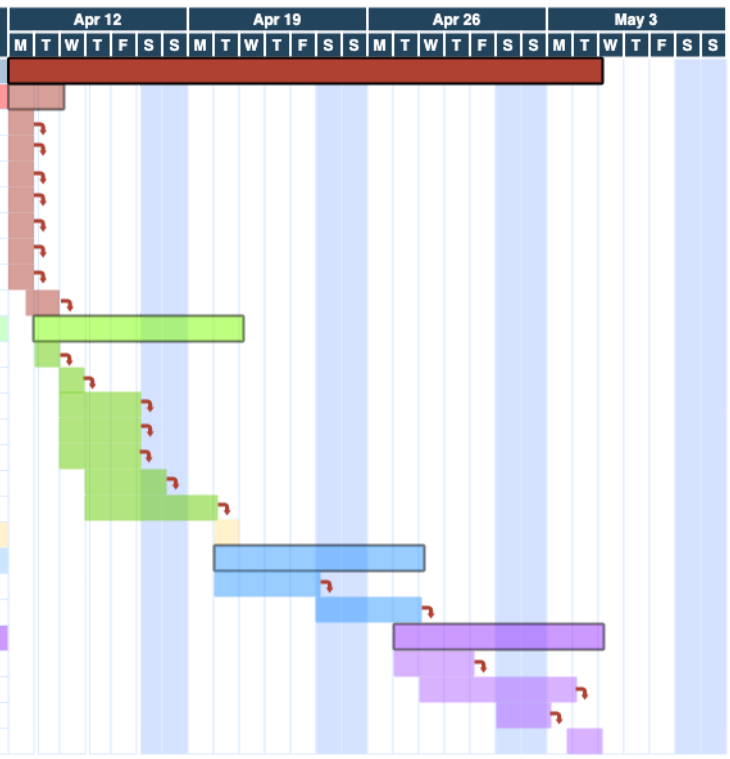
Timeline

Phase 1 tasks are detailed per the assignment on Canvas found here. Tasks required for Phase 2 and Phase 3 will be broken out and described in greater detail at the beginning of that phase and the timeline updated accordingly.

Project conversation and status updates will be communicated through the group's Slack workspace at aml-group2.slack.com. Timeline and status updates will be stored in a Draw.IO diagram on the group's GitHub repository.

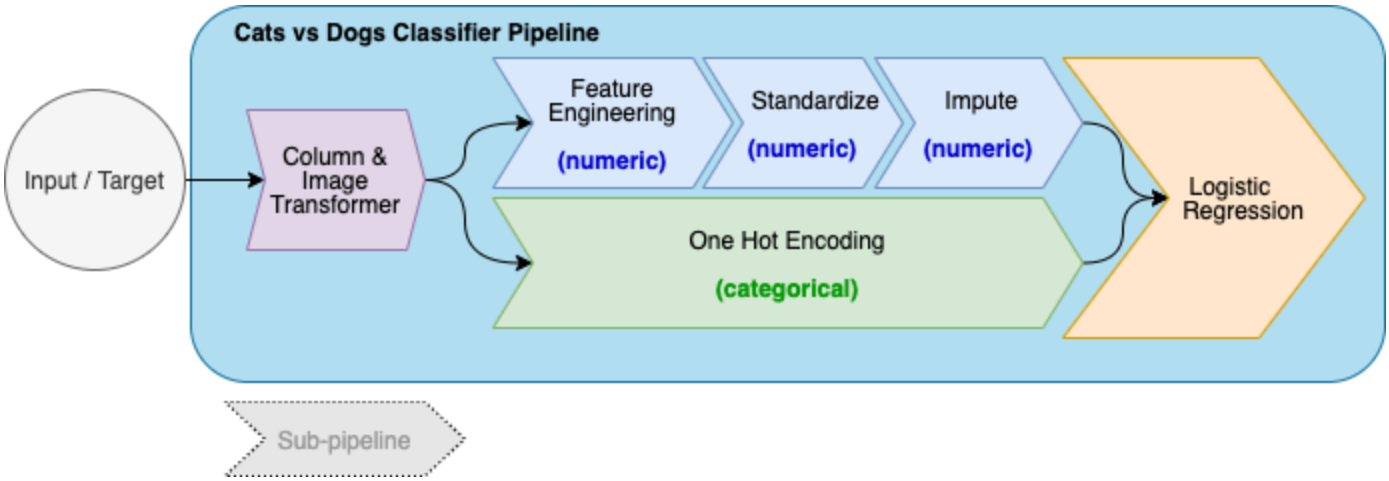
Group 2 - Cats vs Dogs AML Spring 2021 - Project Timeline

	Task Name	Duration	Start	ETA
1	Final Project	23 days	21.04.09	1.06.12
2	Phase 0	4 days	21.04.09	**21.04.13**
3	Divide tasks	0.5 day	21.04.11	21.04.12
4	Write abstract	1 day	21.04.11	21.04.12
5	Describe data	1 day	21.04.11	21.04.12
6	Describe metrics	1 day	21.04.11	21.04.12
7	Describe algorithm	1 day	21.04.11	21.04.12
8	Create timeline	1 day	21.04.11	21.04.12
9	Describe pipeline	1 day	21.04.11	21.04.12
10	Combine & edit final PDF	1 day	21.04.12	21.04.13
11	Phase 1	7 days	21.04.13	**21.04.20**
12	Assemble data	1 day	21.04.13	21.04.13
13	Exploratory data analysis	1 day	21.04.14	21.04.14
14	Feature engineering / pipeline	3 days	21.04.14	21.04.16
15	Hyperparameter tuning / pipeline	3 days	21.04.14	21.04.16
16	SKlearn img classification model	3 days	21.04.14	21.04.16
17	SKlearn regression model	3 days	21.04.15	21.04.17
18	Homegrown logistic regression model (CXE + MSE)	4 days	21.04.15	21.04.19
19	Stretch - logistic regression w/ PyTorch	1 day	21.04.19	21.04.20
20	Phase 2	7 days	21.04.20	**21.04.27**
21	Select 500 imgs w/ cat or dog as main img, label & bound	4 days	21.04.20	21.04.24
22	Baseline PyTorch pipeline for classification + bounding box	3 days	21.04.24	21.04.27
23	Phase 3 (final)	7 days	21.04.27	**21.05.04**
24	Convolutional neural network, single object class/detect	3 days	21.04.27	21.04.30
25	Write report	6 days	21.04.28	21.05.03
26	Video presentation - 2 min	2 days	21.05.01	21.05.02
27	Final edits and post to Canvas	1 day	21.05.04	**21.05.04**



Pipelines

Our initial approach for Phase 1 pipelines will be to create a Classifier pipeline with several sub-pipelines as shown below.



The first subpipe will be a column and image transformer. This subpipe will contain steps to modify data from the provided CSV metadata and process image transformations (scaling, reduction of color for example, if we

determine those transformations to be relevant). Information obtained from these transformations will feed into one of two subpipes depending on the nature of the information (categorical or numeric).

One sub-pipeline for feature engineering based on the output of exploratory data analysis and transformation will focus on numeric data and we'll refer to it as the 'Feature Engineering' subpipe, shown in blue in the diagram above. This subpipe will include steps for handling numeric data, including standardization and imputing of absent values, if required.

For categorical (non-numeric) data, a subpipe or pipeline steps for One Hot Encoding may be added, depending on need determined during exploratory data analysis and transformation.

Both numeric (Feature Engineering) and categorical (OHE) subpipes will feed a Regression subpipe. This subpipe will include steps to determine optimal hyperparameters, and will contain steps to fit/train the SKLearn regression 'baseline' model. Linear and logistic regression steps to predict bounding boxes and then cat/dog subject will be within this pipeline. The Regression subpipe will be modified to use the Homegrown regression model after it is created.

As needed, other pipeline steps (or subpipes) may be inserted to create visualizations and/or data frames, and these steps or subpipes will be documented at the end of each project phase.

Team Members and Project Roles

Ben Perkins - Metrics and evaluation, general ML engineering

Lauren Madar - GitHub repo setup, project timeline, EDA/feature engineering, pipelines, documentation (such as visualizations, video editing, recording tools and methods used, and submitting phase assignments)

Mangesh Walimbe - ML Algorithms and models

Samin Barghan - EDA, hyperparameter tuning and general ML engineering