# PREDICT THE NUMBER OF DOGS IN AN IMAGE

**Project Proposal**

**COMP 4613 WIO1 – Artificial Intelligence**

**EXCELLENCE SOWUNMI**

**100158476**

**March 22, 2023**

## Introduction

I am suggesting creating a model that predicts the number of Dods in an image. I am also proposing labelling each image with the precise number of Dogs in it. The purpose of this project is to demonstrate my ability to identify an image (photo) classification or regression problem, locate an appropriate image dataset, prepare that data, and create and test deep learning models. Success for the project is based on being able to make a functioning accurate prediction model that Uses deep learning approaches with several hidden layers, various epochs, and batches.

## Approach

I will be developing a series of models with a baseline model that will set a minimal model performance standard and a model architecture that can serve as the foundation for further research and development. I will be getting my data from Kaggle and layout the images of dogs, selecting a standard image size, and placing pre-processed images in regular directories. The CNN model layers will use He weight initialization and ReLU activation functions.

### Method of model evaluation

I will be evaluating the models by calculating and plotting the cross-entropy loss and Accuracy on both the test and train datasets. Prediction graphs will be created using the test set to evaluate the model's potential performance. Several parameters will be provided to the graphs to make them different.

When fitting a model, we will consider loss and accuracy and aim to employ strategies that will allow us to reduce loss, increase accuracy, and avoid overfitting.

## Data Collection and Preparation

### Data source

A 2013 Kaggle machine learning competition utilized the dogs vs. cats dataset. It is made up of images of dogs and cats that were provided as a subset of images from a much bigger dataset of 3 million manually annotated images. The dataset was initially employed as a CAPTCHA (or Completely Automated Public Turing test to tell Computers and Humans Apart). For a 70% subsample of the test dataset, Pierre Sermanet attained a classification accuracy of 98.914%. The dataset is simple to comprehend and short enough to fit in memory, making it a useful "hello world" or "getting started" computer vision dataset for beginners.

### Data preparation

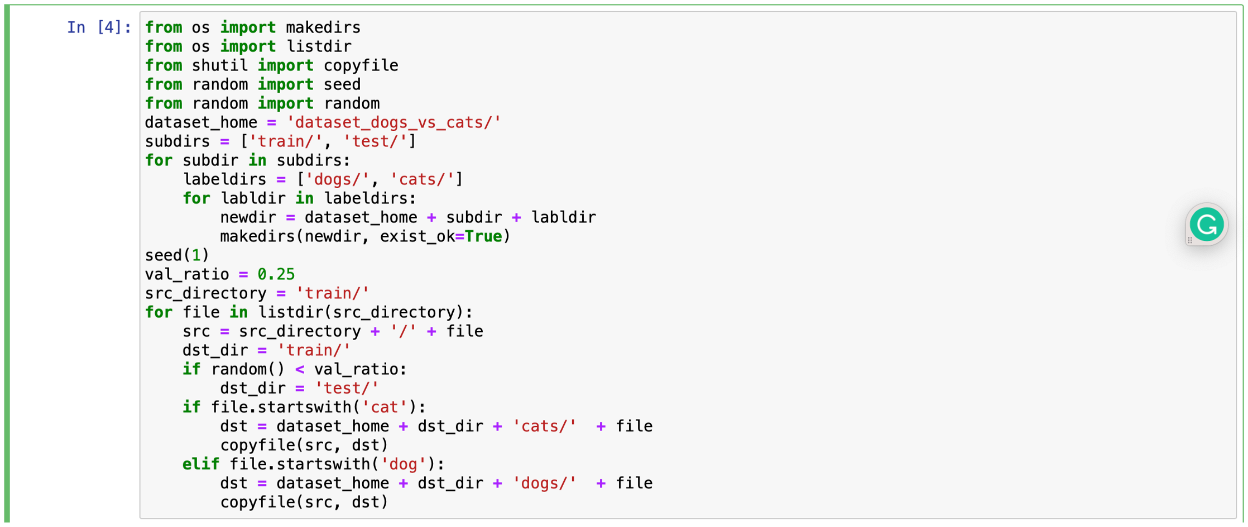
A Kaggle account is necessary to get the dataset for free from the Kaggle website. Visit the Dogs vs. Cats Data page and select "Download All" to obtain the dataset. Create a folder called "train/" and put 25, 000.jpg files of dogs and cats within it after unzipping the file. Use the flow from the directory() API and the Keras ImageDataGenerator class to plot images of dogs and cats. This API prefers that the data be separated into distinct train/ and test/ folders, with a subfolder for each class under each directory. Images are arranged in subdirectories, and a script is built to duplicate the dataset with this preferred structure, which are the two most crucial features in this article. In order to build the dog/ and cat/ subdirectories for both the train/ and test/ directories, the script uses the makedirs() method to create directories in Python. Once all image files in the dataset have been listed, they are all copied, according to their filenames, into the dogs/ or cats/ subdirectories.

**Testing and training Data models**

In order to create a test dataset, it finally chooses 25 percent of the photographs at random.

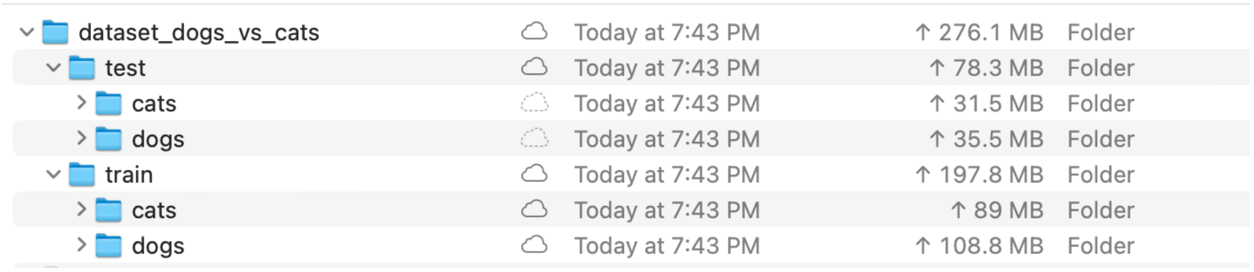
The pseudorandom number generator's seed is fixed to ensure that the data is split evenly each time the function is executed, which achieves consistency in this.

This is the entire code example, which presupposes that the photos are unzipped in the train/current working directory from the train.zip file that was downloaded.



```
In [4]: from os import makedirs
        from os import listdir
        from shutil import copyfile
        from random import seed
        from random import random
        dataset_home = 'dataset_dogs_vs_cats/'
        subdirs = ['train/', 'test/']
        for subdir in subdirs:
            labeldirs = ['dogs/', 'cats/']
            for lablldir in labeldirs:
                newdir = dataset_home + subdir + lablldir
                makedirs(newdir, exist_ok=True)
        seed(1)
        val_ratio = 0.25
        src_directory = 'train/'
        for file in listdir(src_directory):
            src = src_directory + '/' + file
            dst_dir = 'train/'
            if random() < val_ratio:
                dst_dir = 'test/'
            if file.startswith('cat'):
                dst = dataset_home + dst_dir + 'cats/'  + file
                copyfile(src, dst)
            elif file.startswith('dog'):
                dst = dataset_home + dst_dir + 'dogs/'  + file
                copyfile(src, dst)
```

Once the example has been run, you will get new dataset dogs vs cats/ directory with the intended train/ and val/ subfolders and additional dogs/ can cats/ subdirectories.



```
dataset_dogs_vs_cats    Today at 7:43 PM   ↑ 276.1 MB   Folder
  test                  Today at 7:43 PM   ↑ 78.3 MB    Folder
    cats                Today at 7:43 PM   ↑ 31.5 MB    Folder
    dogs                Today at 7:43 PM   ↑ 35.5 MB    Folder
  train                 Today at 7:43 PM   ↑ 197.8 MB   Folder
    cats                Today at 7:43 PM   ↑ 89 MB      Folder
    dogs                Today at 7:43 PM   ↑ 108.8 MB   Folder
```

## Machine Learning Algorithms and software used.

### Learning Parameters

For this project, SGD was chosen as the algorithm due to its improved accuracy and reduced loss. A default set of learning parameters was used.

This project was managed and created using Google Collaboratory. Because it is a prebuilt and ready-to-run environment, it was used.

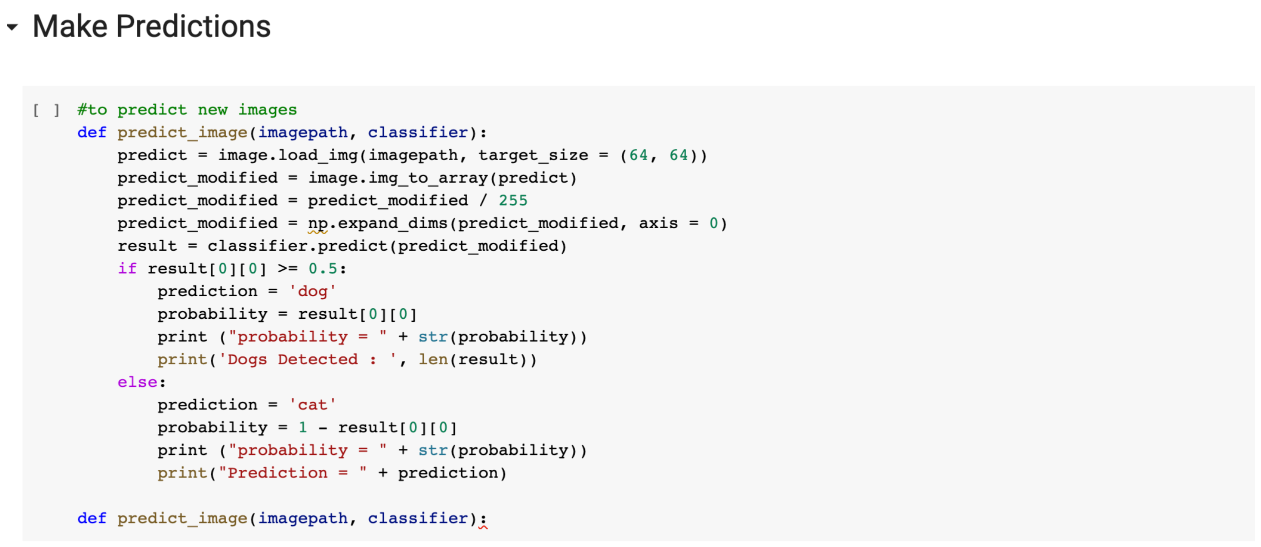
### Network Architecture

A Convolutional Neural Network served as the foundation of the utilized architecture. VGG blocks were included in the updated architectures, along with dropout regularisation, changing the learning rate, and applying a different optimizer. These methods were employed because they often produce positive outcomes while requiring little computing effort.

## Result Evaluation

### Hypothesis Test

In using the model to predict a new image we can identify if a Dog is in the image or not with an accuracy of 80%. If the prediction is > 50% it should be a Dog if not a Cat.



Make Predictions

```
[ ] #to predict new images
    def predict_image(imagepath, classifier):
        predict = image.load_img(imagepath, target_size = (64, 64))
        predict_modified = image.img_to_array(predict)
        predict_modified = predict_modified / 255
        predict_modified = np.expand_dims(predict_modified, axis = 0)
        result = classifier.predict(predict_modified)
        if result[0][0] >= 0.5:
            prediction = 'dog'
            probability = result[0][0]
            print ("probability = " + str(probability))
            print('Dogs Detected : ', len(result))
        else:
            prediction = 'cat'
            probability = 1 - result[0][0]
            print ("probability = " + str(probability))
            print("Prediction = " + prediction)

    def predict_image(imagepath, classifier):
```

## Conclusion

It takes a while to create several models for architecture. Although I only tested a limited number of them, some of which are probably superior, the baseline CNN with dropout model designs was the best of the settings evaluated. The code I wrote will make testing different architecture types and methodologies faster and easier, even though I was unable to reach the desired 100% validation accuracy.

## References

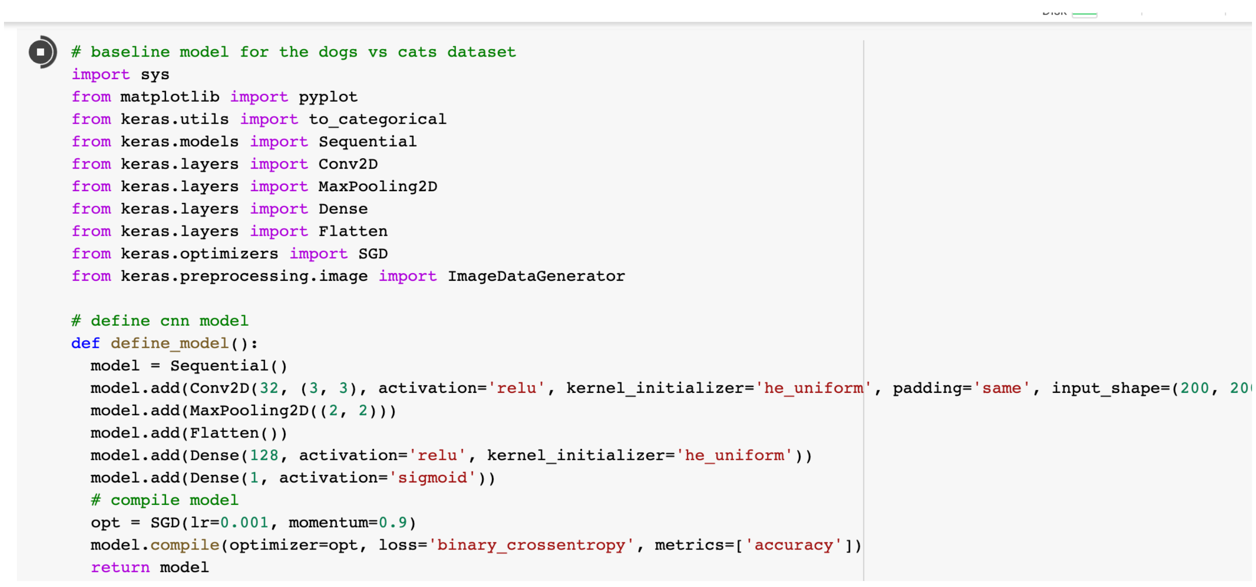
Bibliography

Brownlee, J. (2021, December 7). *How to classify photos of dogs and cats (with 97% accuracy)*. MachineLearningMastery.com. Retrieved March 25, 2023, from https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/

Shah, K. (2021, June 1). *Beginner-friendly Project- Cat and Dog classification using CNN*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/beginner-friendly-project-cat-and-dog-classification-using-cnn/

## Appendices

### Code result



```python
# baseline model for the dogs vs cats dataset
import sys
from matplotlib import pyplot
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Dense
from keras.layers import Flatten
from keras.optimizers import SGD
from keras.preprocessing.image import ImageDataGenerator

# define cnn model
def define_model():
  model = Sequential()
  model.add(Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same', input_shape=(200, 20
  model.add(MaxPooling2D((2, 2)))
  model.add(Flatten())
  model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
  model.add(Dense(1, activation='sigmoid'))
  # compile model
  opt = SGD(lr=0.001, momentum=0.9)
  model.compile(optimizer=opt, loss='binary_crossentropy', metrics=['accuracy'])
  return model
```

**Note: I was unable to run most of the code due to a malfunction of my kennel but I did get my reference for this source. [https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/](https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/) . I am sorry for the inconvenience. Thank you Dr. Mahbub for your understanding.**