How I Solved DOG v CAT classification Using AI:

AI libraries such as TensorFlow and Keras can be used to build Convolutional Neural Networks (CNNs) for the dog vs cat image classification problem.
So, firstly I imported Tensorflow and Keras libraries.

We know we are going to use pandas and numpy we first import them

I downloaded the dataset and uploaded it in the drive from where I mounted it into google colab.
We then read csv files using pd.read_csv
Then we have to convert all the data into float32 using .astype('float32')
Then we have to convert it into numpy array
Now we have to reshape the datasets and preprocess it by normalizing to get values on the scale 0-1.

The next step is building the Convolutional neural network using a deep learning framework such as Tensorflow,Keras with which we can define and train a CNN architecture.
CNN typically consists of convolutional layers, pooling layers and fully connected layers

To take any random image for training purposes we use random.randint(),for that we have to import random again at the top where we imported pandas for clarity of code.

Sequential model means that the layers are going to be stacked up in the sequence.
We know that,first convolutional layer then maxpooling layer then another convolutional layer then next filling layer then we have a couple of fully connected layers and that's how our convolutional neural network is made.

Inside this sequential function we just write our layers,so the first parameter of conv layer Are the number of filters that we want to use followed by the next parameter size of filter used.Now the number of channel part is automatically implemented we just need to mention the height and width of the filter.Then we need to mention the activation function that we have to use. We used 'relu' here.
To add next layer which is maxpooling layer,for maxpooling ,we just need to mention just one parameter that is filter size,we can also mention the stride.

Then again we will be using conv layer followed by maxpooling layer.

Now we will be flattening this and we will be using dense layers which means fully connected layers so let's have our first fully connected layer has 64 neurons so this first parameter indicates the number of neurons that we want to keep in that layer then the second parameter is the activation function that we want to use again.

Then let's say this is our final fully connected output layer and this layer must have the same number of neurons as out output class but for binary classification we only need one output neuron and thus we will be keeping one here and activation function that we will be using is 'sigmoid' because it is a binary classification.

Now we add cost function as in all AI algorithms we get to see one,
Compile function adds back propagation and loss, so the algorithm parameters optimizer and the metrics so here we can specify the type of loss that we want to use as we are implementing binary classification, we will be using binary cross entropy loss.And the optimizer that will be used is 'Adam'.

Now we just need to pass our input data and we need to fit this input data training of this input data is done by fit which is model dot fit.We will just pass training data and then we pass number of epochs for which we want to train our model and also the batch size that we want to take.

Lastly, using 'evaluate' we test data and make predictions using 'predict' function.
If the value predicted is less than .5 it is a dog and if it is more than .5 it is a cat.