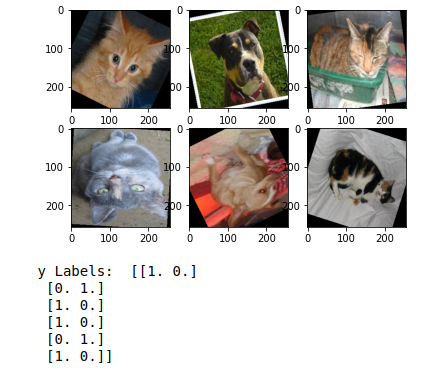
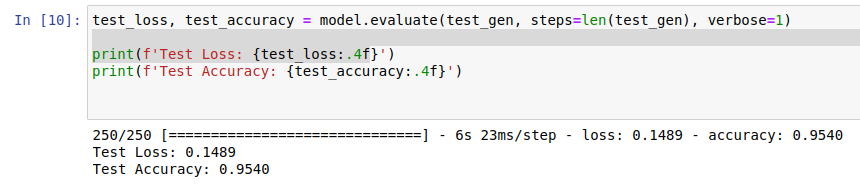
CNN for Image Classification

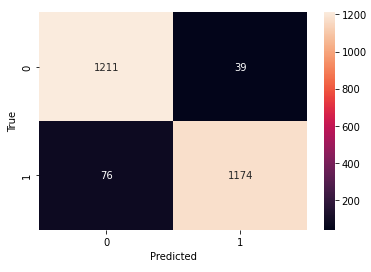
1. Used my Lab server. Created new environment. Opened new Jupyter notebook. Install tensorflow in this environment. Imported or installed required libraries as needed.
2. Downloaded the data set from Kaggle and saved in a folder. Dataset accessed from the save folder and splitted into validation, training and test.
   1. Using **‘splitfolders’**, the main dataset is divided into 70% training, 20% validation and 10% test set.
3. Initiating Data generator
   1. Two data generators have been set up: one for the training set and another for the validation and test sets.
   2. Four distinct augmentations are applied to the dataset: rescaling, a 30-degree rotation, and both horizontal and vertical flipping.
   3. Fill mode is set to "constant" to fill in any gaps or missing pixel values that result from augmentation.
   4. The training, validation, and testing datasets are loaded into the generator with a batch size of 10. I attempted training with batch sizes of 50, 40, 30, and 20, but had to reduce it to 10 due to encountering a ResourceExhaustedError.
   5. Shuffling is turned off for the test dataset since we want to generate results from a single execution.
   6. Sample Data augmentation



1. An alternative approach is to create a custom ImageLoader that carries out a similar function, but applies data augmentation dynamically to the dataset during training.
2. Initiating the ResNet50 model
   1. For a pretrained model, ‘**weights**’ is set to ’**imagenet**’ and top layer is customized
   2. Activation is set to **‘sigmoid’** for binary classification.
3. Hyperparameter
   1. Optimizer = Adam
   2. Learning rate = 1e^-4
   3. Loss function = binary crossentropy
   4. Epoch = 10
   5. Batch size =10
   6. So far, 350 steps per epoch have yielded the best results. However, due to certain limitations and time constraints, we couldn't increase the epoch or step size for improved accuracy.
4. Test results



1. Confusion matrix



1. Plot

