# DEEP LEARNING ASSIGNMENT 2

DETAILS;

NAME: S. SANKEERTHAN

STUDENT ID:111101112

DOCUMENT: DETAILED REPORT OF SOLUTION DURING COMPETITION RUN

**PROBLEM STATEMENT:**

You are not given any training dataset. But, you need to build a deep learning model and predict all the test images for the hat-ness test, i.e., if image contains a person with a hat or not. You know what to do, right?

Our policy while constructing the test-set was:

1. Photos of real people, i.e., absolutely no cartoons, nor apparitions.
2. Photos may contain multiple people with their heads (with or without hat) visible.
3. If a person's head is covered either with a scarf, or bandana, hijab, beanie, etc, might as well be considered person with hat during evaluation.

Submissions are evaluated based on Categorization accuracy. Higher the better.

Please note, a sample with id should contain a corresponding class value that belongs to this set: {'Hat', 'No Hat'}.

For each id corresponding to an image file in the test-set/ directory, you must predict a class label for the class variable.

**PROBLEM EVALUATION:**

The problem requires to code and build a deep learning model that is going to classify whether a person is wearing a hat or not, here a hat is not a clear-cut term, and the person can be wearing any headgear such as a scarf, or bandana, hijab, beanie so even their would be considered as a hat. We need to create a CSV file, that will classify the image IDs as hat or no hat, and the categorization accuracy.

**Test-DataSet**

1490 images are contained.

-Multiple images with hat or no hat

-People with multiple headgear beanie,bandana,scarf, that should also be considered as a hat.

-Multiple people in a single image.

**FIRST ITERATION:**

import os

import numpy as np

import pandas as pd

from tensorflow.keras.preprocessing import image

from tensorflow.keras.applications.inception\_v3 import InceptionV3, preprocess\_input

from tensorflow.keras.models import Model

from tensorflow.keras.layers import GlobalAveragePooling2D, Dense

from tensorflow.keras.optimizers import Adam

# Load InceptionV3 model without top layers

base\_model = InceptionV3(weights='imagenet', include\_top=False, input\_shape=(224, 224, 3))

# Freeze the pre-trained layers

for layer in base\_model.layers:

layer.trainable = False

# Add new layers for binary classification

x = base\_model.output

x = GlobalAveragePooling2D()(x)

x = Dense(128, activation='relu')(x)

predictions = Dense(1, activation='sigmoid')(x)

# Create the model

model = Model(inputs=base\_model.input, outputs=predictions)

# Compile the model

optimizer = Adam(learning\_rate=0.001)

model.compile(optimizer=optimizer, loss='binary\_crossentropy', metrics=['accuracy'])

# Path to the test set folder containing images

test\_set\_folder = '/Users/sankeerthanleo/Desktop/test\_set/test\_set'

# Make predictions on test images

predictions = []

for filename in os.listdir(test\_set\_folder):

if filename.endswith('.jpg'):

img\_path = os.path.join(test\_set\_folder, filename)

img = image.load\_img(img\_path, target\_size=(224, 224))

img\_arr = image.img\_to\_array(img)

img\_arr = preprocess\_input(img\_array)

img\_arr= np.expand\_dims(img\_array, axis=0)

prediction = model.predict(img\_array)

predicted\_class = 'Hat' if prediction >= 0.5 else 'No Hat'

photo\_id = os.path.splitext(filename)[0]

predictions.append({'id': photo\_id, 'class': predicted\_class})

# Create submission file

submission\_df = pd.DataFrame(predictions, columns=['id', 'class'])

submission\_df.to\_csv('/Users/sankeerthanleo/Desktop/test\_set/test\_set', index=False)

Here, I have imported various packages and modules, that are required to train the dataset, having imported tensor flow library, to build and train neural networks, and the InceptionV3 model and preprocessing function from Keras applications. InceptionV3 is a pre-trained convolutional neural network architecture for image classification,and also importing of Pandas library and then we have to load pre-trained InceptionV3 model.

The model should be initialized with weights pre-trained on the ImageNet dataset, which is a large dataset commonly used for image classification tasks. input\_shape specifies the input shape of the images, expected by the model. InceptionV3 expects input images to be 224x224 pixels with 3 channels(RGB).

Then we freeze the pre-trained layers, that loops through the layers in base\_model, It sets the trainable attribute of each layer to False. This prevents the weights of these layers from being updated during training, effectively freezing them.

Then we add new layers for binary classification, that create a new Keras sequential model, adding the pre-trained InceptionV3 model(base\_model) as the first layer, then adding a global average pooling layer, that reduced the spatial dimensions of the feature maps from the convolutional base, then we finally have to add, a dense layer with a single unit and sigmoid activation function for binary classification(Hat and No Hat solution).

Then we can run to compiling the model, for compiling we use the Adam optimizer, binary cross-entropy loss function, and accuracy metrics, we use Adam optimizer to minimize the loss function during training of neural networks. Which specifies the optimization algorithm, loss function, and evaluation metric used during training. Then we finally put a path to test set that basically specifies the path to the folder containing test images.the model gets trained to make predictions on unseen data, we have to direct it to the path where the test images folder exists.

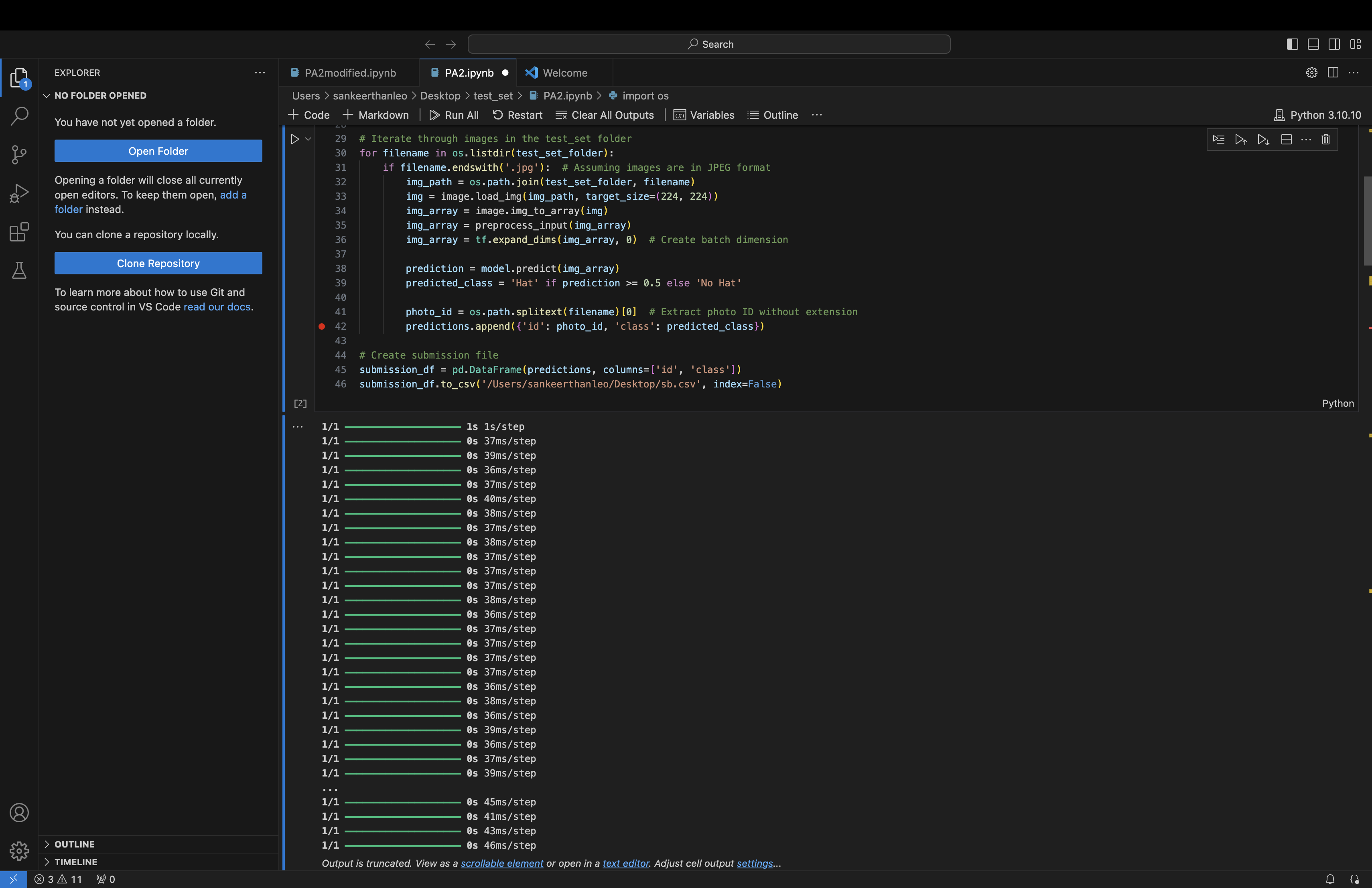
We then initialize an empty list predictions that is going to store the predictions, it iterates through images in the specified test set folder, it then loads each image, preprocesses it, and prepares it for input to the model, it uses the trained model to make predictions on each image.

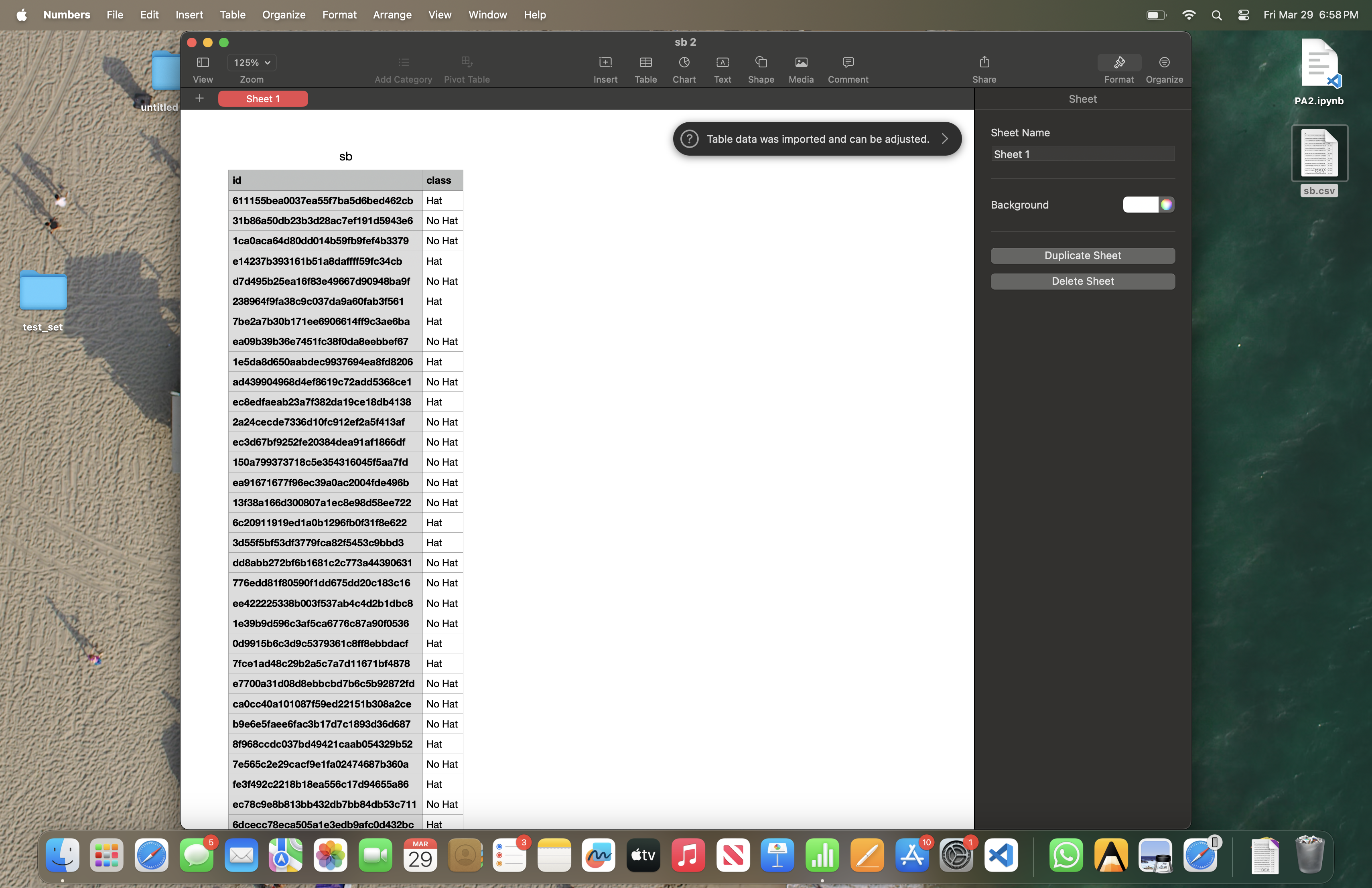
It then assigns the predicted class label based on whether the predicted probability is greater than or equal to 0.5, it then appends the photo ID and predicted class to the predictions list.

We finally create a submission file, that converts the list of predictions into a pandas Dataframe, we convert the data frame into a CSV file, containing the photo IDs and predicted classes, which are labelled hat or no hat.

The submission file can be used for further analysis or evaluation of the model’s performance.

**OUTPUT:**





**Accuracy in Kaggle:**

**Test 1: 56%**

**Test 2:59%**

**Improving Accuracy:**

We can further improve accuracy in an image classification problem like this by involving multiple strategies, like:

1.Data Augmentation: Increasing diversity of training dataset by applying various transformations such as rotation, shifting, scaling ,and flipping to the images.

2.Transfer Learning: Utilize pre-trained models such as InceptionV3 and fine tune them on specific dataset, This allows model to leverage features learned from large dataset.

3.Hyperparameter tuning: Experiment with different parameters such as learning rate, batch size, optimizer,and network architecture to find the optimal combination that improves performance.

4.Regularization: Applying techniques like dropout and L2 regularization to prevent overfitting.