CISC 867 Project 1: Leaf Classification

**BY**

Hanan Fared Mohamed Omara 20398559

**Supervisor**

DR. Hazem Abbas

Eng. Asif Mahfuz

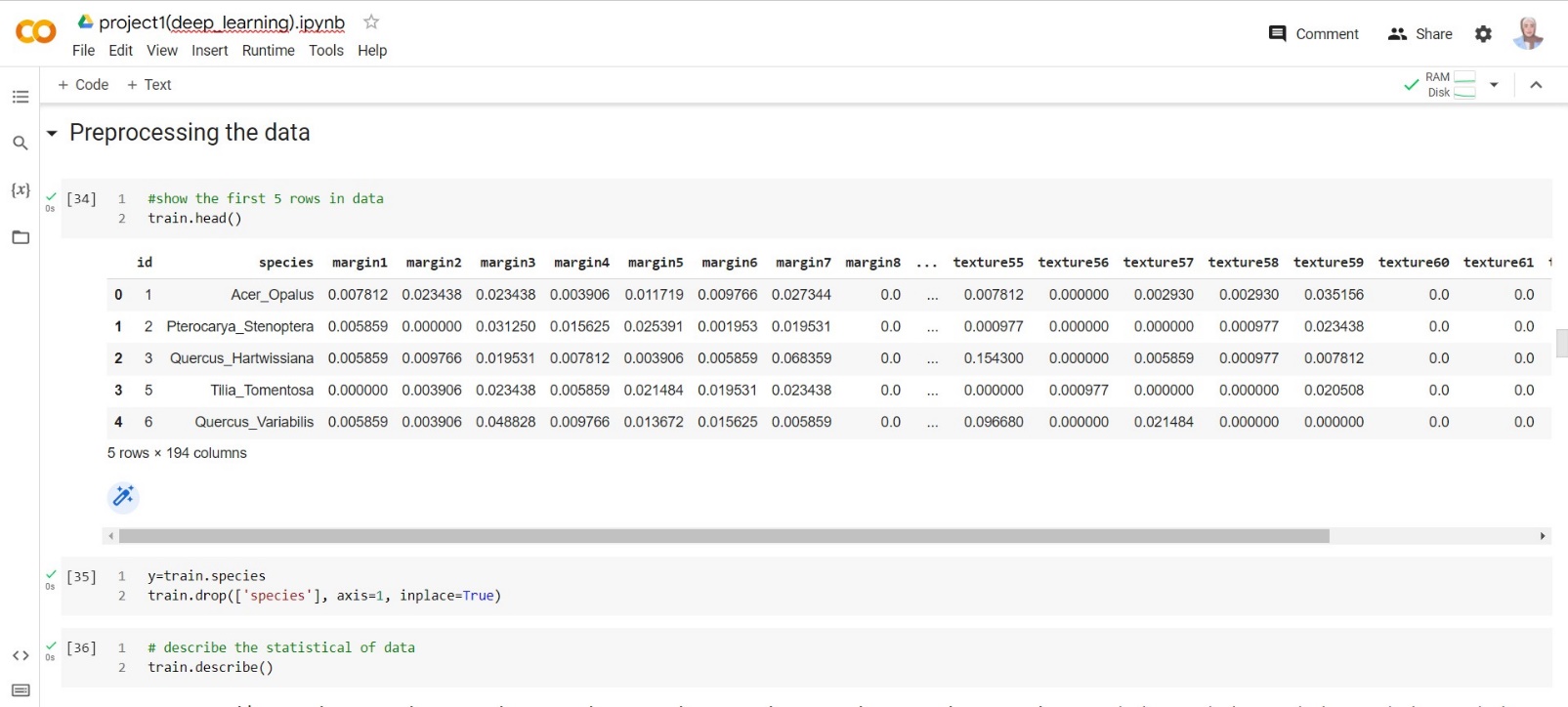
## Part I: Data PreparationData Loading

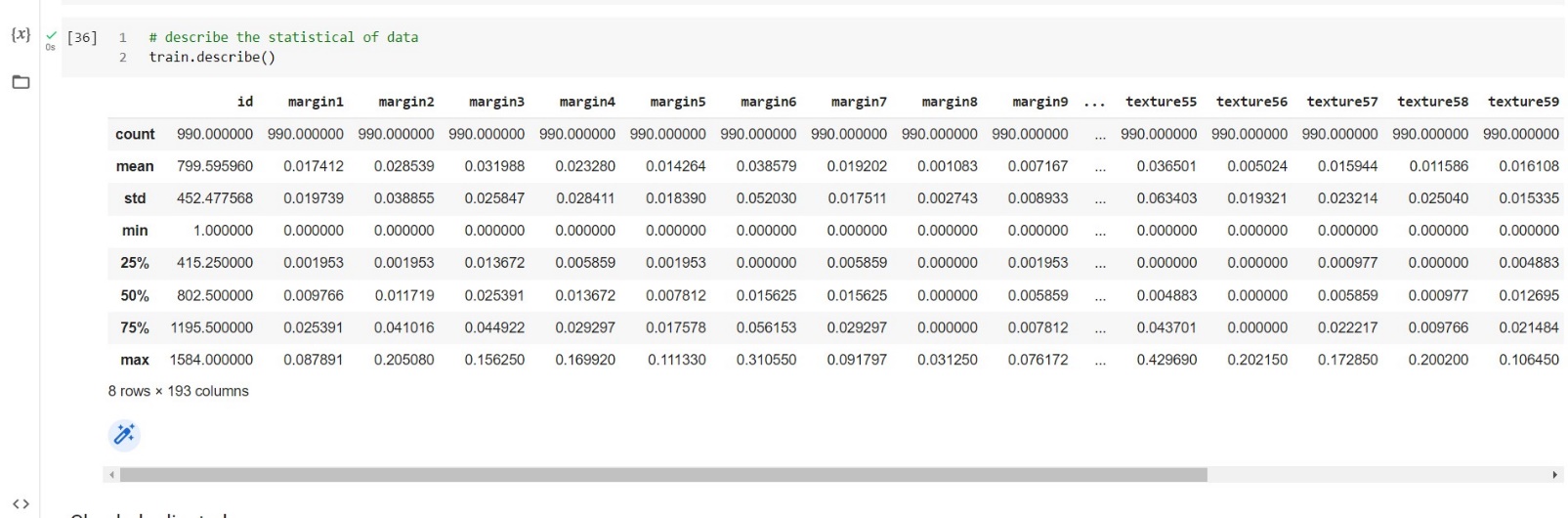
### Load data

Import different libraries and Load data from csv file to the work as a Data Frame using pandas.



### View more information about data





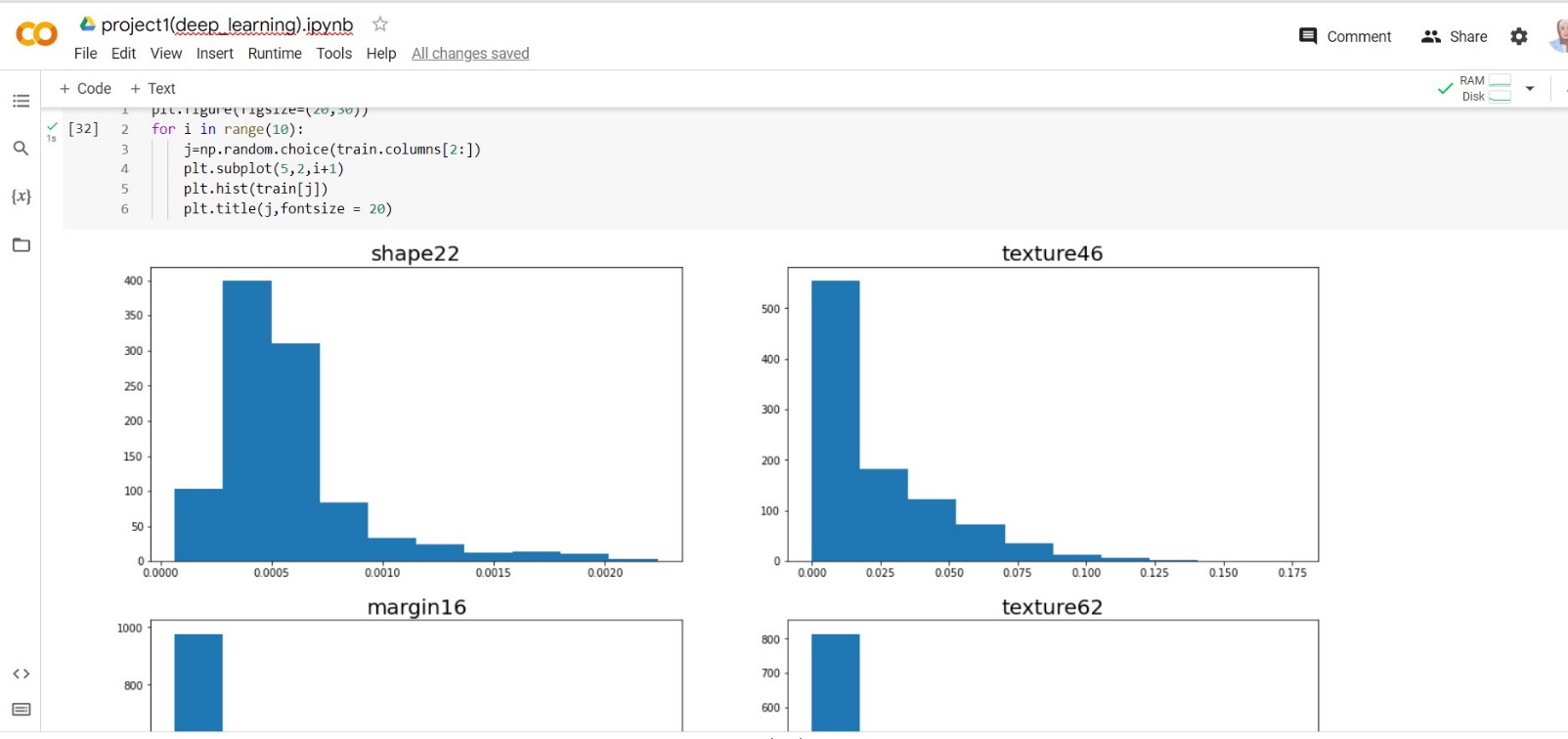
## Visualize Data

In these step we need to view more plots about our data to figure out more information about data set we work with

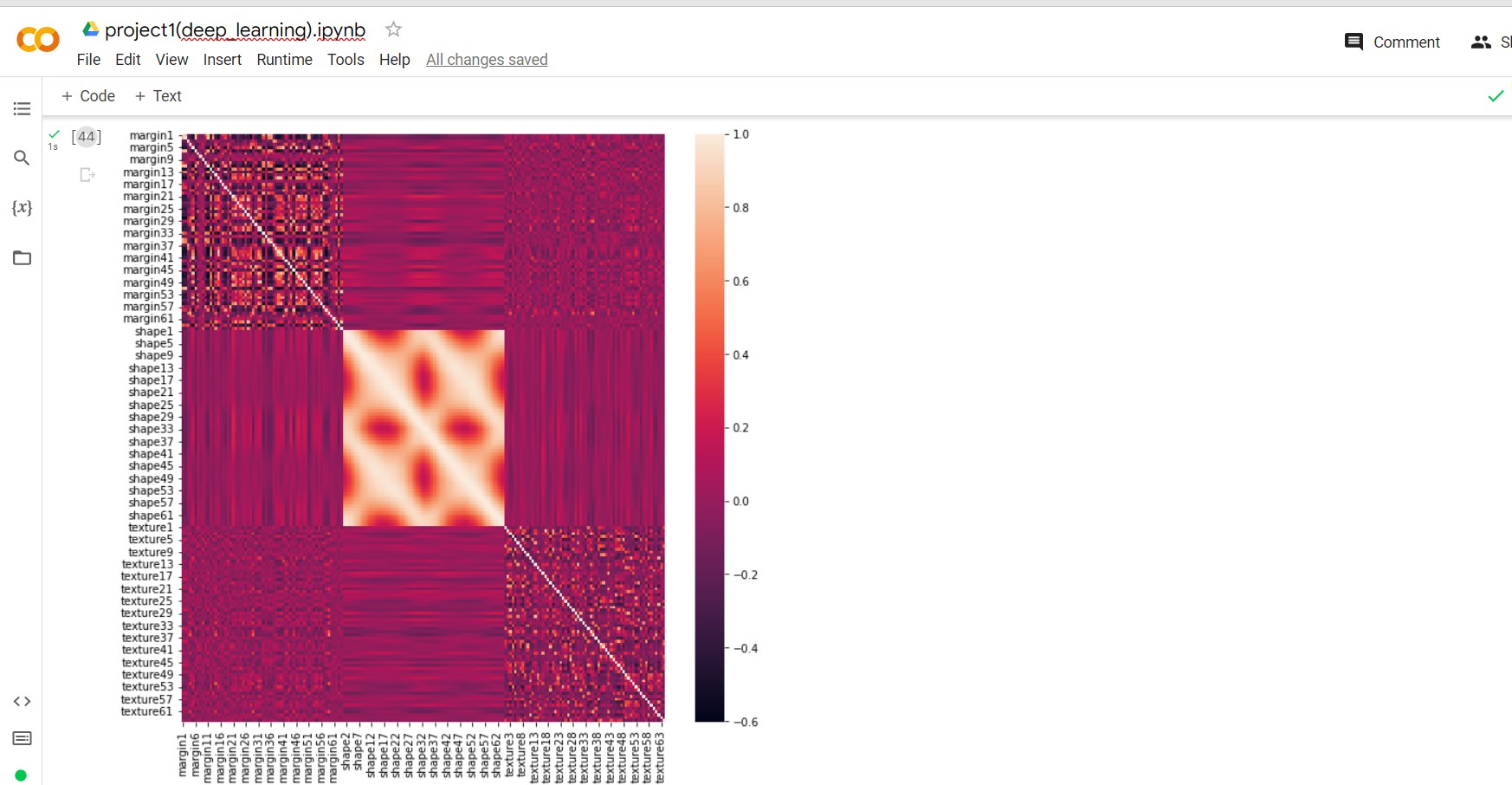
### View the plot of all data



### View different features of data



### View correlation between all features



## Preprocessing Data

### Check null

There is no null in our dataset.

### Check duplicated

There is zero duplicated rows

### Check unique

### Encoding label

As the label includes string data so we encoded it to be numeric and can be input to model to train

### Standardization

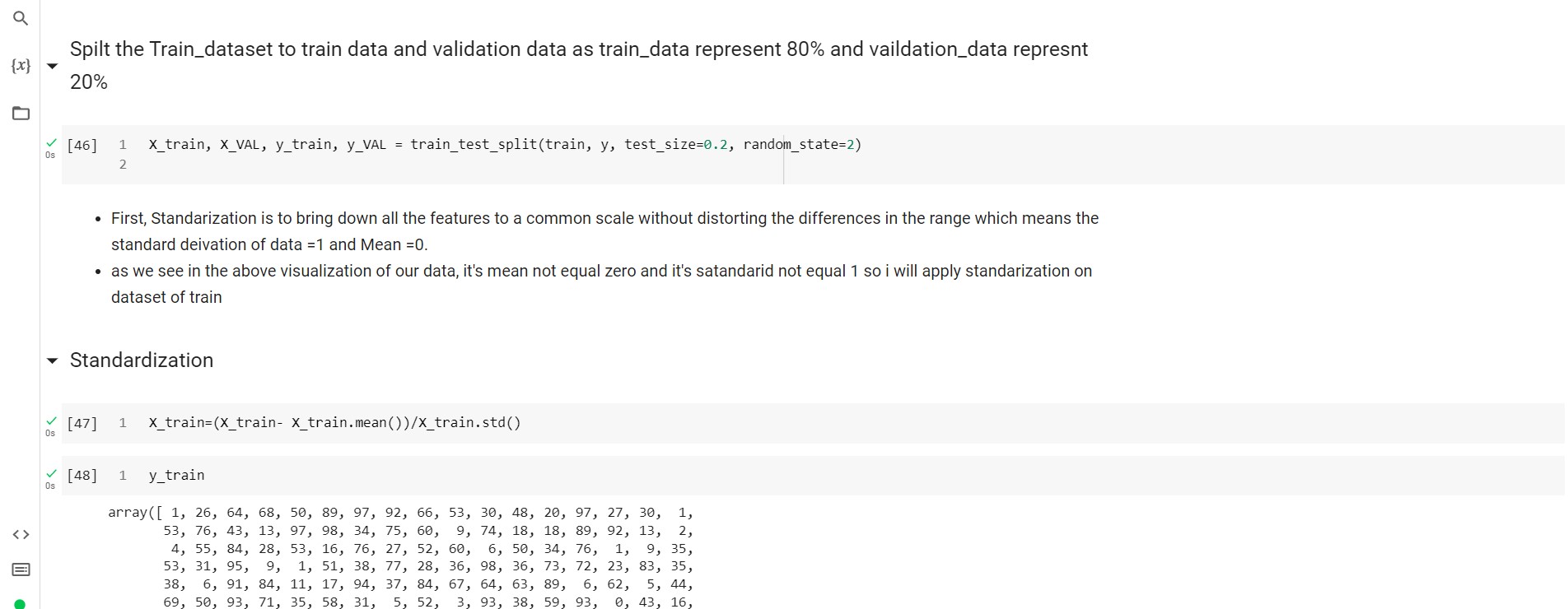
We noetic that our data doesn’t have mean=0 or standard deviation=1 so,

We need to apply standardization on train and validation.

### Split data

Now we need to split our data set into train and test as train percentage=80% and

20%for test(validation)



**Part II: Training a neural network**

## 2.1 Create the Model

Now we will start to create our model as implement a 3-layer MLP model (one input layer, one hidden layer with tanh activation and one output layer) which will be used to classify the data in Part I. then train model at our data set to find what is the most suitable when tune our hyper parameters(

Batch size: Number of examples per training iteration

Hidden size: Try using different number of hidden nodes in your model

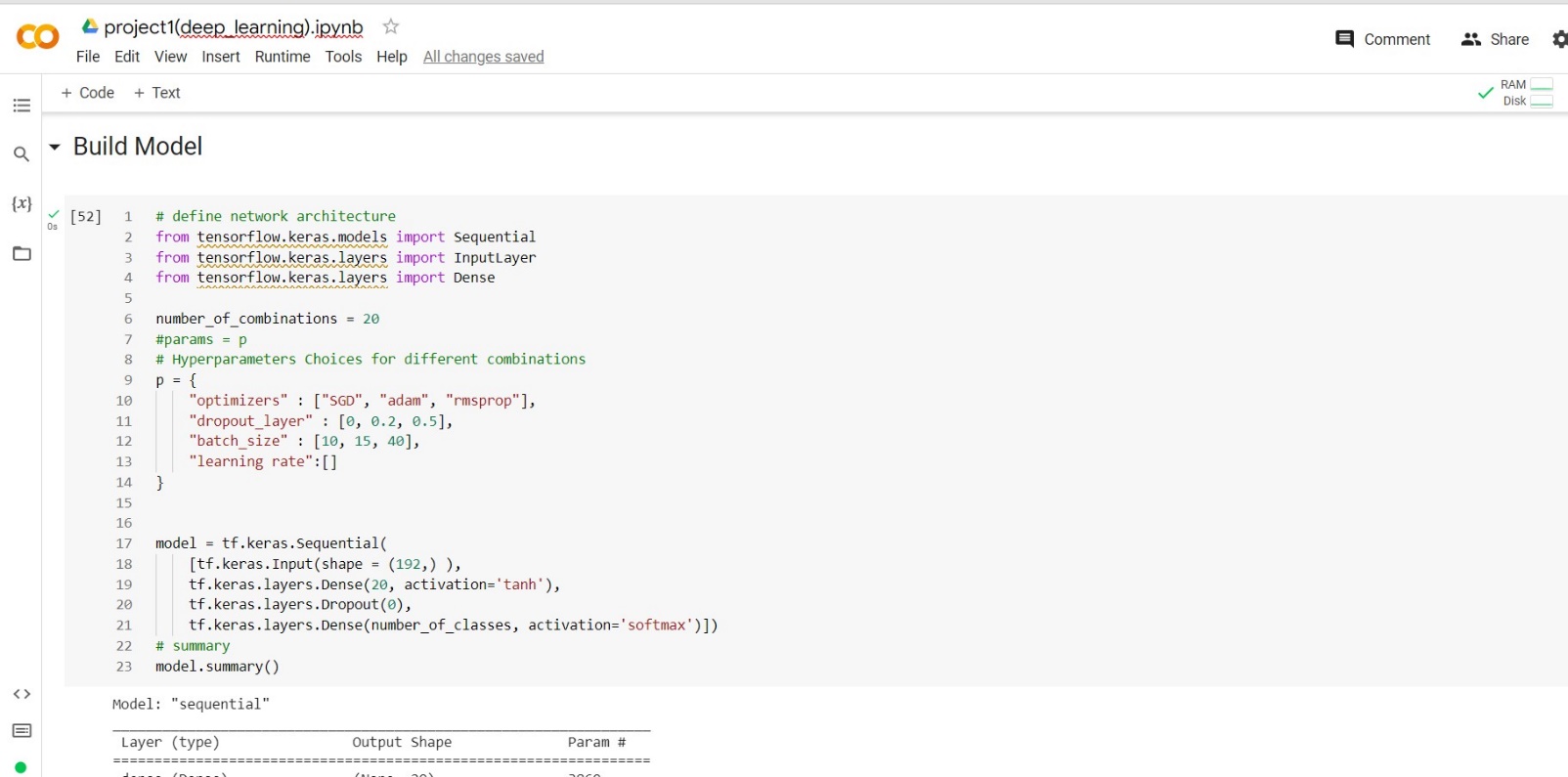
and compare the performances.

Dropout is an effective strategy to defend against overfitting. Adding a dropout layer after

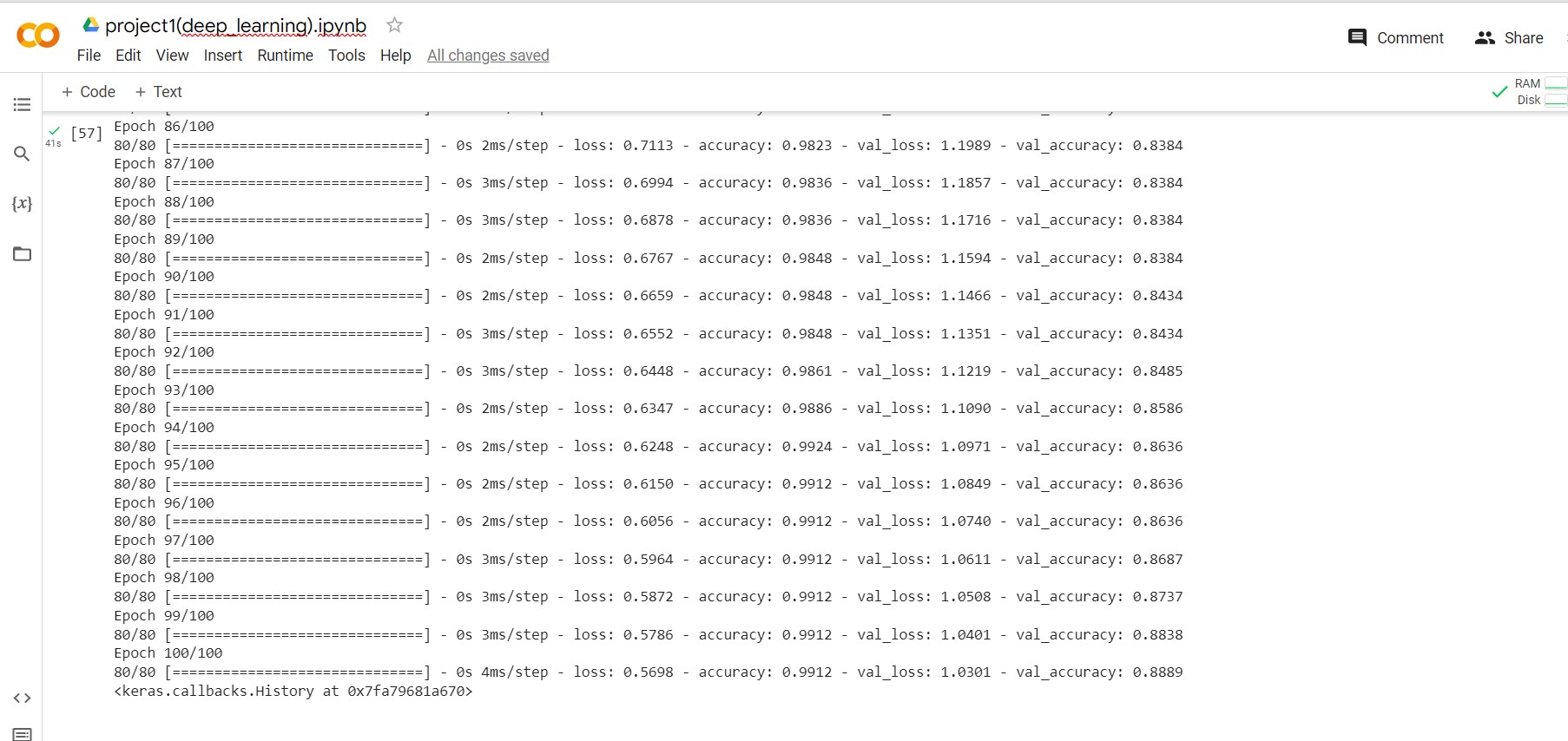
Learning rate, Learning rate scheduler: Learning rate is key hyperparameter in

model training, and you can gradually decreasing the learning rate to further improve your model.

Optimizer: Try using different optimizers such as SGD, Adam, RMSProp.)



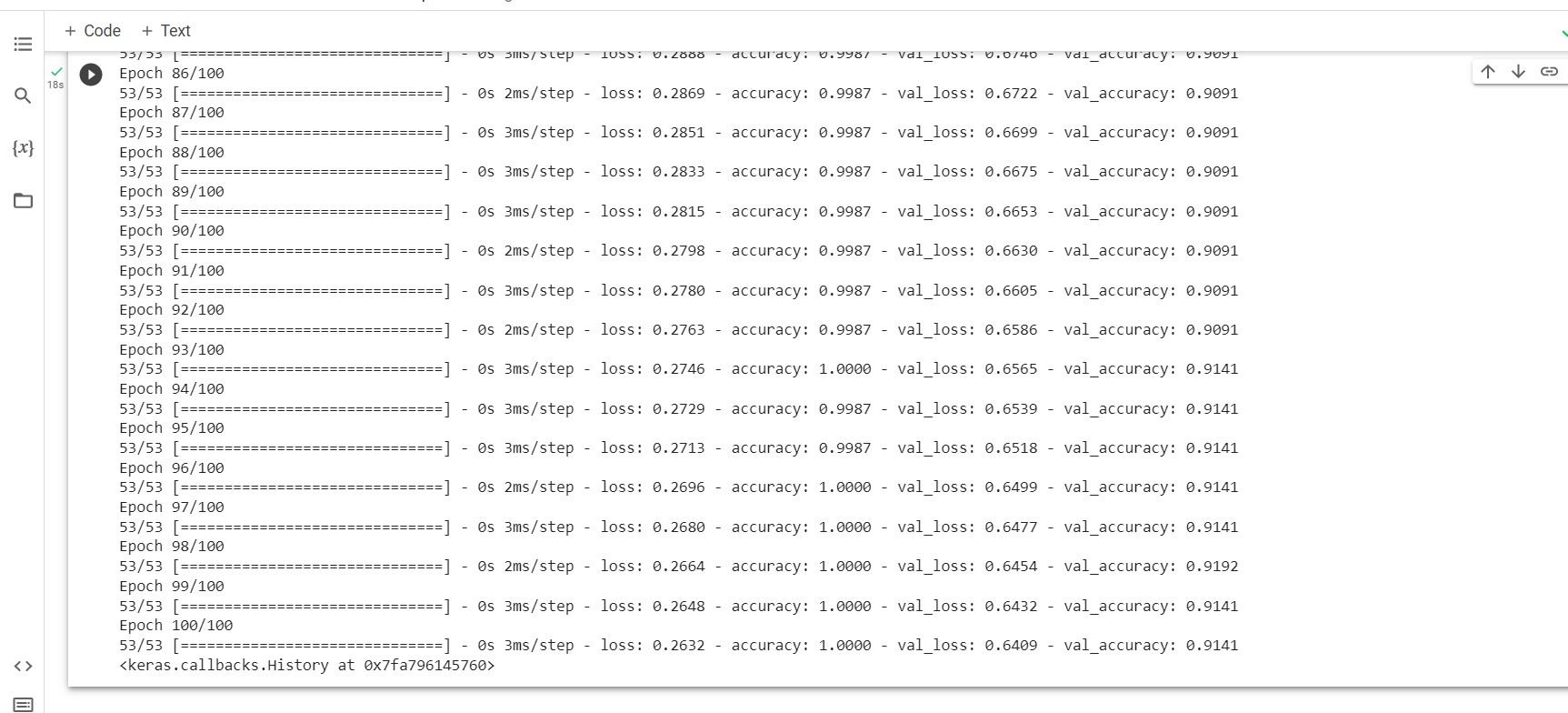
1. **First, I will try three different values of batch size hyperparameter**
2. When batch size equal 10 the accuracy equal as shown in train



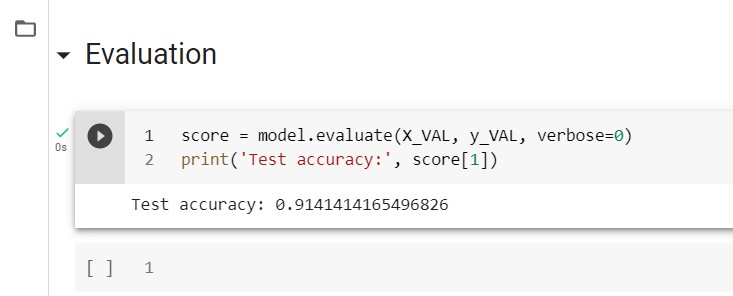
And in test it equal

#### Graphical user interface, application Description automatically generated

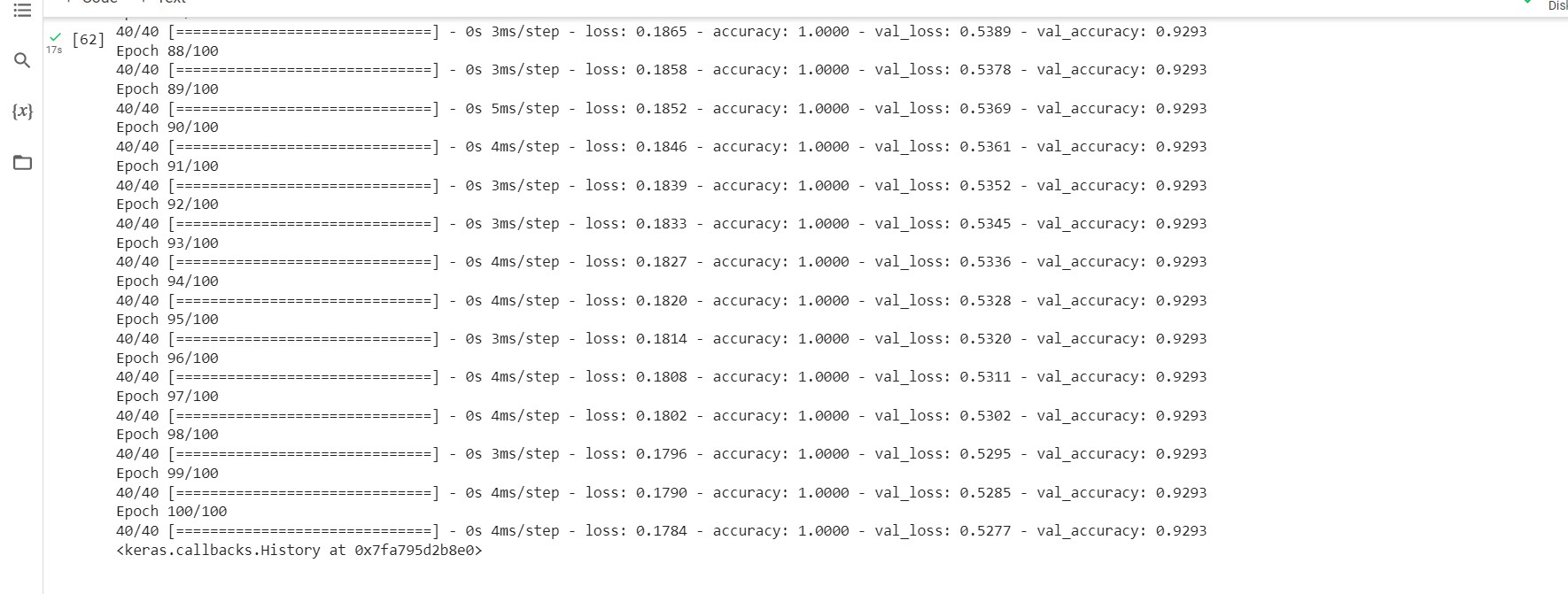
1. When using batch size=15 we find the accuracy on train



And on test accuracy equal as shown



1. When using batch size=20 we find accuracy on train



And on test as shown

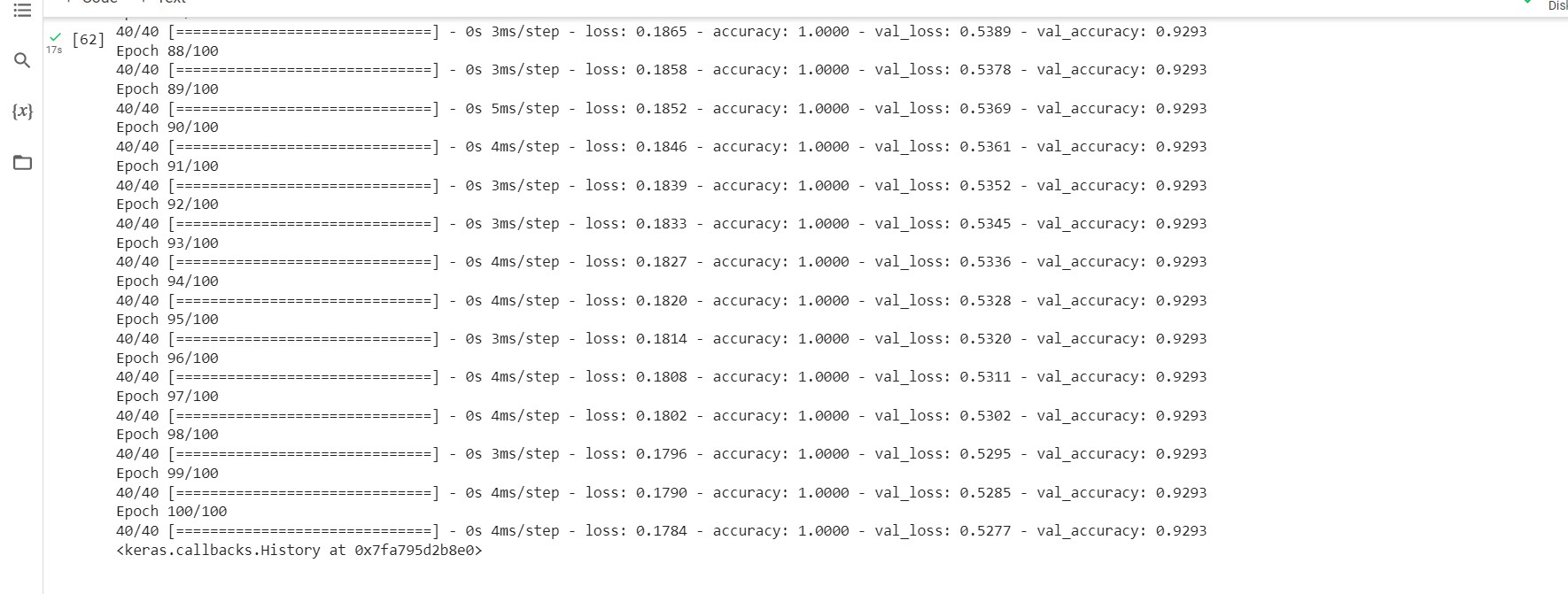


**Observation:-**

The model got high accuracy when I increase the batch size,so the best one =20,when using **(**optimizer='SGD', metrics=['accuracy']

epochs=100, batch\_size=20)

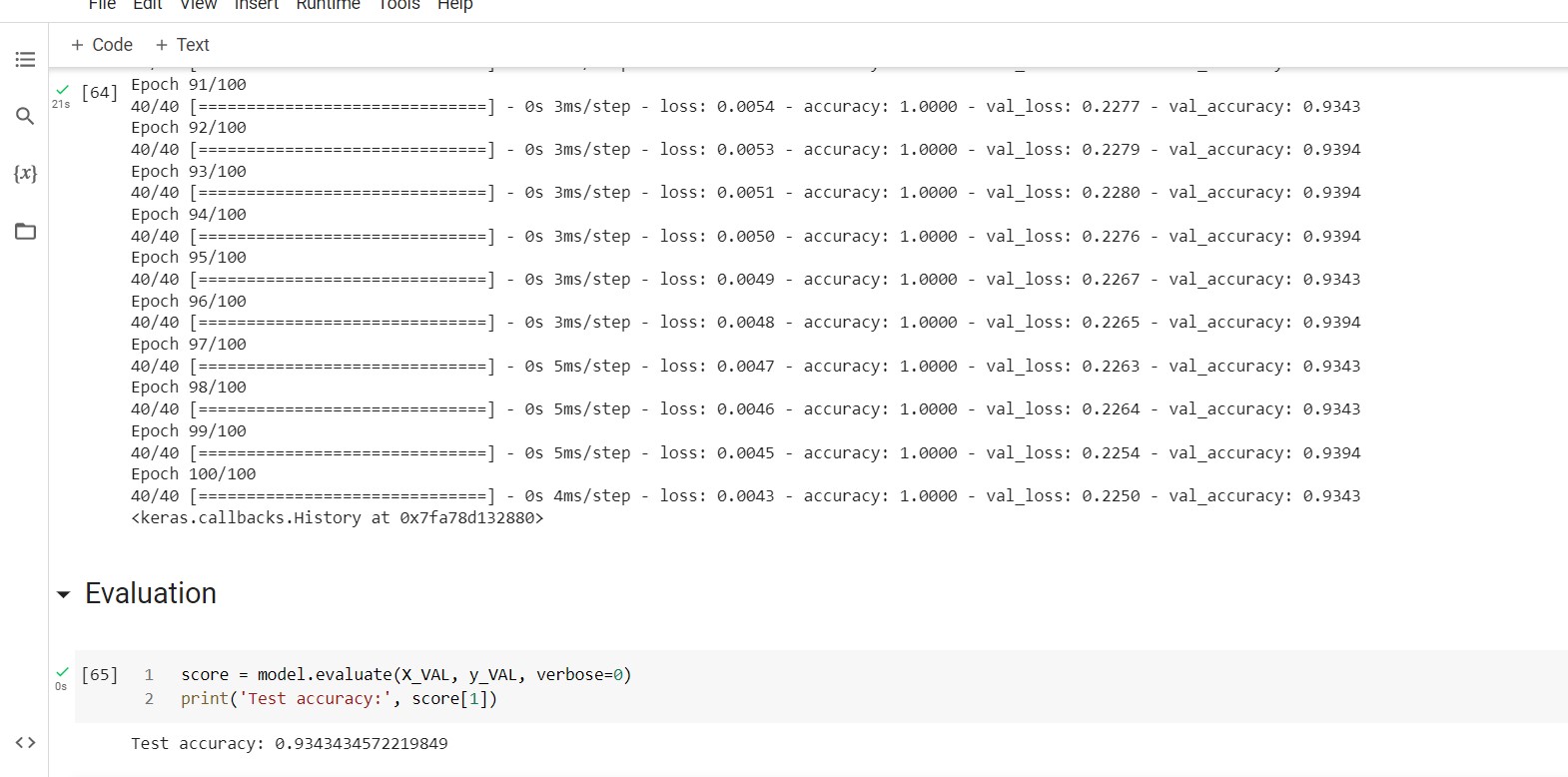
1. **Try different optimizers**
2. First using sgd



Will got loss=0.17, accuracy=1 on train ,and =0.92 on test

1. When using ‘Adam’ optimizer we got

Accuracy as shown



Will got loss=0.004, accuracy=1 on train ,and =0.93 on test

1. When using ‘RMSProp’ optimizer we got



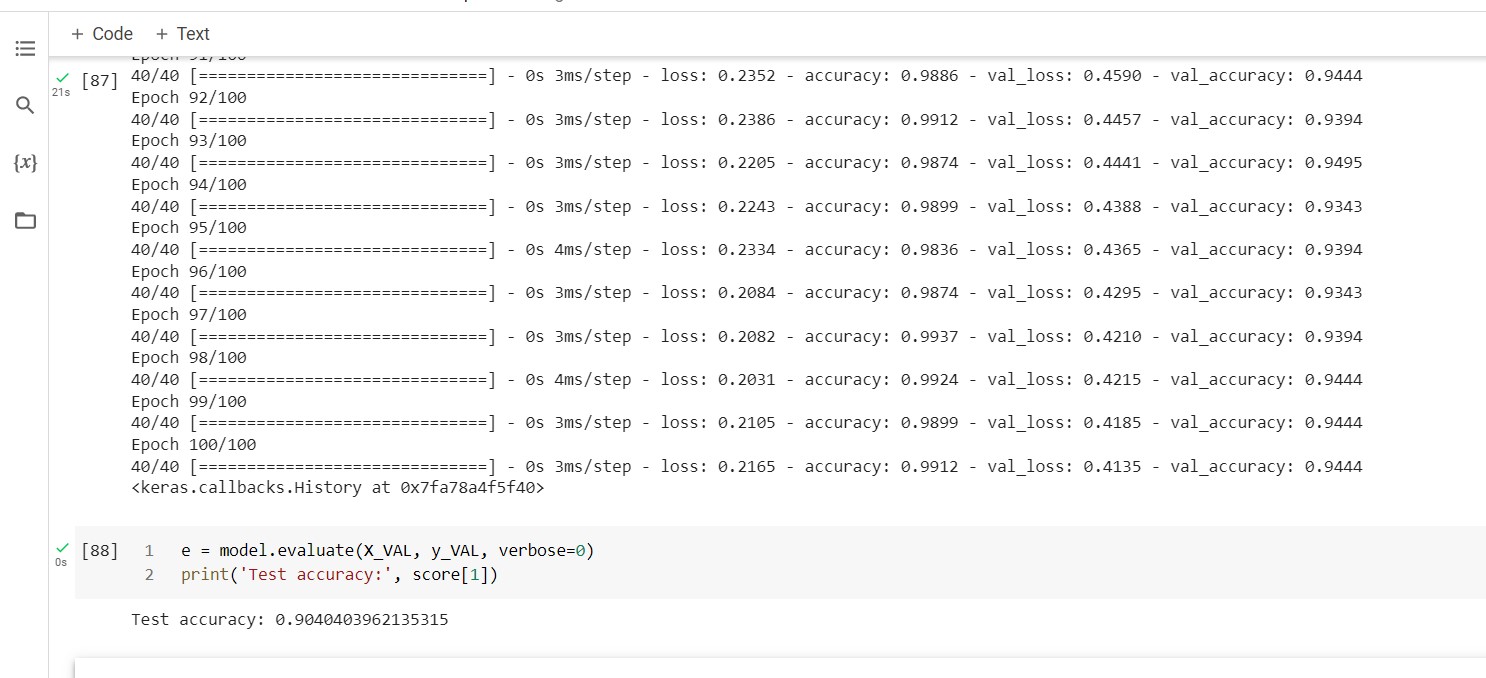
Will got loss=0.001, accuracy=1 on train ,and =0.94 on test

**Observiation:-**

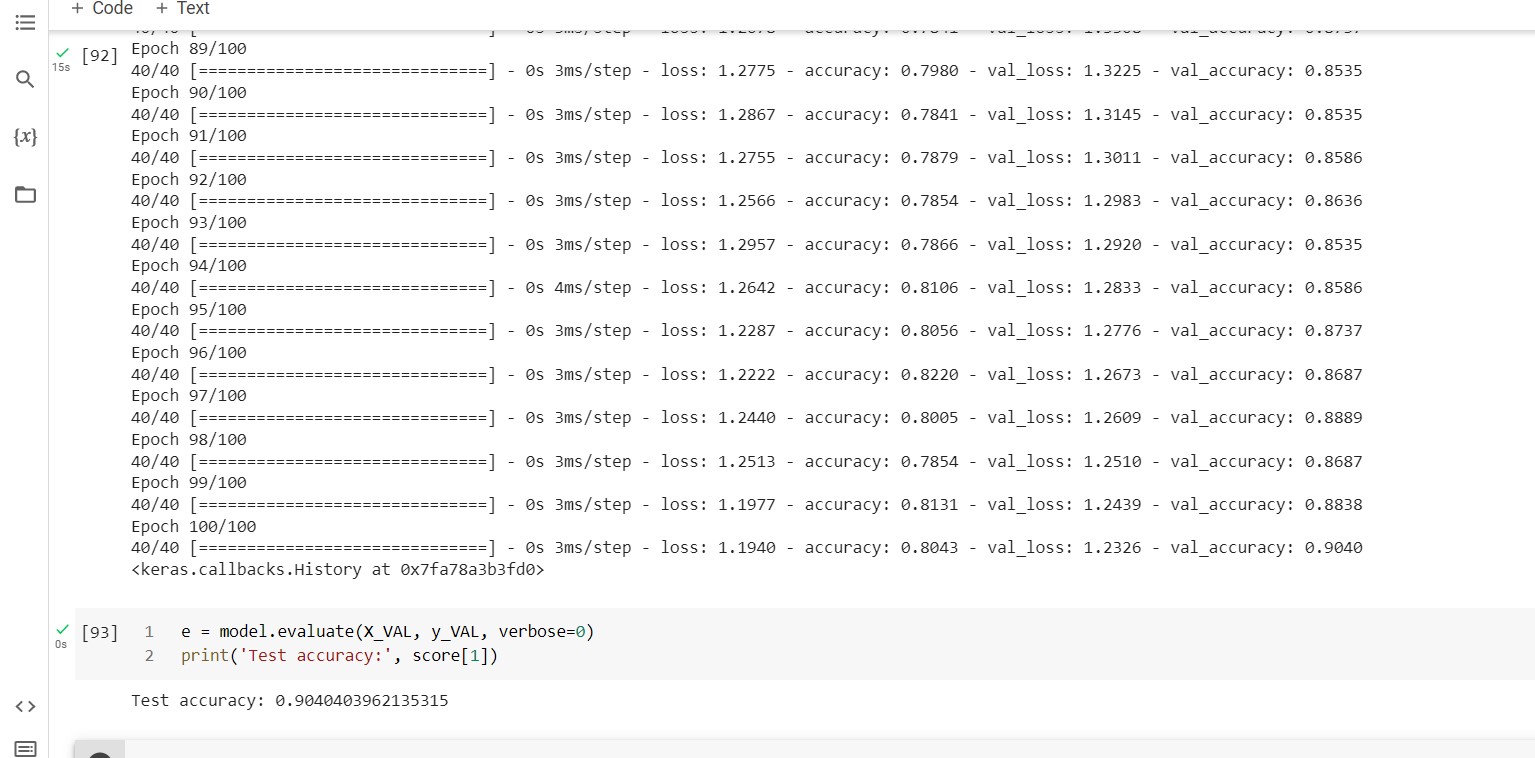
The RMSProp optimizer is the best one here.

**3-Try different values of Dropout**

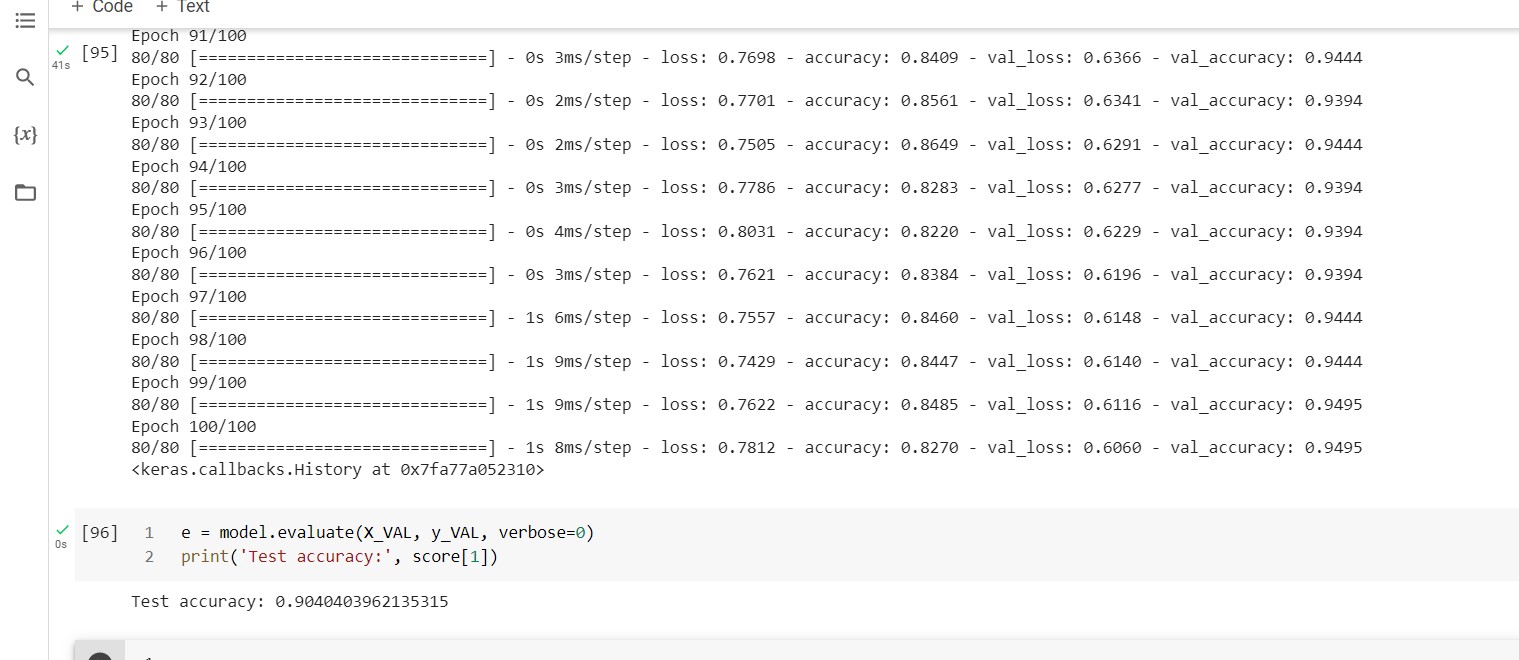
1. First when it equal zero with above optimizer and the higher accuracy got =1 on train and =0.94 on test
2. Second,when it equal 0.2 we got



1. Try when dropout=0.5 we got



1. **try different values of learning rate**
2. **first, Learning rate =0.01**

****

1. **when learning rate=0.001**

**Text

Description automatically generated**

# **learning rate=0.1**



Graphical user interface, text

Description automatically generated

# **Final Result**

At the end when we compare between all the models we find that the best model for our data set is **the model with these parameters( dropout layer= 0 ,optimizer=** RMSProp **, batch size= 20 ,**

**learning rate= default ) as the best results are**  loss=0.001, accuracy=1 on train

and =0.94 on test