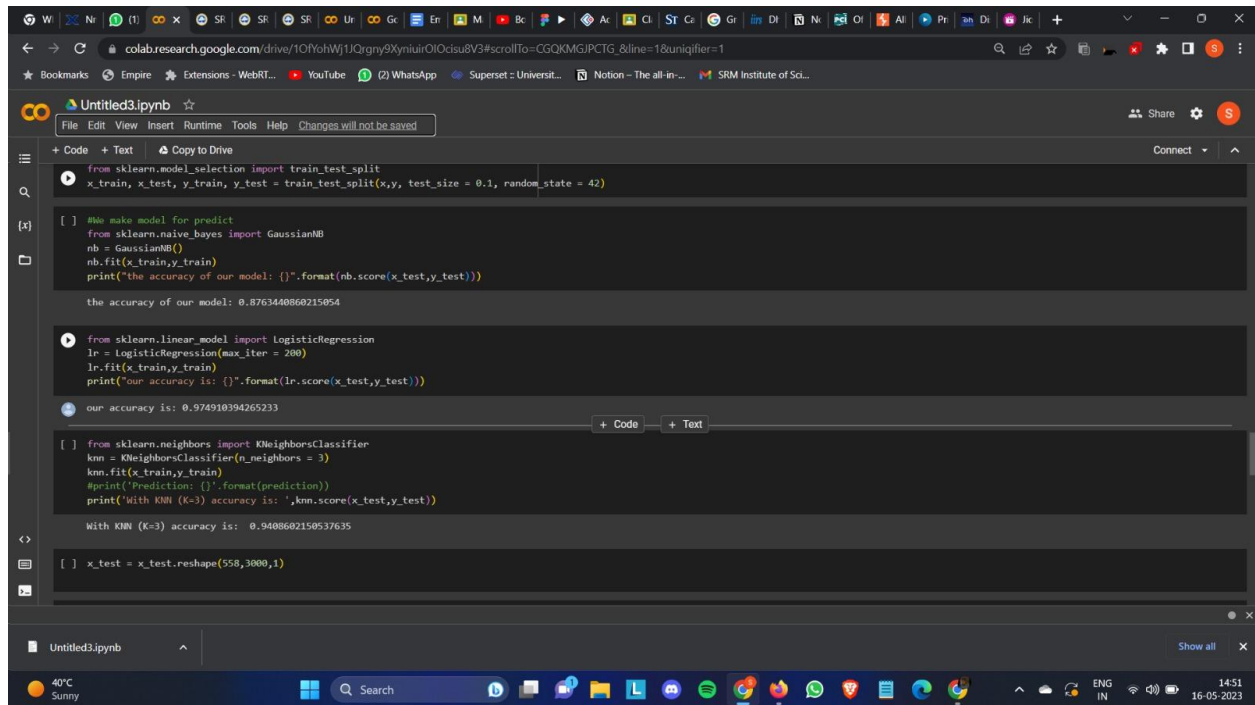


## RESULT SCREENSHOTS:



The screenshot shows a Google Colab notebook titled 'Untitled3.ipynb'. The code is organized into three cells. The first cell imports sklearn.model\_selection and train\_test\_split, then splits the data into training and testing sets. The second cell creates a Gaussian Naive Bayes model, fits it to the training data, and prints the accuracy. The third cell creates a Logistic Regression model, fits it to the training data, and prints the accuracy. The output of the second cell shows an accuracy of 0.8763440860215054. The output of the third cell shows an accuracy of 0.974910394265233. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for code, text, and other functions, and a status bar at the bottom showing the temperature (40°C) and time (14:51, 16-05-2023).

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.1, random_state = 42)

[ ] #We make model for predict
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(x_train, y_train)
print("the accuracy of our model: {}".format(nb.score(x_test, y_test)))

the accuracy of our model: 0.8763440860215054

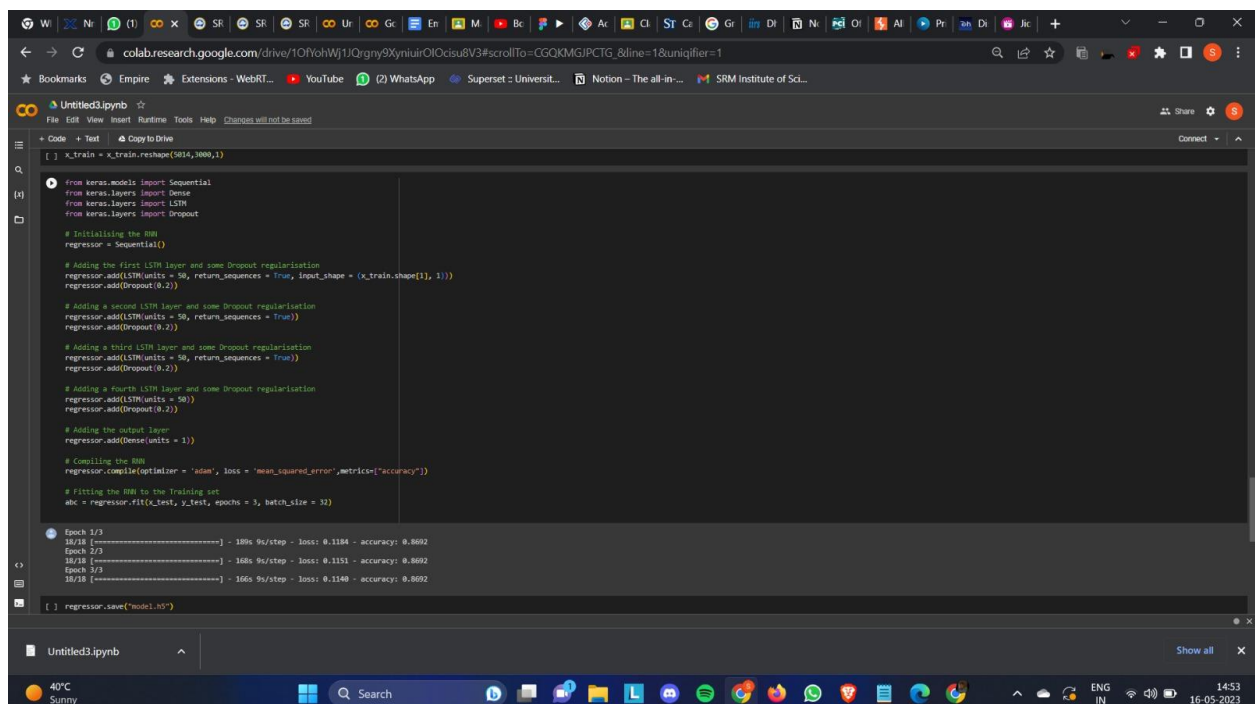
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(max_iter = 200)
lr.fit(x_train, y_train)
print("our accuracy is: {}".format(lr.score(x_test, y_test)))

our accuracy is: 0.974910394265233

[ ] from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(x_train, y_train)
print("Prediction: {}".format(knn.predict(x_test)))
print("With KNN (K=3) accuracy is: {}".format(knn.score(x_test, y_test)))

With KNN (K=3) accuracy is: 0.9408602150537635

[ ] x_test = x_test.reshape(558, 3000, 1)
```



The screenshot shows a Google Colab notebook titled 'Untitled3.ipynb'. The code is organized into two cells. The first cell imports keras.models, Sequential, keras.layers, Dense, LSTM, and Dropout, then initializes the RNN regressor. The second cell adds four LSTM layers with Dropout regularization, compiles the model, and fits it to the training data. The output of the second cell shows the training progress over three epochs. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for code, text, and other functions, and a status bar at the bottom showing the temperature (40°C) and time (14:53, 16-05-2023).

```
[ ] x_train = x_train.reshape(564, 3000, 1)

from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout

# Initializing the RNN
regressor = Sequential()

# Adding the first LSTM layer and some Dropout regularization
regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (x_train.shape[1], 1)))
regressor.add(Dropout(0.2))

# Adding a second LSTM layer and some Dropout regularization
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))

# Adding a third LSTM layer and some Dropout regularization
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))

# Adding a fourth LSTM layer and some Dropout regularization
regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))

# Adding the output layer
regressor.add(Dense(units = 1))

# Compiling the RNN
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error', metrics=['accuracy'])

# Fitting the RNN to the training set
abc = regressor.fit(x_train, y_train, epochs = 3, batch_size = 32)

Epoch 1/3
36/18 [-----] - 100% 9s/step - loss: 0.1184 - accuracy: 0.8692
Epoch 2/3
36/18 [-----] - 100% 9s/step - loss: 0.1151 - accuracy: 0.8692
Epoch 3/3
36/18 [-----] - 100% 9s/step - loss: 0.1140 - accuracy: 0.8692

[ ] regressor.save("model.h5")
```

colab.research.google.com/drive/1OfYohWj1Qrgny9XyniurOI0cisv8V3#scrollTo=CGQKMGJPC7G\_&line=18uniquifier=1

Bookmarks Empire Extensions - WebRT... YouTube (2) WhatsApp Superset - Universit... Notion - The all-in-... SRM Institute of Sci...

Untitled3.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

+ Code + Test Copy to Drive

```
[ ]: # Adding the hidden layer
regressor.add(Dense(units = 1))

# Compiling the RM
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error', metrics=['accuracy'])

# Fitting the RM to the Training set
abc = regressor.fit(x_train, y_train, epochs = 3, batch_size = 32)

Epoch 1/3
32/32 [=====] - 188s 9s/step - loss: 0.1184 - accuracy: 0.8692
Epoch 2/3
32/32 [=====] - 168s 9s/step - loss: 0.1131 - accuracy: 0.8692
Epoch 3/3
32/32 [=====] - 166s 9s/step - loss: 0.1148 - accuracy: 0.8692

[ ]: regressor.save("model.h5")

import matplotlib.pyplot as plt

plt.plot(abc.history['loss'], label='train loss')
plt.plot(abc.history['accuracy'], label='train acc')
plt.legend()
plt.show()
plt.savefig('lossval_loss')
```

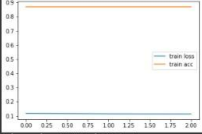


Figure size 432x288 with 0 Axes

Untitled3.ipynb Show all

40°C Sunny Search 14:53 16-05-2023