Student Roll No	812819205001
Student Name	ABARNA S
Team ID	26586-1660030073
Maximum Marks	2 Marks


```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "private_outputs": true,
      "provenance": [],
      "toc_visible": true
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "execution_count": null,
      "metadata": {
        "id": "xPmjSeP6ztWI"
      },
      "outputs": [],
      "source": [
        "import pandas as pd \n",
        "import numpy as np\n",
        "from sklearn.model_selection import train_test_split\n",
        "from sklearn.preprocessing import LabelEncoder\n",
        "from keras.models import Model\n",
        "from keras.layers import LSTM, Activation, Dense, Dropout,
Input, Embedding\n",
        "from keras.optimizers import RMSprop\n",
        "from keras.preprocessing.text import Tokenizer\n",
        "from keras_preprocessing import sequence\n",
        "from keras.utils import to_categorical\n",
        "from keras.models import load_model"
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "import csv\n",
        "import tensorflow as tf\n",
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt\n",
```

```
        "from tensorflow.keras.preprocessing.text import Tokenizer\n",
        "from tensorflow.keras.preprocessing.sequence import
pad_sequences\n",
        "import nltk\n",
        "nltk.download('stopwords')  \n",
        "from nltk.corpus import stopwords\n",
        "STOPWORDS = set(stopwords.words('english'))"
      ],
      "metadata": {
        "id": "Gn4V67q0z0eT"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "from google.colab import drive\n",
        "drive.mount('/content/drive')"
      ],
      "metadata": {
        "id": "E9MGSQZNz6Qp"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "cd/content/drive/MyDrive/Colab Notebooks"
      ],
      "metadata": {
        "id": "3e8BhX8m0Dyf"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "df =
pd.read_csv('/content/drive/MyDrive/spam.csv',delimiter=',',encoding='lat
in-1')\n",
        "df.head()"
      ],
      "metadata": {
        "id": "635W4rCX0Lg5"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
```

```
        "df.drop(['Unnamed: 2','Unnamed: 3', 'Unnamed:
4'],axis=1,inplace=True) \n",
        "df.info()"
      ],
      "metadata": {
        "id": "JZ69G7O24Zsg"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "df.groupby(['v1']).size()"
      ],
      "metadata": {
        "id": "F47AuS5v4e2X"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "#Label Encoding Required Column\n",
        "X = df.v2\n",
        "Y = df.v1\n",
        "le = LabelEncoder()\n",
        "Y = le.fit_transform(Y)\n",
        "Y = Y.reshape(-1,1)"
      ],
      "metadata": {
        "id": "n46sH3zT4ieg"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "# Test and train data split \n",
        "X_train,X_test,Y_train,Y_test =
train_test_split(X,Y,test_size=0.15)"
      ],
      "metadata": {
        "id": "jQ1vSLxw4mbB"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "# Tokenisation function\n",
```

```
        "max_words = 1000\n",
        "max_len = 150\n",
        "tok = Tokenizer(num_words=max_words)\n",
        "tok.fit_on_texts(X_train)\n",
        "sequences = tok.texts_to_sequences(X_train)\n",
        "sequences_matrix =
sequence.pad_sequences(sequences,maxlen=max_len)"
      ],
      "metadata": {
        "id": "EouGpOLE4pt-"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "\n",
        "#LSTM model\n",
        "inputs = Input(name='InputLayer',shape=[max_len])\n",
        "layer = Embedding(max_words,50,input_length=max_len)(inputs)\n",
        "layer = LSTM(64)(layer)\n",
        "layer = Dense(256,name='FullyConnectedLayer1')(layer)\n",
        "layer = Activation('relu')(layer)\n",
        "layer = Dropout(0.5)(layer)\n",
        "layer = Dense(1,name='OutputLayer')(layer)\n",
        "layer = Activation('sigmoid')(layer)"
      ],
      "metadata": {
        "id": "iiDknBZl4tNi"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model = Model(inputs=inputs,outputs=layer)\n",
        "model.summary()\n",

"model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['a
ccuracy'])"
      ],
      "metadata": {
        "id": "dV2l69IR4wqU"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
```

```
"model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,validation_s
plit=0.2)"
      ],
      "metadata": {
        "id": "hLG5kG4240AO"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model.save(\"Ai_Spam_Identifier\")"
      ],
      "metadata": {
        "id": "Tg4RnViS5WJp"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "test_sequences = tok.texts_to_sequences(X_test)\n",
        "test_sequences_matrix =
sequence.pad_sequences(test_sequences,maxlen=max_len)"
      ],
      "metadata": {
        "id": "NfRE9Xky5oq7"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "accuracy = model.evaluate(test_sequences_matrix,Y_test)\n",
        "print('Accuracy: {:0.3f}'.format(accuracy[1]))"
      ],
      "metadata": {
        "id": "MmkiG2Ms5syq"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "y_pred = model.predict(test_sequences_matrix)\n",
        "print(y_pred[25:40].round(3))\n"
      ],
      "metadata": {
        "id": "hF_RFQt95zC9"
```

```
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "print(Y_test[25:40])"
      ],
      "metadata": {
        "id": "puWIeYkr58Xb"
      },
      "execution_count": null,
      "outputs": []
    }
  ]
}
```