

ASSIGNMENT-4

NAME: GOKULAKRISHNAN P

ROLL NUMBER: 714019106026

COLLEGE NAME: SRI SHAKTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY ,COIMBATORE.

SMS SPAM CLASSIFICATION

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": [],
      "collapsed_sections": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "markdown",
      "source": [
        "# **2. IMPORT LIBRARIES**"
      ],
      "metadata": {
        "id": "h206JVlxjJqt"
      }
    },
  ],
}
```

```
{  
  "cell_type": "code",
```

```

"execution_count": null,
"metadata": {
  "id": "jTLGPy_fgQJC"
},
"outputs": [],
"source": [
  "import pandas as pd\n",
  "import numpy as np\n",
  "import matplotlib.pyplot as plt\n",
  "import seaborn as sns\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.callbacks import EarlyStopping\n",
  "from keras.utils import pad_sequences\n",
  "%matplotlib inline"
]
},
{
  "cell_type": "markdown",
  "source": [
    "# **3.1. READ DATASET**"
  ],
  "metadata": {
    "id": "w2-i5_IGjSUh"
  }
}

```

```

},
{
  "cell_type": "code",
  "source": [
    "df = pd.read_csv('spam.csv',delimiter=',',encoding='latin-1')\n",
    "df.head()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 206
    },
    "id": "b2S-rygKh48X",
    "outputId": "bb6b8163-f10b-4479-e328-681429a9c109"
  },
  "execution_count": null,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "   v1                                v2 Unnamed: 2 \\n",
          "0  ham  Go until jurong point, crazy.. Available only ...   NaN \n",
          "1  ham                Ok lar... Joking wif u oni...   NaN \n",
          "2  spam  Free entry in 2 a wkly comp to win FA Cup fina...   NaN \n",
          "3  ham  U dun say so early hor... U c already then say...   NaN \n",
          "4  ham  Nah I don't think he goes to usf, he lives aro...   NaN \n",
          "\n",
          " Unnamed: 3 Unnamed: 4 \n",
          "0    NaN    NaN \n",
          "1    NaN    NaN \n"
        ]
      }
    }
  ]
}

```

```

"2    NaN    NaN \n",
"3    NaN    NaN \n",
"4    NaN    NaN "
],
"text/html": [
  "\n",
  " <div id=\"df-f2140a1b-e52d-4a73-807f-7b169f50a515\">\n",
  " <div class=\"colab-df-container\">\n",
  "   <div>\n",
  "<style scoped>\n",
  "   .dataframe tbody tr th:only-of-type {\n",
  "     vertical-align: middle;\n",
  "   }\n",
  "\n",
  "   .dataframe tbody tr th {\n",
  "     vertical-align: top;\n",
  "   }\n",
  "\n",
  "   .dataframe thead th {\n",
  "     text-align: right;\n",
  "   }\n",
  "</style>\n",
  "<table border=\"1\" class=\"dataframe\">\n",
  " <thead>\n",
  " <tr style=\"text-align: right;\">\n",
  "   <th></th>\n",
  "   <th>v1</th>\n",
  "   <th>v2</th>\n",
  "   <th>Unnamed: 2</th>\n",
  "   <th>Unnamed: 3</th>\n",
  "   <th>Unnamed: 4</th>\n",

```

```
" </tr>\n",
" </thead>\n",
" <tbody>\n",
" <tr>\n",
" <th>0</th>\n",
" <td>ham</td>\n",
" <td>Go until jurong point, crazy.. Available only ...</td>\n",
" <td>NaN</td>\n",
" <td>NaN</td>\n",
" <td>NaN</td>\n",
" </tr>\n",
" <tr>\n",
" <th>1</th>\n",
" <td>ham</td>\n",
" <td>Ok lar... Joking wif u oni...</td>\n",
" <td>NaN</td>\n",
" <td>NaN</td>\n",
" <td>NaN</td>\n",
" </tr>\n",
" <tr>\n",
" <th>2</th>\n",
" <td>spam</td>\n",
" <td>Free entry in 2 a wkly comp to win FA Cup fina...</td>\n",
" <td>NaN</td>\n",
" <td>NaN</td>\n",
" <td>NaN</td>\n",
" </tr>\n",
" <tr>\n",
" <th>3</th>\n",
" <td>ham</td>\n",
" <td>U dun say so early hor... U c already then say...</td>\n",
```

```

"    <td>NaN</td>\n",
"    <td>NaN</td>\n",
"    <td>NaN</td>\n",
"  </tr>\n",
" <tr>\n",
"   <th>4</th>\n",
"   <td>ham</td>\n",
"   <td>Nah I don't think he goes to usf, he lives aro...</td>\n",
" <td>NaN</td>\n",
"   <td>NaN</td>\n",
"   <td>NaN</td>\n",
" </tr>\n",
" </tbody>\n",
"</table>\n",
"</div>\n",
"  <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-f2140a1b-e52d-4a73-807f-7b169f50a515')\" \n",
"    title=\"Convert this dataframe to an interactive table.\" \n",
"    style=\"display:none;\">\n",
"    \n",
"  <svg xmlns=\"http://www.w3.org/2000/svg\" height=\"24px\" viewBox=\"0 0 24 24\" \n",
"    width=\"24px\">\n",
"    <path d=\"M0 0h24v24H0V0z\" fill=\"none\"/>\n",
"    <path d=\"M18.56 5.44l.94 2.06.94 2.06-.94 2.06-.94 2.06-.94 2.06-2.06.94zm-11 11l8.5 8.5l.94 2.06 2.06-.94 2.06-.94l8.5 2.5l-.94 2.06-2.06.94zm10 10l.94 2.06.94 2.06-.94 2.06-.94-.94 2.06-.94 2.06-.94 2.06-2.06.94z\"/><path d=\"M17.41 7.96l-1.37-1.37c-.4-.4-.92-.59-1.43-.59-.52 0-1.04.2-1.43.59L10.3 9.45l-7.72 7.72c-.78.78-.78 2.05 0 2.83L4 21.41c.39.39.92.59 1.41.59.51 0 1.02-.2 1.41-.59l7.78-7.78 2.81-2.81c.8-.78.8-2.07 0-2.86zM5.41 20L4 18.59l7.72-7.72 1.47 1.35L5.41 20z\"/>\n",
"  </svg>\n",
" </button>\n",
"  \n",
" <style>\n",

```

```
" .colab-df-container {\n",
"   display:flex;\n",
"   flex-wrap:wrap;\n",
"   gap: 12px;\n",
" }\n",
"\n",
" .colab-df-convert {\n",
"   background-color: #E8F0FE;\n",
"   border: none;\n",
"   border-radius: 50%;\n",
"   cursor: pointer;\n",
"   display: none;\n",
"   fill: #1967D2;\n",
"   height: 32px;\n",
"   padding: 0 0 0 0;\n",
"   width: 32px;\n",
" }\n",
"\n",
" .colab-df-convert:hover {\n",
"   background-color: #E2EBFA;\n",
"   box-shadow: 0px 1px 2px rgba(60, 64, 67, 0.3), 0px 1px 3px 1px rgba(60, 64, 67, 0.15);\n",
"   fill: #174EA6;\n",
" }\n",
"\n",
" [theme=dark] .colab-df-convert {\n",
"   background-color: #3B4455;\n",
"   fill: #D2E3FC;\n",
" }\n",
"\n",
" [theme=dark] .colab-df-convert:hover {\n",
```



```

"   background-color: #434B5C;\n",
"   box-shadow: 0px 1px 3px 1px rgba(0, 0, 0, 0.15);\n",
"   filter: drop-shadow(0px 1px 2px rgba(0, 0, 0, 0.3));\n",
"   fill: #FFFFFF;\n",
" }\n",
" </style>\n",
"\n",
"   <script>\n",
"       const buttonEl =\n",
"           document.querySelector('#df-f2140a1b-e52d-4a73-807f-7b169f50a515 button.colab-  
df-convert');\n",
"       buttonEl.style.display =\n",
"           google.colab.kernel.accessAllowed ? 'block' : 'none';\n",
"\n",
"       async function convertToInteractive(key) {\n",
"           const element = document.querySelector('#df-f2140a1b-e52d-4a73-807f-  
7b169f50a515');\n",
"           const dataTable =\n",
"               await google.colab.kernel.invokeFunction('convertToInteractive',\n",
"                   [key], {});\n",
"           if (!dataTable) return;\n",
"\n",
"           const docLinkHtml = 'Like what you see? Visit the ' +\n",
"               '<a target=\"_blank\"  
href=https://colab.research.google.com/notebooks/data_table.ipynb>data table notebook</a>'\n",
"               + ' to learn more about interactive tables.';\n",
"           element.innerHTML = \"\n",
"               dataTable['output_type'] = 'display_data';\n",
"               await google.colab.output.renderOutput(dataTable, element);\n",
"               const docLink = document.createElement('div');\n",
"               docLink.innerHTML = docLinkHtml;\n",
"               element.appendChild(docLink);\n",

```

```
"    }\n",  
"  
</script>\n",  
"  
</div>\n",  
"  
</div>\n",  
"  
"  
  
]  
  
,  
"metadata": {},  
"execution_count": 2  
  
}  
  
]  
  
,  
{  
  "cell_type": "markdown",  
  "source": [  
    "# **3.2. PREPROCESSING**"  
  ],  
  "metadata": {  
    "id": "RzavY7S-jiO2"  
  }  
},  
{  
  "cell_type": "code",  
  "source": [  
    "df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)\n",  
    "df.info()"  
  ],  
  "metadata": {  
    "colab": {  
      "base_uri": "https://localhost:8080/"  
    }  
},
```

```

    "id": "UXZj9dv9h-Lq",
    "outputId": "003ad4bf-a18e-4a95-a237-635d94e25ea9"
  },
  "execution_count": null,
  "outputs": [
    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "<class 'pandas.core.frame.DataFrame'>\n",
        "RangeIndex: 5572 entries, 0 to 5571\n",
        "Data columns (total 2 columns):\n",
        "#   Column  Non-Null Count  Dtype \n",
        "-----\n",
        "0   v1      5572 non-null object\n",
        "1   v2      5572 non-null object\n",
        "dtypes: object(2)\n",
        "memory usage: 87.2+ KB\n"
      ]
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "sns.countplot(df.v1)\n",
    "plt.xlabel('Label')\n",
    "plt.title('Number of ham and spam messages')\n",
    "X = df.v2\n",
    "Y = df.v1\n",
    "le = LabelEncoder()\n"
  ]
}

```

```

"Y = le.fit_transform(Y)\n",
"Y = Y.reshape(-1,1)"
],
"metadata": {
  "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 351
  },
  "id": "JqQHhqtuiCbC",
  "outputId": "de038311-bc99-46d1-af4a-fe897347073c"
},
"execution_count": null,
"outputs": [
  {
    "output_type": "stream",
    "name": "stderr",
    "text": [
      "\n/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the
following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be
`data`, and passing other arguments without an explicit keyword will result in an error or
misinterpretation.\n",
      "\n FutureWarning\n"
    ]
  },
  {
    "output_type": "display_data",
    "data": {
      "text/plain": [
        "<Figure size 432x288 with 1 Axes>"
      ]
    },
    "image/png":
      "iVBORw0KGgoAAAANSUhEUgAAAYsAAAEWCAYAAACXGLsWAAAABHNCSVQICAgIfAhkiAAAAAlwSFlz
      AAALeAAACxiB0t1+/AAAADh0RVh0U29mdHdhcmUAAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjlsIGh0dHA

```

6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAZjUIEQVR4nO3deZhdZ3n8feHBEQFJUhesGjDKLbirhGwtbtp
HAFxgdc2wXEJmI72toz7do9I4j4qK0j7nbTioCOlu5p2xZRCrtFSFxAQCDYsKWSAKCoiPwnT/Or+RQVO
VUILeqknq/nuc+dc7vLPd7zr11P/esN1WFJEkbs81MFyBJmv0MCOonSIMNCKjTIsJAKDTIsJEmDDAtJ0iDD
QIOS5KQkx83QcyfJR5NsSHL2BMOPSPldmahtc0qyf5I1M12HNBHDYguV5Nika5Pcvdf24iTfnMGyRuXx
wBOBxVW1z0wXI81FhsWWbR7wqpkuyIMImbeJk9wXuLSqfjOKEiQNMyy2bO8AXp1kp/EDkixJUknm99
q+meTFRfulJP8nyfJrk1ycZl/a+2r21bLsnGz3SXJGUmuT/KtJPftzfuBbdj6JD9P8qzesJOSfCjJl5P8BvirCerd
PcnyNv2qJC9p7UcCHwYem+SGJG+abGUkeWfbVXVJkif12l+Y5MJW98VJXtobtn+SNULe25b5yiSHJTKkyS
9aPf+wked8cplfJfl1W2/HTPAaLEvyys/SvKPveF3betmQ5lLgMds5HnSXqu17bnOS/KQ3vr95428Nu9pt
f06ycokf94bdkySTyf5eJv2vCQPSPKG9lyrkxy4kbouTfKaJOcm+U2SjyTZNcl/tPl9LcmC3vj7Jflee8/9JMN+v
WFHtNfn+vYaPre1378t03VtHX5qist21yQnt/V7YXuN1/SG757ks0nWted7ZW/YPklWtPleneRdk62DOa
OqfGyBD+BS4D8DnwOOa20vBr7ZupcABczvTfNN4MWt+wjJuCFdFsoxwG/BD4A3AU4ELge2KGNf1Lr/
4s2/D3Ad9uwuwOr27zmA48EfgXs3Zv2OuBxdF9Qtp9geb4NfBDYHngEsA44oFrdzeyLo4A/gC8pC3Ly4
ArgLThTwbuBwT4S+C3wKPasP3bengjsG2bxzrgE8COWIOBG4E9J3nu/YGHtuV6GHA1cNi41+BfgbsCDw
d+DzyoDX8b8B1gZ2AP4KfAmkme5yBgJbBTW44HAbsNvTzt+POAe7XX5u+Bq8ZeA+AY4Hdt/vOBU4BL
gH/srY9LBt6HZwG7AouAtcAP23tge+AbwNft3EXANcAhbX09sfUvpHsP/Rr40zbubsCDW/cnWz3btHk+f
orL9jbgW8ACYDFw7tj6bfNa2V737YD/BFwMHNSGfx94fuveAdhvpv/nZ/ox4wX4uIMv3K1h8RC6D+Kfb
HpYXNQb9tA2/q69tmuAR7Tuk4BTen2AG6m+5B7NvCdcfX9S+9D4iTglI0syx5tXjv22t4KnNSrdSgsVvX
679aW5T6TjP8F4FWte3+6MJX+nds0+7bG38lLQCM8Lq8Gzh+3GuwuDf8bODw1n0xcHBv2FFMHhYHA
L8A9gO2GTds0tdmknltAB7euo8BzugNeyppwwTrY6eNvA+f2+v/LPChXv/fAI9o3a8DPjZu+tOBZXRhcS3
w18Bdx41zCnBCfz1uZP33l+2PH/6t/8XcGhb7Ar8cN+0bgl+27m8DbwJ22Zz/t1vyw91QW7iq+inwJeD1d
2Dyq3vdN7b5jW/bode/uve8NwDrqd3pjins23YtXjvWuC5wH0mmnYCuwPrq+r6XtldN9Ep+qqXm2/b
Z07ACR5UpKz2i6la+m+2e7Sm/aaqrq5dd/Y/m5sPfxRkn2TnNI2ZVwH/M24ed+mNrqtmrF57c5t18tlky1
cVX0DeD/dlt/aJCckuUdvIMleG5K8uu2Gua4t/z3H1Th+WX81wfqYcPknmX6ydXdf4Jnj3iePp9tC+g3dl46
/Aa5M8u9JHtimey3d1tTZSc5P8qKxmQ8s2/j12+++L7D7uFr+gW4LCeBI4AHAz5Kck+QpG1n+OcGw2Do
cTbe7oP/hOnYw+G69tv6H9x2xx1hHkh3odp9cQfDp+K2q2qn32KGqXtabdmO3N74C2DnJr22PwEuv5P
1kuQudN9230m31bQT8GW6D5/N4RPAcrpv8fcE/nkT5n0lvXVKt8yTqqr3VtWjgb3pPshe0xs84WvT9uG
/FngWsKAt/3WbUOPmtJpuy6L/PrI7Vb0NoKpOr6on0u2C+hnd7juq6qqqeklv7Q68FPhgO44xtGxX0u1+
GtNf16vpdq/1a9mxqg5pz3lRVT0HuDfwdUaZ6Z15OBcZFluBqlOfAp4Za9tHd2H7fOSzGvfXu53J5/qkCS
PT7Id8GbgRkPaTbdl84Akz0+ybXs8JsmDplj/auB7wFuTbJ/kYXTf7D5+J+uFbn/0XeIOQ9yU7sD3pAds74A
d6baKfpdkH+C/bMK0pwFvSLlgyWK6XTYTautz3yTb0n0R+B1wS2+UyV6bHemOyawD5id5I3APZsbHgac
mOai9J7dPd4LB4nZQ/ND2gxf7ul1htwAkeWZbP9DtZqo2bGjZ+ut3EfCK3rCzgeuTvK4dCJ+X5CFJHtOe83
lJfIbVLX57x+C263vOMSy2Hsf57ftewndt89r6A7Ufu9OPscn6Lzi1gOPpju45Nt9dCBwON1WwIV038bus
gnzfg7dPv4rgM/ThE/42p2sd6y2V9J9cGyg+zBffmf2/NfgWOTXE93sPSOTzj2TXS7ni4Bvgp8bCPj3oPu
m/aGns01dGfDjZnwtaE7JvAVuuMdl9GFzMMZ2CY5MC69D6Xb3rGt1vlbuc2gb4L/Tvf7r6U5EGNsYfQzwg
yQ30L12r6qqixletmOBNXTr92vAZ+iCiLab7SI0J1NcQndCxoFpdmMBHAYc357zPXTHmW5kDhs7WOTSF
rJSXQHbv/HTNcymyV5Gd2H/I/OdC1blrcsJG2VkuyW5HFJtknyp3Sn1n5+puvaUs0fHkWStkjb0Z3CvSfcd
YdT6a7lOR3gbihJ0iB3Q0mSBo10N1SSS+luQ3AzCFNVLU2yM91pnkvorv58VIVtSBK6sw4Oobtw6Yiq+m
GbzzJg7ODdcVv18saed5dddqklS5Zs9uWRpK3ZypUrf1VVCycaNh3HLP6qqn7V63898PWqeluS17f+1wF
PAVzqj32BD9FdfBwz3SmbS+nOr16ZZHIVbZjsCZcsWcKKFStGszSStJVKMuldBGZiN9ShwNiWwcnAYb32
U6pzFrBTkt3obnB2RIWtbwFxBt050JKkaTLqsCjgq+3WwUe1tl2r6srWfRW33otlEbe9oGZNa5us/TaSHN
VuKbxi3bp1m3MZJGnOG/VuqMdX1eVJ7g2ckeRn/YFVVUk2y+IYVXUC3Z0pWbp0qad4SdJmNNiti6q6v
P1dS3cxzD7A1W33Eu3v2jb65dz2RI+LW9tk7ZKkaTKysEhy97G7iLabgx1I9+Muy+nuX0/7+8XWvRx4QTr
7Ade13VWnAwe2m4EtaPM5fVR1S5Jub5S7oXYFPt+dEct84BNV9ZUk5wCnpfu5zMvobi8M3W2jDwFW
OZ06+0KAqlqf5M3AOW28Y6tq/QjrliSns1Vewb106dLy1FIJ2rJrVlbV0omGeQW3JGmQYSFJGuRdZyfx6
NecMtMlaBZa+Y4XzHQJ0oxwy0KSNMiwkCQNMiwkSYMMCOonSIMNCKjTIsJAKDTIsJEmDDAtJ0iDDQpI0
yLCQJA0yLCRJgwwLSdlgw0KSNMiwkCQNMiwkSYMMCOonSIMNCKjTIsJAKDTIsJEmDDAtJ0iDDQpI0yLCQ
JA0yLCRJgwwLSdlgw0KSNMiwkCQNMiwkSYMMCOonSIMNCKjTIsJAKDTIsJEmDRh4WSeYI+VGS7X+PZP
8IMmqJ9Ksl1rv0vrX9WGL+nN4w2t/edJdhp1zZKk25qOLYtXARf2+t8OHF9V9wc2AEe29iOBda39+DYE

SfYGDgceDBwMfDDJvGmoW5LUjDQskiwGngx8uPUHOAD4TBvlZOCw1n1o66cNf0lb/1Dg1Kr6fVVdAq
wC9hll3ZKk2xr1lsW7gdcCt7T+ewHXVtVNrX8NsKh1LwJWA7Th17Xx/9g+wTR/IOSJCuSrFi3bt3mXg5J
mtNGFhZIngKsraqVo3qOvqo6oaqWVtXShQsXTsdTstKcMX+E834c8LQkhWDbA/cA3gPslGR+23pYDFze
xr8c2ANYk2Q+cE/gml77mP40kqRpMLiti6p6Q1UtrqoldAeov1FVzwXOBJ7RRlsGfLF1L2/9tOHfqKpq7Ye
3s6X2BPYCzh5V3ZKk2xvllsVkXgecmuQ44EfAR1r7R4CPJVkFrKcLGKrq/CSnARcANwEvr6qbp79sSZq7pi
UsquqbwDdb98VMcdZTVf0OeOYk078FeMvoKpQkbYxXcEuSBhkWkqRBhoUkaZBhIUkaZFhIkgYZFpKk
QYaFJGmQYSFJGmRYSJIGGRaSpEGGhSRpkGEhSRpkWEiSBhkWkqRBhoUkaZBhIUkaZFhIkgYZFpKkQYa
FJGmQYSFJGmRYSJIGGRaSpEGGhSRpkGEhSRpkWEiSBhkWkqRBhoUkaZBhIUkaZFhIkgYZFpKkQYaFJG
mQYSFJGmRYSJIGGRaSpEEjC4sk2yc5O8IPkpyf5E2tfc8kP0iyKsmnkmzX2u/S+le14Ut683pDa/95koNGV
bMkaWKj3LL4PXBavT0ceARwcJL9gLcdX1fv/YENwJfT/COBDa39+DYeSfYGDgceDBwMfDDJvBHWLUka
Z2RhUZ0bWu+27VHAACbnWvWjWGGt+9DWTxv+hCRp7adW1e+r6hJgFbDPqOqWJN3eSI9ZJmX5MfA
WuAM4P8C11bVTW2UNcCi1r0IWA3QhL8H3KvfPsE0/ec6KsmKJCvWrVs3isWRpDlrpGFRVTdX1SOAXR
bAw8c4XOdUFVLq2rpwoULR/U0kjQnTcvZUFV1LXAm8FhgpyTz26DFwOWt+3JgD4A2/J7ANf32CaaRJE
2DUZ4NtTDJTq37rsATgQvpQuMZbbRlWbDb9/LWTxv+jaqq1n54O1tqT2Av4OxR1S1Jur35w6PcYbsBJ7c
zl7YBTquqLyW5ADg1yXHAj4CpTPE/AnwsySpgPd0ZUFTV+UIOAy4AbgJeXlU3j7BuSdl4lwuLqjoXeOQE7
RczwdlMVfU74JmTzOstWfs2d42SpKnxcm5J0iDDQpI0yLCQJA2aUlgk+fpU2iRjW6eNHuBOsj1wN2CXJA
uAtEH3YIKrqCVJW6ehs6FeCvwdSDuwlkVd4tfA+0dYlyRpFtloWFTVe4D3JPnbqnrNNuKsZplpnSdRVW9
L8mfAUv601TVKSOqS5I0i0wpLJJ8DLgf8GNg7OrpAgwLSZoDpnoF91Jg73avJknSHDPV6yx+CtnlIvIkma
vqW5Z7AJckORsup9LBaCqnjaSqiRJs8pUw+KYURYhSZrdpno21LdGXYgkafaa6tlQ19Od/QSwHbAt8Juqu
seoCpMkzR5T3bLYcaw7SYBDgf1GVZQkaXbZ5LvOVuclWEEjqEeSNAtNdTfU03u929Bdd/G7kVQkSZp1p
no21FN73TcBl9LtiPlkzQFTPWbxwEXIkmaVab640eLk3w+ydr2+GySxaMuTpi000z1APdHgeV0v2uxO/B
vrU2SNAdMNSwWVtVHq+qm9jgJWDjCuiRJs8hUw+KaJM9LMq89ngdcM8rCJEmzx1TD4kXAs4CrgCuBZ
wBHjKgmSdIsM9VTZ48FliXVBaOkOwPvpAsRSdJWbqpbFg8bCwqAqloPPHI0JumSZpuphsU2SRaM9bQt
i6lulUiStnBT/cD/X8D3k3y69T8TeMtoSplkzTZTVL7lCQrgANa09Or6oLRISVJmk2mvCuphYMBIUIz0Cbfo
lySNPCyFpKkQYaFJGnQyMliYR5JzkxyQZLzk7yqte+c5lwkF7W/C1p7krw3yaok5yZ5VG9ey9r4FyVZNqqaJ
UkTG+WWxU3A31fV3nS/1/3yJHsDrwe+XIV7AV9v/QBPavZqj6OAD8Efr+k4GtgX2Ac4un/NhyRp9EYWF
IV1ZVX9sHVfD1wILKL7hb2T22gnA4e17kOBU9pvfJ8F7JRkN7rf+j6jqta3q8jPAA4eVd2SpNublmMWSZb
Q3R7k8CuVXVIG3QVsGvrXgSs7k22prVN1j7+OY5KsiLjinXr1m3W+iVprht5WCTZAFgs8HdV9ev+sKoqo
DbH81TVCVW1tKqWLLzoT21I0uY00rBIsi1dUPzvqpca7667V6i/V3b2i8H9uhNvri1TdYuSZomozwbKsB
HgAur6l29QcuBsTOalgFf7LW/oJ0Vtr9wXdtddTpWYJf7cD2ga1NkjRNRnnn2McBzwfOS/Lj1vYPwNuA0
5lcCVxG96NKAf8GDgFWAb8FXgd7dCTvBk4p413bLtFuiRpmowsLKrqU0AmGfyECcYv4OWTzOtE4MTN
V50kaVN4BbckaZBhIUkaZFhIkgYZFpKkQYaFJGmQYSFJGmRYSJIGGRaSpEGGhSRpkGEhSRpkWEiSBhkW
kqRBhoUkaZBhIUkaZFhIkgYZFpKkQYaFJGmQYSFJGmRYSJIGGRaSpEGGhSRpkGEhSRpkWEiSBhkWkqR
BhoUkaZBhIUkaZFhIkgYZFpKkQYaFJGmQYSFJGmRYSJIGGRaSpEGGhSRpkGEhSRo0srBlcmKstUl+2mVb
OckZSS5qfxe09iR5b5JVSc5N8qjeNMva+BclWTAqeiVJkxvllsVJwMHj2l4PfL2q9gK+3voBngTs1R5HAR+CL
lyAo4F9gX2Ao8cCRpI0fUYWfIX1bWD9uOZDgZNB98nAYb32U6pzFrBTkt2Ag4Azqmp9VW0AzuD2ASRJ
GrHpPmaxa1Vd2bqvAnZt3YuA1b3x1rS2ydpvJ8IRSvYkWBfu3brNW7UkzXEzdoC7qgqozTi/E6ppaVUtX
bhW4eaarSSJ6Q+Lq9vuJdrfta39cmCP3niLW9tk7ZKkaTTdYbEcGDujaRnwxV77C9pZUfsB17XdVacDBYZZ
0A5sH9jaJEnTaP6oZpzk8D+wC5J1tCd1fQ24LQkRwKXAc9qo38ZOARYBfwWeCFaVa1P8mbgnDbesVU
1/qC5JGnERhYwVfWcSQY9YYJxC3j5JPM5ETHxM5YmSdpEXsEtSRpkWEiSBhkWkqRBhoUkaZBhIUkaNL
KzoSSNxi+PfehMl6BZ6E/eeN5I5++WhSRpkGEhSRpkWEiSBhkWkqRBhoUkaZBhIUkaZFhIkgYZFpKkQYa
FJGmQYSFJGmRYSJIGGRaSpEGGhSRpkGEhSRpkWEiSBhkWkqRBhoUkaZBhIUkaZFhIkgYZFpKkQYaFJG
mQYSFJGmRYSJIGGRaSpEGGhSRpkGEhSRpkWEiSBhkWkqRBW0xYJDk4yc+TrEry+pmuR5Lmki0iLJLMA
z4APAnYG3hOkr1ntipJmju2iLAA9gFWVdXfVfx/gFOBQ2e4JkmaM+bPdAFTtAhY3etfA+zbHyHJUcBRrfe
GJD+fptrmgI2AX810EbNB3rlspkvQbfneHHN0Nsdc7jvZgC0LLAZV1QnACTNdx9YoyYqqWjrTdjUjj+d6cPlv
KbqjLgT16/YtbmyRpGmwpYXEOsFeSPZNSbxwOLJ/hmiRpztgidkNV1U1JXgGcDswDTqyq82e4rLnE3Xua
rXxvTpNU1UzXIema5baU3VCSpBlkWEiSBhkWc1iSJUI+OtN1Sjr9DAJ0iDDQvOS/GuS85N8Ncldk7wkyT
IjfpLks0nuBpDkpCQfSnJWkouT7J/kxCQXJlphpdDW7gkd0/y7+1999Mkz05yaZl/SnJekrOT3L+N+9QkP0

jyoyRfS7Jraz8myclJvpPksiRP703/ISTbzuxSbrkMC+0FfKCqHgxcC/w18LmqekxVPRy4EDiyN/4C4LHAf6O
71uV44MHAQ5M8Ylor19bmYOCKqnp4VT0E+Eprv66qHqg8H3h3a/susF9VPZLuXnGv7c3nfsABwNOAj
wNntulvBJ48+sXYOhkWuqSqfty6VwJLgle0b2bnAc+lC4Mx/1bd+dbnAVdX1XIVdQtwfptWuqPOA56Y5
O1J/ryqrmvtn+z9fWzrXgyc3t6jr+G279H/qKo/tPnN49bQOQ/fo3eYYaHf97pvprrtQ8yTgFe3b2JuA7ScY/
5Zx097CFnKRp2anqvoF8Ci6D/XjkrxxbFB/tPb3fcD723v0pUzwHm1fYv5Qt15M5nv0TjAsNJEgSvb/t3nz
nQxmhuS7A78tqo+DryDLjgAnt37+/3WfU9uvT+ctwKeBqasJvl/gR8A69rfHWe2HMOReDwXekeQW4A/A
y4DPAAuSnEu3xfCcNu4xwKeTbAC+Aew5/eXOLd7uQ9KsleRSYGIV+ZsVM8zdUJKkQW5ZSJIGuWUhSRp
kWEiSBhkWkqRBhoV0JyS5YRPGPSbJq0c1f2mUDAtJ0iDDQtrMJrsjavPwJN9PclGSI/SmeU270++5Sd40A
2VLG2VYSJvfxu6I+jC6O6I+Fnhjkt2THEh39999gEcAj07yF9Ncs7RR3u5D2vwWA59KshuwHXBjb9gXq+p
G4MYkZ9IFxOOBA4EftXF2oAuPb09fydLGGRbS5vc+4F1VtTzJ/nT3MRoz/irYAgK8tar+ZXrKkzadu6GkzW
9jd0Q9NMn2Se4F7A+cA5wOvCjJDgBJFiW593QVK02FWxbSnXO3JGt6/e9i43dEPRc4E9gFeHNvXQFcke
RBwPeTANwAPA9YO/rypanx3lCSpEHuhplkDTIsJEmDDAtJ0iDDQpl0yLCQJA0yLCRJgwwLSdKg/w+zxO5
JjPQAtAAAAABJRU5ErkJggg==\n"

```
    },
    "metadata": {
      "needs_background": "light"
    }
  }
]
},
{
  "cell_type": "code",
  "source": [
    "X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)"
  ],
  "metadata": {
    "id": "KBLBUSHgiGR6"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "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 = pad_sequences(sequences,maxlen=max_len)"
],
"metadata": {
  "id": "GD0PDa_6iJ5Q"
},
"execution_count": null,
"outputs": []
},
{
  "cell_type": "markdown",
  "source": [
    "# **4. CREATE MODEL**"
  ],
  "metadata": {
    "id": "voU0xeukjy2s"
  }
},
{
  "cell_type": "code",
  "source": [
    "def RNN():\n",
    "    inputs = Input(name='inputs',shape=[max_len])\n",
    "    layer = Embedding(max_words,50,input_length=max_len)(inputs)\n",
    "    layer = LSTM(64)(layer)\n",
    "    layer = Dense(256,name='FC1')(layer)\n",
    "    layer = Activation('relu')(layer)\n",
    "    layer = Dropout(0.5)(layer)"
  ]
}

```



```

" layer = Dense(1,name='out_layer')(layer)\n",
" layer = Activation('sigmoid')(layer)\n",
" model = Model(inputs=inputs,outputs=layer)\n",
" return model"
],
"metadata": {
  "id": "19_2oCf7iMnG"
},
"execution_count": null,
"outputs": []
},
{
  "cell_type": "markdown",
  "source": [
    "# **5. LSTM LAYERS**"
  ],
  "metadata": {
    "id": "H16MEIYlj-6U"
  }
},
{
  "cell_type": "code",
  "source": [
    "model = RNN()\n",
    "model.summary()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "IoSbYBeSiO71",

```

"outputId": "415360da-252f-4409-e92f-e69b94218c38"

},

"execution_count": null,

"outputs": [

{

 "output_type": "stream",

 "name": "stdout",

 "text": [

 "Model: \"model\\\"\\n",

 "_____\\n",

 " Layer (type) Output Shape Param # \\n",

 "=====\\n",

 " inputs (InputLayer) [(None, 150)] 0 \\n",

 " \\n",

 " embedding (Embedding) (None, 150, 50) 50000 \\n",

 " \\n",

 " lstm (LSTM) (None, 64) 29440 \\n",

 " \\n",

 " FC1 (Dense) (None, 256) 16640 \\n",

 " \\n",

 " activation (Activation) (None, 256) 0 \\n",

 " \\n",

 " dropout (Dropout) (None, 256) 0 \\n",

 " \\n",

 " out_layer (Dense) (None, 1) 257 \\n",

 " \\n",

 " activation_1 (Activation) (None, 1) 0 \\n",

 " \\n",

 "=====\\n",

 "Total params: 96,337\\n",

 "Trainable params: 96,337\\n",

```

    "Non-trainable params: 0\n",
    "_____\\n"
  ]
}
]
},
{
  "cell_type": "markdown",
  "source": [
    "# **6. COMPILING THE MODEL**"
  ],
  "metadata": {
    "id": "AoLKzOk7kKUI"
  }
},
{
  "cell_type": "code",
  "source": [
    "model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])"
  ],
  "metadata": {
    "id": "M6YvLYRPiVKC"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "markdown",
  "source": [
    "# **7. FIT THE MODEL**"
  ],

```

```

"metadata": {
  "id": "oJ9_Px0ykPR3"
},
{
  "cell_type": "code",
  "source": [

"model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.0001)])"

],
"metadata": {
  "id": "fRyicKkgiYRk",
  "colab": {
    "base_uri": "https://localhost:8080/"
  },
  "outputId": "fa35d0e8-8207-4afc-cd6d-ee9c78221a5a"
},
"execution_count": 11,
"outputs": [
  {
    "output_type": "stream",
    "name": "stdout",
    "text": [
      "Epoch 1/10\n",
      "30/30 [=====] - 7s 229ms/step - loss: 0.0452 - accuracy: 0.9873 - val_loss: 0.0302 - val_accuracy: 0.9895\n",
      "Epoch 2/10\n",
      "30/30 [=====] - 7s 230ms/step - loss: 0.0344 - accuracy: 0.9902 - val_loss: 0.0328 - val_accuracy: 0.9916\n"
    ]
  },

```

```
{
  "output_type": "execute_result",
  "data": {
    "text/plain": [
      "<keras.callbacks.History at 0x7fa5b3a662d0>"
    ]
  },
  "metadata": {},
  "execution_count": 11
}
],
{
  "cell_type": "markdown",
  "source": [
    "# **8. SAVE THE MODEL**"
  ],
  "metadata": {
    "id": "PF-_P9vUkVzf"
  }
},
{
  "cell_type": "code",
  "source": [
    "model.save('Spam.h5')"
  ],
  "metadata": {
    "id": "E_ThW_8Rih3M"
  },
  "execution_count": 12,
  "outputs": []
}
```

```
},
{
  "cell_type": "markdown",
  "source": [
    "# **9. TEST THE MODEL**"
  ],
  "metadata": {
    "id": "5RN24hYukYXD"
  }
},
{
  "cell_type": "code",
  "source": [
    "test_sequences = tok.texts_to_sequences(X_test)\n",
    "test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)\n",
    "test_sequences_matrix"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "zWAJtip4i3Ip",
    "outputId": "19759c31-6a03-4d24-fa11-30fc97e7e7f6"
  },
  "execution_count": 13,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "array([[ 0,  0,  0, ..., 45, 44, 197],\n",
```

```

" [ 0, 0, 0, ..., 455, 56, 106],\n",
" [ 0, 0, 0, ..., 2, 171, 41],\n",
" ..., \n",
" [ 0, 0, 0, ..., 59, 170, 718],\n",
" [ 0, 0, 0, ..., 245, 11, 269],\n",
" [ 0, 0, 0, ..., 153, 267, 224]], dtype=int32)"
]
},
"metadata": {},
"execution_count": 13
}
]
},
{
"cell_type": "code",
"source": [
"accr = model.evaluate(test_sequences_matrix,Y_test)\n",
"print('Accuracy:',accr[1])\n",
"print('Loss:',accr[0])"
],
"metadata": {
"colab": {
"base_uri": "https://localhost:8080/"
},
"id": "rXcmXfBKj7",
"outputId": "1457c9a5-08b5-45d8-a0f2-29087cd9da0c"
},
"execution_count": 14,
"outputs": [
{
"output_type": "stream",

```

```
"name": "stdout",
"text": [
  "27/27 [=====] - 1s 21ms/step - loss: 0.0664 - accuracy:
0.9821\n",
  "Accuracy: 0.9820573925971985\n",
  "Loss: 0.06643393635749817\n"
]
}
]
}
]
}
```