```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "markdown",
      "source": [
        "##IMPORT LIBRARIES"
      ],
      "metadata": {
        "id": "LzQq1eTTHyi3"
      }
    },
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "iA3hludMrP6u",
        "outputId": "8868bade-6029-4c9e-8c23-64424b2b24c1"
      },
      "outputs": [
        {
          "output_type": "stream",
          "name": "stderr",
          "text": [
            "[nltk_data] Downloading package stopwords to /root/nltk_data...\n",
            "[nltk_data]   Unzipping corpora/stopwords.zip.\n"
          ]
        }
      ],
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import nltk\n",
        "import re\n",
        "\n",
        "nltk.download('stopwords')\n",
        "\n",
        "from nltk.corpus import stopwords\n",
        "from nltk.stem.porter import PorterStemmer"
      ]
    },
```

```
{
   "cell_type": "markdown",
   "source": [
      "##LOAD DATASET"
   ],
   "metadata": {
      "id": "_5ocKHanHwk5"
   }
},
{
   "cell_type": "code",
   "source": [
      "a = pd.read_csv('/content/spam.csv',encoding='ISO-8859-1')\n",
      "a.head()"
   ],
   "metadata": {
      "colab": {
         "base_uri": "https://localhost:8080/",
         "height": 206
      },
      "id": "H89HzIp_0HLJ",
      "outputId": "b499f656-d75c-4b0c-e2f6-7b8332b4c109"
   },
   "execution_count": 2,
   "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-c81351b7-0702-41a9-988e-0f36162fc87d\">\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-
c81351b7-0702-41a9-988e-0f36162fc87d')\"\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 2.06-.94-2.06-.94-.94-2.06-.94
2.06-2.06.94zm-11 1L8.5 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 2.06-.94-2.06-.94-.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.39.39.9.59 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-c81351b7-0702-41a9-988e-
0f36162fc87d 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-c81351b7-0702-
41a9-988e-0f36162fc87d');\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": "code",
    "source": [
        "a=a[['v1','v2']]\n",
        "a.head()"
    ],
    "metadata": {
        "colab": {
            "base_uri": "https://localhost:8080/",
            "height": 206
        },
        "id": "b3P-R4a74ikf",
```

```
    "outputId": "54b74cbb-1a08-4cff-c2c1-9e1e2cb8b651"
  },
  "execution_count": 3,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "        v1                                                                  v2\n",
          "0     ham   Go until jurong point, crazy.. Available only ...\n",
          "1     ham                                   Ok lar... Joking wif u oni...\n",
          "2    spam   Free entry in 2 a wkly comp to win FA Cup fina...\n",
          "3     ham   U dun say so early hor... U c already then say...\n",
          "4     ham   Nah I don't think he goes to usf, he lives aro..."
        ],
        "text/html": [
          "\n",
          "  <div id=\"df-f401d75c-5a4d-4271-a16e-f0e11134e82a\">\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",
          "    </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",
          "    </tr>\n",
          "    <tr>\n",
          "      <th>1</th>\n",
          "      <td>ham</td>\n",
          "      <td>Ok lar... Joking wif u oni...</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",
          "    </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",
          "    </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",
          "    </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",
"      </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",
"      </tr>\n",
"    </tbody>\n",
"</table>\n",
"</div>\n",
"          <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-
f401d75c-5a4d-4271-a16e-f0e11134e82a')\"\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 2.06-.94-2.06-.94-.94-2.06-.94
2.06-2.06.94zm-11 1L8.5 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 2.06-.94-2.06-.94-.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.39.39.9.59 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-f401d75c-5a4d-4271-a16e-
f0e11134e82a 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-f401d75c-5a4d-
4271-a16e-f0e11134e82a');\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": 3
        }
    ]
},
{
    "cell_type": "code",
    "source": [
        "a.shape"
    ],
```

```
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "KjtGriOz6h9Q",
        "outputId": "3c010e77-468b-404c-9499-35b284229d74"
      },
      "execution_count": 4,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(5572, 2)"
            ]
          },
          "metadata": {},
          "execution_count": 4
        }
      ]
    },
    {
      "cell_type": "markdown",
      "source": [
        "##Text processing (NLP)"
      ],
      "metadata": {
        "id": "3QZ2twu7IOSP"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "ps=PorterStemmer()\n",
        "message=[]\n",
        "for i in range(0,5572):\n",
        "    msg=a['v2'][i]\n",
        "    msg=re.sub('[^a-zA-Z]',' ',msg)\n",
        "    msg=msg.lower()\n",
        "    msg=msg.split(' ')\n",
        "            msg = [ps.stem(word) for word in msg if word not in
set(stopwords.words('english'))]\n",
        "    msg=' '.join(msg)\n",
        "    message.append(msg)\n",
        "\n",
        "message[:6]"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "ywCU6b1F6yp8",
        "outputId": "ed1ab128-8b55-4b66-8c4c-723b5b364750"
      },
      "execution_count": 5,
      "outputs": [
        {
```

```
            "output_type": "execute_result",
            "data": {
              "text/plain": [
                "['go jurong point    crazi      avail bugi n great world la e buffet        cine got
amor wat      ',\n",
                " 'ok lar       joke wif u oni      ',\n",
                " 'free entri       wkli comp win fa cup final tkt       st may              text fa
receiv entri question std txt rate c appli                    ',\n",
                " 'u dun say earli hor       u c alreadi say      ',\n",
                " 'nah think goe usf    live around though',\n",
                " 'freemsg hey darl       week word back     like fun still     tb ok     xxx std chg
send              rcv']"
              ]
            },
            "metadata": {},
            "execution_count": 5
          }
        ]
      },
      {
        "cell_type": "code",
        "source": [
          "from sklearn.feature_extraction.text import CountVectorizer\n",
          "\n",
          "cv = CountVectorizer()\n",
          "x = cv.fit_transform(message).toarray()\n",
          "x"
        ],
        "metadata": {
          "id": "Sd1OR4rj2HeQ",
          "colab": {
            "base_uri": "https://localhost:8080/"
          },
          "outputId": "461a2c68-44a2-4662-cb06-f7fd4e79342d"
        },
        "execution_count": 6,
        "outputs": [
          {
            "output_type": "execute_result",
            "data": {
              "text/plain": [
                "array([[0, 0, 0, ..., 0, 0, 0],\n",
                "       [0, 0, 0, ..., 0, 0, 0],\n",
                "       [0, 0, 0, ..., 0, 0, 0],\n",
                "       ...,\n",
                "       [0, 0, 0, ..., 0, 0, 0],\n",
                "       [0, 0, 0, ..., 0, 0, 0],\n",
                "       [0, 0, 0, ..., 0, 0, 0]])"
              ]
            },
            "metadata": {},
            "execution_count": 6
          }
        ]
      },
      {
        "cell_type": "code",
```

```
    "source": [
      "#LABEL ENCODING\n",
      "\n",
      "from sklearn.preprocessing import LabelEncoder\n",
      "le = LabelEncoder()\n",
      "\n",
      "\n",
      "a['v1']=le.fit_transform(a['v1'])\n",
      "y = a['v1'].values\n",
      "y\n"
    ],
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "id": "n_5xyJst_Js4",
      "outputId": "9904c8d7-cd02-444b-9e45-fddbc378b2b6"
    },
    "execution_count": 7,
    "outputs": [
      {
        "output_type": "execute_result",
        "data": {
          "text/plain": [
            "array([0, 0, 1, ..., 0, 0, 0])"
          ]
        },
        "metadata": {},
        "execution_count": 7
      }
    ]
  },
  {
    "cell_type": "markdown",
    "source": [
      "##MODEL BUILDIND"
    ],
    "metadata": {
      "id": "e2aIZkbmIYLY"
    }
  },
  {
    "cell_type": "code",
    "source": [
      "from tensorflow.keras.models import Sequential\n",
      "from tensorflow.keras.layers import Dense\n",
      "\n",
      "model = Sequential()\n",
      "model.add(Dense(1550,activation='relu'))\n",
      "model.add(Dense(3000,activation='relu'))\n",
      "model.add(Dense(1,activation='sigmoid'))\n",
      "\n",
      "model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])\n",
      "\n",
      "\n",
      "\n",
      "model.fit(x,y,epochs=10)"
```

```
        ],
        "metadata": {
          "colab": {
            "base_uri": "https://localhost:8080/"
          },
          "id": "8o4cvxMz_f7b",
          "outputId": "09395d16-49ca-4f0b-f016-0e1e58391348"
        },
        "execution_count": 8,
        "outputs": [
          {
            "output_type": "stream",
            "name": "stdout",
            "text": [
              "Epoch 1/10\n",
              "175/175 [==============================] - 19s 105ms/step - loss:
0.1128 - accuracy: 0.9646\n",
              "Epoch 2/10\n",
              "175/175 [==============================] - 18s 105ms/step - loss:
0.0131 - accuracy: 0.9968\n",
              "Epoch 3/10\n",
              "175/175 [==============================] - 18s 105ms/step - loss:
0.0013 - accuracy: 0.9996\n",
              "Epoch 4/10\n",
              "175/175 [==============================] - 18s 104ms/step - loss:
1.9955e-04 - accuracy: 1.0000\n",
              "Epoch 5/10\n",
              "175/175 [==============================] - 18s 104ms/step - loss:
8.9791e-05 - accuracy: 1.0000\n",
              "Epoch 6/10\n",
              "175/175 [==============================] - 18s 105ms/step - loss:
5.2074e-05 - accuracy: 1.0000\n",
              "Epoch 7/10\n",
              "175/175 [==============================] - 18s 105ms/step - loss:
3.3522e-05 - accuracy: 1.0000\n",
              "Epoch 8/10\n",
              "175/175 [==============================] - 18s 105ms/step - loss:
2.3012e-05 - accuracy: 1.0000\n",
              "Epoch 9/10\n",
              "175/175 [==============================] - 18s 105ms/step - loss:
1.6572e-05 - accuracy: 1.0000\n",
              "Epoch 10/10\n",
              "175/175 [==============================] - 18s 105ms/step - loss:
1.2497e-05 - accuracy: 1.0000\n"
            ]
          },
          {
            "output_type": "execute_result",
            "data": {
              "text/plain": [
                "<keras.callbacks.History at 0x7f71945e6090>"
              ]
            },
            "metadata": {},
            "execution_count": 8
          }
        ]
```

```
    },
    {
      "cell_type": "markdown",
      "source": [
        "##SAVE THE MODEL"
      ],
      "metadata": {
        "id": "JPP3PT-5JNE5"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "model.save('spam-NLP.h5')"
      ],
      "metadata": {
        "id": "rSbfCUhYDaXK"
      },
      "execution_count": 9,
      "outputs": []
    },
    {
      "cell_type": "markdown",
      "source": [
        "##TEST THE MODEL"
      ],
      "metadata": {
        "id": "2RBqa6BrJIdb"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "msg='FREE MESSAGE Activate your 500 FREE Text Messages by replying to this message with the word FREE'\n",
        "print('THE ORIGINAL MESSAGE IS:    ',msg)\n",
        "msg=re.sub('[^a-zA-Z]',' ',msg)\n",
        "msg=msg.lower()\n",
        "msg=msg.split(' ')\n",
        "msg = [ps.stem(word) for word in msg if word not in set(stopwords.words('english'))]\n",
        "msg=' '.join(msg)\n",
        "print('THE STEMMED MESSAGE IS:    ',msg)\n",
        "    \n",
        "predict = model.predict(cv.transform([msg]))\n",
        "if predict > 0.5:\n",
        "    pred='SPAM'\n",
        "else: pred='NOT SPAM'\n",
        "print('THE MESSAGE IS PREDICTED AS:    ',pred)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "qVmO1KoR_f0K",
        "outputId": "54e4c145-4694-49e3-ee01-21067e081cfb"
      },
```

```
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "THE ORIGINAL MESSAGE IS:    FREE MESSAGE Activate your 500 FREE Text Messages by replying to this message with the word FREE\n",
            "THE STEMMED MESSAGE IS:    free messag activ        free text messag repli messag word free\n",
            "1/1 [==============================] - 0s 158ms/step\n",
            "THE MESSAGE IS PREDICTED AS:    SPAM\n"
          ]
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "msg='Wishing you and your family Merry \\X\\\\" mas and HAPPY NEW Year in advance..\"'\n",
        "print('THE ORIGINAL MESSAGE IS:  ',msg)\n",
        "msg=re.sub('[^a-zA-Z]',' ',msg)\n",
        "msg=msg.lower()\n",
        "msg=msg.split(' ')\n",
        "msg = [ps.stem(word) for word in msg if word not in set(stopwords.words('english'))]\n",
        "msg=' '.join(msg)\n",
        "print('THE STEMMED MESSAGE IS:  ',msg)\n",
        "  \n",
        "predict = model.predict(cv.transform([msg]))\n",
        "if predict > 0.5:\n",
        "  pred='spam'\n",
        "else: pred='NOT SPAM'\n",
        "print('THE MESSAGE IS PREDICTED AS:  ',pred)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "1NxzMhEwGnuw",
        "outputId": "c3e245e4-1330-4bd5-9c39-3a87c48c0265"
      },
      "execution_count": 11,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "THE ORIGINAL MESSAGE IS:    Wishing you and your family Merry \\X\" mas and HAPPY NEW Year in advance..\"\n",
            "THE STEMMED MESSAGE IS:    wish famili merri  x  ma happi new year advanc  \n",
            "1/1 [==============================] - 0s 9ms/step\n",
            "THE MESSAGE IS PREDICTED AS:    NOT SPAM\n"
          ]
        }
      ]
```

```
        ]
      }
    ]
}
```