```
{
 "cells": [
  {
   "cell_type": "markdown",
   "id": "a2ac5e63",
   "metadata": {},
   "source": [
    "## Importing Libraries & Dataset"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 1,
   "id": "dc923180",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "import seaborn as sns"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "id": "6ac5cbab",
   "metadata": {},
   "outputs": [],
   "source": [
   "data=pd.read_csv(\"D:/Collection Of Dataset/dataset_website.csv\")"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 3,
  "id": "689e13b2",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/html": [
      "<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>index</th>\n",
"     <th>having_IPhaving_IP_Address</th>\n",
"     <th>URLURL_Length</th>\n",
"     <th>Shortining_Service</th>\n",
"     <th>having_At_Symbol</th>\n",
"     <th>double_slash_redirecting</th>\n",
"     <th>Prefix_Suffix</th>\n",
"     <th>having_Sub_Domain</th>\n",
"     <th>SSLfinal_State</th>\n",
"     <th>Domain_registeration_length</th>\n",
"     <th>...</th>\n",
"     <th>popUpWidnow</th>\n",
"     <th>Iframe</th>\n",
"     <th>age_of_domain</th>\n",
"     <th>DNSRecord</th>\n",
"     <th>web_traffic</th>\n",
"     <th>Page_Rank</th>\n",
"     <th>Google_Index</th>\n",
"     <th>Links_pointing_to_page</th>\n",
"     <th>Statistical_report</th>\n",
"     <th>Result</th>\n",
"   </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"   <tr>\n",
"     <th>0</th>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>2</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"     <td>0</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>0</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"     <th>2</th>\n",
"     <td>3</td>\n",
"     <td>1</td>\n",
"     <td>0</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>0</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>3</th>\n",
"     <td>4</td>\n",
"     <td>1</td>\n",
"     <td>0</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>...</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>4</th>\n",
"     <td>5</td>\n",
"     <td>1</td>\n",
"     <td>0</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>0</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"   </tr>\n",
"    <tr>\n",
"      <th>...</th>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>11050</th>\n",
"      <td>11051</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>11051</th>\n",
"     <td>11052</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"     <th>11052</th>\n",
"     <td>11053</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>0</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>11053</th>\n",
"     <td>11054</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>...</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>11054</th>\n",
"      <td>11055</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>11055 rows × 32 columns</p>\n",
"</div>"
],
"text/plain": [
"       index  having_IPhaving_IP_Address  URLURL_Length  Shortining_Service  \\\n",
"0          1                          -1              1                   1   \n",
"1          2                           1              1                   1   \n",
"2          3                           1              0                   1   \n",
"3          4                           1              0                   1   \n",
"4          5                           1              0                  -1   \n",
"...      ...                         ...            ...                 ...   \n",
"11050  11051                           1             -1                   1   \n",
"11051  11052                          -1              1                   1   \n",
"11052  11053                           1             -1                   1   \n",
"11053  11054                          -1             -1                   1   \n",
"11054  11055                          -1             -1                   1   \n",
"\n",
"       having_At_Symbol  double_slash_redirecting  Prefix_Suffix  \\\n",
"0                     1                        -1             -1   \n",
"1                     1                         1             -1   \n",
"2                     1                         1             -1   \n",
"3                     1                         1             -1   \n",
"4                     1                         1             -1   \n",
"...                 ...                       ...            ...   \n",
"11050                -1                         1              1   \n",
"11051                -1                        -1             -1   \n",
"11052                 1                         1             -1   \n",
"11053                 1                         1             -1   \n",
"11054             1                    1          -1  \n",
"\n",
"       having_Sub_Domain  SSLfinal_State  Domain_registeration_length  ...  \\\n",
"0                     -1              -1                           -1  ...   \n",
"1                      0               1                           -1  ...   \n",
"2                     -1              -1                           -1  ...   \n",
"3                     -1              -1                            1  ...   \n",
"4                      1               1                           -1  ...   \n",
"...                  ...             ...                          ...  ...   \n",
"11050                 1               1                           -1  ...   \n",
"11051                 1              -1                           -1  ...   \n",
"11052                 1              -1                           -1  ...   \n",
"11053                -1              -1                            1  ...   \n",
"11054                -1              -1                            1  ...   \n",
"\n",
"       popUpWidnow  Iframe  age_of_domain  DNSRecord  web_traffic  Page_Rank  \\\n",
"0                1       1             -1         -1           -1         -1   \n",
"1                1       1             -1         -1            0         -1   \n",
"2                1       1              1         -1            1         -1   \n",
"3                1       1             -1         -1            1         -1   \n",
"4               -1       1             -1         -1            0         -1   \n",
"...            ...     ...            ...        ...          ...        ...   \n",
"11050          -1      -1              1          1           -1         -1   \n",
"11051          -1       1              1          1            1          1   \n",
"11052           1       1              1          1            1         -1   \n",
"11053          -1       1              1          1            1         -1   \n",
"11054           1       1             -1          1           -1         -1   \n",
"\n",
"       Google_Index  Links_pointing_to_page  Statistical_report  Result  \n",
    "0              1                   1              -1    -1  \n",
    "1              1                   1               1    -1  \n",
    "2              1                   0              -1    -1  \n",
    "3              1                  -1               1    -1  \n",
    "4              1                   1               1     1  \n",
    "...           ...                 ...             ...   ...  \n",
    "11050          1                   1               1     1  \n",
    "11051          1                  -1               1    -1  \n",
    "11052          1                   0               1    -1  \n",
    "11053          1                   1               1    -1  \n",
    "11054         -1                   1              -1    -1  \n",
    "\n",
    "[11055 rows x 32 columns]"
   ]
  },
  "execution_count": 3,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "data"
]
},
{
"cell_type": "code",
"execution_count": 4,
"id": "72101b49",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/html": [
    "<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>index</th>\n",
    "      <th>having_IPhaving_IP_Address</th>\n",
    "      <th>URLURL_Length</th>\n",
    "      <th>Shortining_Service</th>\n",
    "      <th>having_At_Symbol</th>\n",
    "      <th>double_slash_redirecting</th>\n",
    "      <th>Prefix_Suffix</th>\n",
"     <th>having_Sub_Domain</th>\n",
"     <th>SSLfinal_State</th>\n",
"     <th>Domain_registeration_length</th>\n",
"     <th>...</th>\n",
"     <th>popUpWidnow</th>\n",
"     <th>Iframe</th>\n",
"     <th>age_of_domain</th>\n",
"     <th>DNSRecord</th>\n",
"     <th>web_traffic</th>\n",
"     <th>Page_Rank</th>\n",
"     <th>Google_Index</th>\n",
"     <th>Links_pointing_to_page</th>\n",
"     <th>Statistical_report</th>\n",
"     <th>Result</th>\n",
"   </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"   <tr>\n",
"     <th>0</th>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"     <th>1</th>\n",
"     <td>2</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>0</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>...</td>\n",
"     <td>1</td>\n",
"     <td>1</td>\n",
"     <td>-1</td>\n",
"     <td>-1</td>\n",
"      <td>0</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>3</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>-1</td>\n",
"     <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
"      <td>4</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>4</th>\n",
"      <td>5</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>...</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>-1</td>\n",
"      <td>-1</td>\n",
"      <td>0</td>\n",
"      <td>-1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>5 rows × 32 columns</p>\n",
"</div>"
],
"text/plain": [
"   index  having_IPhaving_IP_Address  URLURL_Length  Shortining_Service  \\\n",
"0      1                          -1              1                   1   \n",
"1     2                      1           1              1  \n",
"2     3                      1           0              1  \n",
"3     4                      1           0              1  \n",
"4     5                      1           0             -1  \n",
"\n",
"   having_At_Symbol  double_slash_redirecting  Prefix_Suffix  \\\n",
"0             1                   -1          -1  \n",
"1             1                    1          -1  \n",
"2             1                    1          -1  \n",
"3             1                    1          -1  \n",
"4             1                    1          -1  \n",
"\n",
"   having_Sub_Domain  SSLfinal_State  Domain_registeration_length  ...  \\\n",
"0             -1          -1                     -1  ...  \n",
"1              0           1                     -1  ...  \n",
"2             -1          -1                     -1  ...  \n",
"3             -1          -1                      1  ...  \n",
"4              1           1                     -1  ...  \n",
"\n",
"   popUpWidnow  Iframe  age_of_domain  DNSRecord  web_traffic  Page_Rank  \\\n",
"0         1      1          -1       -1         -1       -1  \n",
"1         1      1          -1       -1          0       -1  \n",
"2         1      1           1       -1          1       -1  \n",
"3         1      1          -1       -1          1       -1  \n",
"4        -1      1          -1       -1          0       -1  \n",
"\n",
"   Google_Index  Links_pointing_to_page  Statistical_report  Result  \n",
"0          1                  1              -1     -1 \n",
"1          1                  1               1     -1 \n",
    "2          1                   0              -1    -1  \n",
    "3          1                  -1               1    -1  \n",
    "4          1                   1               1     1  \n",
    "\n",
    "[5 rows x 32 columns]"
   ]
  },
  "execution_count": 4,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "data.head()"
]
},
{
"cell_type": "markdown",
"id": "131dd287",
"metadata": {},
"source": [
 "### Numerical Analysis"
]
},
{
"cell_type": "code",
"execution_count": 5,
"id": "5250f670",
"metadata": {},
 },
 "outputs": [
  {
   "data": {
    "text/plain": [
     "(11055, 32)"
    ]
   },
   "execution_count": 5,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "data.shape"
 ]
},
{
 "cell_type": "code",
 "execution_count": 6,
 "id": "b4f45d4a",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "353760"
    ]
   },
   "execution_count": 6,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "data.size"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 7,
  "id": "1cbd8317",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "<class 'pandas.core.frame.DataFrame'>\n",
     "RangeIndex: 11055 entries, 0 to 11054\n",
     "Data columns (total 32 columns):\n",
     " #   Column                       Non-Null Count  Dtype\n",
     "---  ------                       --------------  -----\n",
     " 0   index                        11055 non-null  int64\n",
     " 1   having_IPhaving_IP_Address   11055 non-null  int64\n",
     " 2   URLURL_Length                11055 non-null  int64\n",
     " 3   Shortining_Service           11055 non-null  int64\n",
     " 4   having_At_Symbol             11055 non-null  int64\n",
     " 5   double_slash_redirecting     11055 non-null  int64\n",
" 6   Prefix_Suffix               11055 non-null  int64\n",
" 7   having_Sub_Domain           11055 non-null  int64\n",
" 8   SSLfinal_State              11055 non-null  int64\n",
" 9   Domain_registeration_length  11055 non-null  int64\n",
" 10  Favicon                     11055 non-null  int64\n",
" 11  port                        11055 non-null  int64\n",
" 12  HTTPS_token                 11055 non-null  int64\n",
" 13  Request_URL                 11055 non-null  int64\n",
" 14  URL_of_Anchor               11055 non-null  int64\n",
" 15  Links_in_tags               11055 non-null  int64\n",
" 16  SFH                         11055 non-null  int64\n",
" 17  Submitting_to_email         11055 non-null  int64\n",
" 18  Abnormal_URL                11055 non-null  int64\n",
" 19  Redirect                    11055 non-null  int64\n",
" 20  on_mouseover                11055 non-null  int64\n",
" 21  RightClick                  11055 non-null  int64\n",
" 22  popUpWidnow                 11055 non-null  int64\n",
" 23  Iframe                      11055 non-null  int64\n",
" 24  age_of_domain               11055 non-null  int64\n",
" 25  DNSRecord                   11055 non-null  int64\n",
" 26  web_traffic                 11055 non-null  int64\n",
" 27  Page_Rank                   11055 non-null  int64\n",
" 28  Google_Index                11055 non-null  int64\n",
" 29  Links_pointing_to_page      11055 non-null  int64\n",
" 30  Statistical_report          11055 non-null  int64\n",
" 31  Result                      11055 non-null  int64\n",
"dtypes: int64(32)\n",
"memory usage: 2.7 MB\n"
]
```

```
   }
  ],
  "source": [
   "data.info()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 8,
  "id": "9110f6e1",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/html": [
      "<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>index</th>\n",
"      <th>having_IPhaving_IP_Address</th>\n",
"      <th>URLURL_Length</th>\n",
"      <th>Shortining_Service</th>\n",
"      <th>having_At_Symbol</th>\n",
"      <th>double_slash_redirecting</th>\n",
"      <th>Prefix_Suffix</th>\n",
"      <th>having_Sub_Domain</th>\n",
"      <th>SSLfinal_State</th>\n",
"      <th>Domain_registeration_length</th>\n",
"      <th>...</th>\n",
"      <th>popUpWidnow</th>\n",
"      <th>Iframe</th>\n",
"      <th>age_of_domain</th>\n",
"      <th>DNSRecord</th>\n",
"      <th>web_traffic</th>\n",
"      <th>Page_Rank</th>\n",
"      <th>Google_Index</th>\n",
"      <th>Links_pointing_to_page</th>\n",
"      <th>Statistical_report</th>\n",
"      <th>Result</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>count</th>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>...</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"      <td>11055.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>mean</th>\n",
"      <td>5528.000000</td>\n",
"      <td>0.313795</td>\n",
"      <td>-0.633198</td>\n",
"      <td>0.738761</td>\n",
"     <td>0.700588</td>\n",
"     <td>0.741474</td>\n",
"     <td>-0.734962</td>\n",
"     <td>0.063953</td>\n",
"     <td>0.250927</td>\n",
"     <td>-0.336771</td>\n",
"     <td>...</td>\n",
"     <td>0.613388</td>\n",
"     <td>0.816915</td>\n",
"     <td>0.061239</td>\n",
"     <td>0.377114</td>\n",
"     <td>0.287291</td>\n",
"     <td>-0.483673</td>\n",
"     <td>0.721574</td>\n",
"     <td>0.344007</td>\n",
"     <td>0.719584</td>\n",
"     <td>0.113885</td>\n",
"    </tr>\n",
"    <tr>\n",
"     <th>std</th>\n",
"     <td>3191.447947</td>\n",
"     <td>0.949534</td>\n",
"     <td>0.766095</td>\n",
"     <td>0.673998</td>\n",
"     <td>0.713598</td>\n",
"     <td>0.671011</td>\n",
"     <td>0.678139</td>\n",
"     <td>0.817518</td>\n",
"     <td>0.911892</td>\n",
"     <td>0.941629</td>\n",
"     <td>...</td>\n",
"     <td>0.789818</td>\n",
"     <td>0.576784</td>\n",
"     <td>0.998168</td>\n",
"     <td>0.926209</td>\n",
"     <td>0.827733</td>\n",
"     <td>0.875289</td>\n",
"     <td>0.692369</td>\n",
"     <td>0.569944</td>\n",
"     <td>0.694437</td>\n",
"     <td>0.993539</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>min</th>\n",
"     <td>1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>...</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>25%</th>\n",
"      <td>2764.500000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>...</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>0.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>0.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>-1.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>50%</th>\n",
"      <td>5528.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>0.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>...</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>-1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>0.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>75%</th>\n",
"     <td>8291.500000</td>\n",
"     <td>1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>-1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>...</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>max</th>\n",
"     <td>11055.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>...</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"     <td>1.000000</td>\n",
"   </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>8 rows × 32 columns</p>\n",
"</div>"
],
"text/plain": [
"             index  having_IPhaving_IP_Address  URLURL_Length  \\\n",
"count  11055.000000               11055.000000   11055.000000   \n",
"mean    5528.000000                   0.313795      -0.633198   \n",
"std     3191.447947                   0.949534       0.766095   \n",
"min        1.000000                  -1.000000      -1.000000   \n",
"25%     2764.500000                  -1.000000      -1.000000   \n",
"50%    5528.000000                 1.000000     -1.000000   \n",
"75%    8291.500000                 1.000000     -1.000000   \n",
"max   11055.000000                 1.000000      1.000000   \n",
"\n",
"      Shortining_Service  having_At_Symbol  double_slash_redirecting  \\\n",
"count       11055.000000     11055.000000            11055.000000   \n",
"mean           0.738761         0.700588                0.741474   \n",
"std            0.673998         0.713598                0.671011   \n",
"min           -1.000000        -1.000000               -1.000000   \n",
"25%            1.000000         1.000000                1.000000   \n",
"50%            1.000000         1.000000                1.000000   \n",
"75%            1.000000         1.000000                1.000000   \n",
"max            1.000000         1.000000                1.000000   \n",
"\n",
"      Prefix_Suffix  having_Sub_Domain  SSLfinal_State  \\\n",
"count   11055.000000      11055.000000    11055.000000   \n",
"mean      -0.734962          0.063953       0.250927   \n",
"std        0.678139          0.817518       0.911892   \n",
"min       -1.000000         -1.000000      -1.000000   \n",
"25%       -1.000000         -1.000000      -1.000000   \n",
"50%       -1.000000          0.000000       1.000000   \n",
"75%       -1.000000          1.000000       1.000000   \n",
"max        1.000000          1.000000       1.000000   \n",
"\n",
"      Domain_registeration_length  ...  popUpWidnow       Iframe  \\\n",
"count                11055.000000  ...  11055.000000  11055.000000   \n",
"mean                    -0.336771  ...      0.613388      0.816915   \n",
"std                      0.941629  ...      0.789818      0.576784   \n",
"min                     -1.000000  ...     -1.000000     -1.000000   \n",
    "25%             -1.000000  ...     1.000000     1.000000   \n",
    "50%             -1.000000  ...     1.000000     1.000000   \n",
    "75%              1.000000  ...     1.000000     1.000000   \n",
    "max              1.000000  ...     1.000000     1.000000   \n",
    "\n",
    "       age_of_domain    DNSRecord   web_traffic    Page_Rank  Google_Index  \\\n",
    "count   11055.000000  11055.000000  11055.000000  11055.000000  11055.000000   \n",
    "mean       0.061239     0.377114     0.287291    -0.483673     0.721574   \n",
    "std        0.998168     0.926209     0.827733     0.875289     0.692369   \n",
    "min       -1.000000    -1.000000    -1.000000    -1.000000    -1.000000   \n",
    "25%       -1.000000    -1.000000     0.000000    -1.000000     1.000000   \n",
    "50%        1.000000     1.000000     1.000000    -1.000000     1.000000   \n",
    "75%        1.000000     1.000000     1.000000     1.000000     1.000000   \n",
    "max        1.000000     1.000000     1.000000     1.000000     1.000000   \n",
    "\n",
    "       Links_pointing_to_page  Statistical_report       Result  \n",
    "count          11055.000000       11055.000000  11055.000000  \n",
    "mean               0.344007           0.719584     0.113885  \n",
    "std                0.569944           0.694437     0.993539  \n",
    "min               -1.000000          -1.000000    -1.000000  \n",
    "25%                0.000000           1.000000    -1.000000  \n",
    "50%                0.000000           1.000000     1.000000  \n",
    "75%                1.000000           1.000000     1.000000  \n",
    "max                1.000000           1.000000     1.000000  \n",
    "\n",
    "[8 rows x 32 columns]"
   ]
  },
  "execution_count": 8,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "data.describe()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 9,
 "id": "4e96f172",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "index                          False\n",
     "having_IPhaving_IP_Address     False\n",
     "URLURL_Length                  False\n",
     "Shortining_Service             False\n",
     "having_At_Symbol               False\n",
     "double_slash_redirecting       False\n",
     "Prefix_Suffix                  False\n",
     "having_Sub_Domain              False\n",
     "SSLfinal_State                 False\n",
     "Domain_registeration_length    False\n",
     "Favicon                        False\n",
     "port                           False\n",
    "HTTPS_token                 False\n",
    "Request_URL                 False\n",
    "URL_of_Anchor               False\n",
    "Links_in_tags               False\n",
    "SFH                         False\n",
    "Submitting_to_email         False\n",
    "Abnormal_URL                False\n",
    "Redirect                    False\n",
    "on_mouseover                False\n",
    "RightClick                  False\n",
    "popUpWidnow                 False\n",
    "Iframe                      False\n",
    "age_of_domain               False\n",
    "DNSRecord                   False\n",
    "web_traffic                 False\n",
    "Page_Rank                   False\n",
    "Google_Index                False\n",
    "Links_pointing_to_page      False\n",
    "Statistical_report          False\n",
    "Result                      False\n",
    "dtype: bool"
   ]
  },
  "execution_count": 9,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
  "data.isnull().any()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 10,
 "id": "c827328f",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "index                        0\n",
     "having_IPhaving_IP_Address     0\n",
     "URLURL_Length                 0\n",
     "Shortining_Service            0\n",
     "having_At_Symbol              0\n",
     "double_slash_redirecting      0\n",
     "Prefix_Suffix                 0\n",
     "having_Sub_Domain             0\n",
     "SSLfinal_State                0\n",
     "Domain_registeration_length    0\n",
     "Favicon                       0\n",
     "port                          0\n",
     "HTTPS_token                   0\n",
     "Request_URL                   0\n",
     "URL_of_Anchor                 0\n",
     "Links_in_tags                 0\n",
     "SFH                           0\n",
    "Submitting_to_email           0\n",
    "Abnormal_URL                  0\n",
    "Redirect                      0\n",
    "on_mouseover                  0\n",
    "RightClick                    0\n",
    "popUpWidnow                   0\n",
    "Iframe                        0\n",
    "age_of_domain                 0\n",
    "DNSRecord                     0\n",
    "web_traffic                   0\n",
    "Page_Rank                     0\n",
    "Google_Index                  0\n",
    "Links_pointing_to_page        0\n",
    "Statistical_report            0\n",
    "Result                        0\n",
    "dtype: int64"
   ]
  },
  "execution_count": 10,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "data.isnull().sum()"
]
},
{
"cell_type": "markdown",
 "id": "85089e0b",
 "metadata": {},
 "source": [
  "## Splitting The Data"
 ]
},
{
 "cell_type": "code",
 "execution_count": 11,
 "id": "efe7fb0d",
 "metadata": {},
 "outputs": [],
 "source": [
  "x=data.iloc[:,1:31].values\n",
  "y=data.iloc[:,-1].values"
 ]
},
{
 "cell_type": "code",
 "execution_count": 12,
 "id": "d06b9e32",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[-1,  1,  1, ...,  1,  1, -1],\n",
     "       [ 1,  1,  1, ...,  1,  1,  1],\n",
     "       [ 1,  0,  1, ...,  1,  0, -1],\n",
    "      ...,\n",
    "      [ 1, -1,  1, ...,  1,  0,  1],\n",
    "      [-1, -1,  1, ...,  1,  1,  1],\n",
    "      [-1, -1,  1, ..., -1,  1, -1]], dtype=int64)"
   ]
  },
  "execution_count": 12,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "x"
]
},
{
"cell_type": "code",
"execution_count": 13,
"id": "b12e3dd5",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "array([-1, -1, -1, ..., -1, -1, -1], dtype=int64)"
   ]
  },
  "execution_count": 13,
  "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "y"
 ]
},
{
 "cell_type": "code",
 "execution_count": 114,
 "id": "1cd850f5",
 "metadata": {},
 "outputs": [],
 "source": [
  "# Creating holders to store the model performance results\n",
  "ML_Model2 = []\n",
  "acc_train = []\n",
  "acc_test = []\n",
  "\n",
  "#function to call for storing the results\n",
  "def storeResults(model, a,b):\n",
  "  ML_Model2.append(model)\n",
  "  acc_train.append(round(a, 3))\n",
  "  acc_test.append(round(b, 3))"
 ]
},
{
 "cell_type": "markdown",
 "id": "aa83769d",
 "metadata": {},
 "source": [
  "## Splitting data into train and test"
 ]
},
{
 "cell_type": "code",
 "execution_count": 14,
 "id": "2c62b3ec",
 "metadata": {},
 "outputs": [],
 "source": [
  "from sklearn.model_selection import train_test_split\n",
  "x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)"
 ]
},
{
 "cell_type": "markdown",
 "id": "a6edcdb1",
 "metadata": {},
 "source": [
  "# MODEL BUILDING"
 ]
},
{
 "cell_type": "code",
 "execution_count": 52,
 "id": "69fc4dfa",
 "metadata": {},
 "outputs": [],
 "source": [
  "from sklearn.metrics import accuracy_score, classification_report"
 ]
},
{
 "cell_type": "markdown",
 "id": "b64c37ed",
 "metadata": {},
 "source": [
  "### Multilayer Perceptrons"
 ]
},
{
 "cell_type": "code",
 "execution_count": 53,
 "id": "f0d189d3",
 "metadata": {},
 "outputs": [],
 "source": [
  "from sklearn.neural_network import MLPClassifier\n",
  "mlp = MLPClassifier(alpha=0.001, hidden_layer_sizes=([100,100,100]))\n",
  "mlp.fit(x_train,y_train)\n",
  "prediction_dt = mlp.predict(x_test)\n",
  "accuracy_dt = accuracy_score(y_test,prediction_dt)*100\n",
  "scores_dict = {}"
 ]
},
{
```

```
"cell_type": "code",
"execution_count": 116,
"id": "03790264",
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "Accuracy score :  97.28629579375848\n",
   "              precision    recall  f1-score   support\n",
   "\n",
   "          -1       0.98      0.96      0.97      1014\n",
   "           1       0.97      0.98      0.98      1197\n",
   "\n",
   "    accuracy                           0.97      2211\n",
   "   macro avg       0.97      0.97      0.97      2211\n",
   "weighted avg       0.97      0.97      0.97      2211\n",
   "\n"
  ]
 }
],
"source": [
 "print('Accuracy score : ',accuracy_dt)\n",
 "scores_dict['Multilayer Perceptrons'] = accuracy_dt\n",
 "y_test_mlp = mlp.predict(x_test)\n",
 "y_train_mlp = mlp.predict(x_train)\n",
 "acc_train_mlp = accuracy_score(y_train,y_train_mlp)*100\n",
 "acc_test_mlp = accuracy_score(y_test,y_test_mlp)*100\n",
   "storeResults('Multilayer Perceptrons', acc_train_mlp, acc_test_mlp)\n",
   "print(classification_report(y_test,prediction_dt))"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 55,
  "id": "ba267d72",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "array([7.02153164e-03, 8.48119933e-03, 2.66413725e-03, 3.12034216e-03,\n",
      "       3.22148690e-03, 1.88894443e-02, 3.07480909e-02, 6.27109392e-01,\n",
      "       1.64672181e-02, 4.10551651e-03, 1.29301933e-03, 3.48331657e-03,\n",
      "       9.05983928e-03, 1.08114562e-01, 3.30627057e-02, 9.39931379e-03,\n",
      "       8.42094003e-03, 2.39105449e-03, 4.57783287e-03, 2.85946906e-03,\n",
      "       1.53911630e-03, 2.06536664e-03, 4.83224351e-04, 1.46171231e-02,\n",
      "       9.18085117e-03, 2.89816074e-02, 5.68897908e-03, 9.38865467e-03,\n",
      "       1.92993684e-02, 4.26529678e-03])"
     ]
    },
    "execution_count": 55,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "dt.feature_importances_"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "9fd7234f",
  "metadata": {},
  "source": [
   "### Logistic Regression"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 83,
  "id": "9b532bbf",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "LogisticRegression()"
     ]
    },
    "execution_count": 83,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
  "from sklearn.linear_model import LogisticRegression\n",
  "lr=LogisticRegression()\n",
  "lr.fit(x_train,y_train)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 121,
 "id": "88045bb6",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "91.67797376752601"
    ]
   },
   "execution_count": 121,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "y_pred1=lr.predict(x_test)\n",
  "from sklearn.metrics import accuracy_score\n",
  "y_test_lr = lr.predict(x_test)\n",
  "y_train_lr = lr.predict(x_train)\n",
  "acc_train_lr = accuracy_score(y_train,y_train_lr)*100\n",
  "acc_test_lr = accuracy_score(y_test,y_test_lr)*100\n",
  "storeResults('Logistic Regression', acc_train_lr, acc_test_lr)\n",
  "log_reg=accuracy_score(y_test,y_pred1)*100\n",
  "log_reg"
 ]
},
{
 "cell_type": "code",
 "execution_count": 58,
 "id": "65ed3f45",
 "metadata": {},
 "outputs": [],
 "source": [
  "scores_dict['LogisticRegression'] = log_reg"
 ]
},
{
 "cell_type": "markdown",
 "id": "4b63ac58",
 "metadata": {},
 "source": [
  "### RANDOM FOREST"
 ]
},
{
 "cell_type": "code",
 "execution_count": 61,
 "id": "e333fd5a",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "RandomForestClassifier(max_depth=5)"
    ]
   },
   "execution_count": 61,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Random Forest model\n",
  "from sklearn.ensemble import RandomForestClassifier\n",
  "\n",
  "# instantiate the model\n",
  "forest = RandomForestClassifier(max_depth=5)\n",
  "\n",
  "# fit the model \n",
  "forest.fit(x_train, y_train)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 62,
 "id": "fd4bff4b",
 "metadata": {},
 "outputs": [],
 "source": [
   "#predicting the target value from the model for the samples\n",
   "y_test_forest = forest.predict(x_test)\n",
   "y_train_forest = forest.predict(x_train)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 118,
  "id": "d568eda2",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Random forest: Accuracy on training Data: 93.340\n",
     "Random forest: Accuracy on test Data: 92.944\n"
    ]
   }
  ],
  "source": [
   "#computing the accuracy of the model performance\n",
   "acc_train_forest = accuracy_score(y_train,y_train_forest)*100\n",
   "acc_test_forest = accuracy_score(y_test,y_test_forest)*100\n",
   "storeResults('Random Forest', acc_train_forest, acc_test_forest)\n",
   "print(\"Random forest: Accuracy on training Data: {:.3f}\".format(acc_train_forest))\n",
   "print(\"Random forest: Accuracy on test Data: {:.3f}\".format(acc_test_forest))"
  ]
 },
{
 "cell_type": "markdown",
 "id": "baa3c131",
 "metadata": {},
 "source": [
  "### Support vector machine model"
 ]
},
{
 "cell_type": "code",
 "execution_count": 66,
 "id": "8268a345",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "SVC(kernel='linear', random_state=12)"
    ]
   },
   "execution_count": 66,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "#Support vector machine model\n",
  "from sklearn.svm import SVC\n",
  "\n",
  "# instantiate the model\n",
  "svm = SVC(kernel='linear', C=1.0, random_state=12)\n",
  "#fit the model\n",
  "svm.fit(x_train, y_train)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 67,
 "id": "e0f12f83",
 "metadata": {},
 "outputs": [],
 "source": [
  "#predicting the target value from the model for the samples\n",
  "y_test_svm = svm.predict(x_test)\n",
  "y_train_svm = svm.predict(x_train)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 119,
 "id": "0251a001",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "SVM: Accuracy on training Data: 93.069\n",
    "SVM : Accuracy on test Data: 91.814\n"
   ]
  }
 ],
 "source": [
  "#computing the accuracy of the model performance\n",
  "acc_train_svm = accuracy_score(y_train,y_train_svm)*100\n",
  "acc_test_svm = accuracy_score(y_test,y_test_svm)*100\n",
  "\n",
  "print(\"SVM: Accuracy on training Data: {:.3f}\".format(acc_train_svm))\n",
  "print(\"SVM : Accuracy on test Data: {:.3f}\".format(acc_test_svm))\n",
  "storeResults('SVM', acc_train_svm, acc_test_svm)"
 ]
},
{
 "cell_type": "markdown",
 "id": "f4746750",
 "metadata": {},
 "source": [
  "### COMPARISON OF MODELS"
 ]
},
{
 "cell_type": "code",
 "execution_count": 122,
 "id": "f012382e",
 "metadata": {},
 "outputs": [
  {
```

```
"data": {
 "text/html": [
  "<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>ML Model</th>\n",
  "      <th>Train Accuracy</th>\n",
  "      <th>Test Accuracy</th>\n",
  "    </tr>\n",
  "  </thead>\n",
  "  <tbody>\n",
  "    <tr>\n",
  "      <th>0</th>\n",
  "      <td>Multilayer Perceptrons</td>\n",
"      <td>98.892</td>\n",
"      <td>97.286</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>Random Forest</td>\n",
"      <td>93.340</td>\n",
"      <td>92.944</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>SVM</td>\n",
"      <td>93.069</td>\n",
"      <td>91.814</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
"      <td>Logistic Regression</td>\n",
"      <td>93.193</td>\n",
"      <td>91.678</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"</div>"
],
"text/plain": [
"                 ML Model  Train Accuracy  Test Accuracy\n",
"0  Multilayer Perceptrons          98.892         97.286\n",
"1           Random Forest          93.340         92.944\n",
    "2                  SVM         93.069        91.814\n",
    "3    Logistic Regression         93.193        91.678"
   ]
  },
  "execution_count": 122,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "results = pd.DataFrame({ 'ML Model': ML_Model2,    \n",
 "    'Train Accuracy': acc_train,\n",
 "    'Test Accuracy': acc_test})\n",
 "results"
]
},
{
"cell_type": "code",
"execution_count": 127,
"id": "ee270e03",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "<BarContainer object of 4 artists>"
   ]
  },
  "execution_count": 127,
    "metadata": {},
    "output_type": "execute_result"
   },
   {
    "data": {
     "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAt4AAAGDCAYAAAAcbBfrAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90b
GliIHZlcnNpb24zLjQuMiwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy8rg+JYAAAACXBIWXMAAAsTAAALEwEA
mpwYAAAlPUlEQVR4nO3dfZTVZb3//9fAyJ2iEA5W6vGomebJoGMqauEBU0TuEtTMkOXp5mSlZhlBQpilhUj
RQZe5TvdanRUmaRIHyzRKAVFKrKSyEgKt4UZJ5E5hPr8//DlfEFAZh2sY5/FYy7UcZu/P/FYy7UcZu/P59oz79nznL2vPVNTV
VUVAABgl2rX0gsAAIC2QHgDAEABwhsAAAoQ3gAAUILwBAKCA2pZewAs9+eTaNDS0j
V8t3qPHXlm16umWXgYUZ/Zpi8w9bVVbmv127WrSvfueO3z/bhfeDQ1VmwnvJG3qtsKWzD5tkbmnrTL7z7
HVBAAACnhZ4f30009n8ODBWbZsWWZJkzpw5GTJkSE499dRMmTKl8XKLFi3KXXHNzI3KiBEjMmDAgIwbNy6bNm6bNm3aNa
sGAIBW5iXDe+HChXnPPe96TxYsXJ0k2bNiQyy67LNdff31mzQQ/3/3ud5k/3e3aSZPTo0fnMZz6TO+64I1VVZdq
0abt08QAA0Fq8ZHHhPmzYtl19+eXr27Jkkeeih3hh3LQQflwAMPTG1tbYYMGZJZs2blsccey4YNG9K7d+8kyfDh
wzNr1qxdungAGGtXvLFlVdddddVWby9fvjx1dXWNb/fs2zTP19euhaZ67Zo2dpzerqurb0Eq
BFmH3aInNPW2X2n7PTv9WkqrZ9VWpNTc0O/31nrVr1dJt55WtdXdesWLGmpZcBxZl92iJzT1vVlma/XbuaF
30Qead/q8l+++2XlStXNr69/PPny9OzZc5t/X7Fi6RP/X7Fb/FAAAaOt2Orx79eqVRx99NEuWLMnmzZsszY8aM9O3bN
/vvvv386SduYBQsWJEluvfXW9O3bt9kXF2wjuOBX2LFj6cmIsuuigbN27MSSedlNOOy1JMnny5IwfPz5zIwlwfPz
5r167NkUcemVGjRjX7ggbNN27VxcUb20G9aEo3osF9XNWtaXb/Y93gAAwM4T3gAAUIDwBgCAAoQ3AAAUsNO/1eTVbM992qdrqWpWpLq+yNm3jlcUA
ALsTW00AAKAA4Q0AAAUlbwAAKEB4AwBAAcIbAAAKEN4AAFCA8AYAgAKENwAAFCC8AQCgAOENAAAFC
G8AAChAeAMAQAHCGwAACqht6QUALWvPPPfdMly7lfwavq+ta7Fzr1jVk7dq1xc4HANsjvKGN9KlXWpqW
noVu1ZVtYvuBqCl2WoCAAFCG8AACjhA2qV2XLnm1v8ChXXUla9e09d0DLYjbSFQAAAAoQ3AAAUIbwBAKAA
4Q0AAAUIbwBAKAA4Q0AAAUIbwAAKEB4AwBAAcIbAAAKEN4AAFCA8AYAgAKENwAAFCC8AQCgAOENAAAFC
G8AAChAeAMAQAHCGwAAChDeAABQgPAGAIAChDcAABQgvAEAoADhDQAABQhvAAAoQHgDAEABwhsAAAoQ3gAAUIDw
BgCAAoQ3AAAUILwBAKAA4Q0AAAUIbwAAKEB4AwBAAcIbAAAKEN4AAFCA8AYAgAJeUXjfdtttGTRoUAYN
GpSrr746SbJo746SbJo0aKMGDEiAwYMyLhx47Jp06ZmWSgAALRmTQ7v7v9evevX56qrrsprspNN92U2267LQ888EDmzJmT
0aNH5zOf+UzuuuuOOOVFWVadOmNed6Qd2Gr/+/evDkNDQ1Zv359nM3PhS59+2x7sZf0Fd2tb07Rv9c8kkyC
ZkZv5OnO+w7snmuuHKvBL55qB3TrzAK5WsM6YbYVPNbYPXMHq1fyVUVIUzUqIk3fvEl6o8J4E4Cq8r4uhhHwExXfgMC0/qSj3+B3Jrg
TDhw/PFFmzumpmut0ADDQatU29Yp77bVXPvaxj2XgwIHp1KlTjjnmmOyxxx6pq6trvExdXV3q6+t3at+enfdu
6m8Joq2Sn6mAwvv/pFiq8Z8s2qP/B8Wd5hzWmPOt9VvfRqGYwLzbUbalSoWWhBRx2Bo5+h0nxjmyEtWBXNd20pa2Uh6mLhoiVXYEBcHaGqJdOkwwG+UY
bbPWIFrvUj2i8RIKsxTm59ZwrPFQ8Yf43t2pWWjhCVNu1QUm6svYaQQZn6fI8/d2tVg3XQUS4rf9Xfd4nI09dMKiVW81TbsOiafq1b85hr3C8SlWPq1Oy5
mnLVxj6x2WXgJgsjv8KC3ujRyGPqDfUMUflzSmQWRAG
4GipgZv5CXIuWSAbX6vr5szmYfYV+yiWH6hWhkq3tKCZDZUo3OWpQTiw+yrBZP0Ggp3VwX2BNXV1YnTHx
ZP5Yrnw0+zUujD/6+2JvzLqzGaVNswIXvW2lNc8q2V90G8S1ctMw6qW4ZevOBAAAm4g8Igu0S6lp8s9PGv
VTs/xM5k1BbwKZgDg0A9VVw73HVjjEBVk9r8dxv3QWrV3QQAvTbUiCSDNkC6zbnsHHFNz4Kp8HxdR6kWjKTM3P
CcHmUvtoyMUlsUHdL79GF9e+nzkMkYi7yzP2jjsrG6z3gMxsXMSw4+cchfd7Qv6rYXDsGnd3mnrdcuoWB2dd9
eLbqBbm+U1iRYbJ4iZa+EXj/N6QyuO+T6dsv0f7OHyBOMzeZQzHZX9DkSzfXA+06w7rnA8wVhP9Iw4ARXFqi4
w8MFrcK3XAVWGAqZpbYQVV37G+Hi7WgYj0vxkP0w/XSsc3HdvsP+s5dWWiEYdlfdFRYQ73ENOswMmUDjqL8
N0oBz/eUKlL8f5u8jFbrQjQtf6pPsJmhVqr3Kxgfsrl0UnVwK60rdyBbAQy/FDbq/7/3LtuC7VCHYXMiq/cRJ0
SgBHs08FdvSOXpMkprOhO1kKZPdLuDD8EXtq6oU4CxWA7pp4+y80wy+iJQfWxtIg5hqVv3q3+PjOYxd6dx7
ee6ad0w9szaddIS06PLiTgO3Jec8p2A5kBE6yhtAtEmGLNGW7b9ojYZTyzqxyQ1Kjmw6vn8OBs83sLHfgAh4
3Ixq2AK9jmvvxCsyDyKtjmrd5Qtij9QGuamHr1MBAC3dN9ZbIfvQKNfnmrFvHRpZB4wAGmzx5kPyNgjSGKlbMl
Z0MApTSKtALzaB3CNwvm4Lp4xtWFZuxPuu0AIGgZ6IL
```

mmWRAADQ2jU5vJcuXZrLL788119/fW6//fY8/PDDmT17di677LJcf/31mTlzZn73u99l9uzZzbleAABolZoc3j/
72c9y+umn57WvfW322GOPTJkyJZ07d85BBx2UAw88MLW1tRkyZEhmzZrVnOsFAIBWqbapV1yyZEn22GO
PvP/978+KFSvSr1+/HHbYYamrq2u8TM+ePVNfX98sCwUAgNasyeG9efPmPPAA7npppvSpUuXfOQjH0nnz
p23uVxNTc1OHbdHj72auiReprq6ri29BCjO3NNWmX3aqt1x9psc3vvuu2+OP/74vOY1r0mSnHzyyZk1a1bat
2/feJnly5enZ8+eO3XcVaueTkND1dRlvSK74ydoV1ixYk1LL4HdiLmnrTL7tEVtZe6Tlpn9du1qXvRB5Cbv8e7Xr
1/ueeeePPXUU9m8eXN+9atf5bTTTsujjz6aJUuWZPPmzZZkxY0b69u3b1FMAAMCrRpMf8e7Vq1c+8IEP5Nxz
z82zzz6bE088Me95z3tyyCGH5KKLLsrGjRtz0kkn5bTTTmvO9QIAQKtUU1VVy+zr2IGW3mpSc8XO7UlvbarLK
087spW6uq7ZyZditDpV5el2tlVX1zVtYfjNPltqE3OftNjs77KtJgAAwMsnvAEAoADhDQAABQhvAAAoQHgDA
EABwhsAAAoQ3gAAUIDwBgCAAoQ3AAAUILwBAKAA4Q0AAAUIbwAAKEB4AwBAAcIbAAAKEN4AAFCA8A
YAgAKENwAAFCC8AQCgAOENAAAFCG8AAChAeAMAQAHCGwAAChDeAABQgPAGAIAChDcAABQgvAEAo
ADhDQAABQhvAAAoQHgDAEABwhsAAAoQ3gAAUIDwBgCAAoQ3AAAUILwBAKAA4Q0AAAUIbwAAKEB4A
wBAAcIbAAAKEN4AAFCA8AYAgAKENwAAFCC8AQCgAOENAAAFCG8AAChAeAMAQAHCGwAAChDeAABQ
gPAGAIAChDcAABQgvAEAoADhDQAABQhvAAAoQHgDAEABwhsAAAoQ3gAAUIDwBgCAAoQ3AAAUILwB
AKAA4Q0AAAUIbwAAKOAVh/fVV1+dsWPHJkkWLVqUESNGZMCAARk3blw2bdr0ihcIAACvBq8ovOfOnZsf/
ehHjW+PHj06l156aWHj06n/nMZ3LHHXekqqpMmzbtFS8QAABeDZoc3qtXr86UKNywQUXJEkee+yxbNiwIb17906SDB
8+PLNmzWqRQIAQGvX5PCeMGFCPv7xj2fvvdOkixfvjx1dXWN76+rq0t0t9f0lXyEAALwK1DblSjffffdd
5fjjj8/06dpan/T52jx57Nb4vfffaBisw4tvQQoztzTVpl92qrdcfabFN4zZ87MihUrMmzYsPsPzzn//
MunXrUlNTk5UrVzZeZsWKzfHenZs+dOH3vVqqftQBtxJewO36CdoUVK9a09BLYjZh72iqzT1vUbLYjZh72iqzT1vUVuY+aZnZb9e
u5kUfRG5SeH/jG99o/Pf06dNz//335z/+4z9yxhln5PHHH88++6zzz5pqAY/zZ3HIzLS8dAu5k1QSh/7TNPZKRwhKy0s0NH+jcLQAAUIDwBgCAAo
Q3AAAUILwBAKAA4Q0AAAUIbwAAKEB4AwBAAcIbAAAKEN4AAFCA8AYAgAKENwAAFCC8AQCggFcU3tdd
d10GDRqUQYMGMGZdKkSUmSOXPmZZMiQITn1Mzz0+vZcqUZlkkAAC0dk0O7zlz5suSee+7Jj370oz9e+7Jj370o9x66635/e9/nxk
zZuSyyy7L9ddfn5kzZ+Z3v/vd/tdZs+e3ZzrBQCAVnqJ4V1XV5XV5exY8emQ4cO2WOPPPPPHnPPLLPooYdm8eLFOeigg5LFOeigg3LggQe
mtrY2Q4YMyaxZs5pzvQAA0Co1ObwPO+yw9O7dO0myePHjw5MzU1Namrq2u8TM+ePVNfX/+KFwkAAK
1d7Ss9wCOPPJIPfehDGTNmTGpra/Poo49u9f6ampdOl6PHnu90iXxEurqurb0EqA4c09bZfZpq3bH2X9F4b
1gwYJcfPHFueyyyzJo0KDMnz8/K1euXqpVT6ehoXoly2qy3fETtCusWLGmpZfAbsTc01aZfdqitjL3ScvMfrt2NS/6IHKTw3vp0qX58Ie
ZfdqitjL3ScvMfrt2NS/6IHKTt5r8/e9/z0c/+t/Mnjw5gwYNs06tUrjz76aJYsWZPNmzdnzdnxowxoZ6du3b1NPAQ
AArxpNfsT7G9/4RjZu3Ju3ZzJiJEyc2/ts555yTRMn5qKLLsrGjRtz0kkn5bTTTmuuWhDAAAAM
Sc8XO7UlvbarLK087spW6uq7ZyZditDpV5el2tlVX1zVtYfjNPltqE3OftNjs77KtJgAAwMsnvAEAoADhDQAAB
QhvAAAoQHgDAEABwhsAAAoQ3gAAUIDwBgCAAoQ3AAAUILwBAKAA4Q0AAAUIbwAAKEB4AwBAAcIbAA
AKEN4AAFCA8AYAgAKENwAAFCC8AQCgAOENAAAFCG8AAChAeAMAQAHCGwAAChDeAABQgPAGAIACh
DcAABQgvAEAoADhDQAABQhvAAAoQHgDAEABwhsAAAoQ3gAAUIDwBgCAAoQ3AAAUILwBAKAA4Q0AA
AUIbwAAKEB4AwBAAcIbAAAKEN4AAFCA8AYAgAKENwAAFCC8AQCgAOENAAAFCG8AAChAeAMAQAHCG
wAAChDeAABQgPAGAIAChDcAABQgvAEAoADhDQAABQhvAAAoQHgDAEABwhsAAAoQ3gAAUIDwBgCAA
oQ3AAAUILwBAKAA4Q0AAAUIbwAAKEB4AwBAAbskvG+//facfvfvrpOeWUU/K9731vV5wCAABaldrmPmB9f
X2mTJmS6dOnp0OHDjnnnHHNy3HHH5Q1veENznwoAAFqNZg/vOXPmE+fPunWrVuSZZSZMCAAZk1a1YuvPD
Cl3X9du1qmntJO+WgfQ5q0fOX0NIfY3Y/B736x97cs31tYPjNPttoA3OftMzsv9Q5mz28ly9fnrq6usa3e/bs
ceeuhlX7979z2be0k7ZfEli1v0/CX06LFXSy+B3czixTxBZ3LNdbWD4zT7baNzn+yes9/ANzn+yes9/ANzn+yes9/ANzn+yes9/ANznBM7X
NAEDb1uzhvd9++++2XlypypWNc9by9fvjw9e/ZsbvObl0ef3Zs7tMAAECr0uzhfffZ6z0bza9EkJ+mf/r0qxNrz+27uvWV5f4q7ir0pz9N3759/

jT7Hu/99tsvH//4xzNq1Kg8++yzOfPMM/OWt7yluU8DAACtSk21vU3ZAABAs/KXKwEAoADhDQAABQhvAA
AoQHgDAEABu314L1u2LIcffngmTJiw1b8vWrQohx9+eKZPn/6i1+/fv3+WLVuWpUuX5rLLLkuS/Pa3v824ce
OSJOedd17uu+++XbP4Fxg7dmz+4z/+I8OGDcuwYcMycODAfPe73y1y7hf6wQ9+kBkzZrTIuSln2bJlefOb39w
4c0OGDEn//v0zderUZjn+9OnTM3bs2GY51pbHPPbYYxvXPGzYsLz//e9v1nNs6aGHHso111yzy47Pq9+sWb
MyfPjwDB06NEOGDMnXv/713Hzzzdud209/+tO58cYbM3369Bx++OHb3A9/+9vfzuGHH55ly5aVWj6twH3
33ZfzzjuvWY41bNiwF33/lud5qctuqX///jn99NMb77f79++fiy++OOvWrWvyWptTfX19PvjBD7b0Mpr/1wnu
Ct26dcuvfvWrbN68Oe3bt0+SzJw5M695zWte9jEef/zxLF26NEly1FFH5aijjtola30pF198cYYPH54kWblyZQY
OHJijjz46b3rTm4qu4ze/+U2OPfbYouekZfTs2TO33XZb49v19fUMGBABg0alEMPPbQFV7Zj/fv3z8SJE4uc6
89//nNWrVpV5Fy8+tTX1+fqq6/O9OnT071796xduzbnnXXdePvKRj+TBBx/MqlWr0qNHjyTJ+vXrc/fdd+dTn/p
U7r777rz2ta/NHXFckcGDBzce72c/+1n23nvvlro5tAFbfj/Ynvnz57/sy77Q//zP/SAAw5IkjzzzDM599xzc+utt+
bcc8/d+YU2s/322y9+rXWnoZrSO899xzzzxxxBG5//7706dPnyTJvffemxNOOKHxMocffnj++/M/JnnuEbP5
8+dv9Y37yiuvzLJlly3LFFVktkNNOy3XXXZebrqp8f2bNm3KYZtJvTJgwIf/4xz9Q/4xz9SU1OTSy+9NCeccEKuvfbaPPjgg/n73/n73+9w9u177775l
//9V+zePHivPGNb8ykSZMyf/78bN68OcOHD8/555+f++67L9dcc00aGhpy2GGH5dOf/nTGjRuXv/71r+nQoU
PGjh2b448/Pr/85S8zderUbNq0KQcccEA+//nPp3v37unfv3/69++fBx54IEnyhS98Ir85dd92VefPmpa6uLj/
5yU+yevXqLFmyJKNHj85rXvOaXHVVdm4cWO6d++ez33Ny3nnn5aijjsqCBQvyxBNPZPz48TnppJ
y++235+tf/3rat2+fAw44INdcc006duzYfANAs1qxYkWqsqe+65w7lfuXJlLrzwwhx22GFZtGhRevTokf/+7/7/9
Ot27dcuutt+arX/1q9tprr+y///7p0qVLkuTBBx/c4aMHz8+N910U/785z/n/PPPPPz/nnn/
+y1/5i59hnn33yyCOP5Ctf+UpWrFix3a+Hq6++Ovfee2/at2+fk08+OaNGjcrUqVMGVrUOzbt26fPWrX82HP/zhXfRR
59XqySefzLPPPPpsNGzYkee771cSJE9OxY8eccsopmTlzMjiHfeeWf69OmT7t27J0mOOeaYLFiwIOvWrUuXL
l3y2GOPZc8990zXrl1b7PbQ+txwww358Y9/nPbt26dr1645a75/JGoB+5rR1JP2+XE7hoT0VdqQ7U7Uu0+N6Z1dz+E7Dc0
K4JmPNq9CWCbU0AA9IyOMyVmnAE3LYgK6qCPQ3gbdpmOSwDyfU2AgpI1qK6yWqDrKjd7hP2C4UaJaTbPdnl3aVz/wpO3/hHPHfbwNAtzakw8EZ7cPgKNzfOL1iL8h1K8BtkmeM6I+6m0CPtJBjtcmOEJQbV13F6Ak/Uajd0aHoqFtyH6ga5pe0iNWPmjthQvHmNbSAGvSQR5sd67i2sl4Y5kmwOt0n+7LcMRHgpmxyeG9WeGsVdHJaNvC0vxK8nEhhIzw9mMwtWJfzs3Ok/3tm8Tx6UxqB1/PONjDDlmqg1+t5kpD0TUqShHHKLUTpjC8iE6JEQr9X+WgHK58A1HkoQeO6LgRWqyrYbtHkVRqKmkj/1q5Jcyqq0HNjPxkWeD8OYm8i4sxSiEPuP2Wys3ZGBWnrxrKqdiOr7Cnyj2+2D4TB1bMAFn/UrgMbHqUm7GN3H+kjXm5FlXIh4+ND4Kxy+KnF0Dr0Zn2K3yIcT9+PxsZJfcuO7V9iU3mD5gaSnz/fWZAk7CUcBy3TkRrnbuGL2LcpmvwtOdRcOcddpg3lvqO1g8xlnDjmKYH9fNuR41RG54c9fi4l2V4p08IncaT4CzwvkKrhn9ue8f8tSdAMjDYwEtzz7SXGfHKfu4RBu2mlzC4nHI8ogQ2QdgDzUNCLHb3d9Z9U9SbGDb8hN+fbAcXA6w8AW0DDYRlYy/D8OCqoaKM/eZu7XNt6Vgd/fJ1HDn8jnh9cKo2aPcK4nSW5Vdj98/eL9VrR1M/3B7hkz7BWkWtHu7hHzyF5oUzuJ1KmvYdP6KdQGYQaBg0bQ+xnt5IhmaWI8RjGiQyRHJxE4VIY2dn5IJ7MRtbXs3+PW8R5zxZmQn8w3f2JMM8G0bphMzEY6vLv+WKSwb5j7wXmCmMb5Rq5cUVcJ4hdnn5eCm5Ejt6Nh1QDMs3lUtZVtkFLmVstKOOOuDzhRMkuB2IqH2gu5+cRfjBwAf8pv/G8pL5/a3h6t7LNgTK7rWplzZqo2w8eaY+7pVhl+nntUuf2yxMBEzE5p2KzwXfv1ZzuZfhc7h/93NeBvtwjhcUGtX+yQdTXfoPnaGc+2IpS3Yg8XVRMWUj9VtVSvU/RZeN3u7UdHcYTvUdXP5f8pcZUoD2HN4T3fuyyzYo+tgvHT8GDnp6eQsqTfxN3cQcbyGyjbfnFfWjK8Uts7u9i/xuyQdx+8+NMWNiiCW1cIJZzWQNnZ1avkbFZrY17b5j1wP3tB8kxYxA/7yaPxBkuVdTs/4gQXeIxmGdXp2kc//e+o0mwxA7yj6aO5ZUXXJZ0fG64eZ7xVFSfIpMeK3mxNYsnaTdmdUoYeIY0AMnRG8y/lgtn0Uzz/7P7H63LyQ0m3FXKLIX7dmR2XHyO8cTEIR5lzmIekzWc6hNEq0rLPsL8ugsxHqfOm0ZVfmr9Mg9qf0QuWLq/8kQd0NE1vMh73byb8gIhr17IlZ4hCzd/y+7FHJl7U+I+HQXj+9dmWQczlVS6zw/ttbhbZa5vZUpqB2v1TwYwJhrfuBLQ1qB7wPvy9tv/0lWqt7J8EwmsChIUbEPkU7qlqW1s8ZzOeqkyyqaPeknbqSZqeY2k2tgXTgnPdkVj4iBJcfyRjbNbAGs+cM2A1YzVRj9PU9RIcMD6s3tl1hIoh4c7dZqIHCw1Hl9s4y7A8nl7iOtp2D7X5+eQbhiZBt7qhqd0fQWX5PGZd0qz2jjq5Kq0q6fXKKNYq1nsf5A8p1NuaMQ8u0lH4ZwpIF8Hy6d9LAV+0Pf3TgfqWa8b/R/pXSrwY7YK/bbmYp/0ZFmfN2m/XTjB0Ukn7oKyjNoD2B6iA5a7jzBXqDrHv5lwXgDDCyFiHjQcpzlfwvDxEPuDmgM3YC88PAAL3PQmEy3c/NeFEbVL2uFXHAlzQkO3C3aEU3AAAVh0ncqGyvAZtJ5DvrXSXrKq0vk9b+Ha5LHeOpk9ZvRKeAV9OMlX/wklvqn3UqG8ye9r5ll/zE/tPX1O8cBeZUYqHhFGx+mdr0x/BSjEeEaxMXOEb/mLYCFbhEyRmx/jHMKjZV07eV2/8ek3RvvNlNm+Xj3xPBW9HIDmGwoxfXjIZuhzl5ecc6wn4Fp8h5NTvBZ8d36hlG3c8c0UzgS1ztvp3yBBhvX+5ijVwRj04kLOOg8S9GQhL/OhtbNvc2B5zlPK3MqpjY4wt6/LtWN1LwX0c/N6x7WHvz4zDMgQgEdnNWVAZSn5jx0zQLs8fJxlbX5L6b3eP9f7xSCmAzpaeUeF+EZ+7kLhz8PoTT+fXcb4kJ18Gx+p7hW4m8/YPQH2LQnqbeYOpJOSPm6ZKmAxzj9bP6EzSdMqLB1w72fvUv0HXVnXNCVYRzg4b1gJy6i4/oQE6WsX18+1SPVh5BQ7LoZ0HW2tkI8JuMn8mj9HQjWmm1l0vEW4lm9mqdD9r3k58V5rcgYkP+nwgzEbHv5jEO+FIThdMLEX+gXmWiQUD+fQO4SqV3c7LfPvwSp7s2sCmBffxRc6ycp5Nz4tGV5Kz1i1jMvRIJgU63m2f6x8q7f5JQ7O0/VA9jLRh7y2mfxXpQo1R26ubO4nx0hWVmPJOG8C/w9kAfJHCu/Tt0cqN3jkv/S0GxwvJ/5Hsh+eb5g/Vu7/7kYH/JlNQBAwECW9qQoKHgBgHOL8mvUdQQbfh+LoU8AGPdiF1fOm8iK1BKkWtpPeMPBwdYgJN4nN2Hm87u7FdOytNcy3tbBPPlizDdB+GCXrzqbGXQ1tqxFXDP0P/FhZ8/ryNGxcY3tWzPxutxBXZHwpkR9JawmF2oUgtqSH1uPq9zdDb0oCEefE6NSo0BOfMiNPvWwadFajP56uHAcV9Xx5G+t7UoY1Sd2FM9v+M0rXvrMnOf5jGQeuVq7uFzqStg2qRw6QHpYpJmbWyNVMr/PWF4Yhz7lEzmU2Z8Yb+DXY/vOVmBOKycNqNNPnOOb3Ptefn36nxfKNybS9Egmb1hDjKjqqXwGl8XjubcvH4YTPRJPmMnUz2zsnsgoY8MB61RtU+s4qJyPSWKjWEBfa4nkf5cPEhTRvF9rbd8f5DNhKzFhEKO+w7hVlm9tW/6K/9zJTH4zFp5rOd9bYxkMhB9xrT89Fl3BPH+tvKP6Fdx1LdBr8gcC16xMgXHcc6O3H/9VGbhKCr2G0h5+SrXNmPfz4FuJvejKJ/32KSdUdxPTq+HEN5RWXMh2GL7/jGi0/rP+Y3jmOCKlGMtcFzEm2JSOLvR9B8WzDo9SXGLr0bXbA6nt0Gi0FeZsWjLF8OJu5ngORRuLtfeQ3XB+9TIh8qomVQDy8/+gQQD9Qw==

6dOSZIRI0bk1ltvzTPPPJN+/fplr732SpIMGjRom/vb5x8dfuc735mTTz55m9nc0vbaY3ue32pyxx135Itf/GL69+
+fmpqaHd4v/+lPf0qPHj1yxBFHJEnOPPPMXHXVVY3Ha+r9+vZu25avm5g3b16uvPLKJMmBBx6YXr16ZeHCh
UmS4447Lh06dEiPHj3SrVu3rFmzplm3f7Wa8N5rr71yxBFHZMGCBZk3b14uvfTSbcK7qqrU1NRk06ZNO338
n//855k6d6dWpGjRqV4cOH58knn0z1//9RzxEjRmTGjBl5/PHHc9FFFyV57pP+ne98p7tPr6+uy777658847G7
8AtmfLPd5b2rx5c0aPHp1TTz01SfLEE0+kS5cuWbhw4VbHq63d+lP2l7/8JZs3b86///u/u/54YbbkiSbNy4MWvX
rt3udRoaGhr3yW/p+XM0NDRs876qqrJ58+YkaQyVmpqaxvvePHHz8+f/jDHzJ79uyMHj06F1544U69IIMy2rVrl
0996lN517velW9+85v50Ic+9KJzv2WU1tTUNH59bkTkjz8/WS83NHnvssc11dtZLneP5Gd7R10NtbW1uvvnmz
J8/P7/85S9zzjnnnbLXXdDJriF7/4RdatW5fTTz89I0aMyIgRIzJt2rT88Ic/zKmnnnnpp3vetdmTFjRjp16pShQ4emXbu
tf6fBcccdl/Hjx+dPf/pTunfvbpsJO2V794ubNm1Ku3bttvvu+LZ1//vnp169f7f7r777lxzzzTV56KGHdritYnvtcfDBB2
8zz88bMGBA7r333kyYMCHf+MY3dni/vHz58hddZ1Pv17d324YMGdJ43Gd7R9e52T/L/vfc1pt/+tJlsaOHB
gvvSlL+XNb37zNkPQvXv3PPLII6mqKnfddd2123fvv2LBvncuXMzcODAjBgxIvvvuu2/uv//+xk/Caaedl2blyY
pXp1atXkqRPnz75/ve/n+S5F2cNHTo069evb/Jt69OnT6ZNm5Znn302a9euXzbnnnntv409eW3va2tzX+wPGXv
/wlH/zgB/OWt7wlDz74YB599NEkyfXXX59JkyY1XucnP/P/lJkudetHPooYdmn332Sfv27Rtv35YOOoeSQrF69Og8
99FCS517E+vrXv77xB4wX2rRpU0499dR07949H/rQhzJs2LAsWrSoyR8Hdq3a2tp86lXxv
z9FHH52FCxemvr4+DQ0NjbO4s3PTFC/3HL169dru18PDDz+ckSNH5phjjsmYMWNy6KGH5tFHH33J+wZ4M
Z06dcqXvvSlxkfTqqrKn//858Zn/s4444z4z87Gc/a/zNJy/Uvn37vP3tb8+ECRNy+umnF107rV+fPn3ya
MimTZtyyy23pE+fPjn++OMze/bsPP3PPP3003nmmWfy05/+dKsHzJLn9nGvXbu8bU28bU2Dz/8cJLt99L22uOFx3uhj3
3sY/nNb36Tu+++e4f3f3y4+JFm6dGl+/etfp3fv3i/14iefh+ueluqpxtfo3X777ds91s7er+/otm/+rvLZ8ejvnvXd9+uvGS
R7+S5LQ3jxo3Lxz72sW3ed+auaCCy7ltvvvum6OPPPjpPvnkVu8ue/tvLk1nSe+xW+VwvumOPPjTPFPvnkVu8/9NBDs2bNmowePTpnnnnnmMNtc/66yz8sl
PfjKzZs1Khw4d0rt378Y70k6d0OqV379554xvf2Hj7qi5aRFVqc+1FYH58KECY0/QU2aNCmOOecc7JKnCccc7JkyZKcY2bRp
U4YPH57jjjtum191ePHF2f8+PEZOnRamtrM2nSpPTs2TrRQQZHaeW0SpPTs2TNf+MIXIXcsK6ShoSH77ShoSH77bffVr8e7de//nV+
Pnzo1P359wwgn58pe/vM2jKx06dMiUKx06dMiUKVPy+c9/PuvXr85vPFHF2f8+PEZOnRoamtr
+/rvyuMOvfu9+4+tf/9aPW6fsrPRzZ8/VNf5UTHsF4cOHTj3s0+yTKVOm7HDnH5Xd3h/vvM+/7d+H24v9uvPCZoX0ZjCrPp9HrnRcXq7f/P/F4XSjSu2OVODHj2IJ5Xcc7ppH3f9/ap+vfW9j/9pvfYWUYsWZ
cSIEdm0aVAmsm8CB0WDm0aj+/5IMtN6NWvG2kX5ZXuu9+RimLsLGMJR7cBbm1W2ODUySULxRT11/6l51d7nn32Kr3Jv/eqHAtJgP+cGB
+GYJ7Y+l0D2C8M8/cXEXMB8cTp8hXnHKu8hvHRP/e++9XvXbbnhX6/9xb3Xvv1nXAaPjT0l7iR3ANX/O2Y9yhQYO8f1RX+xwF1b/iJTZOYR7I8Lu/eD8HGs
u21a9r3l57vFR49+jRIx/84AczadKk3H777du9X+X7QoUMmTZqUMWPGpF27dj44IO3u2tgZ+/XO3fuvN3b9r
xx48ZlwoQJjbfxiuvTM+ePXfuE9JENVVzP4b+KlNVVdauXZt3v/vd+fa3v526urqWXtJO6d+/f2688cbGX+8DA
Ly6Pfroo5k9e3bjb4368Ic/nLPOmu3++GuoaEhkydPzoUXXpguXbrkW9/6Vurr65v970PsTpVrTlrV94Ps4t4be//W0
+8IEP5KMf/Wiri24AoO3Zf//989989vf/jaDBw9OTU1N3v72t+elvtvAD6t6GvXPLt26dcuXXcfuZZ56ZPe49o
tK1bi8
+8IEP5KMf/Wiri24AoO3Zf//989vf/jaDBw9OTU1N3v72t6dfv34tvaxttGvXLt26dcuZZ56ZPfbYI/vvv/9WL658
NfKINwAAFNCqXlwJAACtlfAGAIAChDcAABQgvAEAoADhDQAABQhvAAAo4P8DxUd6DpDsdqUAAAAASUV
ORK5CYII=\n",
"text/plain": [
"<Figure size 892.8x468 with 1 Axes>"
]
},
"metadata": {},
"output_type": "display_data"
}
],

```
  "source": [
   "import matplotlib.pyplot as plt\n",
   "plt.bar(ML_Model2,acc_test,width=0.3,color=['green','blue','red','red'])"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "76391298",
  "metadata": {},
  "source": [
   "## CONCLUSION\n",
   "\n",
   "### Based on different Algorithms we choose Multilayer Perceptrons (MLP's) because it gives better
accuracy compared to other Models"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "id": "8892ae75",
  "metadata": {},
  "outputs": [],
  "source": []
 }
],
"metadata": {
 "kernelspec": {
  "display_name": "Python 3",
  "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.8.6"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```