ASSIGNMENT-2

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": [],
      "collapsed_sections": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "execution_count": null,
   "metadata": {
    "id": "hmz9TPc9mo0E"
   },
   "outputs": [],
   "source": [
    "import numpy as np\n",
    "import pandas as pd"
   ]
  },
  {
   "cell_type": "markdown",
   "source": [
    "1.Descriptive statistics on the data"
   ],
   "metadata": {
    "id": "6CukUMziDoM-"
   }
  },
  {
   "cell_type": "code",
   "source": [
  "df=pd.read_csv(\"/content/Churn_Modelling.csv\")\n",
  "df.head()"
 ],
 "metadata": {
  "colab": {
   "base_uri": "https://localhost:8080/",
   "height": 270
  },
  "id": "Irxh2NEar-4a",
  "outputId": "c4bca3f5-92fb-4052-e087-4d11abdc3c26"
 },
 "execution_count": null,
 "outputs": [
  {
   "output_type": "execute_result",
   "data": {
    "text/plain": [
     "   RowNumber  CustomerId   Surname  CreditScore Geography  Gender  Age  \\\n",
     "0          1    15634602  Hargrave          619    France  Female   42   \n",
     "1          2    15647311      Hill          608     Spain  Female   41   \n",
     "2          3    15619304      Onio          502    France  Female   42   \n",
  "3         4    15701354     Boni         699    France  Female   39   \n",
  "4         5    15737888  Mitchell        850     Spain  Female   43   \n",
  "\n",
  "   Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \\\n",
  "0       2       0.00              1          1               1   \n",
  "1       1   83807.86              1          0               1   \n",
  "2       8  159660.80              3          1               0   \n",
  "3       1       0.00              2          0               0   \n",
  "4       2  125510.82              1          1               1   \n",
  "\n",
  "   EstimatedSalary  Exited  \n",
  "0        101348.88       1  \n",
  "1        112542.58       0  \n",
  "2        113931.57       1  \n",
  "3         93826.63       0  \n",
  "4         79084.10       0  "
],
"text/html": [
  "\n",
  "  <div id=\"df-c703a7e7-9186-4a7d-aee6-043b3319f60c\">\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>RowNumber</th>\n",
"      <th>CustomerId</th>\n",
"      <th>Surname</th>\n",
"      <th>CreditScore</th>\n",
"      <th>Geography</th>\n",
"      <th>Gender</th>\n",
"      <th>Age</th>\n",
"      <th>Tenure</th>\n",
"      <th>Balance</th>\n",
"      <th>NumOfProducts</th>\n",
"      <th>HasCrCard</th>\n",
"      <th>IsActiveMember</th>\n",
"      <th>EstimatedSalary</th>\n",
"      <th>Exited</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>1</td>\n",
"      <td>15634602</td>\n",
"      <td>Hargrave</td>\n",
"      <td>619</td>\n",
"      <td>France</td>\n",
"      <td>Female</td>\n",
"      <td>42</td>\n",
"      <td>2</td>\n",
"      <td>0.00</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>101348.88</td>\n",
"      <td>1</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>2</td>\n",
"      <td>15647311</td>\n",
"      <td>Hill</td>\n",
"      <td>608</td>\n",
"      <td>Spain</td>\n",
"      <td>Female</td>\n",
"      <td>41</td>\n",
"      <td>1</td>\n",
"      <td>83807.86</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>1</td>\n",
"      <td>112542.58</td>\n",
"      <td>0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>3</td>\n",
"      <td>15619304</td>\n",
"      <td>Onio</td>\n",
"      <td>502</td>\n",
"      <td>France</td>\n",
"      <td>Female</td>\n",
"      <td>42</td>\n",
"      <td>8</td>\n",
"      <td>159660.80</td>\n",
"      <td>3</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>113931.57</td>\n",
"    <td>1</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>3</th>\n",
"    <td>4</td>\n",
"    <td>15701354</td>\n",
"    <td>Boni</td>\n",
"    <td>699</td>\n",
"    <td>France</td>\n",
"    <td>Female</td>\n",
"    <td>39</td>\n",
"    <td>1</td>\n",
"    <td>0.00</td>\n",
"    <td>2</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>93826.63</td>\n",
"    <td>0</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>4</th>\n",
"      <td>5</td>\n",
"      <td>15737888</td>\n",
"      <td>Mitchell</td>\n",
"      <td>850</td>\n",
"      <td>Spain</td>\n",
"      <td>Female</td>\n",
"      <td>43</td>\n",
"      <td>2</td>\n",
"      <td>125510.82</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>79084.10</td>\n",
"      <td>0</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"</div>\n",
"        <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-c703a7e7-9186-4a7d-aee6-043b3319f60c')\"\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-c703a7e7-9186-4a7d-aee6-043b3319f60c 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-c703a7e7-9186-4a7d-aee6-043b3319f60c');\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": 56
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "print(f\"Dataset Dimension: **{df.shape[0]}** rows,  **{df.shape[1]}** columns\")"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "3MgsfRwwDTg7",
    "outputId": "7686e40b-0f10-477c-c130-cc92b3e4b064"
  },
  "execution_count": null,
  "outputs": [
    {
      "output_type": "stream",
      "name": "stdout",
          "text": [
            "Dataset Dimension: **10000** rows,  **14** columns\n"
          ]
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "df.info()\n",
        "\n",
        "print(\"<br>**SeniorCitizen** is already in integer form<br><br>**TotalCharges** should be converted to float\")"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "gXOx7TuuD1Gj",
        "outputId": "bd0188e1-d5a2-4e23-b4dd-8afa01f01e09"
      },
"execution_count": null,
"outputs": [
 {
   "output_type": "stream",
   "name": "stdout",
   "text": [
     "<class 'pandas.core.frame.DataFrame'>\n",
     "RangeIndex: 10000 entries, 0 to 9999\n",
     "Data columns (total 14 columns):\n",
     " #   Column           Non-Null Count  Dtype  \n",
     "---  ------           --------------  -----  \n",
     " 0   RowNumber        10000 non-null  int64  \n",
     " 1   CustomerId       10000 non-null  int64  \n",
     " 2   Surname          10000 non-null  object \n",
     " 3   CreditScore      10000 non-null  int64  \n",
     " 4   Geography        10000 non-null  object \n",
     " 5   Gender           10000 non-null  object \n",
     " 6   Age              10000 non-null  int64  \n",
     " 7   Tenure           10000 non-null  int64  \n",
     " 8   Balance          10000 non-null  float64\n",
     " 9   NumOfProducts    10000 non-null  int64  \n",
      " 10  HasCrCard        10000 non-null  int64  \n",
      " 11  IsActiveMember   10000 non-null  int64  \n",
      " 12  EstimatedSalary  10000 non-null  float64\n",
      " 13  Exited           10000 non-null  int64  \n",
      "dtypes: float64(2), int64(9), object(3)\n",
      "memory usage: 1.1+ MB\n",
      "<br>**SeniorCitizen** is already in integer form<br><br>**TotalCharges** should be converted to float\n"
     ]
    }
   ]
  },
  {
   "cell_type": "code",
   "source": [
    "print('Known observations: {}\\nUnique observations: {}'.format(len(df.index),len(df.drop_duplicates().index)))\n",
    "\n",
    "print(\"**No duplicates Found!**\")"
   ],
   "metadata": {
    "colab": {
```

```
   "base_uri": "https://localhost:8080/"
  },
  "id": "522itHTPD-Yd",
  "outputId": "11781897-1515-4db2-da3d-086a64de5331"
 },
 "execution_count": null,
 "outputs": [
  {
   "output_type": "stream",
   "name": "stdout",
   "text": [
    "Known observations: 10000\n",
    "Unique observations: 10000\n",
    "**No duplicates Found!**\n"
   ]
  }
 ]
},
{
 "cell_type": "code",
 "source": [
  "df.describe(include=['object']).T"
],
"metadata": {
  "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 143
  },
  "id": "cKw-rN34EIni",
  "outputId": "a2a9e8b3-e69f-4fcc-ee21-8e38c1ce66f5"
},
"execution_count": null,
"outputs": [
  {
    "output_type": "execute_result",
    "data": {
      "text/plain": [
        "          count unique    top  freq\n",
        "Surname    10000   2932   Smith    32\n",
        "Geography  10000      3  France  5014\n",
        "Gender     10000      2    Male  5457"
      ],
"text/html": [
 "\n",
 "  <div id=\"df-6acd073d-74ad-4ed5-973d-2b91b4a326d1\">\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>count</th>\n",
"      <th>unique</th>\n",
"      <th>top</th>\n",
"      <th>freq</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>Surname</th>\n",
"      <td>10000</td>\n",
"      <td>2932</td>\n",
"      <td>Smith</td>\n",
"      <td>32</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>Geography</th>\n",
"      <td>10000</td>\n",
"      <td>3</td>\n",
"      <td>France</td>\n",
"      <td>5014</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>Gender</th>\n",
"      <td>10000</td>\n",
"      <td>2</td>\n",
"      <td>Male</td>\n",
"      <td>5457</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"</div>\n",
"        <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-6acd073d-74ad-4ed5-973d-2b91b4a326d1')\"\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-6acd073d-74ad-4ed5-973d-2b91b4a326d1 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-6acd073d-74ad-4ed5-973d-2b91b4a326d1');\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": 65
  }
 ]
},
{
 "cell_type": "markdown",
 "source": [
    "2.Missing values"
  ],
  "metadata": {
    "id": "konwvNy7Es0k"
  }
},
{
  "cell_type": "code",
  "source": [
    "df.isna().sum()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "6FKW3ztPEdx4",
    "outputId": "64d85f9f-b0ac-474c-cd88-5decda435d7c"
  },
  "execution_count": null,
  "outputs": [
    {
```

```
"output_type": "execute_result",
"data": {
  "text/plain": [
    "RowNumber          0\n",
    "CustomerId         0\n",
    "Surname            0\n",
    "CreditScore        0\n",
    "Geography          0\n",
    "Gender             0\n",
    "Age                0\n",
    "Tenure             0\n",
    "Balance            0\n",
    "NumOfProducts      0\n",
    "HasCrCard          0\n",
    "IsActiveMember     0\n",
    "EstimatedSalary    0\n",
    "Exited             0\n",
    "dtype: int64"
  ]
},
"metadata": {},
   "execution_count": 67
  }
 ]
},
{
 "cell_type": "markdown",
 "source": [
  "3.Univariate analysis"
 ],
 "metadata": {
  "id": "2udqZB-s-RGN"
 }
},
{
 "cell_type": "markdown",
 "source": [
  "monthly charges"
 ],
 "metadata": {
  "id": "ES6bKl3R-eCm"
 }
```

```
},
{
  "cell_type": "code",
  "source": [
    "stat, p = stats.normaltest(df_churn['MonthlyCharges'])\n",
    "\n",
    "print('Statistics=%.5f, p=%.3f' % (stat, p))\n",
    "\n",
    "# interpret\n",
    "alpha = 0.05\n",
    "if p > alpha:\n",
    "    print('Sample looks Gaussian (fail to reject H0)')\n",
    "else:\n",
    "    print('Sample does not look Gaussian (reject H0)')"
  ],
  "metadata": {
    "id": "-Mw8BZTs-EaX"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "markdown",
  "source": [
    "Total charges"
  ],
  "metadata": {
    "id": "webE-syc-grV"
  }
},
{
  "cell_type": "code",
  "source": [
    "result = stats.anderson(df_churn['TotalCharges'])\n",
    "\n",
    "print('Statistic: %.3f' % result.statistic)\n",
    "\n",
    "p = 0\n",
    "\n",
    "for i in range(len(result.critical_values)):\n",
    "    sl, cv = result.significance_level[i], result.critical_values[i]\n",
    "    if result.statistic < result.critical_values[i]:\n",
    "        print(f'Significance level {sl:.2f} % : critical value {cv:.3f}, data looks normal (fail to reject H0)')\n",
    "    else:\n",
    "            print(f'Significance level {sl:.2f} % : critical value {cv:.3f}, data does not look normal (reject H0)')"
   ],
   "metadata": {
    "id": "AHyHcWxs-VnS"
   },
   "execution_count": null,
   "outputs": []
  },
  {
   "cell_type": "markdown",
   "source": [
    "4.Bivariate analysis"
   ],
   "metadata": {
    "id": "4QGL7SKw79b6"
   }
  },
  {
   "cell_type": "code",
   "source": [
    "def cal_spearmanr(c1, c2):\n",
    "\n",
    "    alpha = 0.05\n",
    "\n",
    "    correlation, p_value = stats.spearmanr(df_churn[c1], df_churn[c2])\n",
    "\n",
    "    print(f'{c1}, {c2} correlation : {correlation}, p : {p_value}')\n",
    "\n",
    "    if p_value > alpha:\n",
    "        print('Probably do not have monotonic relationship (fail to reject H0)')\n",
    "    else:\n",
    "        print('Probably have monotonic relationship (reject H0)')"
   ],
   "metadata": {
    "id": "F2rRacxN7zuU"
   },
   "execution_count": null,
   "outputs": []
  },
{
 "cell_type": "code",
 "source": [
  "def kendall_rank_correlation(feature1, feature2):\n",
  "\n",
  "    coef, p_value = stats.kendalltau(df_churn[feature1], df_churn[feature2])\n",
  "    print(f\"Correlation between {feature1} and {feature2} \")\n",
  "    print('Kendall correlation coefficient = %.5f, p = %.5f' % (coef, p_value))\n",
  "\n",
  "    # interpret the significance\n",
  "    alpha = 0.05\n",
  "    if p_value > alpha:\n",
  "        print('Samples are uncorrelated (fail to reject H0) p=%.3f' % p_value)\n",
  "    else:\n",
  "        print('Samples are correlated (reject H0) p=%.3f' % p_value)\n",
  "    print('----\\n')"
 ],
 "metadata": {
  "id": "SHpKiu1W-uBL"
 },
 "execution_count": null,
  "outputs": []
 },
 {
  "cell_type": "code",
  "source": [
   "ordinal_features = ['tenure-binned','MonthlyCharges-binned', 'TotalCharges-binned']\n",
   "\n",
   "for ord in ordinal_features:\n",
   "    printmd(f\"Correlation with **{ord}**\")\n",
   "    kendall_rank_correlation('tenure',ord)\n",
   "    kendall_rank_correlation('MonthlyCharges',ord)\n",
   "    kendall_rank_correlation('TotalCharges',ord)"
  ],
  "metadata": {
   "id": "iQrL0Mxc--bN"
  },
  "execution_count": null,
  "outputs": []
 },
 {
  "cell_type": "code",
   "source": [
    "def mannwhitneyu_correlation(feature1):\n",
    "    stat, p_value = stats.mannwhitneyu(df_churn[feature1], (df_churn['Churn'] == 'Yes').astype(int))\n",
    "    print(f\"Correlation between {feature1} and Churn\")\n",
    "    print('Statistics = %.5f, p = %.5f' % (stat, p_value))\n",
    "\n",
    "    # interpret the significance\n",
    "    alpha = 0.05\n",
    "    if p_value > alpha:\n",
    "        print('Same distribution (fail to reject H0)')\n",
    "    else:\n",
    "        print('Different distribution (reject H0)')\n",
    "    print('----\\n') "
   ],
   "metadata": {
    "id": "Utn5qFRE_FNe"
   },
   "execution_count": null,
   "outputs": []
  },
{
  "cell_type": "code",
  "source": [
   "def correlation_ratio(categories, measurements):\n",
   "    fcat, _ = pd.factorize(categories)\n",
   "    cat_num = np.max(fcat)+1\n",
   "    y_avg_array = np.zeros(cat_num)\n",
   "    n_array = np.zeros(cat_num)\n",
   "    for i in range(0,cat_num):\n",
   "        cat_measures = measurements[np.argwhere(fcat == i).flatten()]\n",
   "        n_array[i] = len(cat_measures)\n",
   "        y_avg_array[i] = np.average(cat_measures)\n",
   "    y_total_avg = np.sum(np.multiply(y_avg_array,n_array))/np.sum(n_array)\n",
   "    numerator = np.sum(np.multiply(n_array,np.power(np.subtract(y_avg_array,y_total_avg),2)))\n",
   "    denominator = np.sum(np.power(np.subtract(measurements,y_total_avg),2))\n",
   "    if numerator == 0:\n",
   "        eta = 0.0\n",
   "    else:\n",
   "        eta = np.sqrt(numerator/denominator)\n",
   "    return eta"
  ],
  "metadata": {
   "id": "_FJHiPMH_eid"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
   "def cramers_v(x, y):\n",
   "    \"\"\" calculate Cramers V statistic for categorial-categorial association.\n",
   "      uses correction from Bergsma and Wicher,\n",
   "      Journal of the Korean Statistical Society 42 (2013): 323-328\n",
   "    \"\"\"\n",
   "    confusion_matrix = pd.crosstab(x,y)\n",
   "    chi2 = stats.chi2_contingency(confusion_matrix)[0]\n",
   "    n = confusion_matrix.sum().sum()\n",
   "    phi2 = chi2/n\n",
   "    r,k = confusion_matrix.shape\n",
   "    phi2corr = max(0, phi2-((k-1)*(r-1))/(n-1))\n",
   "    rcorr = r-((r-1)**2)/(n-1)\n",
   "    kcorr = k-((k-1)**2)/(n-1)\n",
   "    return np.sqrt(phi2corr/min((kcorr-1),(rcorr-1)))"
  ],
  "metadata": {
   "id": "I1cVrTKWBH_B"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
   "printmd(\"**Correlation Between Polytomous Features with Target : Churn**\")\n",
   "cramer_v_val_dict = {}\n",
   "for col in polytomous_cols:\n",
   "    cramer_v_val_dict[col] = cramers_v(df_churn[col], df_churn['Churn'])\n",
   "\n",
   "cramer_v_val_dict_sorted = sorted(cramer_v_val_dict.items(), key=lambda x:x[1], reverse=True)\n",
   "\n",
     "for k,v in cramer_v_val_dict_sorted:\n",
     "    print(k.ljust(left_padding), v)\n",
     "\n",
     "printmd(\"<br>**Contract, OnlineSecurity, TechSupport, InternetService are moderately correlated with Churn**<br>\")  "
   ],
   "metadata": {
     "id": "CAFC5Ts2BbD9"
   },
   "execution_count": null,
   "outputs": []
},
{
   "cell_type": "code",
   "source": [
     "printmd(\"**Cramers V Heatmap on Polytomous Features and Target: Churn**\")\n",
     "cramers_v_val = pd.DataFrame(index=['Churn'], columns=polytomous_cols)\n",
     "\n",
     "for j in range(0,len(polytomous_cols)):\n",
     "    u = cramers_v(df_churn['Churn'], df_churn[polytomous_cols[j]])\n",
     "    cramers_v_val.loc[:,polytomous_cols[j]] = u\n",
   "\n",
   "cramers_v_val.fillna(value=np.nan,inplace=True)\n",
   "plt.figure(figsize=(20,1))\n",
   "sns.heatmap(cramers_v_val,annot=True,fmt='.3f', cmap=\"YlGnBu\")\n",
   "plt.show()"
  ],
  "metadata": {
   "id": "mV_q1z9gBjkx"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
   "theilu = pd.DataFrame(index=['Churn'], columns=cat_cols)\n",
   "\n",
   "for j in range(0,len(cat_cols)):\n",
   "    u = theil_u(df_churn['Churn'].tolist(),df_churn[cat_cols[j]].tolist())\n",
   "    theilu.loc[:,cat_cols[j]] = u\n",
   "\n",
   "theilu.fillna(value=np.nan,inplace=True)\n",
   "plt.figure(figsize=(20,1))\n",
   "sns.heatmap(theilu,annot=True,fmt='.2f')\n",
   "plt.show()\n",
   "\n",
   "printmd(\"**Contract, OnlineSecurity, TechSupport, tenure-binned are moderately correlated with Churn**\")  "
  ],
  "metadata": {
   "id": "zkCdtArwBtdF"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "markdown",
  "source": [
   "5.Multivariate analysis"
  ],
  "metadata": {
   "id": "2HFfajoyCNJT"
```

```
     }
    },
    {
     "cell_type": "code",
     "source": [
      "# compare samples\n",
      "stat, p = stats.kruskal(df_churn['TotalCharges'], df_churn['tenure'], df_churn['MonthlyCharges'])\n",
      "print('Statistics=%.3f, p=%.3f' % (stat, p))\n",
      "# interpret\n",
      "alpha = 0.05\n",
      "if p > alpha:\n",
      "    print('Same distributions (fail to reject H0)')\n",
      "else:\n",
      "    print('Different distributions (reject H0)')"
     ],
     "metadata": {
      "id": "KAESM5dZCaaL"
     },
     "execution_count": null,
     "outputs": []
```

```
  },
  {
   "cell_type": "code",
   "source": [
    "# compare samples\n",
    "stat, p = stats.kruskal(df_churn['DeviceProtection'], df_churn['StreamingMovies'], df_churn['PhoneService'])\n",
    "print('Statistics=%.3f, p=%.3f' % (stat, p))\n",
    "# interpret\n",
    "alpha = 0.05\n",
    "if p > alpha:\n",
    "    print('Same distributions (fail to reject H0)')\n",
    "else:\n",
    "    print('Different distributions (reject H0)')"
   ],
   "metadata": {
    "id": "nDBLKHPBCbsF"
   },
   "execution_count": null,
   "outputs": []
  },
  {
   "cell_type": "code",
   "source": [
    "# compare samples\n",
    "stat, p = stats.kruskal(df_churn['Contract'], df_churn['PaymentMethod'], df_churn['PhoneService'], df_churn['InternetService'])\n",
    "print('Statistics=%.3f, p=%.3f' % (stat, p))\n",
    "# interpret\n",
    "alpha = 0.05\n",
    "if p > alpha:\n",
    "    print('Same distributions (fail to reject H0)')\n",
    "else:\n",
    "    print('Different distributions (reject H0)')"
   ],
   "metadata": {
    "id": "kuS1HYW9Cf7a"
   },
   "execution_count": null,
   "outputs": []
  },
  {
   "cell_type": "markdown",
   "source": [
    "6.outliers and replace the outliers "
   ],
   "metadata": {
    "id": "3xjghkG8HPov"
   }
  },
  {
   "cell_type": "code",
   "source": [
    "do_col <- function(c){\n",
    "b <- boxplot(c, plot = FALSE)\n",
    "s1 <- c\n",
    "s1[which(c %in% b$out)] <- mean(c[which(! c %in% b$out)],na.rm=TRUE)\n",
    "return(s1)\n",
    "}\n",
    "\n",
    "# (testvec <- c(rep(1,9),100))\n",
    "# do_col(testvec)\n",
    "library(tidyverse)\n",
    "columns_to_do <- names(select_if(iris,is.numeric))\n",
    "\n",
    "purrr::map_dfc(columns_to_do,\n",
    "        ~do_col(iris[[.]])) %>% set_names(columns_to_do)"
   ],
   "metadata": {
    "id": "1t4iZfa3G0j4"
   },
   "execution_count": null,
   "outputs": []
  },
  {
   "cell_type": "markdown",
   "source": [
    "7.Check for Categorical columns and perform encoding."
   ],
   "metadata": {
    "id": "3mFYbXo5HeoF"
   }
  },
  {
   "cell_type": "code",
   "source": [
    "# Define the headers since the data does not have any\n",
    "headers = [\"symboling\", \"normalized_losses\", \"make\", \"fuel_type\", \"aspiration\",\n",
    "           \"num_doors\", \"body_style\", \"drive_wheels\", \"engine_location\",\n",
    "           \"wheel_base\", \"length\", \"width\", \"height\", \"curb_weight\",\n",
    "           \"engine_type\", \"num_cylinders\", \"engine_size\", \"fuel_system\",\n",
    "           \"bore\", \"stroke\", \"compression_ratio\", \"horsepower\", \"peak_rpm\",\n",
    "           \"city_mpg\", \"highway_mpg\", \"price\"]\n",
    "\n",
    "# Read in the CSV file and convert \"?\" to NaN\n",
    "df = pd.read_csv(\"https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data\",\n",
    "                 header=None, names=headers, na_values=\"?\" )\n",
    "df.head()"
   ],
   "metadata": {
    "colab": {
     "base_uri": "https://localhost:8080/",
     "height": 352
    },
  "id": "B25cbb4PHRB5",
  "outputId": "02ac9bd4-36d8-4d1e-dee5-5f4d5dc4df8d"
 },
 "execution_count": null,
 "outputs": [
  {
   "output_type": "execute_result",
   "data": {
    "text/plain": [
     "   symboling  normalized_losses         make fuel_type aspiration num_doors  \\\n",
     "0          3                NaN  alfa-romero       gas        std       two   \n",
     "1          3                NaN  alfa-romero       gas        std       two   \n",
     "2          1                NaN  alfa-romero       gas        std       two   \n",
     "3          2              164.0         audi       gas        std      four   \n",
     "4          2              164.0         audi       gas        std      four   \n",
     "\n",
     "    body_style drive_wheels engine_location  wheel_base  ...  engine_size  \\\n",
     "0  convertible          rwd           front        88.6  ...          130   \n",
     "1  convertible          rwd           front        88.6  ...          130   \n",
     "2    hatchback          rwd           front        94.5  ...          152   \n",
     "3        sedan          fwd           front        99.8  ...          109   \n",
  "4       sedan        4wd         front      99.4  ...        136   \n",
  "\n",
  "   fuel_system  bore  stroke compression_ratio horsepower  peak_rpm city_mpg  \\\n",
  "0        mpfi  3.47    2.68               9.0      111.0    5000.0       21   \n",
  "1        mpfi  3.47    2.68               9.0      111.0    5000.0       21   \n",
  "2        mpfi  2.68    3.47               9.0      154.0    5000.0       19   \n",
  "3        mpfi  3.19    3.40              10.0      102.0    5500.0       24   \n",
  "4        mpfi  3.19    3.40               8.0      115.0    5500.0       18   \n",
  "\n",
  "   highway_mpg    price  \n",
  "0           27  13495.0  \n",
  "1           27  16500.0  \n",
  "2           26  16500.0  \n",
  "3           30  13950.0  \n",
  "4           22  17450.0  \n",
  "\n",
  "[5 rows x 26 columns]"
 ],
 "text/html": [
  "\n",
  "  <div id=\"df-b9726433-fd86-49ba-8b6c-3fb6738d59fe\">\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>symboling</th>\n",

"      <th>normalized_losses</th>\n",
"      <th>make</th>\n",
"      <th>fuel_type</th>\n",
"      <th>aspiration</th>\n",
"      <th>num_doors</th>\n",
"      <th>body_style</th>\n",
"      <th>drive_wheels</th>\n",
"      <th>engine_location</th>\n",
"      <th>wheel_base</th>\n",
"      <th>...</th>\n",
"      <th>engine_size</th>\n",
"      <th>fuel_system</th>\n",
"      <th>bore</th>\n",
"      <th>stroke</th>\n",
"      <th>compression_ratio</th>\n",
"      <th>horsepower</th>\n",
"      <th>peak_rpm</th>\n",
"      <th>city_mpg</th>\n",
"      <th>highway_mpg</th>\n",
"      <th>price</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>3</td>\n",
"      <td>NaN</td>\n",
"      <td>alfa-romero</td>\n",
"      <td>gas</td>\n",
"      <td>std</td>\n",
"      <td>two</td>\n",
"      <td>convertible</td>\n",
"      <td>rwd</td>\n",
"      <td>front</td>\n",
"      <td>88.6</td>\n",
"      <td>...</td>\n",
"      <td>130</td>\n",
"      <td>mpfi</td>\n",
"      <td>3.47</td>\n",
"      <td>2.68</td>\n",
"      <td>9.0</td>\n",
"      <td>111.0</td>\n",
"      <td>5000.0</td>\n",
"    <td>21</td>\n",
"    <td>27</td>\n",
"    <td>13495.0</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>1</th>\n",
"    <td>3</td>\n",
"    <td>NaN</td>\n",
"    <td>alfa-romero</td>\n",
"    <td>gas</td>\n",
"    <td>std</td>\n",
"    <td>two</td>\n",
"    <td>convertible</td>\n",
"    <td>rwd</td>\n",
"    <td>front</td>\n",
"    <td>88.6</td>\n",
"    <td>...</td>\n",
"    <td>130</td>\n",
"    <td>mpfi</td>\n",
"    <td>3.47</td>\n",
"    <td>2.68</td>\n",
"    <td>9.0</td>\n",
"    <td>111.0</td>\n",
"    <td>5000.0</td>\n",
"    <td>21</td>\n",
"    <td>27</td>\n",
"    <td>16500.0</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>2</th>\n",
"    <td>1</td>\n",
"    <td>NaN</td>\n",
"    <td>alfa-romero</td>\n",
"    <td>gas</td>\n",
"    <td>std</td>\n",
"    <td>two</td>\n",
"    <td>hatchback</td>\n",
"    <td>rwd</td>\n",
"    <td>front</td>\n",
"    <td>94.5</td>\n",
"    <td>...</td>\n",
"    <td>152</td>\n",
"    <td>mpfi</td>\n",
"    <td>2.68</td>\n",
"    <td>3.47</td>\n",
"    <td>9.0</td>\n",
"    <td>154.0</td>\n",
"    <td>5000.0</td>\n",
"    <td>19</td>\n",
"    <td>26</td>\n",
"    <td>16500.0</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>3</th>\n",
"    <td>2</td>\n",
"    <td>164.0</td>\n",
"    <td>audi</td>\n",
"    <td>gas</td>\n",
"    <td>std</td>\n",
"    <td>four</td>\n",
"    <td>sedan</td>\n",
"    <td>fwd</td>\n",
"    <td>front</td>\n",
"      <td>99.8</td>\n",
"      <td>...</td>\n",
"      <td>109</td>\n",
"      <td>mpfi</td>\n",
"      <td>3.19</td>\n",
"      <td>3.40</td>\n",
"      <td>10.0</td>\n",
"      <td>102.0</td>\n",
"      <td>5500.0</td>\n",
"      <td>24</td>\n",
"      <td>30</td>\n",
"      <td>13950.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>4</th>\n",
"      <td>2</td>\n",
"      <td>164.0</td>\n",
"      <td>audi</td>\n",
"      <td>gas</td>\n",
"      <td>std</td>\n",
"      <td>four</td>\n",
"      <td>sedan</td>\n",
"      <td>4wd</td>\n",
"      <td>front</td>\n",
"      <td>99.4</td>\n",
"      <td>...</td>\n",
"      <td>136</td>\n",
"      <td>mpfi</td>\n",
"      <td>3.19</td>\n",
"      <td>3.40</td>\n",
"      <td>8.0</td>\n",
"      <td>115.0</td>\n",
"      <td>5500.0</td>\n",
"      <td>18</td>\n",
"      <td>22</td>\n",
"      <td>17450.0</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>5 rows × 26 columns</p>\n",
"</div>\n",
"        <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-b9726433-fd86-49ba-8b6c-3fb6738d59fe')\"\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-b9726433-fd86-49ba-8b6c-3fb6738d59fe 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-b9726433-fd86-49ba-8b6c-3fb6738d59fe');\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": 71
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "cleanup_nums = {\"num_doors\":     {\"four\": 4, \"two\": 2},\n",
    "                \"num_cylinders\": {\"four\": 4, \"six\": 6, \"five\": 5, \"eight\": 8,\n",
    "                                  \"two\": 2, \"twelve\": 12, \"three\":3 }}"
  ],
  "metadata": {
    "id": "tUHhB034H7pK"
  },
  "execution_count": null,
  "outputs": []
},
{
```

```
"cell_type": "code",
"source": [
  "df = df.replace(cleanup_nums)\n",
  "df.head()"
],
"metadata": {
  "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 352
  },
  "id": "Gfc8exG8HyDi",
  "outputId": "f45fc007-8d67-4142-cc5c-3b1632aa2743"
},
"execution_count": null,
"outputs": [
  {
    "output_type": "execute_result",
    "data": {
      "text/plain": [
        "   symboling  normalized_losses         make fuel_type aspiration  num_doors  \\\n",
        "0          3                NaN  alfa-romero       gas        std        2.0   \n",
"1          3                 NaN  alfa-romero      gas       std       2.0   \n",

"2          1                 NaN  alfa-romero      gas       std       2.0   \n",

"3          2              164.0        audi      gas       std       4.0   \n",

"4          2              164.0        audi      gas       std       4.0   \n",

"\n",

"    body_style drive_wheels engine_location  wheel_base  ...  engine_size  \\\n",

"0  convertible          rwd           front        88.6  ...          130   \n",

"1  convertible          rwd           front        88.6  ...          130   \n",

"2    hatchback          rwd           front        94.5  ...          152   \n",

"3        sedan          fwd           front        99.8  ...          109   \n",

"4        sedan          4wd           front        99.4  ...          136   \n",

"\n",

"   fuel_system  bore  stroke compression_ratio  horsepower  peak_rpm city_mpg  \\\n",

"0         mpfi  3.47    2.68               9.0       111.0    5000.0       21   \n",

"1         mpfi  3.47    2.68               9.0       111.0    5000.0       21   \n",

"2         mpfi  2.68    3.47               9.0       154.0    5000.0       19   \n",

"3         mpfi  3.19    3.40              10.0       102.0    5500.0       24   \n",

"4         mpfi  3.19    3.40               8.0       115.0    5500.0       18   \n",

"\n",

"   highway_mpg    price  \n",

"0           27  13495.0  \n",
  "1          27  16500.0  \n",
  "2          26  16500.0  \n",
  "3          30  13950.0  \n",
  "4          22  17450.0  \n",
  "\n",
  "[5 rows x 26 columns]"
 ],
 "text/html": [
  "\n",
  "  <div id=\"df-5b96e3a7-3005-4bcf-a5a3-49ea02baa771\">\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>symboling</th>\n",

"      <th>normalized_losses</th>\n",

"      <th>make</th>\n",

"      <th>fuel_type</th>\n",

"      <th>aspiration</th>\n",

"      <th>num_doors</th>\n",

"      <th>body_style</th>\n",

"      <th>drive_wheels</th>\n",

"      <th>engine_location</th>\n",

"      <th>wheel_base</th>\n",

"      <th>...</th>\n",

"      <th>engine_size</th>\n",

"      <th>fuel_system</th>\n",
"      <th>bore</th>\n",
"      <th>stroke</th>\n",
"      <th>compression_ratio</th>\n",
"      <th>horsepower</th>\n",
"      <th>peak_rpm</th>\n",
"      <th>city_mpg</th>\n",
"      <th>highway_mpg</th>\n",
"      <th>price</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>3</td>\n",
"      <td>NaN</td>\n",
"      <td>alfa-romero</td>\n",
"      <td>gas</td>\n",
"      <td>std</td>\n",
"      <td>2.0</td>\n",
"      <td>convertible</td>\n",
"      <td>rwd</td>\n",
"      <td>front</td>\n",
"      <td>88.6</td>\n",
"      <td>...</td>\n",
"      <td>130</td>\n",
"      <td>mpfi</td>\n",
"      <td>3.47</td>\n",
"      <td>2.68</td>\n",
"      <td>9.0</td>\n",
"      <td>111.0</td>\n",
"      <td>5000.0</td>\n",
"      <td>21</td>\n",
"      <td>27</td>\n",
"      <td>13495.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>3</td>\n",
"      <td>NaN</td>\n",
"      <td>alfa-romero</td>\n",
"      <td>gas</td>\n",
"      <td>std</td>\n",
"    <td>2.0</td>\n",
"    <td>convertible</td>\n",
"    <td>rwd</td>\n",
"    <td>front</td>\n",
"    <td>88.6</td>\n",
"    <td>...</td>\n",
"    <td>130</td>\n",
"    <td>mpfi</td>\n",
"    <td>3.47</td>\n",
"    <td>2.68</td>\n",
"    <td>9.0</td>\n",
"    <td>111.0</td>\n",
"    <td>5000.0</td>\n",
"    <td>21</td>\n",
"    <td>27</td>\n",
"    <td>16500.0</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>2</th>\n",
"    <td>1</td>\n",
"    <td>NaN</td>\n",
"      <td>alfa-romero</td>\n",
"      <td>gas</td>\n",
"      <td>std</td>\n",
"      <td>2.0</td>\n",
"      <td>hatchback</td>\n",
"      <td>rwd</td>\n",
"      <td>front</td>\n",
"      <td>94.5</td>\n",
"      <td>...</td>\n",
"      <td>152</td>\n",
"      <td>mpfi</td>\n",
"      <td>2.68</td>\n",
"      <td>3.47</td>\n",
"      <td>9.0</td>\n",
"      <td>154.0</td>\n",
"      <td>5000.0</td>\n",
"      <td>19</td>\n",
"      <td>26</td>\n",
"      <td>16500.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
"      <td>2</td>\n",
"      <td>164.0</td>\n",
"      <td>audi</td>\n",
"      <td>gas</td>\n",
"      <td>std</td>\n",
"      <td>4.0</td>\n",
"      <td>sedan</td>\n",
"      <td>fwd</td>\n",
"      <td>front</td>\n",
"      <td>99.8</td>\n",
"      <td>...</td>\n",
"      <td>109</td>\n",
"      <td>mpfi</td>\n",
"      <td>3.19</td>\n",
"      <td>3.40</td>\n",
"      <td>10.0</td>\n",
"      <td>102.0</td>\n",
"      <td>5500.0</td>\n",
"      <td>24</td>\n",
"      <td>30</td>\n",
"    <td>13950.0</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>4</th>\n",
"    <td>2</td>\n",
"    <td>164.0</td>\n",
"    <td>audi</td>\n",
"    <td>gas</td>\n",
"    <td>std</td>\n",
"    <td>4.0</td>\n",
"    <td>sedan</td>\n",
"    <td>4wd</td>\n",
"    <td>front</td>\n",
"    <td>99.4</td>\n",
"    <td>...</td>\n",
"    <td>136</td>\n",
"    <td>mpfi</td>\n",
"    <td>3.19</td>\n",
"    <td>3.40</td>\n",
"    <td>8.0</td>\n",
"    <td>115.0</td>\n",
"      <td>5500.0</td>\n",
"      <td>18</td>\n",
"      <td>22</td>\n",
"      <td>17450.0</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>5 rows × 26 columns</p>\n",
"</div>\n",
"        <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-5b96e3a7-3005-4bcf-a5a3-49ea02baa771')\"\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-5b96e3a7-3005-4bcf-a5a3-49ea02baa771 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-5b96e3a7-3005-4bcf-a5a3-49ea02baa771');\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": 76
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "df.dtypes"
```

```
],
"metadata": {
  "colab": {
    "base_uri": "https://localhost:8080/"
  },
  "id": "UQqDVPddIICD",
  "outputId": "826af224-5657-477a-ed7e-87354f5f8da8"
},
"execution_count": null,
"outputs": [
  {
    "output_type": "execute_result",
    "data": {
      "text/plain": [
        "symboling              int64\n",
        "normalized_losses    float64\n",
        "make                  object\n",
        "fuel_type             object\n",
        "aspiration            object\n",
        "num_doors            float64\n",
        "body_style           object\n",
  "drive_wheels          object\n",
  "engine_location       object\n",
  "wheel_base           float64\n",
  "length               float64\n",
  "width                float64\n",
  "height               float64\n",
  "curb_weight            int64\n",
  "engine_type           object\n",
  "num_cylinders          int64\n",
  "engine_size            int64\n",
  "fuel_system           object\n",
  "bore                 float64\n",
  "stroke               float64\n",
  "compression_ratio    float64\n",
  "horsepower           float64\n",
  "peak_rpm             float64\n",
  "city_mpg               int64\n",
  "highway_mpg            int64\n",
  "price                float64\n",
  "dtype: object"
 ]
```

```
     },
     "metadata": {},
     "execution_count": 77
    }
   ]
  },
  {
   "cell_type": "markdown",
   "source": [
    "8.Split the data into training and testing"
   ],
   "metadata": {
    "id": "uNM5nftCI8Ws"
   }
  },
  {
   "cell_type": "code",
   "source": [
    "#Importing Libraries\n",
    "\n",
    "import numpy as np\n",
    "import matplotlib.pyplot as plt\n",
    "import pandas as pd\n",
    "\n",
    "#Importing data\n",
    "dataset = pd.read_csv(‘Decision Tree Data.csv’)\n",
    "x = dataset.iloc[:,1:2].values\n",
    "y =dataset.iloc[:,2].values\n",
    "#Split Training Set and Testing Set\n",
    "from sklearn.cross_validation import train_test_split\n",
    "xtrain, xtest, ytrain, ytest =train_test_split(x,y,test_size=0.2"
   ],
   "metadata": {
    "id": "lIwd7oXxI4Vh"
   },
   "execution_count": null,
   "outputs": []
  },
  {
   "cell_type": "markdown",
   "source": [
    "9.Split the data into dependent and independent variables."
```

```
 ],
 "metadata": {
  "id": "haxKbqG0JPef"
 }
},
{
 "cell_type": "code",
 "source": [
  "X = df.iloc[:, :-1].values\n",
  "print(X)"
 ],
 "metadata": {
  "colab": {
   "base_uri": "https://localhost:8080/"
  },
  "id": "FTc5S_bXI9j5",
  "outputId": "4c270494-aae3-4106-e8ae-a1ed083acf78"
 },
 "execution_count": null,
 "outputs": [
  {
     "output_type": "stream",
     "name": "stdout",
     "text": [
      "[[3 nan 'alfa-romero' ... 5000.0 21 27]\n",
      " [3 nan 'alfa-romero' ... 5000.0 21 27]\n",
      " [1 nan 'alfa-romero' ... 5000.0 19 26]\n",
      " ...\n",
      " [-1 95.0 'volvo' ... 5500.0 18 23]\n",
      " [-1 95.0 'volvo' ... 4800.0 26 27]\n",
      " [-1 95.0 'volvo' ... 5400.0 19 25]]\n"
     ]
    }
   ]
  },
  {
   "cell_type": "code",
   "source": [
    "Y = df.iloc[:, -1].values\n",
    "print(Y)"
   ],
   "metadata": {
```

```
  "colab": {
    "base_uri": "https://localhost:8080/"
  },
  "id": "1-t7q0bDJVig",
  "outputId": "28af3185-e275-406e-8b25-de9efaaaf0f4"
},
"execution_count": null,
"outputs": [
  {
    "output_type": "stream",
    "name": "stdout",
    "text": [
      "[13495. 16500. 16500. 13950. 17450. 15250. 17710. 18920. 23875.    nan\n",
      " 16430. 16925. 20970. 21105. 24565. 30760. 41315. 36880.  5151.  6295.\n",
      "  6575.  5572.  6377.  7957.  6229.  6692.  7609.  8558.  8921. 12964.\n",
      "  6479.  6855.  5399.  6529.  7129.  7295.  7295.  7895.  9095.  8845.\n",
      " 10295. 12945. 10345.  6785.    nan    nan 11048. 32250. 35550. 36000.\n",
      "  5195.  6095.  6795.  6695.  7395. 10945. 11845. 13645. 15645.  8845.\n",
      "  8495. 10595. 10245. 10795. 11245. 18280. 18344. 25552. 28248. 28176.\n",
      " 31600. 34184. 35056. 40960. 45400. 16503.  5389.  6189.  6669.  7689.\n",
      "  9959.  8499. 12629. 14869. 14489.  6989.  8189.  9279.  9279.  5499.\n",
      "  7099.  6649.  6849.  7349.  7299.  7799.  7499.  7999.  8249.  8949.\n",
      "  9549. 13499. 14399. 13499. 17199. 19699. 18399. 11900. 13200. 12440.\n",
      " 13860. 15580. 16900. 16695. 17075. 16630. 17950. 18150.  5572.  7957.\n",
      "  6229.  6692.  7609.  8921. 12764. 22018. 32528. 34028. 37028.    nan\n",
      "  9295.  9895. 11850. 12170. 15040. 15510. 18150. 18620.  5118.  7053.\n",
      "  7603.  7126.  7775.  9960.  9233. 11259.  7463. 10198.  8013. 11694.\n",
      "  5348.  6338.  6488.  6918.  7898.  8778.  6938.  7198.  7898.  7788.\n",
      "  7738.  8358.  9258.  8058.  8238.  9298.  9538.  8449.  9639.  9989.\n",
      " 11199. 11549. 17669.  8948. 10698.  9988. 10898. 11248. 16558. 15998.\n",
      " 15690. 15750.  7775.  7975.  7995.  8195.  8495.  9495.  9995. 11595.\n",
      "  9980. 13295. 13845. 12290. 12940. 13415. 15985. 16515. 18420. 18950.\n",
      " 16845. 19045. 21485. 22470. 22625.]\n"
     ]
    }
   ]
  },
  {
   "cell_type": "markdown",
   "source": [
    "10.Scale the independent variables"
   ],
   "metadata": {
    "id": "zmj4YIejKuzE"
   }
  },
  {
   "cell_type": "code",
   "source": [
    "columns = df.columns\n",
    "binary_cols = []\n",
    "for col in columns:\n",
    "    if df[col].value_counts().shape[0] == 2:\n",
    "        binary_cols.append(col)"
   ],
   "metadata": {
    "id": "G9RMWIQyJhsk"
   },
   "execution_count": null,
   "outputs": []
  }
 ]
}
```