{
  "cells": [
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "Ao28GXYY0OwJ"
      },
      "source": [
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "27Kr-QhPQoLD"
      },
      "source": [
        "# 1. Importing the required libraries"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {
        "id": "PjWxu-T00G6J"
      },
      "outputs": [],
      "source": [
        "import pandas as pd\n",
        "import seaborn as sns\n",
        "import matplotlib.pyplot as plt\n",
        "import numpy as np"
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "Lc8iAaB90eMl"
      },
      "source": [
        "# 2. Loading the dataset"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 4,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 206
        },
        "id": "vu_3hYCR0gjU",
        "outputId": "f31bbf6b-faf2-453e-b1d0-c6c79acc0572"
      },
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              " Sex Length Diameter Height Whole weight Shucked
weight Viscera weight \\\n",

```
    "0   M   0.455     0.365   0.095        0.5140
0.2245         0.1010   \n",
    "1   M   0.350     0.265   0.090        0.2255
0.0995         0.0485   \n",
    "2   F   0.530     0.420   0.135        0.6770
0.2565         0.1415   \n",
    "3   M   0.440     0.365   0.125        0.5160
0.2155         0.1140   \n",
    "4   I   0.330     0.255   0.080        0.2050
0.0895         0.0395   \n",
    "\n",
    "   Shell weight  Rings  \n",
    "0         0.150     15  \n",
    "1         0.070      7  \n",
    "2         0.210      9  \n",
    "3         0.155     10  \n",
    "4         0.055      7  "
   ],
   "text/html": [
    "\n",
    "  <div id=\"df-c24ef3fe-6538-4446-9001-0cf48d107d05\">\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>Sex</th>\n",
    "      <th>Length</th>\n",
    "      <th>Diameter</th>\n",
    "      <th>Height</th>\n",
    "      <th>Whole weight</th>\n",
    "      <th>Shucked weight</th>\n",
    "      <th>Viscera weight</th>\n",
    "  <th>Shell weight</th>\n",
    "      <th>Rings</th>\n",
    "    </tr>\n",
    "  </thead>\n",
    "  <tbody>\n",
    "    <tr>\n",
    "      <th>0</th>\n",
    "      <td>M</td>\n",
    "      <td>0.455</td>\n",
    "      <td>0.365</td>\n",
    "      <td>0.095</td>\n",
    "      <td>0.5140</td>\n",
    "      <td>0.2245</td>\n",
    "      <td>0.1010</td>\n",
```

```
              "      <td>0.150</td>\n",
              "      <td>15</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>1</th>\n",
              "      <td>M</td>\n",
              "      <td>0.350</td>\n",
              "      <td>0.265</td>\n",
              "      <td>0.090</td>\n",
              "      <td>0.2255</td>\n",
              "      <td>0.0995</td>\n",
              "      <td>0.0485</td>\n",
              "      <td>0.070</td>\n",
              "      <td>7</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>2</th>\n",
              "      <td>F</td>\n",
              "      <td>0.530</td>\n",
              "      <td>0.420</td>\n",
              "      <td>0.135</td>\n",
              "      <td>0.6770</td>\n",
              "      <td>0.2565</td>\n",
              "      <td>0.1415</td>\n",
              "      <td>0.210</td>\n",
              "      <td>9</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>3</th>\n",
              "      <td>M</td>\n",
              "      <td>0.440</td>\n",
              "      <td>0.365</td>\n",
              "      <td>0.125</td>\n",
              "      <td>0.5160</td>\n",
              "      <td>0.2155</td>\n",
              "      <td>0.1140</td>\n",
              "      <td>0.155</td>\n",
              "      <td>10</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>4</th>\n",
              "      <td>I</td>\n",
              "      <td>0.330</td>\n",
              "      <td>0.255</td>\n",
              "      <td>0.080</td>\n",
              "      <td>0.2050</td>\n",
              "      <td>0.0895</td>\n",
              "      <td>0.0395</td>\n",
              "      <td>0.055</td>\n",
              "      <td>7</td>\n",
              "    </tr>\n",
              "  </tbody>\n",
              "</table>\n",
              "</div>\n",
              "      <button class=\"colab-df-convert\"
onclick=\"convertToInteractive('df-c24ef3fe-6538-4446-9001-
0cf48d107d05')\"\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-c24ef3fe-6538-4446-
9001-0cf48d107d05 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-
c24ef3fe-6538-4446-9001-0cf48d107d05');\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&lt;/a&gt;'\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",
              "      &lt;/script&gt;\n",
              "    </div>\n",
              "  </div>\n",
              "  "
            ]
          },
          "metadata": {},
          "execution_count": 4
        }
      ],
      "source": [
        "df=pd.read_csv(\"/content/sample_data/abalone.csv\")\n",
        "df.head()"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 6,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "1JxWN30i0x0o",
        "outputId": "d71ecf64-ab69-41a2-bd1d-7e77a8e90d4c"
      },
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(4177, 9)"
            ]
          },
          "metadata": {},
          "execution_count": 6
```

```
        }
      ],
      "source": [
        "df.shape"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 7,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 206
        },
        "id": "_VZo4loE0_y9",
        "outputId": "2c4df058-605a-4965-b8d3-ecd20455c6b2"
      },
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              " Sex Length Diameter Height Whole_weight
Shucked_weight  Viscera_weight  \\\n",
              "0   M   0.455     0.365   0.095        0.5140
0.2245          0.1010   \n",
              "1   M   0.350     0.265   0.090        0.2255
0.0995          0.0485   \n",
              "2   F   0.530     0.420   0.135        0.6770
0.2565          0.1415   \n",
              "3   M   0.440     0.365   0.125        0.5160
0.2155          0.1140   \n",
              "4   I   0.330     0.255   0.080        0.2050
0.0895          0.0395   \n",
              "\n",
              "   Shell_weight   Age  \n",
              "0         0.150  16.5  \n",
              "1         0.070   8.5  \n",
              "2         0.210  10.5  \n",
              "3         0.155  11.5  \n",
              "4         0.055   8.5  "
            ],
            "text/html": [
              "\n",
              "  <div id=\"df-1ebac415-9a3b-46a9-aa15-35f213ad2e13\">\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>Sex</th>\n",
"      <th>Length</th>\n",
"      <th>Diameter</th>\n",
"      <th>Height</th>\n",
"      <th>Whole_weight</th>\n",
"      <th>Shucked_weight</th>\n",
"      <th>Viscera_weight</th>\n",
"  <th>Shell_weight</th>\n",
"      <th>Age</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>M</td>\n",
"      <td>0.455</td>\n",
"      <td>0.365</td>\n",
"      <td>0.095</td>\n",
"      <td>0.5140</td>\n",
"      <td>0.2245</td>\n",
"      <td>0.1010</td>\n",
"      <td>0.150</td>\n",
"      <td>16.5</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>M</td>\n",
"      <td>0.350</td>\n",
"      <td>0.265</td>\n",
"      <td>0.090</td>\n",
"      <td>0.2255</td>\n",
"      <td>0.0995</td>\n",
"      <td>0.0485</td>\n",
"      <td>0.070</td>\n",
"      <td>8.5</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>F</td>\n",
"      <td>0.530</td>\n",
"      <td>0.420</td>\n",
"      <td>0.135</td>\n",
"      <td>0.6770</td>\n",
"      <td>0.2565</td>\n",
"      <td>0.1415</td>\n",
"      <td>0.210</td>\n",
"      <td>10.5</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
"      <td>M</td>\n",
"      <td>0.440</td>\n",
"      <td>0.365</td>\n",
"      <td>0.125</td>\n",
"      <td>0.5160</td>\n",
"      <td>0.2155</td>\n",
"      <td>0.1140</td>\n",
"      <td>0.155</td>\n",
```

```
              "      <td>11.5</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>4</th>\n",
              "      <td>I</td>\n",
              "      <td>0.330</td>\n",
              "      <td>0.255</td>\n",
              "      <td>0.080</td>\n",
              "      <td>0.2050</td>\n",
              "      <td>0.0895</td>\n",
              "      <td>0.0395</td>\n",
              "      <td>0.055</td>\n",
              "      <td>8.5</td>\n",
              "    </tr>\n",
              "  </tbody>\n",
              "</table>\n",
              "</div>\n",
              "      <button class=\"colab-df-convert\"
onclick=\"convertToInteractive('df-1ebac415-9a3b-46a9-aa15-
35f213ad2e13')\"\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",
            "  &lt;/style&gt;\n",
            "\n",
            "      &lt;script&gt;\n",
            "        const buttonEl =\n",
            "          document.querySelector('#df-1ebac415-9a3b-46a9-
aa15-35f213ad2e13 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-
1ebac415-9a3b-46a9-aa15-35f213ad2e13');\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&lt;/a&gt;'\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",
            "      &lt;/script&gt;\n",
            "    </div>\n",
            "  </div>\n",
            "  "
          ]
        },
        "metadata": {},
        "execution_count": 7
      }
    ],
    "source": [
```

```
        "Age=1.5+df.Rings\n",
        "df[\"Age\"]=Age\n",
        "df=df.rename(columns = {'Whole weight':'Whole_weight','Shucked
weight': 'Shucked_weight','Viscera weight': 'Viscera_weight',\n",
        "                              'Shell weight': 'Shell_weight'})\n",
        "df=df.drop(columns=[\"Rings\"],axis=1)\n",
        "df.head()"
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "_I-0WpBS1O3L"
      },
      "source": [
        "# 3. Performing Visualizations"
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "ULjrwPcF1e0l"
      },
      "source": [
        "## **(i) Univariate Analysis** "
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "zSBzfkYg1Qjl"
      },
      "source": [
        "### Histogram"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 8,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 386
        },
        "id": "uvWyiD9Q2JKe",
        "outputId": "6fe98c08-47f6-4847-f4be-23b338684a7e"
      },
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "<seaborn.axisgrid.FacetGrid at 0x7f105a943e10>"
            ]
          },
          "metadata": {},
          "execution_count": 8
        },
        {
          "output_type": "display_data",
          "data": {
```

```
        "text/plain": [
          "<Figure size 360x360 with 1 Axes>"
        ],
        "image/png":           },
      "metadata": {
        "needs_background": "light"
      }
    }
  ],
  "source": [
    "sns.displot(df[\"Age\"], color='lightblue')"
  ]
},
{
  "cell_type": "code",
  "execution_count": 9,
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 296
    },
    "id": "vcayp0l-2MmA",
    "outputId": "e5d27b9f-2295-4212-a04f-830ff4c19331"
  },
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "<matplotlib.axes._subplots.AxesSubplot at 0x7f10579ca050>"
        ]
      },
      "metadata": {},
      "execution_count": 9
    },
    {
      "output_type": "display_data",
      "data": {
        "text/plain": [
          "<Figure size 432x288 with 1 Axes>"
        ],
        "image/png":            },
      "metadata": {
        "needs_background": "light"
      }
    }
  ],
  "source": [
    "sns.histplot(y=df.Age,color='green') "
  ]
},
{
  "cell_type": "code",
  "execution_count": 10,
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 296
    },
    "id": "D9Bm92jk2PmS",
    "outputId": "3975e559-6086-4fc3-de67-7e5d9e90a5da"
```

```
      },
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "<matplotlib.axes._subplots.AxesSubplot at 0x7f10578ae8d0>"
            ]
          },
          "metadata": {},
          "execution_count": 10
        },
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
            ],
            "image/png":           },
          "metadata": {
            "needs_background": "light"
          }
        }
      ],
      "source": [
        "sns.histplot(x=df.Age,color='yellow') "
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "ikzi_PFS2Sib"
      },
      "source": [
        "### Boxplot"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 11,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 296
        },
        "id": "kIinXK0V2UCp",
        "outputId": "cd3239ba-4a75-4493-99e4-b4bb1eb92764"
      },
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "<matplotlib.axes._subplots.AxesSubplot at 0x7f105777ff10>"
            ]
          },
          "metadata": {},
          "execution_count": 11
        },
        {
          "output_type": "display_data",
```

```
        "data": {
          "text/plain": [
            "<Figure size 432x288 with 1 Axes>"
          ],
          "image/png":            },
        "metadata": {
          "needs_background": "light"
        }
      }
    ],
    "source": [
      "sns.boxplot(x=df.Age,color='orange') "
    ]
  },
  {
    "cell_type": "markdown",
    "metadata": {
      "id": "MtC8V3TW2YXt"
    },
    "source": [
      "### Countplot"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 12,
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/",
        "height": 296
      },
      "id": "s1xlos2u2ayh",
      "outputId": "f9eb5356-0fab-4b7f-eb7c-52764efa6d84"
    },
    "outputs": [
      {
        "output_type": "execute_result",
        "data": {
          "text/plain": [
            "<matplotlib.axes._subplots.AxesSubplot at 0x7f1057717510>"
          ]
        },
        "metadata": {},
        "execution_count": 12
      },
      {
        "output_type": "display_data",
        "data": {
          "text/plain": [
            "<Figure size 432x288 with 1 Axes>"
          ],
          "image/png":            },
        "metadata": {
          "needs_background": "light"
        }
      }
    ],
    "source": [
      "sns.countplot(x=df.Age) "
    ]
  },
```

```
{
  "cell_type": "markdown",
  "metadata": {
    "id": "rcGpFWby2g3s"
  },
  "source": [
    "# **(ii) Bi-Variate Analysis**"
  ]
},
{
  "cell_type": "markdown",
  "metadata": {
    "id": "zuuqZw0T2pWk"
  },
  "source": [
    "### Barplot"
  ]
},
{
  "cell_type": "code",
  "execution_count": 14,
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 296
    },
    "id": "WTMjt9JD2tQf",
    "outputId": "06097670-ea52-4fab-e181-9abc9393ab6e"
  },
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "<matplotlib.axes._subplots.AxesSubplot at 0x7f10576f5390>"
        ]
      },
      "metadata": {},
      "execution_count": 14
    },
    {
      "output_type": "display_data",
      "data": {
        "text/plain": [
          "<Figure size 432x288 with 1 Axes>"
        ],
        "image/png":
```

1. Importing the required libraries

In [1]:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

2. Loading the dataset

In [4]:

```
df=pd.read_csv("/content/sample_data/abalone.csv")
df.head()
```

Out[4]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

In [6]:

```
df.shape
```

Out[6]:

```
(4177, 9)
```

In [7]:

```
Age=1.5+df.Rings
df["Age"]=Age
df=df.rename(columns = {'Whole weight':'Whole_weight','Shucked weight':
'Shucked_weight','Viscera weight': 'Viscera_weight',
                              'Shell weight': 'Shell_weight'})
df=df.drop(columns=["Rings"],axis=1)
df.head()
```

Out[7]:

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	16.5

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	8.5
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	10.5
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	11.5
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	8.5

3. Performing Visualizations

(i) Univariate Analysis

Histogram

In [8]:

```
sns.displot(df["Age"], color='lightblue')
```

Out[8]:

```
<seaborn.axisgrid.FacetGrid at 0x7f105a943e10>
```

In [9]:

```
sns.histplot(y=df.Age,color='green')
```

Out[9]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f10579ca050>
```

In [10]:

```
sns.histplot(x=df.Age,color='yellow')
```

Out[10]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f10578ae8d0>
```

Boxplot

In [11]:

```
sns.boxplot(x=df.Age,color='orange')
```

Out[11]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f105777ff10>
```

Countplot

In [12]:

```
sns.countplot(x=df.Age)
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1057717510>
```

(ii) Bi-Variate Analysis

Barplot

In [14]:

```
sns.barplot(x=df.Height,y=df.Age)
```

Out[14]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f10576f5390>
```

Linearplot

In [15]:

```
sns.lineplot(x=df.Age,y=df.Height, color='darkblue')
```

Out[15]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f10577137d0>
```

Scatterplot

In [16]:

```
sns.scatterplot(x=df.Age,y=df.Height,color='green')
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f10575d5150>
```

Pointplot

In [17]:

```
sns.pointplot(x=df.Age, y=df.Height, color="blue")
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f10573fee10>
```

Regplot

In [18]:

```
sns.regplot(x=df.Age,y=df.Height,color='orange')
```

Out[18]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f105719f6d0>
```

(

iii) Multi-Variate Analysis

Pairplot

In [20]:

```
sns.pairplot(data=df[["Height","Length","Diameter","Age","Whole_weight","Shucked_weight","Viscera_weight","Shell_weight"]])
```

Out[20]:

```
<seaborn.axisgrid.PairGrid at 0x7f105710d850>
```

4. Perform descriptive statistics on the dataset

In [21]:

```
df.describe(include='all')
```

Out[21]:

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
count	4177	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000
unique	3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	M	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	1528	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	NaN	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	11.433684
std	NaN	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	NaN	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	2.500000
25%	NaN	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	9.500000
50%	NaN	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	10.500000

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
75%	NaN	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	12.500000
max	NaN	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	30.500000

5. Check for Missing values and deal with them

In [22]:

```
df.isnull().sum()
```

Out[22]:

```
Sex               0
Length            0
Diameter          0
Height            0
Whole_weight      0
Shucked_weight    0
Viscera_weight    0
Shell_weight      0
Age               0
dtype: int64
```

6. Find the outliers and replace the outliers

In [26]:

```
outliers=df.quantile(q=(0.25,0.75))
outliers
```

Out[26]:

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
0.25	0.450	0.35	0.115	0.4415	0.186	0.0935	0.130	9.5
0.75	0.615	0.48	0.165	1.1530	0.502	0.2530	0.329	12.5

In [32]:

```
a = df.Age.quantile(0.25)
b = df.Age.quantile(0.75)
c = b - a
lower_limit = a - 1.5 * c
df.median(numeric_only=True)
```

Out[32]:

```
Length            0.5450
```

```
Diameter           0.4250
Height             0.1400
Whole_weight       0.7995
Shucked_weight     0.3360
Viscera_weight     0.1710
Shell_weight       0.2340
Age               10.5000
dtype: float64
```

In [33]:

```
df['Age'] = np.where(df['Age'] < lower_limit, 7, df['Age'])
sns.boxplot(x=df.Age,showfliers = False)
```

Out[33]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f105535ead0>
```

7. Check for categorical columns and perform encoding

```
df.head()
```

Out[34]:

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	16.5
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	8.5
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	10.5
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	11.5
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	8.5

In [36]:

```
from sklearn.preprocessing import LabelEncoder

lab1 = LabelEncoder()
df.Sex = lab1.fit_transform(df.Sex)

df.head()
```

Out[36]:

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
0	2	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	16.5
1	2	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	8.5
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	10.5
3	2	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	11.5
4	1	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	8.5

8. Split the data into dependent and independent variables

In [39]:

```
y = df["Sex"]
y.head()
```

Out[39]:

```
0    2
1    2
2    0
3    2
4    1
Name: Sex, dtype: int64
```

In [40]:

```
x=df.drop(columns=["Sex"],axis=1)
x.head()
```

Out[40]:

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
0	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	16.5
1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	8.5
2	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	10.5

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
3	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	11.5
4	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	8.5

9. Scale the independent variables

In [41]:

```
from sklearn.preprocessing import scale
x_scaled = pd.DataFrame(scale(x), columns=x.columns)
x_scaled.head()
```

Out[41]:

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
0	-0.574558	-0.432149	-1.064424	-0.641898	-0.607685	-0.726212	-0.638217	1.577830
1	-1.448986	-1.439929	-1.183978	-1.230277	-1.170910	-1.205221	-1.212987	-0.919022
2	0.050033	0.122130	-0.107991	-0.309469	-0.463500	-0.356690	-0.207139	-0.294809
3	-0.699476	-0.432149	-0.347099	-0.637819	-0.648238	-0.607600	-0.602294	0.017298
4	-1.615544	-1.540707	-1.423087	-1.272086	-1.215968	-1.287337	-1.320757	-0.919022

10. Split the data into training and testing

In [42]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(x_scaled, y,
test_size=0.2, random_state=1)
```

In [44]:

```
X_train.shape,X_test.shape
```

Out[44]:

```
((3341, 8), (836, 8))
```

In [45]:

```
Y_train.shape,Y_test.shape
```

Out[45]:

```
((3341,), (836,))
```

In [46]:

```
X_train.head()
```

Out[46]:

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
666	-0.574558	-0.583316	-0.466653	-0.704101	-0.801435	-0.333880	-0.566371	0.329404
2813	-2.240135	-2.145375	-2.020857	-1.542312	-1.490821	-1.492627	-1.565034	-1.855341
1862	-0.033246	0.021352	-0.705762	-0.632721	-0.643732	-0.812890	-0.393939	-0.606915
3684	0.799543	0.626020	0.370226	0.279929	0.394854	-0.087532	0.324524	0.329404
551	0.757903	0.827576	0.370226	0.325817	0.248416	0.131444	0.762786	0.953617

In [47]:

```
X_test.head()
```

Out[47]:

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
17	-0.699476	-0.684094	-0.944870	-0.770383	-0.772147	-0.853947	-0.781909	0.017298
1131	0.341509	0.273297	0.250672	0.328876	0.991872	0.017394	-0.235878	-0.606915

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	Age
299	-1.282428	-1.288762	-0.825316	-1.212942	-1.211462	-1.113981	-1.177064	-0.294809
1338	0.466427	0.474853	-0.107991	-0.067795	0.205611	-0.124028	-0.250247	0.017298
2383	0.008394	-0.180204	-0.107991	-0.465487	-0.598674	-0.452492	-0.207139	1.889936

In [48]:

```
Y_train.head()
```

Out[48]:

```
666     2
2813    1
1862    1
3684    1
551     1
Name: Sex, dtype: int64
```

In [49]:

```
Y_test.head()
```

Out[49]:

```
17      0
1131    2
299     2
1338    2
2383    0
Name: Sex, dtype: int64
```

11. Build the Model

In [50]:

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators=10,criterion='entropy')
```

In [51]:

```
model.fit(X_train,Y_train)
```

Out[51]:

```
RandomForestClassifier(criterion='entropy', n_estimators=10)
```

In [52]:

```
y_predict = model.predict(X_test)
```

In [56]:

```
y_predict_train = model.predict(X_train)
```

12. Train the Model

In [54]:

```
from sklearn.metrics import
accuracy_score,confusion_matrix,classification_report
```

In [60]:

```
print('Training accuracy: ',accuracy_score(Y_train,y_predict_train))
```

```
Training accuracy:  0.980544747081712
```

13.Test the Model

In [61]:

```
print('Testing accuracy: ',accuracy_score(Y_test,y_predict))
```

```
Testing accuracy:  0.5215311004784688
```

14. Measure the performance using Metrics

In [62]:

```
pd.crosstab(Y_test,y_predict)
```

Out[62]:

col_0	0	1	2
Sex			
0	122	31	107
1	26	198	40
2	148	48	116

In [63]:

```
print(classification_report(Y_test,y_predict))
```

```
              precision    recall  f1-score   support

           0       0.41      0.47      0.44       260
           1       0.71      0.75      0.73       264
           2       0.44      0.37      0.40       312

    accuracy                           0.52       836
   macro avg       0.52      0.53      0.52       836
weighted avg       0.52      0.52      0.52       836
```

In [ ]: