{

"cells": [

{

"cell\_type": "markdown",

"id": "457c4795",

"metadata": {},

"source": [

"# Problem Statement: Abalone Age Prediction"

]

},

{

"cell\_type": "markdown",

"id": "1ecae4cb",

"metadata": {},

"source": [

"## 1. Download the dataset: Dataset\n",

"## 2. Load the dataset into the tool."

]

},

{

"cell\_type": "code",

"execution\_count": 172,

"id": "40136f5d",

"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>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",

" <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>15</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>7</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>9</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>10</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>7</td>\n",

" <td>8.5</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"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 Age \n",

"0 0.150 15 16.5 \n",

"1 0.070 7 8.5 \n",

"2 0.210 9 10.5 \n",

"3 0.155 10 11.5 \n",

"4 0.055 7 8.5 "

]

},

"execution\_count": 172,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"import numpy as np\n",

"import pandas as pd\n",

"\n",

"ds=pd.read\_csv(\"abalone.csv\")\n",

"\n",

"# Rings / integer / -- / +1.5 gives the age in years\n",

"\n",

"ds['Age']=ds[\"Rings\"]+1.5\n",

"\n",

"ds.head(5)"

]

},

{

"cell\_type": "markdown",

"id": "8d1151d0",

"metadata": {},

"source": [

"## 3. Perform Below Visualizations.\n",

"## ∙ Univariate Analysis\n",

"## ∙ Bi-Variate Analysis\n",

"## ∙ Multi-Variate Analysis"

]

},

{

"cell\_type": "code",

"execution\_count": 179,

"id": "aef5ac2b",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Frequency table for Age is given below\n",

"11.5 32\n",

"10.5 28\n",

"8.5 20\n",

"9.5 18\n",

"13.5 17\n",

"12.5 16\n",

"14.5 13\n",

"15.5 11\n",

"16.5 10\n",

"17.5 7\n",

"6.5 6\n",

"7.5 5\n",

"21.5 4\n",

"5.5 4\n",

"20.5 3\n",

"19.5 3\n",

"22.5 2\n",

"18.5 1\n",

"Name: Age, dtype: int64\n",

"\n",

"\n",

"\n",

"Mean, Median, std \n",

"\n",

"1-mean of age = 12.235\n",

"2-mean of height = 0.13482500000000003\n",

"3-median value of length = 0.53\n",

"4-standard devation of whole weight = 0.48292555269001314\n",

"5-frequency table for rings = \n",

" 10 32\n",

"9 28\n",

"7 20\n",

"8 18\n",

"12 17\n",

"11 16\n",

"13 13\n",

"14 11\n",

"15 10\n",

"16 7\n",

"5 6\n",

"6 5\n",

"20 4\n",

"4 4\n",

"19 3\n",

"18 3\n",

"21 2\n",

"17 1\n",

"Name: Rings, dtype: int64\n",

"\n",

"Chart\n",

"\n",

"6-boxplot of Diameter\n"

]

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 2 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"# univarient analysis\n",

"\n",

"#frequency table for age\n",

"\n",

"ft = ds1['Age'].value\_counts()\n",

"\n",

"print(\"Frequency table for Age is given below\")\n",

"print(\"{}\\n\\n\\n\".format(ft))\n",

"\n",

"# mean\n",

"\n",

"print(\"Mean, Median, std \\n\")\n",

"ma=ds1['Age'].mean() #mean of age\n",

"mh = ds1['Height'].mean() #mean of height\n",

"mel = ds1['Length'].median() #median value of length\n",

"stw = ds1['Whole weight'].std() #standard devation of whole weight\n",

"\n",

"\n",

"#chart\n",

"\n",

"import matplotlib.pyplot as plt # library for plot or graph\n",

"import seaborn as sns\n",

"\n",

"plt.subplot(1,2,1)\n",

"ch = ds1.boxplot(column='Diameter',grid=True,color ='red')\n",

"plt.title('Box plot')\n",

"\n",

"plt.subplot(1,2,2)\n",

"DC = sns.kdeplot(ds1['Diameter'])\n",

"plt.title('Density Curve')\n",

"\n",

"\n",

"\n",

"print(\"1-mean of age = \",ma) \n",

"print(\"2-mean of height = \",mh) \n",

"print(\"3-median value of length = \",mel)#\n",

"print(\"4-standard devation of whole weight = \",stw) \n",

"print(\"5-frequency table for rings = \\n {}\" .format(fre)) \n",

"print(\"\\nChart\\n\\n6-boxplot of Diameter\",flush=True)\n",

"\n"

]

},

{

"cell\_type": "code",

"execution\_count": 96,

"id": "2f61eac9",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"AxesSubplot(0.125,0.125;0.775x0.755)\n"

]

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"#multi-varient analysis\n",

"\n",

"import matplotlib.pyplot as plt\n",

"import seaborn as sns\n",

"\n",

"ds1=ds.head(200)\n",

"df=sns.countplot(x=\"Viscera weight\",hue='Sex',data=ds1)\n",

"\n",

"print(df)"

]

},

{

"cell\_type": "markdown",

"id": "e7150cef",

"metadata": {},

"source": [

"## 4. Perform descriptive statistics on the dataset."

]

},

{

"cell\_type": "code",

"execution\_count": 97,

"id": "4e1cad43",

"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>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",

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

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" <td>4177.000000</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>0.523992</td>\n",

" <td>0.407881</td>\n",

" <td>0.139516</td>\n",

" <td>0.828742</td>\n",

" <td>0.359367</td>\n",

" <td>0.180594</td>\n",

" <td>0.238831</td>\n",

" <td>9.933684</td>\n",

" <td>11.433684</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>0.120093</td>\n",

" <td>0.099240</td>\n",

" <td>0.041827</td>\n",

" <td>0.490389</td>\n",

" <td>0.221963</td>\n",

" <td>0.109614</td>\n",

" <td>0.139203</td>\n",

" <td>3.224169</td>\n",

" <td>3.224169</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>0.075000</td>\n",

" <td>0.055000</td>\n",

" <td>0.000000</td>\n",

" <td>0.002000</td>\n",

" <td>0.001000</td>\n",

" <td>0.000500</td>\n",

" <td>0.001500</td>\n",

" <td>1.000000</td>\n",

" <td>2.500000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>25%</th>\n",

" <td>0.450000</td>\n",

" <td>0.350000</td>\n",

" <td>0.115000</td>\n",

" <td>0.441500</td>\n",

" <td>0.186000</td>\n",

" <td>0.093500</td>\n",

" <td>0.130000</td>\n",

" <td>8.000000</td>\n",

" <td>9.500000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>50%</th>\n",

" <td>0.545000</td>\n",

" <td>0.425000</td>\n",

" <td>0.140000</td>\n",

" <td>0.799500</td>\n",

" <td>0.336000</td>\n",

" <td>0.171000</td>\n",

" <td>0.234000</td>\n",

" <td>9.000000</td>\n",

" <td>10.500000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>75%</th>\n",

" <td>0.615000</td>\n",

" <td>0.480000</td>\n",

" <td>0.165000</td>\n",

" <td>1.153000</td>\n",

" <td>0.502000</td>\n",

" <td>0.253000</td>\n",

" <td>0.329000</td>\n",

" <td>11.000000</td>\n",

" <td>12.500000</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>0.815000</td>\n",

" <td>0.650000</td>\n",

" <td>1.130000</td>\n",

" <td>2.825500</td>\n",

" <td>1.488000</td>\n",

" <td>0.760000</td>\n",

" <td>1.005000</td>\n",

" <td>29.000000</td>\n",

" <td>30.500000</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" Length Diameter Height Whole weight Shucked weight \\\n",

"count 4177.000000 4177.000000 4177.000000 4177.000000 4177.000000 \n",

"mean 0.523992 0.407881 0.139516 0.828742 0.359367 \n",

"std 0.120093 0.099240 0.041827 0.490389 0.221963 \n",

"min 0.075000 0.055000 0.000000 0.002000 0.001000 \n",

"25% 0.450000 0.350000 0.115000 0.441500 0.186000 \n",

"50% 0.545000 0.425000 0.140000 0.799500 0.336000 \n",

"75% 0.615000 0.480000 0.165000 1.153000 0.502000 \n",

"max 0.815000 0.650000 1.130000 2.825500 1.488000 \n",

"\n",

" Viscera weight Shell weight Rings Age \n",

"count 4177.000000 4177.000000 4177.000000 4177.000000 \n",

"mean 0.180594 0.238831 9.933684 11.433684 \n",

"std 0.109614 0.139203 3.224169 3.224169 \n",

"min 0.000500 0.001500 1.000000 2.500000 \n",

"25% 0.093500 0.130000 8.000000 9.500000 \n",

"50% 0.171000 0.234000 9.000000 10.500000 \n",

"75% 0.253000 0.329000 11.000000 12.500000 \n",

"max 0.760000 1.005000 29.000000 30.500000 "

]

},

"execution\_count": 97,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"ds.describe()"

]

},

{

"cell\_type": "markdown",

"id": "870c7a54",

"metadata": {},

"source": [

"## 5. Check for Missing values and deal with them."

]

},

{

"cell\_type": "code",

"execution\_count": 98,

"id": "a615c07b",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"<class 'pandas.core.frame.DataFrame'>\n",

"RangeIndex: 4177 entries, 0 to 4176\n",

"Data columns (total 10 columns):\n",

" # Column Non-Null Count Dtype \n",

"--- ------ -------------- ----- \n",

" 0 Sex 4177 non-null object \n",

" 1 Length 4177 non-null float64\n",

" 2 Diameter 4177 non-null float64\n",

" 3 Height 4177 non-null float64\n",

" 4 Whole weight 4177 non-null float64\n",

" 5 Shucked weight 4177 non-null float64\n",

" 6 Viscera weight 4177 non-null float64\n",

" 7 Shell weight 4177 non-null float64\n",

" 8 Rings 4177 non-null int64 \n",

" 9 Age 4177 non-null float64\n",

"dtypes: float64(8), int64(1), object(1)\n",

"memory usage: 326.5+ KB\n"

]

}

],

"source": [

"ds.info()"

]

},

{

"cell\_type": "code",

"execution\_count": 180,

"id": "4b69283d",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Sex 0\n",

"Length 0\n",

"Diameter 0\n",

"Height 0\n",

"Whole weight 0\n",

"Shucked weight 0\n",

"Viscera weight 0\n",

"Shell weight 0\n",

"Rings 0\n",

"Age 0\n",

"dtype: int64"

]

},

"execution\_count": 180,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"ds.isnull().sum()\n"

]

},

{

"cell\_type": "code",

"execution\_count": 100,

"id": "294f791d",

"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>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",

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

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</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",

" </tr>\n",

" <tr>\n",

" <th>4172</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4173</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4174</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4175</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4176</th>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" <td>True</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>4177 rows × 10 columns</p>\n",

"</div>"

],

"text/plain": [

" Sex Length Diameter Height Whole weight Shucked weight \\\n",

"0 True True True True True True \n",

"1 True True True True True True \n",

"2 True True True True True True \n",

"3 True True True True True True \n",

"4 True True True True True True \n",

"... ... ... ... ... ... ... \n",

"4172 True True True True True True \n",

"4173 True True True True True True \n",

"4174 True True True True True True \n",

"4175 True True True True True True \n",

"4176 True True True True True True \n",

"\n",

" Viscera weight Shell weight Rings Age \n",

"0 True True True True \n",

"1 True True True True \n",

"2 True True True True \n",

"3 True True True True \n",

"4 True True True True \n",

"... ... ... ... ... \n",

"4172 True True True True \n",

"4173 True True True True \n",

"4174 True True True True \n",

"4175 True True True True \n",

"4176 True True True True \n",

"\n",

"[4177 rows x 10 columns]"

]

},

"execution\_count": 100,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"ds.notnull()"

]

},

{

"cell\_type": "markdown",

"id": "0c242b31",

"metadata": {},

"source": [

"## 6. Find the outliers and replace them outliers"

]

},

{

"cell\_type": "code",

"execution\_count": 101,

"id": "69e9752f",

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/seaborn/\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.\n",

" warnings.warn(\n"

]

},

{

"data": {

"text/plain": [

"<AxesSubplot:xlabel='Diameter'>"

]

},

"execution\_count": 101,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"#occurence of outliers\n",

"#a data point in a data set that is distant from all other observations\n",

"\n",

"sns.boxplot(ds.Diameter)"

]

},

{

"cell\_type": "code",

"execution\_count": 102,

"id": "edf99479",

"metadata": {

"scrolled": true

},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/seaborn/\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.\n",

" warnings.warn(\n"

]

},

{

"data": {

"text/plain": [

"<AxesSubplot:xlabel='Diameter'>"

]

},

"execution\_count": 102,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"\n",

"Q1= ds.Diameter.quantile(0.25)\n",

"Q3=ds.Diameter.quantile(0.75)\n",

"\n",

"IQR=Q3-Q1 #spread the middle values are\n",

"\n",

"upper\_limit =Q3 + 1.5\*IQR\n",

"lower\_limit =Q1 - 1.5\*IQR\n",

"\n",

"ds['Diameter'] = np.where(ds['Diameter']>upper\_limit,30,ds['Diameter'])\n",

"\n",

"sns.boxplot(ds.Diameter)"

]

},

{

"cell\_type": "markdown",

"id": "0fd65d11",

"metadata": {},

"source": [

"## 7. Check for Categorical columns and perform encoding."

]

},

{

"cell\_type": "code",

"execution\_count": 183,

"id": "bc018e8c",

"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>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",

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

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>2</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",

" <td>16.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>2</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",

" <td>8.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>0</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",

" <td>10.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>2</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",

" <td>11.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>1</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",

" <td>8.5</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",

" </tr>\n",

" <tr>\n",

" <th>195</th>\n",

" <td>2</td>\n",

" <td>0.500</td>\n",

" <td>0.405</td>\n",

" <td>0.155</td>\n",

" <td>0.7720</td>\n",

" <td>0.3460</td>\n",

" <td>0.1535</td>\n",

" <td>0.245</td>\n",

" <td>12</td>\n",

" <td>13.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>196</th>\n",

" <td>0</td>\n",

" <td>0.505</td>\n",

" <td>0.410</td>\n",

" <td>0.150</td>\n",

" <td>0.6440</td>\n",

" <td>0.2850</td>\n",

" <td>0.1450</td>\n",

" <td>0.210</td>\n",

" <td>11</td>\n",

" <td>12.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>197</th>\n",

" <td>2</td>\n",

" <td>0.640</td>\n",

" <td>0.500</td>\n",

" <td>0.185</td>\n",

" <td>1.3035</td>\n",

" <td>0.4445</td>\n",

" <td>0.2635</td>\n",

" <td>0.465</td>\n",

" <td>16</td>\n",

" <td>17.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>198</th>\n",

" <td>2</td>\n",

" <td>0.560</td>\n",

" <td>0.450</td>\n",

" <td>0.160</td>\n",

" <td>0.9220</td>\n",

" <td>0.4320</td>\n",

" <td>0.1780</td>\n",

" <td>0.260</td>\n",

" <td>15</td>\n",

" <td>16.5</td>\n",

" </tr>\n",

" <tr>\n",

" <th>199</th>\n",

" <td>2</td>\n",

" <td>0.585</td>\n",

" <td>0.460</td>\n",

" <td>0.185</td>\n",

" <td>0.9220</td>\n",

" <td>0.3635</td>\n",

" <td>0.2130</td>\n",

" <td>0.285</td>\n",

" <td>10</td>\n",

" <td>11.5</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>200 rows × 10 columns</p>\n",

"</div>"

],

"text/plain": [

" Sex Length Diameter Height Whole weight Shucked weight \\\n",

"0 2 0.455 0.365 0.095 0.5140 0.2245 \n",

"1 2 0.350 0.265 0.090 0.2255 0.0995 \n",

"2 0 0.530 0.420 0.135 0.6770 0.2565 \n",

"3 2 0.440 0.365 0.125 0.5160 0.2155 \n",

"4 1 0.330 0.255 0.080 0.2050 0.0895 \n",

".. ... ... ... ... ... ... \n",

"195 2 0.500 0.405 0.155 0.7720 0.3460 \n",

"196 0 0.505 0.410 0.150 0.6440 0.2850 \n",

"197 2 0.640 0.500 0.185 1.3035 0.4445 \n",

"198 2 0.560 0.450 0.160 0.9220 0.4320 \n",

"199 2 0.585 0.460 0.185 0.9220 0.3635 \n",

"\n",

" Viscera weight Shell weight Rings Age \n",

"0 0.1010 0.150 15 16.5 \n",

"1 0.0485 0.070 7 8.5 \n",

"2 0.1415 0.210 9 10.5 \n",

"3 0.1140 0.155 10 11.5 \n",

"4 0.0395 0.055 7 8.5 \n",

".. ... ... ... ... \n",

"195 0.1535 0.245 12 13.5 \n",

"196 0.1450 0.210 11 12.5 \n",

"197 0.2635 0.465 16 17.5 \n",

"198 0.1780 0.260 15 16.5 \n",

"199 0.2130 0.285 10 11.5 \n",

"\n",

"[200 rows x 10 columns]"

]

},

"execution\_count": 183,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.preprocessing import LabelEncoder\n",

"le = LabelEncoder()\n",

"\n",

"ds1['Sex'] = le.fit\_transform(ds1['Sex'])\n",

"ds1\n",

"\n",

" # 0 = female, 1 = infant, 2 = male\n"

]

},

{

"cell\_type": "markdown",

"id": "b26ceb04",

"metadata": {},

"source": [

"## 8. Split the data into dependent and independent variables."

]

},

{

"cell\_type": "code",

"execution\_count": 184,

"id": "9ee35541",

"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>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>2</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>2</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>0</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>2</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>1</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",

" <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",

" </tr>\n",

" <tr>\n",

" <th>195</th>\n",

" <td>2</td>\n",

" <td>0.500</td>\n",

" <td>0.405</td>\n",

" <td>0.155</td>\n",

" <td>0.7720</td>\n",

" <td>0.3460</td>\n",

" <td>0.1535</td>\n",

" <td>0.245</td>\n",

" <td>12</td>\n",

" </tr>\n",

" <tr>\n",

" <th>196</th>\n",

" <td>0</td>\n",

" <td>0.505</td>\n",

" <td>0.410</td>\n",

" <td>0.150</td>\n",

" <td>0.6440</td>\n",

" <td>0.2850</td>\n",

" <td>0.1450</td>\n",

" <td>0.210</td>\n",

" <td>11</td>\n",

" </tr>\n",

" <tr>\n",

" <th>197</th>\n",

" <td>2</td>\n",

" <td>0.640</td>\n",

" <td>0.500</td>\n",

" <td>0.185</td>\n",

" <td>1.3035</td>\n",

" <td>0.4445</td>\n",

" <td>0.2635</td>\n",

" <td>0.465</td>\n",

" <td>16</td>\n",

" </tr>\n",

" <tr>\n",

" <th>198</th>\n",

" <td>2</td>\n",

" <td>0.560</td>\n",

" <td>0.450</td>\n",

" <td>0.160</td>\n",

" <td>0.9220</td>\n",

" <td>0.4320</td>\n",

" <td>0.1780</td>\n",

" <td>0.260</td>\n",

" <td>15</td>\n",

" </tr>\n",

" <tr>\n",

" <th>199</th>\n",

" <td>2</td>\n",

" <td>0.585</td>\n",

" <td>0.460</td>\n",

" <td>0.185</td>\n",

" <td>0.9220</td>\n",

" <td>0.3635</td>\n",

" <td>0.2130</td>\n",

" <td>0.285</td>\n",

" <td>10</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>200 rows × 9 columns</p>\n",

"</div>"

],

"text/plain": [

" Sex Length Diameter Height Whole weight Shucked weight \\\n",

"0 2 0.455 0.365 0.095 0.5140 0.2245 \n",

"1 2 0.350 0.265 0.090 0.2255 0.0995 \n",

"2 0 0.530 0.420 0.135 0.6770 0.2565 \n",

"3 2 0.440 0.365 0.125 0.5160 0.2155 \n",

"4 1 0.330 0.255 0.080 0.2050 0.0895 \n",

".. ... ... ... ... ... ... \n",

"195 2 0.500 0.405 0.155 0.7720 0.3460 \n",

"196 0 0.505 0.410 0.150 0.6440 0.2850 \n",

"197 2 0.640 0.500 0.185 1.3035 0.4445 \n",

"198 2 0.560 0.450 0.160 0.9220 0.4320 \n",

"199 2 0.585 0.460 0.185 0.9220 0.3635 \n",

"\n",

" Viscera weight Shell weight Rings \n",

"0 0.1010 0.150 15 \n",

"1 0.0485 0.070 7 \n",

"2 0.1415 0.210 9 \n",

"3 0.1140 0.155 10 \n",

"4 0.0395 0.055 7 \n",

".. ... ... ... \n",

"195 0.1535 0.245 12 \n",

"196 0.1450 0.210 11 \n",

"197 0.2635 0.465 16 \n",

"198 0.1780 0.260 15 \n",

"199 0.2130 0.285 10 \n",

"\n",

"[200 rows x 9 columns]"

]

},

"execution\_count": 184,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#Splitting the Dataset into the Independent Feature Matrix\n",

"\n",

"x = ds1.iloc[:, 0:9]\n",

"x"

]

},

{

"cell\_type": "code",

"execution\_count": 185,

"id": "46aa2d2c",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

" Age\n",

"0 16.5\n",

"1 8.5\n",

"2 10.5\n",

"3 11.5\n",

"4 8.5\n",

".. ...\n",

"195 13.5\n",

"196 12.5\n",

"197 17.5\n",

"198 16.5\n",

"199 11.5\n",

"\n",

"[200 rows x 1 columns]\n"

]

}

],

"source": [

"#Extracting the Dataset to Get the Dependent Vector\n",

"\n",

"y = ds1.iloc[:,9:10]\n",

"print(y)"

]

},

{

"cell\_type": "markdown",

"id": "1aa23453",

"metadata": {},

"source": [

"## 9. Scale the independent variables"

]

},

{

"cell\_type": "code",

"execution\_count": 114,

"id": "298cf459",

"metadata": {},

"outputs": [],

"source": [

"#scaling the independent variables using scale and MinMaxScaler\n",

"\n",

"from sklearn.preprocessing import scale\n",

"from sklearn.preprocessing import MinMaxScaler\n",

"\n",

"mm = MinMaxScaler()\n",

"\n",

"x\_scaled = mm.fit\_transform(x)\n",

"y\_scaled = mm.fit\_transform(y)"

]

},

{

"cell\_type": "code",

"execution\_count": 115,

"id": "a03029d1",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[1. , 0.51351351, 0.52808989, ..., 0.17680075, 0.14070352,\n",

" 0.64705882],\n",

" [1. , 0.32432432, 0.30337079, ..., 0.07857811, 0.06030151,\n",

" 0.17647059],\n",

" [0. , 0.64864865, 0.65168539, ..., 0.2525725 , 0.20100503,\n",

" 0.29411765],\n",

" ...,\n",

" [1. , 0.84684685, 0.83146067, ..., 0.4808232 , 0.45728643,\n",

" 0.70588235],\n",

" [1. , 0.7027027 , 0.71910112, ..., 0.32086062, 0.25125628,\n",

" 0.64705882],\n",

" [1. , 0.74774775, 0.74157303, ..., 0.38634238, 0.27638191,\n",

" 0.35294118]])"

]

},

"execution\_count": 115,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_scaled"

]

},

{

"cell\_type": "code",

"execution\_count": 116,

"id": "b288b66e",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[0.64705882],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.94117647],\n",

" [0.70588235],\n",

" [0.29411765],\n",

" [0.88235294],\n",

" [0.58823529],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.17647059],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.41176471],\n",

" [0.47058824],\n",

" [0.64705882],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.64705882],\n",

" [0.82352941],\n",

" [0.88235294],\n",

" [0.52941176],\n",

" [0.23529412],\n",

" [0.70588235],\n",

" [0.23529412],\n",

" [0.41176471],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.58823529],\n",

" [0.05882353],\n",

" [0.05882353],\n",

" [0. ],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.17647059],\n",

" [0.11764706],\n",

" [0.29411765],\n",

" [0.23529412],\n",

" [0.17647059],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.23529412],\n",

" [0.23529412],\n",

" [0. ],\n",

" [0.17647059],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.23529412],\n",

" [0.47058824],\n",

" [0.52941176],\n",

" [0.35294118],\n",

" [0.11764706],\n",

" [0.52941176],\n",

" [0.23529412],\n",

" [0.94117647],\n",

" [0.41176471],\n",

" [0.52941176],\n",

" [0.64705882],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.58823529],\n",

" [0.29411765],\n",

" [0.47058824],\n",

" [0.70588235],\n",

" [1. ],\n",

" [0.58823529],\n",

" [0.47058824],\n",

" [0.52941176],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.47058824],\n",

" [0.64705882],\n",

" [0.47058824],\n",

" [0.52941176],\n",

" [0.35294118],\n",

" [0.64705882],\n",

" [0.58823529],\n",

" [0.29411765],\n",

" [0.23529412],\n",

" [0.17647059],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.64705882],\n",

" [0.64705882],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.47058824],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.41176471],\n",

" [0.41176471],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.23529412],\n",

" [0.29411765],\n",

" [0.17647059],\n",

" [0.58823529],\n",

" [0.11764706],\n",

" [0.11764706],\n",

" [0.05882353],\n",

" [0.11764706],\n",

" [0.23529412],\n",

" [0.88235294],\n",

" [0.82352941],\n",

" [0.76470588],\n",

" [0.29411765],\n",

" [0.17647059],\n",

" [0.17647059],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.70588235],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.05882353],\n",

" [0. ],\n",

" [0.64705882],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.35294118],\n",

" [0.52941176],\n",

" [0.70588235],\n",

" [0.52941176],\n",

" [0.52941176],\n",

" [0.52941176],\n",

" [0.52941176],\n",

" [0.47058824],\n",

" [0.82352941],\n",

" [0.70588235],\n",

" [0.58823529],\n",

" [0.94117647],\n",

" [0.94117647],\n",

" [0.58823529],\n",

" [0.47058824],\n",

" [0.58823529],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.23529412],\n",

" [0.05882353],\n",

" [0.17647059],\n",

" [0.05882353],\n",

" [0.23529412],\n",

" [0. ],\n",

" [0.41176471],\n",

" [0.58823529],\n",

" [1. ],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.52941176],\n",

" [0.47058824],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.29411765],\n",

" [0.52941176],\n",

" [0.47058824],\n",

" [0.58823529],\n",

" [0.23529412],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.41176471],\n",

" [0.70588235],\n",

" [0.64705882],\n",

" [0.35294118]])"

]

},

"execution\_count": 116,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y\_scaled"

]

},

{

"cell\_type": "markdown",

"id": "e02aff6b",

"metadata": {},

"source": [

"## 10. Split the data into training and testing"

]

},

{

"cell\_type": "code",

"execution\_count": 187,

"id": "8c64834f",

"metadata": {},

"outputs": [],

"source": [

"\n",

"from sklearn.model\_selection import train\_test\_split # library for split the data into training and testing\n",

"\n",

"x\_train,x\_test,y\_train,y\_test = train\_test\_split(x\_scaled,y\_scaled,train\_size=0.80,test\_size = 0.20,random\_state=0)"

]

},

{

"cell\_type": "code",

"execution\_count": 188,

"id": "481ad85f",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[0.5 , 0.17117117, 0.15730337, ..., 0.0261927 , 0.01809045,\n",

" 0.17647059],\n",

" [0. , 0.71171171, 0.69662921, ..., 0.34985968, 0.31155779,\n",

" 0.47058824],\n",

" [0. , 0.73873874, 0.71910112, ..., 0.49672591, 0.27638191,\n",

" 0.41176471],\n",

" ...,\n",

" [1. , 0.48648649, 0.47191011, ..., 0.16651076, 0.15577889,\n",

" 0.35294118],\n",

" [0. , 0.52252252, 0.5505618 , ..., 0.19363891, 0.14070352,\n",

" 0.17647059],\n",

" [1. , 0.63963964, 0.68539326, ..., 0.42376052, 0.27638191,\n",

" 0.23529412]])"

]

},

"execution\_count": 188,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_train"

]

},

{

"cell\_type": "code",

"execution\_count": 189,

"id": "6f387b8b",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[0.17647059],\n",

" [0.47058824],\n",

" [0.41176471],\n",

" [0.29411765],\n",

" [0.58823529],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.64705882],\n",

" [0.29411765],\n",

" [0.41176471],\n",

" [0.23529412],\n",

" [0.11764706],\n",

" [0.47058824],\n",

" [0.23529412],\n",

" [0. ],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.52941176],\n",

" [0.29411765],\n",

" [0.23529412],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [1. ],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.52941176],\n",

" [0.17647059],\n",

" [0.82352941],\n",

" [0.17647059],\n",

" [0.52941176],\n",

" [0.29411765],\n",

" [0.64705882],\n",

" [0.29411765],\n",

" [0.64705882],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.47058824],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.29411765],\n",

" [0.47058824],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.17647059],\n",

" [0.17647059],\n",

" [0.70588235],\n",

" [0.05882353],\n",

" [0.58823529],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.41176471],\n",

" [0. ],\n",

" [0.17647059],\n",

" [0.11764706],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.52941176],\n",

" [0.47058824],\n",

" [0.23529412],\n",

" [0.64705882],\n",

" [0.64705882],\n",

" [0.29411765],\n",

" [0.58823529],\n",

" [0.23529412],\n",

" [0.94117647],\n",

" [0.58823529],\n",

" [0.11764706],\n",

" [0.29411765],\n",

" [0.11764706],\n",

" [0.47058824],\n",

" [0.35294118],\n",

" [0.52941176],\n",

" [0.29411765],\n",

" [0.47058824],\n",

" [0.23529412],\n",

" [0.41176471],\n",

" [0.47058824],\n",

" [0.41176471],\n",

" [0.47058824],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.70588235],\n",

" [0.64705882],\n",

" [0.94117647],\n",

" [0.35294118],\n",

" [0.58823529],\n",

" [0.17647059],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.52941176],\n",

" [0.47058824],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.23529412],\n",

" [0.64705882],\n",

" [0.23529412],\n",

" [0.23529412],\n",

" [0.23529412],\n",

" [0.17647059],\n",

" [0.29411765],\n",

" [0.47058824],\n",

" [0.05882353],\n",

" [0.47058824],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.17647059],\n",

" [0.35294118],\n",

" [0.70588235],\n",

" [0.88235294],\n",

" [0.52941176],\n",

" [0.64705882],\n",

" [0.41176471],\n",

" [0.29411765],\n",

" [0.64705882],\n",

" [0.94117647],\n",

" [0.23529412],\n",

" [0.05882353],\n",

" [0.82352941],\n",

" [0.70588235],\n",

" [0.47058824],\n",

" [0.29411765],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.70588235],\n",

" [0.58823529],\n",

" [0.41176471],\n",

" [0.05882353],\n",

" [0.23529412],\n",

" [0.94117647],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.58823529],\n",

" [0.47058824],\n",

" [0.41176471],\n",

" [0.05882353],\n",

" [0.52941176],\n",

" [0.29411765],\n",

" [0. ],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.52941176],\n",

" [0.35294118],\n",

" [0.70588235],\n",

" [0.35294118],\n",

" [0.88235294],\n",

" [0.35294118],\n",

" [0.52941176],\n",

" [0.58823529],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.23529412]])"

]

},

"execution\_count": 189,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y\_train"

]

},

{

"cell\_type": "code",

"execution\_count": 190,

"id": "4ce070e6",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[1. , 0.35135135, 0.37078652, 0.21052632, 0.08948413,\n",

" 0.08160377, 0.06828812, 0.09045226, 0.17647059],\n",

" [1. , 0.94594595, 0.94382022, 0.92105263, 0.76448413,\n",

" 0.66226415, 1. , 0.58291457, 0.58823529],\n",

" [0. , 0.59459459, 0.60674157, 0.44736842, 0.25297619,\n",

" 0.23632075, 0.23386342, 0.21105528, 0.35294118],\n",

" [1. , 0.54054054, 0.53932584, 0.47368421, 0.19543651,\n",

" 0.17971698, 0.23666978, 0.15577889, 0.17647059],\n",

" [0.5 , 0.26126126, 0.25842697, 0.23684211, 0.04503968,\n",

" 0.04009434, 0.0767072 , 0.04020101, 0.23529412],\n",

" [0. , 0.7027027 , 0.71910112, 0.63157895, 0.39424603,\n",

" 0.39481132, 0.48924228, 0.29145729, 0.35294118],\n",

" [0.5 , 0.45945946, 0.38202247, 0.28947368, 0.12757937,\n",

" 0.12311321, 0.13283442, 0.11055276, 0.23529412],\n",

" [1. , 0.52252252, 0.49438202, 0.42105263, 0.19246032,\n",

" 0.20141509, 0.1898971 , 0.14723618, 0.35294118],\n",

" [1. , 0.57657658, 0.56179775, 0.5 , 0.20297619,\n",

" 0.19528302, 0.1655753 , 0.18090452, 0.41176471],\n",

" [0. , 0.83783784, 0.86516854, 0.78947368, 0.53234127,\n",

" 0.46792453, 0.55846586, 0.44221106, 0.35294118],\n",

" [1. , 0.6036036 , 0.61797753, 0.36842105, 0.23611111,\n",

" 0.27783019, 0.28718428, 0.16582915, 0.29411765],\n",

" [0.5 , 0.18018018, 0.14606742, 0.10526316, 0.01706349,\n",

" 0.01698113, 0.03180543, 0.0201005 , 0.05882353],\n",

" [1. , 0.72072072, 0.78651685, 0.73684211, 0.3609127 ,\n",

" 0.36650943, 0.36202058, 0.28643216, 0.58823529],\n",

" [0. , 0.71171171, 0.71910112, 0.5 , 0.38035714,\n",

" 0.35518868, 0.26753976, 0.30150754, 0.47058824],\n",

" [0. , 0.72972973, 0.70786517, 0.52631579, 0.36150794,\n",

" 0.35283019, 0.45930776, 0.27638191, 0.29411765],\n",

" [0. , 0.67567568, 0.66292135, 0.44736842, 0.29285714,\n",

" 0.26745283, 0.26753976, 0.25125628, 0.70588235],\n",

" [0. , 0.91891892, 0.94382022, 0.71052632, 0.7015873 ,\n",

" 0.75896226, 0.72217025, 0.44723618, 0.88235294],\n",

" [1. , 0.76576577, 0.78651685, 0.65789474, 0.48888889,\n",

" 0.44622642, 0.51730589, 0.40201005, 0.76470588],\n",

" [0. , 0.5045045 , 0.50561798, 0.34210526, 0.19543651,\n",

" 0.21367925, 0.20579981, 0.13567839, 0.23529412],\n",

" [0. , 0.78378378, 0.71910112, 0.81578947, 0.42380952,\n",

" 0.44386792, 0.52946679, 0.30653266, 0.52941176],\n",

" [0. , 0.81081081, 0.7752809 , 0.71052632, 0.39146825,\n",

" 0.4009434 , 0.38821328, 0.31658291, 0.35294118],\n",

" [0. , 0.57657658, 0.56179775, 0.44736842, 0.20595238,\n",

" 0.22122642, 0.18896165, 0.16482412, 0.35294118],\n",

" [0.5 , 0.3963964 , 0.37078652, 0.28947368, 0.06865079,\n",

" 0.07264151, 0.07202993, 0.06532663, 0.17647059],\n",

" [0. , 0.72972973, 0.74157303, 0.65789474, 0.43412698,\n",

" 0.27169811, 0.32179607, 0.4321608 , 0.52941176],\n",

" [1. , 0.5045045 , 0.48314607, 0.34210526, 0.15138889,\n",

" 0.15990566, 0.19831618, 0.12562814, 0.17647059],\n",

" [1. , 0.36036036, 0.30337079, 0.18421053, 0.07301587,\n",

" 0.075 , 0.08325538, 0.06030151, 0.11764706],\n",

" [1. , 0.73873874, 0.7752809 , 0.57894737, 0.37301587,\n",

" 0.35330189, 0.39289055, 0.34170854, 0.41176471],\n",

" [1. , 0.81081081, 0.80898876, 0.78947368, 0.47142857,\n",

" 0.50471698, 0.54256314, 0.34673367, 0.52941176],\n",

" [0. , 0.62162162, 0.66292135, 0.52631579, 0.29206349,\n",

" 0.27688679, 0.31057063, 0.24623116, 0.58823529],\n",

" [0.5 , 0.07207207, 0.04494382, 0.05263158, 0.0047619 ,\n",

" 0.00660377, 0.01122544, 0.00502513, 0. ],\n",

" [0.5 , 0.33333333, 0.33707865, 0.23684211, 0.10337302,\n",

" 0.07971698, 0.06173994, 0.10552764, 0.17647059],\n",

" [0. , 0.59459459, 0.60674157, 0.52631579, 0.25059524,\n",

" 0.23207547, 0.31618335, 0.21105528, 0.23529412],\n",

" [1. , 0.75675676, 0.7752809 , 0.55263158, 0.40595238,\n",

" 0.40660377, 0.47801684, 0.31658291, 0.64705882],\n",

" [1. , 0.53153153, 0.51685393, 0.34210526, 0.15912698,\n",

" 0.15235849, 0.18802619, 0.16582915, 0.29411765],\n",

" [0. , 0.71171171, 0.69662921, 0.60526316, 0.3609127 ,\n",

" 0.39339623, 0.38821328, 0.26130653, 0.47058824],\n",

" [0. , 0.74774775, 0.74157303, 0.68421053, 0.35813492,\n",

" 0.33443396, 0.494855 , 0.28140704, 0.29411765],\n",

" [1. , 0.97297297, 0.92134831, 0.65789474, 0.76547619,\n",

" 0.71320755, 0.47614593, 0.77386935, 0.82352941],\n",

" [0.5 , 0.28828829, 0.28089888, 0.21052632, 0.06944444,\n",

" 0.0745283 , 0.06173994, 0.04522613, 0.17647059],\n",

" [1. , 0.76576577, 0.7752809 , 0.63157895, 0.5109127 ,\n",

" 0.375 , 0.42563143, 0.57286432, 1. ],\n",

" [0. , 0.67567568, 0.6741573 , 0.65789474, 0.30634921,\n",

" 0.26698113, 0.33021515, 0.27135678, 0.41176471]])"

]

},

"execution\_count": 190,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_test"

]

},

{

"cell\_type": "code",

"execution\_count": 191,

"id": "cab53c98",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[0.17647059],\n",

" [0.58823529],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.35294118],\n",

" [0.23529412],\n",

" [0.35294118],\n",

" [0.41176471],\n",

" [0.35294118],\n",

" [0.29411765],\n",

" [0.05882353],\n",

" [0.58823529],\n",

" [0.47058824],\n",

" [0.29411765],\n",

" [0.70588235],\n",

" [0.88235294],\n",

" [0.76470588],\n",

" [0.23529412],\n",

" [0.52941176],\n",

" [0.35294118],\n",

" [0.35294118],\n",

" [0.17647059],\n",

" [0.52941176],\n",

" [0.17647059],\n",

" [0.11764706],\n",

" [0.41176471],\n",

" [0.52941176],\n",

" [0.58823529],\n",

" [0. ],\n",

" [0.17647059],\n",

" [0.23529412],\n",

" [0.64705882],\n",

" [0.29411765],\n",

" [0.47058824],\n",

" [0.29411765],\n",

" [0.82352941],\n",

" [0.17647059],\n",

" [1. ],\n",

" [0.41176471]])"

]

},

"execution\_count": 191,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y\_test"

]

},

{

"cell\_type": "code",

"execution\_count": 129,

"id": "d227ece0",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"(200, 9)\n",

"(200, 1)\n",

"(160, 9)\n",

"(160, 1)\n",

"(40, 9)\n",

"(40, 1)\n"

]

}

],

"source": [

"print(x\_scaled.shape)\n",

"print(y\_scaled.shape)\n",

"print(x\_train.shape)\n",

"print(y\_train.shape)\n",

"print(x\_test.shape)\n",

"print(y\_test.shape)"

]

},

{

"cell\_type": "markdown",

"id": "be3d3797",

"metadata": {},

"source": [

"## 11. Build the Model"

]

},

{

"cell\_type": "code",

"execution\_count": 135,

"id": "0c02dd4d",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"LinearRegression()"

]

},

"execution\_count": 135,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.linear\_model import LinearRegression\n",

"\n",

"mlr = LinearRegression()\n",

"\n",

"mlr.fit(x\_train,y\_train)"

]

},

{

"cell\_type": "markdown",

"id": "17570045",

"metadata": {},

"source": [

"## 12. Train the Model\n",

"## 13. Test the Model"

]

},

{

"cell\_type": "code",

"execution\_count": 136,

"id": "8ee26275",

"metadata": {},

"outputs": [],

"source": [

"prediction = mlr.predict(x\_test) "

]

},

{

"cell\_type": "code",

"execution\_count": 138,

"id": "fbb8516e",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[1.76470588e-01],\n",

" [5.88235294e-01],\n",

" [3.52941176e-01],\n",

" [1.76470588e-01],\n",

" [2.35294118e-01],\n",

" [3.52941176e-01],\n",

" [2.35294118e-01],\n",

" [3.52941176e-01],\n",

" [4.11764706e-01],\n",

" [3.52941176e-01],\n",

" [2.94117647e-01],\n",

" [5.88235294e-02],\n",

" [5.88235294e-01],\n",

" [4.70588235e-01],\n",

" [2.94117647e-01],\n",

" [7.05882353e-01],\n",

" [8.82352941e-01],\n",

" [7.64705882e-01],\n",

" [2.35294118e-01],\n",

" [5.29411765e-01],\n",

" [3.52941176e-01],\n",

" [3.52941176e-01],\n",

" [1.76470588e-01],\n",

" [5.29411765e-01],\n",

" [1.76470588e-01],\n",

" [1.17647059e-01],\n",

" [4.11764706e-01],\n",

" [5.29411765e-01],\n",

" [5.88235294e-01],\n",

" [2.20691474e-16],\n",

" [1.76470588e-01],\n",

" [2.35294118e-01],\n",

" [6.47058824e-01],\n",

" [2.94117647e-01],\n",

" [4.70588235e-01],\n",

" [2.94117647e-01],\n",

" [8.23529412e-01],\n",

" [1.76470588e-01],\n",

" [1.00000000e+00],\n",

" [4.11764706e-01]])"

]

},

"execution\_count": 138,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"prediction"

]

},

{

"cell\_type": "code",

"execution\_count": 141,

"id": "f3033424",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [1],\n",

" [0]])"

]

},

"execution\_count": 141,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"prediction.astype(int)"

]

},

{

"cell\_type": "code",

"execution\_count": 142,

"id": "2fb38996",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [0],\n",

" [1],\n",

" [0]])"

]

},

"execution\_count": 142,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y\_test.astype(int)"

]

},

{

"cell\_type": "markdown",

"id": "ae92ba9d",

"metadata": {},

"source": [

"## 14. Measure the performance using Metrics."

]

},

{

"cell\_type": "code",

"execution\_count": 143,

"id": "4787dacc",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"1.0"

]

},

"execution\_count": 143,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.metrics import r2\_score\n",

"r2\_score(prediction,y\_test)"

]

},

{

"cell\_type": "code",

"execution\_count": 153,

"id": "6e06804c",

"metadata": {},

"outputs": [],

"source": [

"from sklearn.preprocessing import PolynomialFeatures\n",

"plr = PolynomialFeatures(degree=2)\n",

"x\_poly = plr.fit\_transform(x)"

]

},

{

"cell\_type": "code",

"execution\_count": 154,

"id": "032109c0",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[1.00000e+00, 2.00000e+00, 4.55000e-01, ..., 2.25000e-02,\n",

" 2.25000e+00, 2.25000e+02],\n",

" [1.00000e+00, 2.00000e+00, 3.50000e-01, ..., 4.90000e-03,\n",

" 4.90000e-01, 4.90000e+01],\n",

" [1.00000e+00, 0.00000e+00, 5.30000e-01, ..., 4.41000e-02,\n",

" 1.89000e+00, 8.10000e+01],\n",

" ...,\n",

" [1.00000e+00, 2.00000e+00, 6.40000e-01, ..., 2.16225e-01,\n",

" 7.44000e+00, 2.56000e+02],\n",

" [1.00000e+00, 2.00000e+00, 5.60000e-01, ..., 6.76000e-02,\n",

" 3.90000e+00, 2.25000e+02],\n",

" [1.00000e+00, 2.00000e+00, 5.85000e-01, ..., 8.12250e-02,\n",

" 2.85000e+00, 1.00000e+02]])"

]

},

"execution\_count": 154,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_poly"

]

},

{

"cell\_type": "markdown",

"id": "9c3457f1",

"metadata": {},

"source": [

"## Abalone Age Prediction\n",

"## 1. LinearRegression"

]

},

{

"cell\_type": "code",

"execution\_count": 155,

"id": "473b7680",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"LinearRegression()"

]

},

"execution\_count": 155,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.linear\_model import LinearRegression\n",

"lr = LinearRegression()\n",

"lr.fit(x\_poly,y)"

]

},

{

"cell\_type": "code",

"execution\_count": 163,

"id": "c79c199a",

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning: X does not have valid feature names, but PolynomialFeatures was fitted with feature names\n",

" warnings.warn(\n"

]

},

{

"data": {

"text/plain": [

"array([[17.5]])"

]

},

"execution\_count": 163,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"lr.predict(plr.transform([[1,0.350,0.410,0.185,1.3035,0.3635,0.1010,0.285,16]]))"

]

},

{

"cell\_type": "markdown",

"id": "dfd89f4b",

"metadata": {},

"source": [

"## 2. Ridge"

]

},

{

"cell\_type": "code",

"execution\_count": 159,

"id": "30a0c7c9",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Ridge()"

]

},

"execution\_count": 159,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.linear\_model import Ridge\n",

"r = Ridge()\n",

"r.fit(x,y)"

]

},

{

"cell\_type": "code",

"execution\_count": 164,

"id": "8bf58994",

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning: X does not have valid feature names, but Ridge was fitted with feature names\n",

" warnings.warn(\n"

]

},

{

"data": {

"text/plain": [

"array([[17.49624459]])"

]

},

"execution\_count": 164,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"r.predict([[1,0.350,0.410,0.185,1.3035,0.3635,0.1010,0.285,16]])"

]

},

{

"cell\_type": "markdown",

"id": "70a32095",

"metadata": {},

"source": [

"## 3. Lasso"

]

},

{

"cell\_type": "code",

"execution\_count": 161,

"id": "23fa5dc9",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Lasso()"

]

},

"execution\_count": 161,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.linear\_model import Lasso\n",

"l = Lasso()\n",

"l.fit(x,y)"

]

},

{

"cell\_type": "code",

"execution\_count": 165,

"id": "58b13edc",

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning: X does not have valid feature names, but Lasso was fitted with feature names\n",

" warnings.warn(\n"

]

},

{

"data": {

"text/plain": [

"array([17.08721342])"

]

},

"execution\_count": 165,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"l.predict([[1,0.350,0.410,0.185,1.3035,0.3635,0.1010,0.285,16]])"

]

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python 3 (ipykernel)",

"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": "3.9.12"

}

},

"nbformat": 4,

"nbformat\_minor": 5

}