{

"cells": [

{

"cell\_type": "markdown",

"id": "0d1447b7",

"metadata": {},

"source": [

"# PRODUCT DEMAND PREDICTION - ADS\_PHASE\_5\n",

"\n",

"# TEAM NUMBER : 01\n",

"\n",

"# (Team Member)\n",

"\n",

"\n",

"## Problem Statement : Loading and Preprocessing \n",

"\n",

"In this part we will begin building our project by loading and preprocessing the dataset. \n",

"\n",

"We have begin building the product demand prediction model by loading and preprocessing the dataset. \n"

]

},

{

"cell\_type": "code",

"execution\_count": 28,

"id": "cf8cfba8",

"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>ID</th>\n",

" <th>Store ID</th>\n",

" <th>Total Price</th>\n",

" <th>Base Price</th>\n",

" <th>Units Sold</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>99.0375</td>\n",

" <td>111.8625</td>\n",

" <td>20</td>\n",

" </tr>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>99.0375</td>\n",

" <td>99.0375</td>\n",

" <td>28</td>\n",

" </tr>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>133.9500</td>\n",

" <td>133.9500</td>\n",

" <td>19</td>\n",

" </tr>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>133.9500</td>\n",

" <td>133.9500</td>\n",

" <td>44</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>5</td>\n",

" <td>8091</td>\n",

" <td>141.0750</td>\n",

" <td>141.0750</td>\n",

" <td>52</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150145</th>\n",

" <td>212638</td>\n",

" <td>9984</td>\n",

" <td>235.8375</td>\n",

" <td>235.8375</td>\n",

" <td>38</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150146</th>\n",

" <td>212639</td>\n",

" <td>9984</td>\n",

" <td>235.8375</td>\n",

" <td>235.8375</td>\n",

" <td>30</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150147</th>\n",

" <td>212642</td>\n",

" <td>9984</td>\n",

" <td>357.6750</td>\n",

" <td>483.7875</td>\n",

" <td>31</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150148</th>\n",

" <td>212643</td>\n",

" <td>9984</td>\n",

" <td>141.7875</td>\n",

" <td>191.6625</td>\n",

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

" </tr>\n",

" <tr>\n",

" <th>150149</th>\n",

" <td>212644</td>\n",

" <td>9984</td>\n",

" <td>234.4125</td>\n",

" <td>234.4125</td>\n",

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

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>150150 rows × 5 columns</p>\n",

"</div>"

],

"text/plain": [

" ID Store ID Total Price Base Price Units Sold\n",

"0 1 8091 99.0375 111.8625 20\n",

"1 2 8091 99.0375 99.0375 28\n",

"2 3 8091 133.9500 133.9500 19\n",

"3 4 8091 133.9500 133.9500 44\n",

"4 5 8091 141.0750 141.0750 52\n",

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

"150145 212638 9984 235.8375 235.8375 38\n",

"150146 212639 9984 235.8375 235.8375 30\n",

"150147 212642 9984 357.6750 483.7875 31\n",

"150148 212643 9984 141.7875 191.6625 12\n",

"150149 212644 9984 234.4125 234.4125 15\n",

"\n",

"[150150 rows x 5 columns]"

]

},

"execution\_count": 28,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#importing necessary libraries\n",

"import pandas as pd\n",

"from sklearn.preprocessing import StandardScaler, LabelEncoder\n",

"from sklearn.impute import SimpleImputer\n",

"from sklearn.model\_selection import train\_test\_split\n",

"import warnings\n",

"warnings.simplefilter(action='ignore', category=FutureWarning)\n",

"\n",

"#importing the netflix dataset\n",

"file\_path = r\"C:\\Users\\sanja\\Downloads\\K2\\archive\\PoductDemand.csv\" \n",

"encoding = \"ISO-8859-1\"\n",

"df = pd.read\_csv(file\_path, encoding=encoding)\n",

"df"

]

},

{

"cell\_type": "code",

"execution\_count": 29,

"id": "07c68713",

"metadata": {

"scrolled": true

},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

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

"RangeIndex: 150150 entries, 0 to 150149\n",

"Data columns (total 5 columns):\n",

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

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

" 0 ID 150150 non-null int64 \n",

" 1 Store ID 150150 non-null int64 \n",

" 2 Total Price 150149 non-null float64\n",

" 3 Base Price 150150 non-null float64\n",

" 4 Units Sold 150150 non-null int64 \n",

"dtypes: float64(2), int64(3)\n",

"memory usage: 5.7 MB\n"

]

}

],

"source": [

"df.info()"

]

},

{

"cell\_type": "code",

"execution\_count": 30,

"id": "1121c0e4",

"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>ID</th>\n",

" <th>Store ID</th>\n",

" <th>Total Price</th>\n",

" <th>Base Price</th>\n",

" <th>Units Sold</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>99.0375</td>\n",

" <td>111.8625</td>\n",

" <td>20</td>\n",

" </tr>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>99.0375</td>\n",

" <td>99.0375</td>\n",

" <td>28</td>\n",

" </tr>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>133.9500</td>\n",

" <td>133.9500</td>\n",

" <td>19</td>\n",

" </tr>\n",

" <tr>\n",

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

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

" <td>8091</td>\n",

" <td>133.9500</td>\n",

" <td>133.9500</td>\n",

" <td>44</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>5</td>\n",

" <td>8091</td>\n",

" <td>141.0750</td>\n",

" <td>141.0750</td>\n",

" <td>52</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" ID Store ID Total Price Base Price Units Sold\n",

"0 1 8091 99.0375 111.8625 20\n",

"1 2 8091 99.0375 99.0375 28\n",

"2 3 8091 133.9500 133.9500 19\n",

"3 4 8091 133.9500 133.9500 44\n",

"4 5 8091 141.0750 141.0750 52"

]

},

"execution\_count": 30,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"df.head()"

]

},

{

"cell\_type": "code",

"execution\_count": 31,

"id": "554d9406",

"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>ID</th>\n",

" <th>Store ID</th>\n",

" <th>Total Price</th>\n",

" <th>Base Price</th>\n",

" <th>Units Sold</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150145</th>\n",

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150146</th>\n",

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150147</th>\n",

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150148</th>\n",

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150149</th>\n",

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

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

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

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

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

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>150150 rows × 5 columns</p>\n",

"</div>"

],

"text/plain": [

" ID Store ID Total Price Base Price Units Sold\n",

"0 False False False False False\n",

"1 False False False False False\n",

"2 False False False False False\n",

"3 False False False False False\n",

"4 False False False False False\n",

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

"150145 False False False False False\n",

"150146 False False False False False\n",

"150147 False False False False False\n",

"150148 False False False False False\n",

"150149 False False False False False\n",

"\n",

"[150150 rows x 5 columns]"

]

},

"execution\_count": 31,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#to display null values\n",

"df.isnull()"

]

},

{

"cell\_type": "code",

"execution\_count": 32,

"id": "58fbe257",

"metadata": {},

"outputs": [],

"source": [

"#handling null values\n",

"\n",

"df.fillna(df.mean(), inplace=True) \n",

"df.dropna(inplace=True)"

]

},

{

"cell\_type": "code",

"execution\_count": 33,

"id": "a124cce7",

"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>ID</th>\n",

" <th>Store ID</th>\n",

" <th>Total Price</th>\n",

" <th>Base Price</th>\n",

" <th>Units Sold</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

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

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

" <td>99.0375</td>\n",

" <td>111.8625</td>\n",

" <td>20</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>99.0375</td>\n",

" <td>99.0375</td>\n",

" <td>28</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>133.9500</td>\n",

" <td>133.9500</td>\n",

" <td>19</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>133.9500</td>\n",

" <td>133.9500</td>\n",

" <td>44</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>5</td>\n",

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

" <td>141.0750</td>\n",

" <td>141.0750</td>\n",

" <td>52</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150145</th>\n",

" <td>212638</td>\n",

" <td>75</td>\n",

" <td>235.8375</td>\n",

" <td>235.8375</td>\n",

" <td>38</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150146</th>\n",

" <td>212639</td>\n",

" <td>75</td>\n",

" <td>235.8375</td>\n",

" <td>235.8375</td>\n",

" <td>30</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150147</th>\n",

" <td>212642</td>\n",

" <td>75</td>\n",

" <td>357.6750</td>\n",

" <td>483.7875</td>\n",

" <td>31</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150148</th>\n",

" <td>212643</td>\n",

" <td>75</td>\n",

" <td>141.7875</td>\n",

" <td>191.6625</td>\n",

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

" </tr>\n",

" <tr>\n",

" <th>150149</th>\n",

" <td>212644</td>\n",

" <td>75</td>\n",

" <td>234.4125</td>\n",

" <td>234.4125</td>\n",

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

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>150150 rows × 5 columns</p>\n",

"</div>"

],

"text/plain": [

" ID Store ID Total Price Base Price Units Sold\n",

"0 1 3 99.0375 111.8625 20\n",

"1 2 3 99.0375 99.0375 28\n",

"2 3 3 133.9500 133.9500 19\n",

"3 4 3 133.9500 133.9500 44\n",

"4 5 3 141.0750 141.0750 52\n",

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

"150145 212638 75 235.8375 235.8375 38\n",

"150146 212639 75 235.8375 235.8375 30\n",

"150147 212642 75 357.6750 483.7875 31\n",

"150148 212643 75 141.7875 191.6625 12\n",

"150149 212644 75 234.4125 234.4125 15\n",

"\n",

"[150150 rows x 5 columns]"

]

},

"execution\_count": 33,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"label\_encoder = LabelEncoder()\n",

"df['Store ID'] = label\_encoder.fit\_transform(df['Store ID'])\n",

"df\n"

]

},

{

"cell\_type": "code",

"execution\_count": 34,

"id": "e779f3c8",

"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>ID</th>\n",

" <th>Store ID</th>\n",

" <th>Total Price</th>\n",

" <th>Base Price</th>\n",

" <th>Units Sold</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

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

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

" <td>-1.041443</td>\n",

" <td>111.8625</td>\n",

" <td>20</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>-1.041443</td>\n",

" <td>99.0375</td>\n",

" <td>28</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>-0.703497</td>\n",

" <td>133.9500</td>\n",

" <td>19</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>-0.703497</td>\n",

" <td>133.9500</td>\n",

" <td>44</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>5</td>\n",

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

" <td>-0.634528</td>\n",

" <td>141.0750</td>\n",

" <td>52</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150145</th>\n",

" <td>212638</td>\n",

" <td>75</td>\n",

" <td>0.282754</td>\n",

" <td>235.8375</td>\n",

" <td>38</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150146</th>\n",

" <td>212639</td>\n",

" <td>75</td>\n",

" <td>0.282754</td>\n",

" <td>235.8375</td>\n",

" <td>30</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150147</th>\n",

" <td>212642</td>\n",

" <td>75</td>\n",

" <td>1.462118</td>\n",

" <td>483.7875</td>\n",

" <td>31</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150148</th>\n",

" <td>212643</td>\n",

" <td>75</td>\n",

" <td>-0.627632</td>\n",

" <td>191.6625</td>\n",

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

" </tr>\n",

" <tr>\n",

" <th>150149</th>\n",

" <td>212644</td>\n",

" <td>75</td>\n",

" <td>0.268961</td>\n",

" <td>234.4125</td>\n",

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

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>150150 rows × 5 columns</p>\n",

"</div>"

],

"text/plain": [

" ID Store ID Total Price Base Price Units Sold\n",

"0 1 3 -1.041443 111.8625 20\n",

"1 2 3 -1.041443 99.0375 28\n",

"2 3 3 -0.703497 133.9500 19\n",

"3 4 3 -0.703497 133.9500 44\n",

"4 5 3 -0.634528 141.0750 52\n",

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

"150145 212638 75 0.282754 235.8375 38\n",

"150146 212639 75 0.282754 235.8375 30\n",

"150147 212642 75 1.462118 483.7875 31\n",

"150148 212643 75 -0.627632 191.6625 12\n",

"150149 212644 75 0.268961 234.4125 15\n",

"\n",

"[150150 rows x 5 columns]"

]

},

"execution\_count": 34,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#scaling\n",

"\n",

"scaler = StandardScaler()\n",

"df['Total Price'] = scaler.fit\_transform(df['Total Price'].values.reshape(-1, 1))\n",

"df"

]

},

{

"cell\_type": "code",

"execution\_count": 35,

"id": "e1c171bd",

"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>ID</th>\n",

" <th>Store ID</th>\n",

" <th>Total Price</th>\n",

" <th>Base Price</th>\n",

" <th>Units Sold</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

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

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

" <td>-1.041443</td>\n",

" <td>-0.969377</td>\n",

" <td>20</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>-1.041443</td>\n",

" <td>-1.084958</td>\n",

" <td>28</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>-0.703497</td>\n",

" <td>-0.770322</td>\n",

" <td>19</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

" <td>-0.703497</td>\n",

" <td>-0.770322</td>\n",

" <td>44</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>5</td>\n",

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

" <td>-0.634528</td>\n",

" <td>-0.706110</td>\n",

" <td>52</td>\n",

" </tr>\n",

" <tr>\n",

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

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

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

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

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

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

" </tr>\n",

" <tr>\n",

" <th>150145</th>\n",

" <td>212638</td>\n",

" <td>75</td>\n",

" <td>0.282754</td>\n",

" <td>0.147904</td>\n",

" <td>38</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150146</th>\n",

" <td>212639</td>\n",

" <td>75</td>\n",

" <td>0.282754</td>\n",

" <td>0.147904</td>\n",

" <td>30</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150147</th>\n",

" <td>212642</td>\n",

" <td>75</td>\n",

" <td>1.462118</td>\n",

" <td>2.382466</td>\n",

" <td>31</td>\n",

" </tr>\n",

" <tr>\n",

" <th>150148</th>\n",

" <td>212643</td>\n",

" <td>75</td>\n",

" <td>-0.627632</td>\n",

" <td>-0.250208</td>\n",

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

" </tr>\n",

" <tr>\n",

" <th>150149</th>\n",

" <td>212644</td>\n",

" <td>75</td>\n",

" <td>0.268961</td>\n",

" <td>0.135061</td>\n",

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

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>150150 rows × 5 columns</p>\n",

"</div>"

],

"text/plain": [

" ID Store ID Total Price Base Price Units Sold\n",

"0 1 3 -1.041443 -0.969377 20\n",

"1 2 3 -1.041443 -1.084958 28\n",

"2 3 3 -0.703497 -0.770322 19\n",

"3 4 3 -0.703497 -0.770322 44\n",

"4 5 3 -0.634528 -0.706110 52\n",

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

"150145 212638 75 0.282754 0.147904 38\n",

"150146 212639 75 0.282754 0.147904 30\n",

"150147 212642 75 1.462118 2.382466 31\n",

"150148 212643 75 -0.627632 -0.250208 12\n",

"150149 212644 75 0.268961 0.135061 15\n",

"\n",

"[150150 rows x 5 columns]"

]

},

"execution\_count": 35,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#scaling\n",

"\n",

"scaler = StandardScaler()\n",

"df['Base Price'] = scaler.fit\_transform(df['Base Price'].values.reshape(-1, 1))\n",

"df"

]

},

{

"cell\_type": "code",

"execution\_count": 36,

"id": "ce1361d2",

"metadata": {},

"outputs": [],

"source": [

"#train\_test split\n",

"\n",

"X = df.drop('Units Sold', axis=1)\n",

"y = df['Units Sold']\n",

"X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)\n"

]

},

{

"cell\_type": "code",

"execution\_count": 37,

"id": "e1689750",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"\n",

" X\_test info\n",

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

"Index: 30030 entries, 144782 to 110483\n",

"Data columns (total 4 columns):\n",

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

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

" 0 ID 30030 non-null int64 \n",

" 1 Store ID 30030 non-null int64 \n",

" 2 Total Price 30030 non-null float64\n",

" 3 Base Price 30030 non-null float64\n",

"dtypes: float64(2), int64(2)\n",

"memory usage: 1.1 MB\n",

"None\n"

]

}

],

"source": [

"print(\"\\n X\_test info\")\n",

"print(X\_test.info())"

]

},

{

"cell\_type": "code",

"execution\_count": 38,

"id": "494240c5",

"metadata": {},

"outputs": [],

"source": [

"#train\_test split\n",

"\n",

"#X = df.drop('Units Sold', axis=1)\n",

"#y = df['Units Sold']\n",

"\n",

"#X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)"

]

},

{

"cell\_type": "code",

"execution\_count": 39,

"id": "74089084",

"metadata": {},

"outputs": [],

"source": [

"#print(\"\\n X\_test info\")\n",

"#print(X\_test.info())"

]

},

{

"cell\_type": "code",

"execution\_count": 40,

"id": "1f1f39ee",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Mean Absolute Error (MAE): 35.00456348618878\n",

"Mean Squared Error (MSE): 3281.7516859029047\n",

"Root Mean Squared Error (RMSE): 57.28657509314817\n",

"R-squared (R2): -0.00019064320674355706\n"

]

}

],

"source": [

"# Drop non-numeric columns\n",

"X = df.drop([\"Store ID\", \"Total Price\", \"Base Price\", \"Units Sold\"], axis=1)\n",

"y = df[\"Units Sold\"]\n",

"\n",

"# Import necessary libraries for model training and evaluation\n",

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

"from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score\n",

"\n",

"# Split the dataset into training and testing sets\n",

"X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)\n",

"\n",

"# Initialize the Linear Regression model\n",

"model = LinearRegression()\n",

"\n",

"# Train the model on the training data\n",

"model.fit(X\_train, y\_train)\n",

"\n",

"# Make predictions on the test data\n",

"y\_pred = model.predict(X\_test)\n",

"\n",

"# Evaluate the model\n",

"mae = mean\_absolute\_error(y\_test, y\_pred)\n",

"mse = mean\_squared\_error(y\_test, y\_pred)\n",

"rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)\n",

"r2 = r2\_score(y\_test, y\_pred)\n",

"\n",

"print(f\"Mean Absolute Error (MAE): {mae}\")\n",

"print(f\"Mean Squared Error (MSE): {mse}\")\n",

"print(f\"Root Mean Squared Error (RMSE): {rmse}\")\n",

"print(f\"R-squared (R2): {r2}\")\n",

"\n"

]

},

{

"cell\_type": "code",

"execution\_count": 41,

"id": "0ae01e9f",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Requirement already satisfied: matplotlib in c:\\users\\sanja\\anaconda3\\lib\\site-packages (3.7.2)\n",

"Requirement already satisfied: contourpy>=1.0.1 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (1.0.5)\n",

"Requirement already satisfied: cycler>=0.10 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (0.11.0)\n",

"Requirement already satisfied: fonttools>=4.22.0 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (4.25.0)\n",

"Requirement already satisfied: kiwisolver>=1.0.1 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (1.4.4)\n",

"Requirement already satisfied: numpy>=1.20 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (1.24.3)\n",

"Requirement already satisfied: packaging>=20.0 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (23.1)\n",

"Requirement already satisfied: pillow>=6.2.0 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (9.4.0)\n",

"Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (3.0.9)\n",

"Requirement already satisfied: python-dateutil>=2.7 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from matplotlib) (2.8.2)\n",

"Requirement already satisfied: six>=1.5 in c:\\users\\sanja\\anaconda3\\lib\\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)\n",

"Note: you may need to restart the kernel to use updated packages.\n"

]

}

],

"source": [

"pip install matplotlib"

]

},

{

"cell\_type": "code",

"execution\_count": 42,

"id": "4db88bb7",

"metadata": {},

"outputs": [],

"source": [

"# Import necessary libraries for model training and evaluation\n",

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

"from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score"

]

},

{

"cell\_type": "code",

"execution\_count": 43,

"id": "5dc6ddbb",

"metadata": {},

"outputs": [],

"source": [

"# Split the dataset into training and testing sets\n",

"X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)"

]

},

{

"cell\_type": "code",

"execution\_count": 44,

"id": "97a10eee",

"metadata": {},

"outputs": [],

"source": [

"# Initialize the Linear Regression model\n",

"model = LinearRegression()\n",

"\n",

"# Train the model on the training data\n",

"model.fit(X\_train, y\_train)\n",

"\n",

"# Make predictions on the test data\n",

"y\_pred = model.predict(X\_test)"

]

},

{

"cell\_type": "code",

"execution\_count": 45,

"id": "62b74090",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Mean Absolute Error (MAE): 35.00456348618878\n",

"Mean Squared Error (MSE): 3281.7516859029047\n",

"Root Mean Squared Error (RMSE): 57.28657509314817\n",

"R-squared (R2): -0.00019064320674355706\n"

]

}

],

"source": [

"# Evaluate the model\n",

"mae = mean\_absolute\_error(y\_test, y\_pred)\n",

"mse = mean\_squared\_error(y\_test, y\_pred)\n",

"rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)\n",

"r2 = r2\_score(y\_test, y\_pred)\n",

"\n",

"print(f\"Mean Absolute Error (MAE): {mae}\")\n",

"print(f\"Mean Squared Error (MSE): {mse}\")\n",

"print(f\"Root Mean Squared Error (RMSE): {rmse}\")\n",

"print(f\"R-squared (R2): {r2}\")"

]

},

{

"cell\_type": "code",

"execution\_count": 46,

"id": "839f1946",

"metadata": {},

"outputs": [],

"source": [

"import matplotlib.pyplot as plt\n",

"import seaborn as sns"

]

},

{

"cell\_type": "code",

"execution\_count": 47,

"id": "2d1f700a",

"metadata": {},

"outputs": [

{

"data": {

"image/png": "",

"text/plain": [

"<Figure size 800x600 with 1 Axes>"

]

},

"metadata": {},

"output\_type": "display\_data"

}

],

"source": [

"# Scatter plot of actual IMDb scores vs. predicted IMDb scores\n",

"plt.figure(figsize=(8, 6))\n",

"plt.scatter(y\_test, y\_pred, alpha=0.5)\n",

"plt.xlabel(\"Base Price\")\n",

"plt.ylabel(\"Total Price\")\n",

"plt.title(\"Base vs Total Price\")\n",

"plt.show()"

]

},

{

"cell\_type": "code",

"execution\_count": 48,

"id": "92c43f3c",

"metadata": {},

"outputs": [

{

"data": {

"image/png": "",

"text/plain": [

"<Figure size 600x500 with 1 Axes>"

]

},

"metadata": {},

"output\_type": "display\_data"

}

],

"source": [

"# Distribution plot of the residuals (predicted - actual IMDb scores)\n",

"residuals = y\_pred - y\_test\n",

"plt.figure(figsize=(6, 5))\n",

"sns.histplot(residuals, kde=True)\n",

"plt.xlabel(\"Residuals (Base Price - Total Price)\")\n",

"plt.ylabel(\"Frequency\")\n",

"plt.title(\"Residuals Distribution\")\n",

"plt.show()"

]

},

{

"cell\_type": "code",

"execution\_count": 49,

"id": "1cf07b3b",

"metadata": {},

"outputs": [],

"source": [

"from sklearn.ensemble import RandomForestRegressor\n",

"from sklearn.model\_selection import train\_test\_split\n",

"from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score\n",

"import matplotlib.pyplot as plt\n",

"import seaborn as sns\n"

]

},

{

"cell\_type": "code",

"execution\_count": 50,

"id": "bfd5ed7f",

"metadata": {},

"outputs": [],

"source": [

"X = df.drop([\"Store ID\", \"Total Price\", \"Base Price\", \"Units Sold\"], axis=1)\n",

"y = df[\"Units Sold\"]\n",

"\n",

"X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)\n"

]

},

{

"cell\_type": "code",

"execution\_count": 51,

"id": "dcf08c52",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<style>#sk-container-id-2 {color: black;}#sk-container-id-2 pre{padding: 0;}#sk-container-id-2 div.sk-toggleable {background-color: white;}#sk-container-id-2 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-2 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-2 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-2 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-2 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-2 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-2 input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-2 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-2 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-2 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-2 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-2 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-2 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-2 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-2 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-2 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-2 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-2 div.sk-item {position: relative;z-index: 1;}#sk-container-id-2 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-2 div.sk-item::before, #sk-container-id-2 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-2 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-2 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-2 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-2 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-2 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-2 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-2 div.sk-label-container {text-align: center;}#sk-container-id-2 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-2 div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-2\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>RandomForestRegressor(random\_state=42)</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.</b></div><div class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-2\" type=\"checkbox\" checked><label for=\"sk-estimator-id-2\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">RandomForestRegressor</label><div class=\"sk-toggleable\_\_content\"><pre>RandomForestRegressor(random\_state=42)</pre></div></div></div></div></div>"

],

"text/plain": [

"RandomForestRegressor(random\_state=42)"

]

},

"execution\_count": 51,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"model = RandomForestRegressor(random\_state=42)\n",

"model.fit(X\_train, y\_train)"

]

},

{

"cell\_type": "code",

"execution\_count": 52,

"id": "de066b9b",

"metadata": {},

"outputs": [

{

"data": {

"image/png": "",

"text/plain": [

"<Figure size 1000x600 with 1 Axes>"

]

},

"metadata": {},

"output\_type": "display\_data"

}

],

"source": [

"if isinstance(model, RandomForestRegressor):\n",

" feature\_importance = model.feature\_importances\_\n",

" feature\_names = X\_train.columns\n",

" plt.figure(figsize=(10, 6))\n",

" plt.barh(feature\_names, feature\_importance)\n",

" plt.xlabel(\"Feature Importance\")\n",

" plt.ylabel(\"Features\")\n",

" plt.title(\"Feature Importance Plot\")\n",

" plt.show()"

]

},

{

"cell\_type": "code",

"execution\_count": 26,

"id": "0a0a5e05",

"metadata": {},

"outputs": [],

"source": [

"import pandas as pd\n",

"\n",

"def predict\_product\_demand(ID, store\_id, total\_price, base\_price, units\_sold):\n",

" # Assuming you have a label\_encoder for transforming Store ID\n",

" # You should replace `label\_encoder` with the actual variable you're using\n",

" store\_id\_encoded = label\_encoder.transform([store\_id])[0]\n",

"\n",

" # Create input data for prediction\n",

" input\_data = pd.DataFrame({'Base Price': [base\_price], 'Store ID': [store\_id], 'Total Price': [total\_price], 'Units Sold': [units\_sold]})\n",

"\n",

" # Make a prediction using your 'model'\n",

" # Replace 'model' with the actual machine learning model you have trained\n",

" predicted\_product\_demand = model.predict(input\_data)\n",

"\n",

" return predicted\_product\_demand\n"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"id": "8ca756e9",

"metadata": {},

"outputs": [],

"source": [

"# User input\n",

"ID = input(\"Enter the ID: \")\n",

"store\_id = input(\"Enter the Store ID: \")\n",

"total\_price = input(\"Enter the Total Price: \")\n",

"base\_price = float(input(\"Enter the Base Price: \"))\n",

"units\_sold = input(\"Enter the Units Sold: \")\n",

"\n",

"predicted\_score = predict\_product\_demand(ID, store\_id, total\_price, base\_price, units\_sold)\n",

"print(f\"Predicted Product Demand : {predicted\_score:.2f}\")"

]

}

],

"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": "3.11.5"

}

},

"nbformat": 4,

"nbformat\_minor": 5

}