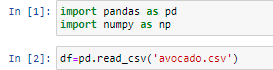
**Avocado Dataset**

**Problem Definition**

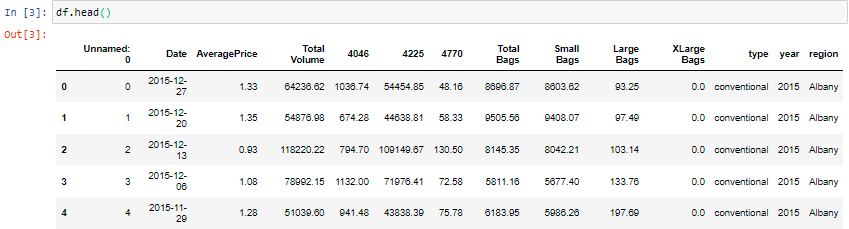
**Avocado is a fruit consumed by people heavily in the United States.**The data was downloaded from the Hass Avocado Board website in May of 2018 & compiled into a single CSV. The dataset represents weekly 2018 retail scan data for National retail volume (units) and price. Retail scan data comes directly from retailers’ cash registers based on actual retail sales of Hass avocados. Starting in 2013, the dataset below reflects an expanded, multi-outlet retail data set. Multi-outlet reporting includes an aggregation of the following channels: grocery, mass, club, drug, dollar and military. The Average Price (of avocados) in the dataset reflects a per unit (per avocado) cost, even when multiple units (avocados) are sold in bags. The Product Lookup codes (PLU’s) in the dataset are only for Hass avocados. Other varieties of avocados (e.g. green skins) are not included in this dataset.

**1).Data Acquistion -** In data acquisition, we collect data from certain websites, like, kaggle and UCI that becomes the training data and is useful to make predictions. However, when we acquire data then we should be sure enough that data must have enough features so that we can make predictions easily. Generally, data must be in the form of CSV, i.e., comma separated value and currently the most supported size is 1.95 but it is not appropriate for big data.

In this project, we have loaded avocado dataset in the form of CSV by using kaggle with the help of pandas, which is the most popular python library that is used for data analysis, and it mainly works on dataset, like, it is used for merging, joining, reshaping, pivoting, aligning, analyzing and arranging datasets. We have imported numpy that is mainly used for calculations in our dataset in the following commands:-



And following is the illustration of dataset in which by using **df.head(),** We can see the first five rows of the dataset:-



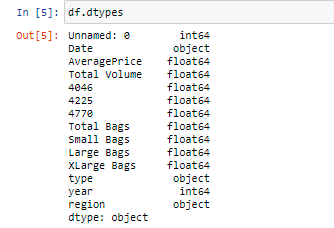
* **Date** - The date of the observation
* **AveragePrice** - the average price of a single avocado
* **type** - conventional or organic
* **year** - the year
* **Region** - the city or region of the observation
* **Total Volume** - Total number of avocados sold
* **4046** - Total number of avocados with PLU 4046 sold
* **4225** - Total number of avocados with PLU 4225 sold
* **4770** - Total number of avocados with PLU 4770 sold

**2).Data Cleaning -** Due the problem of incorrect or inconsistent data in both public and private sector which led to fast inferences and investments, there is a need of some packages which can clean or wash our address data when we enter them into our system using application programming interface which is a set of subroutine definitions, communication protocols and tools for building software. Data cleaning is the process of detecting and correcting corrupt or inaccurate records from a record set, table or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty data. Data is also missing because sometimes user don’t remember to fill it in a field, sometimes data is gone while shifting it manually from a provision database and sometimes a programming error has occurred. Mostly, 80 % of work is done in this step of data cleaning.

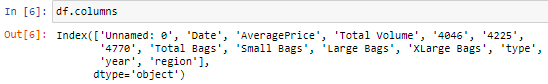
* We can check the shape of the dataset by using the code- **df.shape()** from which we can observe that there are 18249 rows and 14 columns in this dataset using the following command:-



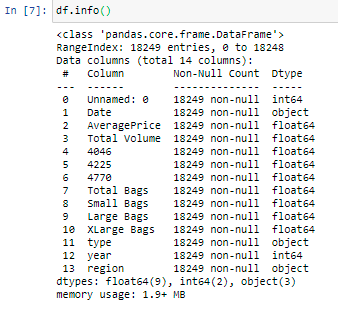
* We can check the datatypes of the columns by using-**df.dtypes**. From the following commands we can observe that there are two categorical features present in this dataset ‘Date’, ‘type’ and ‘region’ and rest are numerical features:’Unnamed:0’, ‘AveragePrice’, ‘Total Volume’, ‘4046’, ‘4225’, ‘4770’, ‘Total Bags’, ‘Small Bags’, ‘Large Bags’, ‘XLarge Bags’ and ‘year’.



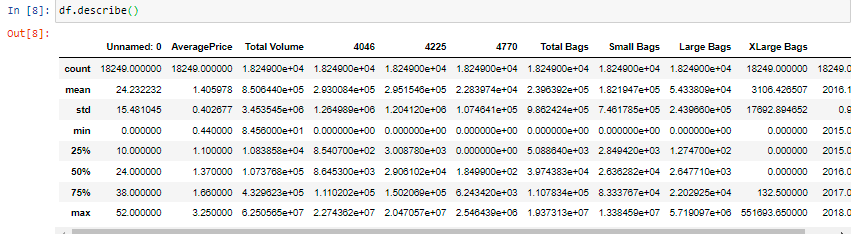
* We can check the columns present in this dataset by using the code-**df.columns** in the following command :-



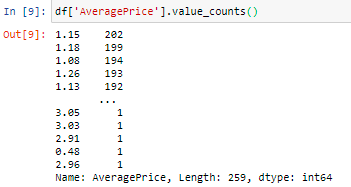
* After loading the file we can start analyzing our dataset such as checking the information about dataset using **df.info()** in the following command :-



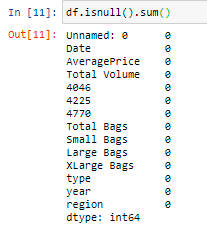
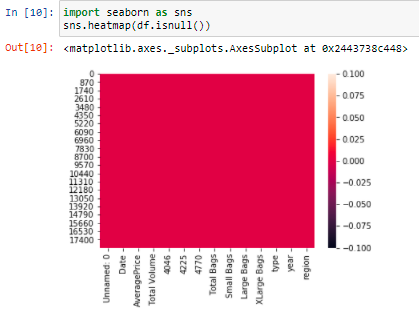
* **Summary Statistics**
* Now, using the following command of data.describe ( ) we find count, mean, standard deviation, minimum, maximum, 25%, 50% and also 75% of our dataset which is used to decide that which column contribute more towards training of our model to make certain predictions and also used in testing of our model in the following commands :-



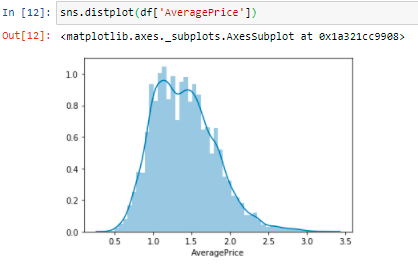
* From this we can observe that :-
* The mean is more than median for all columns except total bags.
* There is large difference between 75 % and max of ‘XLarge Bags’ Column.
* In the following command, we count different types of values in ‘AveragePrice column’ using **df['AveragePrice'].value\_counts()** and as target variable has continuous values, so, this is a regression problem:-



* **Data Visualizations**
* In order to clean our data, we have to clean it by checking that if there are null values in our data or not by using the command- **df.isnull().sum()** and if there are null values in our data then we replace these null values with some objects. Similarly, we can visualize it with the help of heatmap by using the following commands :-

****

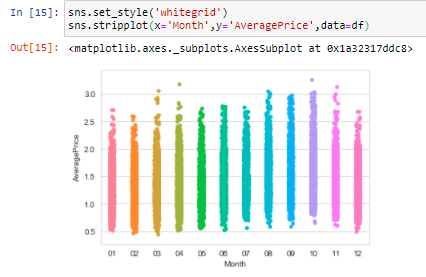
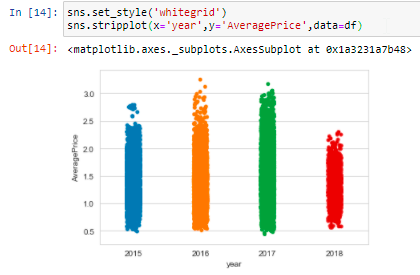
* From this, we can observe that the data has no missing values.
* **Univariate Analysis**
* In this section, we analyze attributes individually; check their distribution and range of values. We use distplot to analyze ‘AveragePrice’ attribute individually. Since this attribute has numerical values and can be better visualized using distplot by the following commands :-

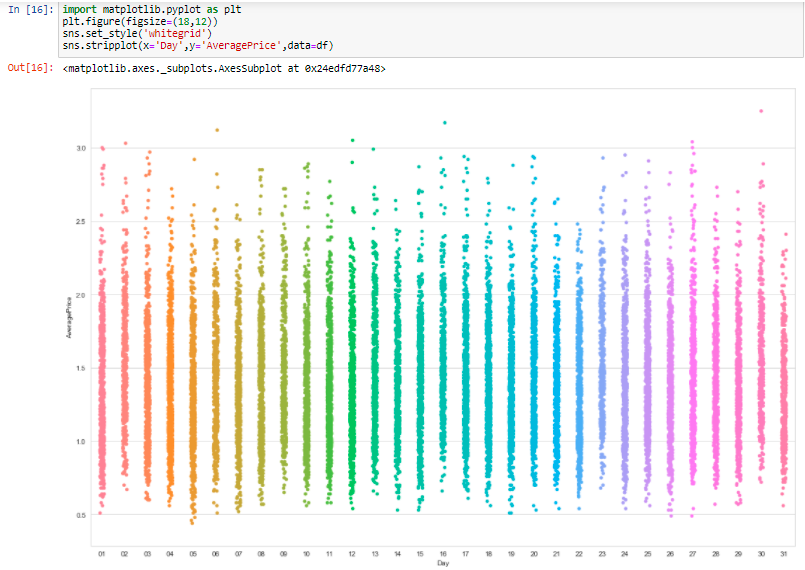


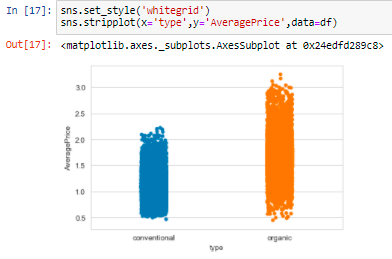
* From these it can be observed that:-
* The average price of single avocado is majorly in the range of 1.0-1.7.
* **Feature Engineering**
* In order to clean data, we have to build some new attributes using the attributes that are already present in data. Here we are using .str.split() method to make ‘Month’ and ‘Day’ columns as following :-



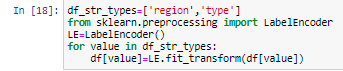
* In this section, we analyzing the impact of each attribute on each other. We use stripplot to analyze the impact of 'year', ’month’, ‘day’ and ‘type’ on ‘AveragePrice’ attribute. Since 'year', ’month’, ‘day’ and ‘type’ attributes have categorical values and ' AveragePrice ' has numerical values, So, It can be better visualized using stripplot by the following commands :-

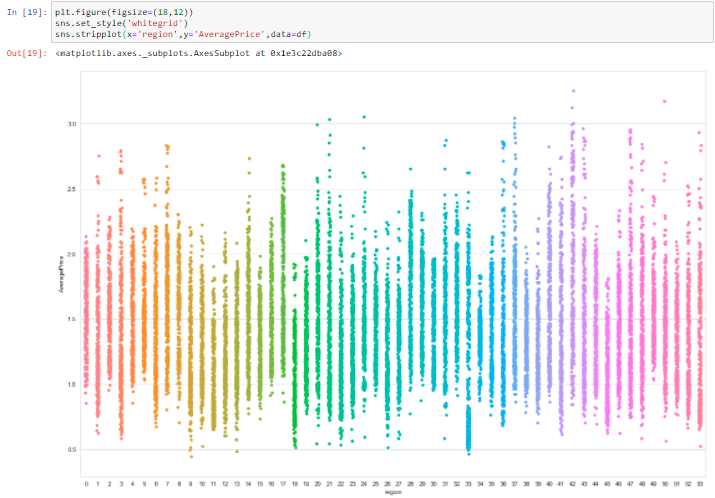




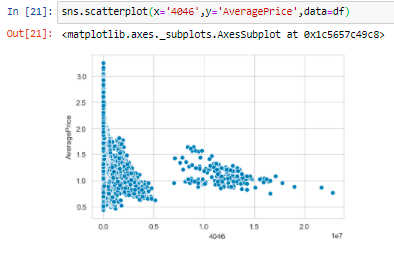
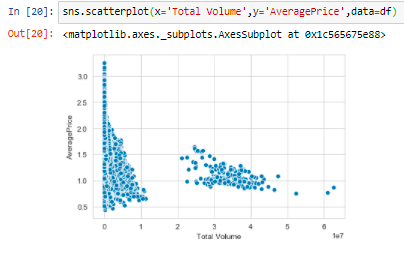


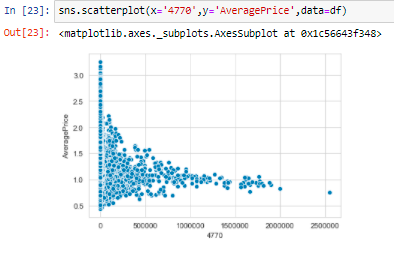
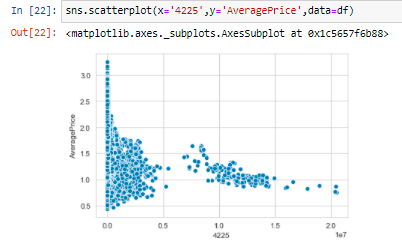
* From these it can be observed that:-
* The average price of single avocado is least in the year 2018.
* The average price of single avocado is highest in the year 2016.
* The average price of single avocado is also high in the year 2017.
* The average price of single avocado is least in the month of February.
* The average price of single avocado is highest in the month of October.
* The average price of single avocado is also high in the month of March, April, August and November.
* The average price of single avocado is least on the 31st day of months.
* The average price of single avocado is highest on the 30th day of months.
* The average price of single avocado is also high for the 27th, 16th, 12th, 6th, 2nd, 1st and 13th day of months.
* The average price of single organic avocado is higher than conventional avocado.
* The average price of single organic avocado is in the range of 0.5-3.8 approximately.
* The average price of single conventional avocado is in the range of 0.55-2.25 approximately.
* **Label Encoding**
* We encode all the categorical variables to numerical ones by using “Label Encoder” as shown below:

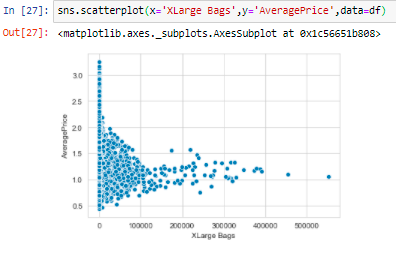
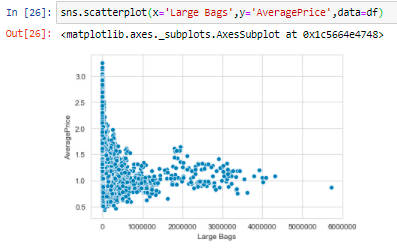
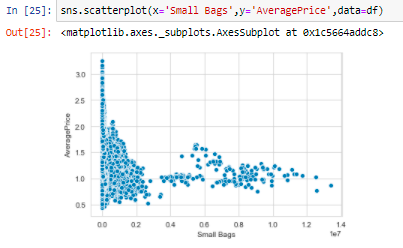
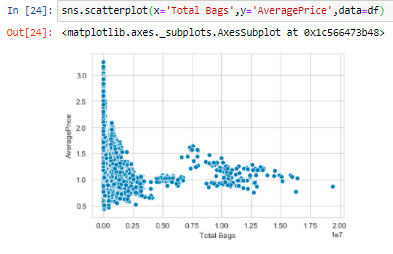




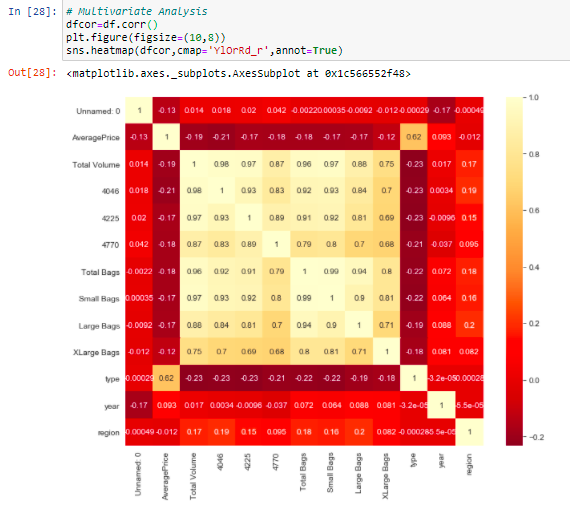
* From these it can be observed that :-
* SanFrancisco has the highest average price of single avocado.
* SouthCentral has the lowest average price of single avocado.
* Pittsburgh, Houston, DallasFtWorth and GreatLakes also have low average price of single avocado.
* Tampa, RaleighGreensboro, MiamiFtLauderdale and LasVegas also have high average price of single avocado.
* **Bivariate Analysis**
* In this section, we analyze the impact of each attribute on each other. We use scatterplot to analyze the impact of 'TotalVolume', ‘4046’, ‘4770’, ‘Total Bags’, ‘Small Bags’ and ‘Large Bags’ on ‘AveragePrice’ attribute. Since all these attributes have numerical values. So, It can be better visualized using violinplot by the following commands :-





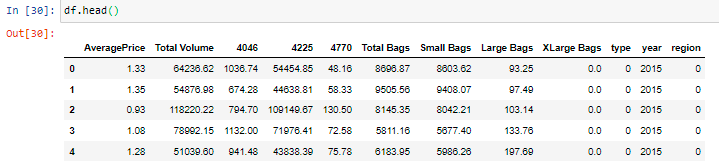


* From these it can be observed that:-
* Total volume of avocados sold is inversely proportional to the average price of a single avocado.
* Total number of avocados with Product Lookup codes (PLU’s) 4046 sold is also inversely proportional to the average price of
* a single avocado.
* Total number of avocados with Product Lookup codes (PLU’s) 4225 sold is also inversely proportional to the average price of a single avocado.
* Total number of avocados with Product Lookup codes (PLU’s) 4770 sold is also inversely proportional to the average price of a single avocado.
* Average price of a single avocado is also inversely proportional to total numbers of bags in which avocados are sold.
* Average price of a single avocado is also inversely proportional to total numbers of small bags in which avocados are sold.
* Average price of a single avocado is also inversely proportional to total numbers of large bags in which avocados are sold.
* Average price of a single avocado is also inversely proportional to total numbers Xlarge bags in which avocados are sold.
* **Multivariate Analysis**
* We can check the correlation of all the attributes with the target variable-Average Price as follows:



* From these it can be observed that :-
* Average price is highly positively correlated to type.
* Type is highly negatively correlated to Total Volume.
* Total Volume is highly positively correlated to 4046.
* Total Volume is highly negatively correlated to 4046.
* **Dropping of Columns**
* We drop Unnamed:0, Date, Day and Month columns as they have no impact on target variable.

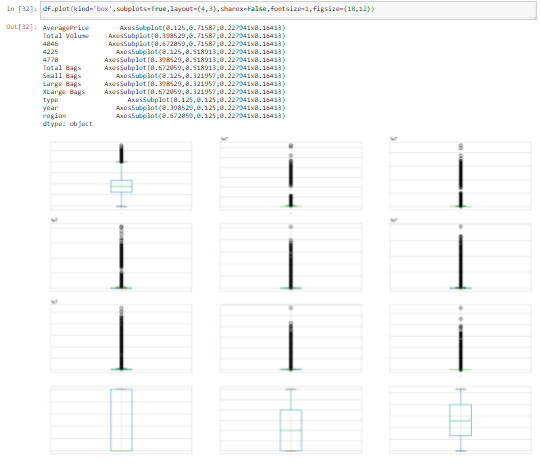
****



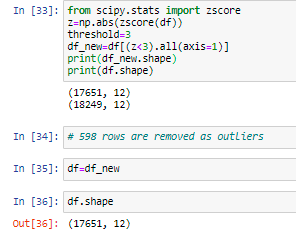
* The final shape of our dataset after dropping 3 columns is (18249,12).

****

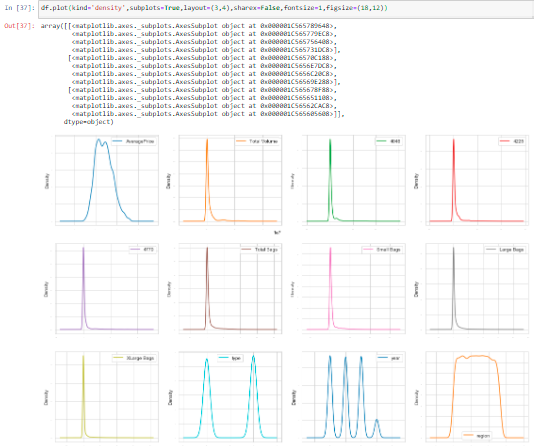
* **Plotting Outliers**
* Now we plot the outliers by using boxplot for all the attributes as follows:

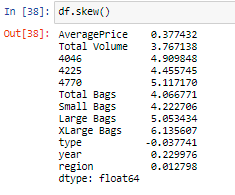


* From these it can be observed that :-
* All the columns have outliers except type, year and region
* **Removing Outliers**
* In order to remove outliers from our data, we have to remove outliers by using z-score or IQR method. Here we are using z-score method as following :-



* We have removed 598 rows from the data as outliers were present in them. Now, the dataframe has 17651 rows and 12 columns.
* **Removing Skewness**
* In order to remove skewness from our data, we have to remove skewness by checking that whether the attributes are left skewed or right skewed using- **df.skew()**. Similarly, we can visualize it with the help of distplot by using the following command :-

****

****

* From these it can be observed that :-
* Total Volume, 4046, 4225, 4770, Total Bags, Small Bags, Large Bags, XLarge Bags are right skewed.
* **Dropping of Columns**
* Because the skewness of 4046, 4225, 4770, Small Bags, Large Bags and XLarge Bags attributes is very high and their corresponding aggregate columns are also present in the dataset as columns of Total Bags and Total Volume. So, We drop 4046, 4225, 4770, Small Bags, Large Bags and XLarge Bags attributes from the dataset.

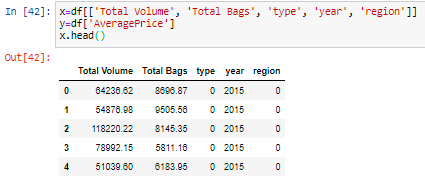




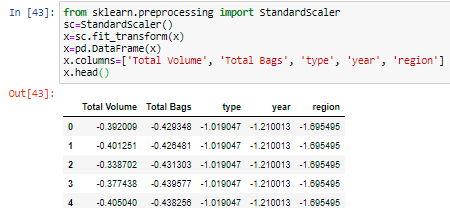
* The final shape of our dataset after dropping 3 columns is (17651,12).



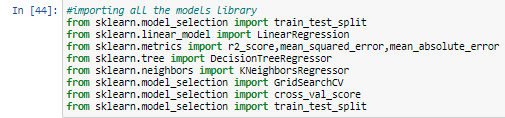
* **Train Test Split**
* We usually split our data into training data which contains a known output so that a model learn on this data in order to be generalized to other data and testing data in order to test our model on subset for prediction. Training data must contains those variables which affect testing data and it should be also clean, properly classified, not containing nan values and should contain more numerical data rather than categorical data. Training and testing data must be divided in the ratio of 80:20. In this project, we have chosen our train and target columns as following to make predictions about the “AveragePrice” column :-



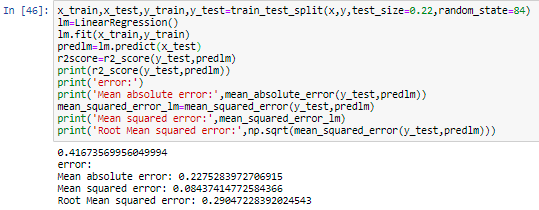
* **Scaling of Data**
* In order to scale our data, we have to scale data by using Standard Scaler or Min-Max Scaler method. Here we are using Standard Scaler method as following :-

****

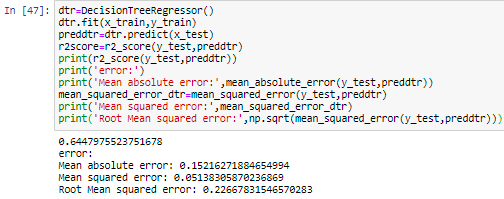
* We have scaled all columns in the same range of -1 to +1.
* **Regression in Machine Learning**
* Regression analysis is a predictive modelling technique which investigates the relationship between a dependent and independent variable. There are many uses of regression analysis, first is, determining the strength of predicators, for example, someone wants to know the relationship between age and income, second is, forecasting an effect, for example, how much the dependent variable will change with the change in independent variable, third is, trend forecasting, for example, what will be the price of bit coins in next 6 months. There are various kind of regression like, linear regression, logistic regression, polynomial regression, decision tree regressor , k-neighbors regressor and so on. But, the widely used regressions are linear regression and logistic regression.
* **Algorithms**
* **Linear Regression**
* In Linear Regression, we fit our data to a straight line equation like y = mx+c and so we want to find the correlation between x and y variable . In this, we use continuous variable while in Logistic Regression we use categorical variable. In Linear Regression, the output is value of the variable.
* Using the scikit learn library of python we have imported train test split to divide train and test data. We have used test size argument here which represents the proportion of dataset which we need to include in the test split and the default argument for test size is 0.25 whereas random state here provides us a seed to the random number generator whose default argument is 12. Here we have used linear regression model to make predictions about our model and we have fitted our model using training data in the following commands :-



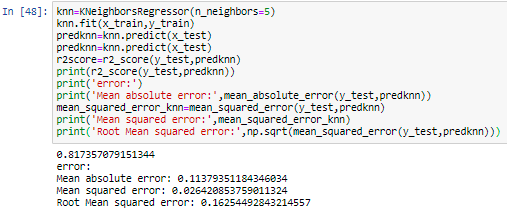




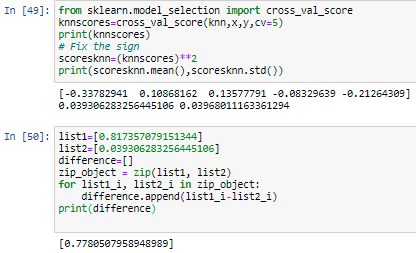
* **Decision Tree Regressor**
* Decision tree represents a function that takes input a vector of attribute values and returns a “decision”- a single output value. The input and output values can be discrete or continuous. A decision tree gives its decision by doing a sequence of tests. Decision tree has so many algorithms but the important one is ID3 (Iterative Dichotomiser) which is the most common algorithm and here dichotomization means dividing into two completely opposite things. Each attribute has divided into two most dominant attributes which is determined by calculating entropy, used, as a measure for information content of probability distribution and information gain answers us how much information an attribute provide us about the class.
* Using the Scikit learn library of python we have imported Decision Tree Regressor and again fitted our model using training data and have made certain predictions:-



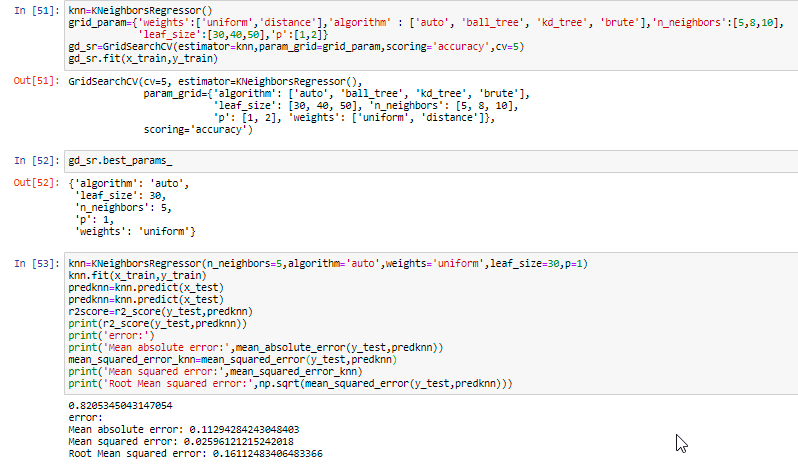
* **K-Neighbors Regressor**
* KNN or K-nearest neighbor can be used for both classification and regression predictive problems in industry.
* Using the scikit learn library of python we have imported K Neighbors Regressor and again fitted our model using training data and have made certain predictions in the following commands :-



* **Concluding Remarks**
* K - Neighbors Regressor is the best algorithm.
* **Cross – Validation Scores**
* In order to find cross validation scores, we calculate cross validation scores of the best model by using- **cross\_val\_score()** in the following command :-

****

* **Concluding Remarks**
* As the difference between accuracy scores and cross validation scores of K-neighbors Regressor is only 7%. So, It is verified that K-neighbors Regressor is the best algorithm for this model with accuracy of 81.735%.
* **Hyper Parameter Tuning**
* In order to increase the accuracy score of the model, we use hyper parameter tuning of the best model in order to find best parameters by using- **GridSearchCV()** in the following commands :-

****

* **Concluding Remarks**
* Accuracy of K-Neighbors Regressor has changed from 81.735% to 82.053% with hyper parameter tuning.

**3) Data Deployment**

* In this last stage, we deploy the model to production environment so that we can make the models available to users for making predictions and important business decisions.
* In order to dump the model which we have developed so that we can use it to make predictions in future, we have saved or dumped the best model ,i.e., K-Neighbors Regressor by using- **joblib.dump()** in the following commands :-

