**Sales Channel Prediction Case Study**

Advertising is the best way to communicate to the customers. So here are top reasons to advertise your business.

* **Advertising makes your company money.**
* **Advertising keeps your consumer up to date.**
* **Advertising keeps your business at the top of your consumer’s mind.**
* **Advertising attracts new customers.**
* **Advertising generates brand loyalty.**

So, it is very important for anyone to spend money wisely on different source mode of marketing to make a profit and increase sales. Here we are analysing the sales proportion from different type of advertisement sources.

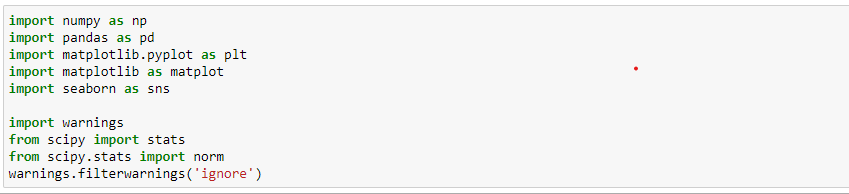
**PROBLEM STATEMENT:**

Build An effective model which predicts sales based on the money spent on different platforms for marketing.

**ABOUT THE DATASET:**

Use the advertising dataset given and analyse how company can select and predict an effective model from its different available sales channel such as TV, radio, and newspaper for good revenue.

**IMPORTING LIBRARIES:**

**

**Numpy** module is mainly used for working with numerical data. It provides us with a powerful object known as an Array. With Arrays, we can perform mathematical operations on multiple values in the Arrays at the same time, and also perform operations between different Arrays, similar to matrix operations.

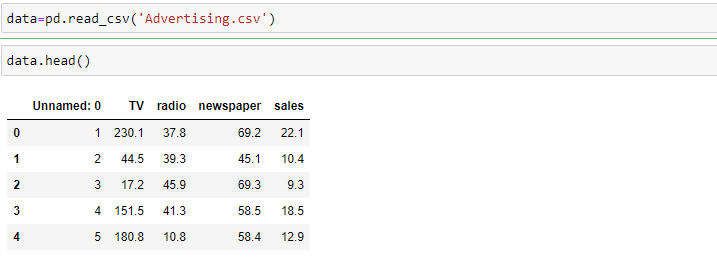
**Pandas** module is used for working with tabular data. It allows us to work with data in table form, such as in CSV or SQL database formats. We can also create tables of our own, and edit or add columns or rows to tables. Pandas provides us with some powerful objects like DataFrames and Series which are very useful for working with and analyzing data.

**Matplotlib** module is used for data visualization. It provides functionality for us to draw charts and graphs, so that we can better understand and present the data visually.

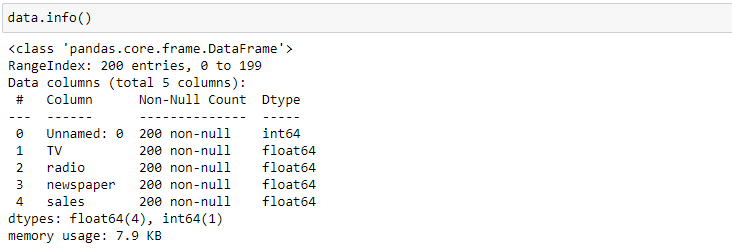
**Seaborn** is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

**SciPy** in Python is an open-source library used for solving mathematical, scientific, engineering, and technical problems. It allows users to manipulate the data and visualize the data using a wide range of high-level Python commands. SciPy is built on the Python NumPy extention.

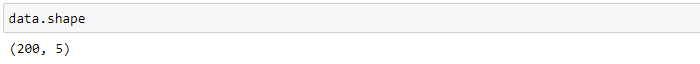
**READING AND EXPLORING THE DATASET:**



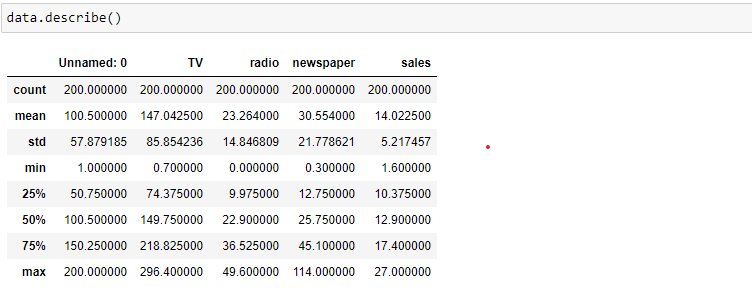
*#It gives the value of first 5 rows for each column.*

****

*#It gives the information of datatype.*



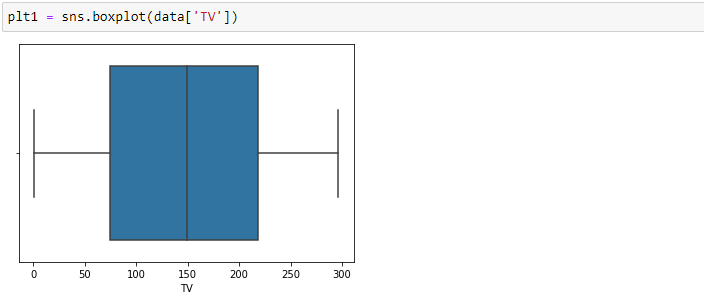
*#It gives total number of ROWS and COLUMNS present in the dataset.*

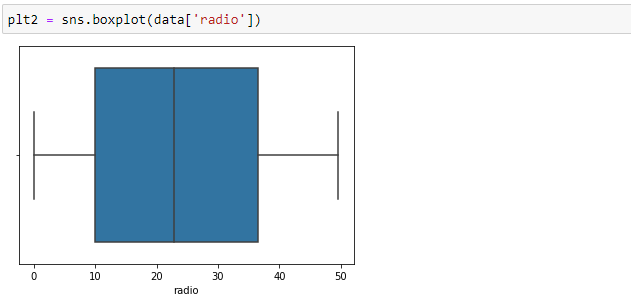


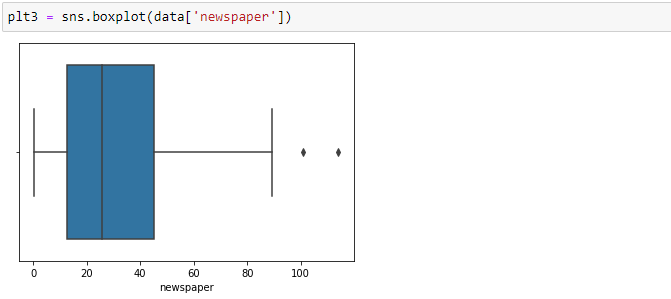
*#It gives some basic statistical details like count, percentile, mean, std etc.*

**EXPLORATORY DATA ANALYSIS:**

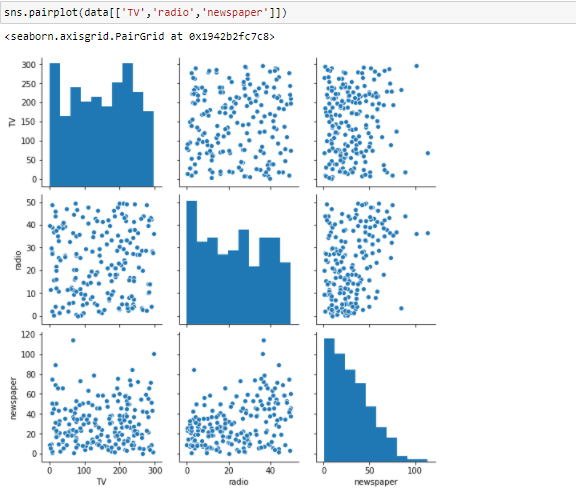
Exploratory Data Analysis (EDA) is an approach to analysing datasets to summarize their main characteristics, often with visual methods. EDA is used for seeing what the data can tell us before the modelling task.

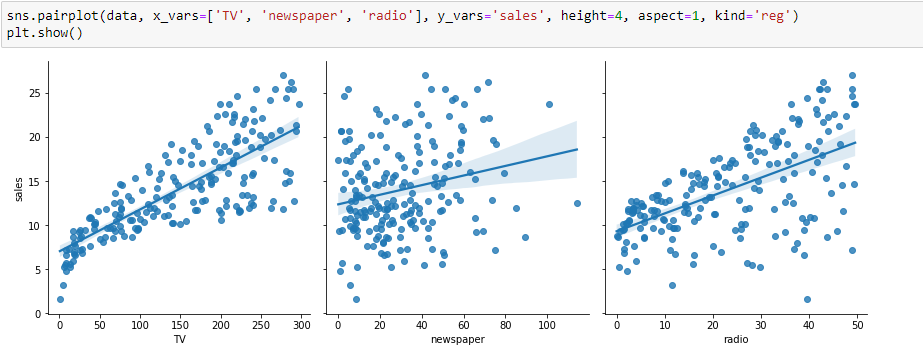




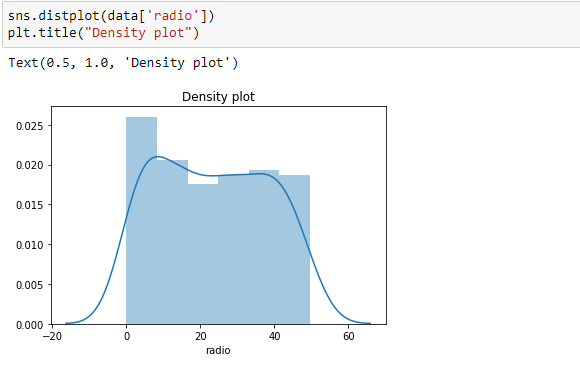
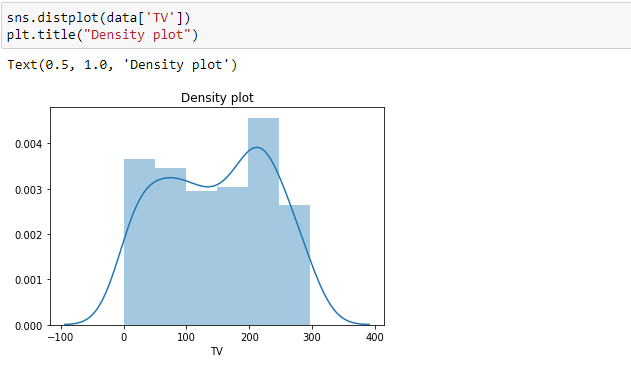


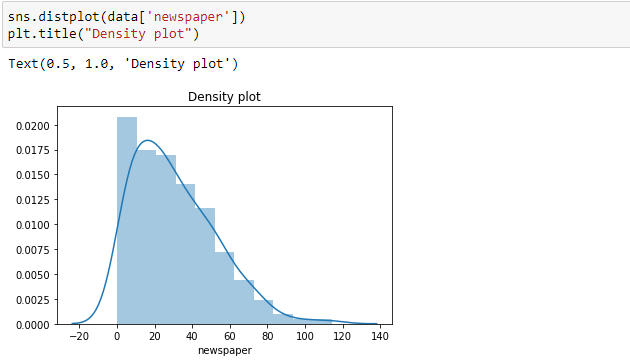
*#We can observe that there are no considerable outliers present in the data.*

**

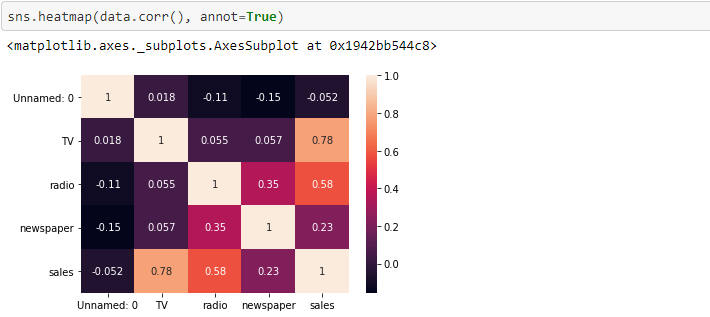


*#It gives relation of TV, NEWSPAPER and RADIO’S SALES.* *Looking at the scatter plots between Sales and Newspaper/TV/Radio it is evident that Sales and TV has strong positive relationship.*

**



*#IDensity plot gives TV, NEWSPAPER and RADIO’S SALES distribution.*

**

*#A heatmap is a two-dimensional graphical representation of data where the individual values that are contained in a matrix are represented as colours. Each square shows the correlation between the variables on each axis. Correlation ranges from -1 to +1. Values closer to zero means there is no linear trend between the two variables. The close to 1 the correlation is the more positively correlated they are; that is as one increases so does the other and the closer to 1 the stronger this relationship is. A correlation closer to -1 is similar, but instead of both increasing one variable will decrease as the other increases.*

*From the Heatmap we can say that there is a strong relationship between TV ads and sales compared to newspaper and radio.*

**PERFORMING LINEAR REGRESSION:**

Equation of linear regression:  
y=c+m1x1+m2x2+...+mnxny=c+m1x1+m2x2+...+mnxn

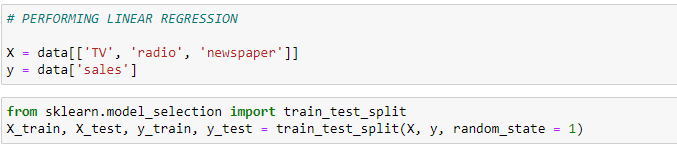
* yy is the response
* cc is the intercept
* m1m1 is the coefficient for the first feature
* mnmn is the coefficient for the nth feature

In our case:

y=c+m1×TVy=c+m1×TV

The mm values are called the model coefficients or model parameters.

Here we are using Train-Test Split method for model. You now need to split our variable into training and testing sets. You'll perform this by importing train\_test\_split from the sklearn.model\_selection library.



IN Train-Test Split Method,

**X\_train** - This includes your all independent variables, these will be used to train the model, for eample if we have specified the test\_size = 0.4, this means 60% of observations from your complete data will be used to train/fit the model and rest 40% will be used to test the model.

**X\_test** - This is remaining 40% portion of the independent variables from the data which will not be used in the training phase and will be used to make predictions to test the accuracy of the model.

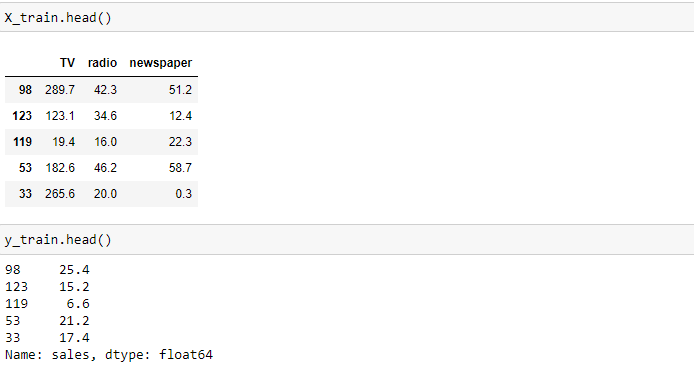
**y\_train** - This is your dependent variable which needs to be predicted by this model, this includes category labels against your independent variables, we need to specify our dependent variable while training/fitting the model.

**y\_test** - This data has category labels for your test data, these labels will be used to test the accuracy between actual and predicted categories

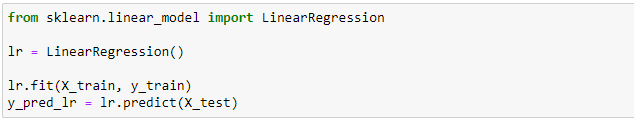
**train\_size** is the number that defines the size of the training set. If you provide a float, then it must be between 0.0 and 1.0 and will define the share of the dataset used for testing. If you provide an int, then it will represent the total number of the training samples. The default value is None.

**test\_size** is the number that defines the size of the test set. It’s very similar to train\_size. You should provide either train\_size or test\_size. If neither is given, then the default share of the dataset that will be used for testing is 0.25, or 25 percent.

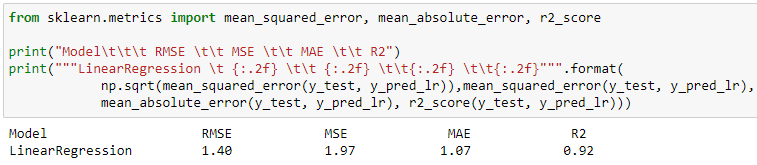
**random\_state** is the object that controls randomization during splitting. It can be either an int or an instance of RandomState. The default value is None.



Now, we are importing LINEAR REGRESSION MODEL as shown below,



Aftet that, we have to import mean\_squared\_error, mean\_absolute\_error, r2\_score from sklearn.metrics library.



Here,

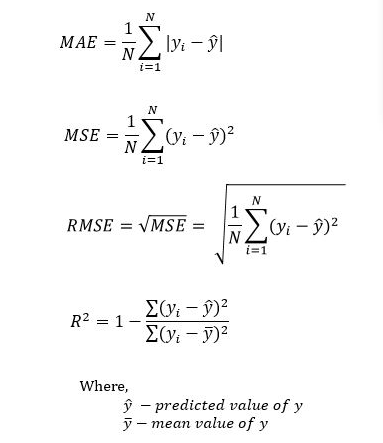
The **MSE, MAE, RMSE,** and **R-Squared metrics** are mainly used to evaluate the prediction error rates and model performance in regression analysis.

**MAE** (Mean absolute error) represents the difference between the original and predicted values extracted by averaged the absolute difference over the data set.

**MSE** (Mean Squared Error) represents the difference between the original and predicted values extracted by squared the average difference over the data set.

**RMSE** (Root Mean Squared Error) is the error rate by the square root of MSE.

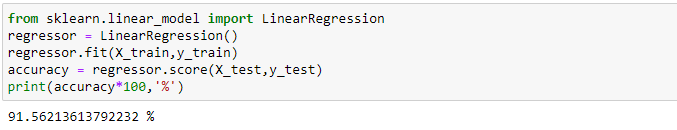
**R-squared** (Coefficient of determination) represents the coefficient of how well the values fit compared to the original values. The value from 0 to 1 interpreted as percentages. The higher the value is, the better the model is.



For this dataset we got,

RMSE MSE MAE R2

1.40 1.97 1.07 0.92



For this model, we got accuracy of 91.5% using Linear Regression technique which consider as a best value for this dataset.



We can save Model using this command as ‘to\_csv()’ function converts Data Frame into CSV data.

***#SOURCE\_CODE:***

*import numpy as np*

*import pandas as pd*

*import matplotlib.pyplot as plt*

*import matplotlib as matplot*

*import seaborn as sns*

*import warnings*

*from scipy import stats*

*from scipy.stats import norm*

*warnings.filterwarnings('ignore')*

*data=pd.read\_csv('Advertising.csv')*

*data.head()*

*data.info()*

*data.shape*

*data.describe()*

*plt1 = sns.boxplot(data['TV'])*

*plt2 = sns.boxplot(data['radio'])*

*plt3 = sns.boxplot(data['newspaper'])*

*sns.pairplot(data[['TV','radio','newspaper']])*

*sns.pairplot(data, x\_vars=['TV', 'newspaper', 'radio'], y\_vars='sales', height=4, aspect=1, kind='reg')*

*plt.show()*

*sns.distplot(data['TV'])*

*plt.title("Density plot")*

*sns.distplot(data['radio'])*

*plt.title("Density plot")*

*sns.distplot(data['newspaper'])*

*plt.title("Density plot")*

*sns.heatmap(data.corr(), annot=True)*

*X = data[['TV', 'radio', 'newspaper']]*

*y = data['sales']*

*from sklearn.model\_selection import train\_test\_split*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, random\_state = 1)*

*X\_train.head()*

*y\_train.head()*

*from sklearn.linear\_model import LinearRegression*

*lr = LinearRegression()*

*lr.fit(X\_train, y\_train)*

*y\_pred\_lr = lr.predict(X\_test)*

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

*print("Model\t\t\t RMSE \t\t MSE \t\t MAE \t\t R2")*

*print("""LinearRegression \t {:.2f} \t\t {:.2f} \t\t{:.2f} \t\t{:.2f}""".format(*

*np.sqrt(mean\_squared\_error(y\_test, y\_pred\_lr)),mean\_squared\_error(y\_test, y\_pred\_lr),*

*mean\_absolute\_error(y\_test, y\_pred\_lr), r2\_score(y\_test, y\_pred\_lr)))*

*from sklearn.linear\_model import LinearRegression*

*regressor = LinearRegression()*

*regressor.fit(X\_train,y\_train)*

*accuracy = regressor.score(X\_test,y\_test)*

*print(accuracy\*100,'%')*

*df=pd.DataFrame(y\_pred\_lr)*

*df.to\_csv('Sales prediction model.csv')*

**CONCLUSION:**

Till now,

* I explored the advertising data set and our goal was to predict the sales using the given features.
* I used matplotlib and seaborn libraries for EDA and got some useful visual graphs and charts for prediction.
* I used Scikit learn to fit linear regression to the entire data set and calculated the mean squared error.
* I made a train-test split and calculated the mean squared error for my training data and test data and after that predicted the accuracy for model.

**REFERENCE:**

* *https://github.com/dsrscientist/DSData/blob/master/Advertising.csv*
* [*https://www.kaggle.com/ashydv/sales-prediction-simple-linear-regression*](https://www.kaggle.com/ashydv/sales-prediction-simple-linear-regression)
* *https://github.com/ditikrushna/Predict-Sales-Revenue-Using-Multiple-Regression-Model*