Article on Sales prediction

If you are doing business and you are not into TV Advertising, you are losing business. It is the most dependable and traditional way of advertising. TV advertising channel appeals to masses at large.

INTRODUCTION:

If you're selling a product or service and want to succeed right away, you need to communicate it to people through advertising and marketing. Television, radio and newspapers are the most important media for effective offline marketing. However, over the last 15 years, these top three marketing tools have changed significantly.

This article describes some of the key changes that have occurred and how they can have the greatest impact on your advertising budget.

PROBLEM DEFINITION:

The advertising dataset captures sales revenue generated with respect to advertisement spends across multiple channels like radio, tv and newspaper. The distribution strategy and the channel design have to be right the first time. It is not possible for clients to directly increase sales of the product, but they can control the advertising expenditure in each of the three media.
Therefore, if we determine that there is an association between advertising and sales, then we can instruct our client to adjust advertising budgets, thereby indirectly
Increasing sales.

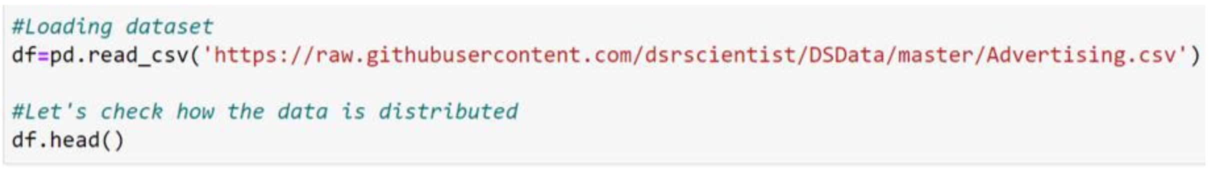
The case study of Sales channel include the detailed study of TV, radio and newspaper channel and predict the total sales generated from all the sales channel.

ABOUT THE DATSET:

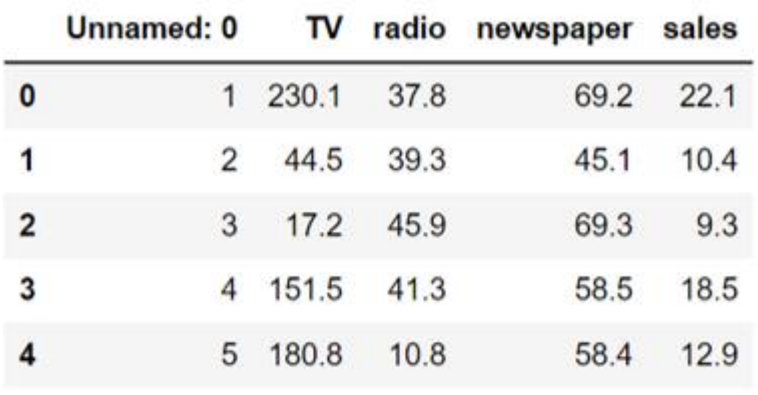
Features:

- TV: advertising dollars spent on TV for a single product in a given market (in thousands of dollars)
- Radio: advertising dollars spent on Radio
- Newspaper: advertising dollars spent on Newspaper

Target: Sales budget in thousands of dollars



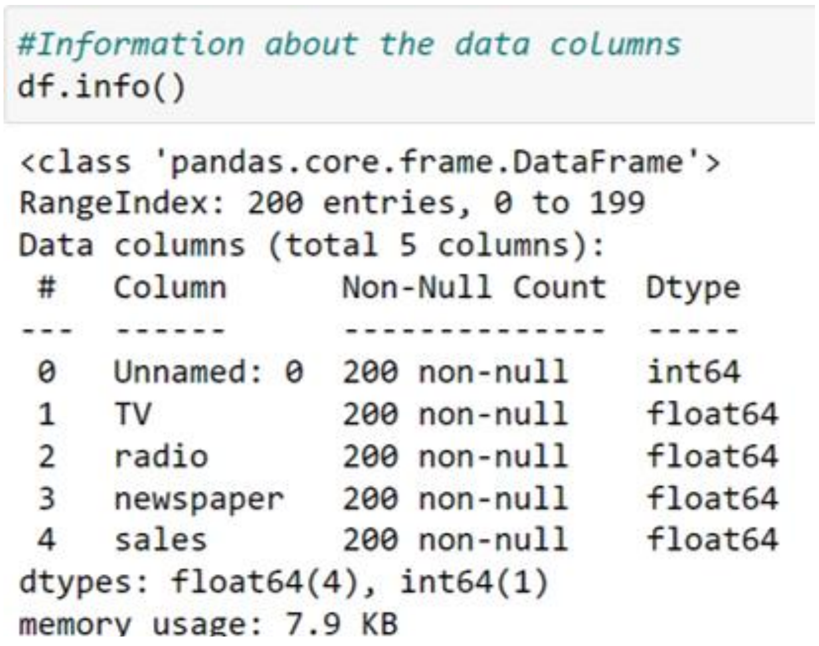
```
#Loading dataset
df=pd.read_csv('https://raw.githubusercontent.com/dsrscientist/DSData/master/Advertising.csv')

#Let's check how the data is distributed
df.head()
```



	Unnamed: 0	TV	radio	newspaper	sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9

Observation:

- By, reviewing Data Set at initial stage, found continous datatypes; Target variable data is sales which seems to represent regression problem.
- Each market is an observation, but each column is the amount spent on TV, radio, newspaper advertising.
- Each column corresponds to the values of advertising budget for different media.



```
#Information about the data columns
df.info()
```

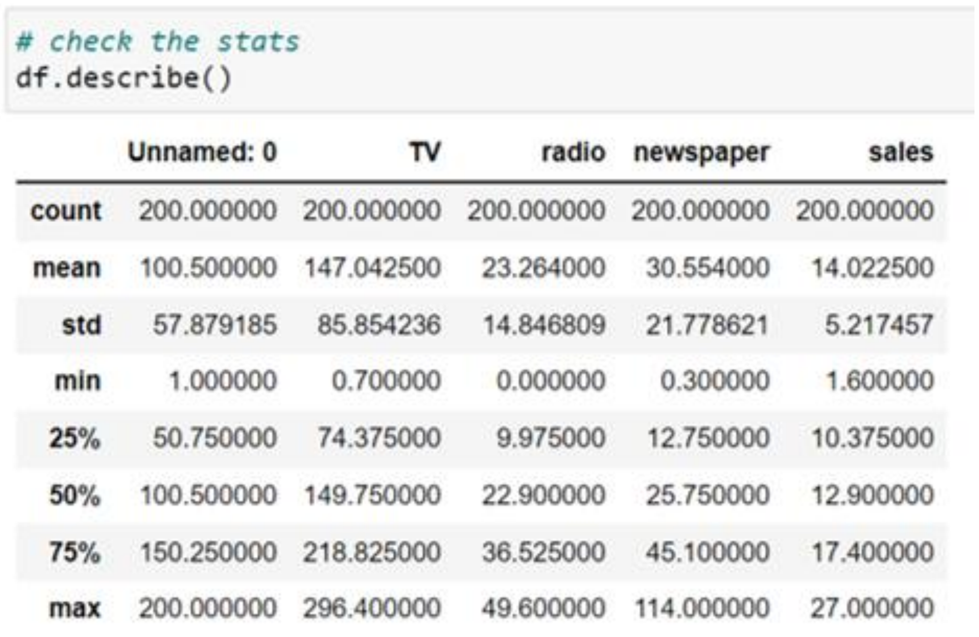
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Unnamed: 0    200 non-null    int64  
 1   TV            200 non-null    float64
 2   radio         200 non-null    float64
 3   newspaper     200 non-null    float64
 4   sales         200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
```

Observation:
There are 200 Rows and 5 Columns in DataSet. TV, radio, newspaper, sales are floating data columns; unnamed: 0 is integer data type columns and the amazing part is there are no missing values in our Dataset.

EXPLORATORY DATA ANALYSIS:

Exploratory Data Analysis, or EDA, is an important step in any Data Analysis or Data Science project. EDA is the process of investigating the dataset to discover patterns, and anomalies (outliers), and form hypotheses based on our understanding of the dataset.

Let’s know more about our dataset:



```
# check the stats
df.describe()
```

	Unnamed: 0	TV	radio	newspaper	sales
count	200.000000	200.000000	200.000000	200.000000	200.000000
mean	100.500000	147.042500	23.264000	30.554000	14.022500
std	57.879185	85.854236	14.846809	21.778621	5.217457
min	1.000000	0.700000	0.000000	0.300000	1.600000
25%	50.750000	74.375000	9.975000	12.750000	10.375000
50%	100.500000	149.750000	22.900000	25.750000	12.900000
75%	150.250000	218.825000	36.525000	45.100000	17.400000
max	200.000000	296.400000	49.600000	114.000000	27.000000

Observation:
We can find some skewness in the datset, let's visualize and confirm the same.

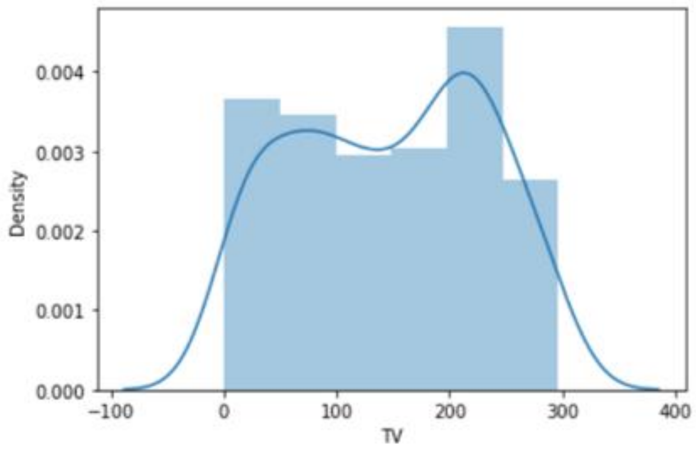
DATA VISUALIZATION:

Data visualization is the presentation of data in a pictorial or graphical format. It enables decision makers to see analytics presented visually, so they can grasp difficult concepts or identify new patterns.

Let's plot and explore our dataset!

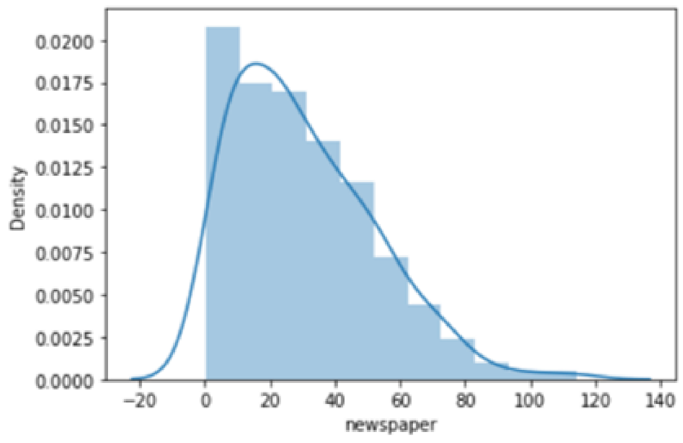
```
sns.distplot(df['TV'])
```

```
<AxesSubplot:xlabel='TV', ylabel='Density'>
```



```
sns.distplot(df['newspaper'])
```

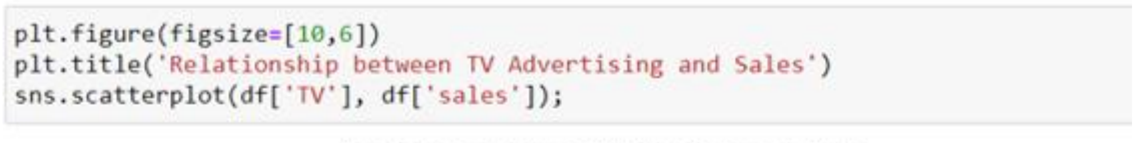
```
<AxesSubplot:xlabel='newspaper', ylabel='Density'>
```



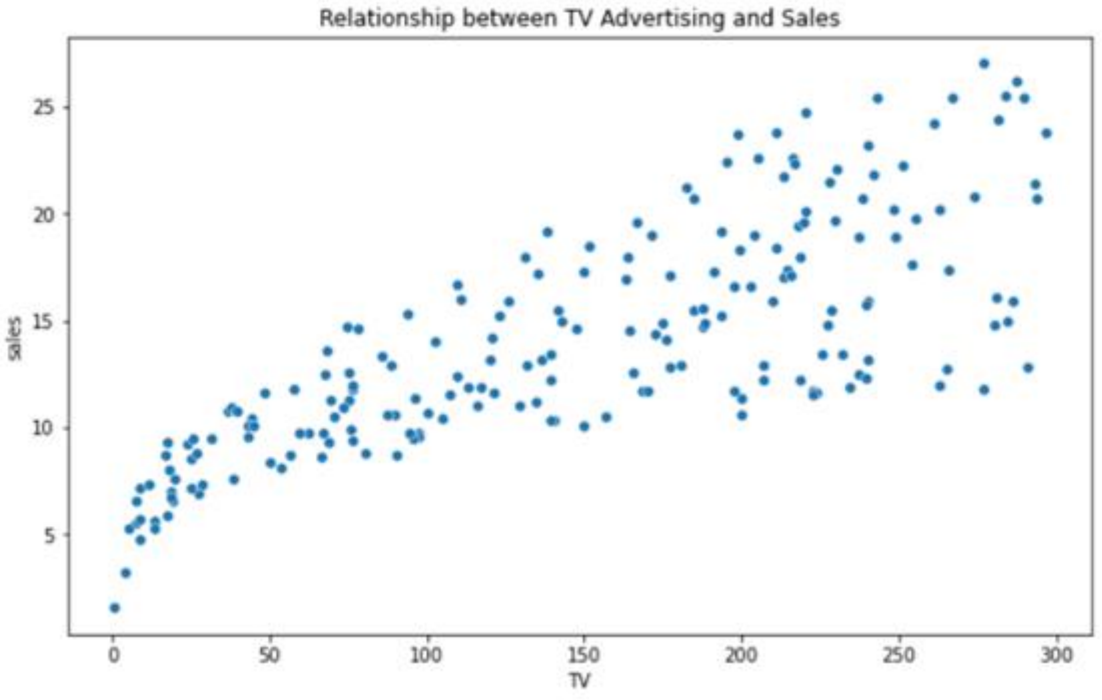
Observation:

Using distplot, found there are skewness in newspaper column.

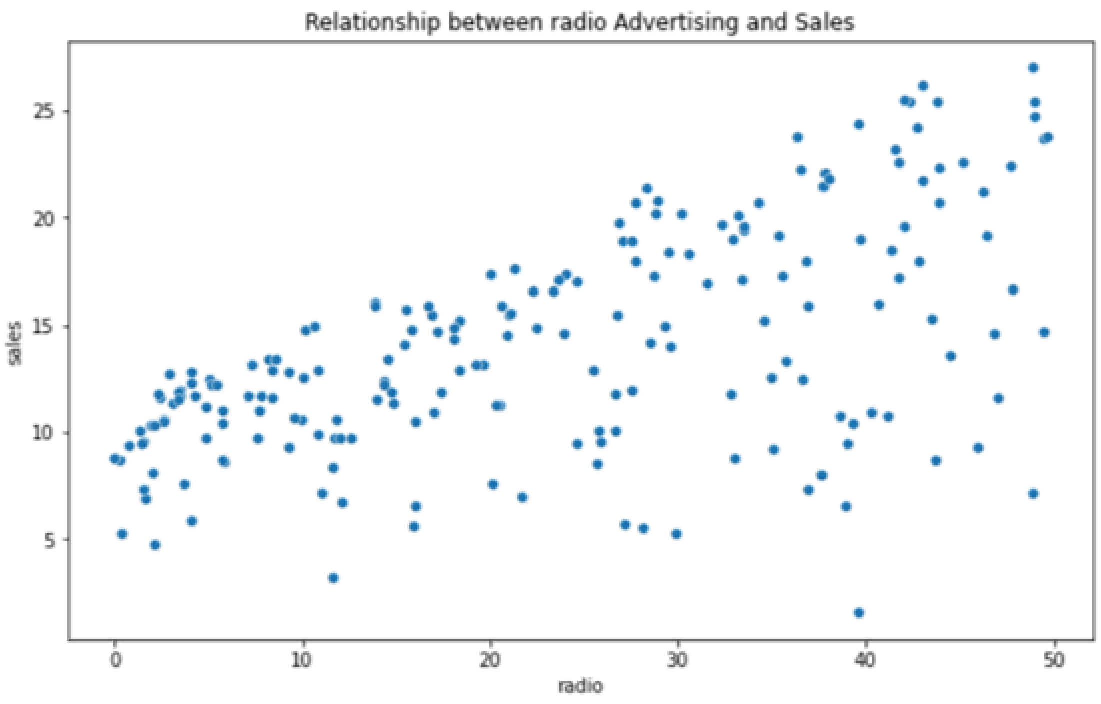
Now, let’s check the relationship between Features and the Target Variable



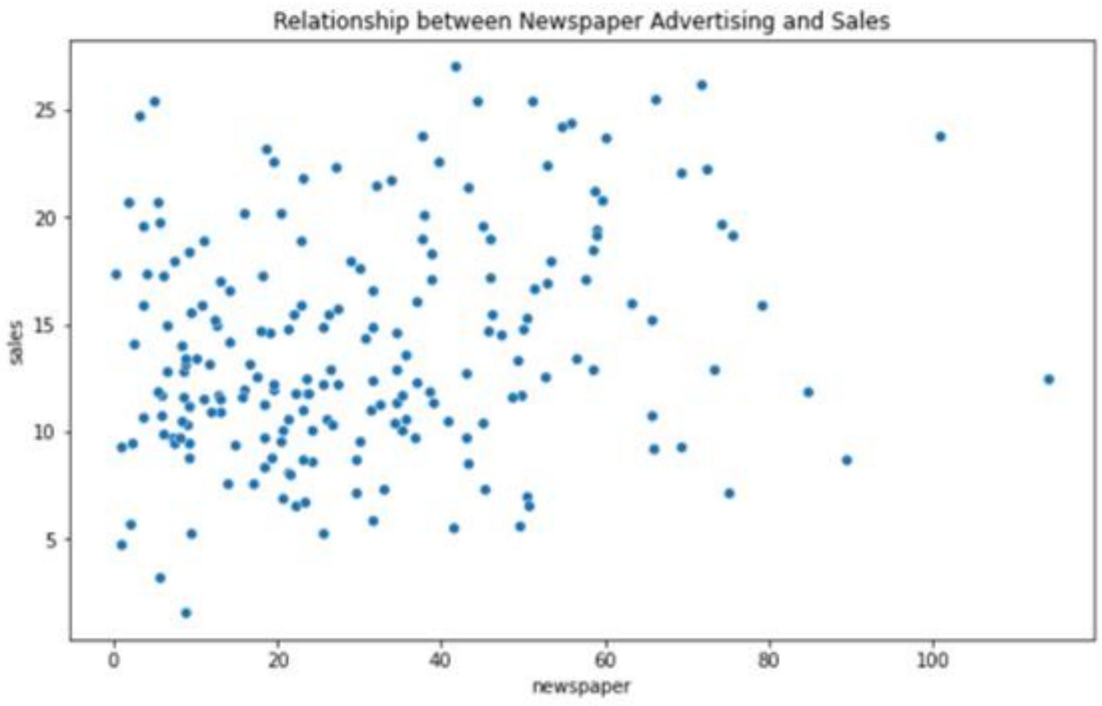
```
plt.figure(figsize=[10,6])
plt.title('Relationship between TV Advertising and Sales')
sns.scatterplot(df['TV'], df['sales']);
```



```
plt.figure(figsize=[10,6])
plt.title('Relationship between radio Advertising and Sales')
sns.scatterplot(df['radio'], df['sales']);
```

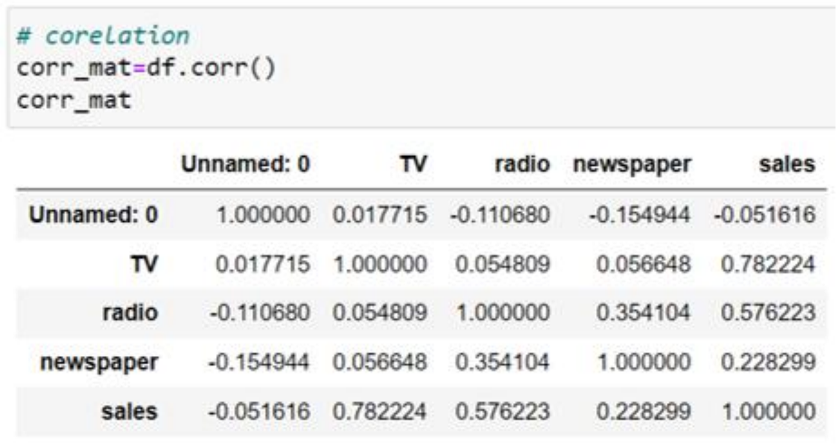


```
plt.figure(figsize=[10,6])
plt.title('Relationship between Newspaper Advertising and Sales')
sns.scatterplot(df['newspaper'], df['sales']);
```



Observation:

- ❖ TV vs. Sales - Strong positive linear relationship i.e., as TV advertising is increased, sales are also increased, although, sometimes increasing the TV budget didn't increase sales.
- ❖ The relationship between these two variables is approximately linear.
- ❖ Radio vs. Sales - Weak positive relationship (comparatively) i.e., similar to TV advertising, as radio advertising is increased, sales are also increased. But Radio advertising is less effective than TV.
- ❖ Newspaper vs. Sales - no relationship (we can discard such kind of variables which do not have a linear relationship) i.e., newspaper advertising has high effect on sales positively at initial. However, as advertising is increased, it has low effect on sales and almost nil at max.

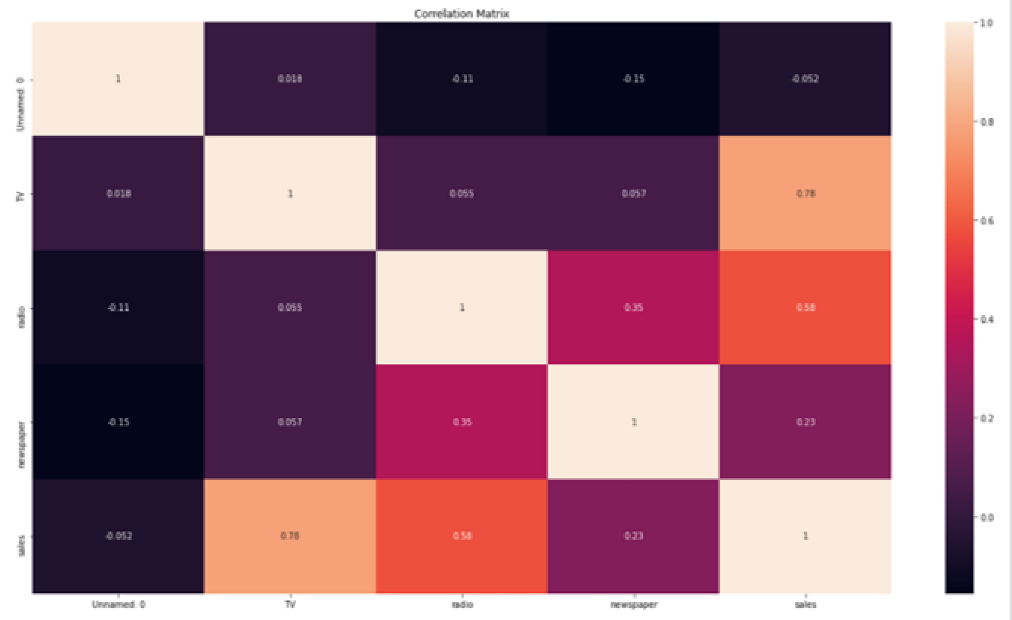


```
# corelation
corr_mat=df.corr()
corr_mat
```

	Unnamed: 0	TV	radio	newspaper	sales
Unnamed: 0	1.000000	0.017715	-0.110680	-0.154944	-0.051616
TV	0.017715	1.000000	0.054809	0.056648	0.782224
radio	-0.110680	0.054809	1.000000	0.354104	0.576223
newspaper	-0.154944	0.056648	0.354104	1.000000	0.228299
sales	-0.051616	0.782224	0.576223	0.228299	1.000000

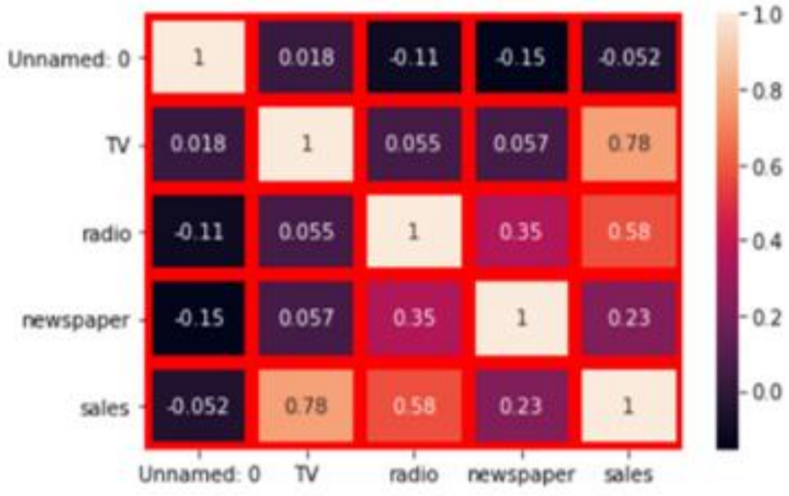
TV advertising has higher positive impact on sales and followed by radio. So, TV and radio will have higher correlation with sales.

```
plt.figure(figsize=[22,12])
sns.heatmap(corr_mat,annot=True)
plt.title("Correlation Matrix")
plt.show
```



```
sns.heatmap(corr_mat,annot=True,linewidths=6,linecolor='r')
```

```
<AxesSubplot:>
```



```
corr_mat["sales"].sort_values(ascending=False)
```

```
sales         1.000000
TV            0.782224
radio         0.576223
newspaper     0.228299
Unnamed: 0   -0.051616
Name: sales, dtype: float64
```

+1: positively correlated

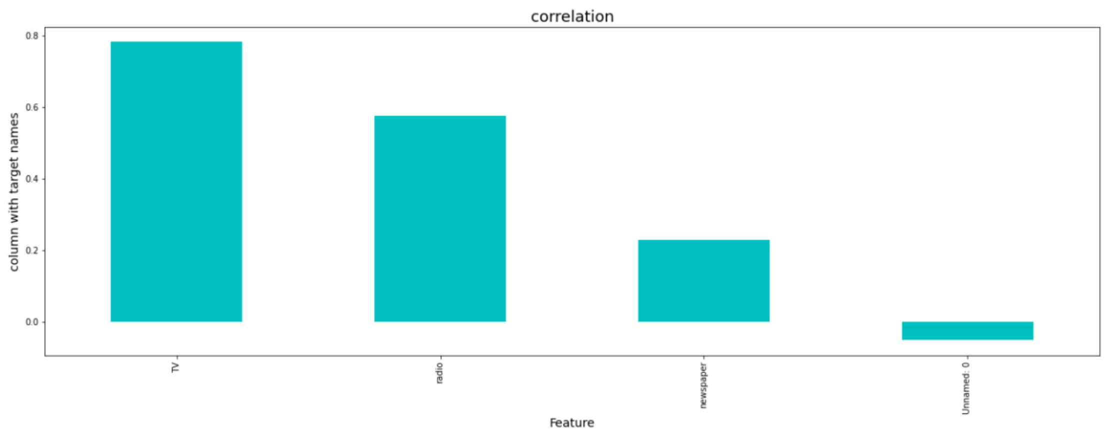
-1: negatively correlated

0: no correlation

Observation:

- We can see that there is no multi collinearity among independent/predictor variables
- TV and Radio are highly correlated with sales data which coincides with the graphical data analysis done in above steps.

```
In [25]: plt.figure(figsize=(22,7))
         df.corr()['sales'].sort_values(ascending=False).drop(['sales']).plot(kind='bar',color='c')
         plt.xlabel('Feature',fontsize=14)
         plt.ylabel('column with target names',fontsize=14)
         plt.title('correlation',fontsize=18)
         plt.show()
```



Observation:

TV Advertising increases the sales high which is followed by Radio Advertising.

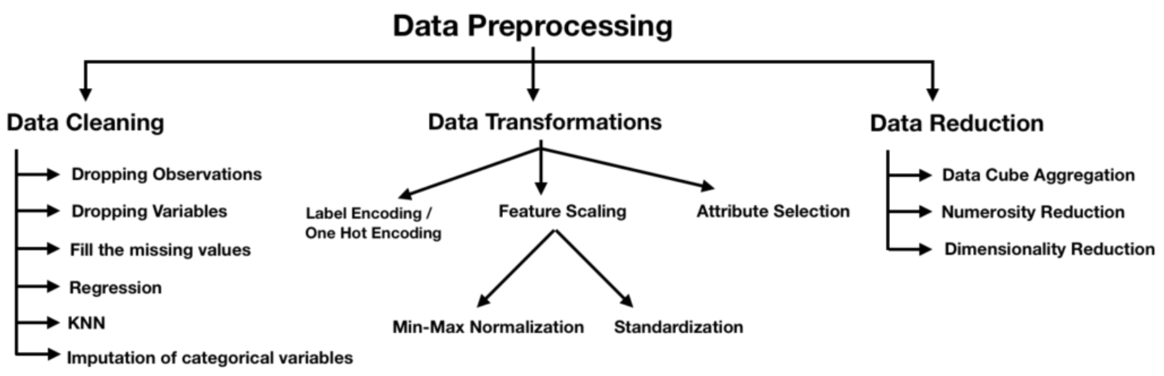
EDA REMARKS:

TV, Newspaper and radio advertising affect the target variable Sales. From the heatmap, we can observe that ‘Sales’ and ‘TV’ have a higher correlation as compared to others because it shows a linear pattern as well as giving high correlation.

So, as we saw that we have done a complete EDA process, getting data insights, feature engineering, and data visualization as well so after all these steps one can go for the prediction using machine learning model-making steps.

Usually, we go for feature engineering or feature selection steps after EDA. But we have fewer features and emphasis on actually using the model. So we are moving forward towards the next steps.

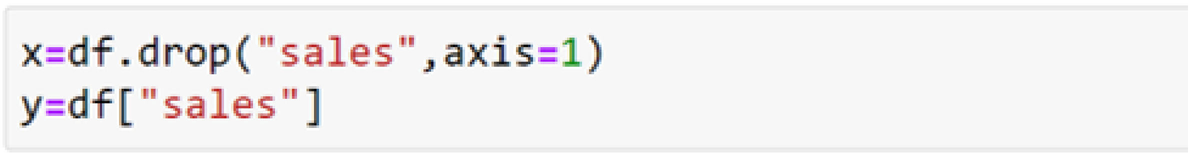
PREPROCESSING PIPELINE:



Data pre-processing is a data mining technique which is used to transform raw data into a useful format.

Let’s start building our model, firstly, we will seperate Features and Label Column.

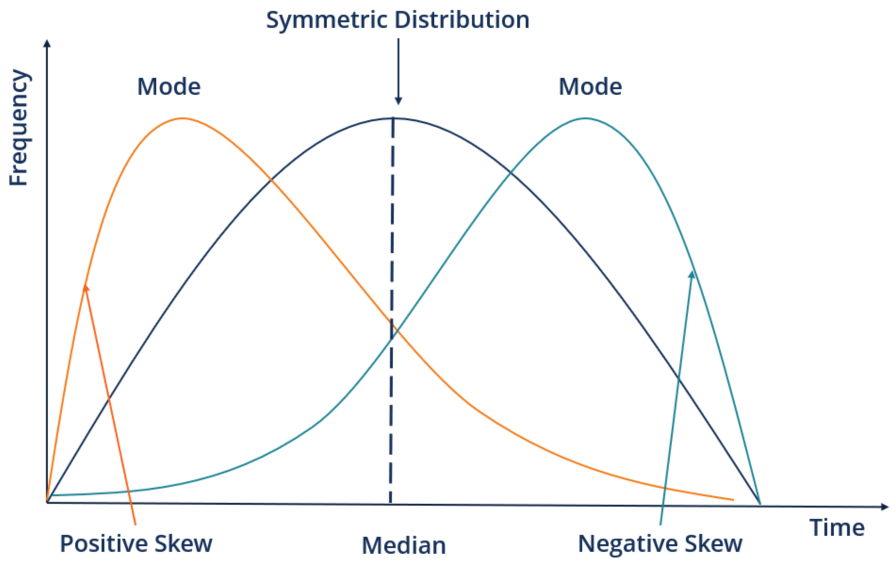
Feature and Label Seperation



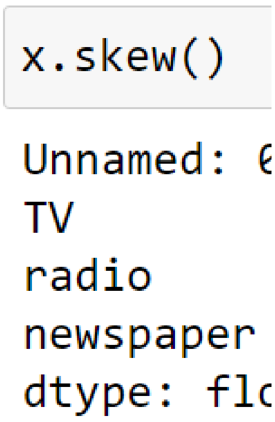
```
x=df.drop("sales",axis=1)
y=df["sales"]
```

As we have seen in the visualization, that “Newspaper” column contains skewness, let’s check the same.

Skewness:



It refers to a distortion or asymmetry that deviates from the symmetrical bell curve, or normal distribution, in a set of data. If the curve is shifted to the left or to the right, it is said to be skewed.



```
x.skew()
```

```
Unnamed: 6
TV
radio
newspaper
dtype: flc
```

Observation:

After checking for the Skewness, we found that newspaper has more skewness as predicted by distplot analysis.

We will now perform “Power Transform” method to remove the skewness.

```
from sklearn.preprocessing import power_transform
df_skew=power_transform(x)

df_skew=pd.DataFrame(df_skew,columns=x.columns)
```

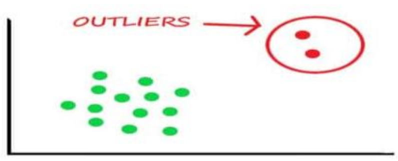
```
# Removing Skewness through Power Transforms
df_skew.skew()
```

```
Unnamed: 0    -0.268270
TV            -0.315199
radio         -0.242968
newspaper     -0.077942
dtype: float64
```

Observation:

Rechecking the skewness after power transforms, we found that skewness of all columns are within range of -0.5 to +0.5

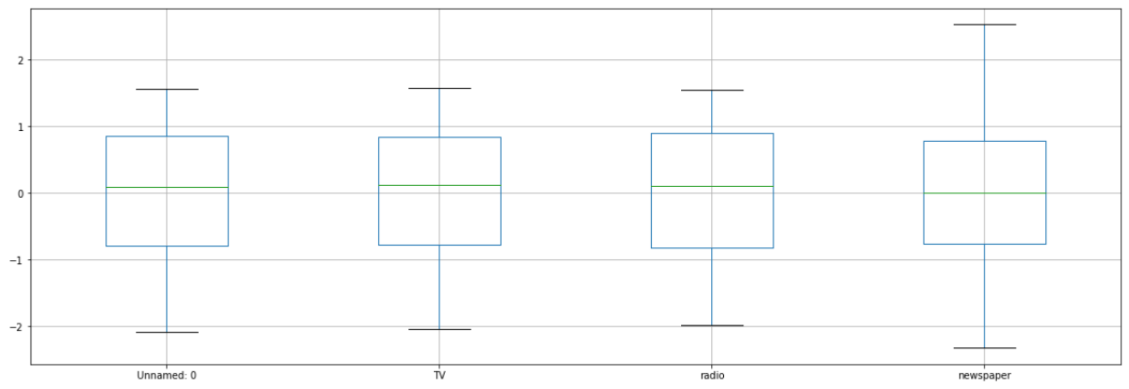
OUTLIER DETECTION:



Outliers are observations in a dataset that don't fit in some way. An outlier is a data point that differs significantly from other observations. They may be due to variability in the measurement or may indicate experimental errors.

Outlier Detection

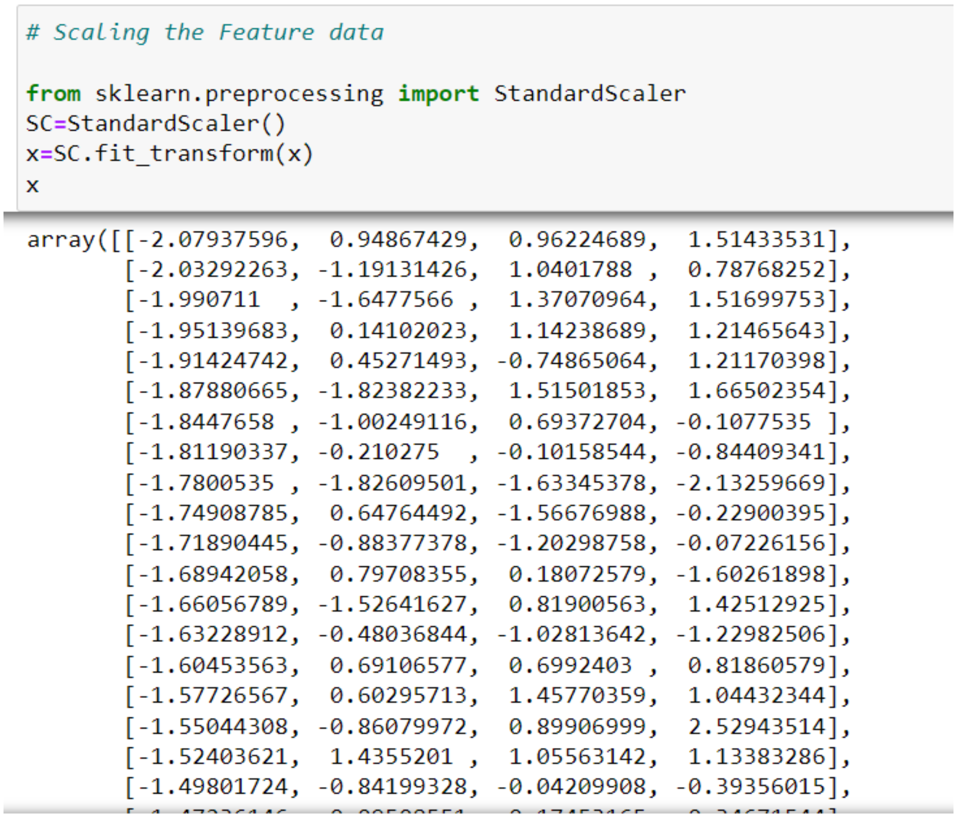
```
x.boxplot(figsize=[20,8])
plt.subplots_adjust(bottom=0.25)
plt.show()
```



Checking for Outliers through box plot graphs and found no outliers;

Next step is Data Standardization,

DATA STANDARDIZATION: It is the process of bringing data into a uniform format that allows analysts and others to research, analyze, and utilize the data. In statistics, standardization refers to the process of putting different variables on the same scale in order to compare scores between different types of variables.



```python
# Scaling the Feature data

from sklearn.preprocessing import StandardScaler
SC=StandardScaler()
x=SC.fit_transform(x)
x
```

```
array([[-2.07937596,  0.94867429,  0.96224689,  1.51433531],
       [-2.03292263, -1.19131426,  1.0401788 ,  0.78768252],
       [-1.990711  , -1.6477566 ,  1.37070964,  1.51699753],
       [-1.95139683,  0.14102023,  1.14238689,  1.21465643],
       [-1.91424742,  0.45271493, -0.74865064,  1.21170398],
       [-1.87880665, -1.82382233,  1.51501853,  1.66502354],
       [-1.8447658 , -1.00249116,  0.69372704, -0.1077535 ],
       [-1.81190337, -0.210275  , -0.10158544, -0.84409341],
       [-1.7800535 , -1.82609501, -1.63345378, -2.13259669],
       [-1.74908785,  0.64764492, -1.56676988, -0.22900395],
       [-1.71890445, -0.88377378, -1.20298758, -0.07226156],
       [-1.68942058,  0.79708355,  0.18072579, -1.60261898],
       [-1.66056789, -1.52641627,  0.81900563,  1.42512925],
       [-1.63228912, -0.48036844, -1.02813642, -1.22982506],
       [-1.60453563,  0.69106577,  0.6992403 ,  0.81860579],
       [-1.57726567,  0.60295713,  1.45770359,  1.04432344],
       [-1.55044308, -0.86079972,  0.89906999,  2.52943514],
       [-1.52403621,  1.4355201 ,  1.05563142,  1.13383286],
       [-1.49801724, -0.84199328, -0.04209908, -0.39356015],
```

BUILDING MACHINE LEARNING MODEL:

Supervised learning — is a machine learning task that establishes the mathematical relationship between input X and output Y variables. Such X, Y pair constitutes the labeled data that are used for model building in an effort to learn how to predict the output from the input.

Let’s first understand the theory by Linear Model and then we will do Python Implementation:

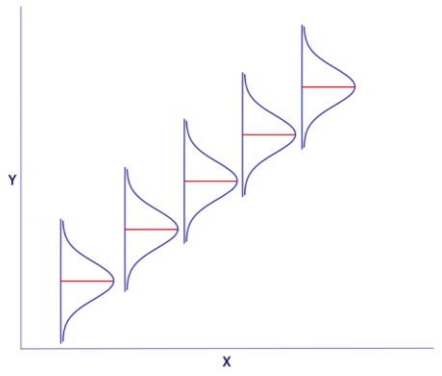
Linear Relationship

A perfect linear relationship between an independent variable x and dependent variable y has the mathematical form:

$$y = \beta 0 + \beta 1 \, x.$$

iop β0 is called the -intercept and y is called the slope.

Simple Linear Regression

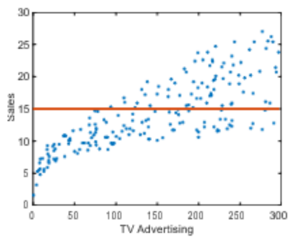


Simple linear regression is used to model the relationship between two continuous variables. Often, the objective is to predict the value of an output variable (or response) based on the value of an input (or predictor) variable.

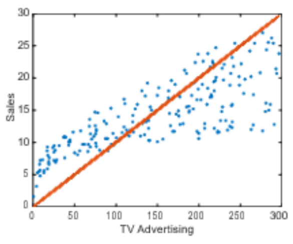
Recall Notation

- m=number of training examples
- x =“input” variable/features
- y =“output” variable/“target” variable
- (x (i), y (i)) the i th training example
- x (1) = 230.1, y (1) = 22.1, x (2) = 44.5, y (2) = 10.4

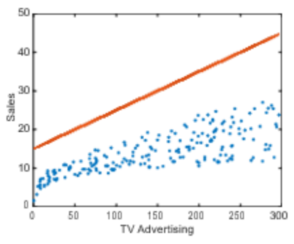
- Prediction: $y = h_\theta(x) = \theta_0 + \theta_1 x$
- θ_0, θ_1 are (unknown) parameters



$\theta_0 = 15, \theta_1 = 0$



$\theta_0 = 0, \theta_1 = 0.1$



$\theta_0 = 15, \theta_1 = 0.1$

The best fit line for the data points is nothing but the line that best expresses the data point relationship. Residual Sum of Squares (RSS) is computed to find the best-fit line, such line will have the lowest value of RSS.

$$\text{RSS}=\sum_{k=1}^{n}(Actual - Predicted)^2$$

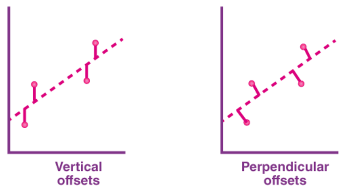
In simple linear regression, if the coefficient of x is positive, it can be concluded that the relationship between independent variable and dependent variables is positive.

i.e., in $Y_i = ß_0 + ß_1 X_i + Ɛ_i$

if $ß_1 > 0$ the relationship is positive.

if $ß_1 < 0$ the relationship is negative.

Least Squares



The Least Squares approach is to find the y-intercept β0 and slope β1of the straight line that is closest to as many of the points as possible.

The least-squares explain that the curve that best fits is represented by the property that the sum of squares of all the deviations from given values must be minimum, i.e:

$$S = \sum_{i=1}^{n} d_i^2$$

$$S = \sum_{i=1}^{n} [y_i - f_{x_i}]^2$$

$$S = d_1^2 + d_2^2 + d_3^2 + \cdots + d_n^2$$

Sum = Minimum Quantity

Suppose when we have to determine the equation of line of best fit for the given data, then we first use the following formula.

The equation of least square line is given by Y = a + bX
Normal equation for 'a':

$$\sum Y = na + b\sum X$$

Normal equation for 'b':

$$\sum XY = a\sum X + b\sum X2$$

Solving these two normal equations we can get the required trend line equation.
Thus, we can get the line of best fit with formula y = ax + b

The estimated linear regression of sales on newspaper is:

$$y_i = 12.35 + 0.05x_i,$$

where y_i is sales in i^{th} the market and x_i is the dollar amount spent on newspaper advertising in the i^{th} market.

- ❖ The slope $\hat{\beta}_1$ is the amount of change in for a unit change in x.
- ❖ Sales increase by 0.05 for each dollar spent on advertising.
- ❖ The intercept $\hat{\beta}_0$ is the average of y when $x_{i=}0$.
- ❖ The average sales are 12.35 when the amount spent on advertising is zero.

Prediction using a Linear Regression Model

Prediction is determined by the value of the variable. Accuracy and fitness is measured by loss, R square, adjusted R square etc. After a linear regression model is estimated from data it can be used to calculate predicted values using the regression equation

$$\hat{y}_i = \beta_0 + \beta_1 x_i.$$

$\hat{y}_i$ is the predicted value of the ith response y_i.

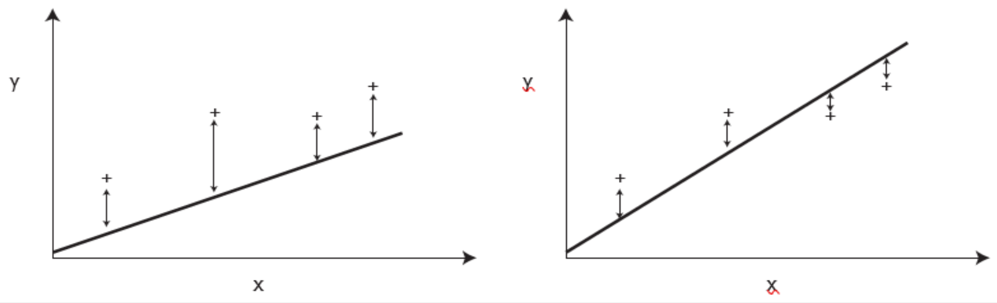
The ith residual is:

$$e_i = y_i - \hat{y}_i$$

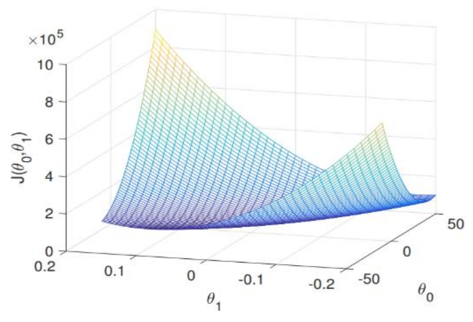
Cost Function: How to choose model parameters ϑ?

- Prediction: $y = h_\vartheta(x) = \theta_0 + \theta_1 x$
- Idea: Choose θ_0 and θ_1 so that $h_\vartheta(x^{(i)})$ is close to $y^{(i)}$ for each of our training examples $(x^{(i)}, y^{(i)})$, $i = 1, \ldots, m$.
- Least squares case: select the values for θ_0 and θ_1 that minimise cost function:

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\vartheta(x^{(i)}) - y^{(i)})^2$$



Gradient Descent



It is an optimization algorithm used for minimizing the cost function in various machine learning algorithms.

Linear Regression With Multiple Variables:

Advertising Dataset:

TV x_1	Radio x_2	Newspaper x_3	Sales y
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	9.3
.	.	.	.

- n=number of features (3 in this example)
- $(x^{(i)}, y^{(i)})$ the ith training example e.g.
  $$x^{(1)} = [230.1, 37.8, 69.2]^T = \begin{bmatrix} 230.1 \\ 37.8 \\ 69.2 \end{bmatrix}$$
- $x_j^{(i)}$ is feature j in the ith training example, e.g. $x_2^{(1)} = 37.8$

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$
e.g. $\underbrace{h_\theta(x)}_{Sales} = 15 + 0.1 \underbrace{x_1}_{TV} - 5 \underbrace{x_2}_{Radio} + 10 \underbrace{x_3}_{Newspaper}$

More generally, when have n features:

- For convenience, define $x_0 = 1$
  i.e. $x_0^{(1)} = 1$, $x_0^{(2)} = 1$ etc
- Feature vector $x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$
- Parameter vector $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$
- $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \theta^T x$

- Hypothesis: $h_\theta(x) = \theta^T x$
- Parameters: θ
- Cost Function: $J(\theta) = J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$
- Learn Parameters: Select θ that minimises $J(\theta)$. E.g. can find θ using gradient descent.

Gradient Descent with Multiple Variables:

For $J(\theta) = \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$ with $h_\theta(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$:

- $\frac{\partial}{\partial\theta_0} J(\theta) = \frac{2}{m} \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})$
- $\frac{\partial}{\partial\theta_1} J(\theta) = \frac{2}{m} \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_1^{(i)}$
- $\frac{\partial}{\partial\theta_j} J(\theta) = \frac{2}{m} \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$

So gradient descent algorithm is:

- Start with some θ
- Repeat:

```
        for $j$=0 to $n$ $\{tempj := \theta_j - \frac{2\alpha}{m} \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\}$
        for $j$=0 to $n$ $\{\theta_j := tempj\}$
```

So, it’s time for Python implementation, Let’s begin with spliting our Dataset into Train and Test Data;

Train-Test-Split:

The train-test split is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm. The procedure involves taking a dataset and dividing it into two subsets.

```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.20, random_state=100)
```

Now let’s prepare the data to fit into a machine learning model and then we will use different regression algorithms to train a sales prediction model using Python.

Ridge Regression Model Build:

```
r = Ridge()
r.fit(x_train,y_train)
predr = r.predict(x_test)
print(r2_score(y_test,predr))
print(mean_squared_error(y_test,predr))
print(np.sqrt(mean_squared_error(y_test,predr)))
```

```
0.924807681691509
1.597907726805838
1.264083749917638
```

```
scr = cross_val_score(r,x,y,cv=5)
print('Cross validation score for Ridge Regression Model is', scr.mean())
```

```
Cross validation score for Ridge Regression Model is 0.8953866482738437
```

Ridge Regression Model Difference is 92.48 - 89.53 = 2.95

Lasso Regression Model Build:

```
l = Lasso()
l.fit(x_train,y_train)
predl = l.predict(x_test)
print(r2_score(y_test,predl))
print(mean_squared_error(y_test,predl))
print(np.sqrt(mean_squared_error(y_test,predl)))
```

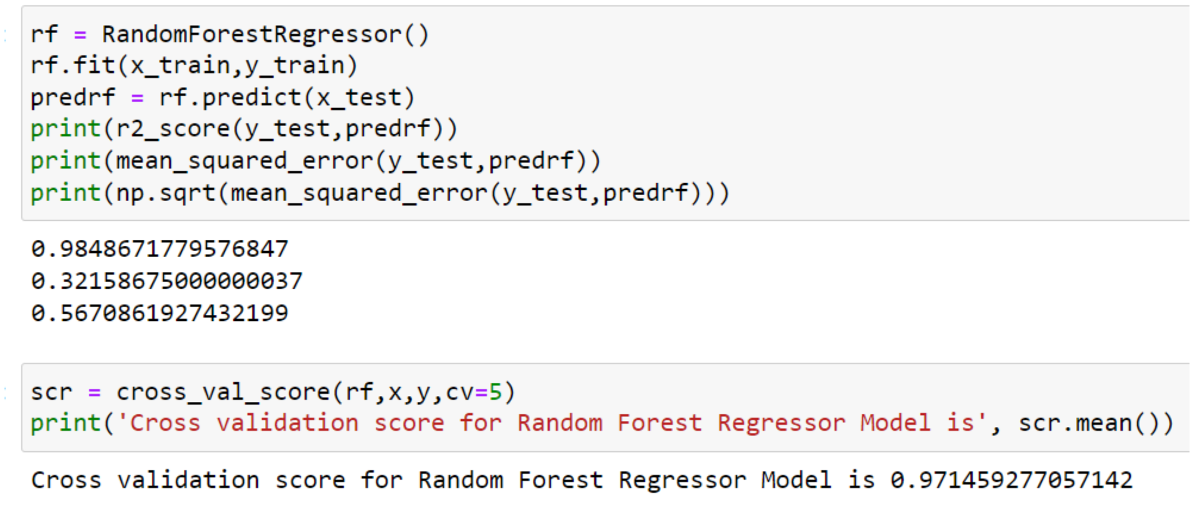
```
0.8285851353014799
3.6427276476221095
1.908593106877972
```

```
scr = cross_val_score(l,x,y,cv=5)
print('Cross validation score for Lasso Regression Model is ', scr.mean())
```

```
Cross validation score for Lasso Regression Model is  0.8270960357806555
```

Lasso Regression Model Difference is 82.85 - 82.70 = 0.15

Random Forest Regressor Model Build:



```
rf = RandomForestRegressor()
rf.fit(x_train,y_train)
predrf = rf.predict(x_test)
print(r2_score(y_test,predrf))
print(mean_squared_error(y_test,predrf))
print(np.sqrt(mean_squared_error(y_test,predrf)))
```

```
0.9848671779576847
0.32158675000000037
0.5670861927432199
```

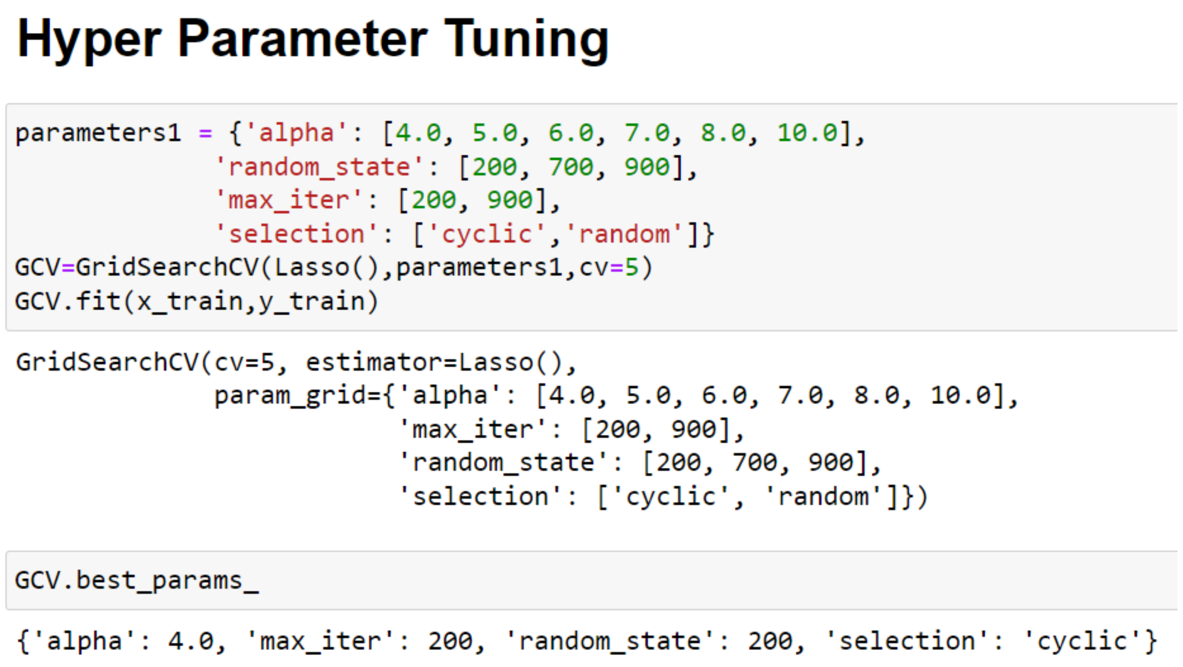
```
scr = cross_val_score(rf,x,y,cv=5)
print('Cross validation score for Random Forest Regressor Model is', scr.mean())
```

```
Cross validation score for Random Forest Regressor Model is 0.971459277057142
```

Random Forest Regressor Model Difference is 98.53 - 97.33 = 1.20

After training different models, we found that Lasso Regression Model has the least difference in r2_score so it is the best model. We will now improve the accuracy of the model with Hyper parameter tuning.

Hyperparameter tuning: It is choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a model argument whose value is set before the learning process begins. The key to machine learning algorithms is hyperparameter tuning.



Hyper Parameter Tuning

```
parameters1 = {'alpha': [4.0, 5.0, 6.0, 7.0, 8.0, 10.0],
               'random_state': [200, 700, 900],
               'max_iter': [200, 900],
               'selection': ['cyclic','random']}
GCV=GridSearchCV(Lasso(),parameters1,cv=5)
GCV.fit(x_train,y_train)
```

```
GridSearchCV(cv=5, estimator=Lasso(),
             param_grid={'alpha': [4.0, 5.0, 6.0, 7.0, 8.0, 10.0],
                         'max_iter': [200, 900],
                         'random_state': [200, 700, 900],
                         'selection': ['cyclic', 'random']})
```

```
GCV.best_params_
```

```
{'alpha': 4.0, 'max_iter': 200, 'random_state': 200, 'selection': 'cyclic'}
```

Obtaining the Best Parameters for Hyper parameters tuning

```
mod1=Lasso(alpha=4.0,random_state=200,selection='cyclic',max_iter=200)

mod1.fit(x_train,y_train)
pred=mod1.predict(x_test)
print(r2_score(y_test,predrf)*100)
print(mean_squared_error(y_test,predrf))
print(np.sqrt(mean_squared_error(y_test,predrf)))
```

```
98.48671779576847
0.32158675000000037
0.5670861927432199
```

Observation:
After Hyper parameter Tuning, Lasso Model r2_score is increased from 83% to 98.53%
Now, the Final Step is to save the model for Future Predictions:

Save the model

```
import joblib
joblib.dump(mod1,"Advertising_Sales_Lasso.pkl")
```

```
['Advertising_Sales_Lasso.pkl']
```

CONCLUDING REMARKS:

- ❖ TV, Newspaper and radio advertising affect the target variable Sales. The order of correlation with sales given by TV Adv> Radio Adv> Newspaper Adv.
- ❖ The linear model created with TV, Newspaper and radio as input variable and Sales as Target variable gave the higher r2score.
- ❖ The linear model created with TV as input variable and Sales as Target variable gave the second highest r2score.
- ❖ The linear models created with Radio and Newspaper as input variables and Sales as Target variable has very less r2 score.
- ❖ Therefore, in order to create a suitable model which approximates the scenario correctly, we need to use the combination of TV, Radio and Newspaper as target variable.
- ❖ It is clear from the model and correlation matrix that Amount spend on TV ads directly impact the Sales of a particular product.
- ❖ We can conclude from the R2 value that only input variable TV advertisements provide a satisfactory result.

About the Author: Pursuing Post Graduate Diploma in Data Science, Machine Learning and Neural Network. I am an aspiring Data Scientist whose purpose is to learn in detail all the concepts needed for Data Science. I am passionate about Data Science and have skills that help me derive valuable insights from data, such as Data Manipulation, Data Visualization, Data Analysis, EDA, and Machine Learning.