FUTURE SALES PREDICTION – PROJECT DOCUMENTATION

ABSTRACT

Sales forecasting is the process of predicting future sales. It is the vital part of the financial planning of the business. Most of the companies heavily depend on the future prediction of the sales. Accurate sales forecasting empower the organizations to make informed business decisions and it will help to predict the short-term and long-term performances. A precise forecasting can avoid overestimating or underestimating of the future sales, which may leads to great loss to companies. The past and current sales statistics is used to estimate the future performance. But it is difficult to deal with accuracy of sales forecasting by traditional forecasting. For this purpose, various machine learning techniques have been discovered. In this work, we have taken Black Friday dataset and made a detailed analysis over the dataset. Here, we have implemented the different machine learning techniques with different metrics. By analysing the performance, we have trying to suggest the suitable predictive algorithm to our problem statement.

INTRODUCTION

Sales play a key role in the business. At the company level, sales forecasting is the major part of the business plan and significant inputs for decision-making activities. It is essential for organizations to produce the required quantity at the 7 specified time. For that, sales forecasting will gives the idea about how an organization should manage its budgeting, workforce and resources. This forecasting helps the business management to determine how much products should be manufacture, how much revenue can be expected and what could be the requirement of employees, investment and equipment. By analyzing the future trends and needs, Sales forecasting helps to improve the business growth.

PROBLEM STATEMENT

The problem is to develop a predictive model that uses historical sales data to forecast future sales for a retail company. The objective is to create a tool that enables the company to optimize inventory management and make informed business decisions based on datadriven sales predictions. This project involves data preprocessing, feature engineering, model selection, training, and evaluation.

Phase 1

Design Thinking :

- ✓ **Data Source:** Utilize a dataset containing historical sales data, including features like date, product ID, store ID, and sales quantity.
- ✓ **Data Preprocessing:** Clean and preprocess the data, handle missing values, and convert categorical features into numerical representations.
- ✓ **Feature Engineering:** Create additional features that could enhance the predictive power of the model, such as time-based features (e.g., day of the week, month).
- ✓ **Model Selection:** Choose suitable time series forecasting algorithms (e.g., ARIMA, Exponential Smoothing) for predicting future sales.
- ✓ **Model Training:** Train the selected model using the preprocessed data.
- ✓ **Evaluation:** Evaluate the model's performance using appropriate time series forecasting metrics (e.g., Mean Absolute Error, Root Mean Squared Error).

Phase 2

- ➢ Data Gathering
- ➢ DataSet Description
- ➢ Import Libraries
- ➢ Training and Testing
- ➢ Metrics used for checking accuracy

Data Gathering :

Dataset is taken from kaggle competition and can be downloaded from

https://www.kaggle.com/datasets/chakradharmattapalli/future-sales-prediction

DataSet Description :

→The dataset given here contains the data about the sales of the product. The dataset is about the advertising cost incurred by the business on various advertising platforms.

Below is the description of all the columns in the dataset:

- **TV:** Advertising cost spent in dollars for advertising on TV;
- **Radio:** Advertising cost spent in dollars for advertising on Radio;
- **Newspaper:** Advertising cost spent in dollars for advertising on Newspaper;
- **Sales:** Number of units sold;

Import Libraries :

→Let’s start the task of future sales prediction with machine learning by importing the necessary Python libraries and the dataset:

- **NumPy:** For numerical computations and working with arrays.

  ```
  import numpy as np
  ```

- **Pandas:** For data manipulation and analysis.

  ```
  import pandas as pd
  ```

- **import Linear Regression:** This allows you to use Linear Regression for tasks like regression analysis, where you want to model the relationship between a dependent variable and one or more independent variables.

  ```
  from sklearn.model_linear import LinearRegression
  ```

- **import train set: "**train set” or "training set" is a portion of a dataset that is used to train a model. This set of data is used to teach the model patterns and relationships in the data so that it can make predictions or classifications on new, unseen data.

  ```
  from sklearn.model_selection import train_test_split
  ```

Training and Testing :

→ Data is typically split into a training set and a testing set. The training set is used to build and train the predictive model, while the testing set is used to evaluate its accuracy and performance.

Typically, you might use 70-80% of the data for training and the remaining 20-30% for testing. Train the chosen model on the training data. This involves finding the model parameters that best fit the historical data.

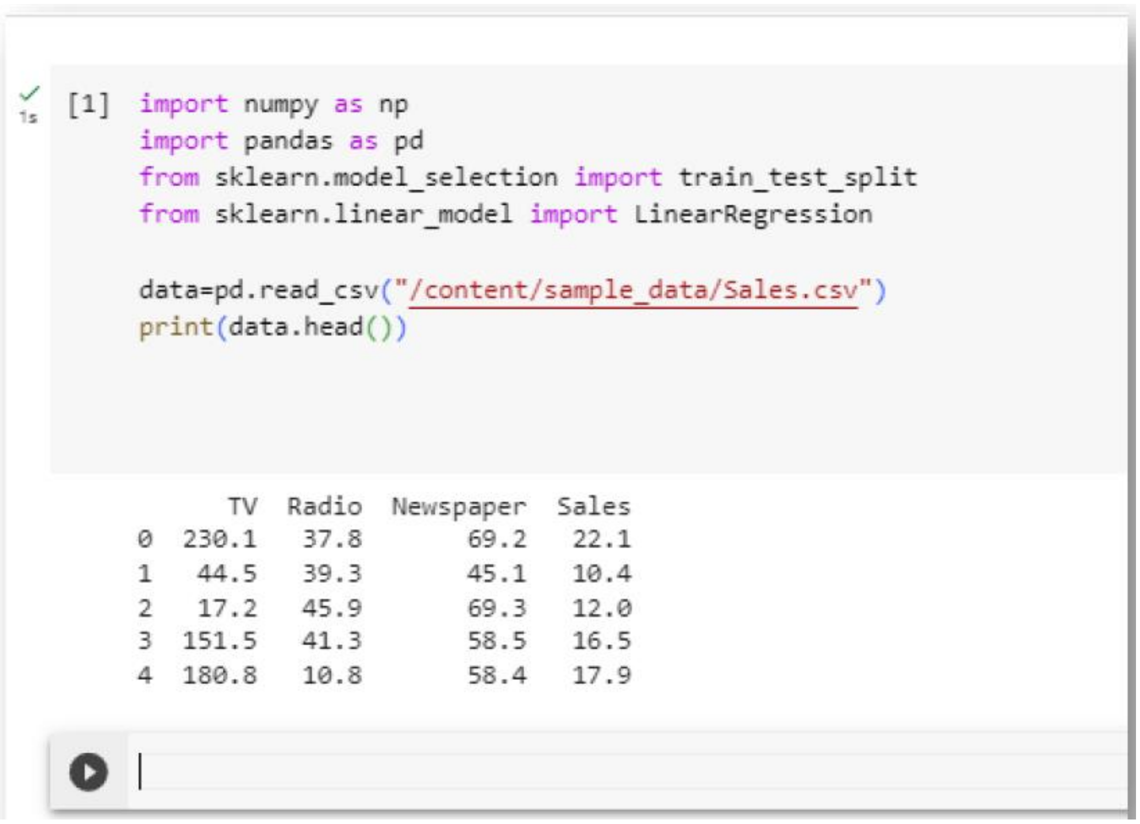
Metrics used for checking accuracy:

→ We can use the **test-market analysis** method to forecast sales for a new product or service. In this method, you launch your new product/service to a certain group of people based on their market segregation and study their results to make an accurate forecast for the full release.

Phase 3

- Handling missing value
- Encoding categorical
- Train and Test dataset
- Feature Scaling

→ Let's start the development part of future sales prediction with machine learning by importing the necessary Python libraries



```
[1] import numpy as np
    import pandas as pd
    from sklearn.model_selection import train_test_split
    from sklearn.linear_model import LinearRegression

    data=pd.read_csv("/content/sample_data/Sales.csv")
    print(data.head())
```

```
      TV  Radio  Newspaper  Sales
0  230.1   37.8       69.2   22.1
1   44.5   39.3       45.1   10.4
2   17.2   45.9       69.3   12.0
3  151.5   41.3       58.5   16.5
4  180.8   10.8       58.4   17.9
```

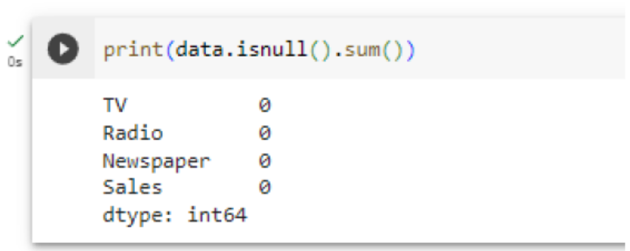
Handling missing values :

→ Missing Data is a very big problem in a real-life scenarios. Missing Data can also refer to as NA(Not Available) values in pandas. In DataFrame sometimes many datasets simply arrive with missing data, either because it exists and was not collected or it never existed.
For Example, Suppose different users being surveyed may choose not to share their income, some users may choose not to share the address in this way many datasets went missing.

Checking for missing values using isnull()

- In order to check null values in Pandas DataFrame, we use isnull() function this function return dataframe of Boolean values which are True for NaN values

→ Let's have a look at whether this dataset contains any null values or not:



```
print(data.isnull().sum())
```

```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

→ So this dataset doesn't have any null values.

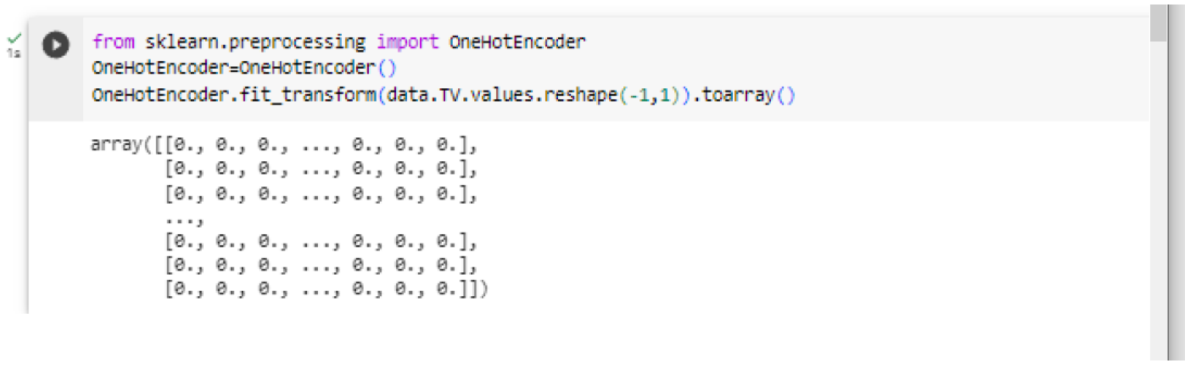
Encoding categorical :

→ Encoding categorical data is a process of converting categorical data into integer format so that the data with converted categorical values can be provided to the models to give and improve the predictions.

One-Hot Encoding

- In One-Hot Encoding, each category of any categorical variable gets a new variable. It maps each category with binary numbers (0 or 1). This type of encoding is used when the data is nominal. Newly created binary features can be considered dummy variables. After one hot encoding, the number of dummy variables depends on the number of categories presented in the data.

✓ The way to achieve this in python is illustrated below.



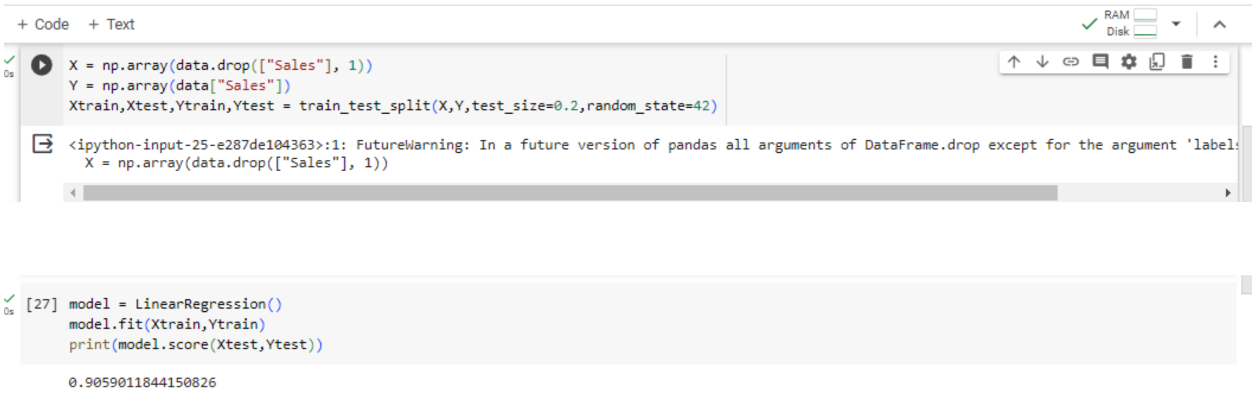
```
from sklearn.preprocessing import OneHotEncoder
OneHotEncoder=OneHotEncoder()
OneHotEncoder.fit_transform(data.TV.values.reshape(-1,1)).toarray()
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

Train and Test dataset :

→ Now in this section, I will train a machine learning model to predict the future sales of a product. But before I train the model, let's split the data into training and test sets:

✓ Now let's train the model to predict future sales:



```
X = np.array(data.drop(["Sales"], 1))
Y = np.array(data["Sales"])
Xtrain,Xtest,Ytrain,Ytest = train_test_split(X,Y,test_size=0.2,random_state=42)
```

```
<ipython-input-25-e287de104363>:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'label
  X = np.array(data.drop(["Sales"], 1))
```

```
[27] model = LinearRegression()
     model.fit(Xtrain,Ytrain)
     print(model.score(Xtest,Ytest))
```

```
0.9059011844150826
```

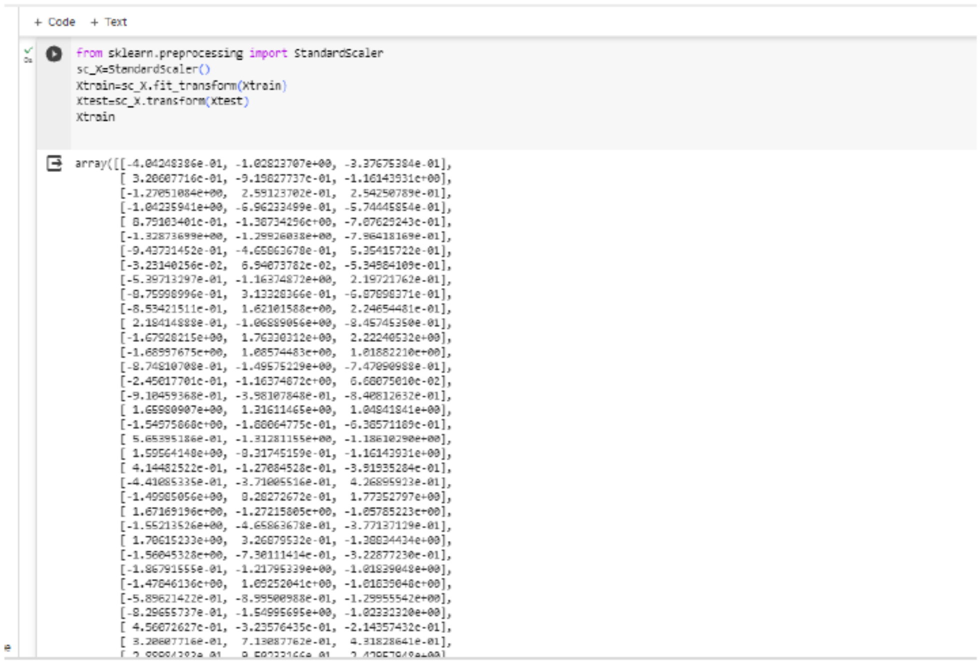
Feature Scaling :

→ Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing to handle highly varying magnitudes or values or units. If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider smaller values as the lower values, regardless of the unit of the values.

Standardization

→ This method of scaling is basically based on the central tendencies and variance of the data.

1. First, we should calculate the mean and standard deviation of the data we would like to normalize.
2. Then we are supposed to subtract the mean value from each entry and then divide the result by the standard deviation.



```python
from sklearn.preprocessing import StandardScaler
sc_X=StandardScaler()
Xtrain=sc_X.fit_transform(Xtrain)
Xtest=sc_X.transform(Xtest)
Xtrain
```

```
array([[-4.04248386e-01, -1.02823707e+00, -3.37675384e-01],
       [ 3.20607716e-01, -9.19827737e-01, -1.16143931e+00],
       [-1.27051084e+00,  2.59123702e-01,  2.54250789e-01],
       [-1.04235941e+00, -6.96233499e-01, -5.74445854e-01],
       [ 8.79103401e-01, -1.38734296e+00, -7.07629243e-01],
       [-1.32873699e+00, -1.29926038e+00, -7.96418169e-01],
       [-9.43731452e-01, -4.65863678e-01,  5.35415722e-01],
       [-3.23140256e-02,  6.94073782e-02, -5.34984109e-01],
       [-5.39713297e-01, -1.16374872e+00,  2.19721762e-01],
       [-8.75998996e-01,  3.13328366e-01, -6.87898371e-01],
       [-8.53421511e-01,  1.62101588e+00,  2.24654481e-01],
       [ 2.18414888e-01, -1.06889056e+00, -8.45745350e-01],
       [-1.67928215e+00,  1.76330312e+00,  2.22240532e+00],
       [-1.68997675e+00,  1.08574483e+00,  1.01882210e+00],
       [-8.74810708e-01, -1.49575229e+00, -7.47090988e-01],
       [-2.45017701e-01, -1.16374872e+00,  6.68075010e-02],
       [-9.10459368e-01, -3.98107848e-01, -8.40812632e-01],
       [ 1.65980907e+00,  1.31611465e+00,  1.04841841e+00],
       [-1.54975868e+00, -1.88064775e-01, -6.38571189e-01],
       [ 5.65395186e-01, -1.31281155e+00, -1.18610290e+00],
       [ 1.59564148e+00, -8.31745159e-01, -1.16143931e+00],
       [ 4.14482522e-01, -1.27084528e-01, -3.91935284e-01],
       [-4.41085335e-01, -3.71005516e-01,  4.26895923e-01],
       [-1.49985056e+00,  8.28272672e-01,  1.77352797e+00],
       [ 1.67169196e+00, -1.27215805e+00, -1.05785223e+00],
       [-1.55213526e+00, -4.65863678e-01, -3.77137129e-01],
       [ 1.70615233e+00,  3.26879532e-01, -1.38834434e+00],
       [-1.56045328e+00, -7.30111414e-01, -3.22877230e-01],
       [-1.86791555e-01, -1.21795339e+00, -1.01839048e+00],
       [-1.47846136e+00,  1.09252041e+00, -1.01839048e+00],
       [-5.89621422e-01, -8.99500988e-01, -1.29955542e+00],
       [-8.29655737e-01, -1.54995695e+00, -1.02332320e+00],
       [ 4.56072627e-01, -3.23576435e-01, -2.14357432e-01],
       [ 3.20607716e-01,  7.13087762e-01,  4.31828641e-01],
```

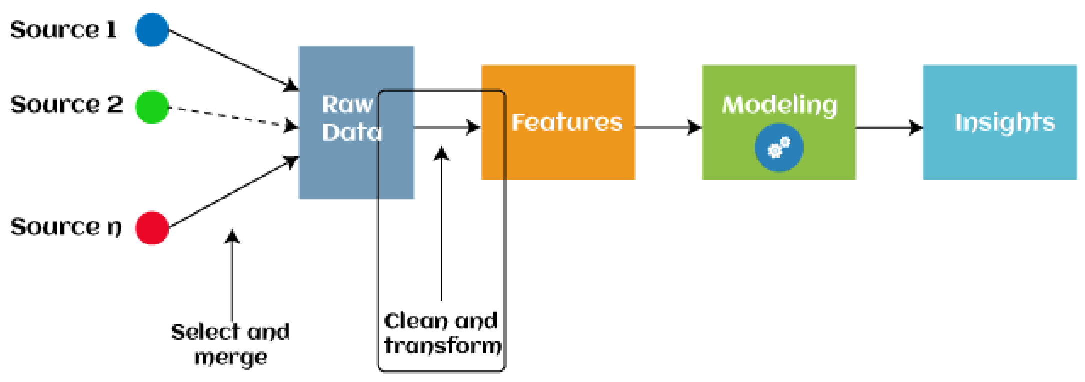
→ So this is how we can train a machine learning model to predict the future sales of a product.

Phase 4

- Feature Engineering
- Feature Engineering Techniques
    - Imputation
    - Handling Outliers
    - Log transform
    - Binning

Feature Engineering :

→ Feature engineering is the pre-processing step of machine learning, which extracts features from raw data**.** It helps to represent an underlying problem to predictive models in a better way, which as a result, improve the accuracy of the model for unseen data.
The predictive model contains predictor variables and an outcome variable, and while the feature engineering process selects the most useful predictor variables for the model.

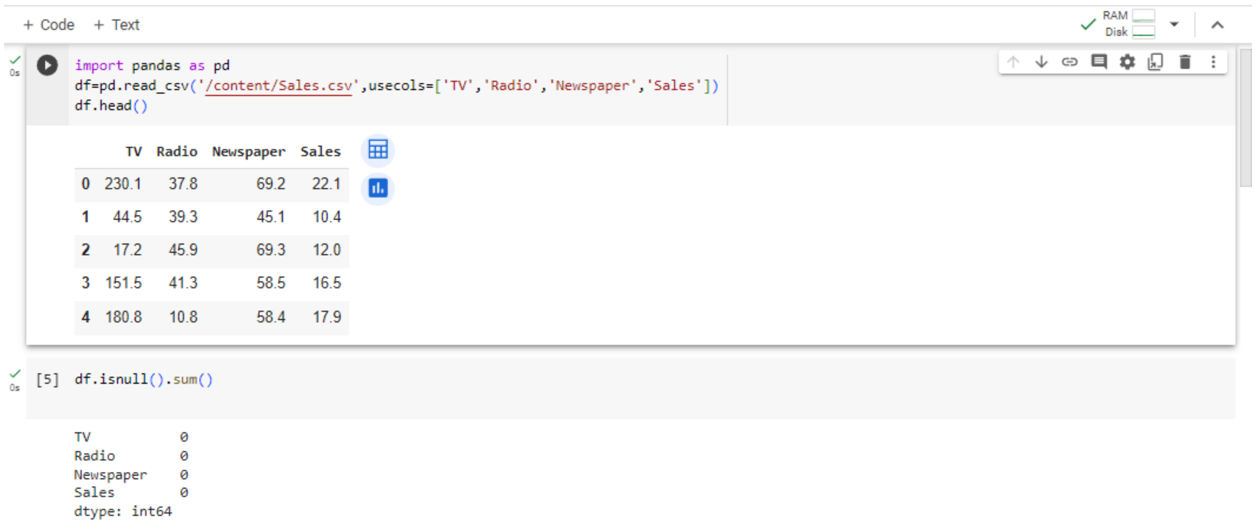


Feature Engineering Techniques :

→Imputation
→Handling Outliers
→Log transform
→Binning

Imputation :

→ Feature engineering deals with inappropriate data, missing values, human interruption, general errors, insufficient data sources, etc. Missing values within the dataset highly affect the performance of the algorithm, and to deal with them "Imputation" technique is used. Imputation is responsible for handling irregularities within the dataset .

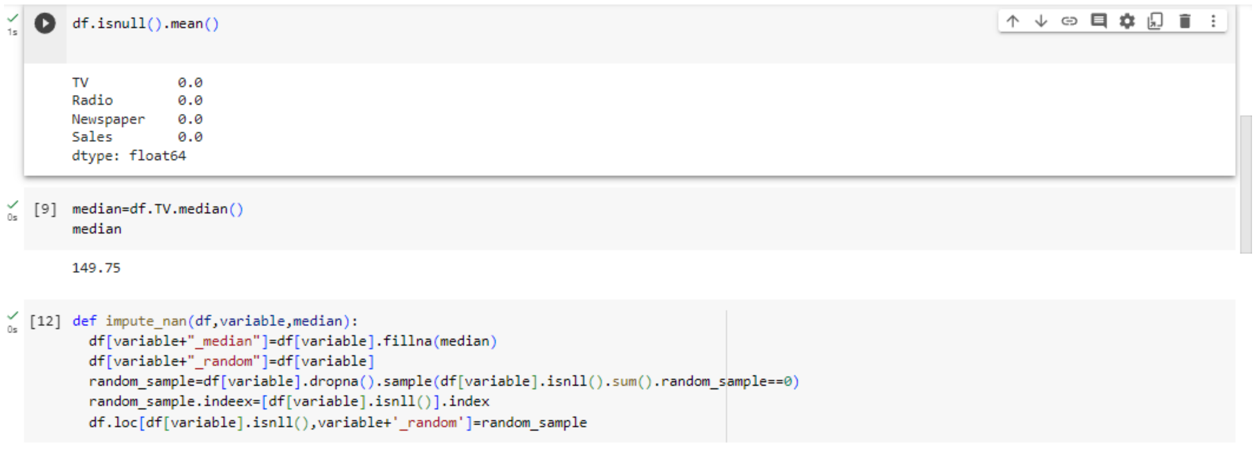


```
import pandas as pd
df=pd.read_csv('/content/Sales.csv',usecols=['TV','Radio','Newspaper','Sales'])
df.head()
```

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

```
[5] df.isnull().sum()
```

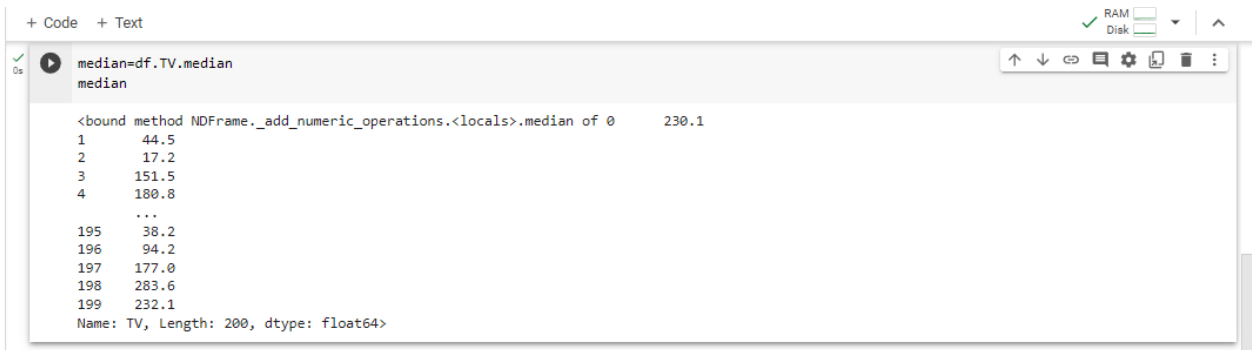
```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```



```
df.isnull().mean()

TV           0.0
Radio        0.0
Newspaper    0.0
Sales        0.0
dtype: float64
```

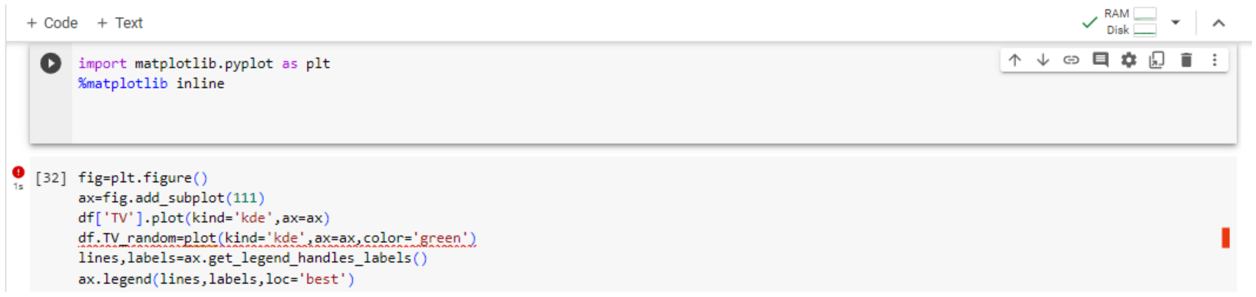
```
[9] median=df.TV.median()
    median

    149.75
```

```
[12] def impute_nan(df,variable,median):
       df[variable+"_median"]=df[variable].fillna(median)
       df[variable+"_random"]=df[variable]
       random_sample=df[variable].dropna().sample(df[variable].isnll().sum().random_sample==0)
       random_sample.indeex=[df[variable].isnll()].index
       df.loc[df[variable].isnll(),variable+'_random']=random_sample
```



+ Code + Text

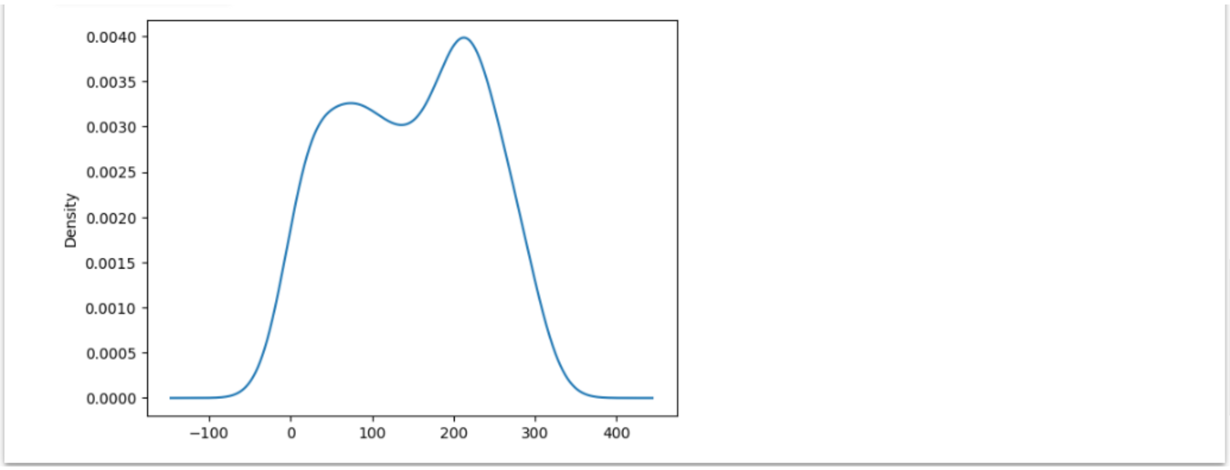
```
median=df.TV.median
median

<bound method NDFrame._add_numeric_operations.<locals>.median of 0      230.1
1       44.5
2       17.2
3      151.5
4      180.8
       ...
195     38.2
196     94.2
197    177.0
198    283.6
199    232.1
Name: TV, Length: 200, dtype: float64>
```



+ Code + Text

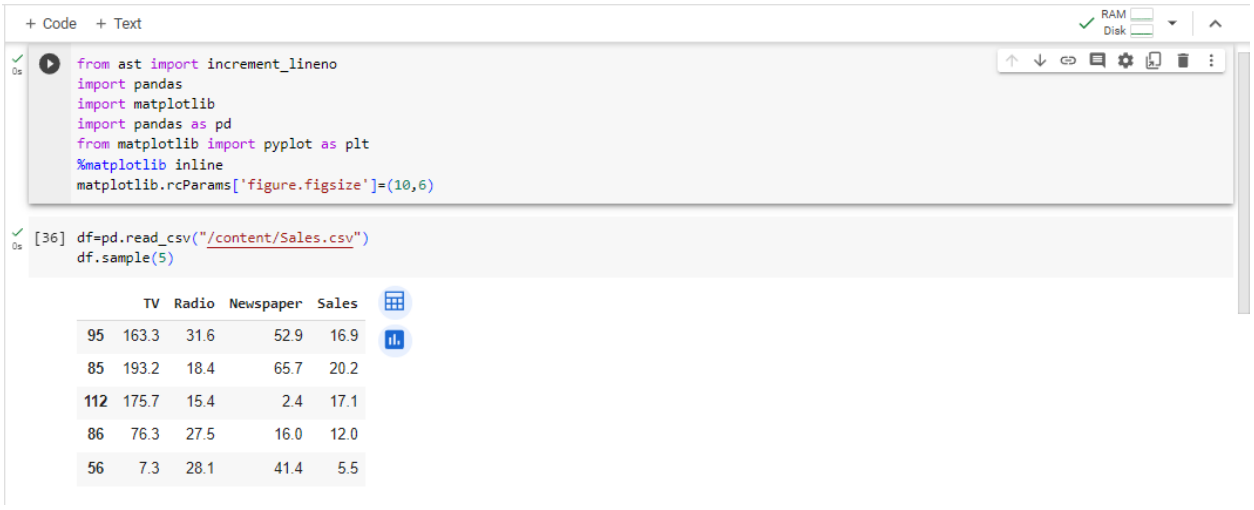
```
import matplotlib.pyplot as plt
%matplotlib inline
```

```
[32] fig=plt.figure()
     ax=fig.add_subplot(111)
     df['TV'].plot(kind='kde',ax=ax)
     df.TV_random=plot(kind='kde',ax=ax,color='green')
     lines,labels=ax.get_legend_handles_labels()
     ax.legend(lines,labels,loc='best')
```



Handling Outliers :

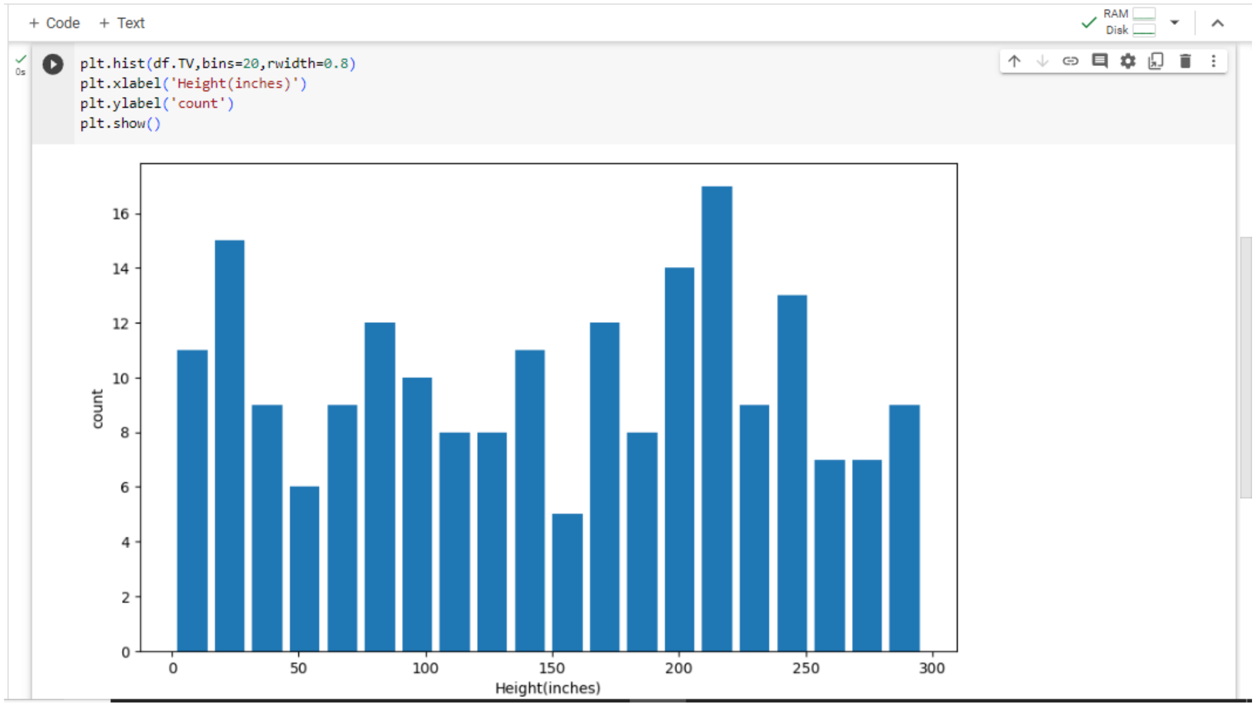
→ Standard deviation can be used to identify the outliers. For example, each value within a space has a definite to an average distance, but if a value is greater distant than a certain value, it can be considered as an outlier. Z-score can also be used to detect outliers.



```
from ast import increment_lineno
import pandas
import matplotlib
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
matplotlib.rcParams['figure.figsize']=(10,6)
```

```
[36] df=pd.read_csv("/content/Sales.csv")
     df.sample(5)
```

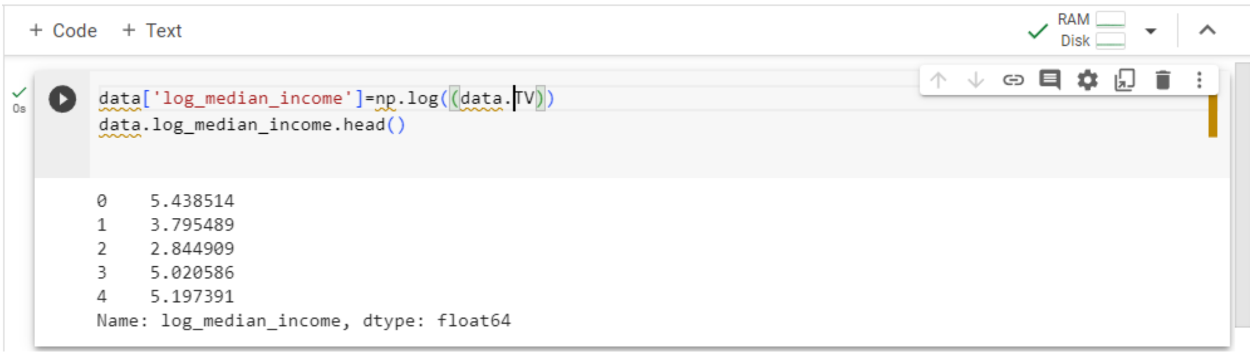
	TV	Radio	Newspaper	Sales
95	163.3	31.6	52.9	16.9
85	193.2	18.4	65.7	20.2
112	175.7	15.4	2.4	17.1
86	76.3	27.5	16.0	12.0
56	7.3	28.1	41.4	5.5



```
plt.hist(df.TV,bins=20,rwidth=0.8)
plt.xlabel('Height(inches)')
plt.ylabel('count')
plt.show()
```

Log transform :

→ Logarithm transformation or log transform is one of the commonly used mathematical techniques in machine learning. Log transform helps in handling the skewed data, and it makes the distribution more approximate to normal after transformation. It also reduces the effects of outliers on the data, as because of the normalization of magnitude differences, a model becomes much robust.

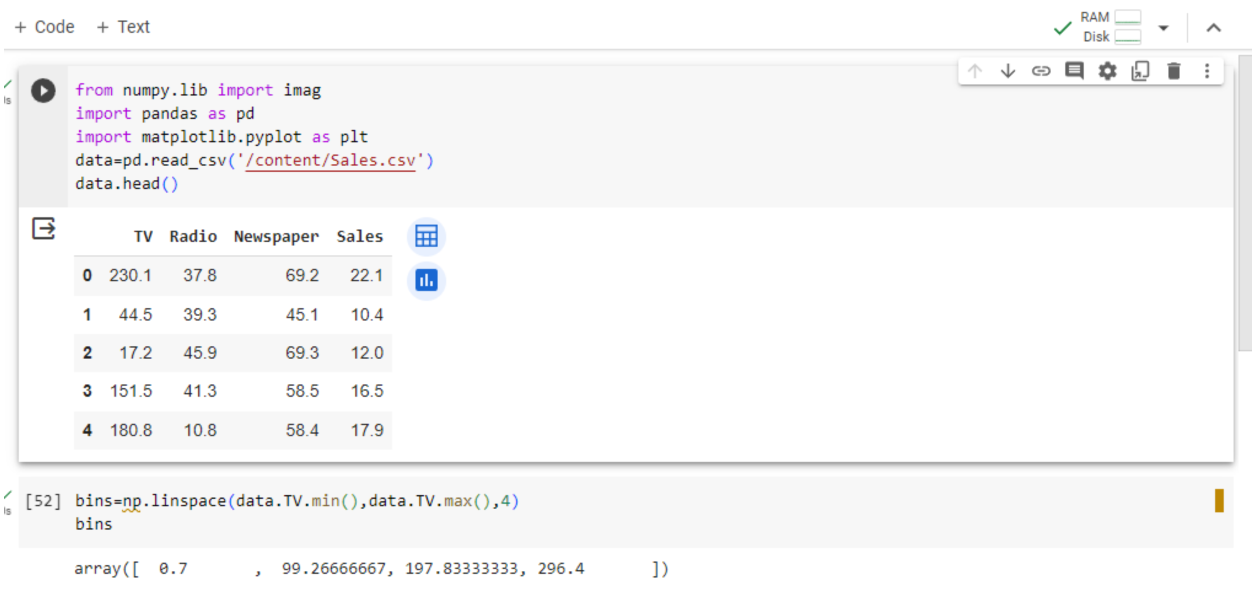


```
data['log_median_income']=np.log((data.TV))
data.log_median_income.head()
```

```
0    5.438514
1    3.795489
2    2.844909
3    5.020586
4    5.197391
Name: log_median_income, dtype: float64
```

Binning :

→ In machine learning, overfitting is one of the main issues that degrade the performance of the model and which occurs due to a greater number of parameters and noisy data. However, one of the popular techniques of feature engineering, "binning", can be used to normalize the noisy data. This process involves segmenting different features into bins.



```
from numpy.lib import imag
import pandas as pd
import matplotlib.pyplot as plt
data=pd.read_csv('/content/Sales.csv')
data.head()
```

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

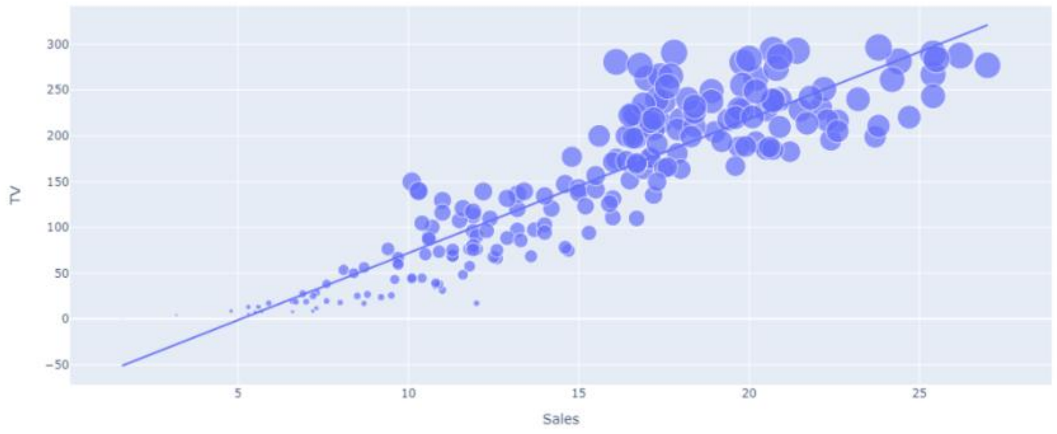
```
[52] bins=np.linspace(data.TV.min(),data.TV.max(),4)
     bins
```

```
array([  0.7       ,  99.26666667, 197.83333333, 296.4       ])
```

Evaluation :

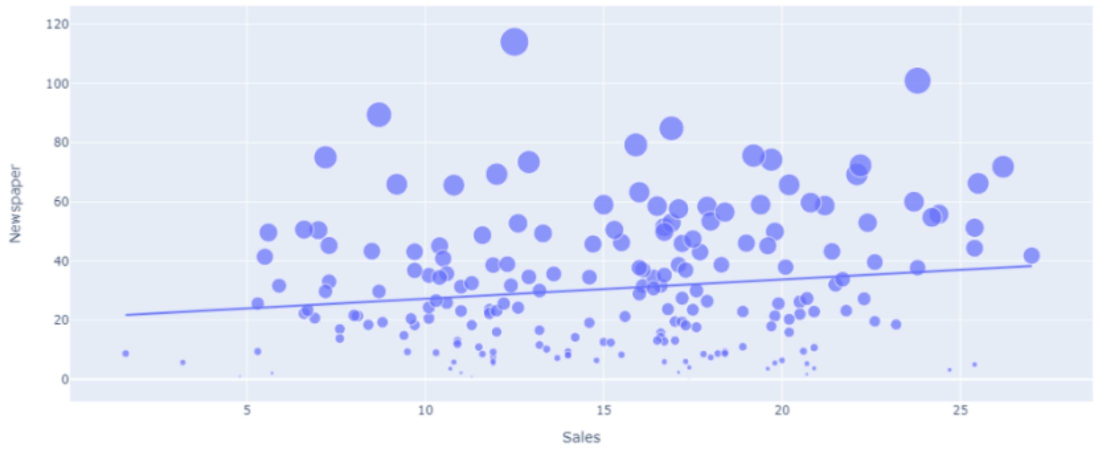
- ➢ Now let's visualize the relationship between the amount spent on advertising on TV and units sold:

```python
import plotly.express as px
import plotly.graph_objects as go
figure = px.scatter(data_frame = data, x="Sales",
                    y="TV", size="TV", trendline="ols")
figure.show()
```



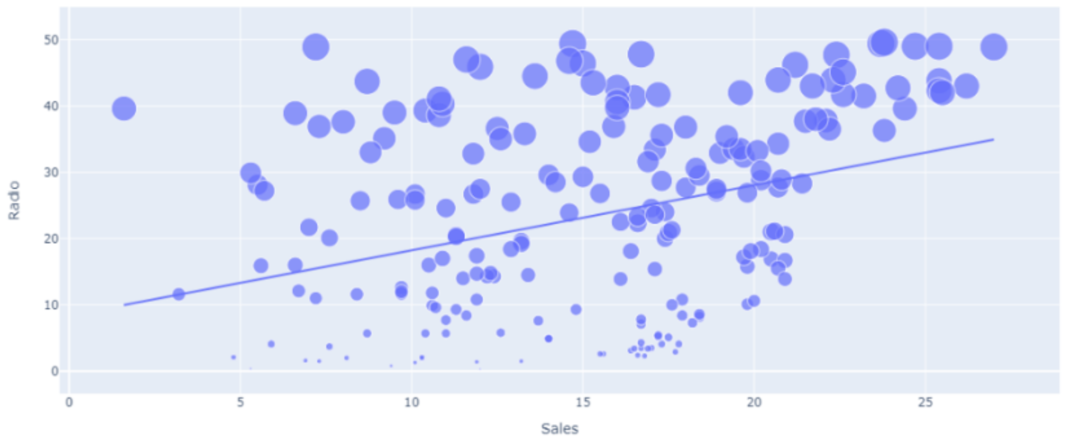
- Now let’s visualize the relationship between the amount spent on advertising on newspapers and units sold:

```
figure = px.scatter(data_frame = data, x="Sales",
                    y="Newspaper", size="Newspaper", trendline="ols")
figure.show()
```



➢ Now let's visualize the relationship between the amount spent on advertising on radio and units sold:

```
figure = px.scatter(data_frame = data, x="Sales",
                    y="Radio", size="Radio", trendline="ols")
figure.show()
```



- Out of all the amount spent on advertising on various platforms, I can see that the amount spent on advertising the product on TV results in more sales of the product. Now let’s have a look at the correlation of all the columns with the sales column:

```python
correlation = data.corr()
print(correlation["Sales"].sort_values(ascending=False))
```

```
Sales        1.000000
TV           0.901208
Radio        0.349631
Newspaper    0.157960
Name: Sales, dtype: float64
```

→ So this is how we can evaluate a machine learning model and perform the feature engineering for future sales prediction

Phase 5

- ALGORITHMS
- ADVANCED TECHNIQUES

ALGORITHMS

Linear Regression :

→ Linear Regression is one of the common ML and data analysis technique. This algorithm is helpful for forecasting based on linear regression equation. The Linear regression technique is the type of regression, which combines the set of 13 independent features(x) to predict the output value(y) or dependent variable. The linear equation assigns a factor to each independent variable called coefficients represented by β.

XGBoost:

→ XGBoost also known as Extreme Gradient Boosting has been used in order to get an efficient model with high computational speed and efficacy. The formula makes predictions using the ensemble method that models the anticipated errors of some decision trees to optimize last predictions. Production of this model also reports the value of each feature's effects in determining the last building performance score prediction.

Gradient Boosting:

→ Gradient Boost is the one of the major boosting algorithm. Boosting is a ensemble technique in which the successive predictors learn from the mistakes of the previous or predecessor predictors. It is the method of improving the weak learners and create a combined prediction model. In this algorithm, decision trees are mainly used as base learners and trains the model in sequential manner.

Random Forest :

→ Random forest is referred as a supervised machine learning ensemble method, which uses the multiple decision trees. It involves the technique called Bootstrap aggregation also known as bagging which aims to reduce the complexity of the models that overfit the training data . In this algorithm, rather than depending on individual decision tree it will combines the multiple decision trees to find the final outcome.

Extra Trees Algorithm:

→ This algorithm works by creating a large number of unpruned decision trees from the training dataset. Predictions are made by averaging the prediction of the decision trees in the case of regression or using majority voting in the case of classification.

ADVANCED TECHNIQUES

TIME SERIES FORECASTING MODEL

→ Time series forecasting models are used to project future sales based on past sales trends. These models take into account the seasonality of a product or service to better predict future demand. Essentially, a time series model is a mathematical equation that takes in data from previous periods, such as monthly or quarterly sales, and uses it to estimate future outcomes.

LSTM Neural Network

- LSTM is a type of recurrent neural network (RNN) that excels in modeling sequences and time dependencies. Unlike traditional feedforward neural networks, LSTM networks have a unique memory cell structure that enables them to retain and utilize information over longer time periods. This makes LSTM particularly effective in capturing long-term dependencies in time series data

Prophet

- Developed by Facebook, Prophet is a robust algorithm specifically designed for time series forecasting. It utilizes an additive model that captures non-linear trends, yearly, weekly, and daily seasonality, as well as holiday effects. It performs exceptionally well when applied to time series data with prominent seasonal patterns and a substantial historical record. One of Prophet's notable strengths is its ability to handle missing data and adapt to shifts in trends, while also demonstrating robustness in handling outliers.

CONCLUSION

Sales forecasting is mainly required for the organizations for business decisions. Accurate forecasting will help the companies to enhance the market growth. Machine learning techniques provides the effective mechanism in prediction and data mining as it overcome the problem with traditional techniques. These techniques enhances the data optimization along with improving the efficiency with better results and greater predictability. After predicting the purchase amount, the companies can apply some marketing strategies for certain sections of customers so that the profit could be enhanced.