Project Report

On

SALES ESTIMATION OF A COMPANY THROUGH MACHINE LEARNING MODEL

Submitted by

P Harshavardhan Reddy-R170130
N Tharuni-R171034
N Puneeth-R170182

Under the guidance of

S Shabana

Department of Computer Science and Engineering

Rajiv Gandhi University of Knowledge and Technologies(RGUKT),

R.K.Valley, Kadapa, Andra Pradesh.

Rajiv Gandhi University of Knowledge Techonolgies

RK Valley, Kadapa (Dist), Andhra Pradesh, 516330

<u>CERTIFICATE</u>

This is to certify that the project work titled "**SALES ESTIMATION OF A COMPANY**" is a bonafied project work submitted by **P Harshavardhan Reddy, N Tharuni, and N Puneeth** in **COMPUTER SCIENCE AND ENGINEERING** in partial fulfillment of requirements for the award of degree of **Bachelor of Technology** for the year **2021-2022** carried out the work under the supervision.

INTERNAL GUIDE	**HEAD OF THE DEPARTMENT**
(S SHABANA)	(HARINADH)

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success.

I am extremely grateful to our respected Director,**Prof. K. SANDHYA RANI** for fostering an excellent academic climate in our institution.

I also express my sincere gratitude to our respected Head of the Department **Mr HARINADH** for his encouragement, overall guidance in viewing this project a good asset and effort in bringing out this project.

I would like to convey thanks to our guide at college **S SHABANA** for his guidance, encouragement, co-operation and kindness during the entire duration of the course and academics.

My sincere thanks to all the members who helped me directly and indirectly in the completion of project work. I express my profound gratitude to all our friends and family members for their encouragement.

•

INDEX

ABSTRACT:

With the increasing influence of the Internet on people's life, the development of e-commerce platforms is more rapid, with users and earnings of these platforms showing a growing trend. In recent years, the strong support of national policies has also provided a good environment for the development of the e-commerce industry. Under the impact of the epidemic this year, the role of the e-commerce industry in the development of the national economy has become more prominent. In such cases, the number and the competitiveness of e-commerce platforms and e-commerce enterprises are increasing. If a platform wants to maintain its advantage in the competition, it must be able to better meet the needs of users, and do a good job in all aspects of coordination and management. At this point, the accurate forecast of the sales volume of e-commerce platforms is particularly important. At present, there are many studies on e-commerce sales prediction, but we are still exploring the prediction model that can be better applied in different scenarios.Let’s consider a problem statement, a e-commerce company wants to increase its sells to a certain amount. Now the challenge is to find the amount of investment on advertisement that will result the gain in sells.This project involves building the sales estimator for a company which utilizes the value of the money invested to predict the sales get by advertisements.

INTRODUCTION

In recent years, machine learning models based on big data have been introduced into marketing in order to transform customer data into meaningful insights and to make strategic decisions by making more accurate predictions. Although there is a large amount of literature on demand forecasting, there is a lack of research about how marketing strategies such as advertising and other promotional activities affect demand. Therefore, an accurate demand-forecasting/sales estimation model can make significant academic and practical contributions for business sustainability.

The purpose of this project is to evaluate machine learning methods to provide accuracy in forecasting demand based on advertising expenses. The study focuses on a prediction mechanism based on Machine Learning technique — REGRESSION —to deal with demand forecasting based on advertising expenses.

Accordingly, a manufacturer's real market dataset consisting of advertising expenditures on **TV ,SOCIAL MEDIA,NEWSPAPER**, sales and demand forecasting via chosen machine learning methods was analyzed and compared in terms of the accuracy of demand forecasting.
As a result,**MULTIPLE LINEAR REGRESSION** has been used here for providing highly accurate prediction results for sales estimation based on advertising expenses.

PURPOSE

The main purpose of Sales Estimation is to provide a platform where Companies can have an **virtual idea to spend** on what advertisement media so that they can utilise and get maximum benefits. And with the help of this project we are bringing the use of technology in the field of Sales Estimation where Comapnies can get an clear idea where to Invest how much of Investment to get maximum profits.

SCOPE

In earlier days, we are investing a lot amount of investment in advertisement without knowing **WHERE TO INVEST HOW MUCH**.As Technology is growing rapidly we are also moving to a technical world where everything we want to be online. So with the help of this project we are bringing the use of technology in the field of **sales estimation through advertisements** where Companies can avail all the results of sales by using past data at their door steps.This access friendly software provides quick and effective services which helps to increase their sales and profit.

REQUIREMENT SPECIFICATION

HARDWARE:

Ram	:	4GB
Hardisk	:	512GB
Processor	:	2GHz

SOFTWARE:

Language	: Python
Tools	:Jupyter-Notebook
GUI	:Tkinter
Additional Modules	:Numpy,Matplotlib,Seaborn,Sklearn,Pandas

ANALYSIS AND DESIGN

Data preparation:

#Importing the dataset

data = pd.read_csv("data/Advertising_data.csv")

data.head()

output:

	TV	social_media	newspaper	sales
0	230.1	37.8	69.2	2210
1	44.5	39.3	45.1	1040
2	17.2	45.9	69.3	930
3	151.5	41.3	58.5	1850
4	180.8	10.8	58.4	1290

First,we get the dataset into an variable “data”,here dataset include 3 parameters/input to our ML algorithm

1.Tv

2.social_media

3.newspaper

and Output is **SALES.**

Model configuration:

Applied models in this study is **Multiple Linear Regression**.

Problem Statement:

With the above dataset,we are required to estimate the sales by user provied investment for TV,

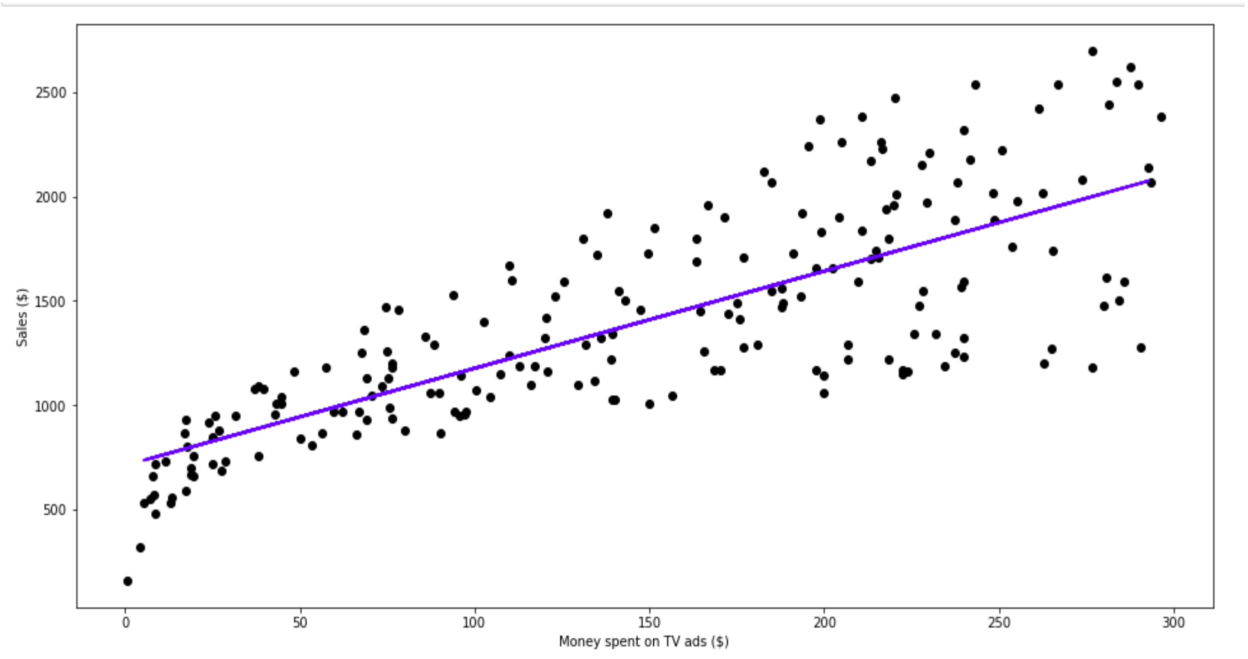
SOCIAL MEDIA, NEWSPAPER, just like:

Enter the amount you will invest on:

```
TV : 10
Social media : 10
Newspaper : 20
you will get Rs537.35 sales by advertising Rs10.0 on TV, Rs10.0 on Radio
and Rs20.0 on newspaper.
```

We would like to fit a linear regression model (shown below as the blue straight line) through these above points, so you can then predict sales.

A DEMO GRAPH FOR LINEAR REGRESSION USING ONLY ONE VARIABLE:



Model function:

For above Graph,the model function is linear regression (which is a function that maps from x to y) is represented as
The formula above is how you can represent straight lines - different values of w give you straight lines on the plot.

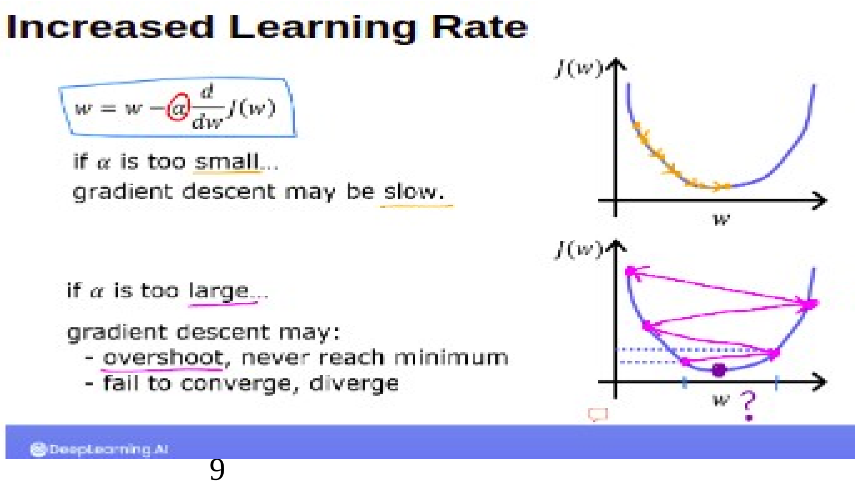
$f_{w,b}(x^{(i)})=wx^{(i)}+b$(1)

Code:

```
#Splitting our dataset to Training and Testing dataset
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


#Fitting Linear Regression to the training set
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit(X_train, y_train)

#And  then plotting the graph
```



Increased Learning Rate

$w = w - \alpha \frac{d}{dw}J(w)$

if α is too small...
gradient descent may be slow.

if α is too large...
gradient descent may:
- overshoot, never reach minimum
- fail to converge, diverge

NOW FOR BETTER RESULTS WE USE MULTIPLE LINEAR REGRESSION:

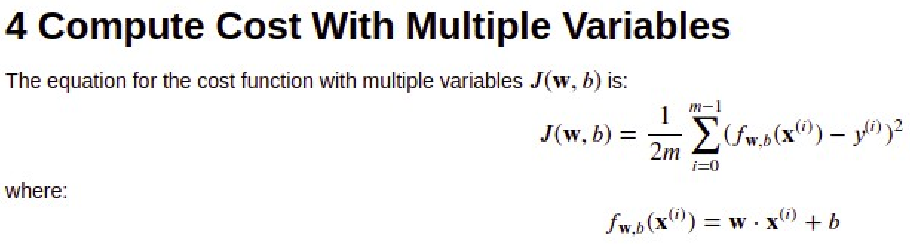
Linear regression provides a means of building models of the form:

$f_{\mathbf{w},b}=w_0x_0+w_1x_1+...+w_{n-1}x_{n-1}+b$

It is a combination of more than two variables:

. For our dataset example, we use this function : $y=w0x0+w1x21+w2x32+b$

For above Dataset we compute Cost Function:



4 Compute Cost With Multiple Variables

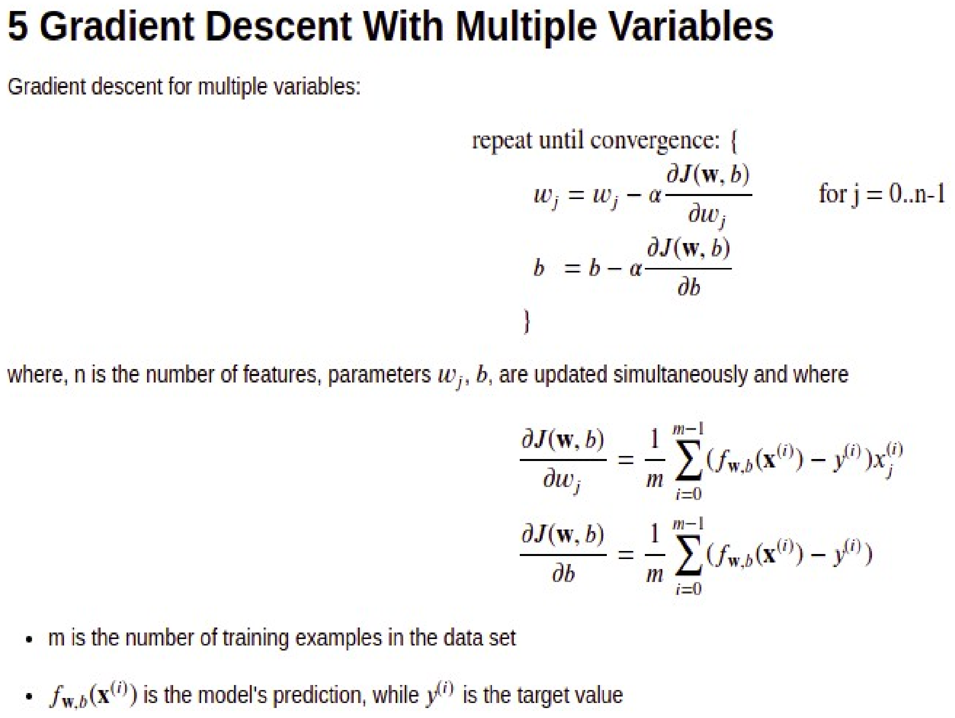
The equation for the cost function with multiple variables $J(\mathbf{w},b)$ is:

$$J(\mathbf{w},b)=\frac{1}{2m}\sum_{i=0}^{m-1}(f_{\mathbf{w},b}(\mathbf{x}^{(i)})-y^{(i)})^2$$

where:

$$f_{\mathbf{w},b}(\mathbf{x}^{(i)})=\mathbf{w}\cdot\mathbf{x}^{(i)}+b$$

For finding BEST w1, w2, w3 we use Gradient descentTechnique:



5 Gradient Descent With Multiple Variables

Gradient descent for multiple variables:

$$\begin{aligned}&\text{repeat until convergence: } \{\\ &\quad w_j = w_j - \alpha\frac{\partial J(\mathbf{w},b)}{\partial w_j} \qquad \text{for j = 0..n-1}\\ &\quad b\ \ = b - \alpha\frac{\partial J(\mathbf{w},b)}{\partial b}\\ &\}\end{aligned}$$

where, n is the number of features, parameters w_j, b, are updated simultaneously and where

$$\frac{\partial J(\mathbf{w},b)}{\partial w_j} = \frac{1}{m}\sum_{i=0}^{m-1}(f_{\mathbf{w},b}(\mathbf{x}^{(i)})-y^{(i)})x_j^{(i)}$$

$$\frac{\partial J(\mathbf{w},b)}{\partial b} = \frac{1}{m}\sum_{i=0}^{m-1}(f_{\mathbf{w},b}(\mathbf{x}^{(i)})-y^{(i)})$$

- m is the number of training examples in the data set
- $f_{\mathbf{w},b}(\mathbf{x}^{(i)})$ is the model's prediction, while $y^{(i)}$ is the target value

After computing all the above process we find best w1,w2,w3 using sklearn module called.

from sklearn.linear_model import LinearRegression.

<u>Steps Involved are:</u>

1.Storing the data in variables.

```
#Initializing the variables
X = data.drop(['sales'], axis=1)
y = data['sales'].values.reshape(-1,1)
```

2.Dividing the dataset into test and train.

```
#Splitting our dataset to Training and Testing dataset
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

3.Applyting Linear Regression Model on the train data set

```
#Fitting Linear Regression to the training set
from sklearn.linear_model import LinearRegression
multiple_reg = LinearRegression()
multiple_reg.fit(X_train, y_train)
```

4.After traing we get co-efficient of function using

multiple_reg.coef_

o/p:Example

```
array([[ 4.47295175, 18.91950542,  0.27611143]])
```

by using this co-efficient in the function we get value of sales

sales=$w0x0+w1x21+w2x32+b$

<u>INTERNAL WORKING MECHANISM OF MULTIPLE LINEAR REGRESSION:</u>

Section-1:

```
# initialize parameters
w_init = 0
b_init = 0
# some gradient descent settings
iterations = 10000
tmp_alpha = 1.0e-2
# run gradient descent
w_final, b_final, J_hist, p_hist = gradient_descent(x_train ,y_train, w_init, b_init, tmp_alpha, iterations, compute_cost, compute_gradient)
print(f"(w,b) found by gradient descent: ({w_final:8.4f},{b_final:8.4f})")
```

Section-2:

```
def gradient_descent(x, y, w_in, b_in, alpha, num_iters, cost_function, gradient_function):
    """
    Performs gradient descent to fit w,b. Updates w,b by taking
    num_iters gradient steps with learning rate alpha

    Args:
      x (ndarray (m,))  : Data, m examples
      y (ndarray (m,))  : target values
      w_in,b_in (scalar): initial values of model parameters
      alpha (float):     Learning rate
      num_iters (int):   number of iterations to run gradient descent
      cost_function:     function to call to produce cost
      gradient_function: function to call to produce gradient

    Returns:
      w (scalar): Updated value of parameter after running gradient descent
      b (scalar): Updated value of parameter after running gradient descent
      J_history (List): History of cost values
      p_history (list): History of parameters [w,b]
      """

    w = copy.deepcopy(w_in) # avoid modifying global w_in
    # An array to store cost J and w's at each iteration primarily for graphing later
    J_history = []
    p_history = []
    b = b_in
    w = w_in

    for i in range(num_iters):
        # Calculate the gradient and update the parameters using gradient_function
        dj_dw, dj_db = gradient_function(x, y, w , b)

        # Update Parameters using equation (3) above
        b = b - alpha * dj_db
        w = w - alpha * dj_dw

        # Save cost J at each iteration
        if i<100000:      # prevent resource exhaustion
            J_history.append( cost_function(x, y, w , b))
            p_history.append([w,b])
        # Print cost every at intervals 10 times or as many iterations if < 10
        if i% math.ceil(num_iters/10) == 0:
            print(f"Iteration {i:4}: Cost {J_history[-1]:0.2e} ",
                  f"dj_dw: {dj_dw: 0.3e}, dj_db: {dj_db: 0.3e}  ",
                  f"w: {w: 0.3e}, b:{b: 0.5e}")

    return w, b, J_history, p_history #return w and J,w history for graphing
```

Section-3:

```
def compute_gradient(x, y, w, b):
    """
    Computes the gradient for linear regression
    Args:
      x (ndarray (m,)): Data, m examples
      y (ndarray (m,)): target values
      w,b (scalar)    : model parameters
    Returns
      dj_dw (scalar): The gradient of the cost w.r.t. the parameters w
      dj_db (scalar): The gradient of the cost w.r.t. the parameter b
     """

    # Number of training examples
    m = x.shape[0]
    dj_dw = 0
    dj_db = 0

    for i in range(m):
        f_wb = w * x[i] + b
        dj_dw_i = (f_wb - y[i]) * x[i]
        dj_db_i = f_wb - y[i]
        dj_db += dj_db_i
        dj_dw += dj_dw_i
    dj_dw = dj_dw / m
    dj_db = dj_db / m

    return dj_dw, dj_db
```

Getting input from user and predicting sales value:

Section-4:

```
#Function to calculate the cost
def compute_cost(x, y, w, b):

    m = x.shape[0]
    cost = 0

    for i in range(m):
        f_wb = w * x[i] + b
        cost = cost + (f_wb - y[i])**2
    total_cost = 1 / (2 * m) * cost

    return total_cost
```

Getting input from user and predicting sales value:

```
#Taking the input from the user
print("Enter the ammount you will invest on:")
tv = float(input("TV : "))
social_media = float(input("Social Media : "))
newspaper = float(input("Newspaper : "))

#predicting the sales with respect to the inputs
output = multiple_reg.predict([[tv,radio,newspaper]])
print("you will get Rs{:.2f} sales by advertising Rs{} on TV, Rs{} on Social Media and Rs{} on newspaper."\
      .format(output[0][0] if output else "not predictable",tv,social_media,newspaper))
```

```
Enter the ammount you will invest on:
TV : 34
Social Media : 34
Newspaper : 39
```

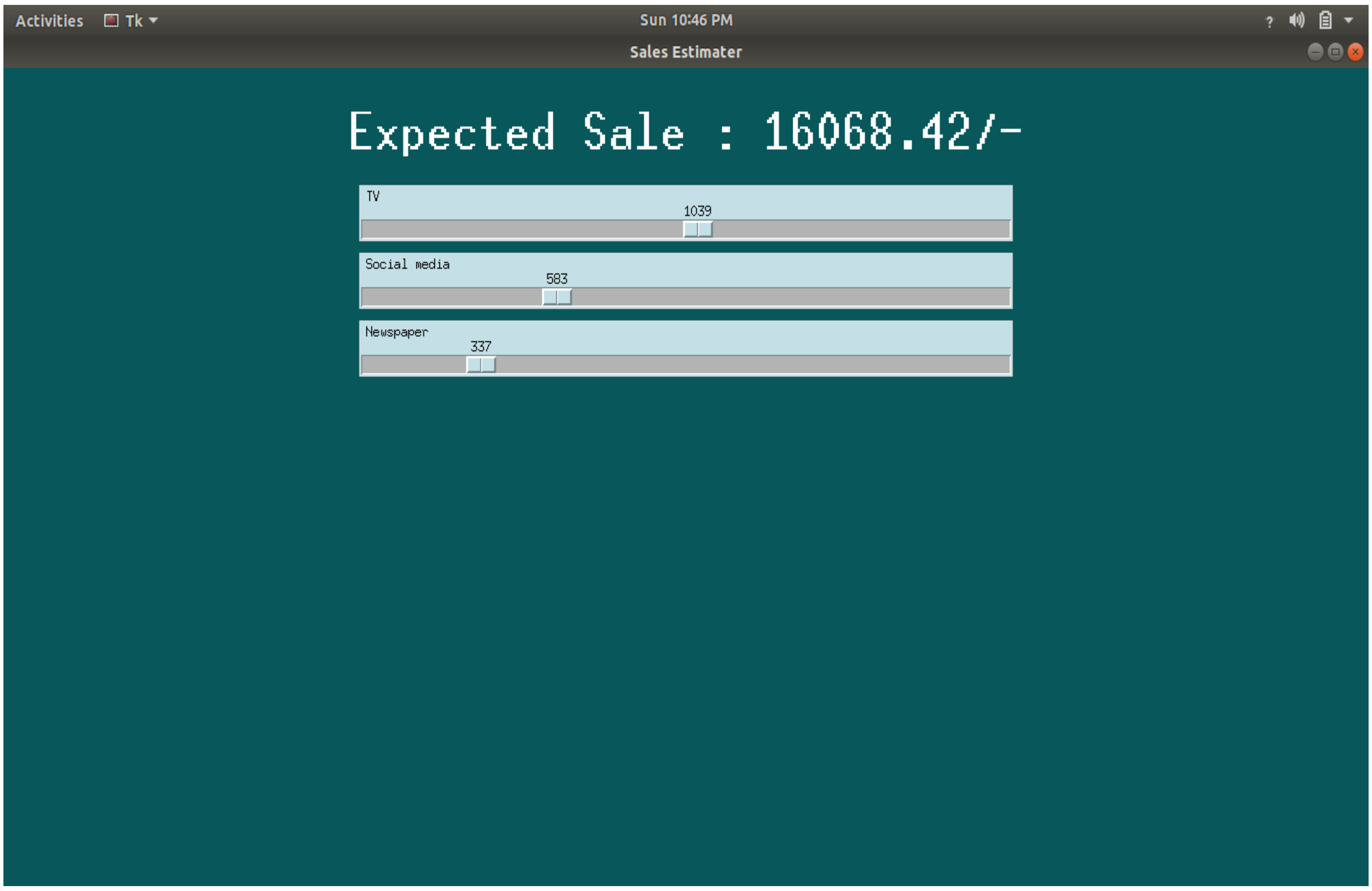
```
/home/puneeth/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning: X does not have valid featur
e names, but LinearRegression was fitted with feature names
  warnings.warn(
```

```
you will get Rs1312.13 sales by advertising Rs34.0 on TV, Rs45.0 on Social Media and Rs39.0 on newspaper.
```

Getting input from user and predicting sales value using GUI:

When Investment in

TV	:	1039
Social Media	:	583
Newspaper	:	337



Expected Sale : 16068.42/-

TV 1039
Social media 583
Newspaper 337

then sales will be 16068.42/-

CONCLUSION

ML methods are popularly used in marketing data in recent years to understand customer data better and to obtain meaningful insights. In this study the **effect of advertising on sales** is analyzed with using ML techniques. In other words, sales estimation of ML methods are compared based on the amount of advertising expenses. The other notable finding of the study is that the **accuracy of the prediction of sales increases as advertising expenses increase**. In this ML methods, prediction accuracy is best when advertising expenses are high. This result demonstrates that as the detail in the input data increases, the performance of the ML techniques improves substantially.

REFERENCES

- **https://numpy.org/doc/stable/**
- https://matplotlib.org/
- https://scikit-learn.org/stable/index.html
- https://seaborn.pydata.org/

***************************THANK YOU**************************