FAKULTI TEKNOLOGI MAKLUMAT DAN KOMUNIKASI

BITI 2223 MACHINE LEARNING

PROJECT REPORT

LECTURER: PROF DR AZAH KAMILAH MUDA

GROUP MEMBERS

NAME	MATRIC NO	SECTION/GROUP
ANG WEI KANG	B032110301	BITI S1G2
SIM WENG JIN	B032110376	BITI S1G2
LUM FU YUAN	B032110251	BITI S1G2
TEH XIAO THONG	B032110141	BITI S1G2

CONTENTS

1. EXECUTIVE SUMMARY

Artificial intelligence is used in many different fields including business field through machine learning algorithms. Machine learning is a part of computer science where a computer system can learn to perform a specific task without definite instruction. It is normally used in prediction and classification. Type of machine learning can be separate in to three which are supervised learning, unsupervised learning, and reinforcement learning. Supervised learning is an algorithm that trained by labeled datasets to make classification or prediction accurately. Unsupervised learning is the use of machine learning algorithms in analyzing and clustering unlabeled datasets. Reinforcement learning is an area of machine learning where it concerns how an intelligent agent can act in an environment to maximize the cumulative reward. In this project, the machine learning used is supervised learning where it is normally used in classification and regression. This type of learning will use data or feedback from humans to learn the relationship between input and next give the output.

2. PROJECT BACKGROUND

In this technology era, advertisements through different medium such as television, social media, radio, and influencer is one of the biggest ways for companies to promote their products or services. We can see advertisements everywhere and anytime in our lives. However, companies have limited budget to promote a product or service. They need to use the budget in an appropriate way to get the highest sales or benefits. They cannot try every of the budget separation possibility since it will spend a lot of time and money. Hence, they need a machine which can make predictions on the sales based on different separation of budget.

3. OBJECTIVE

The objectives of this project are shown as below:

1. To identify the major medium for a company to promote a product or service.
2. To predict the sales based on different separation of budget.

4. SCOPE

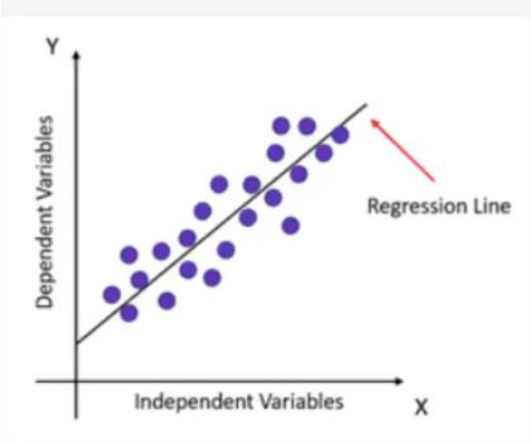
The target user of this project is the company admin and sales manager. They need to be clear about the benefits or feedback which can be brought by different separations of budget. Without this system, they need to have try on every possibility and will cause source wasting such as waste on financial or time. In this case, the system can help them to calculate and predict the best way of budget separation on promoting product or service to overcome the problem stated. To test whether the problem is solved or not, the company can compare the real sales with the predicted sales. If it is equal or larger, the problem is solved and vice versa.

5. PROJECT SIGNIFICANT

The project can read all the training data and learn the relationship between every advertising medium and sales. It can also make predictions on the sales when testing data which is budget on every advertising medium is inserted.

6. EXPECTED OUTCOME

Linear Regression is a type of supervised method in machine learning. It tries to apply connections that will forecast sales outcomes based on various budget separations from independent variables. The regression line is typically a straight line that fits the data points as closely as possible. So that the expected outcome will be like two-dimension graph as shown as below:



Linear regression may predict to find out the optimal medium for boosting sales while reducing the budget to a minimum.

7. TOOLS AND ALGORITHM TO USE

There are several algorithms in machine learning, including Decision Tree, Logistic Regression, Random Forest, K-nearest neighbour, K-Means clustering, Q-learning, and others. To identify the best optimum method, the machine learning algorithm will be chosen based on the task. In this project, the machine learning algorithm used is Linear Regression. It performs regression tasks. Regression can predict the numeric target label of a data point. The prediction will be based on learning from known datasets. Regression is basically used for predicting the value of a dependent variable (y=output) based on a given independent variable (x = input), which is useful for estimating revenues as a result of spending on various marketing methods. Another tool used in this project is the Cost Function of Linear Regression. The cost function is used to find the Root Mean Squared Error (RMSE) between the predicted y value (predict value) and true y value (true value) to make sure accuracy.

8. JUSTIFIABLE AND EXPLORATORY DATA ANALYSIS

According to IBM, exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their characteristics, often employing data visualization methods. The data set used in this project is Dummy Data HSS, where it use the data of television, influencer, radio, and social media ads budget to predict sales. The explanation of this data set using EDA is shown as below:

Importing Libraries

```
import pandas as ps
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
```

First, we need to import necessary libraries. Pandas is a package to make importing and analyze data. Numerical Python (Numpy) is a library for doing arrays and using in some domains like linear algebra and matrices. Seaborn is a library to making statistical graphics. Matplotlib is a Python package for building static, animated, and interactive visualizations. Linear Regression is a library built on top of NumPy and a few other libraries.

Read Data

read data

```
In [136]: df = ps.read_csv("Dummy Data HSS.csv")
          df.head()
```

Out[136]:

	TV	Radio	Social Media	Influencer	Sales
0	16.0	6.566231	2.907983	Mega	54.732757
1	13.0	9.237765	2.409567	Mega	46.677897
2	41.0	15.886446	2.913410	Mega	150.177829
3	83.0	30.020028	6.922304	Mega	298.246340
4	15.0	8.437408	1.405998	Micro	56.594181

We need to import the data from our device into Jupyter notebook. The data printed is the data in the data set imported. The data is ready to be explored.

Data Understanding

```
In [3]: df.shape
Out[3]: (4572, 5)
```

We can use ".shape" to show the total number of rows and columns in the data set. The total rows of the data set are 4572 rows while the total columns is 5 columns.

```
In [4]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4572 entries, 0 to 4571
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   TV            4562 non-null   float64
 1   Radio         4568 non-null   float64
 2   Social Media  4566 non-null   float64
 3   Influencer    4572 non-null   object
 4   Sales         4566 non-null   float64
dtypes: float64(4), object(1)
memory usage: 178.7+ KB
```

".info" is used to show the columns' data type and finding whether the data contain null values or not. The data type of television, radio, social media, and sales is float64 while influencer is object. The influencer column may not be shown in some of the following analysis since most of the analysis can only be done on numerical data.

```
In [5]: df.describe()
```

Out[5]:

	TV	Radio	Social Media	Sales
count	4562.000000	4568.000000	4566.000000	4566.000000
mean	54.066857	18.160356	3.323956	192.466602
std	26.125054	9.676958	2.212670	93.133092
min	10.000000	0.000684	0.000031	31.199409
25%	32.000000	10.525957	1.527849	112.322882
50%	53.000000	17.859513	3.055565	189.231172
75%	77.000000	25.649730	4.807558	272.507922
max	100.000000	48.871161	13.981662	364.079751

The describe() function in pandas I used to get various summary statistics. This function will return the count, mean, standard deviation, minimum, quantiles, and maximum of the data. The summary statistic of the data set used is shown as above.

Convert to make coding easy

```
In [139]: df.replace('Nano',1, inplace = True)
          df.replace('Micro',2, inplace = True)
          df.replace('Macro',3, inplace = True)
          df.replace('Mega',4, inplace = True)
```

Convert the column “influencer” from String to int, making coding easy to handling.

```
In [141]: df.columns = df.columns.str.lower()
          df.columns = df.columns.str.replace(' ','_')
```

Set the head column to lower case and replace the space to "_", make it easy to code.

Data Cleaning

```
In [143]: #Dropping rows with missing values.
          print(df.isnull().sum())
          df = df.dropna()
```

```
tv              10
radio            4
social_media     6
influencer       0
sales            6
dtype: int64
```

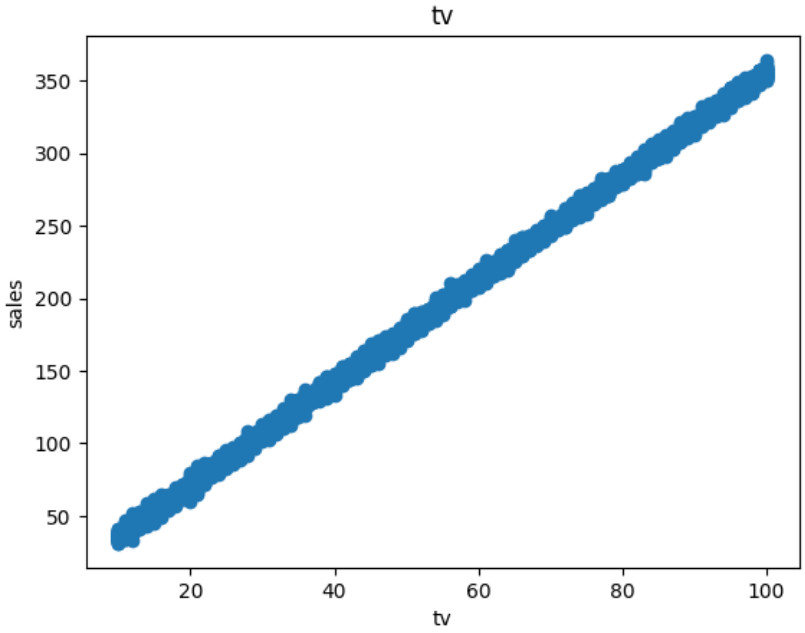
Drop the rows which with missing values to decrease affecting to the accuracy.

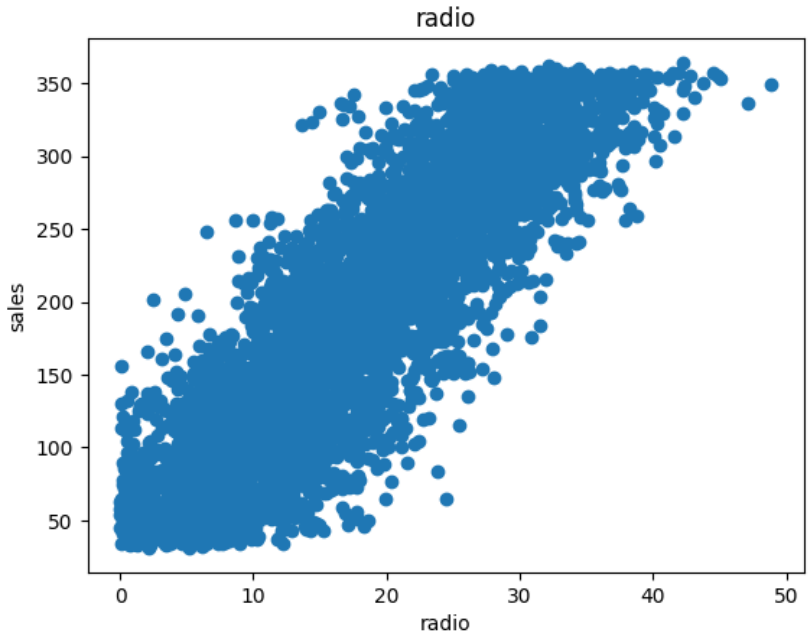
Data Exploratory Analysis

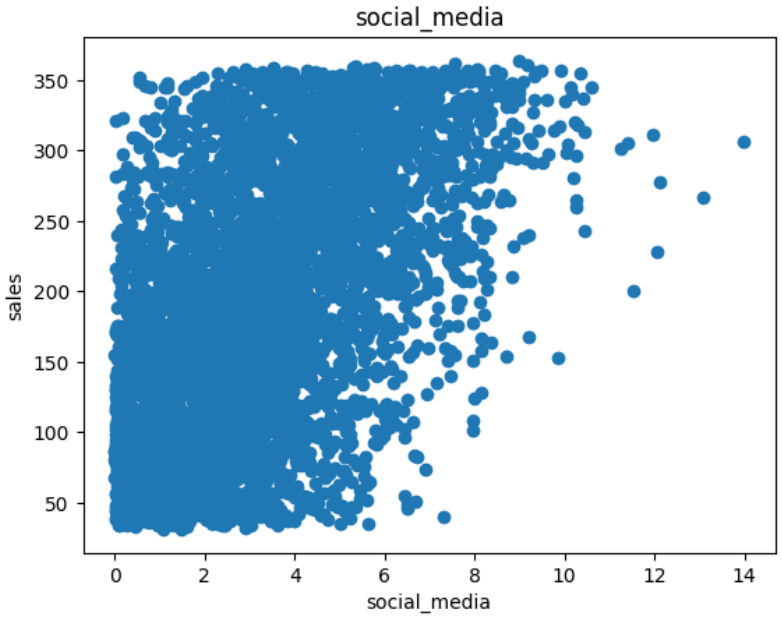
Data Exploratory Analysis

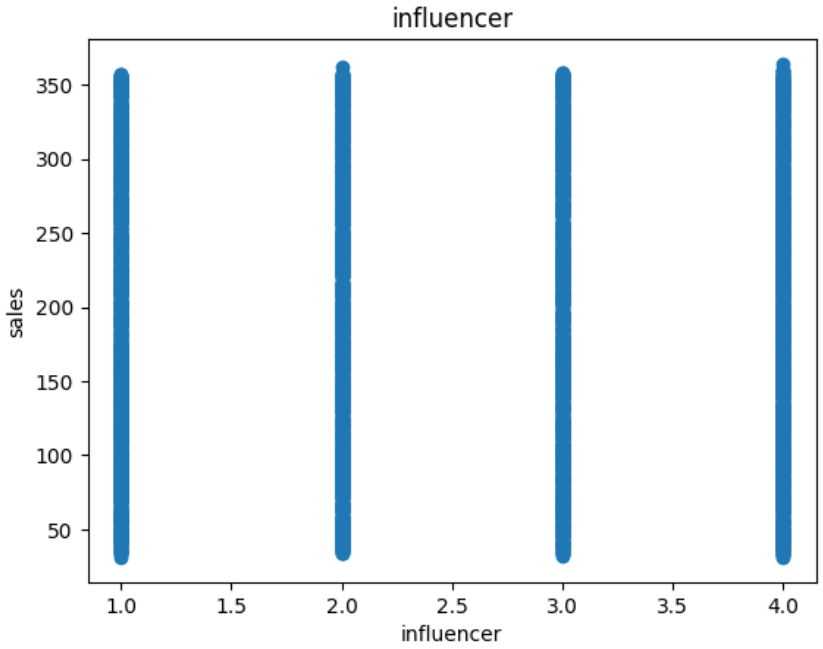
In [144]:
```
for label in df.columns[0:-1]:
    plt.scatter(df[label],df["sales"])
    plt.title(label)
    plt.ylabel("sales")
    plt.xlabel(label)
    plt.show()
```

Then, we used the matplotlib function to plot each graph, show below:



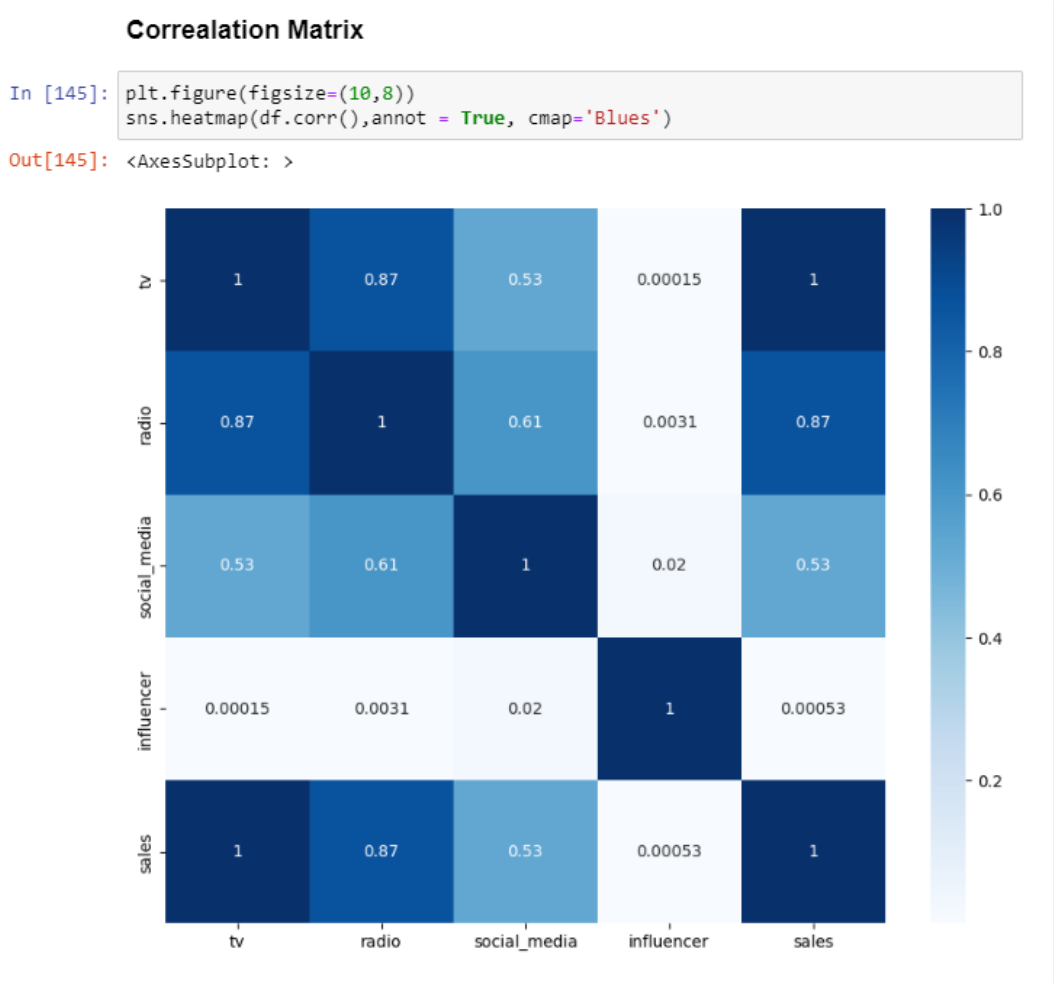






From the 4 graph above, we can discover the independent variables (x) relation between Dependent variable (y).

To investigate the relationship between sales and its variables.



Above is the visualization of correlation strength between variables using seaborn library.

This is the use of correlation matrix to find the correlation among the variables where it will give an idea of the correlation strength between different variables. The range of the correlation is from +1 to 0 as +1 is highly positively correlated while 0 is highly negatively correlated.

9. FEATURE ENGINEERING

Feature engineering

In [146]:
```
#To drop the irrelevant column
df.drop(['influencer'], axis = 1, inplace = True)
```

In [147]:
```
df
```

Out[147]:

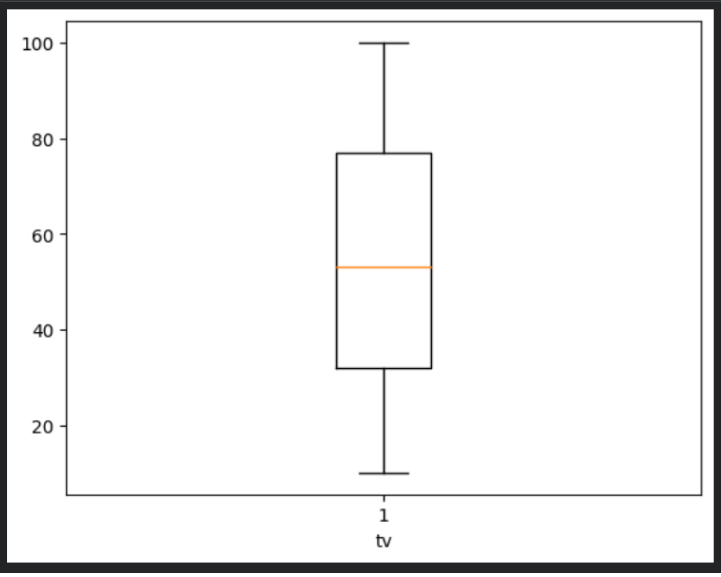
	tv	radio	social_media	sales
0	16.0	6.566231	2.907983	54.732757
1	13.0	9.237765	2.409567	46.677897
2	41.0	15.886446	2.913410	150.177829
3	83.0	30.020028	6.922304	298.246340
4	15.0	8.437408	1.405998	56.594181
...	...	...	...	...
4567	26.0	4.472360	0.717090	94.685866
4568	71.0	20.610685	6.545573	249.101915
4569	44.0	19.800072	5.096192	163.631457
4570	71.0	17.534640	1.940873	253.610411
4571	42.0	15.966688	5.046548	148.202414

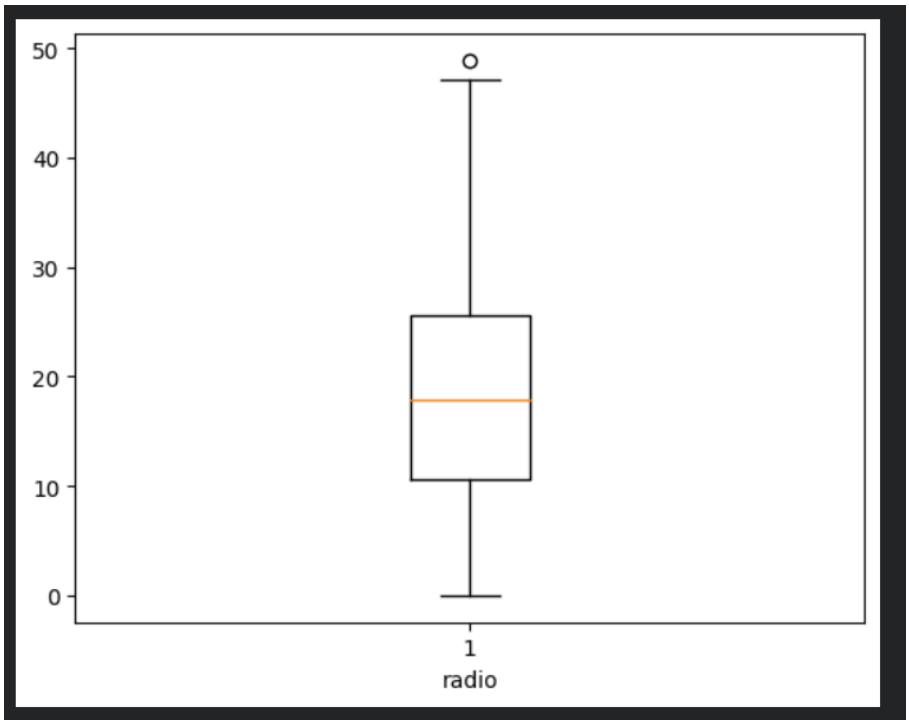
4546 rows × 4 columns

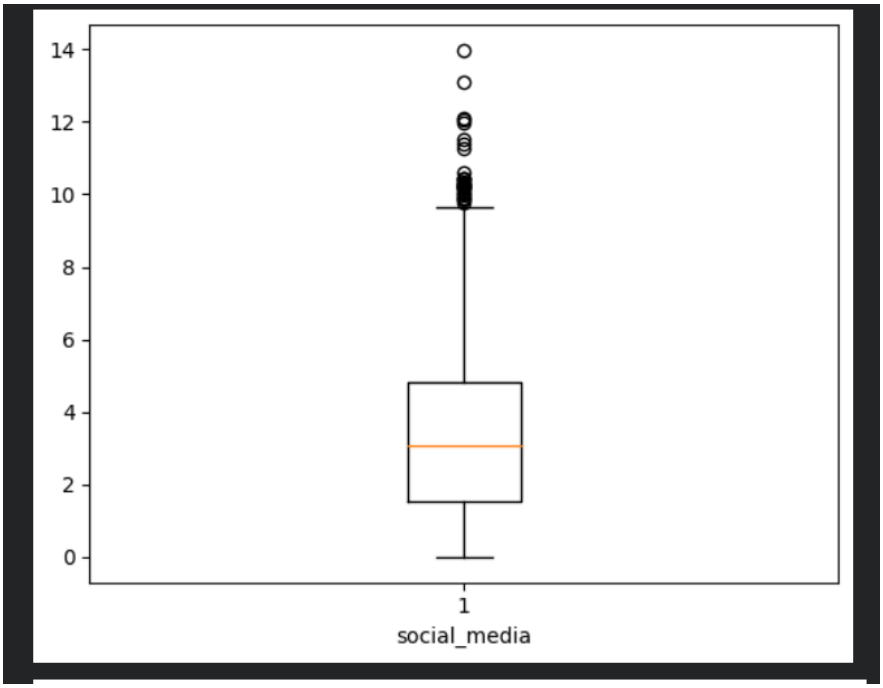
We discovered from the exploratory data analysis that we can decide to drop the irrelevant column to increase the accuracy for the final result. So, from graph and correlation matrix that, we decide to drop column “influencer” because it is most irrelevant to the sales, that only 0.00053% in correlation.

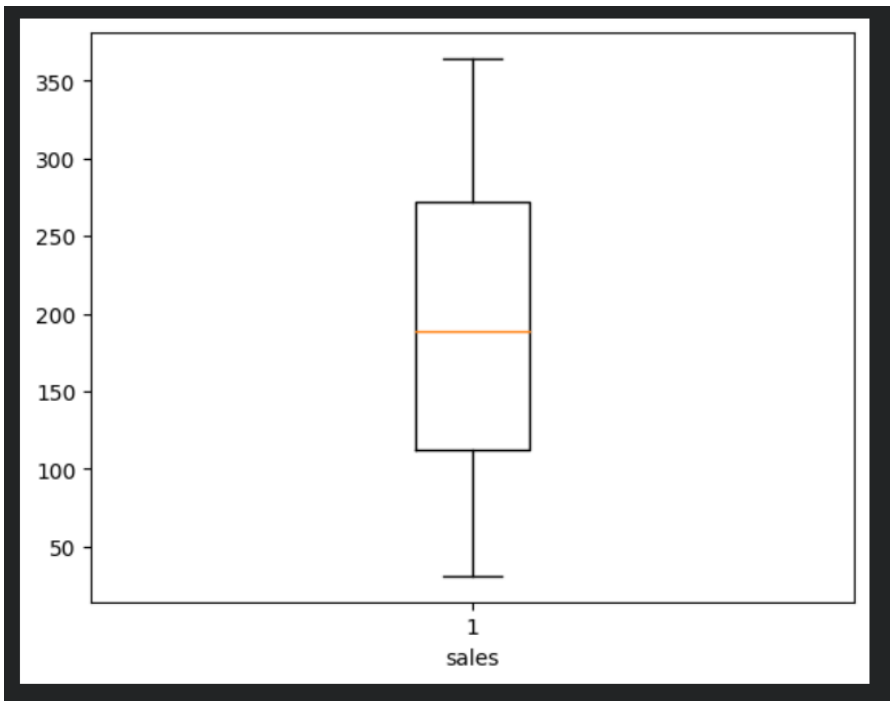
Detecting Outlier Visually

```
In [148]:  for c in df.columns:
               plt.boxplot(df[c])
               plt.xlabel(c)
               plt.show()
```









According to the boxplot, we can discover that a lot of point on graph “social_media” had out of range (outliers). So we need to finding the threshold for below the lower whisker and beyond the upper whisker to remove the outliers.

In [16]:
```
# finding lower quantile Q1 and upper quantile Q3 and Inner Quantile Range IQR
Q1s= df['social_media'].quantile(0.25)
Q3s= df['social_media'].quantile(0.75)
IQRs= Q3s-Q1s
print("lower quantile :",Q1s)
print("upper quantile :",Q3s)
print("Inner Quantile Range :",IQRs)
# finding the lower whisker and upper whisker
Lower_Whiskers = Q1s-1.5*IQRs
Upper_Whiskers = Q3s+1.5*IQRs
print(Lower_Whiskers,Upper_Whiskers)
#So we create a thresold for below the lower whisker and beyond the upper whisker
```

```
lower quantile : 1.5308215755
upper quantile : 4.80491913
Inner Quantile Range 3.2740975545
-3.3803247562500003 9.71606546175
```

To finding the lower whisker and upper whisker of the graph “social_media”, we need to find out the Inner quantile range (IQR). To obtain IQR, using Lower quantile(Q1) minus Upper quartile (Q3). Thus, to obtain lower whiskers by using Lower quantile(Q1) minus 1.5 times IQR, then obtain lower whiskers by Upper quartile(Q3) plus 1.5 times IQR.

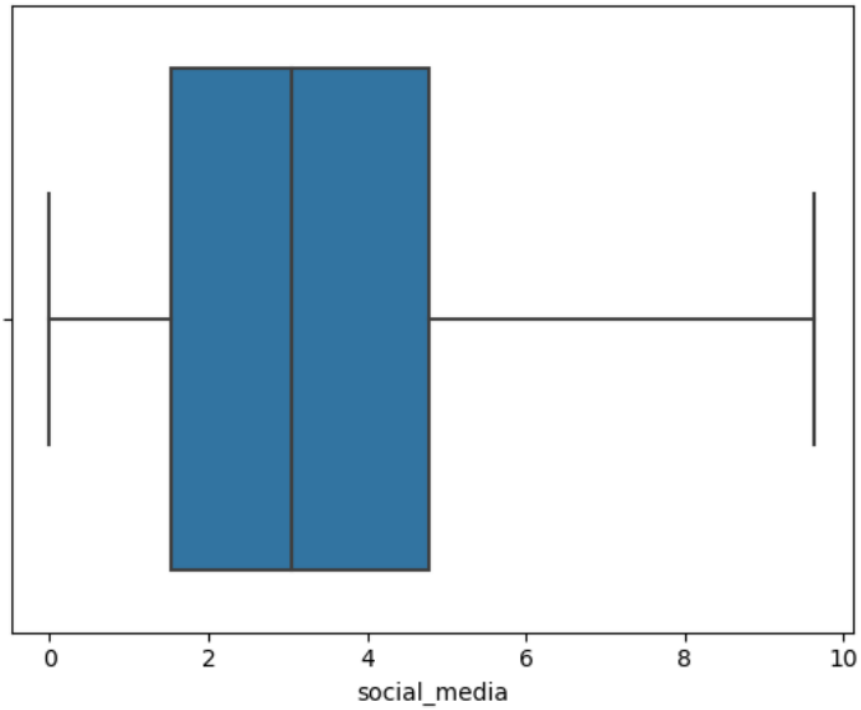
In [17]:
```
#Apply conditions to remove outliers
df = df[df['social_media']< Upper_Whiskers]
df = df[df['social_media']> Lower_Whiskers]
df['social_media']
```

Out[17]:
```
0       2.907983
1       2.409567
2       2.913410
3       6.922304
4       1.405998
          ...
4567    0.717090
4568    6.545573
4569    5.096192
4570    1.940873
4571    5.046548
Name: social_media, Length: 4518, dtype: float64
```

To set the conditions to remove the outliers, to make sure that can increase the quality and accuracy in training and testing data.

In [18]:
```
# check boxplot
sns.boxplot(x=df['social_media'])
```

Out[18]: <AxesSubplot: xlabel='social_media'>



Check again, make sure that are not any outliers in the boxplot of social media.

10. MACHINE LEARNING SOLUTION

In [19]:
```
import statsmodels.formula.api as smf

model1 = smf.ols('sales~tv+radio+social_media', data=df).fit()
print(model1.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  sales   R-squared:                       0.999
Model:                            OLS   Adj. R-squared:                  0.999
Method:                 Least Squares   F-statistic:                 1.488e+06
Date:                Sun, 22 Jan 2023   Prob (F-statistic):               0.00
Time:                        14:49:28   Log-Likelihood:                -11300.
No. Observations:                4518   AIC:                         2.261e+04
Df Residuals:                    4514   BIC:                         2.263e+04
Df Model:                           3
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept       -0.1274      0.103     -1.231      0.218      -0.330       0.075
tv               3.5629      0.003   1047.562      0.000       3.556       3.570
radio           -0.0043      0.010     -0.441      0.659      -0.024       0.015
social_media    -0.0016      0.026     -0.062      0.950      -0.052       0.049
==============================================================================
Omnibus:                        0.054   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.973   Jarque-Bera (JB):                0.032
Skew:                          -0.001   Prob(JB):                        0.984
Kurtosis:                       3.013   Cond. No.                         149.
==============================================================================
```

For this project, we will use Python's stats models module to implement the Ordinary Least Squares(OLS) linear regression method. A linear regression model establishes the relation between a dependent variable(**y**) and at least one independent variable**(**x**)**. In OLS method, we must choose the values of and such that, the total sum of squares of the difference between the calculated and observed values of y, is minimized. R-squared is the coefficient of determination or accuracy. It is the proportion of the variance in the dependent variable that is predictable/explained. Adj. R-squared (Adjusted R-squared) is the modified form of R-squared adjusted for the number of independent variables in the model. Value of adj. R-squared increases when we include extra variables which improve the model. F-statistic is the ratio of the mean squared error of the model to the mean squared error of residuals. It determines the overall significance of the model. Coef is the coefficients of the independent variables and the constant term in the equation. It is similar with y=mx+c formula. t is the value of the t-statistic. It is the ratio of the difference between the estimated and hypothesized value of a parameter to the standard error. The formula to calculate the sales is sales = -0.1274 + 3.5629tv + (-0.0043)radio + (-0.0016)social media.

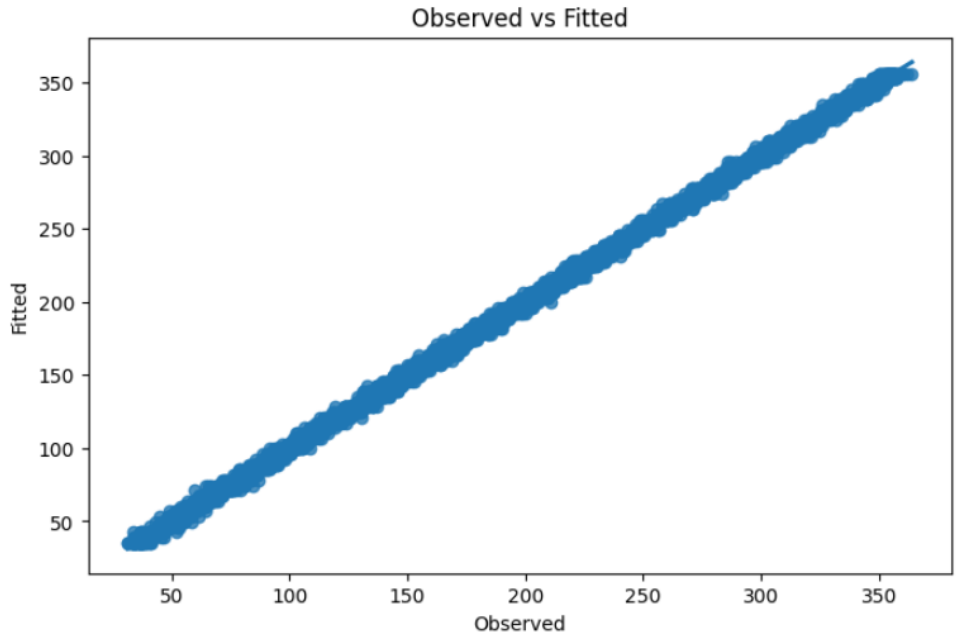
```
[20]: model1_pred= model1.predict(df)
      model1_pred
```

```
Out[20]: 0        56.846016
         1        46.146555
         2       145.878264
         3       295.452638
         4        53.277424
                    ...
         4567     92.487593
         4568    252.739098
         4569    156.546560
         4570    252.759758
         4571    149.437409
         Length: 4518, dtype: float64
```

This is the data prediction after the machine learnt from its experience.

In [21]:
```
fig, ax = plt.subplots()
fig.set_size_inches(8,5)
sns.regplot(x=df['sales'], y=model1_pred)
plt.xlabel("Observed")
plt.ylabel("Fitted")
plt.title('Observed vs Fitted')
```

Out[21]: Text(0.5, 1.0, 'Observed vs Fitted')



From the results table, we note the coefficient of x and the constant term. These values are substituted in the original equation, and the regression line is plotted using matplotlib. x is original data, y is predicted data, the graph shows that the machine learns perfectly.

11. EXPERIMENTATION

In [22]:
```
# Splitting the data into train and test data
from sklearn.model_selection import train_test_split
dataset_train,dataset_test  = train_test_split(df.iloc[:,:],test_size = 0.2)
dataset_train.columns
```

Out[22]: Index(['tv', 'radio', 'social_media', 'sales'], dtype='object')

In [23]:
```

t_tv= dataset_train['tv']
t_radio=dataset_train['radio']
t_socialmedia= dataset_train['social_media']
t_sales= dataset_train['sales']
```

In [24]:
```
s#Train set performance
from sklearn.metrics import mean_absolute_error
model_train = smf.ols('t_sales~t_tv+t_radio+t_socialmedia',data=dataset_train).fit()
train_pred = model_train.predict(dataset_train)
train_resid = train_pred - dataset_train['sales']
train_rmse = np.sqrt(np.mean(train_resid*train_resid))
print("RMSE train: ",train_rmse)
print("MAE train: ",mean_absolute_error(dataset_train['sales'],train_pred))
```

```
RMSE train:  2.961100941447648
MAE train:  2.373196807327511
```

In experimentation, we choose Root Mean Square Error(RMSE) and Mean Absolute Error(MAE) to determine the error metrics, which enable us to track efficiency and accuracy. First, we use a 0.2 test size to train the data in the dataset. Then we stored the trained data in t_tv, t_radio, t_socialmedia, and t_sales. Hence, we train its performance using trained data. After the evaluation, the RMSE result is 2.9611, while MAE is 2.3732. It turns out that the variance in the individual error in the sample is more minor as the difference between the RMSE and MAE is not large.

12. RESULT AND DISCUSSION

```
In [25]: Tv_Budget = 60
         Radio_Budget = 18.02
         Social_media_Budget = 1.92
         Prediction = ps.DataFrame({"tv":[Tv_Budget], "radio":[Radio_Budget], "social_media":[Social_media_Budget]})
         Prediction["Expected Sales"] = model1.predict(Prediction)
         Prediction
```

Out[25]:

	tv	radio	social_media	Expected Sales
0	60	18.02	1.92	213.565768

To test whether the machine works or not, we need to set the budget for television ad, radio ad, and social media ad. For example, according to the diagram above, the budgets for the three ad stated are 60, 18.02, and 1.92 (in million) respectively. The machine will calculate and print out the expected sale based on its experience. The expected output of this budget separation is 213.565768 (in million). Hence, by using this machine, the company can have an expectation on the sale based on different separation of ad budgets.

13. CONCLUSION

In conclusion, the machine has fulfilled the requirements of the company and achieved the objectives of this project. This supervised machine has learnt from the learning data to earn experience. It can use the experience to make well prediction of sales. Thus, the machine can be said as a successful machine.

14. REFERENCE

1. Exploratory Data Analysis
   https://www.ibm.com/topics/exploratory-data-analysis

2. Multiple Linear Regression (MLR)
https://www.investopedia.com/terms/m/mlr.asp

Presentation Video (YouTube link)

https://youtu.be/R65EIMQlLtI