MACHINE LEARNING

REGRESSION

MACHINE LEARNING - CLASSIFICATION

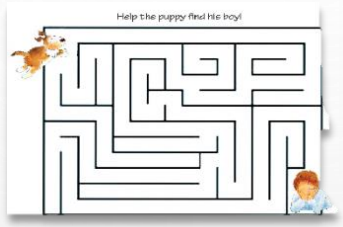
OVERVIEW OF THREE TYPES OF ML ALGORITHMS

SUPERVISED LEARNING

This type of learning algorithms work from learning the ground truth from LABELLED data.

The Algorithms in this classification tries to learn the pattern from the examples shown to it,

Example: Linear Regression, Decision Trees, and Naïve Bayes.



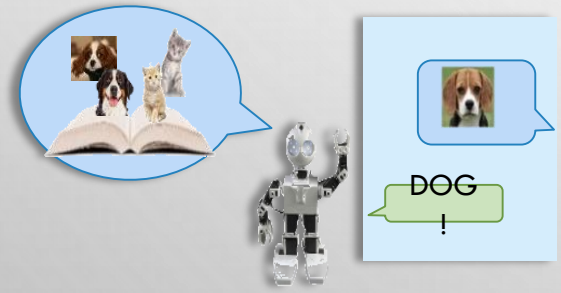
REINFORCEMENT LEARNING

This is another interesting genre of algorithms where the objective of the "agent" is defined.

The means of achieving the goals are directed by dictating the acts that will attract "rewards" and "penalty."

The algorithms tries to complete the objective with rewards and least penalty.

Q learning is an example algorithm in this area.

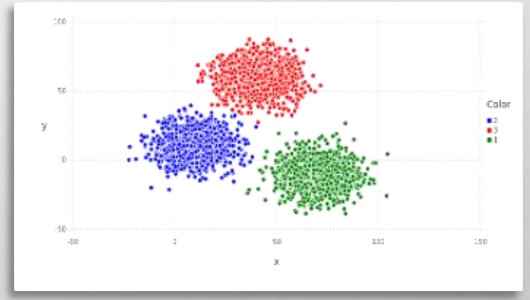


UNSUPERVISED LEARNING

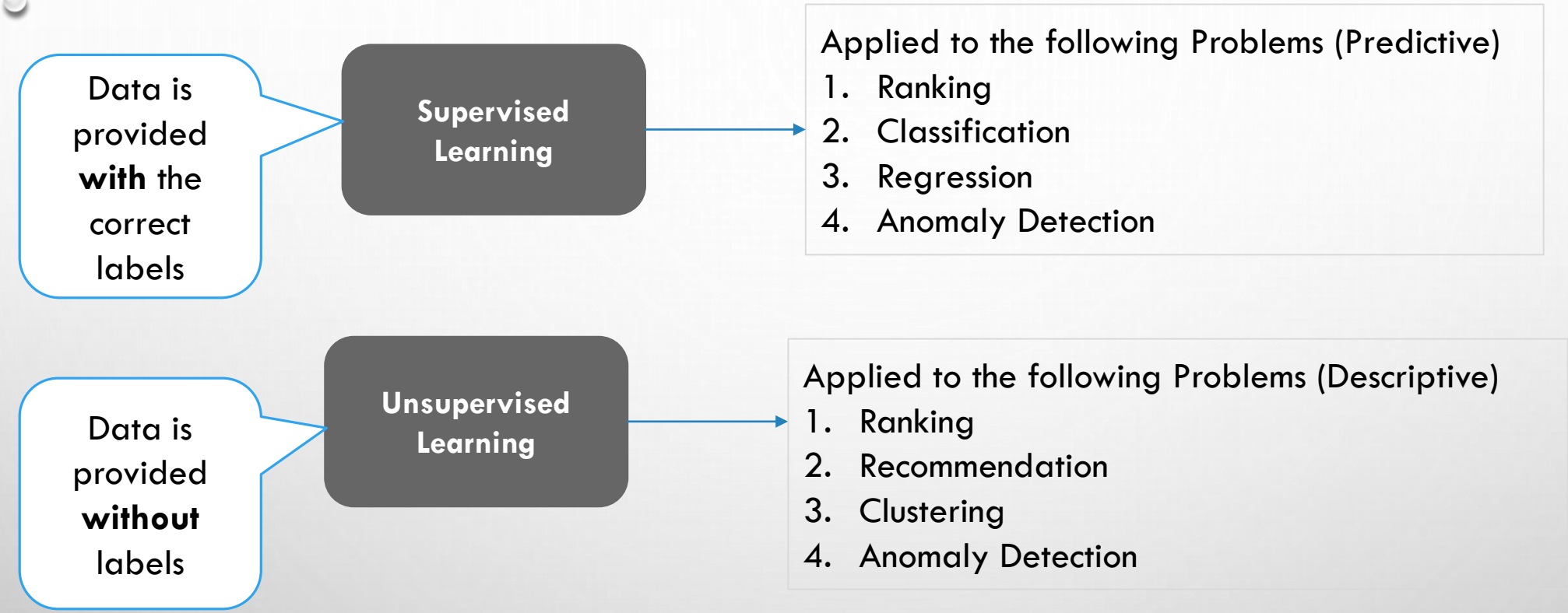
Techniques that don't require examples for training data because they don't attempt to make predictions.

These algorithms make effort to quantify the similarity within each data points and try to club those data points that are considered related.

K-Means clustering, Hierarchical clustering.



ML APPLICATIONS



SUPERVISED LEARNING

01 As one can imagine, This segment, arguably dominates most of the commercial application of Machine learning in current state of the subject.

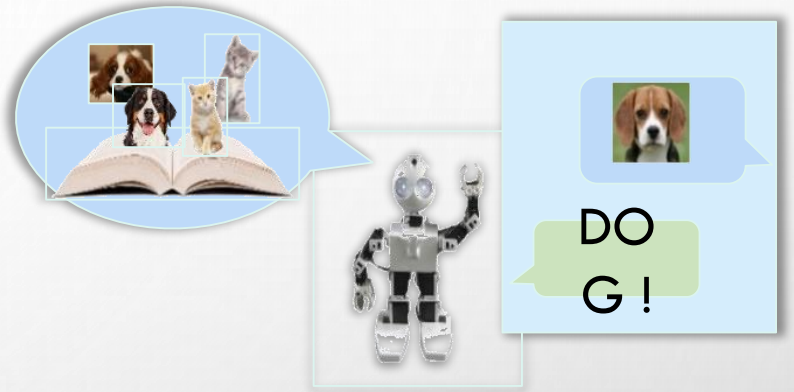
02 This extends from classifying the Medical condition to predicting the selling price of a house in a given locality.

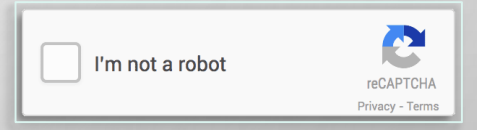
03 Since these section of algorithms need labelled data in mass for learning, this poses a huge problem for employing them into use.

04 Some of the interesting ways of collecting labelled data includes captcha

05 In Supervised learning, there is usually a phase where the algorithm learns the pattern from the labelled data, aka Training Phase

06 The phase where the knowledge learnt from the training is put to action and the algorithm is asked to perform is called Test phase. These phases are unique to Supervised learning.





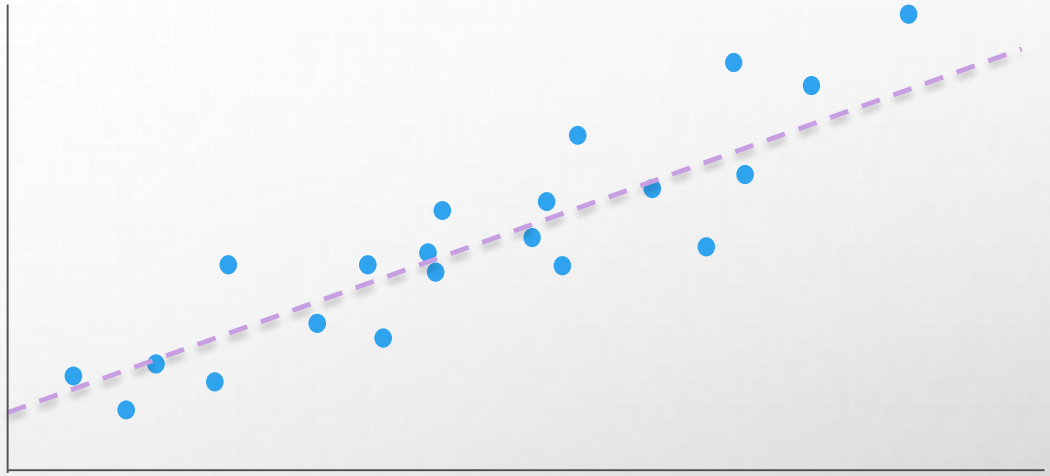
SUPERVISED LEARNING

Data is provided **with** the correct labels

Supervised Learning

Regression
(Quantity)

Classification
(Category)



Target

Features

Salary	Experience	Age
860000	10	35
640000	6	27

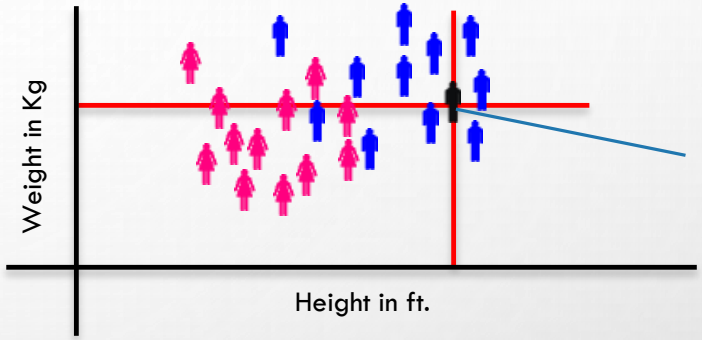
SUPERVISED LEARNING

Data is provided **with** the correct labels

Supervised Learning

Regression
(Quantity)

Classification
(Category)

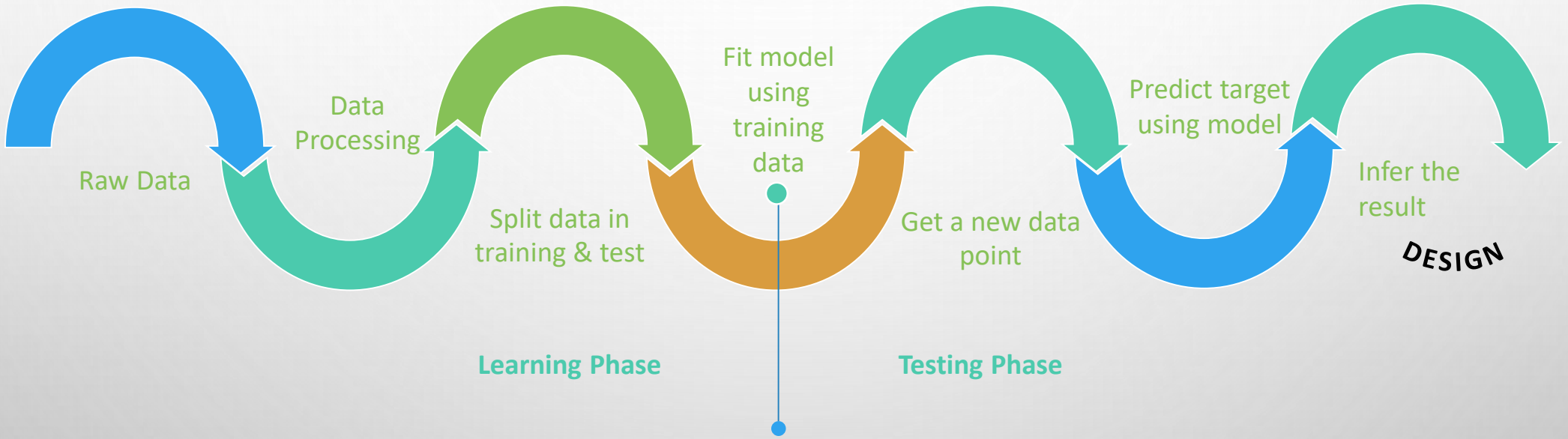


Target | Features

Gender	Height in ft	Weight in Kg
M	5.9	66
F	5.5	70

SUPERVISED LEARNING - NUTSHELL

let us take a look at the process of any supervised learning.



LINEAR REGRESSION

- A simple but powerful approach to predict quantitative response i.e. predicting numerical variables
- Many complex static learning approaches can be seen as generalizations or extensions of linear regression
- Jump start algorithm for any one who wants to start learning machine learning algorithms

Examples :

Predicting sales values based on expenses incurred on TV, Newspaper and Radio advertisements.

Determining the housing price given the parameters like the total area in sqft of the house, distance from the railway station, crime rate in the area and so on.

Determining the purchasing power of an individual given his/her salary.

Estimation of the water usage / electricity usage in a family given the number of people ratio of genders.

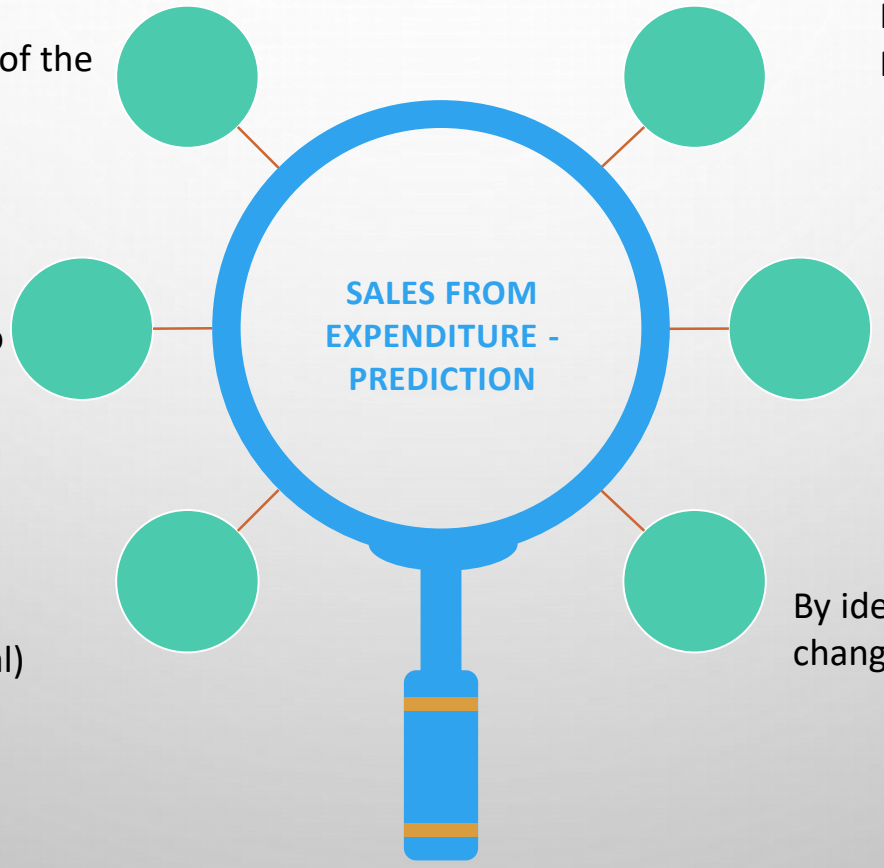
Supervised Approach: Need labeled data for prediction

PREDICTING SALES FROM EXPENDITURE

The data set (advertising.csv) contains sales of the product in 200 different markets

Along with sales, they have also collected the spending in TV, Newspaper and Radio advertisements

Target : Sales (numerical)



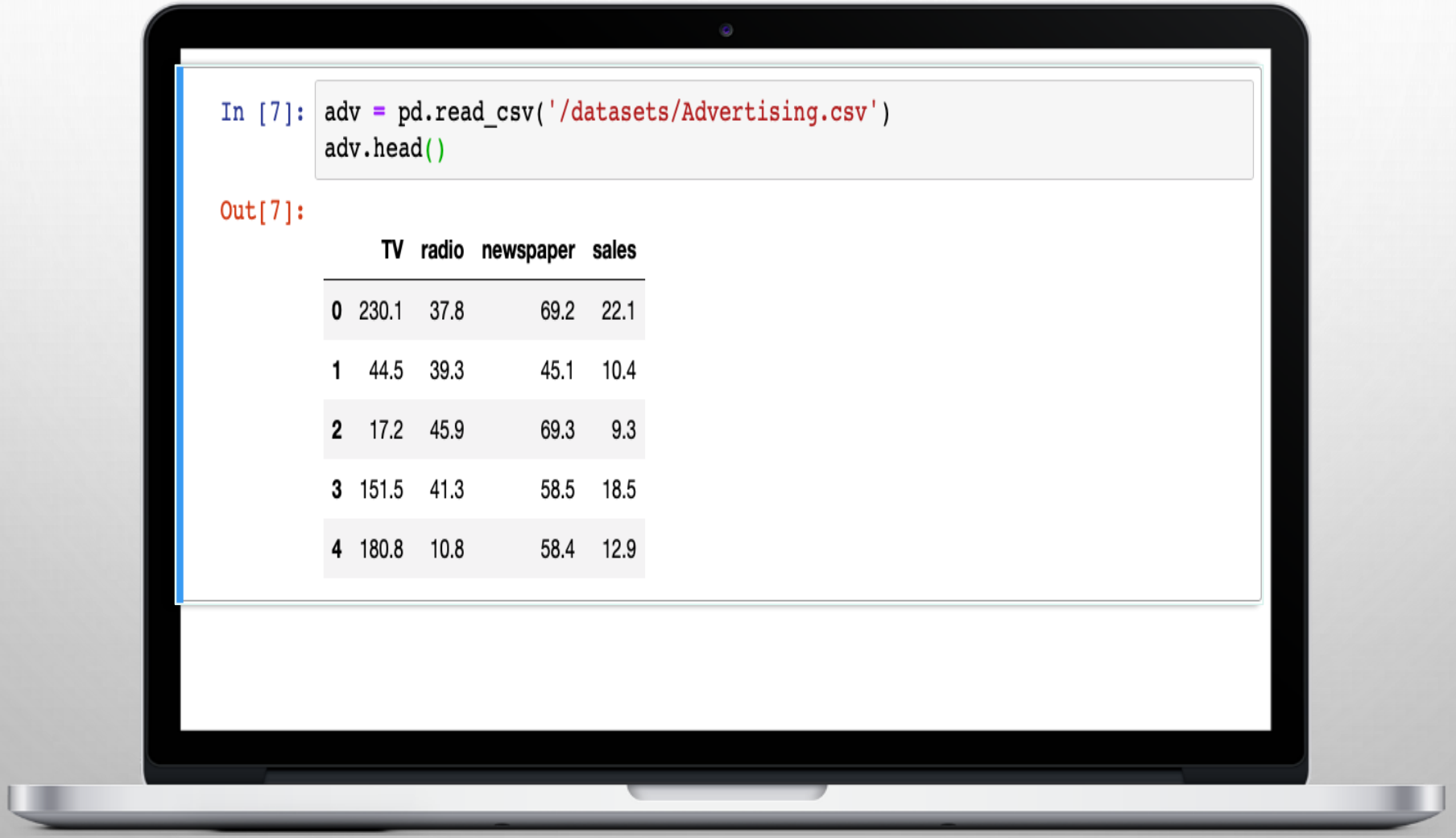
Input features:

- TV Expenses (numerical)
- Newspaper Expenses (numerical)
- Radio Expenses (numerical)

Problem Statement: Given three different expenses predict the sales of the product

By identifying this relationship, the company can change their spending patterns to improve sales

DATA PREVIEW

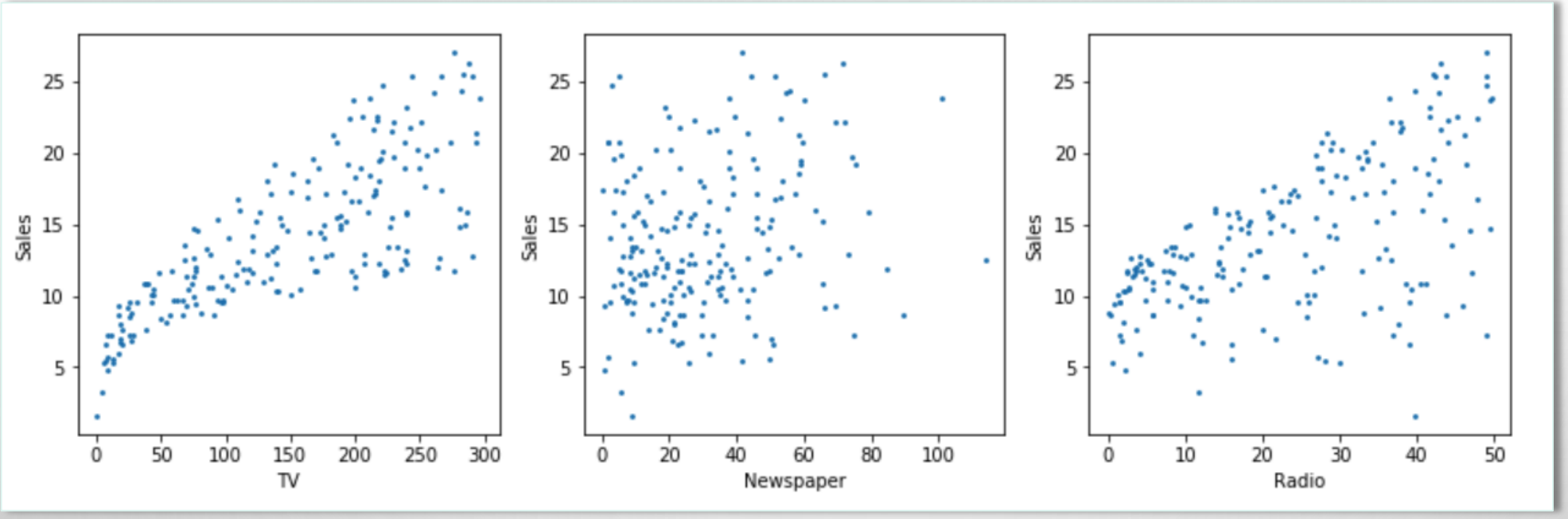


```
In [7]: adv = pd.read_csv('/datasets/Advertising.csv')
        adv.head()
```

Out[7]:

	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	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9

SCOPE FOR PREDICTION



Good linear relationship

Weak linear relationship

Mild linear relationship

CORRELATION MATRIX

Correlation values

- **TV & Sales: 0.78**
- **Radio & Sales: 0.57**
- **Newspaper & Sales: 0.22**

```
In [6]: adv.corr()
```

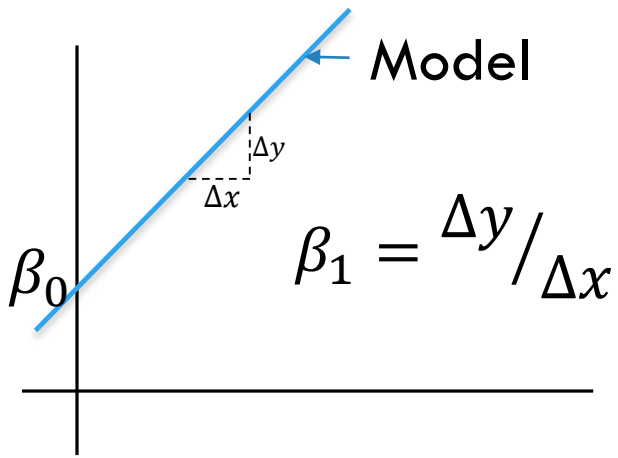
Out[6]:

	TV	radio	newspaper	sales
TV	1.000000	0.054809	0.056648	0.782224
radio	0.054809	1.000000	0.354104	0.576223
newspaper	0.056648	0.354104	1.000000	0.228299
sales	0.782224	0.576223	0.228299	1.000000

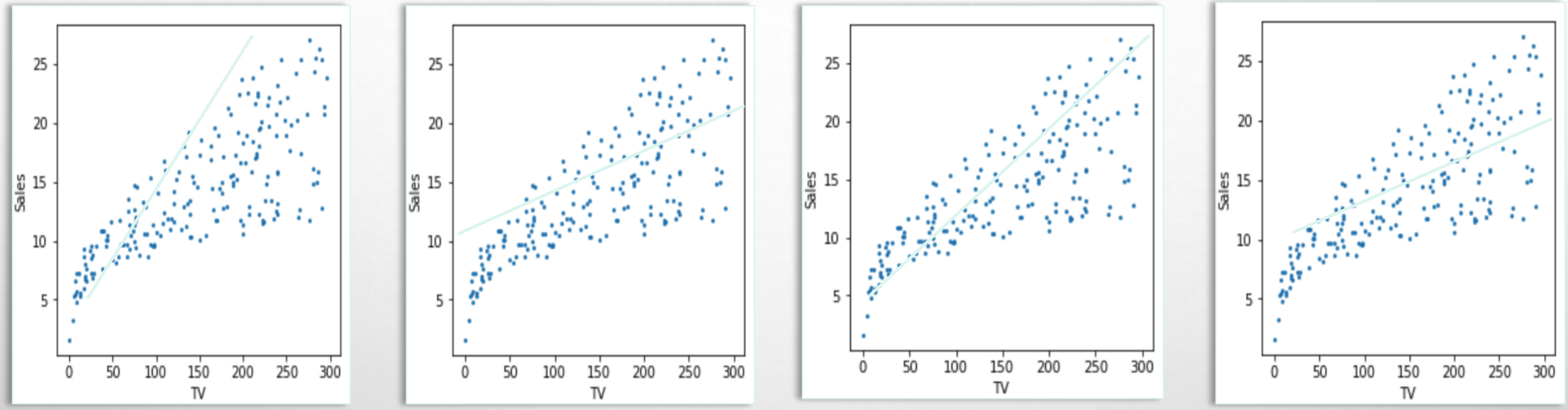
SIMPLE LINEAR REGRESSION

- Simple Linear Regression: When we use only one input feature to predict the target variable
- Linear regression assumes that there is approximately a linear relationship between Y (sales) and X (say TV)
- Representing the assumed relationship mathematically
- β_0 & β_1 are the unknown or model's coefficients that has to be deduced using a training data set
- β_0 is also called as intercept , β_1 is called as slope
- Effectively we need to identify a line which best fits our data

$$Y \approx \beta_0 + \beta_1 X$$
$$\text{sales} \approx \beta_0 + \beta_1 * TV$$



BEST LINE OF FIT



Above plots show four different lines which can explain the relationship between sales and TV

All lines seems to be passing through many data points. Which lines to choose?

Line with best fit: We would like to pick a line which passes through maximum number of data points or the line whose residual error is the least

MOBILE APPS PROJECT FEATURES

01 The co - efficient β_1 and β_0 for the line which best fits the data can be calculated using the below formula

02 This formula is derived from the algorithm “Ordinary Least Squares”

$$\beta_1 = \frac{\sum_{i=1}^{n}(x_i - mean(x))(y_i - mean(y))}{\sum_{i=1}^{n}(x_i - mean(x))^2}$$

$$\beta_0 = mean(y) - \beta_1 \text{mean(x)}$$

03 Where x_i is the input (TV spend) value of the i^{th} sample.

04 Where $mean(x)$ and $mean(y)$ average TV spend and average sales respectively

COEFFICIENTS ESTIMATES

Lets split the data in training and testing

```
In [67]: train_x, test_x, train_y, test_y = train_test_split(adv[['TV']],
                                                            adv['sales'],
                                                            test_size=0.2,
                                                            random_state=100)
         train_x.shape, test_x.shape, train_y.shape, test_y.shape

Out[67]: ((160, 1), (40, 1), (160,), (40,))
```

Using the earlier mentioned formula, calculate the slope and intercept

```
In [69]: mean_x = train_x['TV'].mean()
         mean_y = train_y.mean()
         num_b1 = ((train_x['TV'] - mean_x) * (train_y - mean_y)).sum()
         den_b1 = (np.square(train_x['TV'] - mean_x)).sum()
         b1 = num_b1 / den_b1
         b0 = mean_y - b1 * mean_x
         print('Slope:', b1)
         print('Intercept:', b0)

         Slope: 0.04610975647855533
         Intercept: 7.113008222196274
```

MODEL FITTING USING SCIKIT LEARN

Lets fit a model using scikit-learn library to estimate the coefficients automatically

Initiate a Linear Regression model and pass training input and training output values to fit a model

```
In [70]: from sklearn.linear_model import LinearRegression
         model = LinearRegression()
         model.fit(train_x, train_y)

Out[70]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=Fal
         se)
```

Model's coefficients can be easily extracted from the model after fitting the training data

The coefficients value matches with the manual calculation

```
In [55]: model.coef_, model.intercept_

Out[55]: (array([0.04610976]), 7.113008222196275)
```

Note: Additional 1000 spent on TV advertising in associated with selling 46 additional units of products

PREDICT SALES

Using .predict() function we can very easily predict the sales values for the test data

```
In [71]: model.predict(test_x)

Out[71]: array([ 7.47266432, 18.09635222, 13.3470473 , 17.15110221, 18.25773636,
                16.64850586, 13.53148632, 16.2242961 , 17.09115952, 17.10960343,
                12.51707168, 17.69519733,  9.70437654, 15.77242049, 11.13377899,
                11.45654728, 14.01563877, 14.96088877, 14.65195341, 12.31879973,
                17.01277294, 13.07961071, 16.12285464, 15.27443512, 15.6387022 ,
                17.27098757, 17.2479327 , 10.58507289, 15.6387022 , 12.78911924,
                10.26691557, 10.29458142, 12.69228876, 15.80008634,  9.41849605,
                12.66923388, 11.59487655, 14.87789121, 17.36781806, 16.02141317])
```

Using the available training data, we have identified the best linear relationship between sales and TV expenses incurred

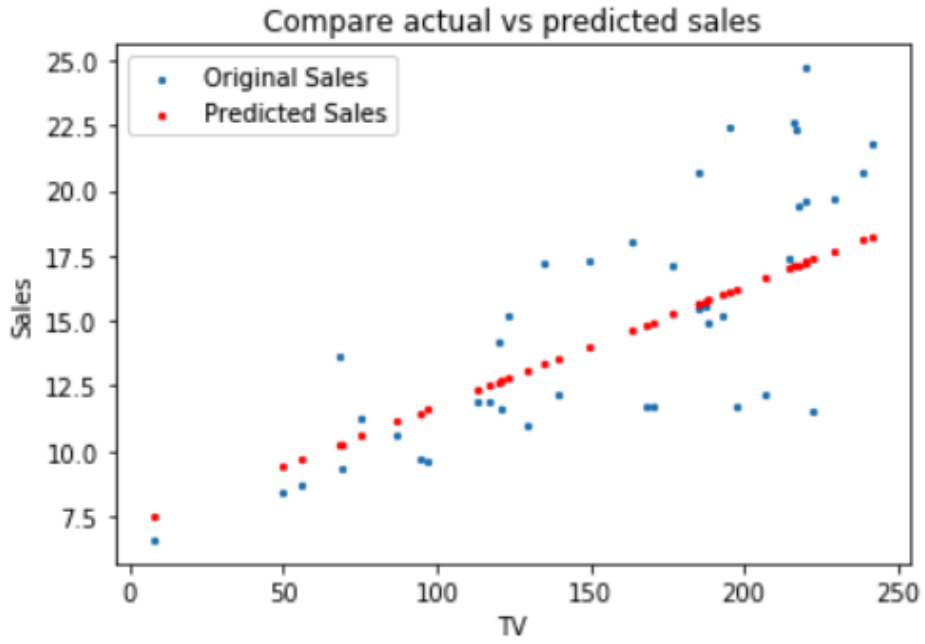
But is our prediction good enough? How do we test them?

Let’s compare the test actual sales and predicted sales values visually and mathematically

COMPARE ACTUAL VS. PREDICTED SALES

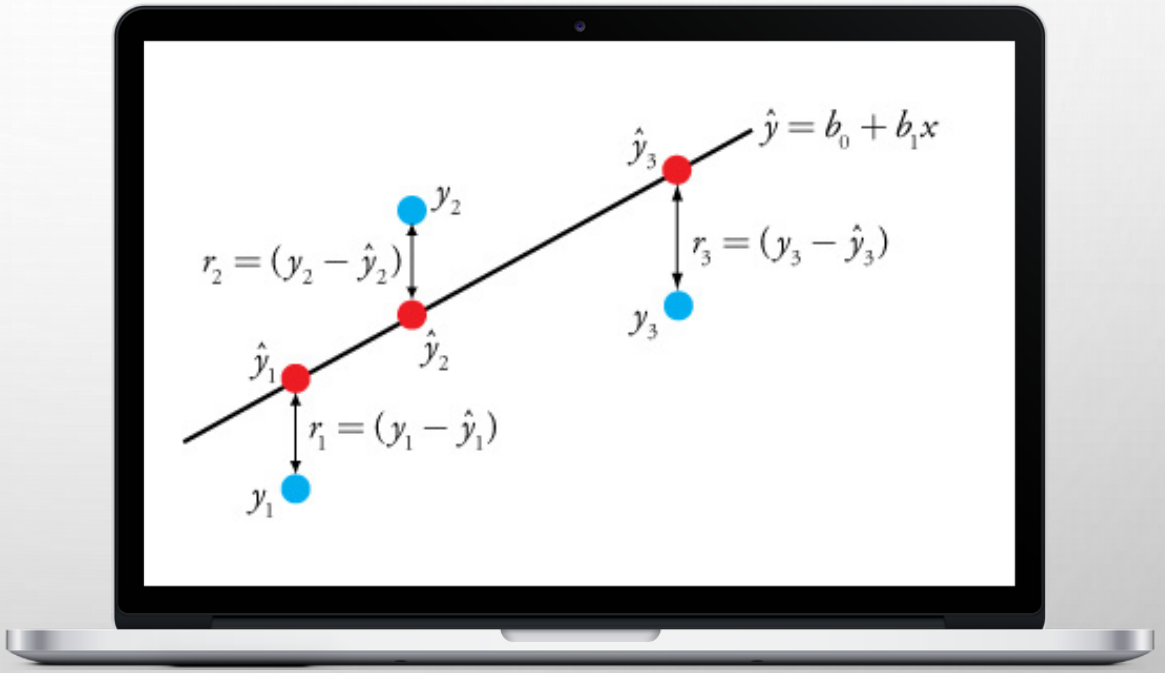
In [89]:
```
test_y_pred = model.predict(test_x)
plt.scatter(test_x['TV'], test_y, s=5)
plt.scatter(test_x['TV'], test_y_pred, color='red', s=5)
plt.xlabel('TV')
plt.ylabel('Sales')
plt.legend(['Original Sales', 'Predicted Sales'])
plt.title('Compare actual vs predicted sales')
```

Out[89]:
```
Text(0.5,1,'Compare actual vs predicted sales')
```



PROJECT SHOWCASE IN LAPTOP

- Residuals are used to compare and evaluate the predicted values
- For each TV expenditure, difference between the actual sales and the predicted sales is called the Residuals or the Prediction Errors
- We have to aggregate all residuals to get an idea about the overall residual
- The quality of a linear regression fit is typically assessed using two related quantities
  - Mean Squared Error (MSE)
  - R – square statistic



MEAN SQUARED ERROR (MSE)

01 If $\hat{y}_i$ is the predicted value of the i^{th} sample and y_i is the corresponding true value, then the Mean Squared Error (MSE) estimated over n samples is defined as follows

02 It is an estimate of variance of the error

03 MSE can be very easily calculated using metrics module from 'sklearn'

04 The value of MSE should be as least as possible. Ideally it is excepted to have zero mean square error

05 Variance of the error in our model is around 9. The difference between actual vs predicted sales is expected to be ± 3 (standard deviation of the error

06 If we have more than one model, we select one model whose MSE is the least

$$MSE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

```
In [90]: from sklearn.metrics import mean_squared_error
         mean_squared_error(test_y, test_y_pred)

Out[90]: 9.687069547890871
```

R SQUARED STATISTIC

- Usually MSE is used to compare across two models. If we have only one model, we cannot properly judge if our model is really good. It is not always clear what constitutes a good MSE.
- The R – Square statistic provides an alternative measure of fit.
- It measures the proportion of variability in Y that can be explained using X
- Always takes values between 0 to 1. A model whose R – Square value is close to 1 is ideally expected

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

Where TSS is the total sum of squares. Measures total variance in the response

$$TSS = \sum_{i=1}^{n} (y_i - mean(y)^2$$

- Where RSS is the residual sum of squares. Measures the amount of variability in residuals

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

R SQUARED STATISTIC

Similar to MSE, we can easily calculate the R squared statistic using scikit measures

```
In [94]: from sklearn.metrics import mean_squared_error, r2_score
         mse = mean_squared_error(test_y, test_y_pred)
         r2_value = r2_score(test_y, test_y_pred)
         print('MSE: ', mse)
         print('R squared:', r2_value)

         MSE:  9.687069547890871
         R squared: 0.5441581483697225
```

The input variable (TV expenses) is able to explain 54% of variance in output variable (sales)

This is much more easier to interpret than MSE

Nevertheless both the measures are widely used together to compare the best model among many

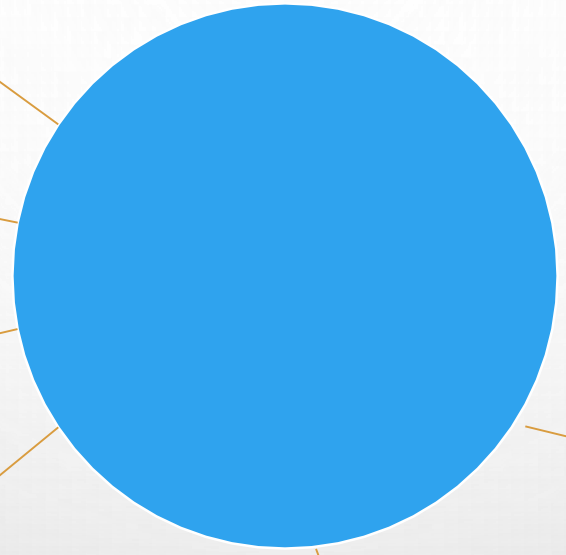
MULTIPLE LINEAR REGRESSION

01 Input feature TV explains 54% variance seen in the output variable i.e. sales

02 What can we do to increase the variance explained by our model

03 Utilize other input features in our model to jointly explain the variance seen in sales

04 When more than one input feature is used to explain the output variable, it is called as Multiple Linear Regression



06 Ordinary Least Squares method is used to identify the coefficients and intercept

05 The mathematical equation for multiple linear regression is expressed as follows

$$Y \approx \beta_0 + \beta_1 X1 + \beta_2 X2 + \beta_3 X3 + \cdots + \beta_p Xp$$

$$\text{sales} \approx \beta_0 + \beta_1 * TV + \beta_2 * Radio + \beta_3 * Newspaper$$

Where β_1, β_2 and β_3 represents association of TV, Radio and Newspaper with sales respectively

IMPLEMENTATION

Drop output column (sales) while passing data as first parameter to **train_test_split()** function

```
In [28]: train_x, test_x, train_y, test_y = train_test_split(adv.drop('sales',
                                                                       axis=1),
                                                             adv['sales'],
                                                             test_size=0.2,
                                                             random_state=100)
         train_x.shape, test_x.shape, train_y.shape, test_y.shape

Out[28]: ((160, 3), (40, 3), (160,), (40,))
```

SK Learn returns coefficients in the same order in which we pass the input features

```
In [37]: mlr = LinearRegression()
         mlr.fit(train_x, train_y)
         print('Input Features:', train_x.columns.values)
         print('Coefficients:', mlr.coef_)
         print('Intercept:', mlr.intercept_)

         Input Features: ['TV' 'radio' 'newspaper']
         Coefficients: [0.0455864  0.18569816 0.00223281]
         Intercept: 2.8172751352950005
```

PREDICTION AND EVALUATION

The MSE value is very less compared to the simple linear regression.

Hence the current model is a better choice compared to the model which uses TV spend alone.

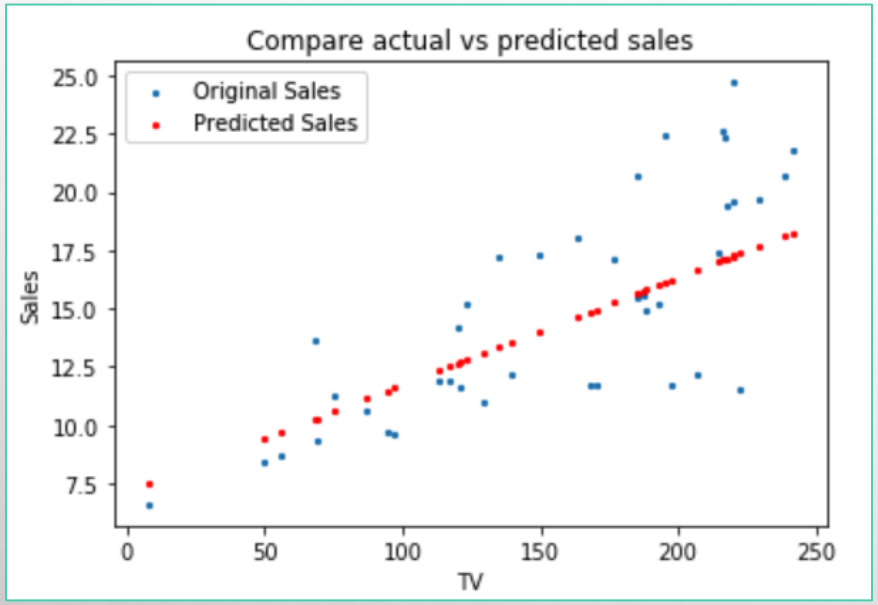
Also the R square value is high and close to one. This model explains 91% variance in sales using all three input predictors

```
In [39]: test_y_pred = mlr.predict(test_x)
         mse = mean_squared_error(test_y, test_y_pred)
         r2_value = r2_score(test_y, test_y_pred)
         print('MSE: ', mse)
         print('R squared:', r2_value)

         MSE:  1.7332927815807762
         R squared: 0.9184369032278495
```

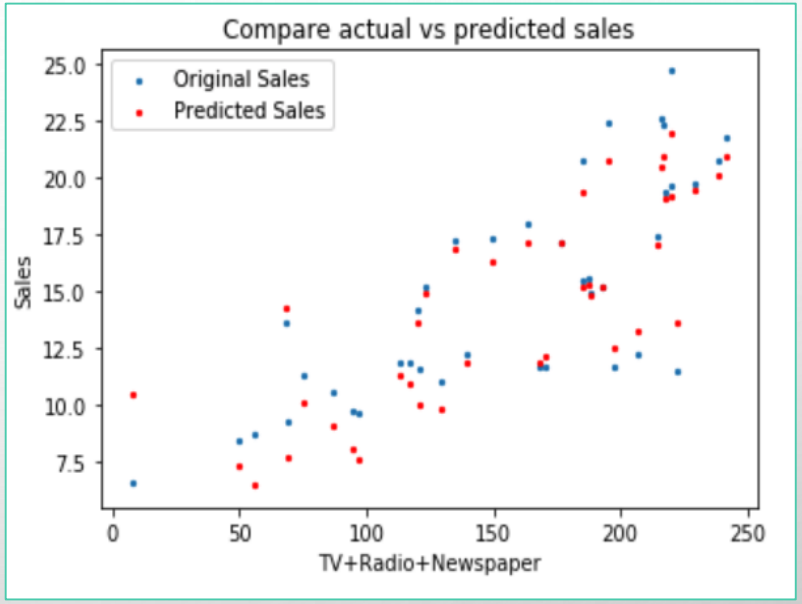
SIMPLE VS. MULTIPLE LINEAR REGRESSION

SINGLE LINEAR REGRESSION



Sales prediction using TV expenses alone. Model is able to capture the trend but residual variance is high

MULTIPLE LINEAR REGRESSION



Sales prediction using TV, Radio & Newspaper expenses. Model is able to capture the trend and also close matches with the actual sales with minimal error

HANDS-ON PRACTICE

Lets Try

- Regression

Workout: Predict the Tips

'tips.csv' dataset contains many factors that influencing the Food servers' tips in a restaurant. In one restaurant, a food server recorded some data on all customers they served during an interval of two and a half months in 1990. This includes the nature of the restaurant, size of the party, gender of the customers and so on.

#'tips.csv' contains 7 features: total_bill, tip, sex, smoker, day, time and size. Using this dataset, you have to perform a predictive analysis and build a model to predict Food servers' tips using total bill amount paid by a customer.