

## FUTURE SALES PREDICTION – DEVOLPMENT PART 1

→ 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:

```

0s ✓ ▶ 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.

```

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

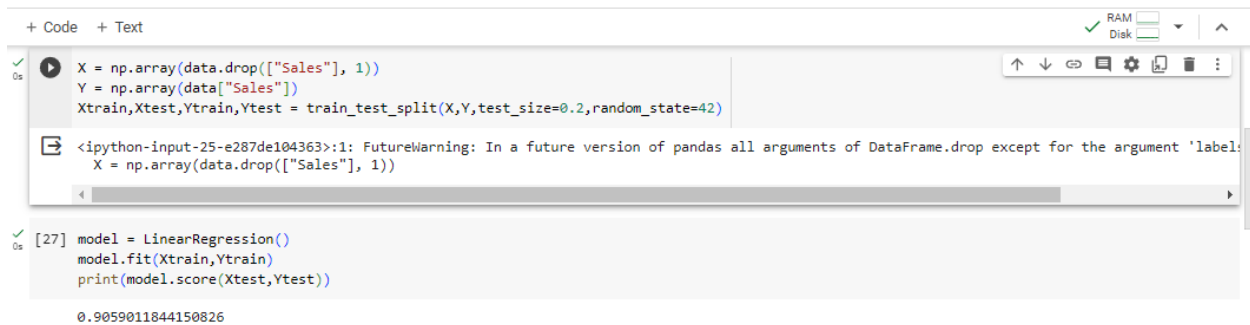
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])

```

### 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:



```
+ Code + Text
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.

```
+ Code + Text

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],
       [-8.74810700e-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],
       [ 1.65900907e+00,  1.31611465e+00,  1.04041841e+00],
       [-1.54975068e+00, -1.88064775e-01, -6.38571189e-01],
       [ 5.65395186e-01, -1.31281155e+00, -1.18610290e+00],
       [ 1.59564140e+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],
       [ 2.88901272e-01,  8.50222156e-01,  2.12657018e+00]]
```

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