Packages Import

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dropout,Dense,LSTM
```

Business & Data Understaning

```
df=pd.read_csv("/content/AirPassengers.csv")
df.head()
```

	Month	#Passengers
0	1949-01	112
1	1949-02	118
2	1949-03	132
3	1949-04	129
4	1949-05	121

```
df.tail()
```

	Month	#Passengers
139	1960-08	606
140	1960-09	508
141	1960-10	461
142	1960-11	390
143	1960-12	432

```
df.shape
```

```
(144, 2)
```

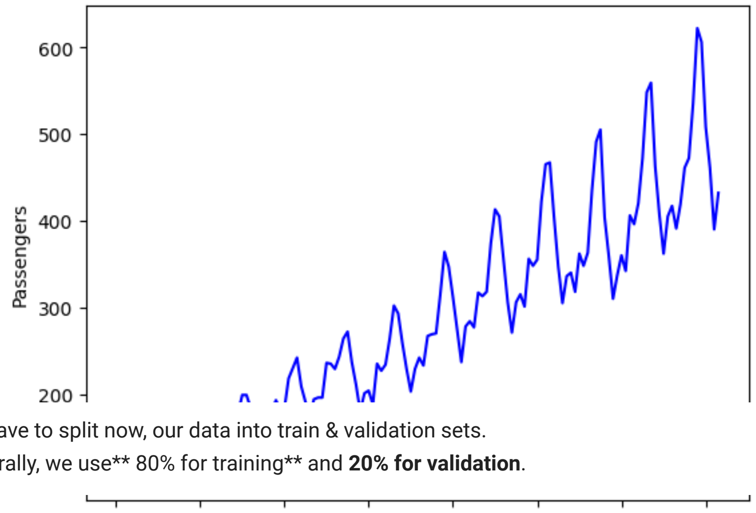
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Month        144 non-null    object
 1   #Passengers  144 non-null    int64 
dtypes: int64(1), object(1)
memory usage: 2.4+ KB
```

- The data ranges from **January 1949** to **December 1960**, or **12 years**, with ***144 observations***;
- No Null Values in this dataset;

Data Preparation

```
plt.xlabel("Months")
plt.ylabel("Passengers")
plt.plot(df['#Passengers'], color='blue')
plt.show()
```



We have to split now, our data into train & validation sets.
Generally, we use** 80% for training** and **20% for validation**.

```
training_size = int(len(df['#Passengers'])*0.8)
training_size
```

```
     115
```

```
def load_data(data, seq_len):
    x = []
    y = []
    for i in range(seq_len, len(data)):
        x.append(data.iloc[i-seq_len : i, 1])
        y.append(data.iloc[i,1])
    return x,y
```

```
x, y = load_data(df, 20)
```

```
len(x)
```

```
     124
```

```
x_train = x[:training_size]
y_train = y[:training_size]
x_test = x[training_size:]
y_test = y[training_size:]
```

```
x_train = np.array(x_train)
y_train = np.array(y_train)
x_test = np.array(x_test)
y_test = np.array(y_test)
```

```
print('x_train.shape = ',x_train.shape)
print('y_train.shape = ', y_train.shape)
print('x_test.shape = ', x_test.shape)
print('y_test.shape = ',y_test.shape)
```

```
     x_train.shape =  (115, 20)
     y_train.shape =  (115,)
     x_test.shape =  (9, 20)
     y_test.shape =  (9,)
```

```
x_train = np.reshape(x_train, (training_size, 20, 1))
x_test = np.reshape(x_test, (x_test.shape[0], 20, 1))
```

```
print('x_train.shape = ',x_train.shape)
print('y_train.shape = ', y_train.shape)
print('x_test.shape = ', x_test.shape)
print('y_test.shape = ',y_test.shape)
```

```
     x_train.shape =  (115, 20, 1)
     y_train.shape =  (115,)
     x_test.shape =  (9, 20, 1)
     y_test.shape =  (9,)
```

Data Modeling

```
model = Sequential()
model.add(LSTM(40, input_shape=(x_train.shape[1],x_train.shape[-1]),return_sequences=True,activation='sigmoid'))
model.add(LSTM(40,activation='sigmoid'))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(x_train, y_train, epochs=500, batch_size=2, verbose=2)
```

```
Epoch 1/500
58/58 - 4s - loss: 92458.0234 - 4s/epoch - 68ms/step
Epoch 2/500
58/58 - 1s - loss: 90803.3906 - 935ms/epoch - 16ms/step
Epoch 3/500
58/58 - 1s - loss: 89753.7812 - 909ms/epoch - 16ms/step
Epoch 4/500
58/58 - 1s - loss: 88857.3359 - 939ms/epoch - 16ms/step
Epoch 5/500
58/58 - 1s - loss: 87997.1094 - 1s/epoch - 24ms/step
Epoch 6/500
58/58 - 1s - loss: 87190.3906 - 1s/epoch - 25ms/step
Epoch 7/500
58/58 - 1s - loss: 86418.1328 - 941ms/epoch - 16ms/step
Epoch 8/500
58/58 - 1s - loss: 85661.4844 - 920ms/epoch - 16ms/step
Epoch 9/500
58/58 - 1s - loss: 84922.3281 - 911ms/epoch - 16ms/step
Epoch 10/500
58/58 - 1s - loss: 84193.1406 - 940ms/epoch - 16ms/step
Epoch 11/500
58/58 - 1s - loss: 83475.1016 - 919ms/epoch - 16ms/step
Epoch 12/500
58/58 - 1s - loss: 82770.1016 - 958ms/epoch - 17ms/step
Epoch 13/500
58/58 - 1s - loss: 82068.5234 - 910ms/epoch - 16ms/step
Epoch 14/500
58/58 - 1s - loss: 81372.0312 - 929ms/epoch - 16ms/step
Epoch 15/500
58/58 - 1s - loss: 80686.7031 - 914ms/epoch - 16ms/step
Epoch 16/500
58/58 - 1s - loss: 80005.6641 - 925ms/epoch - 16ms/step
Epoch 17/500
58/58 - 1s - loss: 79330.2188 - 987ms/epoch - 17ms/step
Epoch 18/500
58/58 - 1s - loss: 78658.8125 - 1s/epoch - 25ms/step
Epoch 19/500
58/58 - 1s - loss: 77994.7422 - 1s/epoch - 24ms/step
Epoch 20/500
58/58 - 1s - loss: 77342.2500 - 915ms/epoch - 16ms/step
Epoch 21/500
58/58 - 1s - loss: 76687.7891 - 917ms/epoch - 16ms/step
Epoch 22/500
58/58 - 1s - loss: 76044.3594 - 925ms/epoch - 16ms/step
Epoch 23/500
58/58 - 1s - loss: 75397.8984 - 920ms/epoch - 16ms/step
Epoch 24/500
58/58 - 1s - loss: 74762.0547 - 921ms/epoch - 16ms/step
Epoch 25/500
58/58 - 1s - loss: 74128.3906 - 917ms/epoch - 16ms/step
Epoch 26/500
58/58 - 1s - loss: 73499.5156 - 920ms/epoch - 16ms/step
Epoch 27/500
58/58 - 1s - loss: 72877.7266 - 916ms/epoch - 16ms/step
Epoch 28/500
58/58 - 1s - loss: 72262.5469 - 899ms/epoch - 16ms/step
Epoch 29/500
58/58 - 1s - loss: 71647.0859 - 931ms/epoch - 16ms/step
```

```
y_pred=model.predict(x_test)
```

```
1/1 [==============================] - 0s 386ms/step
```

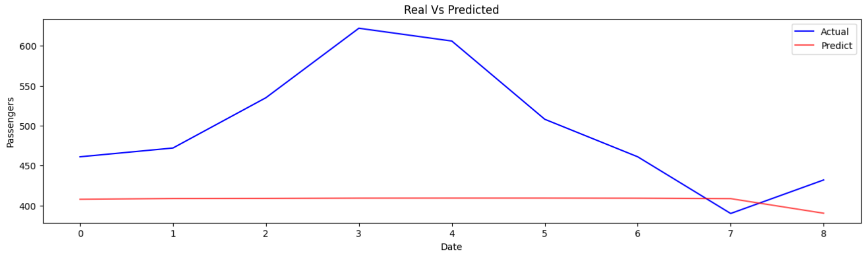
```
y_pred
```

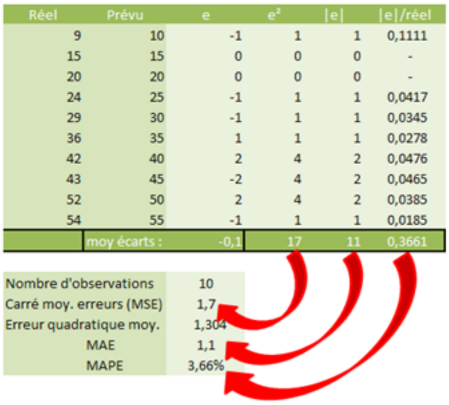
```
array([[407.79535],
       [408.75543],
       [408.88037],
       [409.24603],
       [409.3068 ],
       [409.30347],
       [409.16528],
       [408.60483],
       [390.31732]], dtype=float32)
```

```
y_test
```

```
array([461, 472, 535, 622, 606, 508, 461, 390, 432])
```

```
plt.figure(figsize=(16,4))
plt.plot(y_test, color='blue',label='Actual')
plt.plot(y_pred, alpha=0.7, color='red',label='Predict')
plt.title('Real Vs Predicted')
plt.xlabel('Date')
plt.ylabel('Passengers')
plt.legend()
plt.show()
```





Réel	Prévu	e	e²	|e|	|e|/réel
9	10	-1	1	1	0,1111
15	15	0	0	0	-
20	20	0	0	0	-
24	25	-1	1	1	0,0417
29	30	-1	1	1	0,0345
36	35	1	1	1	0,0278
42	40	2	4	2	0,0476
43	45	-2	4	2	0,0465
52	50	2	4	2	0,0385
54	55	-1	1	1	0,0185
	moy écarts :	-0,1	17	11	0,3661

Nombre d'observations	10
Carré moy. erreurs (MSE)	1,7
Erreur quadratique moy.	1,304
MAE	1,1
MAPE	3,66%

```
from sklearn.metrics import mean_squared_error
rmse = np.sqrt(mean_squared_error(y_test, y_pred)).round(2)
mape = np.round(np.mean(np.abs(y_test-y_pred)/y_test)*100,2)
```

```
rmse
mape
```

```
17.64
```