

“SALES FORECASTING USING LINEAR REGRESSION”

ABSTRACT:

An important part of present-day business intelligence is sales prediction. Sales prediction can be termed a complex problem, and it gets harder in the case of lack of data or missing data values, and the presence of outliers. Sales prediction is more of a regression problem than time series. Using machine learning algorithms, we can find complicated patterns in the sales dynamics including various risk variables as well, using supervised machine learning methods. Sales forecasting plays a huge role in a company's success. An accurate sales prediction model can help businesses find potential risks and make better knowledgeable decisions. An accurate sales prediction can benefit a business by helping save money on excess inventory, planning properly for the future, and increasing the profit earned. Thus, it is also important to evaluate the model using statistical methods like RMSE and MAPE. The results are used in understanding which is a more suitable classifier for sales prediction.

INTRODUCTION

Sales forecasting, a method that predicts sales performance based on historical performance, is one way to get this understanding. Sales forecasting is important because it can help you identify what is going right, as well as what areas of your current strategy need to be adapted and changed to ensure future success.

For example, if your team is consistently below quotas, sales forecasting can help determine where and why these issues are happening. Forecasting can also help you decide on future business endeavors, like when you'd have the revenue to invest in new products or expand your business.

Some forecasting methods involve doing basic math, like adding up month to month sales, and others are more in-depth. Regression analysis is one of these methods, and it requires in-depth statistical analysis.

If you're anything like me and not at all mathematically inclined, conducting this type of forecast may seem daunting. Thankfully, this piece will give an easy to understand breakdown of regression analysis in sales.

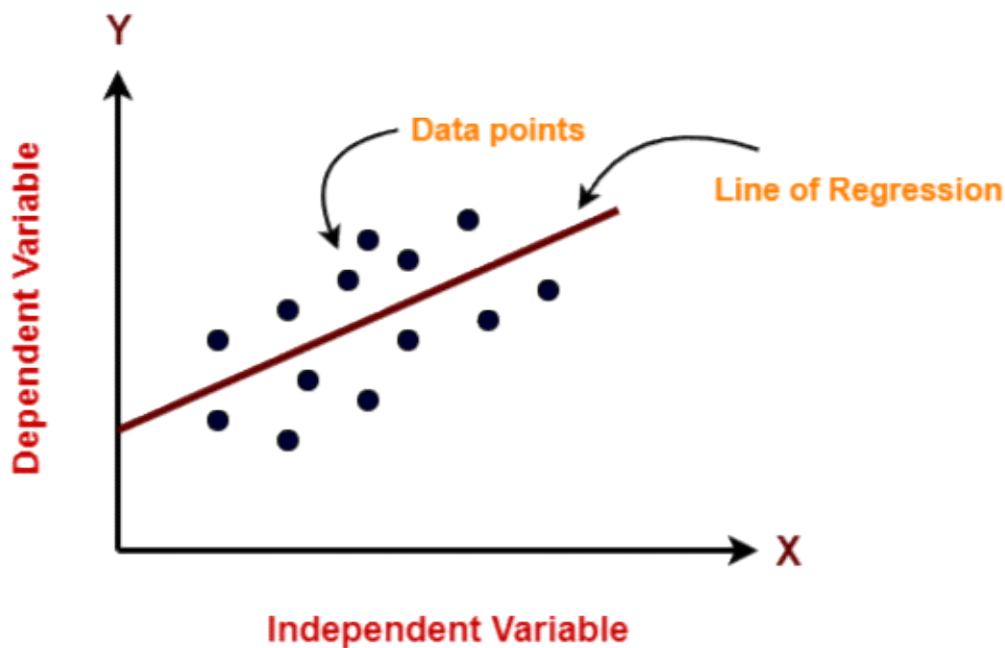
PROPOSED METHODOLOGY

First, all inputs are preprocessed to be understandable by the machine. This is a linear regression model based on supervised learning, so the output will be provided along with the input. Then inputs are then fed to the model along with desired output. The model will plot(learn) a relation(function) between the input and output. This function or relation is then used to predict the output for a specific set of inputs. In this case, input parameters like date and previous sales are labeled as input, and the amount of sales is marked as output. The model will predict a number between 0 and 1 as a sigmoid function is used in the last layer. This output can be multiplied by a specific number(in this case, maximum sales), this will be our corresponding sales amount for a certain day. This output is then provided as input to calculate sales data for the next day. This cycle of steps will be continued until a certain date arrives.

ALGORITHM

Linear Regression:

Linear Regression is an ML algorithm used for supervised learning. Linear regression performs the task to predict a dependent variable(target) based on the given independent variable(s). So, this regression technique finds out a linear relationship between a dependent variable and the other given independent variables. Hence, the name of this algorithm is Linear Regression.



In the figure above, on X-axis is the independent variable and on Y-axis is the output. The regression line is the best fit line for a model. And our main objective in this algorithm is to find this best fit line.

Linear Regression can also be represented as:

$y = a_0 + a_1x + \epsilon$ Here,

Y= Target Variable

X= predictor Variable

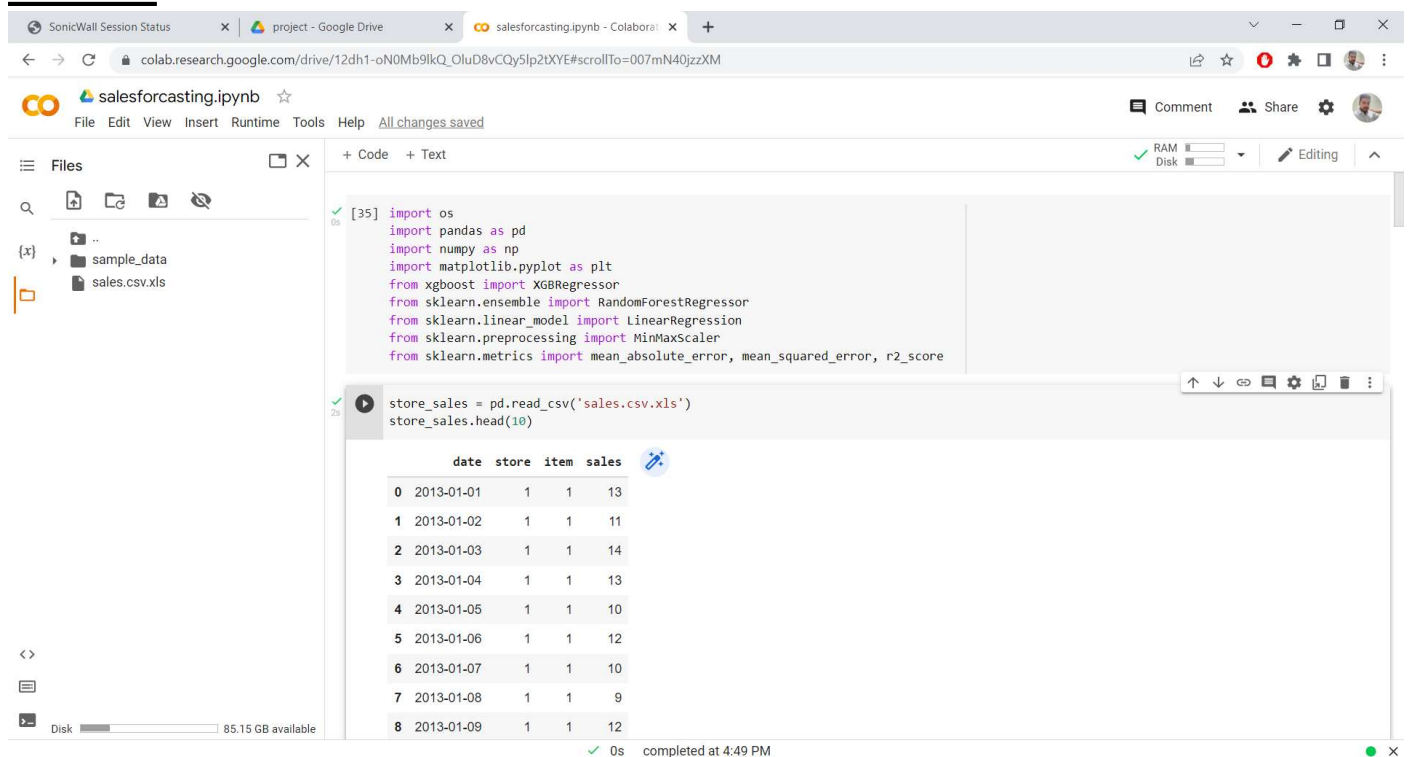
a_0 = intercept of the line

a_1 = coefficient of Linear regression

ϵ = error

x and y variables are datasets values for Linear Regression model

SOURCE CODE



The screenshot shows a Google Colab notebook interface. The left sidebar displays the file explorer with a folder named 'sample_data' containing a file 'sales.csv.xls'. The main code area contains two cells. The first cell, labeled '[35]', imports various libraries: `import os, pandas as pd, numpy as np, matplotlib.pyplot as plt`, and then imports `XGBRegressor` from `xgboost`, `RandomForestRegressor` from `sklearn.ensemble`, `LinearRegression` from `sklearn.linear_model`, `MinMaxScaler` from `sklearn.preprocessing`, and `mean_absolute_error, mean_squared_error, r2_score` from `sklearn.metrics`. The second cell, labeled '28', executes `store_sales = pd.read_csv('sales.csv.xls')` and `store_sales.head(10)`. Below the code, a preview of the first 10 rows of the dataset is shown as a table.

	date	store	item	sales
0	2013-01-01	1	1	13
1	2013-01-02	1	1	11
2	2013-01-03	1	1	14
3	2013-01-04	1	1	13
4	2013-01-05	1	1	10
5	2013-01-06	1	1	12
6	2013-01-07	1	1	10
7	2013-01-08	1	1	9
8	2013-01-09	1	1	12

At the bottom of the notebook, a status bar indicates '0s completed at 4:49 PM'.

SonicWall Session Status

project - Google Drive

salesforecasting.ipynb - Colabora

+

colab.research.google.com/drive/12dh1-oN0Mb9IkQ_OluD8vCQy5lp2tXYE#scrollTo=007mN40jzzXM

salesforecasting.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk Editing

+ Code + Text

[36] store_sales.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 913000 entries, 0 to 912999
Data columns (total 4 columns):
Column Non-Null Count Dtype

0 date 913000 non-null object
1 store 913000 non-null int64
2 item 913000 non-null int64
3 sales 913000 non-null int64
dtypes: int64(3), object(1)
memory usage: 27.9+ MB

[37] store_sales = store_sales.drop(['store','item'], axis=1)
store_sales.head(10)

	date	sales
0	2013-01-01	13
1	2013-01-02	11
2	2013-01-03	14
3	2013-01-04	13
4	2013-01-05	10
5	2013-01-06	12
6	2013-01-07	10
7	2013-01-08	9

0s completed at 4:49 PM

SonicWall Session Status

project - Google Drive

salesforecasting.ipynb - Colabora

+

colab.research.google.com/drive/12dh1-oN0Mb9IkQ_OluD8vCQy5lp2tXYE#scrollTo=007mN40jzzXM

salesforecasting.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk Editing

+ Code + Text

[37] 5 2013-01-06 12
6 2013-01-07 10
7 2013-01-08 9
8 2013-01-09 12
9 2013-01-10 9

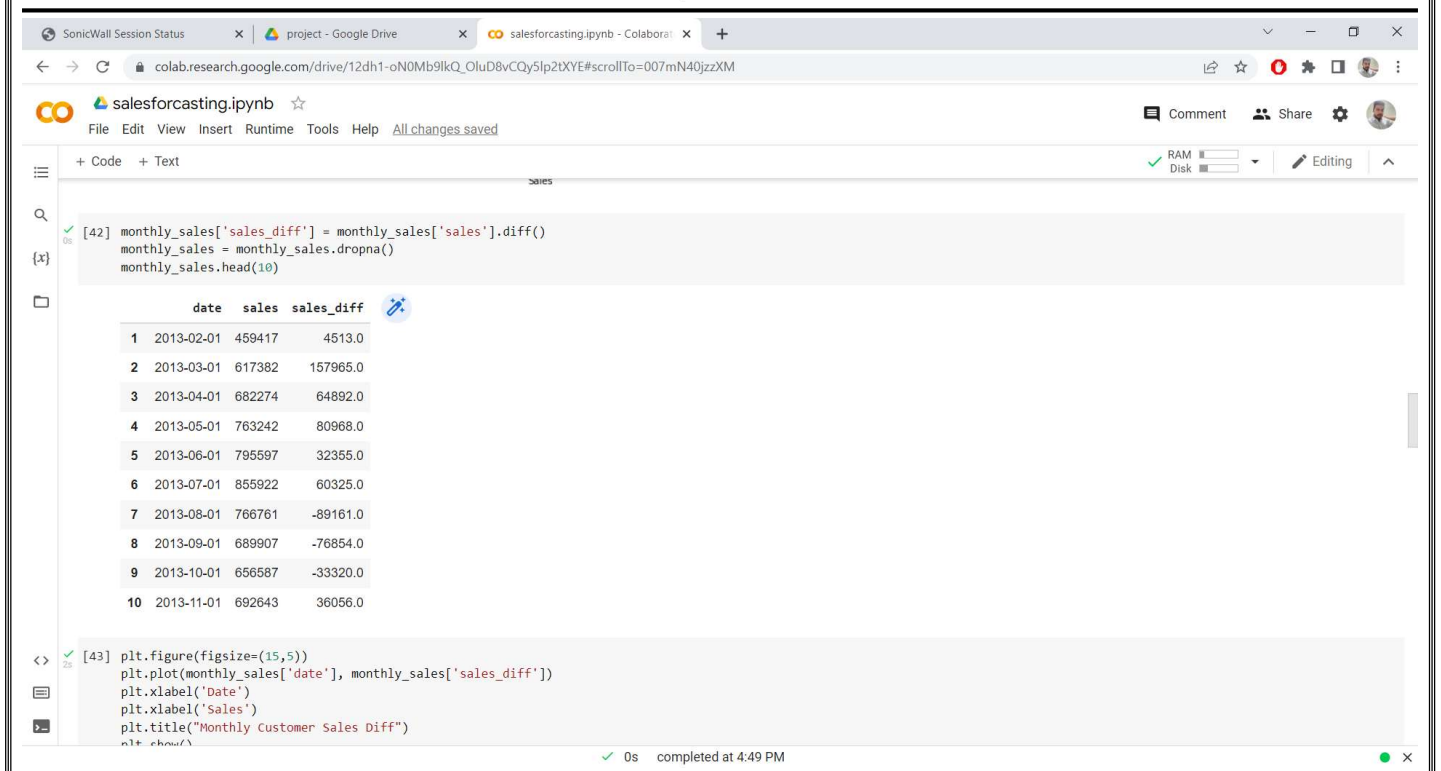
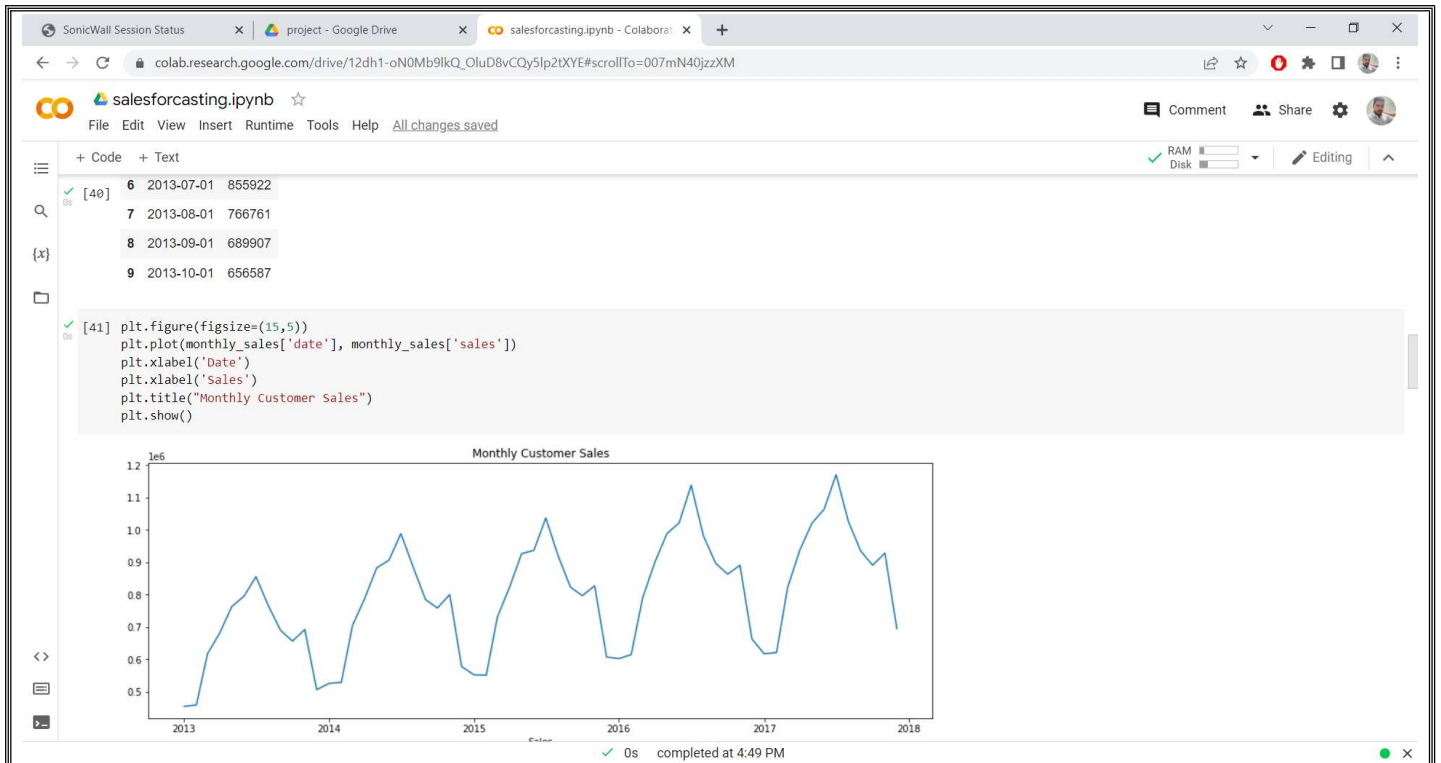
[38] store_sales['date'] = pd.to_datetime(store_sales['date'])

[39] store_sales['date'] = store_sales['date'].dt.to_period('M')
monthly_sales = store_sales.groupby('date').sum().reset_index()

[40] monthly_sales['date'] = monthly_sales['date'].dt.to_timestamp()
monthly_sales.head(10)

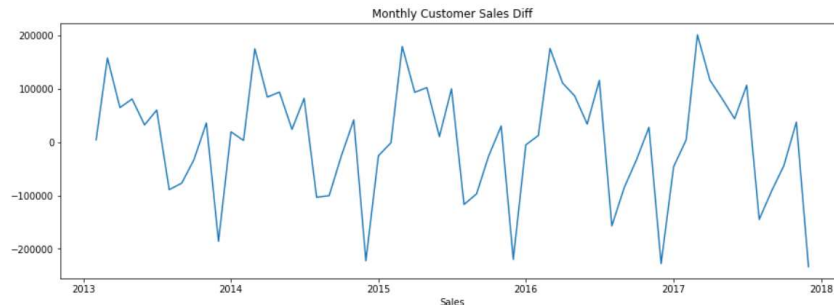
	date	sales
0	2013-01-01	454904
1	2013-02-01	459417
2	2013-03-01	617382
3	2013-04-01	682274
4	2013-05-01	763242
5	2013-06-01	795597

0s completed at 4:49 PM



+ Code + Text

```
[43] plt.figure(figsize=(15,5))
plt.plot(monthly_sales['date'], monthly_sales['sales_diff'])
plt.xlabel('date')
plt.xlabel('Sales')
plt.title("Monthly Customer Sales Diff")
plt.show()
```



```
[44] supervised_data = monthly_sales.drop(['date','sales'], axis=1)
```

```
[45] for i in range(1,13):
    col_name = 'month_' + str(i)
```

0s completed at 4:49 PM

+ Code + Text

```
[45] for i in range(1,13):
    col_name = 'month_' + str(i)
    supervised_data[col_name] = supervised_data['sales_diff'].shift(i)
supervised_data = supervised_data.dropna().reset_index(drop=True)
supervised_data.head(10)
```

	sales_diff	month_1	month_2	month_3	month_4	month_5	month_6	month_7	month_8	month_9	month_10	month_11	month_12
0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0	-89161.0	60325.0	32355.0	80968.0	64892.0	157965.0	4513.0
1	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0	-89161.0	60325.0	32355.0	80968.0	64892.0	157965.0
2	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0	-89161.0	60325.0	32355.0	80968.0	64892.0
3	93963.0	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0	-89161.0	60325.0	32355.0	80968.0
4	23965.0	93963.0	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0	-89161.0	60325.0	32355.0
5	82168.0	23965.0	93963.0	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0	-89161.0	60325.0
6	-103414.0	82168.0	23965.0	93963.0	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0	-89161.0
7	-100472.0	-103414.0	82168.0	23965.0	93963.0	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0	-76854.0
8	-26241.0	-100472.0	-103414.0	82168.0	23965.0	93963.0	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0	-33320.0
9	41900.0	-26241.0	-100472.0	-103414.0	82168.0	23965.0	93963.0	84613.0	175184.0	3130.0	19380.0	-186036.0	36056.0

```
[46] train_data = supervised_data[:-12]
test_data = supervised_data[-12:]
print('Train Data Shape:', train_data.shape)
print('Test Data Shape:', test_data.shape)
```

0s completed at 4:49 PM

SonicWall Session Status

project - Google Drive

salesforecasting.ipynb - Colabora

+

colab.research.google.com/drive/12dh1-oN0Mb9IkQ_OluD8vCQy5lp2tXYE#scrollTo=007mN40jzzXM

salesforecasting.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk

Editing

+ Code + Text

[46] Train Data Shape: (35, 13)
Test Data Shape: (12, 13)

scaler = MinMaxScaler(feature_range=(-1,1))
scaler.fit(train_data)
train_data = scaler.transform(train_data)
test_data = scaler.transform(test_data)

[48] X_train, y_train = train_data[:,1:], train_data[:,0:1]
X_test, y_test = test_data[:,1:], test_data[:,0:1]
y_train = y_train.ravel()
y_test = y_test.ravel()
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: (35, 12)
y_train Shape: (35,)
X_test Shape: (12, 12)
y_test Shape: (12,)

[49] sales_dates = monthly_sales['date'][-12:].reset_index(drop=True)
predict_df = pd.DataFrame(sales_dates)

[50] act_sales = monthly_sales['sales'][-13:].to_list()

0s completed at 4:49 PM

SonicWall Session Status x project - Google Drive x salesforecasting.ipynb - Colabora x +

colab.research.google.com/drive/12dh1-oN0Mb9IkQ_OluD8vCQy5lp2tXYE#scrollTo=007mN40jzzXM

salesforecasting.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[49] sales_dates = monthly_sales['date'][-12:].reset_index(drop=True)
predict_df = pd.DataFrame(sales_dates)

[50] act_sales = monthly_sales['sales'][-13:].to_list()

Forecast Sales using Linear Regression

[51] linreg_model = LinearRegression()
linreg_model.fit(X_train, y_train)
linreg_pred = linreg_model.predict(X_test)

[52] linreg_pred = linreg_pred.reshape(-1,1)
linreg_pred_test_set = np.concatenate([linreg_pred,X_test], axis=1)
linreg_pred_test_set = scaler.inverse_transform(linreg_pred_test_set)

[53] result_list = []
for index in range(0, len(linreg_pred_test_set)):
    result_list.append(linreg_pred_test_set[index][0] + act_sales[index])
linreg_pred_series = pd.Series(result_list,name='linreg_pred')
predict_df = predict_df.merge(linreg_pred_series, left_index=True, right_index=True)

[54] linreg_rmse = np.sqrt(mean_squared_error(predict_df['linreg_pred'], monthly_sales['sales'][-12:]))
linreg_mae = mean_absolute_error(predict_df['linreg_pred'], monthly_sales['sales'][-12:]))
linreg_r2 = r2_score(predict_df['linreg_pred'], monthly_sales['sales'][-12:]))
print('Linear Regression RMSE: ', linreg_rmse)
```

0s completed at 4:49 PM

SonicWall Session Status x project - Google Drive x salesforecasting.ipynb - Colabora x +

colab.research.google.com/drive/12dh1-oN0Mb9IkQ_OluD8vCQy5lp2tXYE#scrollTo=007mN40jzzXM

salesforecasting.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing


```
[53] result_list = []
for index in range(0, len(linreg_pred_test_set)):
    result_list.append(linreg_pred_test_set[index][0] + act_sales[index])
linreg_pred_series = pd.Series(result_list,name='linreg_pred')
predict_df = predict_df.merge(linreg_pred_series, left_index=True, right_index=True)

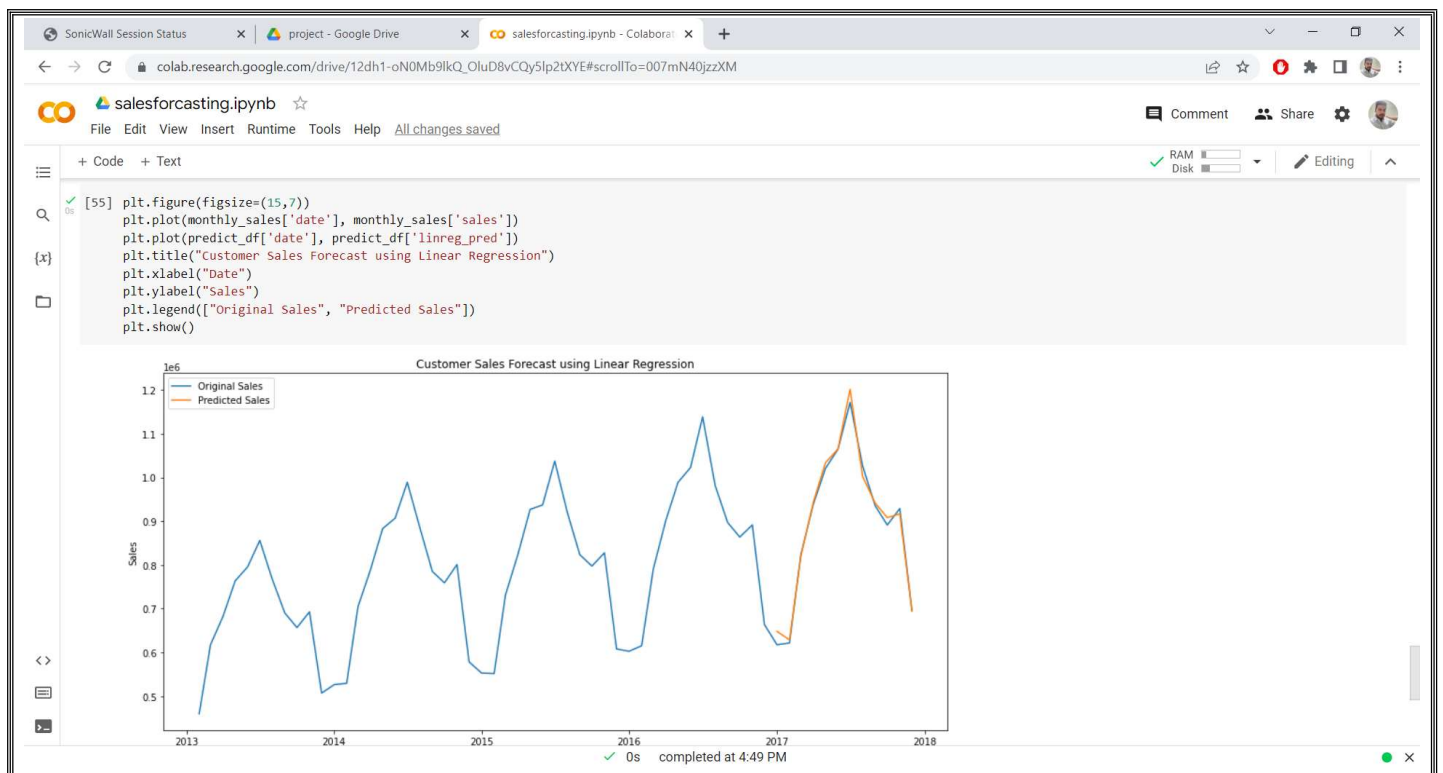
[54] linreg_rmse = np.sqrt(mean_squared_error(predict_df['linreg_pred'], monthly_sales['sales'][-12:]))
linreg_mae = mean_absolute_error(predict_df['linreg_pred'], monthly_sales['sales'][-12:]))
linreg_r2 = r2_score(predict_df['linreg_pred'], monthly_sales['sales'][-12:]))
print('Linear Regression RMSE: ', linreg_rmse)
print('Linear Regression MAE: ', linreg_mae)
print('Linear Regression R2 Score: ', linreg_r2)

Linear Regression RMSE: 16221.272385416898
Linear Regression MAE: 12433.184266490765
Linear Regression R2 Score: 0.9906152516380968

[55] plt.figure(figsize=(15,7))
plt.plot(monthly_sales['date'], monthly_sales['sales'])
plt.plot(predict_df['date'], predict_df['linreg_pred'])
plt.title("Customer Sales Forecast using Linear Regression")
plt.xlabel("Date")
plt.ylabel("Sales")
plt.legend(["Original Sales", "Predicted Sales"])
plt.show()
```

0s completed at 4:49 PM





ADVANTAGES AND DISADVANTAGES OF PROPOSED METHODOLOGY:

Pros:

Linear Regression is simple to implement.

Less complexity compared to other algorithms.

Linear Regression may lead to over-fitting but it can be avoided using some dimensionality reduction techniques, regularization techniques, and cross-validation.

Cons:

Outliers affect this algorithm badly.

It over-simplifies real-world problems by assuming a linear relationship among the variables, hence not recommended for practical use-cases.

CONCLUSION

In this project, basics of machine learning and the associated data processing and modelling algorithms are described, and their application in predicting sales. The implementation, show the correlation among different attributes considered and how a particular location of medium size recorded the highest sales, suggesting that other shopping locations should follow similar patterns for improved sales. Multiple instances parameters and various other factors can also be used for predicting the sales more innovatively and successfully. Accuracy plays a major role in prediction systems, can be significantly increased when the parameters used are increased. Also how the sub-models work also can lead to improving the productivity of the system.

As the profit made is directly proportional to the sales predictions made accurately, the Big marts aim accurate predictions so that the company will not suffer any losses

