

Name – Shruti Kumari
Reg.No.- 221FA14031
Sec – 'A' BI

LINEAR PLOT
REGRESSION BETWEEN
RV1 AND T1

INTRODUCTION TO LINEAR REGRESSION

- A statistical method used to model the relationship between a dependent variable and one or more independent variables.
 - The objective is to find the best-fit line that minimizes the distance between the predicted and actual values.
 - In this case, the relationship between rv1 (independent) and T1 (dependent) will be explored.
-

UNDERSTANDING THE DATASET

linear regression.ipynb

```
import pandas as pd

df = pd.read_excel('/content/data_application_energy.xlsx')
```

df

	date	Appliances	lights	T1	RH_1	T2	RH_2	T3	RH_3	T4	...	T9	RH_9	T_out	Press_mm_hg	RH_10
0	2016-01-11 17:00:00	60	30	19.890000	47.596667	19.200000	44.790000	19.790000	44.730000	19.000000	...	17.033333	45.5300	6.600000	733.5	92.000
1	2016-01-11 17:10:00	60	30	19.890000	46.693333	19.200000	44.722500	19.790000	44.790000	19.000000	...	17.066667	45.5600	6.483333	733.6	92.000
2	2016-01-11 17:20:00	50	30	19.890000	46.300000	19.200000	44.626667	19.790000	44.933333	18.926667	...	17.000000	45.5000	6.366667	733.7	92.000
3	2016-01-11 17:30:00	50	40	19.890000	46.066667	19.200000	44.590000	19.790000	45.000000	18.890000	...	17.000000	45.4000	6.250000	733.8	92.000

Connected to Python 3 Google Compute Engine backend

- Columns in focus:
- rv1: Independent variable (X)
- T1: Dependent variable (y)
- We're interested in finding how rv1 affects T1.

PYTHON CODE BREAKDOWN

1	2016-01-11 17:10:00	60	30	19.890000	46.693333	19.200000	44.722500	19.790000	44.790000	19.000000	...	17.066667	45.5600	6.483333	733.6	92.000
2	2016-01-11 17:20:00	50	30	19.890000	46.300000	19.200000	44.626667	19.790000	44.933333	18.926667	...	17.000000	45.5000	6.366667	733.7	92.000
3	2016-01-11 17:30:00	50	40	19.890000	46.066667	19.200000	44.590000	19.790000	45.000000	18.890000	...	17.000000	45.4000	6.250000	733.8	92.000
4	2016-01-11 17:40:00	60	40	19.890000	46.333333	19.200000	44.530000	19.790000	45.000000	18.890000	...	17.000000	45.4000	6.133333	733.9	92.000
...
19730	2016-05-27 17:20:00	100	0	25.566667	46.560000	25.890000	42.025714	27.200000	41.163333	24.700000	...	23.200000	46.7900	22.733333	755.2	55.666
19731	2016-05-27 17:30:00	90	0	25.500000	46.500000	25.754000	42.080000	27.133333	41.223333	24.700000	...	23.200000	46.7900	22.600000	755.2	56.000
19732	2016-05-27 17:40:00	270	10	25.500000	46.596667	25.628571	42.768571	27.050000	41.690000	24.700000	...	23.200000	46.7900	22.466667	755.2	56.333

```
import matplotlib.pyplot as plt
```

```
import numpy as np
```

```
from sklearn.linear_model import LinearRegression
```

```
# Create a linear regression model
```

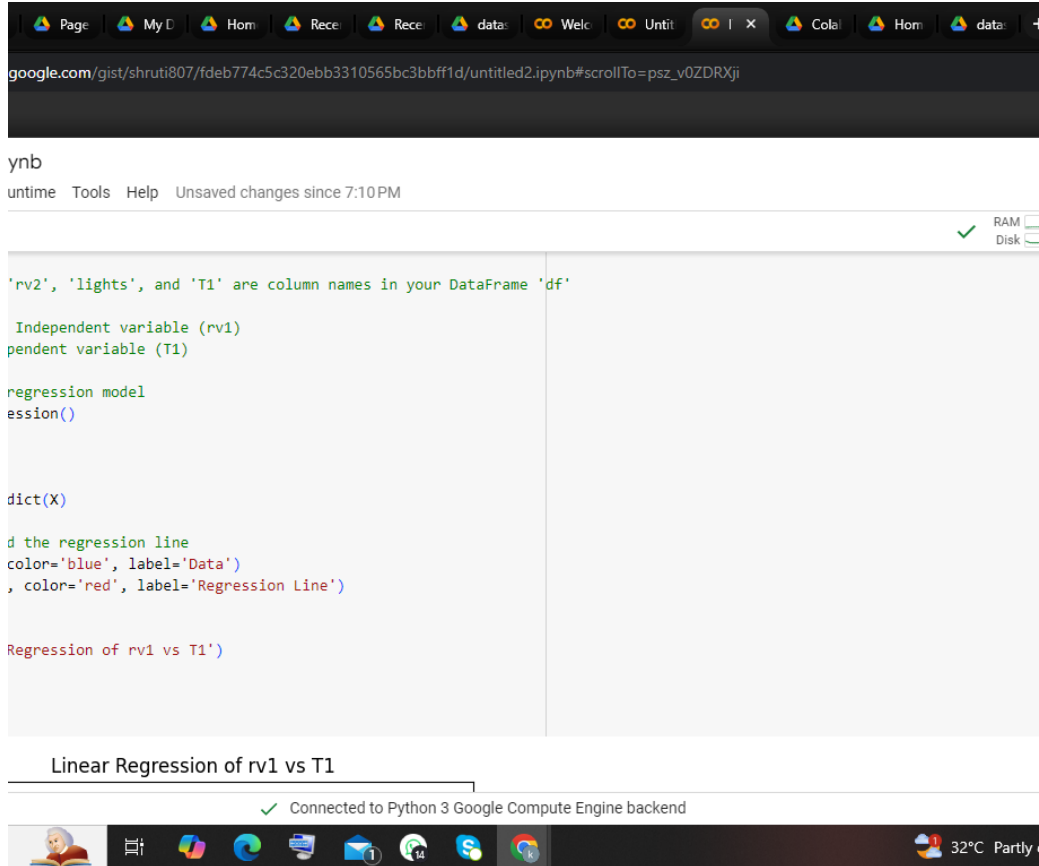
```
model = LinearRegression()
```

```
model.fit(X, y)
```

```
# Make predictions
```

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

PYTHON CODE FOR VISUALIZATION



The screenshot shows a Jupyter Notebook interface with a dark theme. The browser address bar at the top displays a Google Gist URL. The notebook's toolbar includes options like 'untime', 'Tools', 'Help', and a status message 'Unsaved changes since 7:10 PM'. The code editor contains the following Python code:

```
'rv2', 'lights', and 'T1' are column names in your DataFrame 'df'  
  
Independent variable (rv1)  
pendent variable (T1)  
  
regression model  
ession()  
  
dict(X)  
  
d the regression line  
color='blue', label='Data')  
, color='red', label='Regression Line')  
  
Regression of rv1 vs T1)
```

Below the code editor, a plot titled 'Linear Regression of rv1 vs T1' is displayed. The status bar at the bottom indicates 'Connected to Python 3 Google Compute Engine backend' and shows system icons for temperature (32°C) and weather (Partly cl).

```
# Plot the data and the regression line
```

```
plt.scatter(X, y, color='blue', label='Data')
```

```
plt.plot(X, y_pred, color='red', label='Regression Line')
```

```
plt.xlabel('rv1')
```

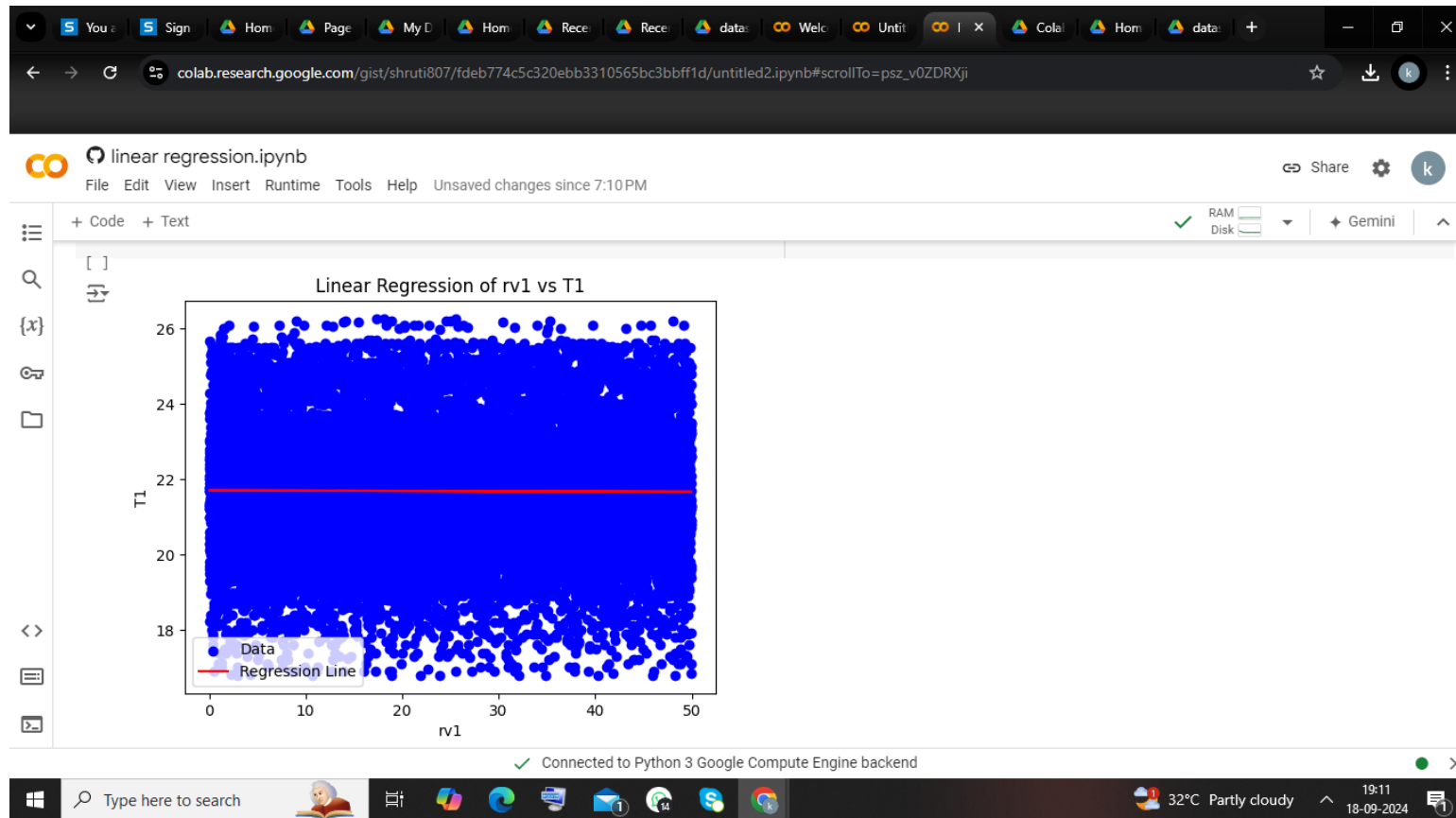
```
plt.ylabel('T1')
```

```
plt.title('Linear Regression of rv1 vs T1')
```

```
plt.legend()
```

```
plt.show()
```

RESULTING PLOT



- Insert the generated plot here, showing the data points and the regression line.
- Explain that the blue points represent actual data, and the red line represents the best-fit linear model.

CONCLUSION

- Linear regression helps us visualize the relationship between $rv1$ and $T1$.
 - Python's libraries such as Matplotlib and scikit-learn make it easy to implement and visualize linear regression models.
 - This approach is valuable in understanding trends and making predictions based on data.
-