

CS 6065 - Intro to Cloud Computing

Final Group Project

Energy Consumption Prediction and Optimization for Smart Homes

By

Samraysh Pellakur – M16474470

Sri Mani Sravika Nuthalapati - M16162004

Tulasi Rama Raju Chittiraju – M16407796

Link to Web App: <https://finalproject-dataentry.azurewebsites.net/>

1) Write-Up on ML Models

i. Linear Regression

Linear regression is a statistical method used for predicting a continuous dependent variable based on one or more independent variables. It assumes a linear relationship between the dependent and independent variables. The model estimates the coefficients of the linear equation by minimizing the difference between predicted and actual values. It's simple, interpretable, and often used for forecasting and trend analysis.

ii. Random Forest

Random Forest is an ensemble learning technique that builds multiple decision trees during training and combines their predictions to improve accuracy and prevent overfitting. It works by randomly selecting subsets of features and data points, creating diversity in the trees. Each tree independently makes predictions, and the final output is determined by aggregating results. It is effective for handling large datasets and capturing complex relationships.

iii. Gradient Boosting

Gradient Boosting is another ensemble method that builds decision trees sequentially, where each tree attempts to correct the errors of the previous one. It focuses on minimizing the residual errors by adding weak learners one at a time. It is a powerful technique for predictive modeling and is known for its high predictive accuracy, but it can be sensitive to noise and prone to overfitting without proper tuning.

The modelling question is “Predict the energy consumption (energy_consumption_kWh) of a home based on factors such as temperature setting (temperature_setting_C), occupancy status (occupancy_status), appliance used (appliance), usage duration (usage_duration_minutes), and day of the week (day_of_week).” and we will use random forest to prepare the answer for the above question.

2) Web Server Setup

The screenshot shows a web browser window with the address bar displaying 'finalproject-dataentry.azurewebsites.net'. The page contains a registration form titled 'Register' with the following fields:

- Username
- Password
- Email

A blue 'Submit' button is located at the bottom of the form.

The browser's taskbar at the bottom shows the time as 09:08 and the date as 09-12-2024. The system tray includes icons for weather (11°C Cloudy), search, and various application icons.

The screenshot shows the same web browser window with the registration form. The fields are now populated with the following data:

- Username: Simran Chaudry
- Password: *****
- Email: simranchaudry342@gmail.com

The blue 'Submit' button remains at the bottom of the form.

The browser's taskbar at the bottom shows the time as 09:09 and the date as 09-12-2024. The system tray includes icons for weather (11°C Cloudy), search, and various application icons.

3) Datastore and Data Loading

The screenshot shows the Microsoft Azure portal interface for the 'chittitu_ssms' SQL database. The left sidebar contains navigation options like Overview, Activity log, Tags, and Settings. The main content area displays the database's 'Essentials' tab, which includes details such as the Resource group, Server name, Status, Location, Subscription ID, and Tags. A 'Database data storage' section is also visible at the bottom.

chittitu_ssms (chittitu/chittitu_ssms)
SQL database

Search resources, services, and docs (G+)

Overview

- Activity log
- Tags
- Diagnose and solve problems
- Query editor (preview)
- Mirror database in Fabric (preview)
- Settings
 - Compute + storage
 - Connection strings
 - Properties
 - Locks
- Data management
 - Replicas
 - Sync to other databases
- Integrations
 - Azure Synapse Link

Essentials [JSON View](#)

Resource group ([move](#))
[Assignments](#) [Resource](#)

Server name
[chittitu.database.windows.net](#)

Status
Online

Connection strings
[Show database connection strings](#)

Location
East US 2

Pricing tier
[General Purpose - Serverless: Gen5, 1 vCore](#)

Subscription ([move](#))
[Azure for Students](#)

Auto-pause delay
[1 hour](#)

Subscription ID
93022546-6cb6-4588-98ad-6589fa5d7e15

Earliest restore point
2024-12-02 03:48 UTC

Tags ([edit](#))
[Add tags](#)

Getting started **Monitoring** Properties Features Notifications (0) Integrations Tutorials

Database data storage

The screenshot shows the SQL Server Enterprise Manager (SSMS) interface. The left pane displays the 'chittitu.database.windows.net' server and its 'dbo' database. The central pane shows a SQL query being executed, and the bottom pane displays the results of the query.

File Edit View Help

CONNECTIONS

- SERVERS
 - chittitu.database.windows.net, chittitu_ssms (chittitu)
 - Tables
 - dbo.EnergyUsage
 - Columns
 - timestamp (nvarchar(30), null)
 - home_id (int, null)
 - energy_consumption_kWh (float, null)
 - temperature_setting_C (float, null)
 - occupancy_status (nvarchar(50), null)
 - appliance (nvarchar(50), null)
 - usage_duration_minutes (int, null)
 - season (nvarchar(20), null)
 - day_of_week (nvarchar(20), null)
 - holiday (nvarchar(10), null)
 - Keys
 - Constraints
 - Triggers
 - Indexes
 - Statistics
 - dbo.Students
 - Dropped Ledger Tables
 - Views
 - Synonyms
 - Programmability
 - External Resources
 - Storage
 - Security
 - chittitu.database.windows.net, final-project (chittitu)
 - Tables

SQLQuery_1 - (79) c:\t\t\t\t\t

Run Cancel Disconnect Change Database: chittitu_ssms Estimated Plan Enable Actual Plan Parse Enable SQLCMD To Notebook

```
1 CREATE TABLE EnergyUsage1 (  
2     timestamp NVARCHAR(30), -- More precise and flexible for date and time values  
3     home_id INT, -- For unique home identifiers  
4     energy_consumption_kWh FLOAT, -- For energy consumption values with decimals  
5     temperature_setting_C FLOAT, -- For temperature values with decimals  
6     occupancy_status NVARCHAR(50), -- For text values indicating occupancy status  
7     appliance NVARCHAR(50), -- For appliance names  
8     usage_duration_minutes INT, -- For duration in minutes  
9     season NVARCHAR(20), -- For season names  
10    day_of_week NVARCHAR(20), -- For day of the week  
11    holiday NVARCHAR(10) -- For binary values (0 or 1)  
12 );  
13  
14  
15 select * from EnergyUsage  
16 where [timestamp] < '28-01-2007 13:00'  
17  
18 drop table EnergyUsage1
```

Results Messages

timestamp	home_id	energy_consumption_kWh	temperature_setting_C	occupancy_status	appliance	usage_duration_minutes
28-01-2002 00:00	100	100	30	Occupied	HVAC	120

Ln 14, Col 1 Spaces: 4 UTF-8 CRLF 999,185 rows MSSQL 00:00:05 chittitu.database.windows.net: chittitu_ssms (57)

51°F Cloudy

ENG IN 22:05 08-12-2024

4) Interactive Web Page

Username: Simran Chaudry
Email: simranchaudry342@gmail.com
[Add Data](#) [Show Statistics](#)

Find Energy Consumption of Household

Please enter your Home ID: [Search](#)

Timestamp	Home ID	Appliance	Energy Consumption (kWh)	Usage Duration (minutes)
20-06-2024 02:00	2	Electronics	5	100
05-08-2035 03:00	2	Dishwasher	5	8
03-06-2055 19:00	2	HVAC	5	40
29-04-2058 12:00	2	Electronics	5	24
02-06-2089 11:00	2	Dishwasher	5	93
07-01-2101 02:00	2	Refrigerator	5	46
04-12-2120 04:00	2	Lighting	5	25
24-08-2120 12:00	2	HVAC	4.99	63
08-10-2112 06:00	2	Lighting	4.99	68
26-11-2101 09:00	2	HVAC	4.99	2
23-06-2099 14:00	2	Lighting	4.99	67
23-04-2095 14:00	2	Refrigerator	4.99	42
16-11-2097 18:00	2	Electronics	4.99	52
20-10-2082 20:00	2	Machines, Machines	4.00	118

5) Data Loading Web App

Energy Usage Data Entry

Timestamp:

Home ID: Energy Consumption (kWh):

Temperature Setting (°C): Occupancy Status:

Appliance: Usage Duration (minutes):

Season: Day of Week:

Holiday (0 or 1):

[Submit](#)

finalproject-dataload.azurewebsites.net says
Data inserted successfully!

Timestamp:
2024-09-07 05:08:23

Home ID:
1

Energy Consumption (kWh):
2

Temperature Setting (°C):
1.3

Occupancy Status:
Occupied

Appliance:
Lighting

Usage Duration (minutes):
60

Season:
Summer

Day of Week:
Thursday

Holiday (0 or 1):
1

Submit

6) Web Page with Dashboard

I. Energy Consumption by Appliance Trend

- HVAC contributes the most to energy consumption at 24.93%, followed by Lighting at 16.01%, and Washing Machine at 14.81%. The least energy is consumed by Electronics at 14.7%.
- Potential strategies to optimize such high consumption from some of the appliances include better insulation or smart lighting systems.

II. Energy Consumption by Day of the Week

- Energy consumption steadily decreases from Friday (highest) to Tuesday (lowest). Friday has the highest energy usage at 364K, while Tuesday has the lowest at around 358K.
- Possible explanations include reduced occupancy or altered activity patterns during weekdays versus weekends.

III. Occupancy Trend

- Occupied energy consumption (51.97%) is higher than unoccupied (48.03%). This indicates significant energy is consumed when the house is both occupied and unoccupied. Potential reasons for such high consumption even when it's unoccupied could likely be due to devices or appliances left running.
- Solutions to optimize the high consumption may include automated systems, such as smart plugs or energy-saving modes.

IV. Seasonal Trends

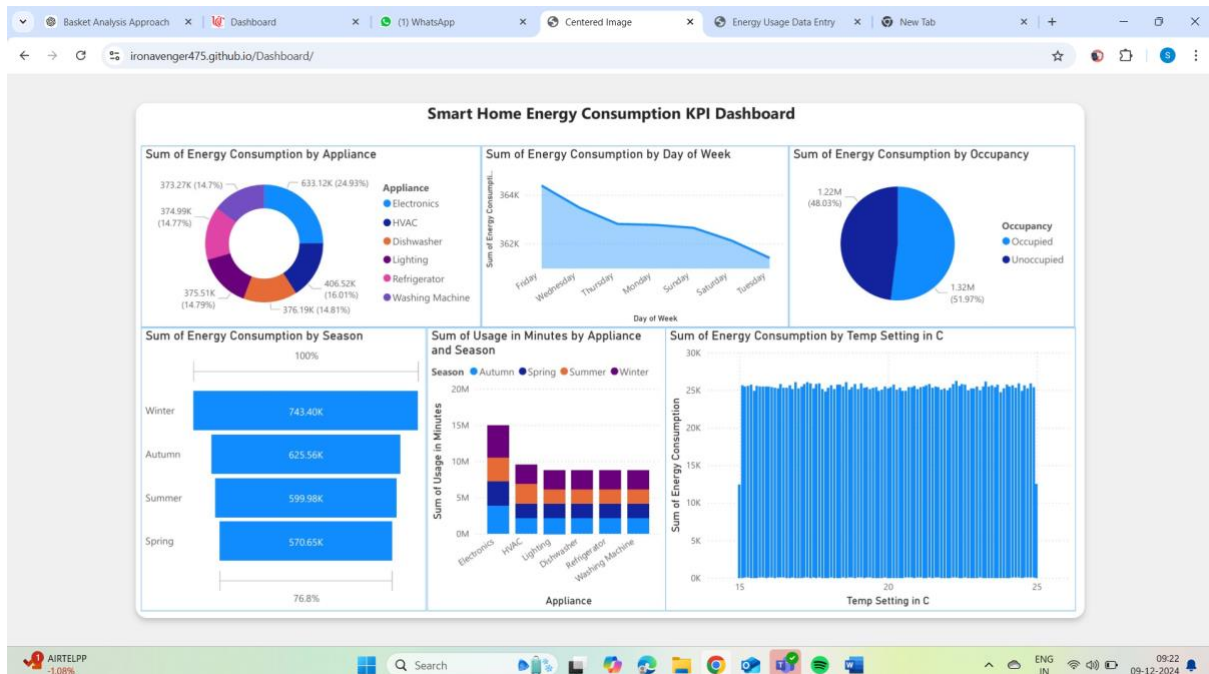
- Winter has the highest energy consumption (743.40K), while spring has the lowest (570.65K). Autumn and summer have similar values.
- What is the Season with the highest energy consumption?
Winter.
- What contributes to the spike in winter energy consumption?
It is clear that heating systems (HVAC) were being operated for longer hours during winter.

V. Energy Consumption by Temperature Setting

- Energy consumption remains relatively constant across temperature settings (15°C to 24°C).
- It is clear that the temperature setting has minimal to no impact in energy consumption.

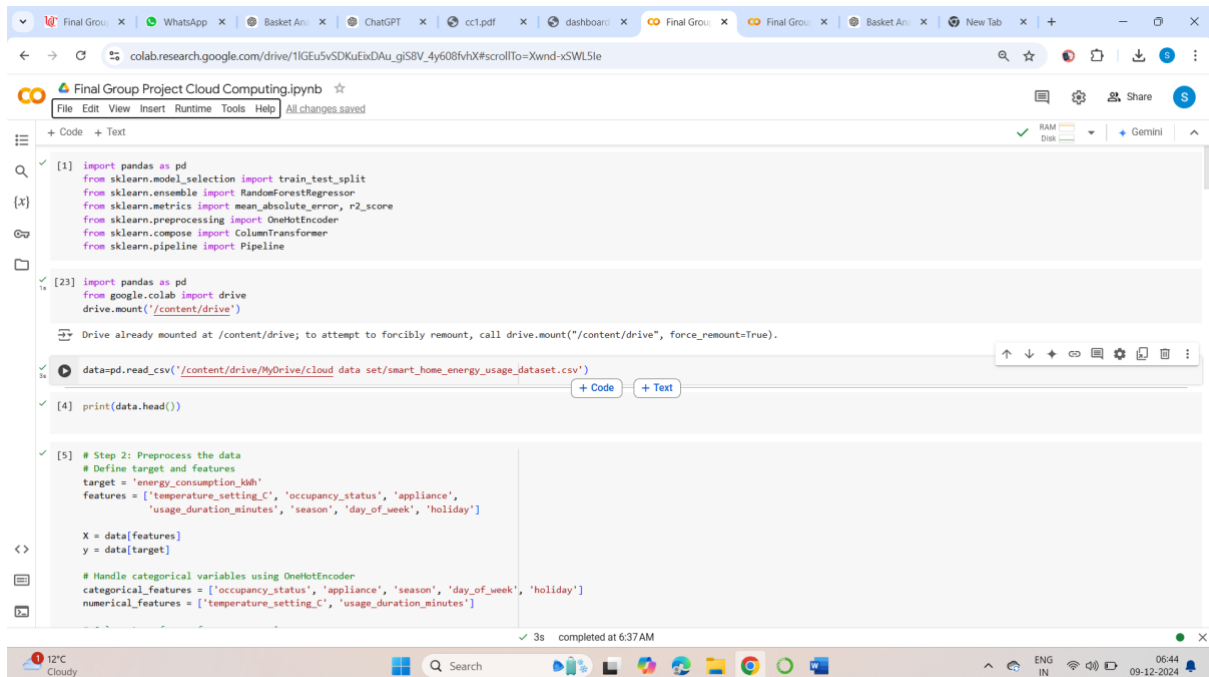
VI. Appliance Usage by Appliance and Season

- Electronics dominate usage time across all seasons.
- HVAC has higher usage in winter compared to other seasons. Other appliances such as Lighting and Washing Machines have consistent usage across seasons.



7) ML Model Application

We used random forest for the basket analysis. Initially, we explored the dataset with a Random Forest Regressor to understand the relationship between the features (e.g., energy_consumption_kWh, temperature_setting_C, occupancy_status, appliance, etc.) and the target variable. The regressor helped identify key drivers of energy consumption, offering preliminary insights into feature importance and their influence on churn behavior.



The screenshot shows a Jupyter Notebook titled "Final Group Project Cloud Computing.ipynb" in a Google Colab environment. The notebook is open to a code cell containing the following Python code:

```
[1] import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, r2_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

[23] import pandas as pd
from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

data=pd.read_csv('/content/drive/MyDrive/cloud data set/smart_home_energy_usage_dataset.csv')

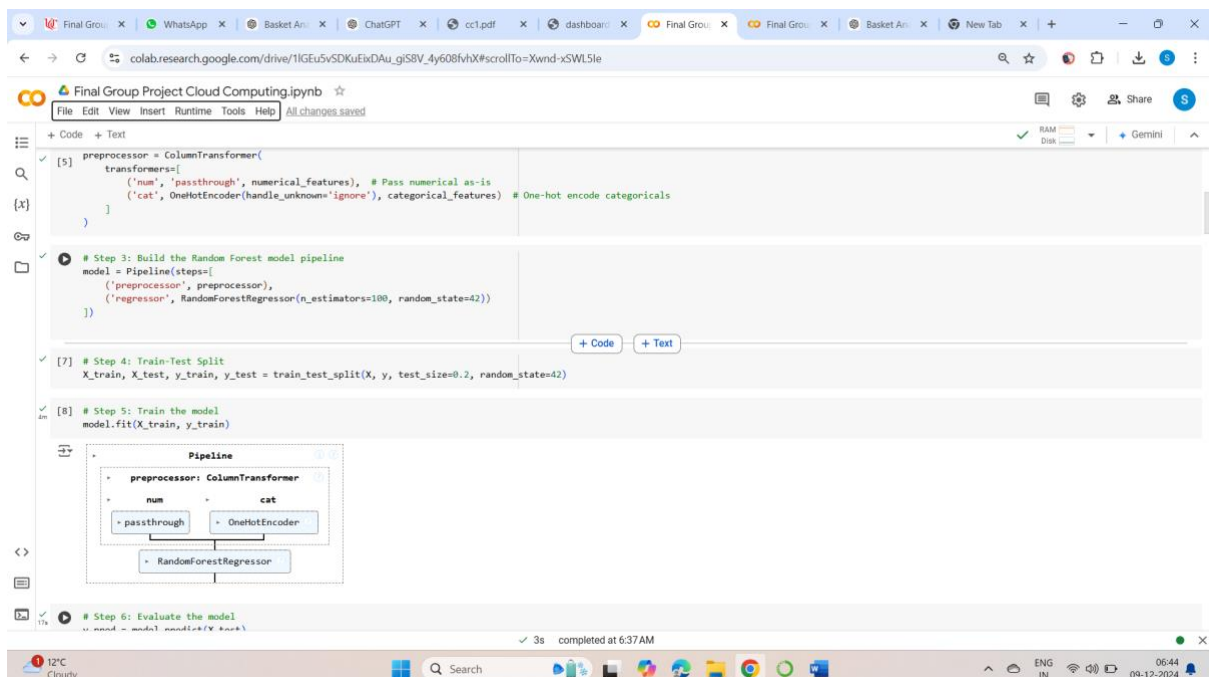
[4] print(data.head())

[5] # Step 2: Preprocess the data
# Define target and features
target = 'energy_consumption_kwh'
features = ['temperature_setting_C', 'occupancy_status', 'appliance',
           'usage_duration_minutes', 'season', 'day_of_week', 'holiday']

X = data[features]
y = data[target]

# Handle categorical variables using OneHotEncoder
categorical_features = ['occupancy_status', 'appliance', 'season', 'day_of_week', 'holiday']
numerical_features = ['temperature_setting_C', 'usage_duration_minutes']
```

The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for file operations, and a status bar at the bottom showing the temperature (12°C) and time (06:44, 09-12-2024).



The screenshot shows the continuation of the Jupyter Notebook, focusing on building and evaluating the Random Forest model. The code cells contain the following Python code:

```
[5] preprocessor = ColumnTransformer(
    transformers=[
        ('num', 'passthrough', numerical_features), # Pass numerical as-is
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_features) # One-hot encode categorical
    ]
)

# Step 3: Build the Random Forest model pipeline
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(n_estimators=100, random_state=42))
])

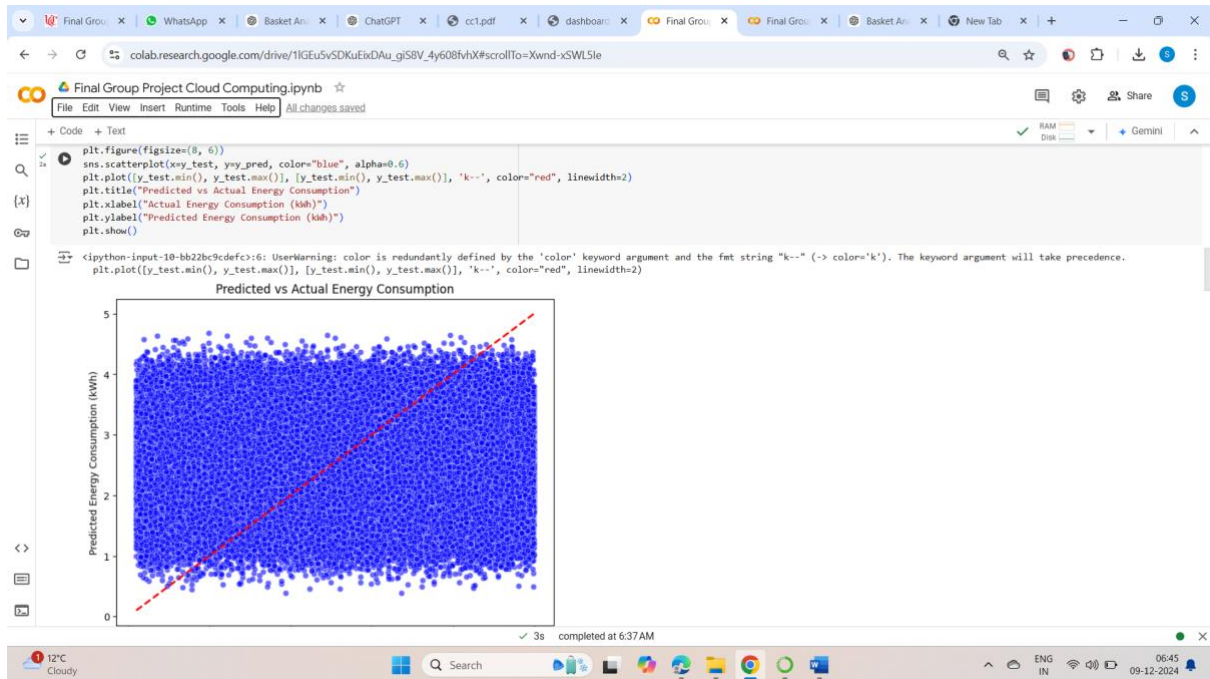
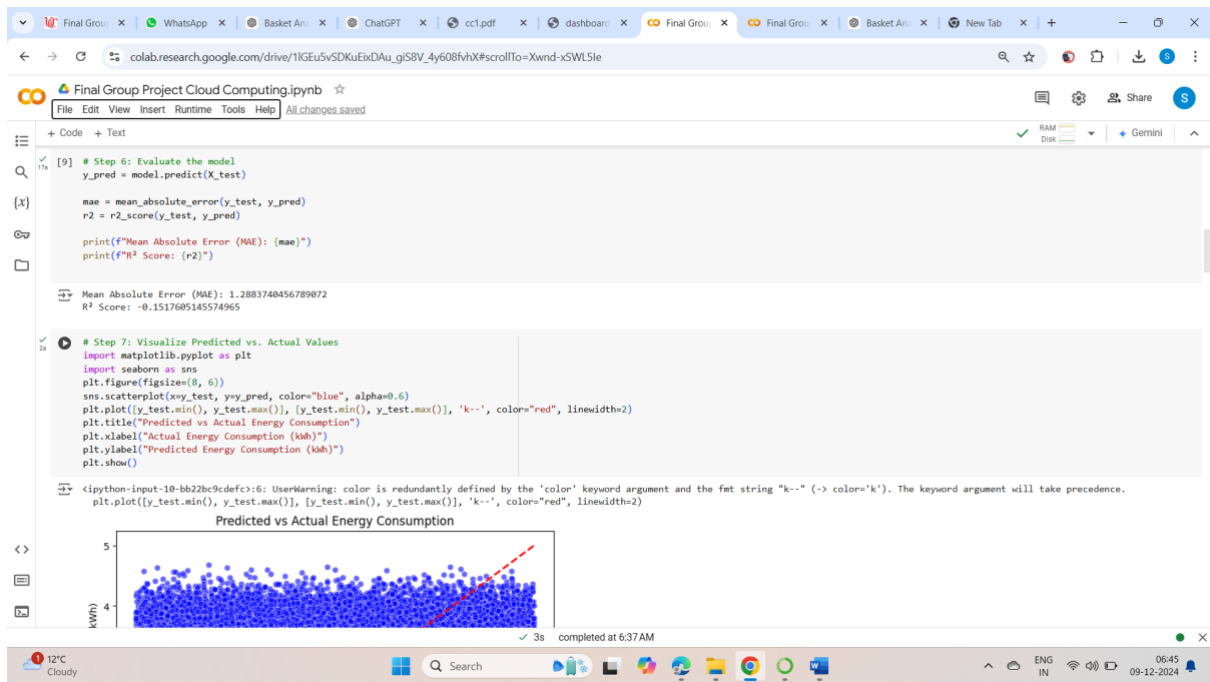
[7] # Step 4: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

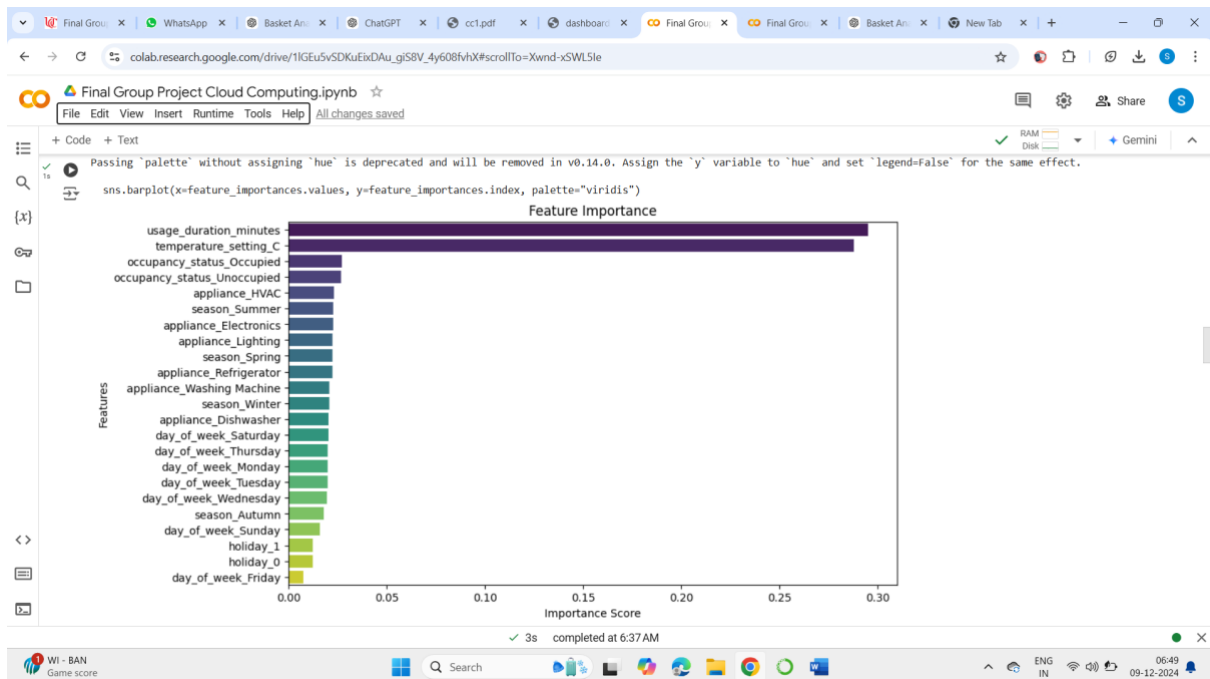
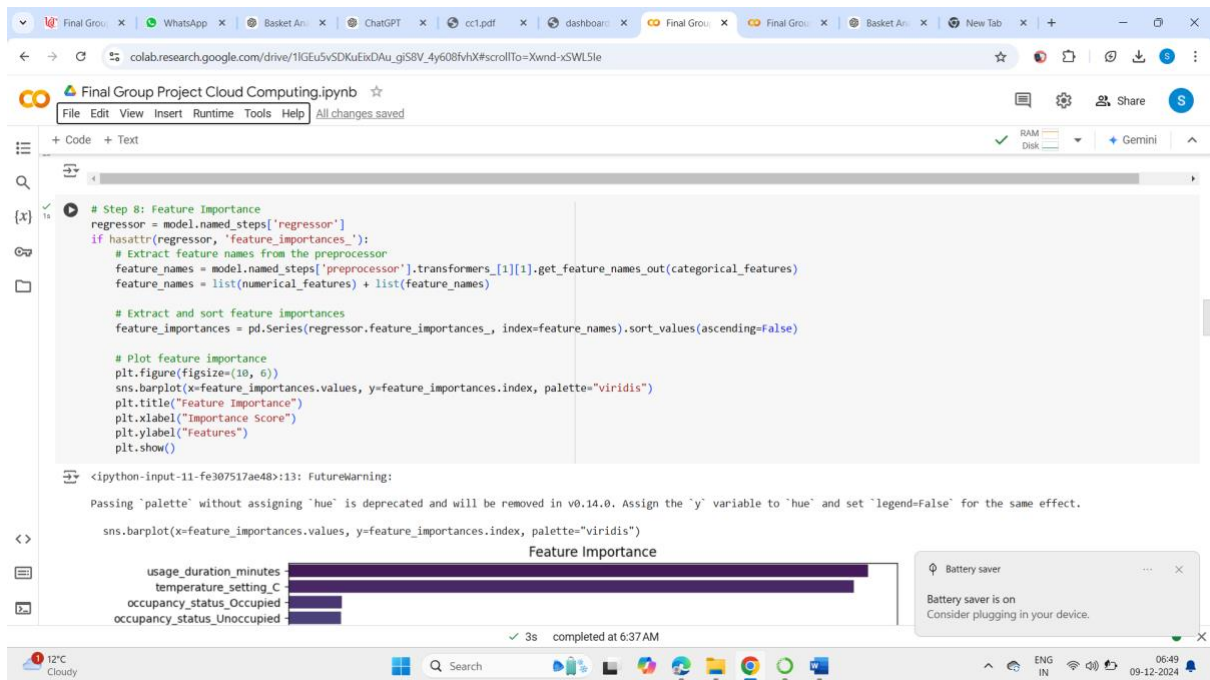
[8] # Step 5: Train the model
model.fit(X_train, y_train)
```

A visual representation of the pipeline is shown, illustrating the flow from the input data through the preprocessor (which handles numerical and categorical features) to the RandomForestRegressor model.

```
graph TD
    Input --> Preprocessor
    subgraph Pipeline
        direction TB
        Preprocessor --> RandomForestRegressor
    end
    Preprocessor --> Output
```

The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for file operations, and a status bar at the bottom showing the temperature (12°C) and time (06:44, 09-12-2024).





```
# Step 77: Feature Importance (Optional)
regressor = model.named_steps["regressor"]
if hasattr(regressor, "feature_importances_"):
    feature_names = model.named_steps["preprocessor"].transformers_[1][1].get_feature_names_out(categorical_features)
    feature_names = list(numerical_features) + list(feature_names)
    feature_importances = pd.Series(regressor.feature_importances_, index=feature_names).sort_values(ascending=False)
    print("Feature Importances:\n", feature_importances)
```

Feature Importances:	
usage_duration_minutes	0.295056
temperature_setting_C	0.287726
occupancy_status_occupied	0.026995
occupancy_status_unoccupied	0.026953
appliance_hvac	0.023230
season_summer	0.022794
appliance_electronics	0.022776
appliance_lighting	0.022475
season_spring	0.022304
appliance_refrigerator	0.022190
appliance_washing_machine	0.020740
season_winter	0.020730
appliance_dishwasher	0.020572
day_of_week_saturday	0.020256
day_of_week_thursday	0.019955
day_of_week_monday	0.019846
day_of_week_tuesday	0.019835
day_of_week_wednesday	0.019685
season_autumn	0.017965
day_of_week_sunday	0.015848
holiday_1	0.012315
holiday_0	0.012301
day_of_week_friday	0.007453

dtype: float64

3s completed at 6:37 AM

8) Churn Prediction

In this project, we developed a churn prediction model using energy usage data from households. Churn was defined as a significant drop in energy consumption (energy_consumption_kWh) below 10% of the average usage. We selected relevant features such as temperature_setting_C, occupancy_status, appliance, and contextual factors like season, day_of_week, and holiday to predict churn behavior. Categorical features were one-hot encoded, while numerical features were passed directly into a preprocessing pipeline. A Random Forest Classifier was chosen due to its ability to handle non-linear relationships and provide insights into feature importance.

The dataset was split into training and testing subsets, and the model was trained and evaluated using metrics like precision, recall, F1-score, and a confusion matrix. Additionally, we visualized churn distribution and feature importance to better understand the key drivers of churn. This approach helps identify households likely to disengage, enabling businesses to design proactive strategies for customer retention based on energy consumption patterns and behavioral insights.

Final Group Project Cloud Computing.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[13] import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

[14] # Step 2: Define the target (churn) variable
# Example: Churn if energy consumption is less than 10% of mean
threshold = data['energy_consumption_kwh'].mean() * 0.1
data['churn'] = (data['energy_consumption_kwh'] < threshold).astype(int)

[15] # Step 3: Define features and target
features = ['temperature_setting_C', 'occupancy_status', 'appliance',
           'usage_duration_minutes', 'season', 'day_of_week', 'holiday']
target = 'churn'

X = data[features]
y = data[target]

[16] # Step 4: Preprocess data (handle categorical variables)
categorical_features = ['occupancy_status', 'appliance', 'season', 'day_of_week', 'holiday']
numerical_features = ['temperature_setting_C', 'usage_duration_minutes']

preprocessor = ColumnTransformer(
    transformers=[
        ('num', 'passthrough', numerical_features),
        ('cat', OneHotEncoder, categorical_features)
    ])
3s completed at 6:37 AM
```

Watchlist Ideas

Search

ENG IN 06:50 09-12-2024

Final Group Project Cloud Computing.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[17] # Step 5: Create a Random Forest pipeline
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier(n_estimators=100, random_state=42))
])

[18] # Step 6: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

[19] # Step 7: Train the model
model.fit(X_train, y_train)

[20] # Step 8: Evaluate the model
y_pred = model.predict(X_test)
3s completed at 6:37 AM
```

Watchlist Ideas

Search

ENG IN 06:51 09-12-2024

Pipeline

```

graph TD
    subgraph preprocessor [preprocessor: ColumnTransformer]
        direction LR
        num[num]
        cat[cat]
        num --> passthrough[passthrough]
        cat --> onehot[OneHotEncoder]
    end
    passthrough --> rf[RandomForestClassifier]
    onehot --> rf
  
```

Final Group Project Cloud Computing.ipynb

```
[20] # Step 8: Evaluate the model
y_pred = model.predict(X_test)

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

Confusion Matrix:

	192374	1233		
	6350	43		


Classification Report:

	precision	recall	f1-score	support
0	0.97	0.99	0.98	193607
1	0.03	0.01	0.01	6393
accuracy			0.96	200000
macro avg	0.50	0.50	0.50	200000
weighted avg	0.94	0.96	0.95	200000

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.countplot(data=data, x='churn')
plt.title("Churn Distribution")
plt.show()
```

Churn Distribution



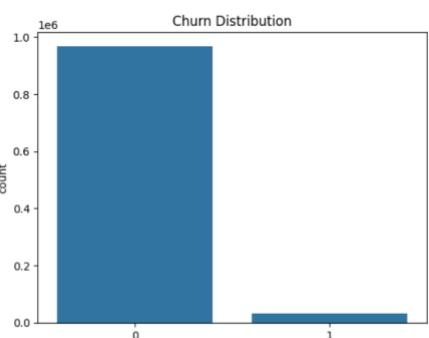
3s completed at 6:37 AM

Final Group Project Cloud Computing.ipynb

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.countplot(data=data, x='churn')
plt.title("Churn Distribution")
plt.show()
```

Churn Distribution



3s completed at 6:37 AM

