EXPERIMENT 1

Shravanya Andhale
D15A-30

AIM: Implement Linear and Logistic Regression on real-world datasets

THEORY:

1. Dataset Source

Dataset provided by Council on Energy, Environment and Water under the India Residential Energy Survey.
Source link:
https://www.kaggle.com/code/raitest/india-residential-energy-survey-ires-2020-eda/input?select=CEEW+-+IRES+Data.csv

2. Dataset Description

The India Residential Energy Survey is a nationally representative dataset that captures information on household energy access, consumption, expenditure, infrastructure, and technology adoption behaviour.

For this experiment, two targets are derived from the same survey to demonstrate both regression paradigms.

Predictor variables

- State abbreviation
- Gender of respondent
- Age of respondent
- Education of primary income earner
- Number of household members
- Type of dwelling
- Ownership status

These variables represent demographic capacity, economic stability, and housing conditions that are expected to influence energy behaviour.

Target variables

For Logistic Regression:
q320_c_solar_mini_grid_yn
Binary indicator representing whether the household is connected to or uses a solar mini grid.

For Linear Regression:
avg_monthly_bill
Continuous variable representing average electricity expenditure.

Size
After cleaning and removal of missing values, several thousand household records remain for analysis.

Data characteristics
The dataset includes structured survey responses with both categorical and numerical variables.

3. Mathematical Formulation of the Algorithm

Linear Regression

Linear regression models the dependent variable as a linear combination of predictors.

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \epsilon$$

Parameters are estimated by minimizing squared errors between observed and predicted values.

Logistic Regression

Logistic regression models the probability of belonging to class one.

$$P(y = 1|x) = \frac{1}{1 + e^{-(\beta_0 + \sum_{i=1}^{n} \beta_i x_i)}}$$

The model estimates coefficients through maximum likelihood estimation.

4. Algorithm Limitations

Both approaches assume linear relationships between predictors and outcome. They may fail to capture nonlinear interactions unless additional transformations are introduced. Logistic regression can be affected by severe class imbalance. Linear regression is sensitive to outliers and multicollinearity. Performance also depends on quality and completeness of survey data.

5. Methodology / Workflow

1. Obtain survey data
2. Select relevant demographic and housing predictors
3. Remove records with missing values
4. Encode categorical variables using one hot encoding
5. Standardize numerical features
6. Split dataset into training and testing subsets
7. Train logistic and linear models
8. Perform cross validated hyperparameter tuning
9. Evaluate predictions
10. Interpret results for behavioural insights

6. Performance Analysis

Logistic Regression Results

Accuracy: **0.9946**
ROC AUC: **0.8731**

Classification performance:

Class 0
Precision: 0.99
Recall: 1.00

F1 score: 1.00
Support: 2955

Class 1
Precision: 0.00
Recall: 0.00
F1 score: 0.00
Support: 16

Macro average F1: 0.50
Weighted average F1: 0.99

The extremely high accuracy is driven by the dominance of non adoption cases. Since adopters form a very small share, the model predicts the majority class more often. However, the ROC AUC indicates that probability ranking ability remains strong. This means the model can still be useful for prioritizing households when designing targeted interventions.

Linear Regression Results

Root Mean Squared Error: **601.88**
R squared: **0.292**

The R squared value shows that nearly 29 percent of variation in electricity expenditure is explained by demographic and housing characteristics. Considering behavioural uncertainty and absence of detailed appliance usage data, this is a reasonable predictive performance.

7. Hyperparameter Tuning

For logistic regression, the regularization parameter C was optimized using grid search and cross validation to balance bias and variance. The tuned model improved generalization and ROC performance.

For linear regression, Ridge regularization parameter alpha was tuned. Regularization helped stabilize coefficient estimates and prevented overfitting while maintaining explanatory power.

OUTPUT:

Logistic Regression:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

from sklearn.linear_model import LogisticRegression, Ridge
from sklearn.metrics import accuracy_score, classification_report, roc_auc_score
from sklearn.metrics import mean_squared_error, r2_score, RocCurveDisplay
```

```
from google.colab import files
uploaded = files.upload()

file_name = list(uploaded.keys())[0]
df = pd.read_csv("/content/CEEW - IRES Data.csv")

df.shape
```

```
Choose files CEEW - IRES Data.csv
CEEW - IRES Data.csv(text/csv) - 13336177 bytes, last modified: 15/02/2026 - 100% done
Saving CEEW - IRES Data.csv to CEEW - IRES Data (1).csv
/tmp/ipython-input-2454589002.py:5: DtypeWarning: Columns (6,15,22,46,84,98,163,172,
  df = pd.read_csv("/content/CEEW - IRES Data.csv")
(14851, 517)
```

```
predictor_cols = [
    "state_abbv",
    "q201_resp_gender",
    "q202_resp_age",
    "q208_priminc_earner_edu",
    "q213_no_members",
    "q216_house_pucca_kachha",
    "q220_own_house_yn",
]

predictor_cols = [c for c in predictor_cols if c in df.columns]
predictor_cols
```

```
['state_abbv',
 'q201_resp_gender',
 'q202_resp_age',
 'q208_priminc_earner_edu',
 'q213_no_members',
 'q216_house_pucca_kachha',
 'q220_own_house_yn']
```

```
log_target = "q320_c_solar_mini_grid_yn"

log_df = df[predictor_cols + [log_target]].dropna().copy()
log_df[log_target] = log_df[log_target].apply(lambda x: 1 if x == 1 else 0)

X = log_df[predictor_cols]
y = log_df[log_target]

num_cols = X.select_dtypes(include=np.number).columns
cat_cols = X.select_dtypes(exclude=np.number).columns
```

```
    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.2, random_state=42, stratify=y
    )
```

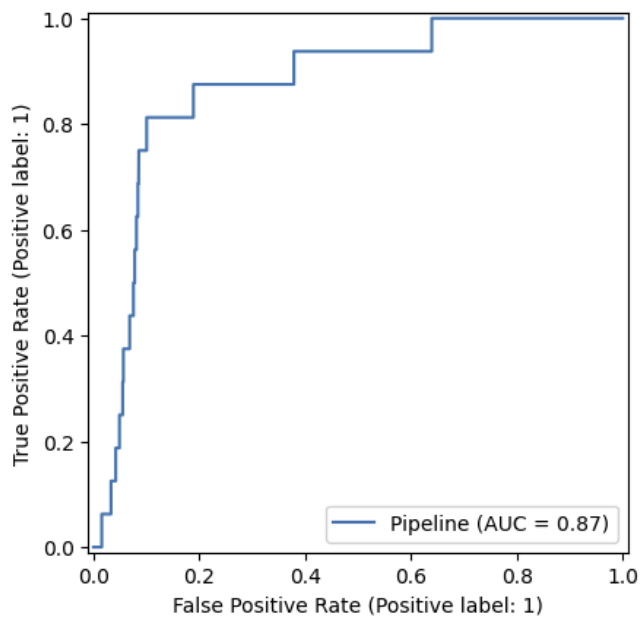
```
preprocess = ColumnTransformer([
    ("num", StandardScaler(), num_cols),
    ("cat", OneHotEncoder(handle_unknown="ignore"), cat_cols)
])

pipe = Pipeline([
    ("prep", preprocess),
    ("model", LogisticRegression(max_iter=2000))
])

param_grid = {"model__C": [0.01, 0.1, 1, 10]}

grid_log = GridSearchCV(pipe, param_grid, cv=5, scoring="roc_auc")
grid_log.fit(X_train, y_train)

best_log = grid_log.best_estimator_
grid_log.best_params_
```

```
{'model__C': 10}
```

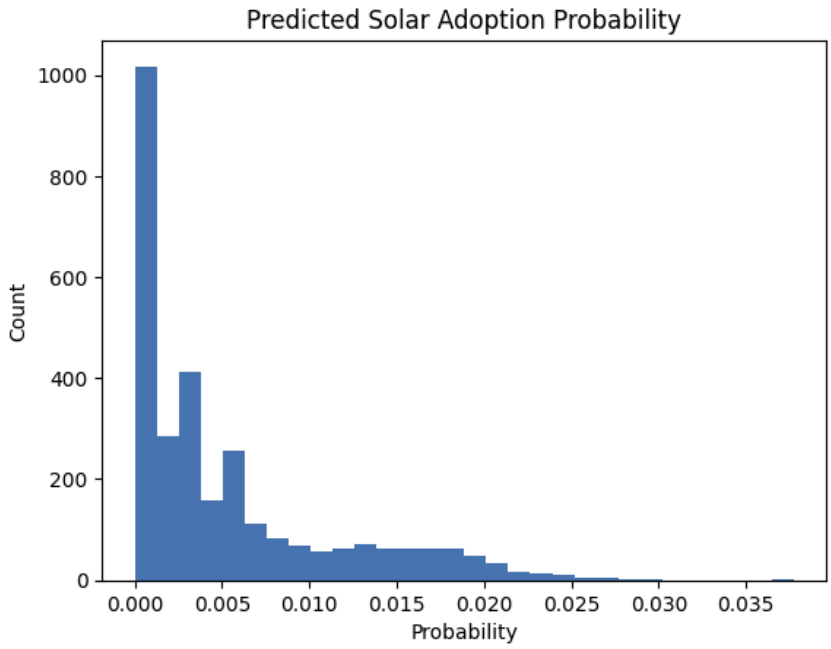
```
pred = best_log.predict(X_test)
prob = best_log.predict_proba(X_test)[:,1]

print("Accuracy:", accuracy_score(y_test, pred))
print("ROC AUC:", roc_auc_score(y_test, prob))
print(classification_report(y_test, pred))
```

```
Accuracy: 0.994614607876136
ROC AUC: 0.8730647208121828
              precision    recall  f1-score   support

           0       0.99      1.00      1.00      2955
           1       0.00      0.00      0.00        16

    accuracy                           0.99      2971
   macro avg       0.50      0.50      0.50      2971
weighted avg       0.99      0.99      0.99      2971

/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
RocCurveDisplay.from_estimator(best_log, X_test, y_test)
plt.show()
```




```
plt.hist(prob, bins=30)
plt.title("Predicted Solar Adoption Probability")
plt.xlabel("Probability")
plt.ylabel("Count")
plt.show()
```



Linear regression:

```
lin_target = "avg_monthly_bill"

lin_df = df[predictor_cols + [lin_target]].dropna().copy()

X = lin_df[predictor_cols]
y = lin_df[lin_target]

num_cols = X.select_dtypes(include=np.number).columns
cat_cols = X.select_dtypes(exclude=np.number).columns
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

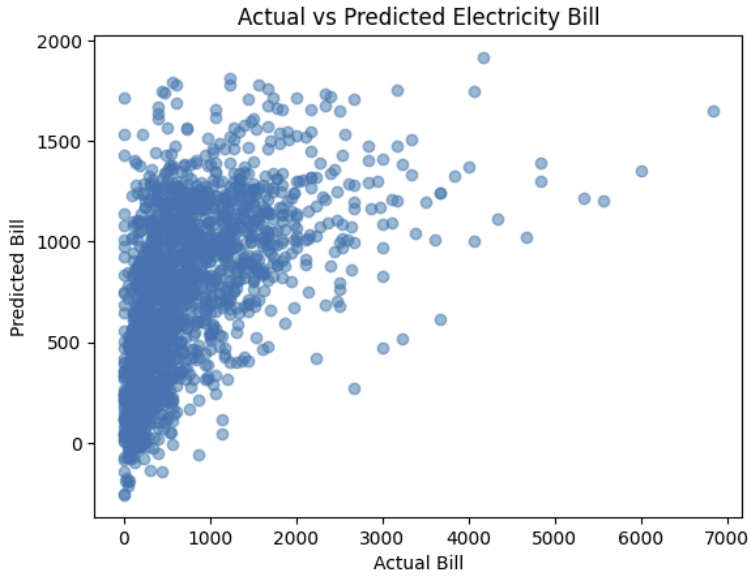
```
preprocess = ColumnTransformer([
    ("num", StandardScaler(), num_cols),
    ("cat", OneHotEncoder(handle_unknown="ignore"), cat_cols)
])

pipe = Pipeline([
    ("prep", preprocess),
    ("model", Ridge())
])

param_grid = {"model__alpha": [0.1, 1, 10, 100]}

grid_lin = GridSearchCV(pipe, param_grid, cv=5, scoring="r2")
grid_lin.fit(X_train, y_train)

best_lin = grid_lin.best_estimator_
grid_lin.best_params_
```

```
{'model__alpha': 1}
```

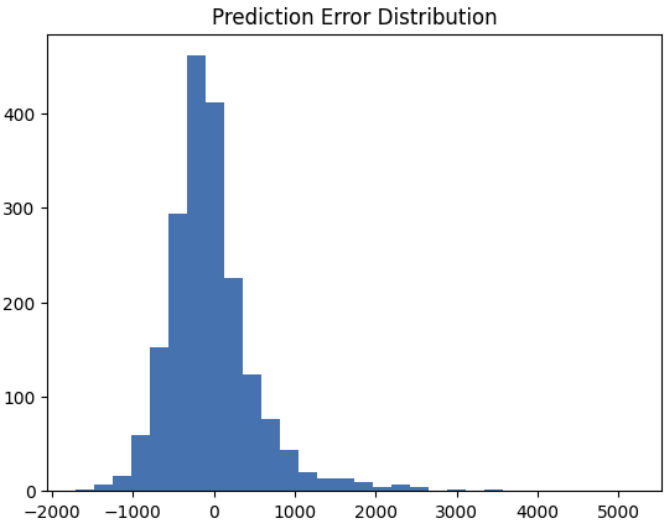
```
pred = best_lin.predict(X_test)

print("RMSE:", np.sqrt(mean_squared_error(y_test, pred)))
print("R2:", r2_score(y_test, pred))
```

```
RMSE: 601.8757014494711
R2: 0.2920731049344363
```

```
plt.scatter(y_test, pred, alpha=0.5)
plt.xlabel("Actual Bill")
plt.ylabel("Predicted Bill")
plt.title("Actual vs Predicted Electricity Bill")
plt.show()
```




```
errors = y_test - pred
plt.hist(errors, bins=30)
plt.title("Prediction Error Distribution")
plt.show()
```



CONCLUSION: Linear and Logistic Regression were successfully implemented on a large scale residential energy dataset. The study demonstrated that household characteristics meaningfully influence both renewable adoption and consumption outcomes. Despite class imbalance challenges, probability based insights from logistic regression and expenditure modeling from linear regression provide a quantitative basis for designing targeted digital communication, subsidy awareness, and behavioural change strategies.