Load Type Prediction from Power System Data

Objective

The primary objective of this project is to develop a machine learning model capable of predicting the **Load_Type** of a power system based on historical features. The target variable **Load_Type** includes three categories:

- `Light_Load`
- `Medium_Load`
- `Maximum_Load`

This classification problem emphasizes data preprocessing, exploratory data analysis (EDA), feature engineering, model selection, evaluation, and explainability.

Dataset Description

The dataset includes the following features:

Feature	Description
`Date_Time`	Timestamp (first of every month)
`Usage_kWh`	Industry Energy Consumption in kWh
`Lagging_Current_Reactive.Power_kVarh`	Lagging current reactive power
`Leading_Current_Reactive_Power_kVarh`	Leading current reactive power
`CO2(tCO2)`	Carbon dioxide emission (ppm)
`NSM`	Number of seconds from midnight
`Load_Type`	Target: Light_Load, Medium_Load, Maximum_Load

Google Drive Setup

```
/My Drive/
└── load_type_prediction/
    ├── data/
    │   ├── raw/              # Contains raw power_data.csv
    │   ├── train/            # Train split after cleaning
    │   └── test/             # Last month data as test
    ├── processed_data/       # Intermediate transformed data
    └── fine_tuned_model/     # Saved models (.pkl)
```

Step-by-Step Pipeline

1. Exploratory Data Analysis (EDA) on Full Raw Dataset

- Visualized distributions of all continuous features.
- Boxplots for understanding `Usage_kWh` across `Load_Type`.
- Scatterplots: `Usage_kWh` vs `NSM` colored by `Load_Type`.
- Monthly aggregation to inspect seasonality trends.
- Checked data quality (missing values, invalid entries).
- Result: Insights about outliers, skewness, time influence, and class imbalance.

2. Data Cleaning

- Removed NaNs and invalid entries.
- Converted date formats.

- Saved cleaned dataset as `power_cleaned.csv`.

3. Train-Test Split

- Used **last month** data as test set for real-world evaluation.
- Remaining data used as training dataset.
- Saved as `power_train.csv` and `power_test.csv`.

4. EDA on Cleaned Train Set

- Repeated visual analysis.
- Boxplots revealed strong influence of `Usage_kWh`, `Lagging_Current_Reactive.Power_kVarh`, and `CO2(tCO2)`.
- Heatmap showed some strong correlations.
- Temporal features (`Month`, `Hour`, `Day`) derived.

5. Feature Engineering

- Dropped:
  - `Leading_Current_Power_Factor` (constant)
  - `NSM` (less informative after visual inspection)
- Created:
  - `Month`, `Day`, `Hour` from `Date_Time`
- Log-transformed skewed features:
  - `Usage_kWh`
  - `Lagging_Current_Reactive.Power_kVarh`
  - `Leading_Current_Reactive_Power_kVarh`
  - `CO2(tCO2)`
- Label encoded `Load_Type`:
  - Light_Load: 0, Medium_Load: 1, Maximum_Load: 2
- Saved processed files: `train_transformed.csv`, `test_transformed.csv`

Model Training & Evaluation

Initial Model Candidates

Trained and evaluated the following models on the training dataset (80-20 split):

- Random Forest
- XGBoost
- MLPClassifier (with StandardScaler)
- Logistic Regression (with StandardScaler)

Evaluation Metrics

Used `accuracy`, `precision`, `recall`, `f1-score`, and `confusion_matrix`.

Validation Results (20% of training data)

Model	Accuracy
XGBoost	**0.94**
Random Forest	0.93
MLP Classifier	0.93
Logistic Regression	0.75

Test Set Evaluation (Last month data)

Model	Accuracy
XGBoost	**0.90**
Random Forest	0.87
MLP Classifier	0.79

Model	Accuracy
Logistic Regression	0.61

Decision: Proceed with XGBoost

Model Fine-Tuning (XGBoost)

Used `RandomizedSearchCV` to tune the following parameters:

- `max_depth`
- `learning_rate`
- `n_estimators`
- `subsample`
- `colsample_bytree`
- `gamma`
- `reg_alpha`
- `reg_lambda`

Best Hyperparameters

```
{
  "max_depth": 8,
  "learning_rate": 0.0708,
  "n_estimators": 188,
  "subsample": 0.6558,
  "colsample_bytree": 0.6727,
  "gamma": 0.9170,
  "reg_alpha": 0.2912,
  "reg_lambda": 1.1119
}
```

Post-Tuning Validation Accuracy: 0.94

Post-Tuning Test Accuracy: 0.90 (Improved precision and F1-score)

Model saved as: `xgboost_tuned_model.pkl`

Summary of Key Decisions

Step	Decision & Justification
EDA on Full Data	To explore structure, trends, and quality issues
Test Set Strategy	Last month's data to simulate unseen future data
Dropping Features	Removed constant/redundant fields (`NSM`, `Leading_PF`)
Log Transformations	For normalizing skewed distributions
Model Choice	XGBoost performed best on both validation & test
Hyperparameter Tuning	Boosted model precision and generalization

---

Scripts Provided

- `01_eda_power_data.py` : EDA on complete raw dataset
- `02_data_cleaning.py` : Clean & export final dataset
- `03_train_test_split.py` : Last month test split
- `04_train_eda.py` : Visuals and correlations on train set

- `05_feature_engineering.py` : For both train and test
- `model_xgb.py` : Train and save baseline XGBoost
- `model_xgb_tuned.py` : Fine-tune and save best model
- `test_xgb_tuned.py` : Load tuned model and evaluate on test set

Final Notes

- The full workflow is designed to be modular and interpretable.
- Future improvements can involve:
  - Feature selection based on importance or SHAP
- All scripts are compatible with Google Colab + Drive setup.

Evaluation-ready submission includes:

- Scripts (Colab / `.py`)
- Transformed CSVs
- Saved `.pkl` model files
- README with end-to-end documentation

Maintainer: Pratik Kumar
Institute: NIT Silchar
Task Deadline: 23rd June 2025