HR Analytics - Predict Employee Attrition

Introduction

Employee attrition is a major concern for modern organisations, often resulting in increased hiring costs and loss of experienced talent. This analysis aims to leverage historical HR data to understand the causes behind attrition and develop predictive tools using machine learning and statistical analysis to help HR take preventive actions.

Dataset Overview

The primary dataset `HR_Analytics.csv` contains employee data including attributes like department, salary,promotions, and whether an employee has left. Derived datasets include attrition summaries, salary distributions, and encoded versions used for model training and evaluation.

Tools and Libraries Used

pandas: Data manipulation - seaborn & matplotlib: Visualizations - scikit-learn: Model training & evaluation

Used for: Data loading, cleaning, and transformation.
Example: read_csv(), groupby(), get_dummies()

Shap: Model interpretability - StandardScaler & LabelEncoder: Preprocessing - DecisionTreeClassifier,

- Used for: Explaining model predictions (feature impact).
- Example: shap.summary_plot(), shap.Explainer()

LogisticRegression: Predictive modeling

seaborn

- Used for: Creating statistical plots (e.g., heatmaps, boxplots).
- Example: heatmap(), barplot()

matplotlib

- Used for: Saving and customizing visualizations.
- Example: pyplot.savefig(), pyplot.title()

scikit-learn

- Used for: Preprocessing, model building, and evaluation.
- Key modules:
    - train_test_split(): Split data into train/test sets.
    - LogisticRegression, DecisionTreeClassifier: Build models.
    - classification_report(), confusion_matrix(): Evaluate performance.

StandardScaler (from sklearn)

- Used for: Scaling features to improve model performance.

LabelEncoder (from sklearn)

- Used for: Converting categorical labels into numeric values.

EDA (Exploratory Data Analysis)

Insights into attrition by department, salary bands, and years since last promotion were visualized using bar-charts and histograms. These visualizations helped identify key problem areas such as departments with high turnover or infrequent promotions.

Data Preprocessing

- Dropped null values and constant columns - Encoded categorical data using LabelEncoder and One-Hot **Encoding.**
- Applied standard scaling using StandardScaler - Separated features and target into X and y - Split into training and test datasets

1. Data Loading and Initial Exploration:

The code starts by loading the dataset from `HR_Analytics.csv` into a pandas DataFrame.
It then performs initial exploratory data analysis by summarizing attrition rates by department, analyzing salary band distributions, and examining years since last promotion.
Visualizations (bar charts and histograms) are generated for these initial insights and saved as image files.

2. Data Preprocessing for Modeling:

A heatmap is generated to visualize the correlation between different features in the dataset. Before creating the heatmap, categorical columns are label-encoded. This helps identify potential relationships between variables, including those related to attrition.

The code prepares the data for predictive modeling:

It handles missing values by dropping rows with any nulls.
It removes potentially non-informative or constant columns like 'SalarySlab', 'EmployeeCount', 'StandardHours', 'Over18', and 'emp_id'.

Categorical features are encoded:

Label encoding is applied to columns with two unique values (likely binary).

One-hot encoding (`pd.get_dummies`) is applied to nominal categorical columns with more than two unique values. `drop_first=True` is used to avoid multicollinearity.

3. Predictive Modeling:

The preprocessed data is split into training and testing sets using `train_test_split` with a test size of 20% and stratified sampling to maintain the proportion of the target variable (Attrition) in both sets.

1)Logistic Regression Model:

A Logistic Regression model is initialized and trained on the training data. `max_iter=1000` is used to ensure convergence, and' `solver='liblinear'` is specified, which is suitable for smaller datasets and provides L1/L2 regularization. `random_state=42` ensures reproducibility. Predictions are made on the test set.

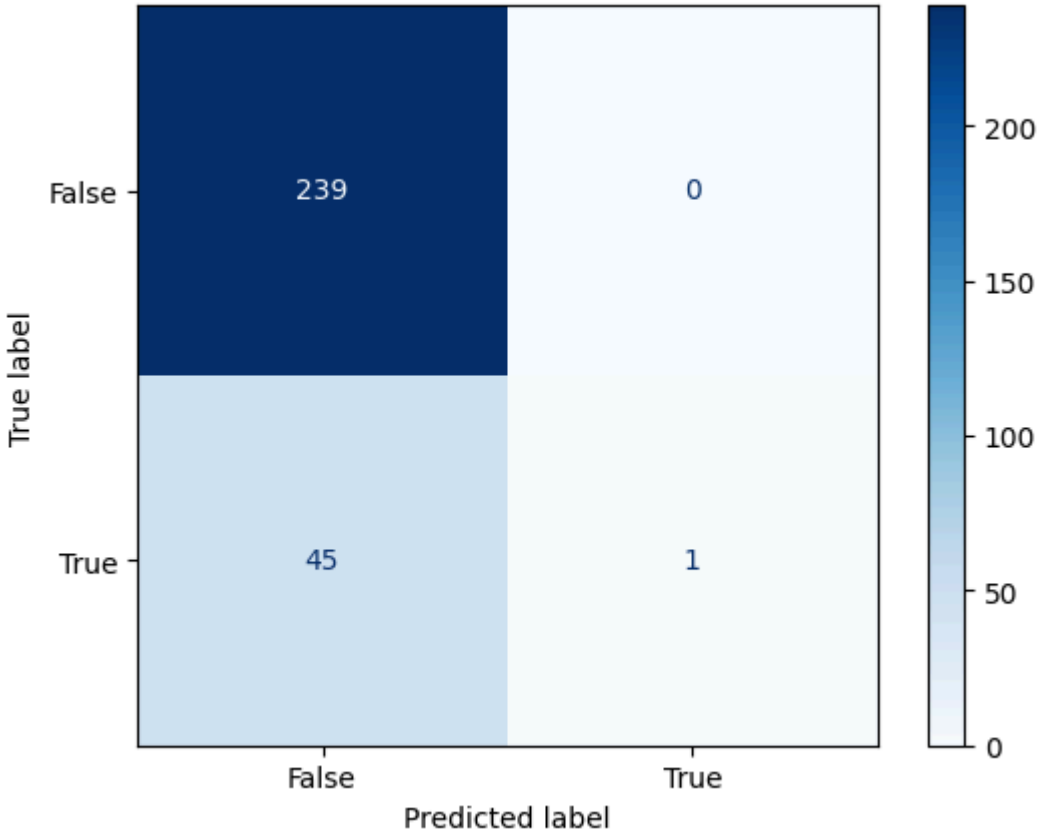
The model's performance is evaluated using standard metrics:

a.Accuracy:

The proportion of correctly predicted instances. Classification Report:Provides precision,recall, F1-score, and support for each class (Attrition: Yes/No).

b.Confusion Matrix:

A table showing the counts of true positive, true negative, false positive, and false negative predictions.

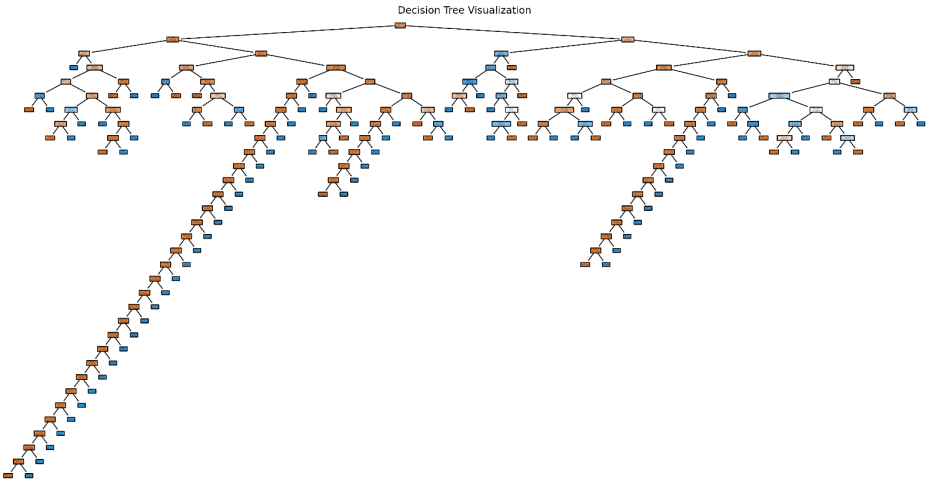


2)Decision Tree Model:

A Decision Tree Classifier is initialized and trained on the training data. `random_state=42` is used for reproducibility. Predictions are made on the test set.

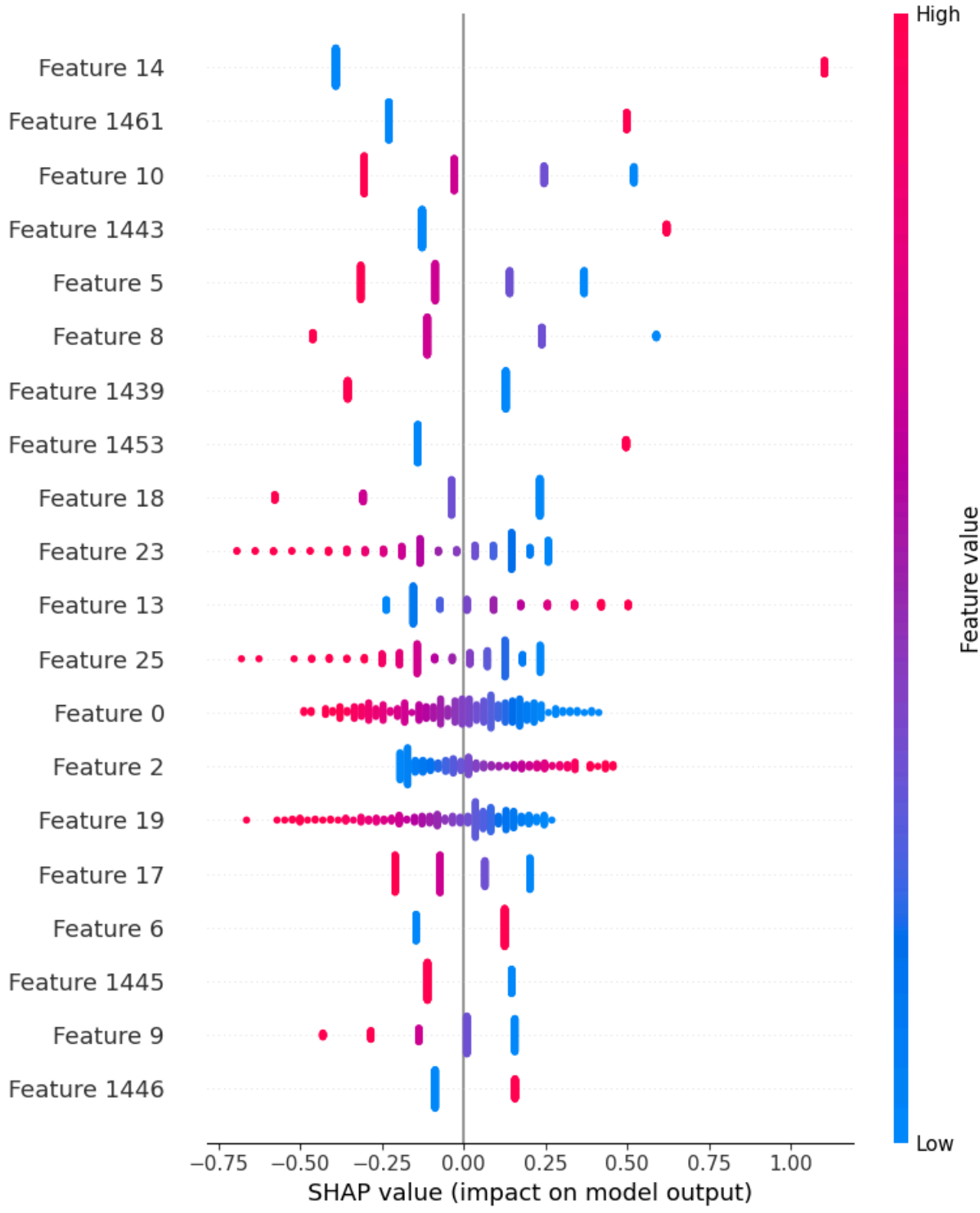
The model's performance is evaluated using accuracy, classification report, and confusion matrix, similar to the Logistic Regression model.

The code then visualizes the trained Decision Tree, showing the decision rules learned by the model. This visualization is saved as `decision_tree.png`.



4. SHAP (SHapley Additive exPlanations) Analysis:

1. SHAP analysis is performed to explain the predictions of the trained Decision Tree model.
2. It uses the `shap` library to create an explainer and calculate SHAP values for the test set.
3. Before calculating SHAP values, the code explicitly converts features in `X_train` and `X_test` to numeric types and handles any resulting NaNs by filling them with the mean of the respective column. This is a crucial step as SHAP often requires numerical input.
4. A SHAP summary plot is generated, which provides a global view of feature importance and how each feature impacts the model's output (the likelihood of attrition).



5. Additional Model Training and Evaluation (Logistic Regression with Standardization and Class Weighting):

The code repeats the model building process, but this time for a Logistic Regression model with additional steps:

1. It re-applies one-hot encoding to `X_train`, `X_test`, `y_train`, and `y_test`. Note that applying `get_dummies` to `y_train` and `y_test` when `y` is already binary (0 or 1 after label encoding) might be unnecessary and potentially cause issues depending on the Logistic Regression implementation. Assuming 'Attrition' was encoded as 0/1 earlier, this step might be a remnant or intended for a different model.
2. It uses `StandardScaler` to standardize the features in `X_train` and `X_test`. Standardization is often beneficial for models like Logistic Regression.
3. It initializes a Logistic Regression model with `class_weight="balanced"`. This is important for imbalanced datasets (where the number of employees who attrit is significantly less than those who don't) to prevent the model from being biased towards the majority class.
4. The model is trained on the standardized training data.
5. Predictions are made and evaluated using accuracy, confusion matrix, and classification report.
   A Confusion Matrix Display is also plotted.
6. Finally, SHAP analysis is performed again, this time explaining the predictions of this standardized Logistic Regression model.

Summary:

The code performs a comprehensive analysis of HR data focusing on employee attrition. It includes:

1. **Exploratory Data Analysis (EDA):** Understanding basic patterns in attrition, salary, and **promotions.**
2. **Data Preprocessing:**Handling categorical data and preparing features for modeling.
3. **Predictive Modeling:**Building and evaluating Logistic Regression and Decision Tree models to predict employee attrition.
4. **Model Interpretation:**Using Decision Tree visualization and SHAP analysis to understand how the models make predictions and which features are most influential in predicting attrition.
5. **Addressing Class Imbalance:** Using `class_weight="balanced"` in one of the Logistic Regression models to mitigate the impact of an unequal distribution of the target variable.

The analysis aims to identify factors contributing to employee attrition and build models that can predict which employees are likely to leave, providing valuable insights for HR decision-making.