IBM HR ANALYTICS EMPLOYEE ATTRITION PROJECT REPORT

Executive Summary

This project aimed to predict employee attrition at IBM using a provided dataset. The problem was framed as a binary classification task. Initial model training with `RandomForestClassifier` showed high accuracy for employees who stay but very poor recall for employees who leave, indicating a class imbalance problem. To address this, SMOTE (Synthetic Minority Over-sampling Technique) was applied to the training data, significantly improving the model's ability to identify employees at high risk of leaving. The final model achieved an improved recall for the 'Attrition: Yes' class, making it more effective for the intended purpose. Feature importance analysis revealed key factors influencing attrition, such as MonthlyIncome, Age, and OverTime.

Project Goal

The primary goal of this project was to predict which employees are at high risk of leaving the company. This is a binary classification problem, where the target variable is `Attrition`, with possible outcomes of "Yes" (employee will leave) or "No" (employee will stay).

Data Loading and Initial Exploration

The dataset, `IBM HR Employee Attrition Data.csv`, was loaded into a pandas DataFrame. Initial inspection using `df.head()` showed the first five rows of the data, revealing a mix of numerical and categorical features.

`df.info()` revealed that the dataset contains 1470 entries and 35 columns. All columns were found to have 1470 non-null entries, indicating no missing values in the dataset. The data types included 26 integer columns (`int64`) and 9 object columns, which represented categorical features.

`df.describe()` provided statistical summaries for the numerical columns, showing distributions, means, standard deviations, and quartiles for features like Age, DailyRate, MonthlyIncome, etc. This gave an initial understanding of the data's scale and spread.

Further checks for data quality confirmed that there were **no null values** (`df.isnull().sum()` showed all zeros) and **no duplicate rows** (`df.duplicated().sum()` returned 0).

Data Cleaning and Preprocessing

Several preprocessing steps were performed to convert categorical features into a numerical format suitable for machine learning models:

a. **Binary Encoding**: The following categorical columns were converted to numerical binary representations:

- `Attrition`: 'Yes' was mapped to 1, 'No' to 0.

- `Gender`: 'Male' was mapped to 1, 'Female' to 0.

- `Over18`: 'Y' was mapped to 1.

- `OverTime': 'Yes' was mapped to 1, 'No' to 0.

b. **One-Hot Encoding**: Five categorical columns with multiple unique values were one-hot encoded using `pd.get_dummies()`:

- `BusinessTravel`

- `Department`

- `EducationField`

- `JobRole`

- `MaritalStatus`

For each of these columns, new binary columns were created for each unique category (e.g., `Travel_Rarely`, `Department_Sales`). The original categorical columns were then dropped from the DataFrame. After creating dummy variables, the resulting `True`/`False` boolean values in the new columns were explicitly converted to 1s and 0s.

c. **Column Dropping**: Several columns were identified as constant or irrelevant for prediction and were subsequently dropped:

- `EmployeeNumber`: Unique identifier, not predictive.

- `EmployeeCount`: Contained only one value (1) for all entries.

- `Over18`: Contained only one value (1, after binary encoding) for all entries.

- `StandardHours`: Contained only one value (80) for all entries.

These columns provided no variance and thus no predictive power.

After these preprocessing steps, the DataFrame had 50 columns, all in numerical format.

Initial Model Training and Evaluation

A `RandomForestClassifier` was chosen for the initial model training. The dataset was split into training (80%) and testing (20%) sets using `train_test_split` with `random_state=42` for reproducibility. The target variable `Attrition` was separated as `y`, and the remaining features as `X`.

The initial model was trained and evaluated:

- **Accuracy Score**: The model achieved an accuracy of approximately **0.8776 (87.76%)** on the test set.

- **Classification Report Analysis**:

```
precision recall f1-score support

0 0.88 1.00 0.93 255

1 1.00 0.08 0.14 39

accuracy 0.88 294

macro avg 0.94 0.54 0.54 294

weighted avg 0.89 0.88 0.83 294
```

The report highlighted a significant issue: while the accuracy seemed high, it was misleading due to class imbalance. The model was excellent at predicting employees who would **not** leave (Class 0, `recall: 1.00`), successfully identifying all of them. However, for employees who **would** leave (Class 1, `Attrition: Yes`), the recall was critically low at **0.08**. This meant the model could only identify 8% of the actual employees who attrited, rendering it ineffective for the project's goal of identifying high-risk employees.

Addressing Class Imbalance with SMOTE

To resolve the severe class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) was applied to the training data. SMOTE works by creating synthetic samples for the minority class, thereby balancing the class distribution in the training set.

SMOTE was applied using `imblearn.over_sampling.SMOTE` with `random_state=42`. After resampling:

- **Training shape**: `(1956, 49)`

- **Training target counts**:

`Attrition`

`0 978`

`1 978`

This shows that the training set now has an equal number of samples for both 'No' (0) and 'Yes' (1) attrition classes.

The `RandomForestClassifier` was retrained on the SMOTE-resampled training data. The model was then evaluated on the original, untouched test set.

- **Classification Report (After SMOTE)**:
```
Classification Report (After SMOTE):
precision recall f1-score support

0 0.90 0.97 0.94 255
1 0.63 0.31 0.41 39

accuracy 0.88 294
macro avg 0.77 0.64 0.67 294
weighted avg 0.87 0.88 0.87 294
```

After applying SMOTE, the accuracy remained similar (0.88), but there was a notable improvement in the recall for the minority class (Class 1, 'Attrition: Yes'), increasing from 0.08 to **0.31**. This indicates that the model is now better at identifying employees who are likely to leave, capturing 31% of them. While the precision for Class 1 decreased (from 1.00 to 0.63), indicating more false positives, the trade-off was necessary to improve the detection of actual attrition cases. The `f1-score` for Class 1 also improved significantly (from 0.14 to 0.41), showing a better balance between precision and recall for the minority class.

Feature Importance Analysis

To understand which features were most influential in predicting attrition, the `feature_importances_` attribute of the trained `RandomForestClassifier` model was analyzed. The top 10 features were extracted and sorted by their importance score:

Top 10 Features Driving Attrition:
```
Feature Importance
10 MonthlyIncome 0.068380
0 Age 0.059436
13 OverTime 0.055198
1 DailyRate 0.051242
11 MonthlyRate 0.050756
18 TotalWorkingYears 0.048678
6 HourlyRate 0.046383
2 DistanceFromHome 0.044959
```

```
21 YearsAtCompany 0.041681

14 PercentSalaryHike 0.033438
```

This analysis shows that `MonthlyIncome`, `Age`, and `OverTime` are the most significant predictors of employee attrition. Other important factors include `DailyRate`, `MonthlyRate`, `TotalWorkingYears`, `HourlyRate`, `DistanceFromHome`, `YearsAtCompany`, and `PercentSalaryHike`.

A bar chart visualization of these feature importances clearly illustrated the relative contribution of each feature, with the top features standing out prominently. This helps in understanding the underlying drivers of attrition and can guide HR strategies to retain employees.

Conclusion:

The analysis identified key factors influencing employee attrition and developed a classification model. Although SMOTE improved the model's ability to detect employees who leave, there is still room for improvement in recall for the minority class. Understanding these important features can guide HR strategies to retain employees.