Employee Attrition Prediction & Analysis Full Documentation

This document provides full documentation for an employee attrition analysis project, covering each step from data understanding to Deployment.

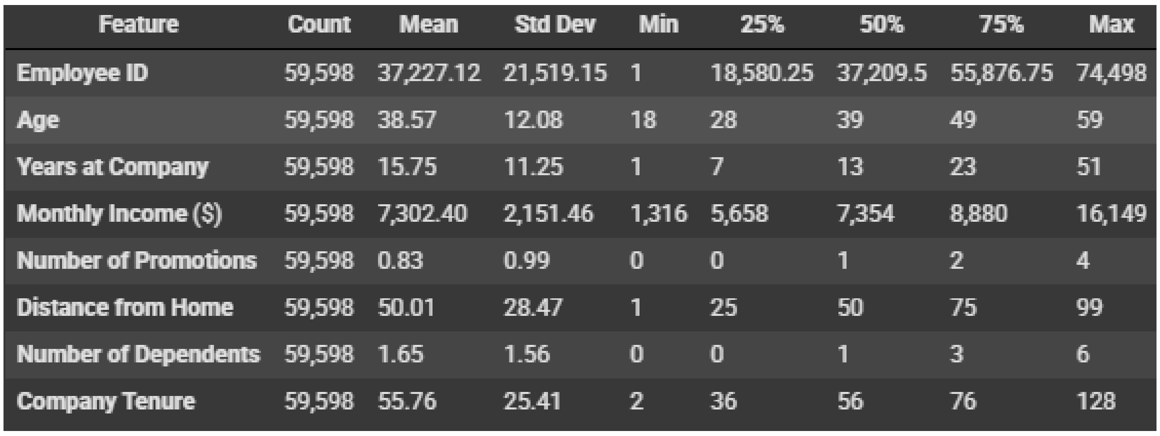
Table Of Contents

1. Define the Project and Objective

The purpose of this project is to analyze employee attrition to uncover patterns and factors that influence employees to leave an organization. We aim to:
- Identify key predictors of attrition.
- Understand trends across demographics, job roles, and satisfaction metrics.
- Help HR departments devise better retention strategies.

2. Data Presentation

The dataset contains over 59,598 employee records and includes:
- Demographics: Age, Gender, Marital Status
- Career Info: Job Role, Job Level, Monthly Income
- Satisfaction Metrics: Work-Life Balance, Job Satisfaction, Performance Rating
- Target Variable: Attrition (1 = Left, 0 = Stayed)
- All fields are complete with no missing values.



Feature	Count	Mean	Std Dev	Min	25%	50%	75%	Max
Employee ID	59,598	37,227.12	21,519.15	1	18,580.25	37,209.5	55,876.75	74,498
Age	59,598	38.57	12.08	18	28	39	49	59
Years at Company	59,598	15.75	11.25	1	7	13	23	51
Monthly Income ($)	59,598	7,302.40	2,151.46	1,316	5,658	7,354	8,880	16,149
Number of Promotions	59,598	0.83	0.99	0	0	1	2	4
Distance from Home	59,598	50.01	28.47	1	25	50	75	99
Number of Dependents	59,598	1.65	1.56	0	0	1	3	6
Company Tenure	59,598	55.76	25.41	2	36	56	76	128

3. Data Wrangling

3.1 Data Cleaning

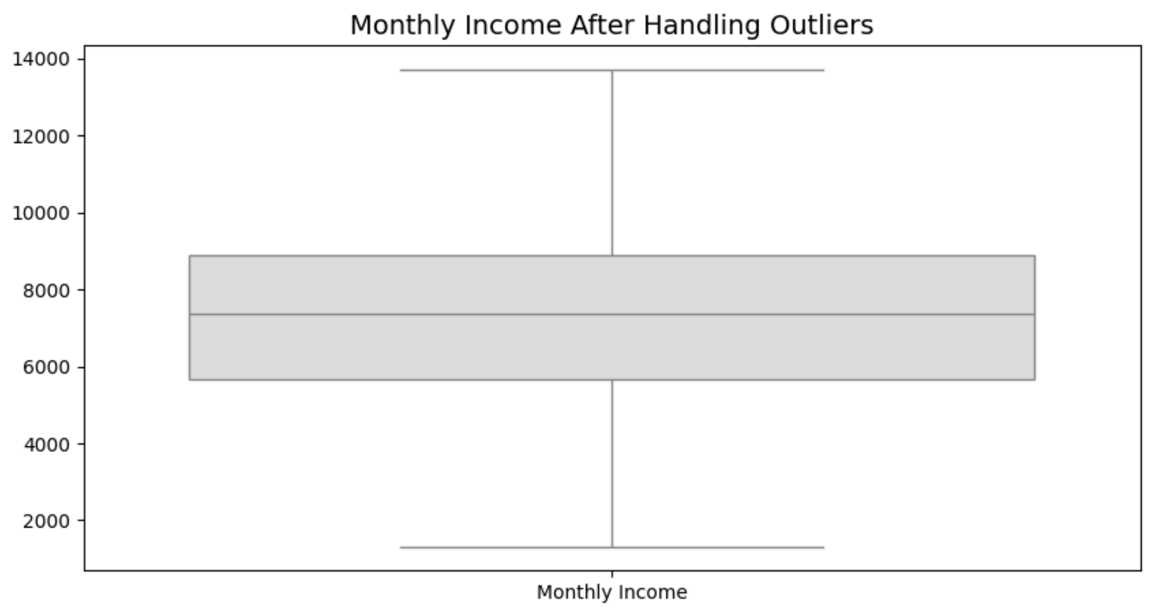
- Verified data types.
- Removed irrelevant columns such as Employee ID.

3.2 Handling Duplicates

- No duplicate records found.

3.3 Handling Outliers

- Detected 50 outliers in Monthly Income.
- Applied Winsorization to cap extreme values.



3.4 Dropping Unnecessary Columns

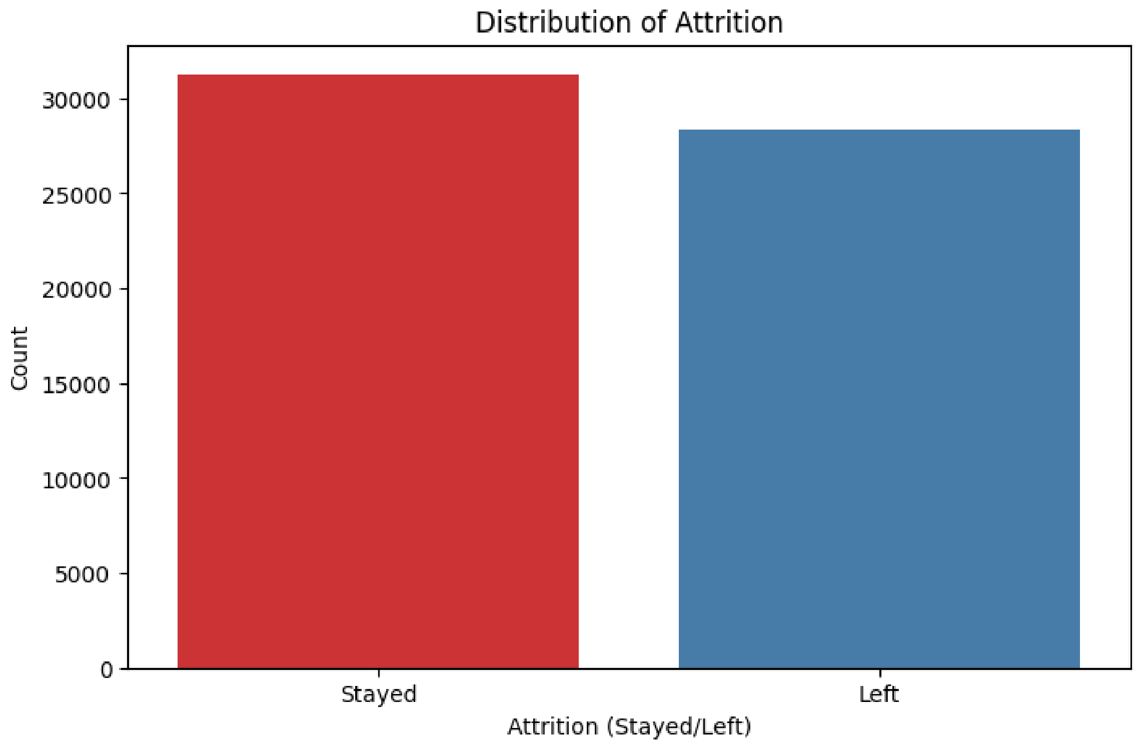
- Dropped columns with constant values or no analytical use, such as **Employee ID**

4. 📊 Exploratory Data Analysis

4.1 Univariate Analysis

Each column was analyzed individually to understand its distribution:

- **Attrition: Slight imbalance**; more employees stayed (31,260) than left (28,338).



- **Age**: Range 18-59; most common ages 34–47.
- **Job Role**: Technology most common (15,507), followed by Healthcare and Education.
- **Work-Life Balance**: Mostly 'Good' and 'Fair'; 8,305 rated 'Poor'.
- **Job Satisfaction**: Diverse spread from Very Low to High.
- **Monthly Income**: Right-skewed; large concentration under $5,000/month.
- **Performance Rating**: Majority rated Average (35,810).
- **Marital Status**: Mostly Married, then Single and Divorced.
- **Years at Company**: Many employees between 1-5 years.
- **Distance from Home**: Range 9–99 miles; common values between 18 and 53.

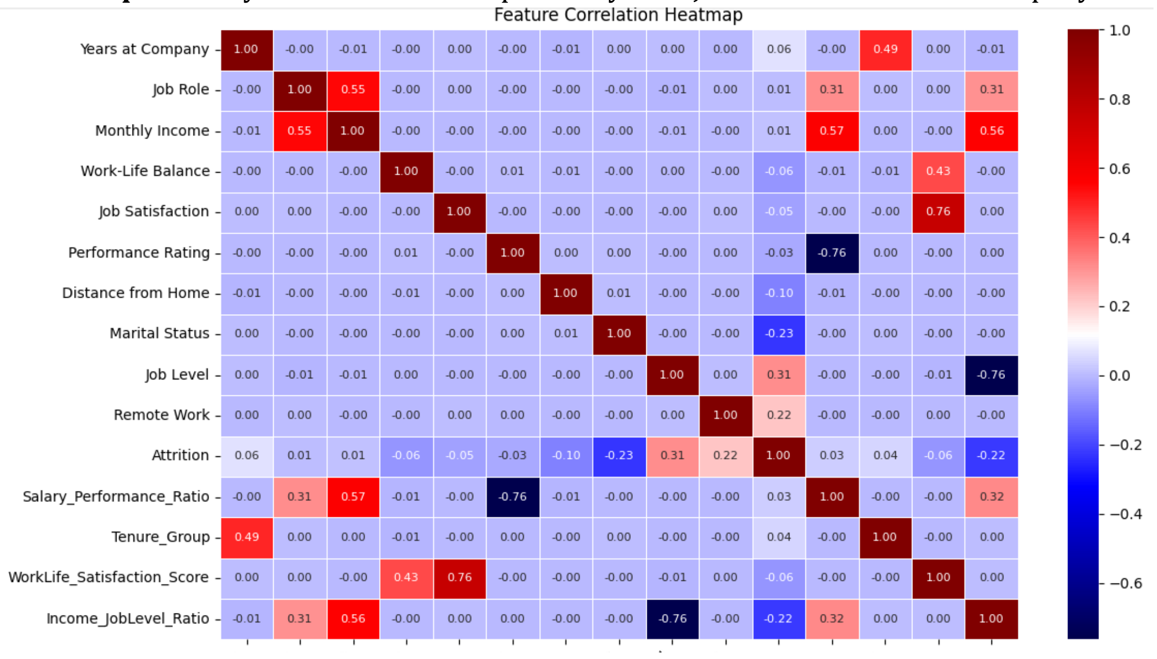
4.2 Bivariate Analysis

Analyzed how each feature correlates with attrition:

- **Age vs Monthly Income**: Income increases with age; highest variability in younger employees.
- **Attrition vs Monthly Income**: Employees who left had lower median salaries.
- **Attrition vs Job Satisfaction**: Lower satisfaction linked with higher attrition.
- **Attrition vs Work-Life Balance**: 'Poor' and 'Fair' balance groups had higher turnover.
- **Attrition vs Performance Rating**: Low performers left more often.
- **Attrition vs Marital Status: Single** employees had higher attrition rates.
- **Attrition vs Overtime**: Strong link between overtime and attrition.
- **Attrition vs Job Role:** Technology and Media had the highest attrition.
- **Attrition vs Years at Company**: Early tenure (1–5 years) showed higher risk.

4.3 Multivariate Analysis

- **Heatmap**: Monthly Income correlated positively with Job Level and Years at Company.



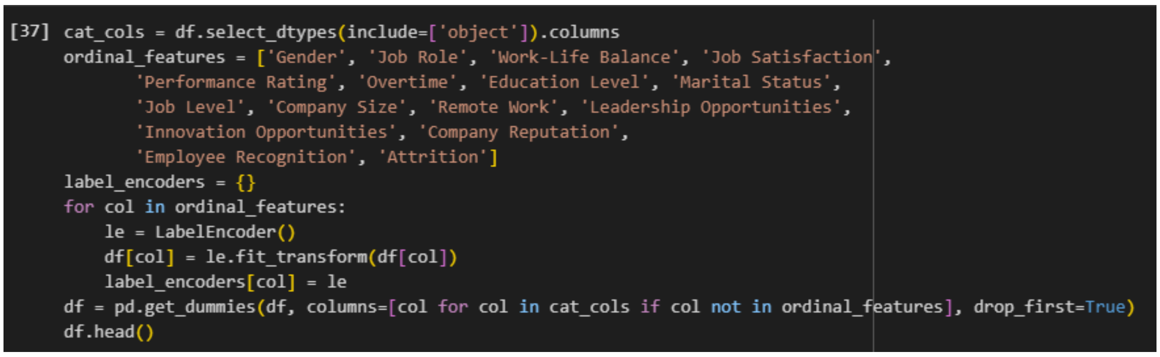
Years at Company	1.00	-0.00	-0.01	-0.00	0.00	-0.00	-0.01	0.00	0.00	0.00	0.06	-0.00	0.49	0.00	-0.01
Job Role	-0.00	1.00	0.55	-0.00	0.00	-0.00	-0.00	-0.00	-0.01	0.00	0.01	0.31	0.00	0.00	0.31
Monthly Income	-0.01	0.55	1.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.01	-0.00	0.01	0.57	0.00	-0.00	0.56
Work-Life Balance	-0.00	-0.00	-0.00	1.00	-0.00	0.01	-0.01	-0.00	0.00	-0.00	-0.06	-0.01	-0.01	0.43	-0.00
Job Satisfaction	0.00	0.00	-0.00	-0.00	1.00	-0.00	-0.00	-0.00	-0.00	0.00	-0.05	-0.00	-0.00	0.76	0.00
Performance Rating	-0.00	-0.00	-0.00	0.01	-0.00	1.00	0.00	0.00	-0.00	0.00	-0.03	-0.76	0.00	-0.00	0.00
Distance from Home	-0.01	-0.00	-0.00	-0.01	-0.00	0.00	1.00	0.01	-0.00	-0.00	-0.10	-0.01	-0.00	-0.00	-0.00
Marital Status	0.00	-0.00	-0.00	-0.00	-0.00	0.00	0.01	1.00	-0.00	-0.00	-0.23	-0.00	0.00	-0.00	-0.00
Job Level	0.00	-0.01	-0.01	0.00	-0.00	-0.00	-0.00	-0.00	1.00	0.00	0.31	-0.00	-0.00	-0.01	-0.76
Remote Work	0.00	0.00	-0.00	-0.00	0.00	0.00	-0.00	-0.00	0.00	1.00	0.22	-0.00	-0.00	0.00	-0.00
Attrition	0.06	0.01	0.01	-0.06	-0.05	-0.03	-0.10	-0.23	0.31	0.22	1.00	0.03	0.04	-0.06	-0.22
Salary_Performance_Ratio	-0.00	0.31	0.57	-0.01	-0.00	-0.76	-0.01	-0.00	-0.00	-0.00	0.03	1.00	-0.00	-0.00	0.32
Tenure_Group	0.49	0.00	0.00	-0.01	-0.00	0.00	-0.00	0.00	-0.00	-0.00	0.04	-0.00	1.00	-0.00	0.00
WorkLife_Satisfaction_Score	0.00	0.00	-0.00	0.43	0.76	-0.00	-0.00	-0.00	-0.01	0.00	-0.06	-0.00	-0.00	1.00	0.00
Income_JobLevel_Ratio	-0.01	0.31	0.56	-0.00	0.00	0.00	-0.00	-0.00	-0.76	-0.00	-0.22	0.32	0.00	0.00	1.00

- Attrition showed inverse correlation with Monthly Income, Job Satisfaction, and Performance Rating.
- Pairplots: Explored distributions of Age, Income, Job Level vs Attrition.
- Parallel Coordinates Plot: Clearly highlighted factors like low Job Satisfaction and Overtime linked with

attrition.
- Animated Scatter: Visualized attrition dynamics over tenure, age, and salary.
- Sunburst Chart: Illustrated attrition by Job Role and Years at Company.

5. Encoding

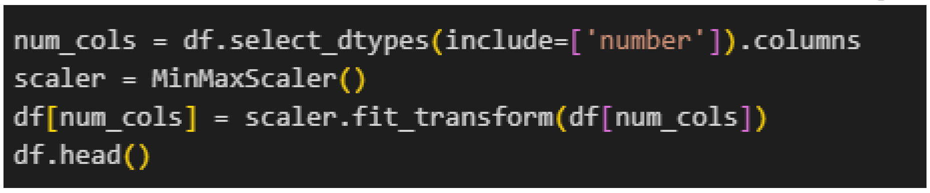
- Label Encoding: Applied to binary columns (e.g., Gender, Remote Work).
- One-Hot Encoding: Used for non-ordinal categorical columns (e.g., Job Role, Marital Status) to preserve interpretability.



```
[37] cat_cols = df.select_dtypes(include=['object']).columns
     ordinal_features = ['Gender', 'Job Role', 'Work-Life Balance', 'Job Satisfaction',
            'Performance Rating', 'Overtime', 'Education Level', 'Marital Status',
            'Job Level', 'Company Size', 'Remote Work', 'Leadership Opportunities',
            'Innovation Opportunities', 'Company Reputation',
            'Employee Recognition', 'Attrition']
     label_encoders = {}
     for col in ordinal_features:
         le = LabelEncoder()
         df[col] = le.fit_transform(df[col])
         label_encoders[col] = le
     df = pd.get_dummies(df, columns=[col for col in cat_cols if col not in ordinal_features], drop_first=True)
     df.head()
```

6. Normalizing

- StandardScaler applied to numerical columns like Age, Monthly Income, and Distance From Home.
- Ensures features have zero mean and unit variance for better model performance.



```
num_cols = df.select_dtypes(include=['number']).columns
scaler = MinMaxScaler()
df[num_cols] = scaler.fit_transform(df[num_cols])
df.head()
```

7. Advanced Analysis Explanation

This section details the advanced statistical and feature selection techniques used to enhance the understanding and predictive power of the attrition model.

7.1 Overview

Advanced analysis focused on identifying the most influential factors affecting attrition through statistical tests and feature selection methods. These techniques improved model accuracy by prioritizing relevant predictors.

the most important factors influencing attrition.

7.2 Statistical Tests for Feature Analysis

Statistical tests were performed to determine the relationship between various employee attributes and attrition. The following methods were used:

7.2.1 T-Test (Independent Samples T-Test)

The T-test compares the means of numerical variables (e.g., salary) between two groups: employees who left and those who stayed. A low p-value (< 0.05) indicates that the variable significantly influences attrition. For example, a low p-value for 'Monthly Income' suggests that salary plays a crucial role in an employee's decision to leave.

```
 t_test_results = {feature: ttest_ind(df[df["Attrition"] == 0][feature],
                                      df[df["Attrition"] == 1][feature],
                                      equal_var=False)[1]
                   for feature in cat_cols}
```

7.2.2 ANOVA (Analysis of Variance)

ANOVA compares the means of multiple groups (e.g., job levels) to see if at least one differs significantly. This helps analyze whether factors like 'Job Level' or 'Performance Rating' impact attrition. A p-value < 0.05 confirms a significant difference among groups.

```
anova_results = {feature: f_oneway(df[df["Attrition"] == 0][feature],
                                   df[df["Attrition"] == 1][feature])[1]
                 for feature in cat_cols}
```

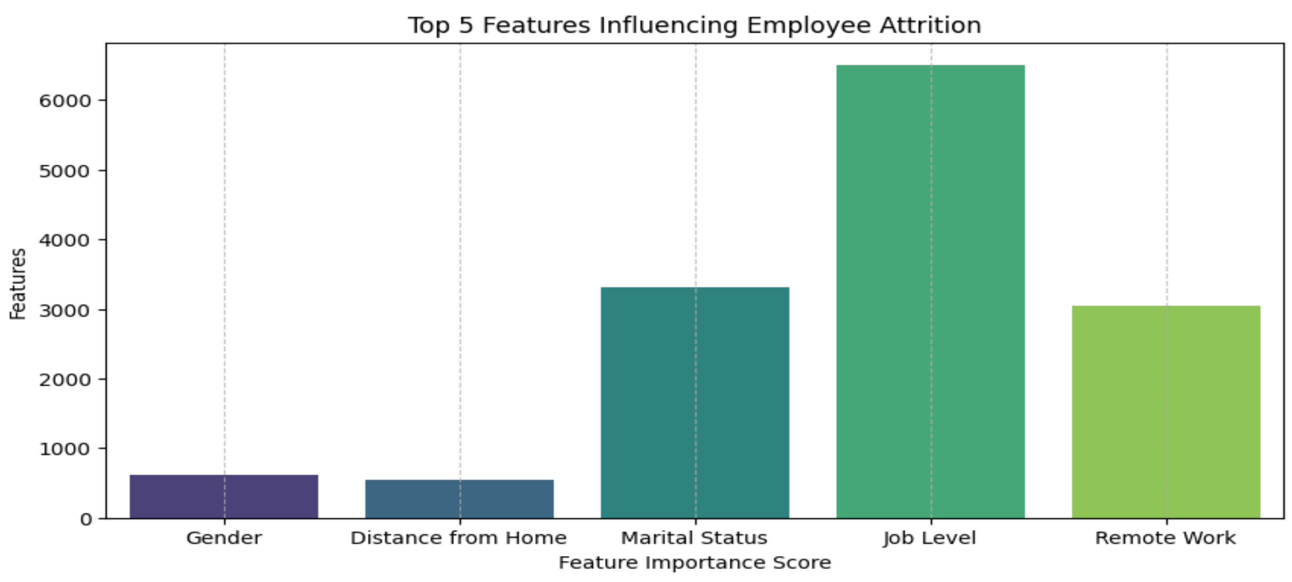
7.2.3 Chi-Squared Test (For Categorical Features)

The Chi-Square test checks whether categorical features (e.g., Job Role, Remote Work) are related to attrition. A low p-value means the feature is significantly related to an employee's decision to stay or leave. For example, if 'Remote Work' has a low p-value, it suggests that employees with flexible work options are less likely to leave.

```
chi2_results = {}
for feature in num_cols:
    contingency_table = pd.crosstab(df[feature], df["Attrition"])
    chi2_results[feature] = chi2_contingency(contingency_table)[1]
```

7.3 Feature Selection Techniques

To enhance model accuracy, feature selection techniques were used to identify the most important predictors of attrition.



7.3.1 Recursive Feature Elimination (RFE)

RFE iteratively removes less important features until only the best predictors remain. A machine learning model (Random Forest) was used to rank the most influential variables, ensuring the selection of features with the highest impact on attrition.

7.3.2 SelectKBest (ANOVA F-test)

SelectKBest ranks features based on their statistical significance using ANOVA F-tests. The top-ranked variables are selected as the most relevant for predicting attrition. This method ensures that only the strongest predictors are included in the final model.

7.4 Key Insights & Business Recommendations

- Employees with lower salaries and fewer promotions are more likely to leave.
- Remote Work reduces attrition, while longer commutes increase it.
- Job Level is a strong predictor – higher positions have lower turnover.
- The selected features should be used in the predictive model to enhance accuracy.

8. Machine Learning Model Development

This section outlines the development, evaluation, and optimization of machine learning models to predict employee attrition.

8.1 Introduction

Employee attrition impacts organizational stability and productivity. This report details the process of building a machine learning model to predict attrition based on historical employee data, enabling proactive retention strategies.

8.2 Data Description

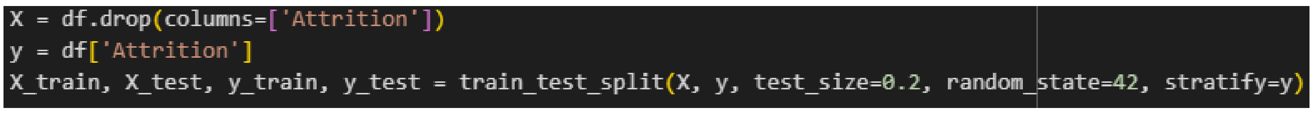
The dataset includes employee attributes such as:

- Age, Gender, Years at Company, Monthly Income
- Job Role, Job Satisfaction Level These features were used to train models to predict the binary target variable (Attrition: 0 = Stayed, 1 = Left).

8.3 Machine Learning Model Development

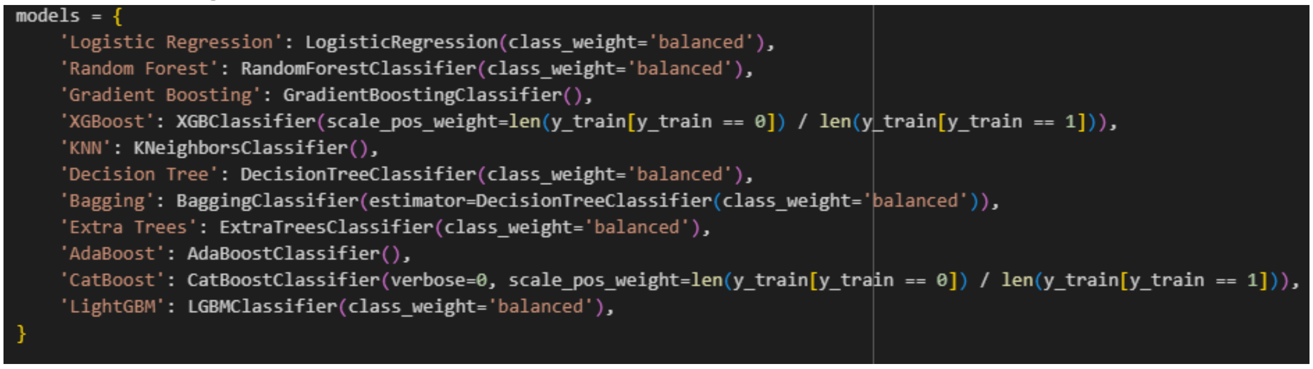
The model development process involved several stages:

- **8.3.1 Data Splitting**: The dataset was split into 80% training and 20% testing sets.



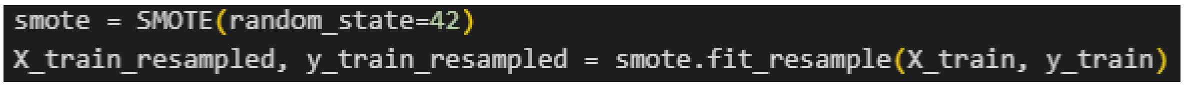
```
X = df.drop(columns=['Attrition'])
y = df['Attrition']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
```

- **8.3.2 Model Selection**: Multiple algorithms were tested, including Logistic Regression, Random Forest, XGBoost, Gradient Boosting, KNN, Decision Tree, Bagging, Extra Trees, AdaBoost, CatBoost, and LightGBM.



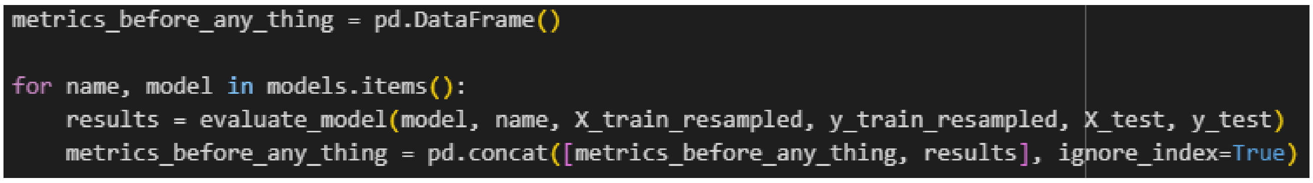
```
models = {
    'Logistic Regression': LogisticRegression(class_weight='balanced'),
    'Random Forest': RandomForestClassifier(class_weight='balanced'),
    'Gradient Boosting': GradientBoostingClassifier(),
    'XGBoost': XGBClassifier(scale_pos_weight=len(y_train[y_train == 0]) / len(y_train[y_train == 1])),
    'KNN': KNeighborsClassifier(),
    'Decision Tree': DecisionTreeClassifier(class_weight='balanced'),
    'Bagging': BaggingClassifier(estimator=DecisionTreeClassifier(class_weight='balanced')),
    'Extra Trees': ExtraTreesClassifier(class_weight='balanced'),
    'AdaBoost': AdaBoostClassifier(),
    'CatBoost': CatBoostClassifier(verbose=0, scale_pos_weight=len(y_train[y_train == 0]) / len(y_train[y_train == 1])),
    'LightGBM': LGBMClassifier(class_weight='balanced'),
}
```

- **8.3.3 Handling Imbalanced Data**: SMOTE (Synthetic Minority Over-sampling Technique) was applied to address the mild class imbalance (47% attrition rate), generating synthetic samples for the minority class.



```
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)
```

- **8.3.4Model Training**:
    - *8.3.4.1 Before Anything*: Models were trained on raw data to establish a performance benchmark.

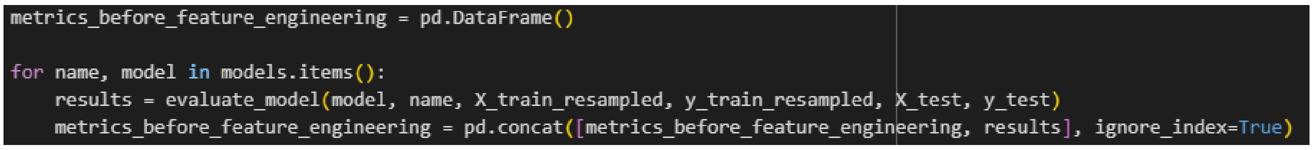


```
metrics_before_any_thing = pd.DataFrame()

for name, model in models.items():
    results = evaluate_model(model, name, X_train_resampled, y_train_resampled, X_test, y_test)
    metrics_before_any_thing = pd.concat([metrics_before_any_thing, results], ignore_index=True)
```

Model	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*ROC-AUC*
Logistic Regression	0.711913	0.734365	0.706174	0.719993	0.791736
Random Forest	0.741023	0.756275	0.746961	0.751589	0.829436
Gradient Boosting	0.757215	0.773542	0.759437	0.766425	0.845837
XGBoost	0.748826	0.769613	0.743762	0.756467	0.837836
KNN	0.661829	0.690972	0.642674	0.665948	0.718812
Decision Tree	0.667114	0.684134	0.678663	0.681388	0.666519
Bagging	0.713423	0.753486	0.674184	0.711633	0.792697
Extra Trees	0.740268	0.753535	0.750160	0.751844	0.825442
AdaBoost	0.754866	0.767427	0.764235	0.765828	0.845456
CatBoost	0.758054	0.778347	0.753199	0.765567	0.846873
LightGBM	0.754698	0.766240	0.765995	0.766117	0.846497

- *8.3.4.2 Before Feature Engineering*: Models were retrained after applying encoding, scaling, drop unneeded columns, and outlier handling, improving performance.

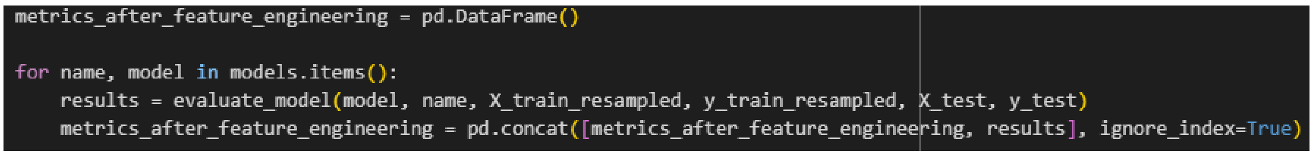


```python
metrics_before_feature_engineering = pd.DataFrame()

for name, model in models.items():
    results = evaluate_model(model, name, X_train_resampled, y_train_resampled, X_test, y_test)
    metrics_before_feature_engineering = pd.concat([metrics_before_feature_engineering, results], ignore_index=True)
```

Model	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*ROC-AUC*
Logistic Regression	0.695302	0.710882	0.706334	0.708601	0.763883
Random Forest	0.706124	0.727906	0.702175	0.714809	0.787999
Gradient Boosting	0.733305	0.760380	0.717690	0.738418	0.816114
XGBoost	0.724748	0.752250	0.708573	0.729759	0.807443
KNN	0.575336	0.599299	0.574376	0.586573	0.601148
Decision Tree	0.639597	0.659180	0.647793	0.653437	0.639113
Bagging	0.681460	0.722333	0.637876	0.677482	0.759104
Extra Trees	0.696477	0.716398	0.697377	0.706760	0.777815
AdaBoost	0.731376	0.760684	0.711772	0.735416	0.815582
CatBoost	0.730621	0.758897	0.712892	0.735175	0.813122
LightGBM	0.731460	0.753447	0.725368	0.739141	0.814421

- *8.3.4.3 After Feature Engineering*: Models were retrained after applying encoding, scaling, drop unneeded columns, Creating New Feature, and outlier handling, improving performance.

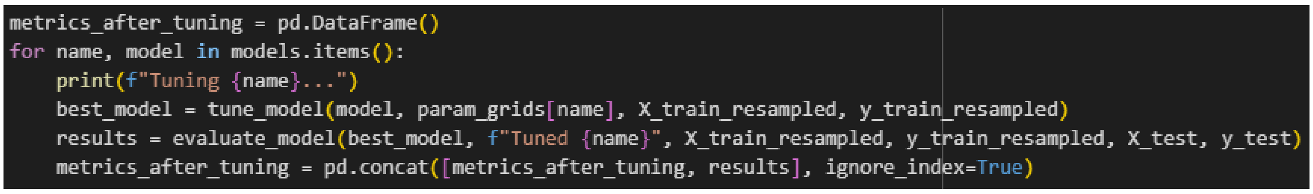


```
metrics_after_feature_engineering = pd.DataFrame()

for name, model in models.items():
    results = evaluate_model(model, name, X_train_resampled, y_train_resampled, X_test, y_test)
    metrics_after_feature_engineering = pd.concat([metrics_after_feature_engineering, results], ignore_index=True)
```

Model	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*ROC-AUC*
Logistic Regression	0.711074	0.734312	0.703775	0.718719	0.791950
Random Forest	0.739178	0.754576	0.745042	0.749779	0.827726
Gradient Boosting	0.756963	0.772720	0.760237	0.766427	0.845643
XGBoost	0.749413	0.770505	0.743762	0.756898	0.836607
KNN	0.663507	0.692031	0.645873	0.668156	0.718670
Decision Tree	0.667282	0.687070	0.671465	0.679178	0.667066
Bagging	0.712919	0.752048	0.675304	0.711613	0.796563
Extra Trees	0.737668	0.754189	0.741523	0.747802	0.824882
AdaBoost	0.754950	0.767464	0.764395	0.765927	0.845443
CatBoost	0.755705	0.774490	0.753679	0.763943	0.846536
LightGBM	0.756879	0.768750	0.767274	0.768012	0.846882

- *8.3.4.4 Fully Optimized Model Performance (After Tuning)*: Models were trained with both **feature engineering and hyperparameter tuning** applied. This scenario showed the best performance, combining all enhancements.



```
metrics_after_tuning = pd.DataFrame()
for name, model in models.items():
    print(f"Tuning {name}...")
    best_model = tune_model(model, param_grids[name], X_train_resampled, y_train_resampled)
    results = evaluate_model(best_model, f"Tuned {name}", X_train_resampled, y_train_resampled, X_test, y_test)
    metrics_after_tuning = pd.concat([metrics_after_tuning, results], ignore_index=True)
```

Model	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*ROC-AUC*
Logistic Regression	0.711409	0.734646	0.704095	0.719046	0.791321
Random Forest	0.745134	0.761897	0.747761	0.754763	0.833280
Gradient Boosting	0.757383	0.769231	0.767754	0.768492	0.846541
XGBoost	0.758893	0.781876	0.749360	0.765273	0.847719
KNN	0.663339	0.691925	0.645553	0.667935	0.720448
Decision Tree	0.712500	0.711737	0.759437	0.734814	0.783606
Bagging	0.736242	0.756944	0.732246	0.744390	0.820562
Extra Trees	0.736242	0.749759	0.746161	0.747956	0.826511
AdaBoost	0.705034	0.723383	0.708573	0.715902	0.788885
CatBoost	0.756544	0.777134	0.751280	0.763988	0.847587
LightGBM	0.759732	0.772172	0.768714	0.770439	0.848601

- 8.3.5 Model Evaluation: Models were assessed using Accuracy, Precision, Recall, F1-Score, ROC-AUC, and Confusion Matrix.

8.4 Results and Recommendations

The LightGBM model outperformed others, achieving the highest accuracy and robustness. Key attrition drivers include:

- Low job satisfaction
- Below-market salaries
- Excessive work hours without promotions **Recommendations**:
- Enhance salary and benefits policies.
- Improve work environment through promotions and professional development.
- Regularly use data analytics to predict and address attrition risks.

8.5 Conclusion

Machine learning models provide valuable insights for reducing attrition. The LightGBM model's high performance supports its use in production, with periodic updates to maintain accuracy as new data becomes available.

9. Employee Attrition MLOps & Deployment

This section describes the MLOps pipeline for automating, deploying, and monitoring the employee attrition prediction model, ensuring scalability and reliability.

9.1 Objective

The MLOps implementation aimed to create a reproducible, maintainable, and production-ready machine learning pipeline for predicting employee attrition. It supports both technical and non-technical users while ensuring continuous performance monitoring.

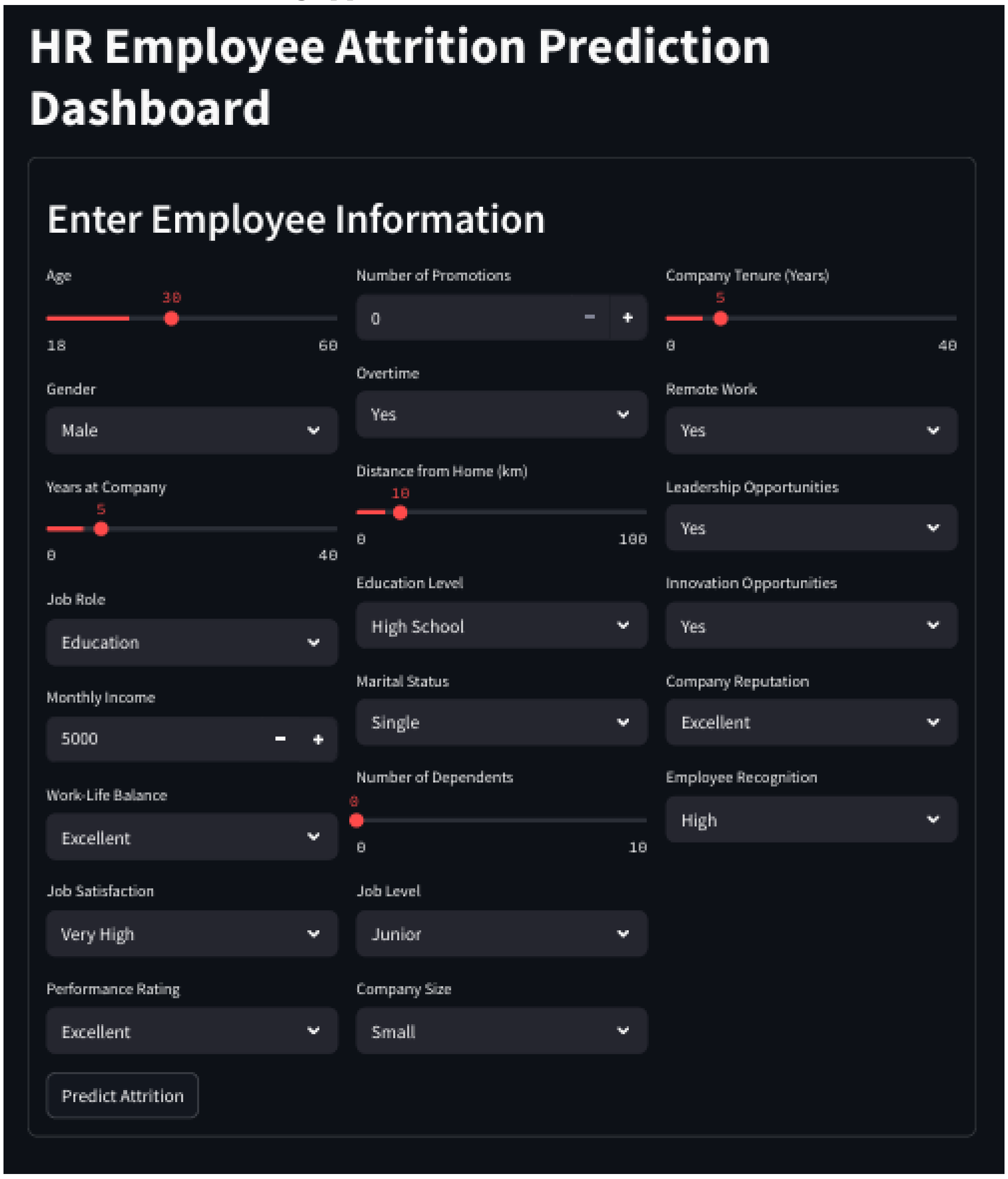
9.2 Tools and Frameworks Summary

- 9.2.1 MLflow: Used for experiment tracking and model registry.
  - Initialized experiments with mlflow.set_experiment().
  - Logged model parameters (e.g., num_leaves, learning_rate for LightGBM) and metrics (accuracy, precision, recall, F1-score).
  - Serialized and logged models using mlflow.sklearn.log_model() for version control.
  - Enabled comparison and selection of the best-performing model via MLflow's UI.

9.3 Model Deployment

The model was deployed in two formats to serve different stakeholders:

- **9.3.1 Flask API**:
    - A RESTful API built with Flask accepts JSON-formatted employee data and returns predictions.
    - The model is loaded using mlflow.sklearn.load_model().
    - Designed for integration into backend systems.
- **9.3.2 Streamlit Dashboard**:
    - A user-friendly interface for HR personnel to input employee data and receive real-time predictions and explanations.
    - Ideal for business-facing applications and demonstrations.



HR Employee Attrition Prediction Dashboard

Enter Employee Information

Age	30 (18–60)
Gender	Male
Years at Company	5 (0–40)
Job Role	Education
Monthly Income	5000
Work-Life Balance	Excellent
Job Satisfaction	Very High
Performance Rating	Excellent
Number of Promotions	0
Overtime	Yes
Distance from Home (km)	10 (0–100)
Education Level	High School
Marital Status	Single
Number of Dependents	0 (0–10)
Job Level	Junior
Company Size	Small
Company Tenure (Years)	5 (0–40)
Remote Work	Yes
Leadership Opportunities	Yes
Innovation Opportunities	Yes
Company Reputation	Excellent
Employee Recognition	High

9.4 Monitoring

Monitoring ensures the deployed model remains reliable:

- 9.4.1 Prometheus: Tracks operational metrics via a /metrics endpoint in the Flask API.
    - Metrics include total API calls, prediction class distribution, and average response latency.
    - Configured for periodic data scraping.
- 9.4.2 Grafana: Can visualize metrics and set alerts for anomalies, enhancing proactive maintenance.

9.5 Automated Model Retraining

An automated retraining pipeline ensures the model stays current:

- Triggered by monitoring alerts or on a schedule.
- Reloads new and historical data, retrains the LightGBM model, and logs experiments in MLflow.
- Compares performance with previous versions, automatically deploying improved models via the model registry.

9.6 Conclusion

The MLOps pipeline ensures a reliable, scalable, and maintainable system for predicting employee attrition. By integrating tracking, deployment, monitoring, and automated retraining, it delivers long-term value to both developers and business stakeholders.

📌 10. Final Business Recommendations

Based on the insights derived from the data analysis and model performance, the following business actions are recommended to reduce attrition:

- **Focus on Early Tenure Retention**: Employees with 1–5 years at the company show a higher tendency to leave. Implement engagement initiatives targeting this group.
- **Adjust Salary Structures**: Lower income was a consistent predictor of attrition. Regular compensation reviews and market alignment are essential to retain talent.
- **Promote Work-Life Balance**: Employees reporting poor work-life balance or frequent overtime are more likely to resign. Offering flexible schedules and promoting wellness can improve retention.
- **Encourage Remote Work**: Employees with remote work options showed lower attrition. Expanding remote/flexible roles could reduce turnover.
- **Invest in Career Development**: High attrition was linked with low job satisfaction and few promotions. Introduce structured mentorship, training, and leadership programs.

- **Monitor High-Risk Roles**: Roles in Technology and Media experienced higher attrition. These departments may need role-specific retention strategies.

11. Conclusion and Future Work

This documentation outlines a full-cycle employee attrition analysis — from raw data exploration through machine learning model deployment. The findings confirm that attrition is influenced by multiple factors including income, satisfaction, commute distance, job role, and opportunities for advancement.

The deployed LightGBM model, supported by a robust MLOps pipeline, provides HR teams with a reliable tool for predicting and managing attrition proactively.

11.1 Future Enhancements:

- Integrate **explainable AI tools** like SHAP for model transparency.
- Feed predictions into HR systems (e.g., SAP, Oracle) to trigger real-time alerts.
- Add survey or performance appraisal data for even richer predictions.
- Expand the model to forecast **employee engagement** or **promotion readiness**.

This project serves as a foundational framework for strategic HR analytics, combining business insight with scalable, automated solutions.

12. Team Members

This project was successfully completed through the collaboration and dedication of the following team members. Each member contributed based on their specialization, mapped to specific milestones:

Ibrahim Abdelsattar – Team Leader – (Milestones 1, 2 & 5)
Directed the overall project vision, coordinated the team's workflow, and ensured timely delivery across all phases. He led the initial data exploration and preprocessing, oversaw advanced data analysis and feature engineering, and compiled the final documentation and presentation materials.

Ali Madian - (*Milestones 1 & 2*)
Focused on early-stage data acquisition, cleaning, and exploratory visualizations. He also contributed to advanced analysis and feature transformation efforts during, helping build a solid analytical foundation for modeling.

Ahmed Tarek - (*Milestone 3***)**
Led the development and optimization of predictive models. He evaluated multiple machine learning algorithms, implemented feature selection techniques, and delivered the final model evaluation report, completing the objectives.

Reem Ashraf – (Milestone 3)
Collaborated with Ahmed Tarek on training and evaluating machine learning models. She contributed to

hyperparameter tuning, model validation, and ensuring the models were optimized for accuracy and generalization

Yousif Saad – (Milestone 4)
Built and deployed the machine learning pipeline into production. He developed the REST API using Flask and implemented monitoring solutions using Prometheus and Grafana, fulfilling the deployment and automation goals of Milestone 4.

Mohamed Abd el-meged – (Milestone 4)
Worked alongside Yousif on backend automation, including model versioning, automated retraining workflows, and integration with MLflow. His contributions ensured scalability and maintainability, completing the operational requirements of Milestone 4.

Each member’s targeted contributions ensured a seamless progression from raw data to a deployed, monitored predictive system. The team's collective expertise delivered a robust, end-to-end solution for understanding and reducing employee attrition.