# Employee Turnover Prediction

Jessica Pham

# Content



**GitHub Repository:** View here

**Source of Data: Kaggle - Link to the Dataset**

**Programming Language: Python**

**Machine Learning Techniques:** Random Forest Classifier, Logistic Regression, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Recursive Feature Elimination (RFE)

**Model Evaluation Metrics:** Accuracy, Precision, Recall, F1-Score, Confusion Matrix, ROC-AUC

# Introduction

Employee turnover is a critical challenge for organizations,, as losing valuable employees can lead to increased recruitment and training costs, loss of institutional knowledge, and disruption in operations. Predicting which employees are likely to leave allows organizations to proactively address the root causes of turnover and retain top talent.

This case study aims to build and evaluate machine learning models for predicting employee turnover using historical employee data. By analyzing various demographic and employment factors, such as job satisfaction, years at the company, and salary levels, the goal is to develop a model that accurately predicts the likelihood of employees leaving.

In this analysis, four machine learning models were employed:

1. Random Forest Classifier
2. Logistic Regression
3. Support Vector Machine (SVM)
4. K-Nearest Neighbors (KNN)

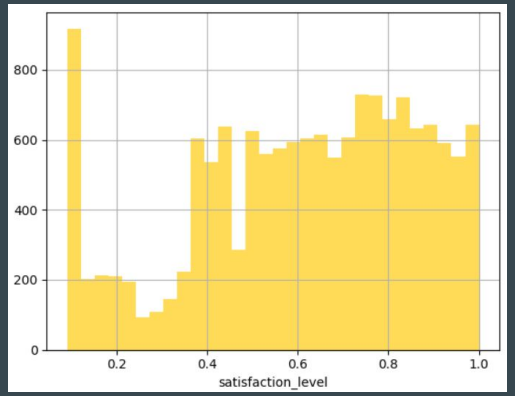
The purpose is to evaluate these models, comparing their accuracy, precision, recall, F1-score, and the ability to handle class imbalance, especially in predicting employees likely to leave (the minority class).

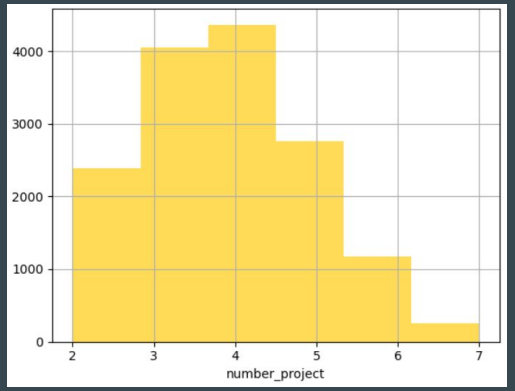
# Dataset Overview

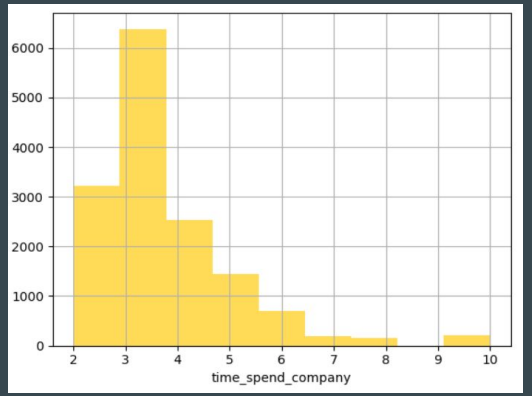
The dataset contains 14,999 entries with the following columns:

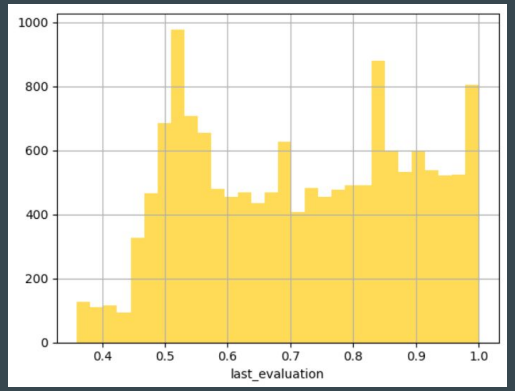
1. **satisfaction_level** : A float value representing the employee's satisfaction level (0–1).
2. **last_evaluation** : A float value indicating the time since the last evaluation (in years).
3. **number_project** : An integer representing the number of projects completed while at work.
4. **average_montly_hours** : A float value indicating the average monthly hours worked.
5. **time_spend_company** : A float value representing the time spent at the company (in years).
6. **work_accident** : An integer indicating whether the employee had a workplace accident (0 or 1).
7. **left**: A categorical value indicating whether the employee left the workplace or not (yes or no).
8. **promotion_last_5years** : An integer indicating whether the employee was promoted in the last five years (0 or 1).
9. **is_smoker** : A categorical value indicating whether the employee is a smoker (yes or no), but this column has many missing values.
10. **department** : A categorical value representing the department the employee works in.
11. **salary**: A categorical value indicating the relative level of salary (low, medium, high).

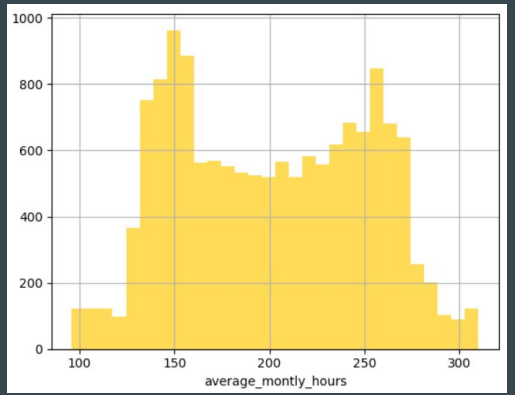
# Data Distribution

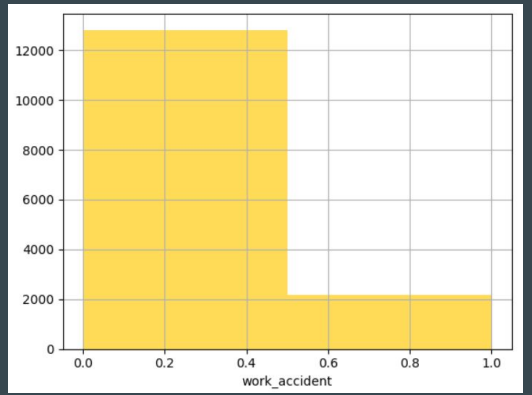


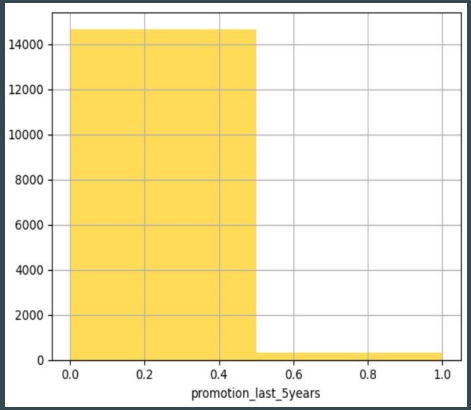



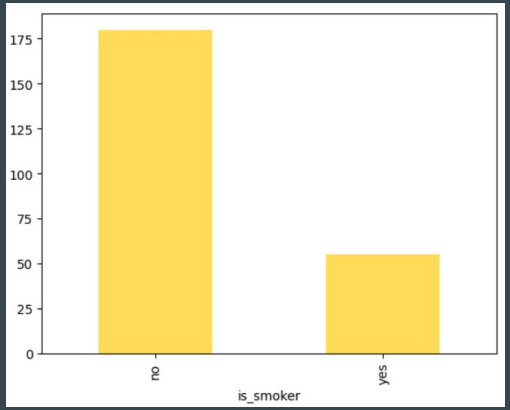



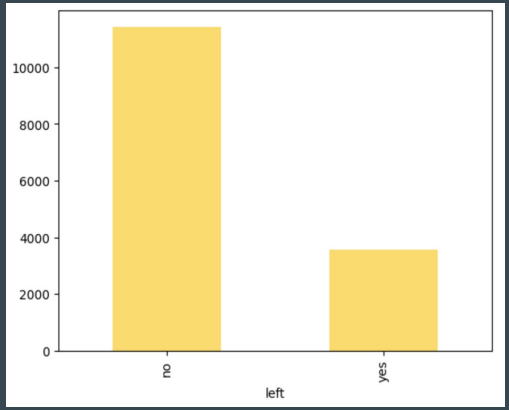



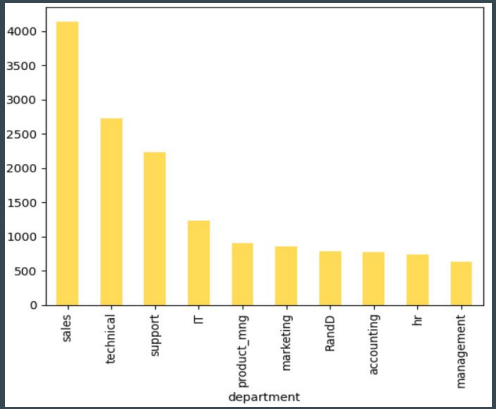



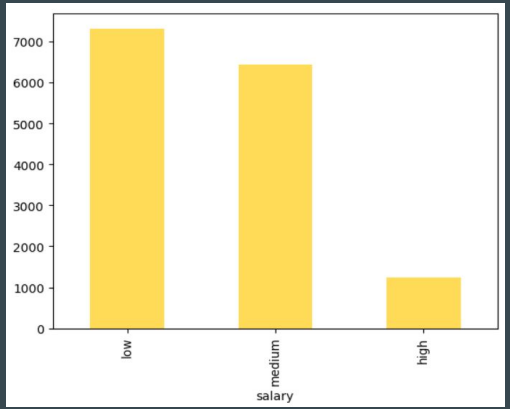
# Data Distribution




| left | count |
|---|---|
| no | 11428 |
| yes | 3571 |

# Data Cleaning

```
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 11 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   satisfaction_level     14999 non-null  float64
 1   last_evaluation        14999 non-null  float64
 2   number_project         14999 non-null  int64
 3   average_montly_hours   14631 non-null  float64
 4   time_spend_company     14848 non-null  float64
 5   work_accident          14999 non-null  int64
 6   left                   14999 non-null  object
 7   promotion_last_5years  14999 non-null  int64
 8   is_smoker              235 non-null    object
 9   department             14999 non-null  object
 10  salary                 14999 non-null  object
dtypes: float64(4), int64(3), object(4)
```

- **Missing Values** :
  - The average_montly_hours column has some missing values.
  - The time_spend_company column has a few missing values.
  - The is_smoker column has significant missing data, 98.43%, which may affect its usability. Thus, this variable is deleted.
- **Data Cleaning** :
  - The average_montly_hours column: replace mean values
  - The time_spend_company column: replace median values
- **Employee Retention** :
  - The left column indicates that most employees did not leave the company (11,428 out of 14,999).
  - Convert categorical variable to binary integer representation

```
df.left = df.left.map({'no': 0, 'yes': 1})
```

# Project objective

Build and compare several machine learning models—including Logistic Regression, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Random Forest—to predict the target variable 'left,' which indicates whether an employee has left the company or not. The goal is to evaluate each model's performance in terms of accuracy, precision, recall, and F1-score, while also assessing their effectiveness in handling class imbalance.

# Comparing Model Performance - Accuracy

| | | |
|---|---|---|
| Random Forest Model | 98.6% & | percent accuracy score per class:<br>left = 0 : 99.74%<br>left = 1 : 94.99% |
| Logistic Regression Model | 79.4% & | Percent accuracy score per class:<br>left = 0 : 92.49%<br>left = 1 : 35.55% |
| Super Vector Machine | 78.3% & | Percent accuracy score per class:<br>left = 0 : 93.96%<br>left = 1 : 26.01% |
| K-Nearest Neighbors | 94.2% & | Percent accuracy score per class:<br>left = 0 : 95.00%<br>left = 1 : 91.71% |

# Summary & Insights

**The Accuracy:**

- Random Forest has the highest accuracy, followed by KNN, with Logistic Regression and SVM performing worse

**Class Imbalance:**

- Logistic Regression and SVM struggle with class imbalance, where the minority class (Class 1) is underrepresented, leading to poorer performance for this class.

**Model Performance:**

- Random Forest and KNN demonstrate balanced, high performance across both classes, indicating that they manage the data more effectively.
- Logistic Regression and SVM show significant discrepancies between the classes, particularly struggling with the minority class.

**Conclusion:** Based on these results, Random Forest appears to be the better choice for further analysis

# Random Forest Model

Best Model Selection

# Hyperparameter Tuning

Use GridSearchCV to find the best hyperparameters for the model

```
Best parameters: {'max_depth': None, 'n_estimators': 300}
```

The result {'max_depth': None, 'n_estimators': 300} indicates that the best hyperparameters for the Random Forest model, based on the grid search, are:

- **max_depth:** None (meaning there is no limit on the depth of the trees in the forest. Trees can grow until they reach pure leaves or the minimum sample split criterion is met.)
- **n_estimators:** 300 (the number of trees in the forest. This value suggests that a higher number of trees has been found to improve model performance in your case.)

# Model Evaluation

The result after refitting model with the best parameters: Refit your Random Forest model using these best parameters.

```
Confusion Matrix:
[[3450   12]
 [  51  987]]
Accuracy: 0.986
left = 0 : 99.65%
left = 1 : 95.09%
```

True Negatives (TN): 3450 (Class 0 correctly predicted)
False Positives (FP): 12 (Class 0 incorrectly predicted as Class 1)
False Negatives (FN): 51 (Class 1 incorrectly predicted as Class 0)
True Positives (TP): 987 (Class 1 correctly predicted)

**Accuracy:**

- **Overall Accuracy:** 98.6%, this means that the model correctly classified 98.6% of all instances in the test set.

**Percent Accuracy Score per Class:**

- **Class 0 (left = 0):** 99.65%
  - The model is very accurate at predicting Class 0.
- **Class 1 (left = 1):** 95.09%
  - The model is also quite accurate at predicting Class 1, though it has slightly lower accuracy compared to Class 0.

# Insights

**Model Performance:**

- The high accuracy for both classes indicates that the Random Forest model is performing excellently overall.
- The model handles Class 0 with very high accuracy and performs well with Class 1 too.

**Class Imbalance Handling:**

- The high accuracy for Class 1 (95.09%) suggests that the model, even with class imbalance (if present), is managing to predict the minority class quite well.

**Error Analysis:**

- The false positives and false negatives are relatively low, which means the model is making few mistakes.

# Model Evaluation - Evaluate Other Metrics

Although accuracy is high, consider evaluating other metrics like Precision, Recall, F1-Score, and ROC-AUC to get a more detailed view of the model's performance, especially for the minority class.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.99 | 1.00 | 0.99 | 3462 |
| 1 | 0.99 | 0.95 | 0.97 | 1038 |
| accuracy | | | 0.99 | 4500 |
| macro avg | 0.99 | 0.97 | 0.98 | 4500 |
| weighted avg | 0.99 | 0.99 | 0.99 | 4500 |

ROC-AUC score: 0.992025586911683

- **ROC-AUC Score:** 0.9915

The ROC-AUC score is very close to 1, indicating excellent performance. It measures the ability of the model to distinguish between the classes, with a higher score representing better performance.

**Accuracy:** 0.99

The model correctly predicted 99% of the instances in the test set.

**Macro Average:**

- **Precision:** 0.99
- **Recall:** 0.97
- **F1-Score:** 0.98
  - Average metrics across classes without considering class imbalance.

**Weighted Average:**

- **Precision:** 0.99
- **Recall:** 0.99
- **F1-Score:** 0.99
  - Average metrics weighted by the number of true instances for each class.

# Model Interpretation

- **Class 0 (left = 0):**
  - **Precision:** 0.99
    - Of all instances predicted as Class 0, 99% were actually Class 0.
  - **Recall:** 1.00
    - Of all actual Class 0 instances, 100% were correctly predicted.
  - **F1-Score:** 0.99
    - The harmonic mean of precision and recall for Class 0.
  - **Support:** 3462
    - The number of actual instances of Class 0 in the test set.
- **Class 1 (left = 1):**
  - **Precision:** 0.99
    - Of all instances predicted as Class 1, 99% were actually Class 1.
  - **Recall:** 0.95
    - Of all actual Class 1 instances, 95% were correctly predicted.
  - **F1-Score:** 0.97
    - The harmonic mean of precision and recall for Class 1.
  - **Support:** 1038
    - The number of actual instances of Class 1 in the test set.

# Insights - Overall Model Performance

**High Precision and Recall:**

- The model shows high precision and recall for both classes, indicating that it is effective at correctly identifying both classes while minimizing false positives and false negatives.

**Class 1 Performance:**

- While Class 1 has slightly lower recall compared to Class 0, the overall performance is still strong. The F1-Score for Class 1 is high, suggesting a good balance between precision and recall.

**Overall Model Performance:**

- The overall accuracy, precision, recall, and F1-Score are excellent, and the ROC-AUC score further confirms that the model is well-calibrated.

# Cross Validation

```
Cross-validated accuracy scores: [0.98238095 0.99190476 0.98571429 0.98666667 0.98380181]
Mean cross-validated accuracy: 0.986093695410513
```

- These scores represent the model's accuracy on each of the five folds used in cross-validation.
- Mean Accuracy: 0.9858. This is the average accuracy of the model across the five folds. It provides a more generalized estimate of the model's performance compared to a single train-test split.

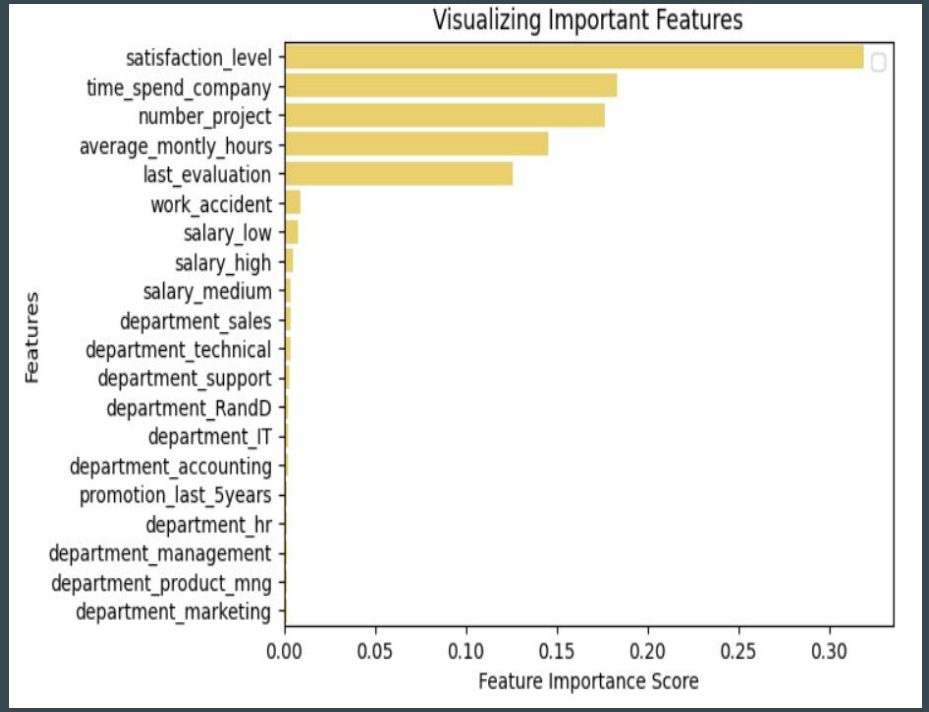
# Insights

**Consistency:** The accuracy scores are consistently high across all folds, ranging from approximately 98.28% to 99.14%. This indicates that the model is robust and performs well across different subsets of data.

**Reliability:** The mean cross-validated accuracy of 98.58% confirms the reliability of the model. It shows that the model generalizes well and is not just performing well on a particular subset of the data.

**Validation:** Cross-validation helps to ensure that the model's performance is not overly optimistic and provides a better estimate of how the model will perform on unseen data.

# Feature Engineering




**Satisfaction Level:** The analysis revealed that the satisfaction level is the most significant feature in predicting employee turnover. This suggests that employees with lower satisfaction are more likely to leave the company. Understanding this feature's impact can guide targeted interventions to improve job satisfaction and retention.

# Recursive Feature Elimination

Implementing **Recursive Feature Elimination (RFE)** with the Random Forest model helps validate the feature importance scores by iteratively selecting the most important features and eliminating the least significant ones.

Random Forest RFE Model Accuracy: 0.99

The Random Forest RFE model achieved an accuracy of 0.99, indicating excellent performance even with fewer features. This confirms that the selected features are highly predictive of employee turnover, and the model retains its high accuracy, further reinforcing the significance of these features.

**Key Insights:**

- **Random Forest's Strength** : Even after reducing the feature set using RFE, the Random Forest model maintains its strong predictive power. This highlights its robustness in capturing the most critical factors influencing employee turnover, such as job satisfaction, time spent at the company and number project..

# Conclusion

Based on the analysis, the **Random Forest Classifier** emerged as the best-performing model, achieving an overall accuracy of 98.6%. It also had balanced performance across both classes, with a high recall of 95.09% for employees who left (Class 1), meaning it was particularly good at identifying employees likely to leave. The model's precision and recall for both classes indicate a robust performance with minimal false positives and false negatives.

In contrast, **Logistic Regression** and **Support Vector Machine** struggled with predicting the minority class (employees who left), showing recall scores of 35.55% and 26.01%, respectively. These models failed to handle the class imbalance effectively, making them less reliable for predicting turnover. **K-Nearest Neighbors** performed reasonably well, with a recall of 91.71% for employees who left, though not as strong as Random Forest.

# Recommendations

**Implement the Random Forest Model:**

- Based on its excellent performance in predicting employee turnover, the Random Forest model is recommended for implementation. It can be deployed in an HR analytics platform to provide **real-time predictions** of employee turnover risk. The model's ability to handle class imbalance makes it ideal for identifying employees likely to leave, even if they form a minority of the workforce.

**Focus on Feature Importance:**

- Use the Random Forest model's feature importance scores to identify the **key drivers** of employee turnover. Insights such as low satisfaction level, salary discrepancies, and years at the company can guide **HR professionals** in focusing on the most critical factors influencing turnover. This helps in addressing the **root causes** proactively.

**Monitor and Update the Model** :

- Continuously monitor the model's performance over time to ensure its predictions remain accurate. Updating the model with **new employee data** (e.g., recent job satisfaction surveys, promotions, or performance reviews) ensures it adapts to **changing employee behavior patterns** and company trends. Regular updates will also improve the model's predictive capabilities as new trends emerge.

# Recommendations

**Develop Targeted Employee Retention Programs** ::

- Based on the model's predictions, the company can implement targeted retention programs to mitigate turnover risk. Potential initiatives include:
  i. **Employee engagement initiatives** such as surveys or focus groups to understand dissatisfaction.
  ii. **Professional development opportunities** for career growth, mentorship, and leadership training.
  iii. **Salary adjustments or bonuses** for high-performing employees identified as at-risk. By focusing resources on employees flagged as likely to leave, the company can maximize retention efforts.

**Data-Driven Decision Making:**

- Incorporate the model's insights into **strategic HR decision-making** processes. For example, the HR department can use predictive analytics to allocate resources more effectively, such as targeting high-risk employees for retention efforts or forecasting potential gaps in critical roles.

# Business Impact

The analysis and predictive model for employee turnover provide significant insights that can directly benefit the organization in several ways:

**Proactive Retention Strategies** : By accurately identifying employees who are at risk of leaving, the company can implement personalized retention strategies. For instance, employees with low job satisfaction or those who haven't been promoted in recent years can be targeted with engagement programs or professional development opportunities. This proactive approach can reduce turnover and the associated costs of hiring and training new employees.

**Cost Savings** : Reducing turnover leads to cost savings in recruitment, onboarding, and training. Additionally, retaining experienced employees preserves institutional knowledge, improving operational efficiency and productivity.

**Resource Allocation** : HR departments can use the model's insights to allocate resources more effectively. For example, departments with high turnover risks may receive more attention in terms of budget for employee satisfaction programs, salary adjustments, or work-life balance initiatives.

# Business Impact

**Strategic Decision-Making** : By using the Random Forest model's feature importance, HR leaders can identify the key drivers of turnover, such as dissatisfaction, salary discrepancies, or length of service. This helps the organization make informed decisions to enhance employee well-being and loyalty.

**Long-Term Organizational Health** : Over time, implementing data-driven strategies to reduce employee turnover strengthens the organizational culture, improves employee morale, and fosters a positive work environment, all of which contribute to long-term success.

.