Machine Learning Operations on Employee Attrition Classification Dataset

Student Name: Divya Sri Kasarla

Student ID: 23070831

Table of Contents

Introduction 4
Background 4
Aim 4
Scope 4
Technique Overview 5
Definition and Explanation 5
Key Concepts 5
SVM 5
Random Forest 5
Use Cases 5
Application of the Technique in Practice 7
Dataset selection 7
Code Walkthrough 7
Data Preprocessing 7
EDA 9
Feature Selection 11
Model Training 11
Model Evaluation 13
Visual Aids 14
Code and Repository 16
Code Explanation 16
Technical Complexity 16
Teaching Tools 16
Clarity of Communication 16
Conclusion 17
References 18

List of Figures

Figure 1 Dataset Information 8
Figure 2 Descriptive Statistics 8
Figure 3 Label encoding 8
Figure 4 Standardization 9
Figure 5 Features Distribution 9
Figure 6 Featured count Value 10
Figure 7 Correlation Matrix 11
Figure 8 Feature Selection 11
Figure 9 SVM Model Training 12
Figure 10 Random forest Training 13
Figure 11 Random Forest Evaluation 13
Figure 12 SVM Model Evaluation 14
Figure 13 Random forest Confusion Matrix 14
Figure 14 SVM Confusion matrix 15

Introduction

Background

Artificial intelligence (AI) is a quickly expanding field which is machine learning algorithms are developed to be learnt from data. These algorithms get better over time with no requirement to be programmatically instructed for every task. Such type of algorithms for which it can categorize supervised learning algorithms is those that simply learn from labelled data in order to make a predictive model. The most useful and broadly used algorithms from this category are Random Forests and Support Vector Machines (SVMs). As a classification technique, SVM decides to sets up a hyperplane in the high-dimensional space to perform the classification. As an ensemble method, Random Forest forms multiple decision trees and employs them to provide better and more stable predictions by making an ensemble of them. They are both better suitable for the high-dimensional spaces and have found application in various applications such as image recognition, medical diagnosis, and text classification, respectively. It is known that the most powerful stuff about SVM is that it is capable of building complex decision boundaries and works with kernel to avoid non linearity and RF is known to be simple, robust, but it can cope with massive datasets that have a lot of features. The purpose of this task is to show how the two techniques can be used for a classification problem, compare their performance, as well as discuss their strong and weak points.

Aim

To enable a reader to better understand how Support Vector Machines (SVMs) and Random Forests perform for classification tasks.

Scope

To apply SVM and Random Forest to the binary classification task of predicting whether an employee is going to leave or stay in a company using attributes like age, job satisfaction, and salary. Both models can be used for multi-class classification, but in this case, they will be used to learn and applied to binary classification problems. This research will not cover mathematically all these algorithms, but it will give an intuitive idea and working code. The data is publicly available, and it will detail how to preprocess the data, train the model, evaluate the model on our data, and optimize the hyperparameters. A comparison of the two models in terms of accuracy and other parameters of performance parameters will also be included.

Technique Overview

Definition and Explanation

They are classification algorithms for supervised learning, i.e., Support Vector Machines (SVM) and Random Forest.

- **Support Vector Machine (SVM):** SVM seeks for best hyperplane for classifying the points of different classes (Mohit Uniyal, 2024). The algorithm goal is to maximize the margin between data from different classes, which improves generalization for novel data. SVM uses the kernel trick to project the data to higher dimensions, where the data would be separated using a linear hyperplane in high dimensions.
- **Random Forest:** Random Forest is an ensemble technique that creates a forest of decision trees (Rukshan Pramoditha, 2020). Each tree is then trained on a random subset of the training data, and the ultimate prediction, which comes from the aggregation of the predictions of all the trees, is the prediction. It decreases the overfitting and strengthens the model.

Key Concepts

SVM

The Boundary line dividing different classes in the feature space is called a Hyperplane (Deepchecks, 2024). Its Margin is the distance between the support vectors (data points as close to the hyperplane) and the hyperplane. The purpose of SVM is to maximize the margin. The most important Dataset points contributing to the decision hyperplane position. SVM or Structural Risk Minimization is a Supervised Machine Learning (SML) Algorithm that deals with a notion of structural risk minimization and uses the Kernel Trick.

Random Forest

- **Decision Trees:** Decision Trees partition data according to the values of the features, constructing a binary tree for prediction (christophm, 2025).
- **Ensemble Learning:** Random Forest makes predictions by combining the forecasts of many decision trees to increase model accuracy.
- **Bagging:** Random Forest uses bootstrapped training data samples to build each tree, thereby decreasing the variance and preventing overfitting.

Use Cases

Various kinds of applications are used for SVM and Random Forest:

- SVM performs particularly well in high-dimensional spaces and has widespread applications in fields including:

    Image recognition (e.g., facial recognition).

    Text classification (e.g., spam filtering).

    Bioinformatics (e.g., gene expression analysis)

- Random Forest is used when dealing with large datasets and complex decision-making tasks. It has the following applications:

    Medical diagnosis (e.g., disease prediction from patient data).

    Financial prediction (e.g., predicting the stock market

    Customer segmentation (e.g., marketing and selling approaches).

Application of the Technique in Practice

Dataset selection

This research uses a data set containing employee information such as age, sex, years at the company, job role, monthly salary, and a few additional features. The response variable Attrition, indicating the binary class label if the employee has attrited or not, has been utilized for this dataset. This data set is well-suited for classification issues as it contains numerical and categorical features so that demonstrate the usage of Support Vector Machines (SVM) as well as Random Forest easily. The dataset contains 59,598 records, each instance corresponding to an employee, and a total of 24 features. There are both numerical and categorical data among the features. Preprocessing steps are required for data preparation to train the models, such as encoding categorical features, handling missing values, and feature scaling.

Code Walkthrough

The next section provides the step-by-step procedure for applying SVM and Random Forest models to the provided dataset. The key steps include data preprocessing, feature selection, training the models, and model evaluation.

Data Preprocessing

Data pre-processing is the crucial first step for machine learning. Initially handled the treatment of the missing data, where scanned the data for any missing values and applied appropriate imputation techniques accordingly (Ahmad et al., 2024). If any of the columns had a significant number of missing values, it was imputed or excluded from the analysis. Numerical attributes with missing values were imputed using the mean or the median, while the categorical variables were handled with the mode or excluded if the variables were severely incomplete.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 59598 entries, 0 to 59597
Data columns (total 24 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Employee ID                59598 non-null  int64
 1   Age                        59598 non-null  int64
 2   Gender                     59598 non-null  object
 3   Years at Company           59598 non-null  int64
 4   Job Role                   59598 non-null  object
 5   Monthly Income             59598 non-null  int64
 6   Work-Life Balance          59598 non-null  object
 7   Job Satisfaction           59598 non-null  object
 8   Performance Rating         59598 non-null  object
 9   Number of Promotions       59598 non-null  int64
 10  Overtime                   59598 non-null  object
 11  Distance from Home         59598 non-null  int64
 12  Education Level            59598 non-null  object
 13  Marital Status             59598 non-null  object
 14  Number of Dependents       59598 non-null  int64
 15  Job Level                  59598 non-null  object
 16  Company Size               59598 non-null  object
 17  Company Tenure             59598 non-null  int64
 18  Remote Work                59598 non-null  object
 19  Leadership Opportunities   59598 non-null  object
 20  Innovation Opportunities   59598 non-null  object
 21  Company Reputation         59598 non-null  object
 22  Employee Recognition       59598 non-null  object
 23  Attrition                  59598 non-null  object
dtypes: int64(8), object(16)
memory usage: 10.9+ MB
```

Figure 1 Dataset Information

	Employee ID	Age	Years at Company	Monthly Income	Number of Promotions	Distance from Home	Number of Dependents	Company Tenure
count	59598.000000	59598.000000	59598.000000	59598.000000	59598.000000	59598.000000	59598.000000	59598.000000
mean	37227.118729	38.565875	15.753901	7302.397983	0.832578	50.007651	1.648075	55.758415
std	21519.150028	12.079673	11.245981	2151.457423	0.994991	28.466459	1.555689	25.411090
min	1.000000	18.000000	1.000000	1316.000000	0.000000	1.000000	0.000000	2.000000
25%	18580.250000	28.000000	7.000000	5658.000000	0.000000	25.000000	0.000000	36.000000
50%	37209.500000	39.000000	13.000000	7354.000000	1.000000	50.000000	1.000000	56.000000
75%	55876.750000	49.000000	23.000000	8880.000000	2.000000	75.000000	3.000000	76.000000
max	74498.000000	59.000000	51.000000	16149.000000	4.000000	99.000000	6.000000	128.000000

Figure 2 Descriptive Statistics

The categorical variables were encoded using Label Encoding for binary variables like "Gender" and "Attrition," and One-Hot Encoding for multi-class variables like "Job Role." This encoding transforms categorical values into a numerical format processable by the machine learning algorithms.

```python
# Label encode binary categorical columns (Gender, Attrition, etc.)
binary_columns = ['Gender', 'Attrition', 'Overtime', 'Remote Work', 'Leadership Opportunities', 'Innovation Opportunities']
label_encoder = LabelEncoder()

for column in binary_columns:
    df[column] = label_encoder.fit_transform(df[column])
```

Figure 3 Label encoding

This project also performed feature scaling, which is particularly important for scale-sensitive feature models like SVM. For this StandardScaler to scale the numeric featuresused (e.g., "Age",

"Monthly Income"). Random Forest, while not scale-sensitive specifically, benefits from scaling in the sense of consistency between models, so it scaled the entire dataset.

```
# Feature Scaling (SVM is sensitive to feature magnitudes)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figure 4 Standardization

EDA

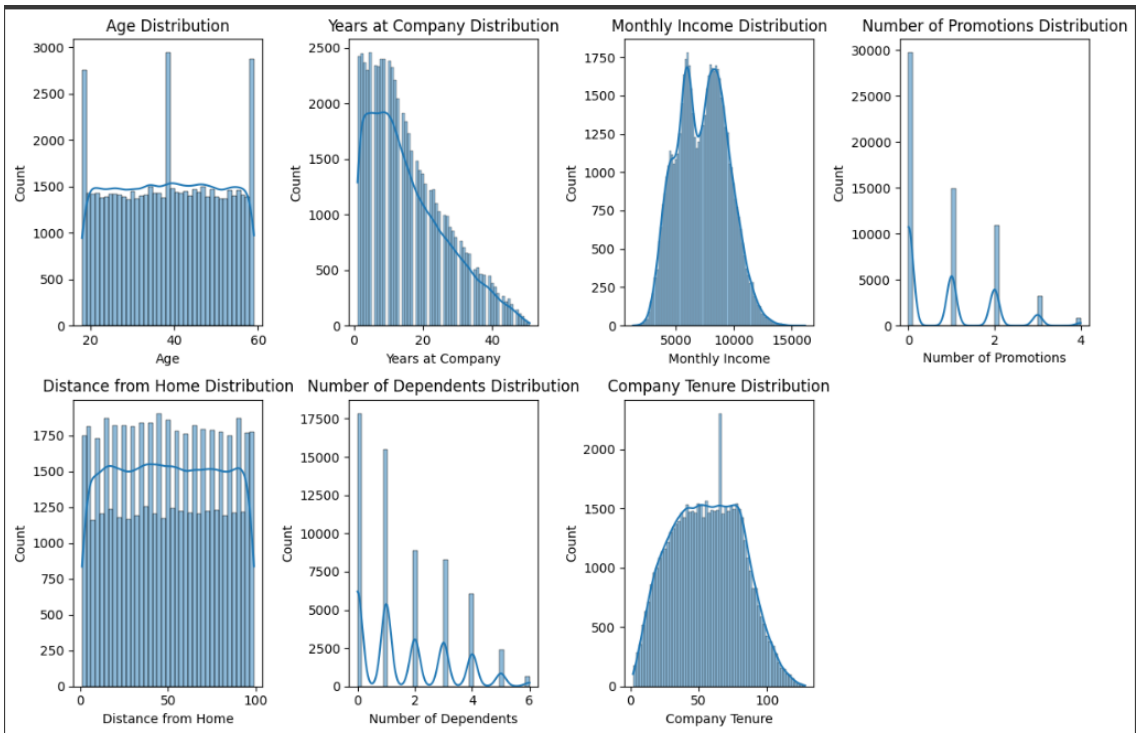

Figure 5 Features Distribution

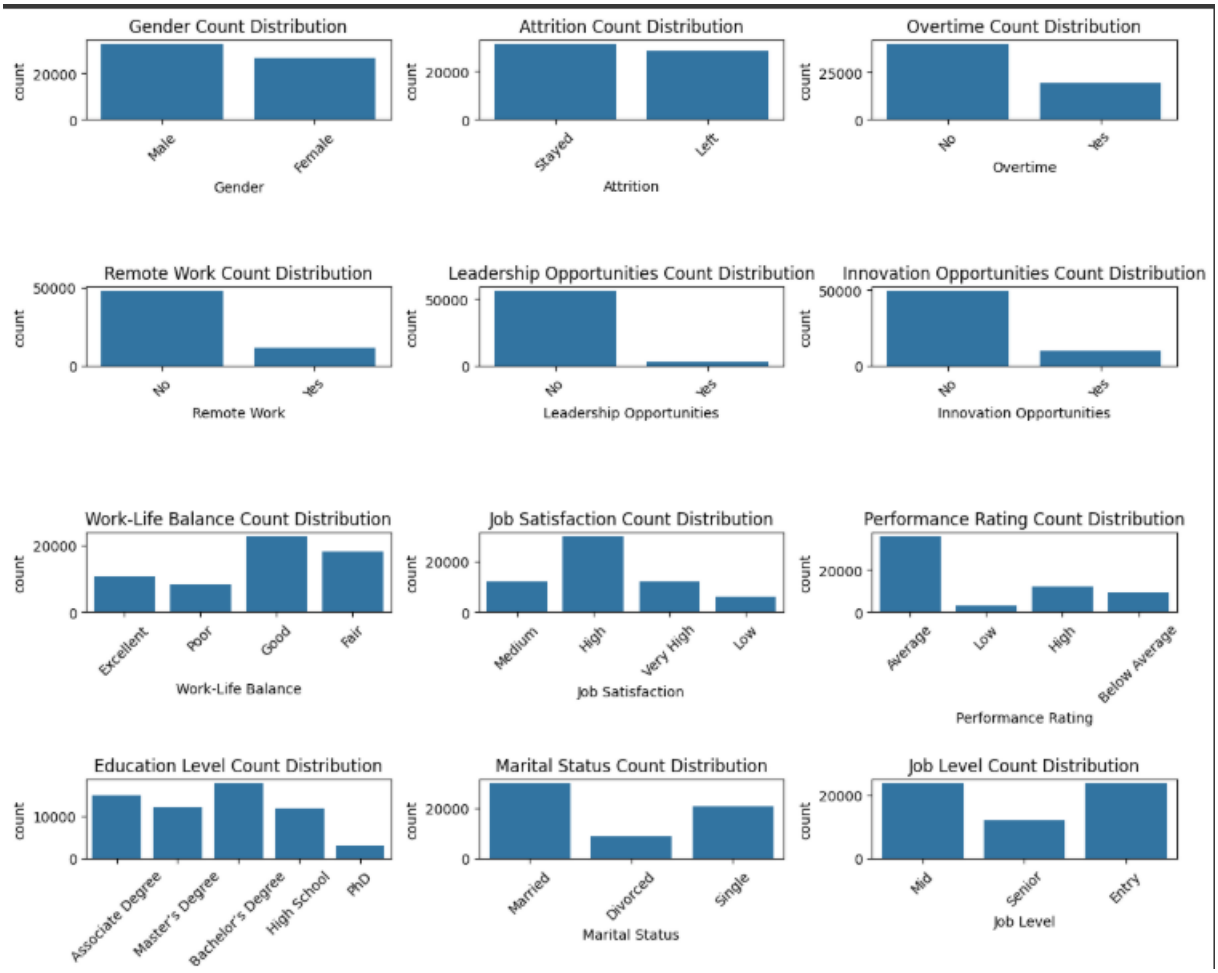

Figure 6 Featured count Value

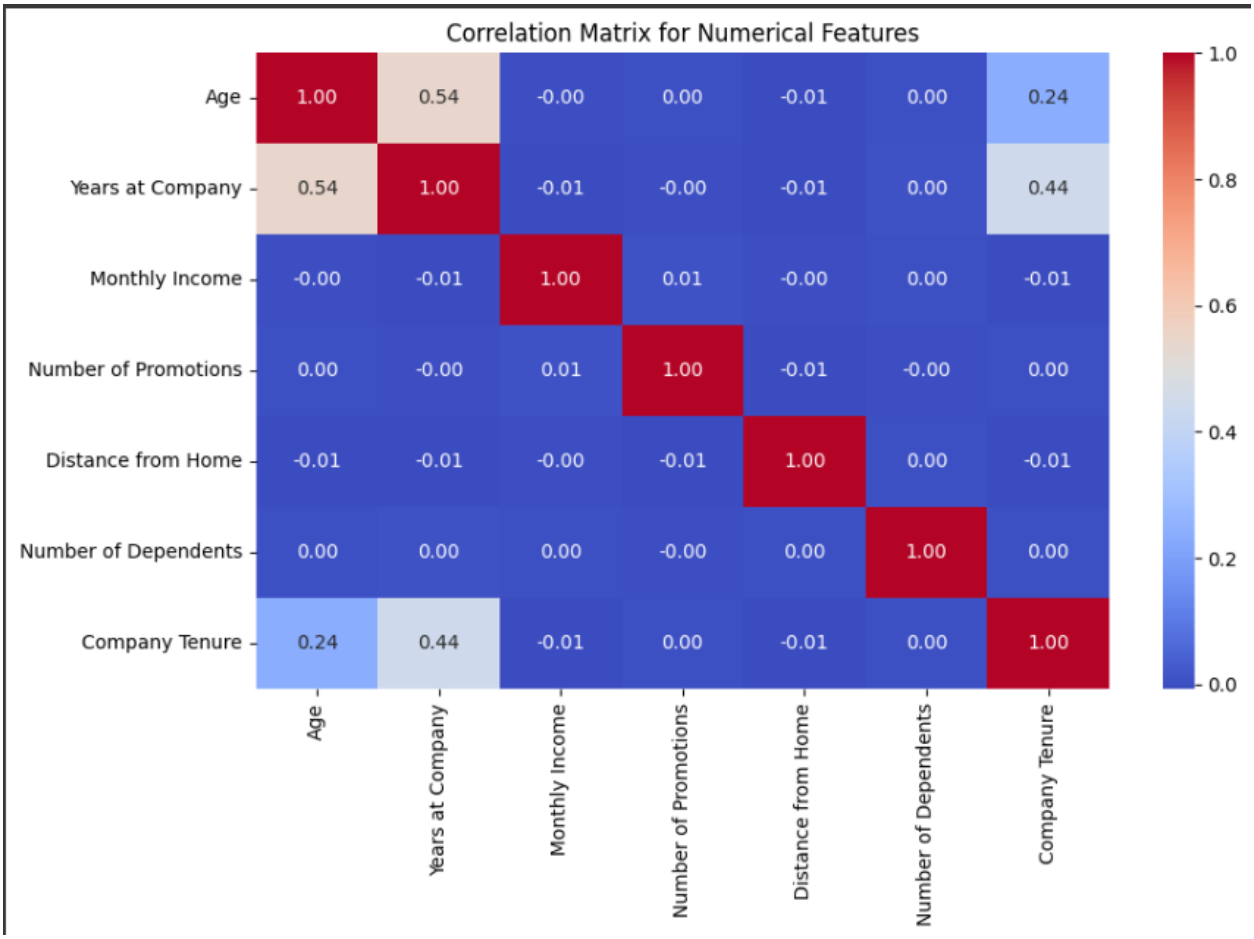

Figure 7 Correlation Matrix

Feature Selection

Feature selection is one of the most important steps in improving model performance by choosing the most informative features. First conducted a correlation analysis to identify highly correlated features. The feature importance functions in Random Forest to identify the features that contributed most to the model predictions. The most impactful features were chosen, and the less informative features were dropped based on the importance scores to make the training process more efficient.

```
X = df.drop('Attrition', axis=1)  # Features
y = df['Attrition']  # Target variable
```

Figure 8 Feature Selection

Model Training

Having preprocessed the data, this proceeds to model training. Both the SVM and Random Forest models were trained with the preprocessed data. When training the SVM model, it utilized the

SVC class in scikit-learn. For the SVM, some of the crucial parameters were the C (regularization parameter), the kernel (RBF) kernel for non-linear classification), and gamma, which controls the effect of single data points. Such hyperparameters were tuned to obtain the best from the model.

```
# Feature Scaling (SVM is sensitive to feature magnitudes)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Initialize and train the SVM model
svm_model = SVC(kernel='rbf', random_state=42)  # 'rbf' is the default kernel (Gaussian SVM)
svm_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred_svm = svm_model.predict(X_test)

# Evaluate the SVM model
svm_accuracy = accuracy_score(y_test, y_pred_svm)
svm_report = classification_report(y_test, y_pred_svm)

# Print the results
print(f"SVM Accuracy: {svm_accuracy:.4f}")
print("\nSVM Classification Report:\n", svm_report)
```

Figure 9 SVM Model Training

For Random Forest, uses RandomForestClassifier module from scikit-learn. Important hyperparameters for this model included n_estimators (the number of trees in the forest), max_features (the number of features to consider when deciding on each node), and max_depth (the depth of the trees). Both models were then trained on an 80-20 split for the training and test data, respectively, for proper evaluation.

```python
# Initialize and train the Random Forest model
rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred_rf = rf_model.predict(X_test)

# Evaluate the model
rf_accuracy = accuracy_score(y_test, y_pred_rf)
rf_report = classification_report(y_test, y_pred_rf)

# Print the results
print(f"Random Forest Accuracy: {rf_accuracy:.4f}")
print("\nRandom Forest Classification Report:\n", rf_report)
```

Figure 10 Random forest Training

Model Evaluation

The program compared the performance of the models with several metrics. Accuracy was employed to compute the proportion of the predictions that were true, and precision, recall, and F1-score were also employed to calculate the performance of the models for the two classes. On Random Forest, it achieved an accuracy of 0.7415, with the precision for class 0 (employees who stay) being 0.73 and for class 1 (employees who leave) being 0.76. The recall for Random Forest was 0.73 for class 0 and 0.75 for class 1, while the F1-score for class 0 was 0.73 and for class 1 was 0.75.

```
Random Forest Accuracy: 0.7415

Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.73      0.73      0.73      5667
           1       0.76      0.75      0.75      6253

    accuracy                           0.74     11920
   macro avg       0.74      0.74      0.74     11920
weighted avg       0.74      0.74      0.74     11920
```

Figure 11 Random Forest Evaluation

For SVM, the accuracy slightly improved to 0.7456, with the precision of 0.75 for class 1 and 0.73 for class 0. SVM recall stood at 0.75 for class 1 and 0.74 for class 0, while the F1-score for SVM

stood at 0.75 for class 1 and 0.74 for class 0. Both models were similar, with SVM having the slight advantage in the context of accuracy.

```
SVM Accuracy: 0.7456

SVM Classification Report:
              precision    recall  f1-score   support

           0       0.73      0.74      0.73      5667
           1       0.76      0.75      0.76      6253

    accuracy                           0.75     11920
   macro avg       0.74      0.75      0.75     11920
weighted avg       0.75      0.75      0.75     11920
```

Figure 12 SVM Model Evaluation

Visual Aids

To assist with the interpretation of the performance of the model, several visualizations were generated. Confusion matrices were generated for both models to allow visualization of the ratio of correct and incorrect predictions. Feature importance scores were also plotted for Random Forest, indicating the most influential features to the model's predictions.

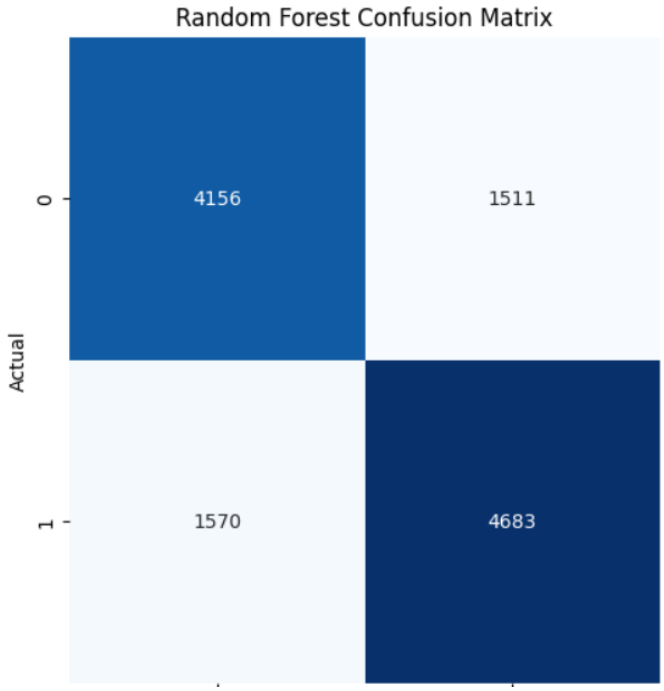

Figure 13 Random forest Confusion Matrix

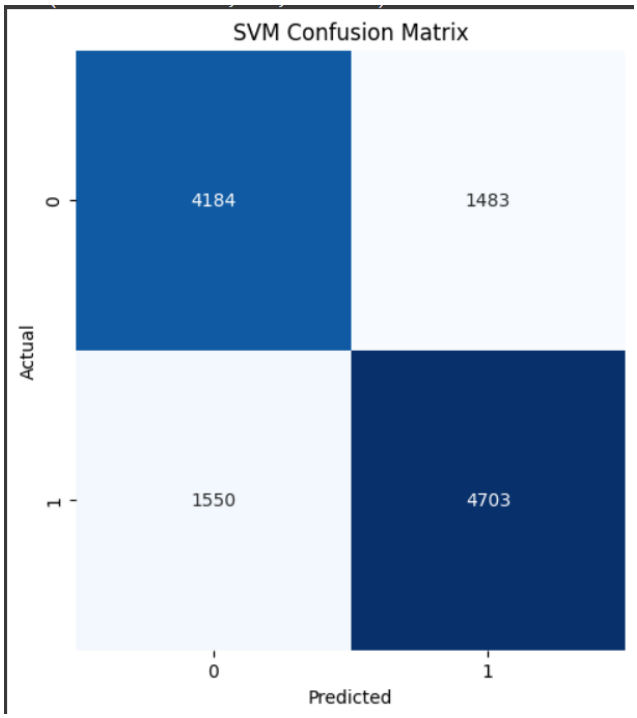

Figure 14 SVM Confusion matrix

Code and Repository

Code Explanation

The code is written with the help of Python and the popular libraries such as pandas, numpy, matplotlib, seaborn, and scikit-learn. The implementation starts with data loading and pre-processing, and then encodes the categorical variables and scales the features. Finally, the code also shows the steps to take to train both SVM and Random Forest models with the hyperparameter tuning process to achieve optimal performance. When it trains the models, it uses the performance measures to test the models and plot the results.

Technical Complexity

The technical difficulty of this study lies in the application of SVM and Random Forest to an actual classification problem. While the models themselves are easy to implement and straightforward to utilize in scikit-learn, the process of optimizing the models' hyperparameters for optimal performance proves difficult. SVM, in particular, has the potential to be mathematically complicated based on its kernel-based transformation of nonlinear data. Scikit-learn does provide, though, a user-friendly interface that hides much of the complexity. Random Forest, while simpler in theory, requires careful tuning of parameters like the number of trees and tree depth to not overfit and achieve good performance.

Teaching Tools

Throughout this research, multiple tools were used to facilitate learning:

- **Code Snippets:** The code has been segmented into parts with proper comments to explain each step.
- **Visualizations:** Plots such as confusion matrices, feature importance plots, and ROC curves were used to visualize model performance.

Clarity of Communication

The study is laid out so that beginners could understand it and have valuable insights from experienced practitioners, too. Simple explanations were given in terms of concepts, and each code block was accompanied by a clear explanation. With the use of visual aids, the study became even clearer, which made it easy for readers to follow along and see what sort of behaviour the model would exhibit.

Conclusion

It uses a real-world employee dataset and shows how to apply Support Vector Machines (SVM) and a Random Forest model in a binary classification problem. This is used to look into major steps like the data preprocessing, feature choice, training of a model, and performance evaluation. The performance metrics for the models were compared based on accuracy, precision, recall, and F1-score. It is being able to use ML models such as SVM and Random Forest and be able to implement and evaluate them would greatly aid in making decisions regarding retention of employees, predicting churn for customers, etc. These models are very easy to understand, and they help the practitioner to understand how to tackle the problems inside a variety of domains. Advancements can be made to include multi-classification problems or some more advanced methods like Gradient Boosting or Deep Learning to improve the performance even further. Further, other feature engineering techniques and domain-specific knowledge could expand the accuracy and the interpretability of a model as well.

GitHub Repository:

https://github.com/divyasri0618/ml-23070831

References

Ahmad, A.F., Sayeed, M.S., Alshammari, K. and Ahmed, I. (2024), 'Impact of Missing Values in Machine Learning: A Comprehensive Analysis', *arXiv* https://doi.org/10.48550/arxiv.2410.08295.

christophm (2025), ' Decision Tree Interpretable Machine Learning', https://christophm.github.io/interpretable-ml-book/tree.html

Deepchecks. (2024), 'What is Hyperplane? Role in Machine Learning', https://www.deepchecks.com/glossary/hyperplane/

Mohit Uniyal (2024), 'Support Vector Machine (SVM) Algorithm', Applied AI Blog. https://www.appliedaicourse.com/blog/support-vector-machine-algorithm/#:~:text=In%20classification%20tasks%2C%20SVM%20seeks,class%2C%20known%20as%20support%20vectors.

Rukshan Pramoditha (2020), 'Random forests - An ensemble of decision trees', Towards Data Science. https://towardsdatascience.com/random-forests-an-ensemble-of-decision-trees-37a003084c6c/