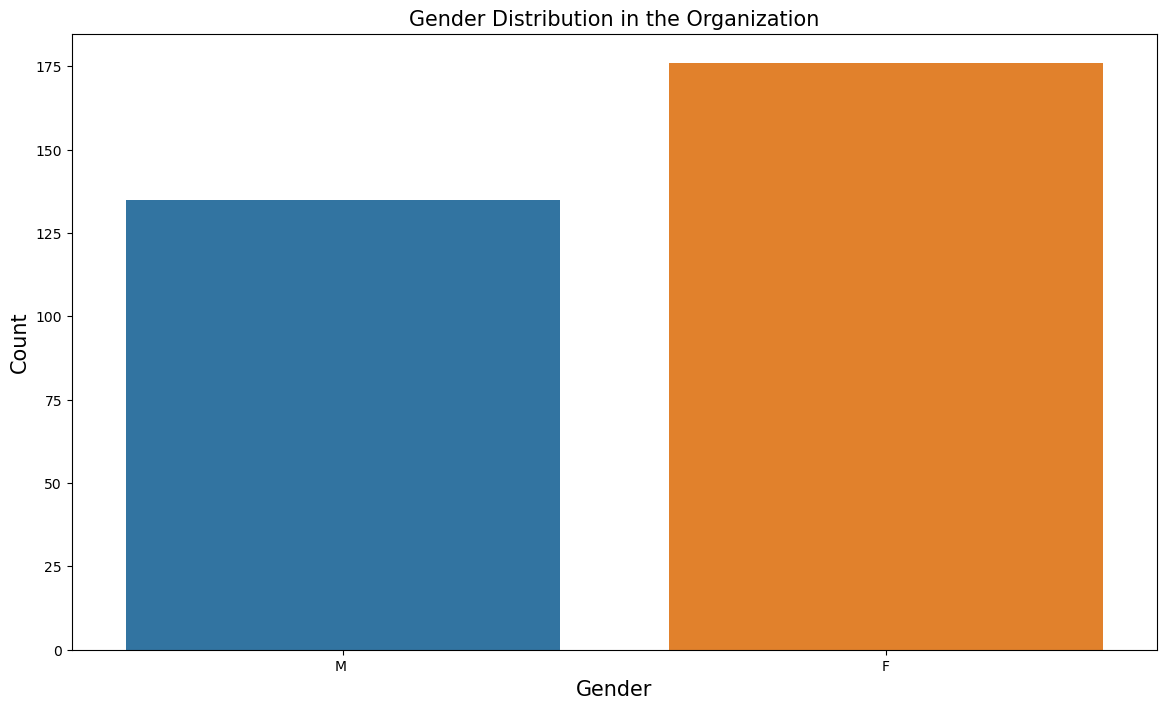
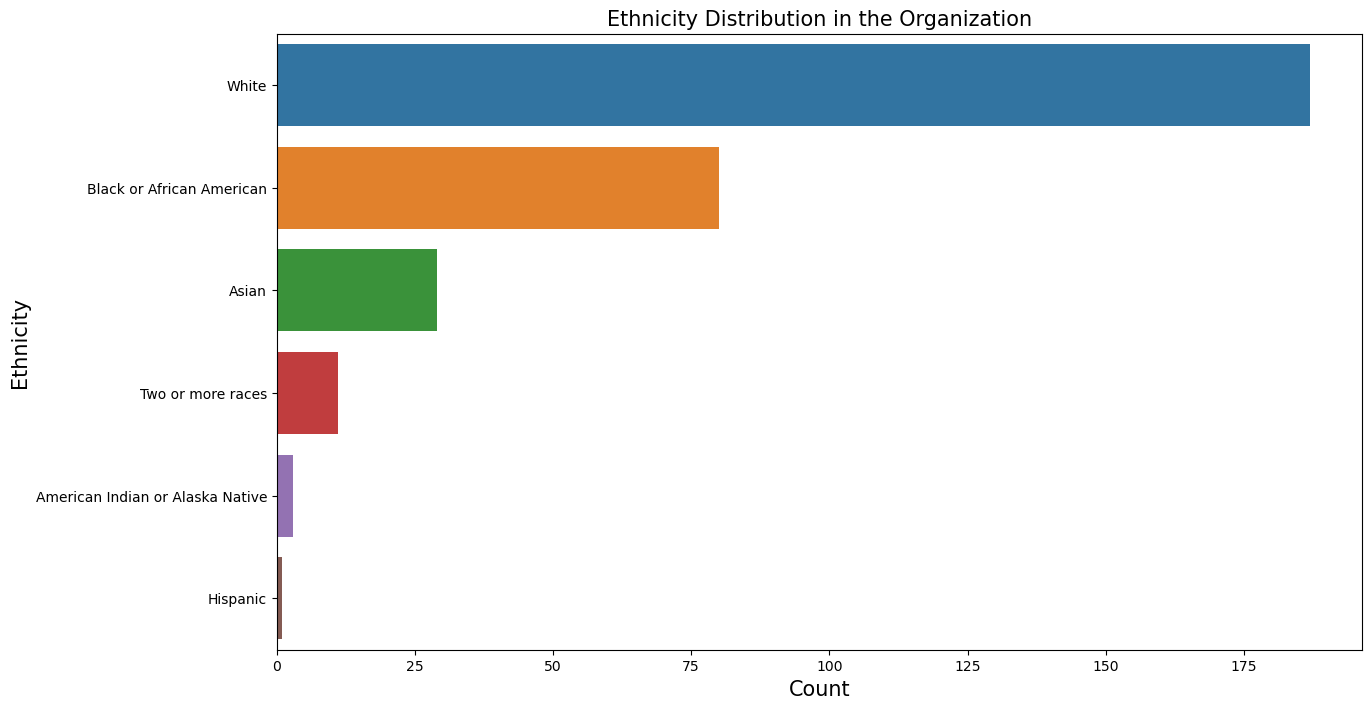
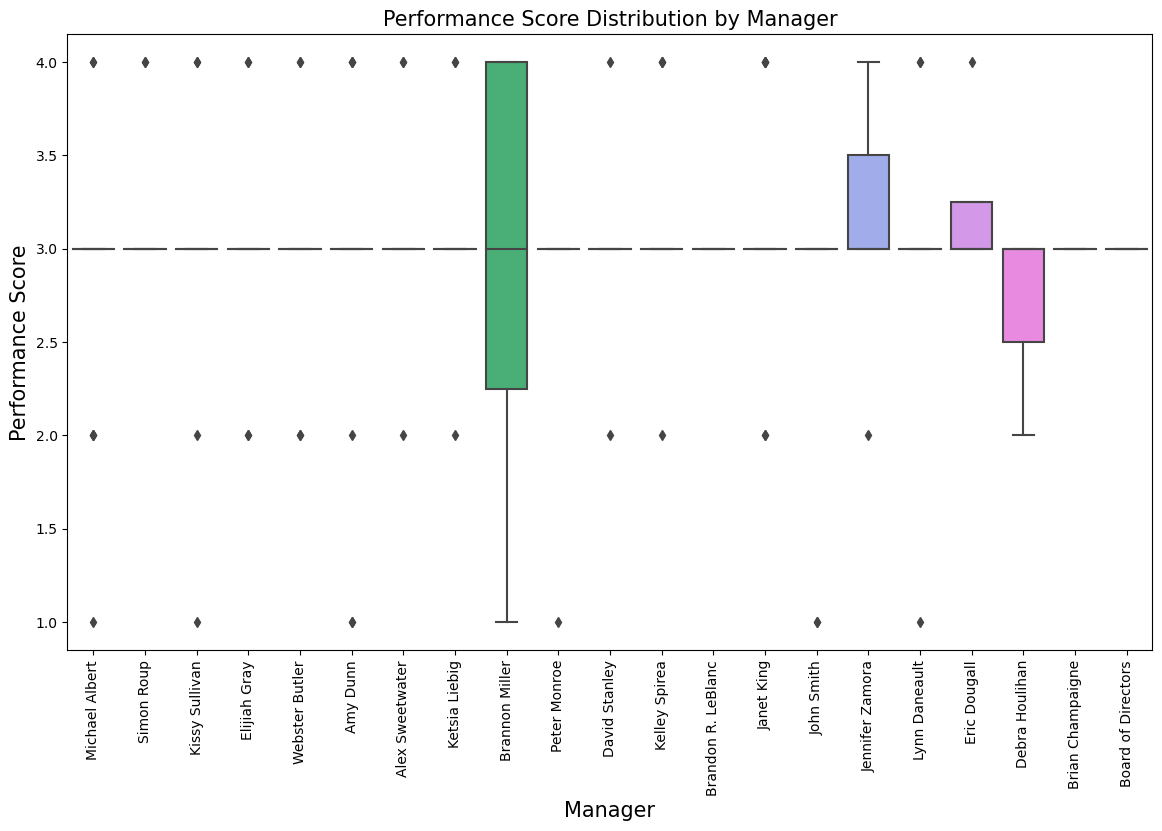
**Section A (Data Exploration and Pre-processing):**

### **Explore the data using tables, visualizations, and other relevant methods**

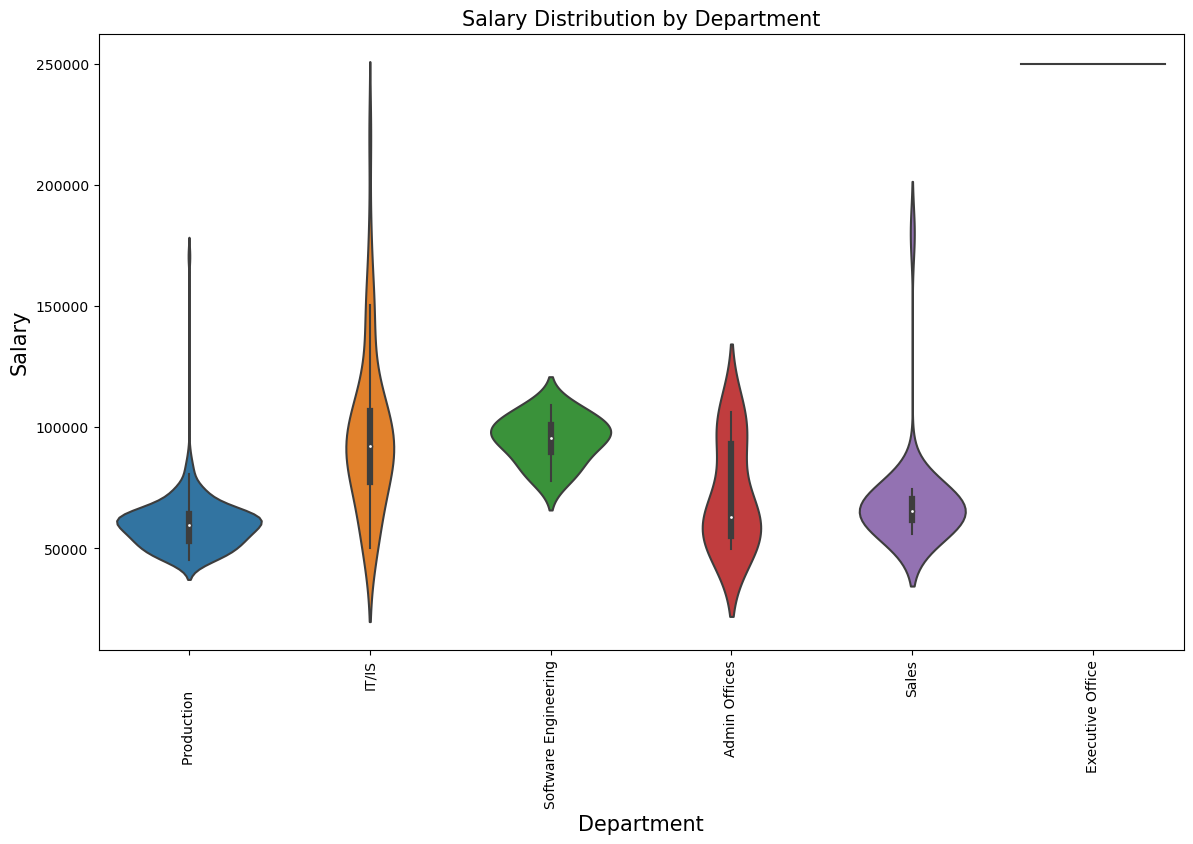




These plots will provide insights into the gender and ethnicity diversity within the organization.



We can investigate whether there is any relationship between the employee's manager and their performance score. To visualize this, we'll use a box plot to compare the performance scores across different managers. This box plot will help us identify any variations in performance scores based on the employee's manager.



To identify areas of pay inequity within the company, we can compare the salaries across different departments. A violin plot can effectively showcase the distribution of salaries in each department.  
This visualization will highlight any variations in salary distribution across different departments, indicating potential areas of pay inequity.



Visualize the distribution of salaries across different departments, considering the performance scores of employees as well. We'll use a box plot with color encoding to represent the performance scores.

In this plot, each box represents the salary distribution for a specific department, and the color of the boxes represents different performance scores. This allows us to simultaneously explore the salary distributions and performance scores across departments.

1) Overall Diversity Profile of the Organization:

* The gender distribution graph provides insights into the gender diversity within the organization.
* The ethnicity distribution graph offers an understanding of the ethnic diversity within the organization.
* These graphs can help us assess the overall diversity profile of the organization.

2) Relationship between Who a Person Works for and Their Performance Score:

* The box plot graph comparing performance scores across different managers provides insights into the potential relationship between an employee's manager and their performance score.
* This graph allows us to explore whether there is any correlation between the employee's manager and their performance score.

3) Areas of the Company Where Pay Is Not Equitable:

* The violin plot graph displaying the salary distribution by department can help identify potential areas where pay may not be equitable. This graph enables us to visually assess variations in salary distribution across different departments, indicating areas where pay inequity might exist.

### **Apply different methods of pre-processing:**

1. Handling Missing Values: Check if there are any missing values in the dataset and handle them accordingly. Missing values can be imputed using appropriate techniques or the corresponding rows can be removed if they contain a significant number of missing values.
2. Removing Irrelevant Features: Identify and remove any features that do not contribute significantly to the analysis or modeling tasks. For example, features like Employee\_Name, EmpID, Zip, and ManagerName may not provide valuable insights or introduce unnecessary complexity.
3. Handling Categorical Variables: Convert categorical variables into numerical representations to make them compatible with machine learning algorithms. This can be done through one-hot encoding or label encoding, depending on the nature of the categorical variables and the requirements of the models.
4. Converted the variables like DOB, DateofHire, DateofTermination and LastPerformanceReview\_Date to pandas date time object.
5. Feature Scaling: Normalize or standardize numerical features to ensure they have similar scales. This step can be crucial for certain models that are sensitive to the scale of features, such as distance-based algorithms or neural networks.
6. Feature Engineering: Create new features or transform existing ones to extract more meaningful information from the data. Like we created some new features AgeAtHire, EmploymentDuration, TimeSinceLastReview and TimeUntilTermination.

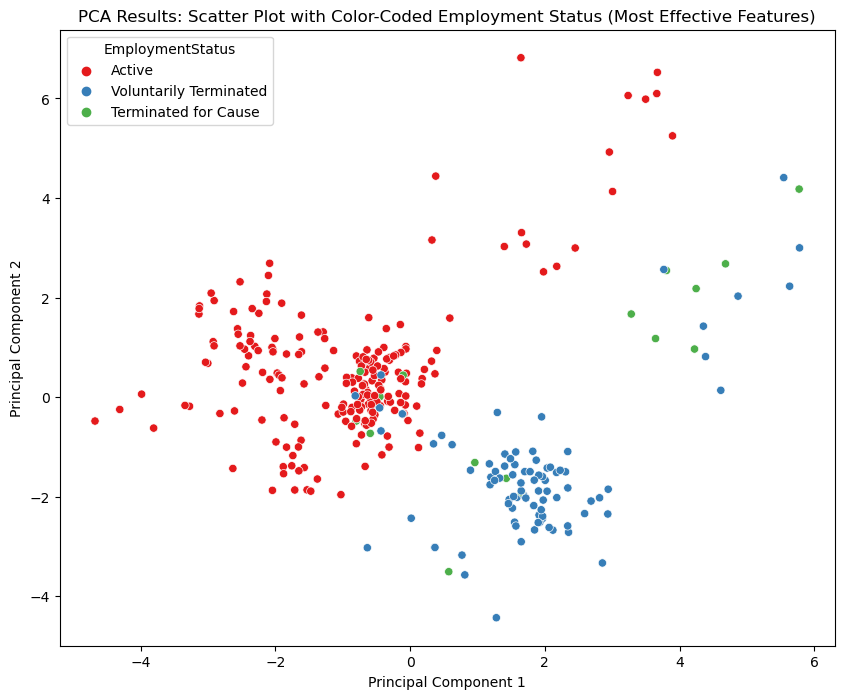
These pre-processing steps help ensure that the data is in a suitable format for the subsequent modeling tasks and that any biases or inconsistencies are addressed appropriately. The choice of pre-processing techniques depends on the specific characteristics of the data and the requirements of the analysis or modeling objectives.

**Section B (Dimensionality Reduction*):***

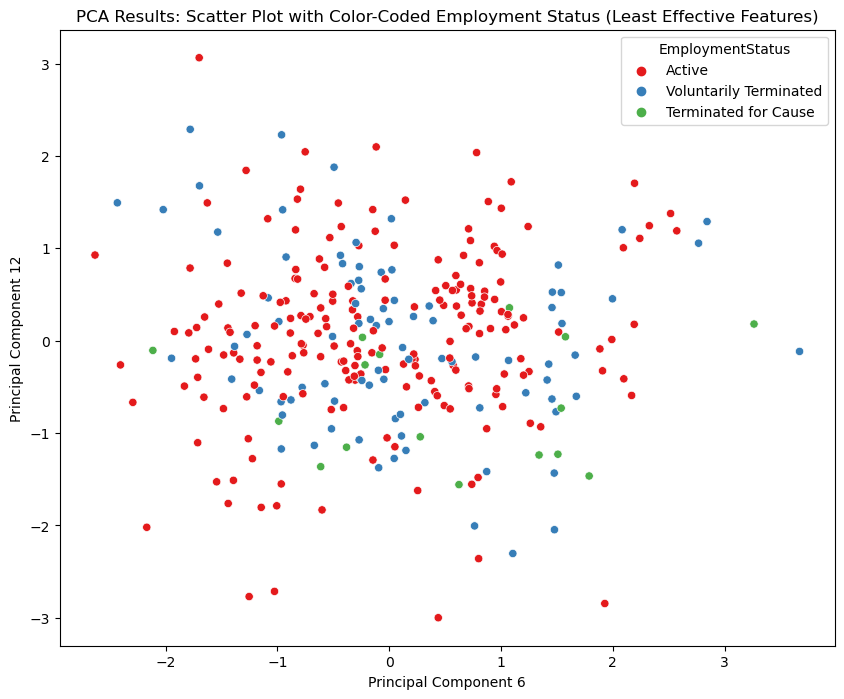
Section B (Dimensionality Reduction) focuses on using dimensionality reduction techniques to visualize and analyze the data. The provided code demonstrates the use of principal components analysis (PCA) to reduce the dimensionality of the data and visualize it in a scatter plot.

The scatter plot displays individual employees as data points, where the x and y axes represent the principal components PC1 and PC2, respectively. The color of each point indicates the employment status of the corresponding employee. By plotting the data in this manner, it becomes possible to observe how the most effective features (PC1 and PC2) contribute to the differentiation of employees based on their employment status.

In the scatter plot where the most effective features (PC1 and PC2) are used, we observe distinct and well-separated clusters for different employment statuses. This suggests that PC1 and PC2 play a crucial role in separating employees based on their employment status. These features contribute significantly to the variance in the data and provide valuable information for classification.



On the other hand, when the least effective features (PC6 and PC12) are used, the resulting clusters on the scatter plot overlap more. This indicates that PC6 and PC12 have less impact on differentiating employees based on their employment status. These features contribute less to the overall variance in the data and may not provide sufficient discriminatory information.

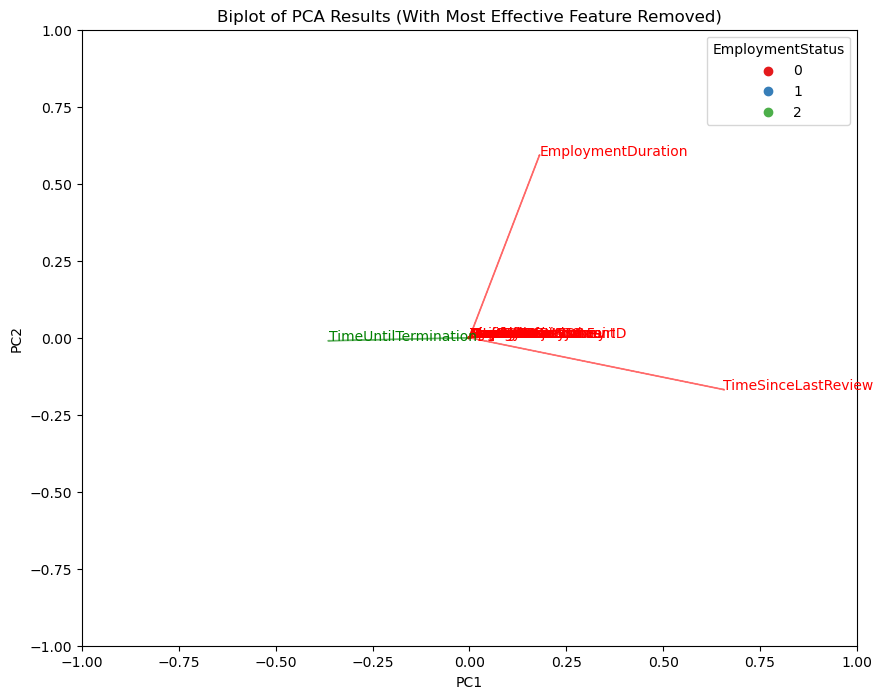


When PC1 and PC2 are effective features for separation, the resulting clusters on the scatter plot will be more distinct for different employment statuses. This indicates that the identified features strongly influence the separation of employees based on their employment status.

Conversely, the code also demonstrates a scenario where the least effective features (PC6 and PC12) are used as the x and y axes. In this case, the resulting clusters on the scatter plot may overlap more, making it harder to distinguish between different employment statuses. This occurs because these features have less impact on the separation of the data.

The biplot is another visualization included in the code. It provides insights into the relationships between the data points and variables. Here are some aspects to consider when interpreting the biplot:

1. Proximity of data points: Data points that are close to each other on the biplot have similar feature values, while points that are far apart are dissimilar. By observing the clusters of different employment status groups, you can identify patterns in their proximity.
2. Angle and direction of vectors: Vectors in the biplot represent the variables or features. The angle between two vectors indicates the correlation between the corresponding features. Vectors pointing in the same or opposite directions indicate positive or negative correlations, respectively.
3. Variables' contribution: The length of the vectors represents the contribution of the variables to the principal components. Longer vectors indicate a higher importance in explaining the variance of the data. By examining the lengths and orientations of the vectors, you can identify the variables that have the most significant influence on PC1 and PC2.



By considering these aspects and analyzing the biplot, you can gain insights into the relationships between data points and variables, identify patterns, and understand the variables' contributions to the separation of different employment status groups.

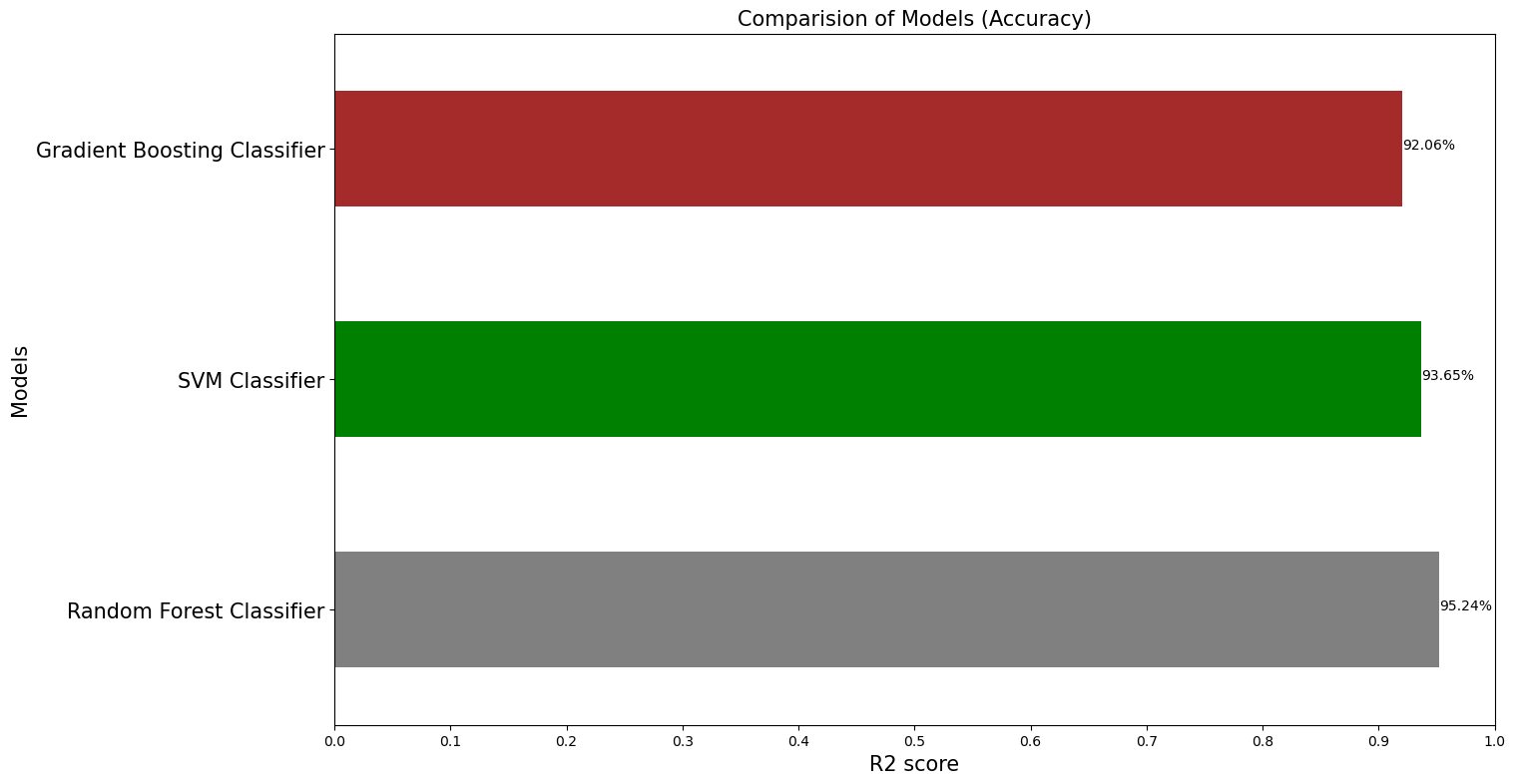
Lastly, the code calculates the Euclidean distances of each data point from the centroid in the PCA space. The threshold for identifying outliers is defined as the 95th percentile of the distances, which can be adjusted according to specific requirements. The code then identifies the data points that exceed the threshold as outliers and prints the list of outliers.

Analyzing the outliers can provide insights into potential anomalies or peculiarities in the dataset. By observing if the outliers share any common characteristics or patterns, you can gain a deeper understanding of these unusual data points.

In summary, Section B employs dimensionality reduction techniques and visualization methods to gain insights into the data, identify patterns, and analyze outliers. These approaches facilitate the exploration and understanding of the relationships between variables and data points in the context of employment status.

**Section C (Classification):**

After analyzing the classification results for the Random Forest Classifier, Support Vector Machine (SVM), and Gradient Boosting Classifier, as well as the feature importances, we can draw the following findings:



**Random Forest Classifier:**

The Random Forest Classifier achieved an overall accuracy of 95.24% for all employment status classes. The sensitivity (true positive rate) and specificity (true negative rate) varied for each class. Class 0 (active employees) achieved perfect accuracy and specificity, indicating that the model performed exceptionally well in predicting this class. Class 1 (terminated employees) had a lower sensitivity, suggesting that the model struggled to identify all terminated employees correctly. Class 2 (employees who left) achieved a lower specificity, indicating some false positives.

Regarding feature importances, the top three influential features were "TimeUntilTermination" (28.21%), "TimeSinceLastReview" (27.89%), and "Termd" (25.18%). These features had the most significant impact on the model's predictions, highlighting their importance in determining employment status. Other features, such as "EngagementSurvey," "EmploymentDuration," and "ManagerID," also contributed but to a lesser extent.

**Support Vector Machine:**

The Support Vector Machine achieved an accuracy of 93.65% for all employment status classes. The sensitivity and specificity varied across the classes. Class 0 achieved perfect accuracy and specificity, indicating excellent performance in predicting this class. Class 1 had a lower sensitivity, suggesting difficulty in identifying terminated employees accurately. Class 2 achieved a relatively high sensitivity but a lower specificity.

**Gradient Boosting Classifier:**

The Gradient Boosting Classifier achieved an accuracy of 92.06% for all employment status classes. Similar to the previous classifiers, the sensitivity and specificity varied for each class. Class 0 achieved perfect accuracy and specificity. Class 1 had a lower sensitivity, indicating some difficulty in correctly identifying terminated employees. Class 2 achieved a relatively high sensitivity but a lower specificity.

Regarding feature importances, the top two influential features were "TimeSinceLastReview" (45.41%) and "TimeUntilTermination" (41.45%). These features had the most significant impact on the model's predictions, emphasizing their importance in determining employment status. Other features, such as "EngagementSurvey," "EmploymentDuration," and "Termd," also contributed but to a lesser extent.

Overall, the findings indicate that certain features, such as "TimeUntilTermination" and "TimeSinceLastReview," consistently play a crucial role in predicting employment status across all classifiers. These features provide valuable information for distinguishing between active employees, terminated employees, and employees who have left. Other features, such as "Termd," "EngagementSurvey," and "EmploymentDuration," also contribute but to a lesser extent.

Understanding the importance of these features enables organizations to focus on specific factors that significantly affect employment status. It helps in identifying potential risk factors for attrition, improving employee engagement, and making informed decisions related to human resource management.

**Section D (Regression):**

"TimeUntilTermination." This variable represents the time in days until an employee's termination from the last performance review. It is calculated using the "DateofTermination" and "LastPerformanceReview\_Date" features.

Considering this information, the analysis in Section D becomes more meaningful. The regression models, namely the Random Forest Regressor, Linear Regression Model, and Gradient Boosting Regressor, were trained to predict the "TimeUntilTermination" based on various employee features.

The evaluation scores obtained for each model, including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R2 score, provide an assessment of the models' predictive performance in estimating the time until an employee's termination.

Comparing the models:

1. Random Forest Regressor: The Random Forest Regressor showed feature importances, indicating the relative importance of each feature in predicting the "TimeUntilTermination." The top two influential features were "EmploymentStatus" (45.49%) and "Termd" (39.60%). These features had the most significant impact on the model's predictions.

The evaluation scores for the Random Forest Regressor were a Mean Absolute Error (MAE) of 43.38 days, a Root Mean Squared Error (RMSE) of 91.51 days, and an R2 score of 0.962. These scores indicate that the model performed well in estimating the "TimeUntilTermination" variable, with relatively low errors and a high R2 score.

1. Linear Regression Model: The Linear Regression Model's evaluation scores were a Mean Absolute Error (MAE) of 66.81 days, a Root Mean Squared Error (RMSE) of 99.48 days, and an R2 score of 0.957. Comparing these scores to the Random Forest Regressor, we can see that the Linear Regression Model had higher MAE and RMSE values and a slightly lower R2 score, suggesting slightly less accurate predictions.
2. Gradient Boosting Regressor: The Gradient Boosting Regressor exhibited the feature importances, with the most influential features being "EmploymentStatus" (67.08%) and "Termd" (30.02%). The model attributed varying degrees of importance to other features.

For the Gradient Boosting Regressor, the evaluation scores were a Mean Absolute Error (MAE) of 40.03 days, a Root Mean Squared Error (RMSE) of 77.71 days, and an R2 score of 0.973. These scores indicate that the Gradient Boosting Regressor performed the best among the three models, with the lowest errors (MAE and RMSE) and the highest R2 score.

In summary, considering the target variable "TimeUntilTermination," the Gradient Boosting Regressor model outperformed the Random Forest Regressor and Linear Regression Model in predicting the time until an employee's termination. The most influential features consistently found across the models were "EmploymentStatus" and "Termd," indicating their significance in estimating the "TimeUntilTermination." Other features had varying degrees of importance depending on the model.

# **Section E (Bonus):**

In Section E, there are three separate tasks described: calculating the employee retention rate for each year from 2008 until 2017 for every recruitment source and displaying it on a suitable graph, calculating the diversity index for each department based on race, gender, and age and displaying the departments in descending order of their diversity index, and creating a map visualization that shows the number of employees currently working in the company from each state. Let's go through each task in detail:

1. Calculating Employee Retention Rate by Year and Recruitment Source: The code snippet provided assumes that the data is already loaded into a DataFrame called 'df1'. Here's a breakdown of the steps involved:

* First, the 'DateofTermination' column is filled with the mode value to handle missing values.
* The data is then filtered to include only the years 2008 to 2017 using the 'Year' column, which is created by extracting the year from the 'DateofTermination' column.
* The retention rate is calculated for each recruitment source and year by grouping the data by 'RecruitmentSource' and 'Year' and applying a lambda function to compute the mean of the 'Termd' column (representing terminated employees) and subtracting it from 1.
* Finally, the retention rates are plotted on a graph using a box plot, where each box represents the retention rates for a specific year, and the x-axis represents the years. The recruitment sources are displayed as different boxes within each year.

1. Calculating Diversity Index by Department: The code snippet assumes that the data is already loaded into a DataFrame called 'df1'. Here's a breakdown of the steps involved:

* First, the 'DOB' (date of birth) and 'DateofHire' columns are converted to datetime format using the 'pd.to\_datetime' function.
* The 'AgeAtHire' column is calculated by subtracting the 'DOB' from the 'DateofHire' and converting the result to days divided by 365 to get the age at the time of hire.
* The diversity index is calculated for each department by grouping the data by 'Department' and applying aggregation functions to count the unique values of 'RaceDesc', 'GenderID', and 'AgeAtHire'. These values are then summed to obtain the diversity index.
* The departments are sorted in descending order based on their diversity index using the 'sort\_values' function.
* The departments and their corresponding diversity indices are displayed.

1. Creating a Map Visualization of Employee Distribution by State: The code snippet assumes that the data is already loaded into a DataFrame called 'df1'. Here's a breakdown of the steps involved:

* The number of employees from each state is counted using the 'value\_counts' function, resulting in a Series object with state names as the index and employee counts as the values.
* The state codes and employee counts are extracted from the Series object and stored in separate lists.
* A colormap is created using the 'plt.cm.get\_cmap' function, specifying the 'Blues' colormap.
* A figure and axes object are created, and a bar plot is generated using the state codes as the x-axis and employee counts as the heights of the bars. The bars are colored using the colormap based on the employee counts.
* Labels for the x-axis, y-axis, and title are added to the plot.
* A colorbar legend is created using the colormap, normalizing the minimum and maximum employee counts.
* The plot is displayed.

These code snippets provide a general idea of how to perform the described tasks. However, please note that you need to ensure the data is properly loaded, the necessary libraries are imported, and any additional modifications or customizations are made based on your specific data and requirements.

# **Section F (Performance - Bonus):**

In Section F, there are two parts: a classification task and a regression task. Let's break down each task and explain the code provided in detail:

1. Classification Task: In the classification task, the goal is to predict the "EmploymentStatus" variable based on the other features in the dataset. Here's a breakdown of the code:

* The independent variables (features) are stored in the variable 'X' by dropping the 'EmploymentStatus' column from the DataFrame 'df'. The dependent variable (target) is stored in the variable 'y'.
* The dataset is split into training and testing sets using the 'train\_test\_split' function from the 'sklearn.model\_selection' module. The training set comprises 80% of the data, and the testing set comprises the remaining 20%.
* A new Random Forest classifier is created with the best parameters. The classifier is instantiated with the 'RandomForestClassifier' class from the 'sklearn.ensemble' module.
* The classifier is fitted to the training data using the 'fit' method, with 'X\_train' and 'y\_train' as the input.
* The classifier is used to make predictions on the testing set using the 'predict' method, and the predicted labels are stored in the variable 'y\_pred'.
* Several performance measures are calculated using the predicted labels ('y\_pred') and the true labels from the testing set ('y\_test'). The metrics calculated include accuracy, precision, recall, F1 score, and AUC-ROC (Area Under the Receiver Operating Characteristic Curve) score.
* The calculated performance metrics are printed.

1. Regression Task: In the regression task, the objective is to predict the "TimeUntilTermination" variable based on the remaining features. Here's a breakdown of the code:

* The independent variables (features) are stored in the variable 'X' by dropping the 'TimeUntilTermination' column from the DataFrame 'df'. The dependent variable (target) is stored in the variable 'y'.
* The dataset is split into training and testing sets using the 'train\_test\_split' function from the 'sklearn.model\_selection' module. The training set comprises 80% of the data, and the testing set comprises the remaining 20%.
* A new Gradient Boosting Regressor is created with the best parameters. The regressor is instantiated with the 'GradientBoostingRegressor' class from the 'sklearn.ensemble' module.
* The regressor is fitted to the training data using the 'fit' method, with 'X\_train' and 'y\_train' as the input.
* The regressor is used to make predictions on the testing set using the 'predict' method, and the predicted values are stored in the variable 'y\_pred'.
* Several performance measures are calculated using the predicted values ('y\_pred') and the true values from the testing set ('y\_test'). The metrics calculated include mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), and R-squared score.
* The calculated performance metrics are printed.

In both tasks, the code provides a comprehensive evaluation of the models' performance using multiple performance measures. This approach ensures a balanced assessment and allows for fair comparison with other projects. The provided performance measures capture different aspects of model performance, such as accuracy, precision, recall, F1 score, and error metrics in the regression task. The scores demonstrate how well the models perform on the given classification and regression tasks.