PREPARED BY :
KARTHICK R

EASY VISA

2025-2026

Table of Contents

Table of Contents

List of figures

List of figures

List of figures

List of figures

List of Tables

List of Tables

List of Tables

List of Tables

Objective

Primary Objective

- To develop a Machine Learning-based classification system that predicts whether a visa application should be Certified or Denied, using historical data of employer-submitted applications. This predictive model aims to support the U.S. Department of Labor's Office of Foreign Labor Certification (OFLC) in making data-driven, scalable decisions amidst a rising volume of applications.

Business Problem Statement

In FY 2016, the OFLC processed over 775,979 applications for 1.7 million positions, marking a 9% increase from the previous year. The manual review process is becoming increasingly unsustainable. As the number of visa applications continues to grow annually, there is a critical need for an intelligent solution that:

- **Automates initial shortlisting**
- **Reduces manual effort**
- **Improves decision turnaround time**
- **Maintains fairness and consistency**

Objective

Analytical Goals

- Build a classification model to predict the **Case_Status** (Certified or Denied) based on applicant and job-related features.
- Identify the key variables that significantly influence the visa certification decision (e.g., wage, job title, location).
- Generate a recommendatory system that suggests the most favorable applicant profiles likely to succeed.
- Evaluate the model using precision-focused metrics to minimize false positives in certification.
- Deploy a prototype solution (optional) for use by internal analysts or caseworkers to input new applications and receive predictions.

Machine Learning Problem Formulation

- **Type:** Supervised Learning
- **Category:** Binary Classification
- **Target Variable:** Case_Status
  - Certified (Positive Class)
  - Denied (Negative Class)
- **Features (based on the dataset):** Employer details, job title, full-time/part-time, wage rate, job location, and others.

Data overview

- **Total Rows:** 25,480
- **Total Columns:** 12
- **Target Variable:** case_status (Certified or Denied)

Displaying the first 5 rows of the dataset

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_employees	region_of_employment	prevailing_wage	unit_of_wage	full_time_position	case_status
0	EZYV01	Asia	High School	N	N	14513	West	592.203	Hour	Y	Denied
1	EZYV02	Asia	Master's	Y	N	2412	Northeast	83425.650	Year	Y	Certified
2	EZYV03	Asia	Bachelor's	N	Y	44444	West	122996.860	Year	Y	Denied
3	EZYV04	Asia	Bachelor's	N	N	98	West	83434.030	Year	Y	Denied
4	EZYV05	Africa	Master's	Y	N	1082	South	149907.390	Year	Y	Certified

Table 1: Top five rows of dataset

Checking the shape of the dataset

- The dataset contains 25480 rows and 12 columns

Checking the data types of the columns for the dataset

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25480 entries, 0 to 25479
Data columns (total 12 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   case_id                25480 non-null  object 
 1   continent              25480 non-null  object 
 2   education_of_employee  25480 non-null  object 
 3   has_job_experience     25480 non-null  object 
 4   requires_job_training  25480 non-null  object 
 5   no_of_employees        25480 non-null  int64  
 6   yr_of_estab            25480 non-null  int64  
 7   region_of_employment   25480 non-null  object 
 8   prevailing_wage        25480 non-null  float64
 9   unit_of_wage           25480 non-null  object 
 10  full_time_position     25480 non-null  object 
 11  case_status            25480 non-null  object 
dtypes: float64(1), int64(2), object(9)
memory usage: 2.3+ MB
```

Table 2: Data types of the column

- Only 3 variables(no_of_employees, yr_of_estab, prevailing_wage) are numerical rest all are object types.

Statistical summary of the dataset

	no_of_employees	yr_of_estab	prevailing_wage
count	25480.000	25480.000	25480.000
mean	5667.043	1979.410	74455.815
std	22877.929	42.367	52815.942
min	-26.000	1800.000	2.137
25%	1022.000	1976.000	34015.480
50%	2109.000	1997.000	70308.210
75%	3504.000	2005.000	107735.513
max	602069.000	2016.000	319210.270

Table 3: Statistical summary of the dataset

- The average number of employes are 5667
- The average wage is 74455.815.

Value counts of case_status

case_status	count
Certified	17018
Denied	8462

Table 4: Value counts of dataset

Checking for duplicate values

- There are no duplicate values in the data.

Checking for null values

	0
case_id	0
continent	0
education_of_employee	0
has_job_experience	0
requires_job_training	0
no_of_employees	0
yr_of_estab	0
region_of_employment	0
prevailing_wage	0
unit_of_wage	0
full_time_position	0
case_status	0

Table 5: Data on null values

- There are no null values in the data.

Exploratory Data Analysis

Problem Definition

- To explore and understand the visa application dataset with the goal of identifying key patterns, trends, and influential factors that impact the Case Status (Certified / Denied) of visa applications. This will inform feature selection and guide the development of a robust machine learning model for predicting visa approval.

- Univariate Analysis
  - Distribution and summary statistics of individual variables.
  - Identify anomalies or outliers in continuous features.
  - Frequency counts for categorical features (like Job_Title, Employer_Name, Visa_Class, etc.).

- Bivariate Analysis
  - How does each feature relate to Case_Status?
  - E.g., Does a certain Visa_Class have higher certification rates?
  - Use cross-tabulations, groupby analysis, and visualizations.

- Multivariate Analysis
  - Explore interactions between multiple features.
  - Analyze correlations (if numeric features exist).
  - Investigate multicollinearity (if needed).

Univariate analysis

Analysis on no_of_employees

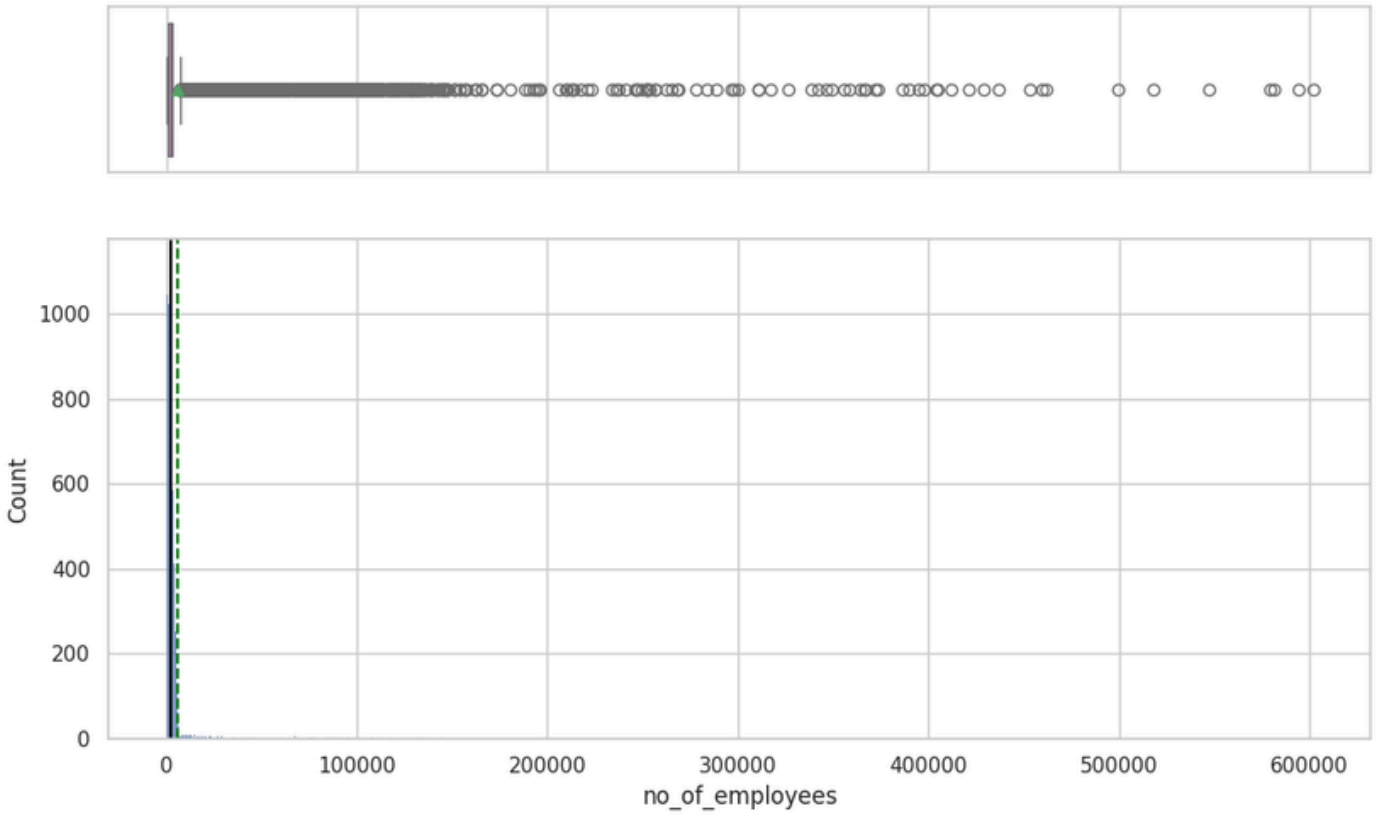

Figure 1: No_of_employees

Observations on Content views

- The histogram shows a long tail toward high values, meaning a small number of companies report extremely large employee counts (outliers), while the majority have a low number of employees.
- The majority of observations fall between 0 and a few hundred employees.
- This suggests most visa applications come from small to mid-sized companies.

Analysis on Year of Establishment

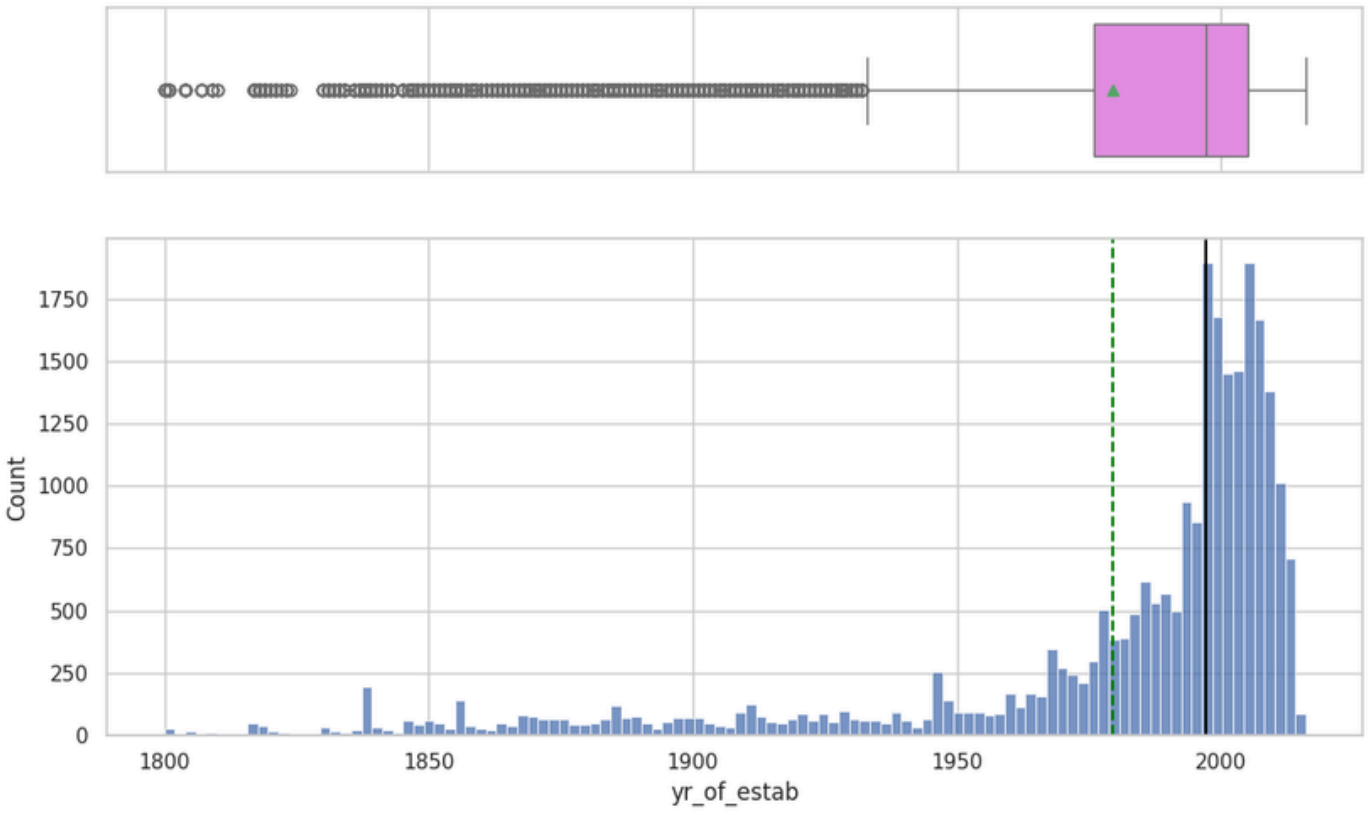

Figure 2: Year of Establishment

Observations on Year of Establishment

- The histogram shows a sharp rise in the number of establishments starting around 1980, with a peak around 2000–2010.
- The box plot indicates many outliers on the lower end (early establishment years, e.g., 1800s).
- The box plot's median line (also marked by a black vertical line in the histogram) falls in the late 1990s, indicating that at least half of the entities were established after this point.

Analysis on prevailing wage

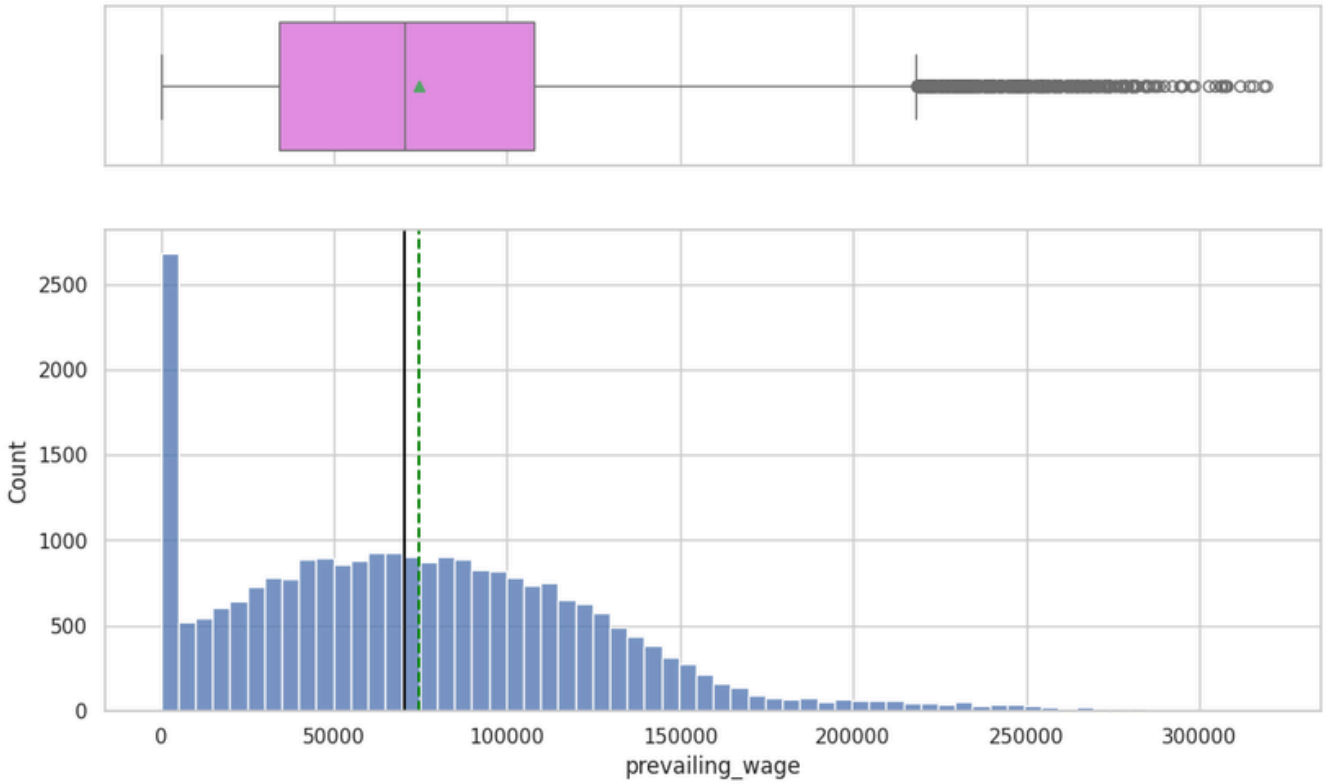

Figure 3: prevailing wage

Observations on prevailing wage

- Many outliers exist on the higher end of the wage range, indicating a right-skewed distribution.
- A sharp spike is observed at or near zero wages, suggesting possible data entry issues or placeholders.
- The median wage is around $70,000, while the mean is slightly higher, confirming positive skewness.

Analysis on Continent

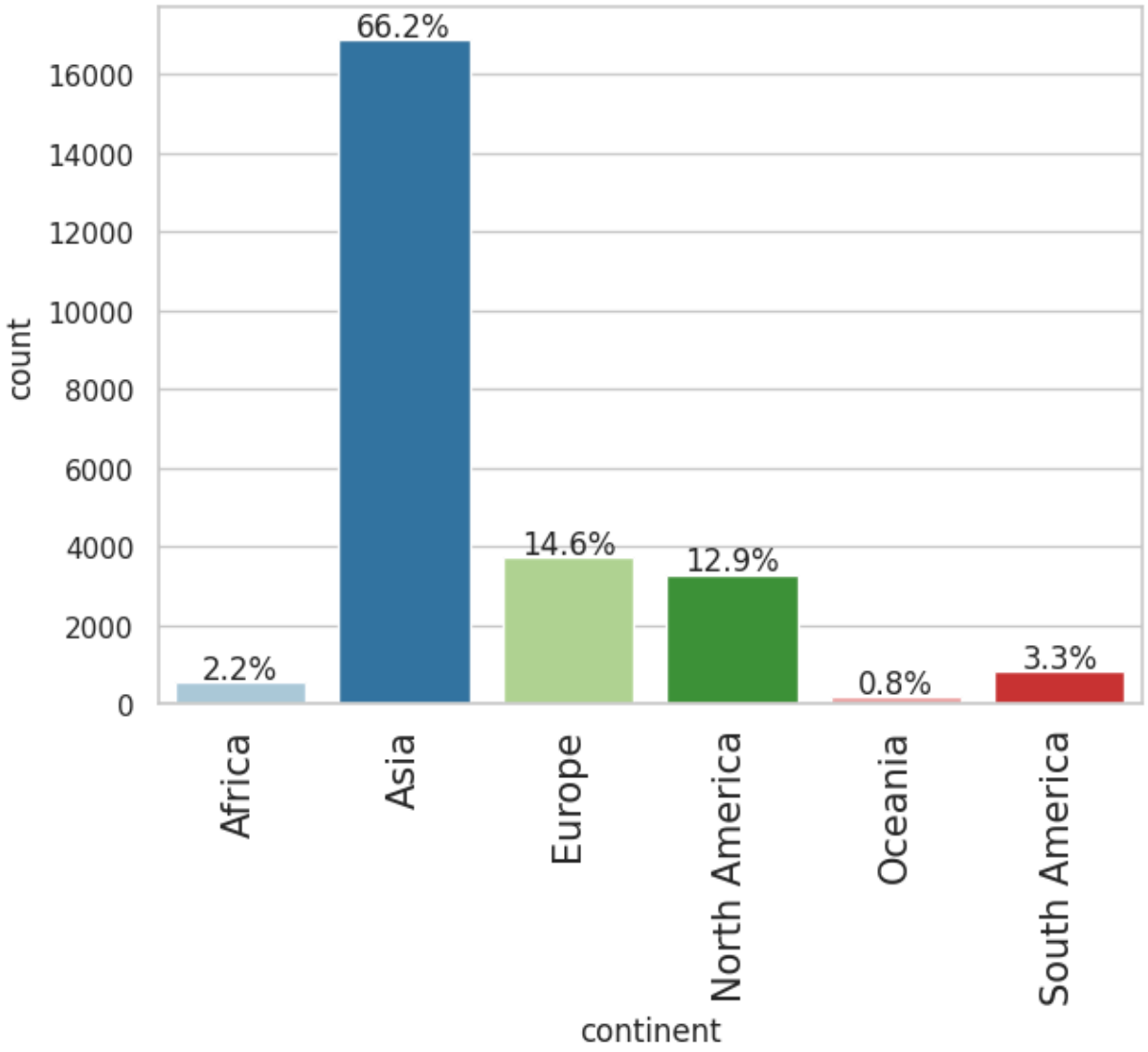

Figure 4: Continent analysis

Observations on Continent

- Asia contributes the majority of applications, accounting for 66.2% of the total.
- Europe and North America follow distantly with 14.6% and 12.9% respectively.
- Africa, South America, and Oceania contribute minimal shares, all under 3.5%.

Analysis on Education of employee

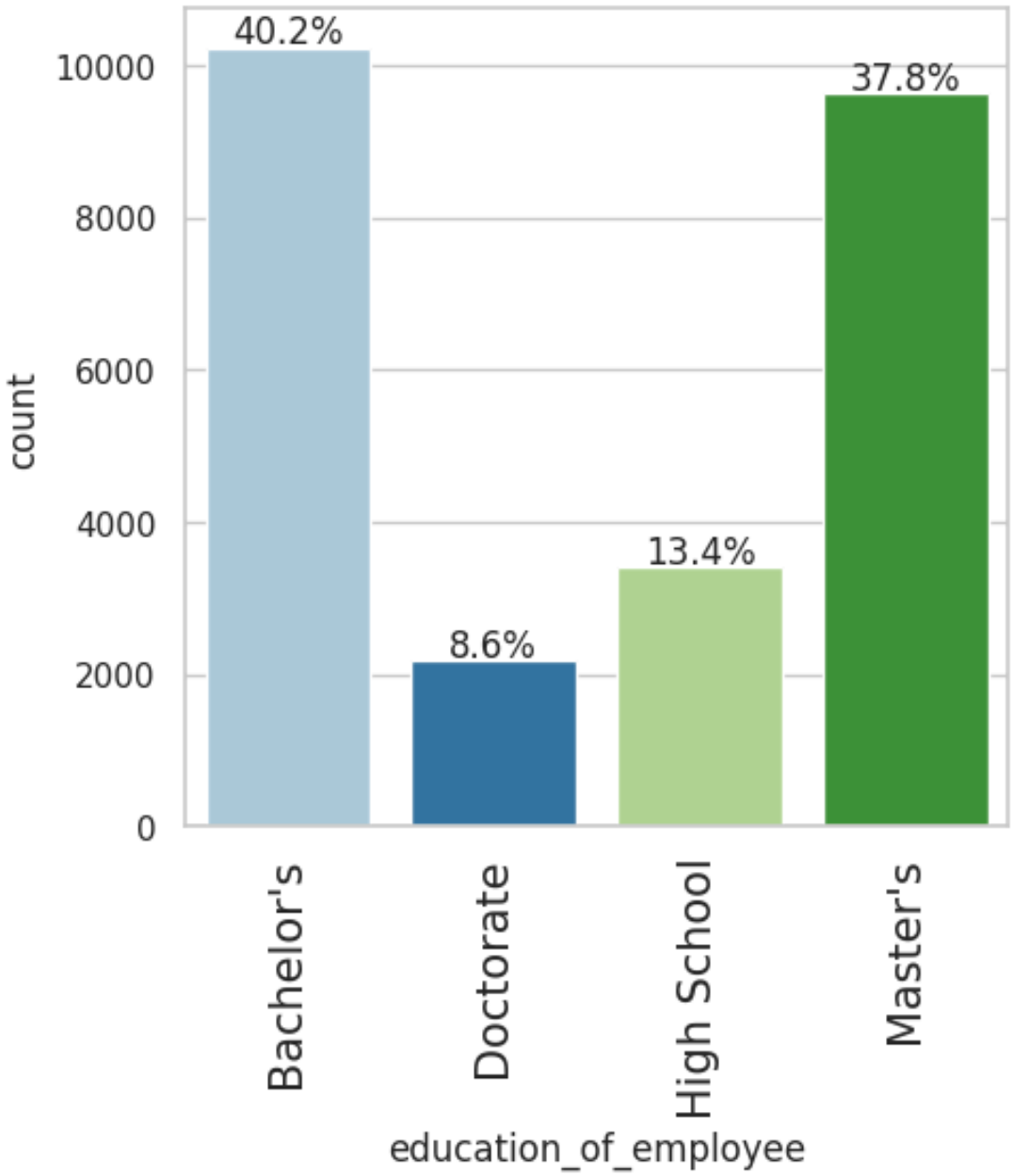

Figure 5: Education of employee

Observations on Education of employee

- Bachelor's degree holders make up the largest group at 40.2%.
- Master's degree holders follow closely at 37.8%.
- Doctorate and High School qualifications are much less common, comprising 8.6% and 13.4% respectively.

Analysis on has_job_experience

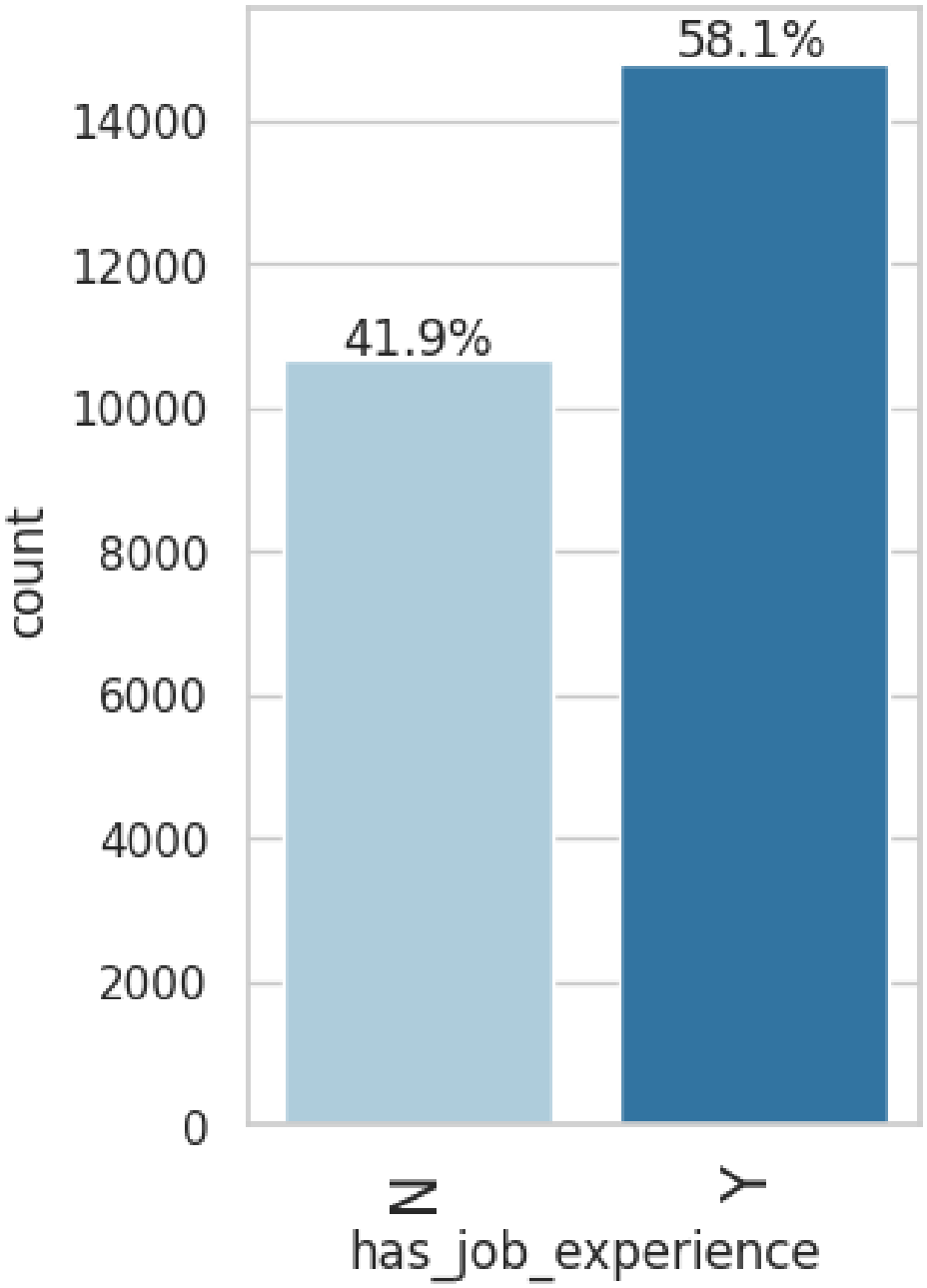

Figure 6: has_job_experience

Observations on has_job_experience

- 58.1% of applicants have prior job experience.
- 41.9% of applicants do not have prior job experience.
- Applicants with job experience form the majority in the dataset.

Analysis on Requires job training

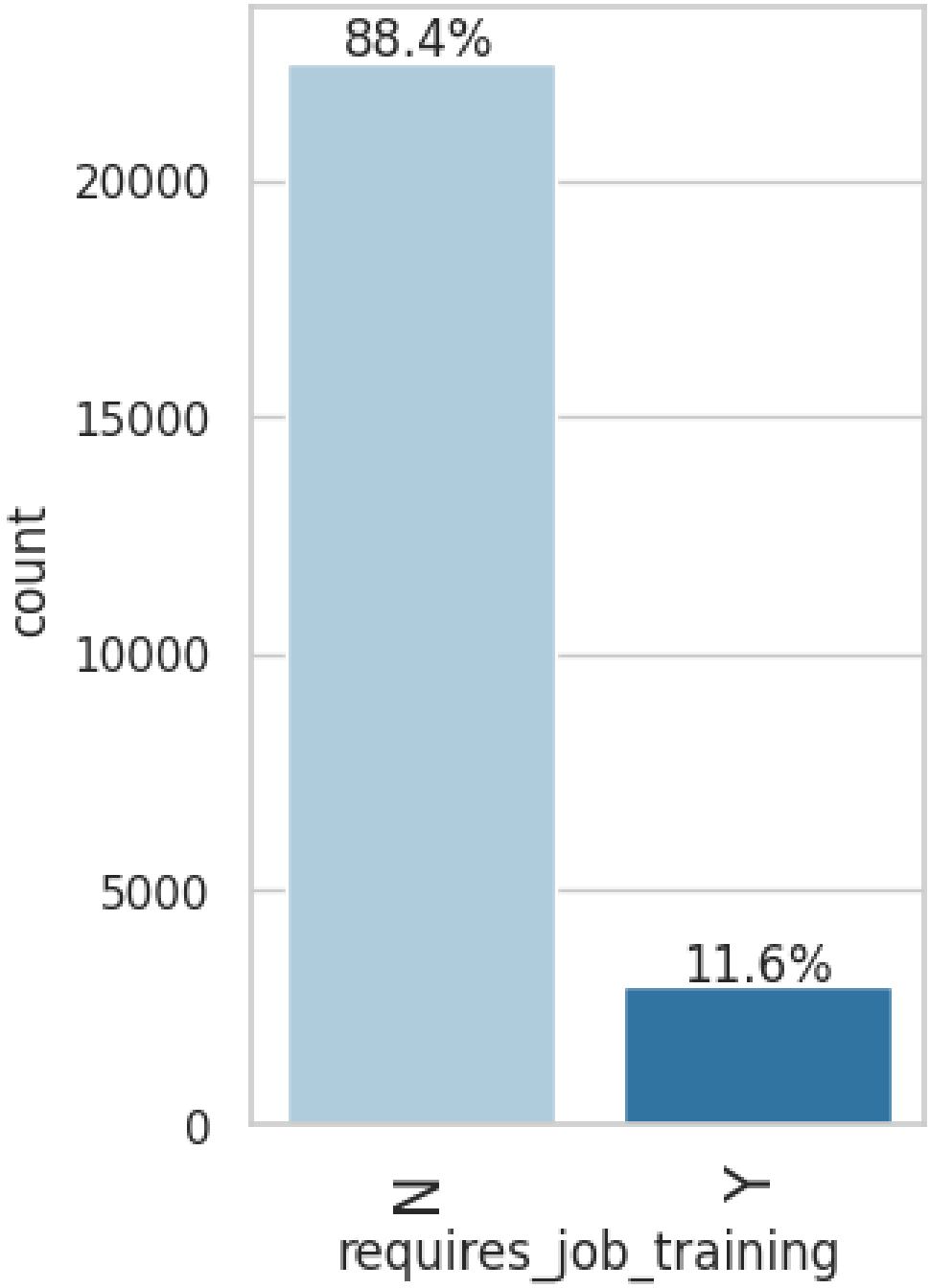

Figure 7: Requires job training

Observations on Requires job training

- 88.4% of applicants do not require job training.
- Only 11.6% of applicants need job training.
- Most roles appear to target candidates who are already job-ready.

Analysis on Region of employment

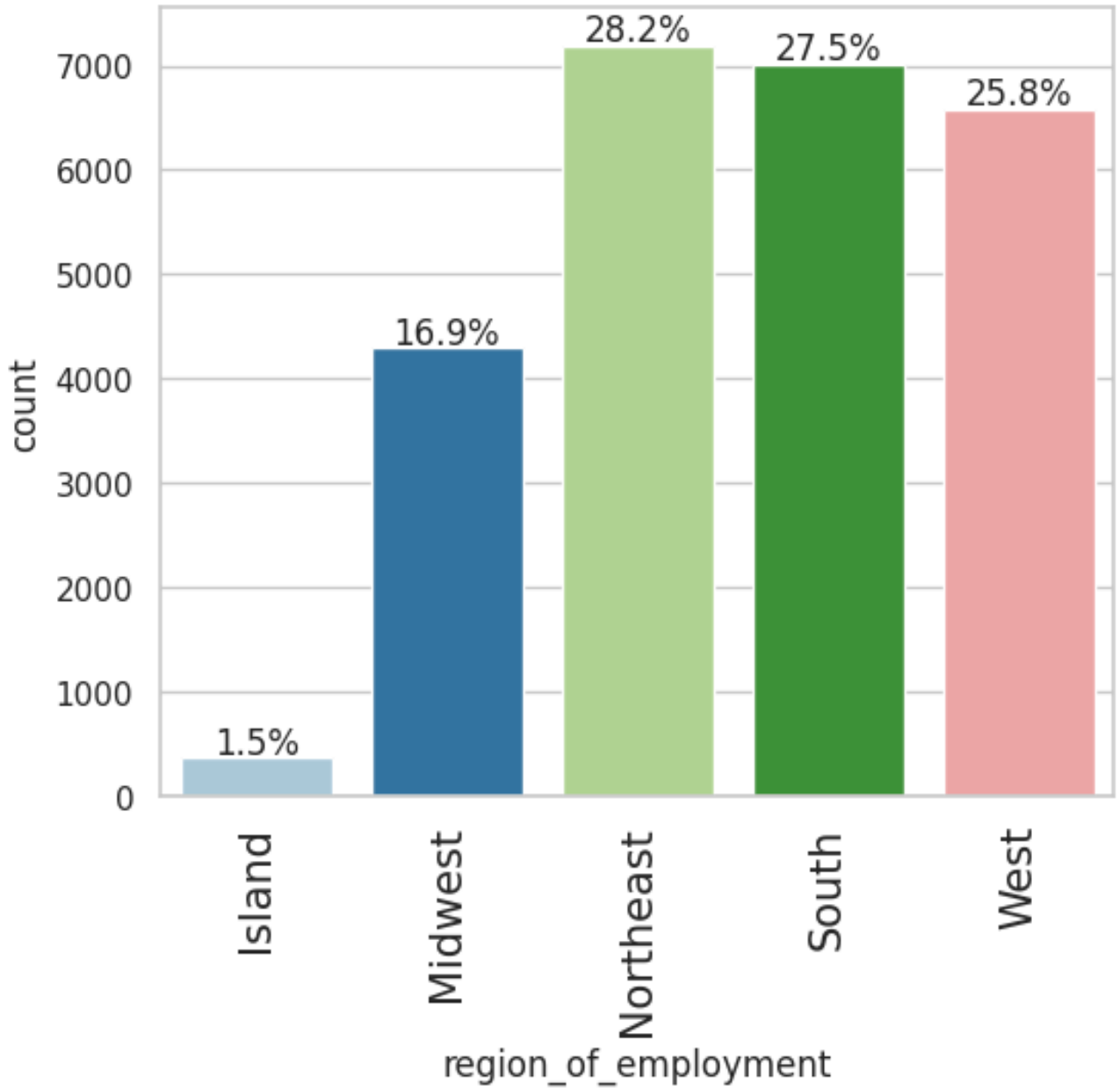

Figure 8: Region of employment

Observations on Region of employment

- Northeast has the highest share of employment at 28.2%, closely followed by South at 27.5%.
- West accounts for 25.8% of employment, showing fairly even distribution across the top three regions.
- Island region contributes the least, with only 1.5% of the total.

Analysis on Unit of wage

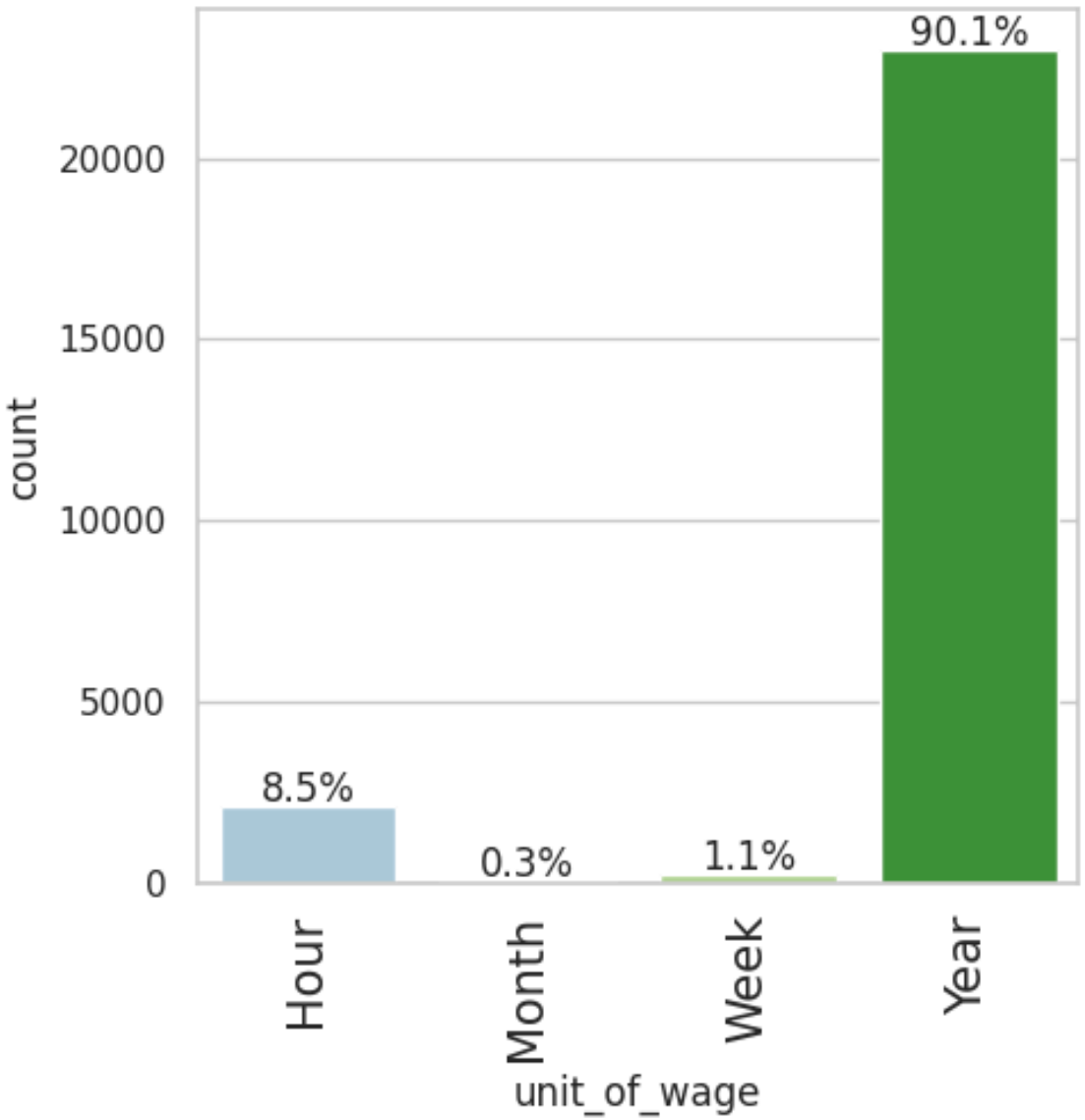

Figure 9: Unit of wage

Observations on Unit of wage

- 90.1% of wage entries are reported on a yearly basis.
- Hourly wages account for 8.5% of the data.
- Weekly and monthly wage units are rare, making up only 1.1% and 0.3% respectively.

Analysis on Full time position

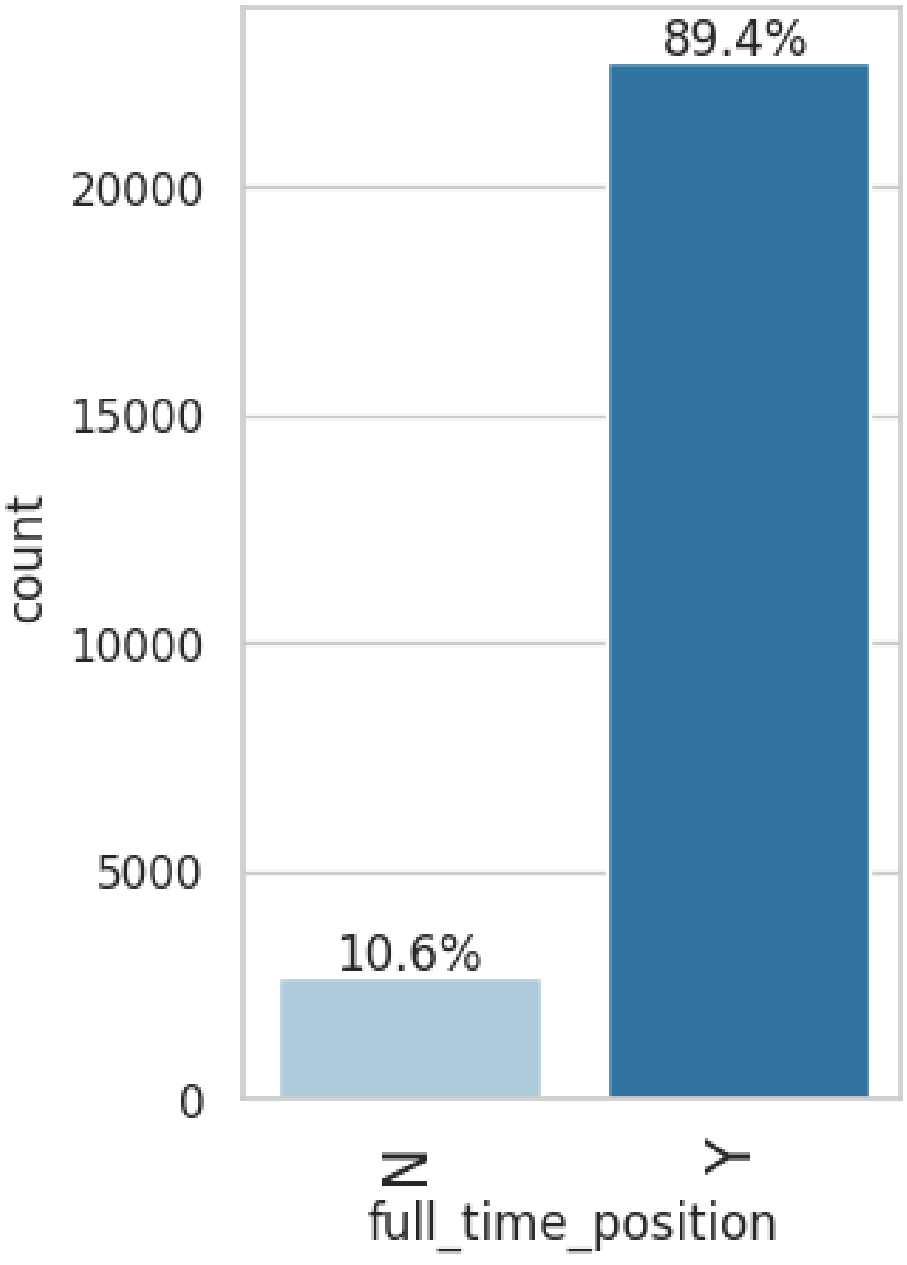

Figure 10: Full time position

Observations on Full time position

- 89.4% of the positions are full-time.
- Only 10.6% of the applications are for non–full-time roles.
- Full-time roles clearly dominate the dataset.

Analysis on Case status

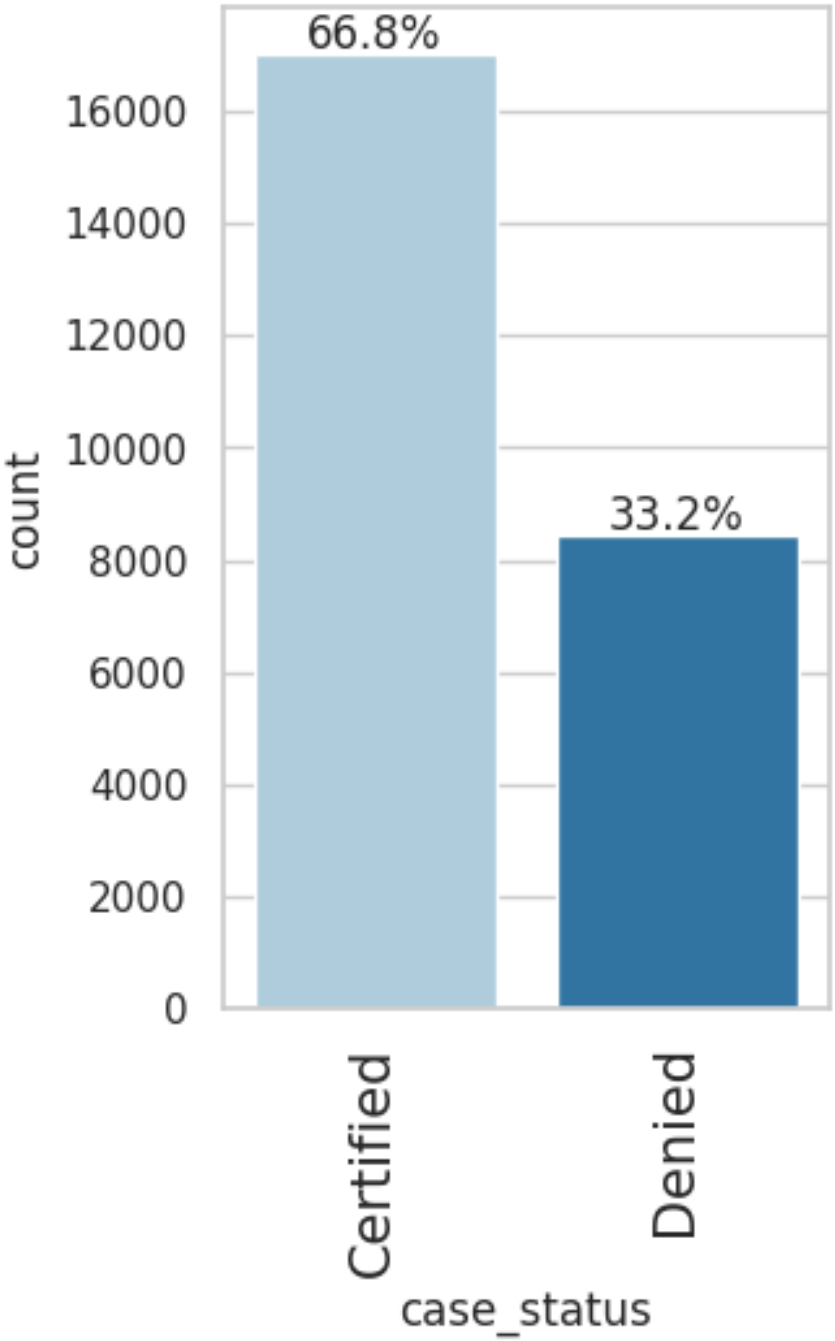

Figure 11: Case status

Observations on Case status

- 66.6% of the applicants are certified.
- Only 33.2% of the applicants are denied.

Analysis between continent and case status

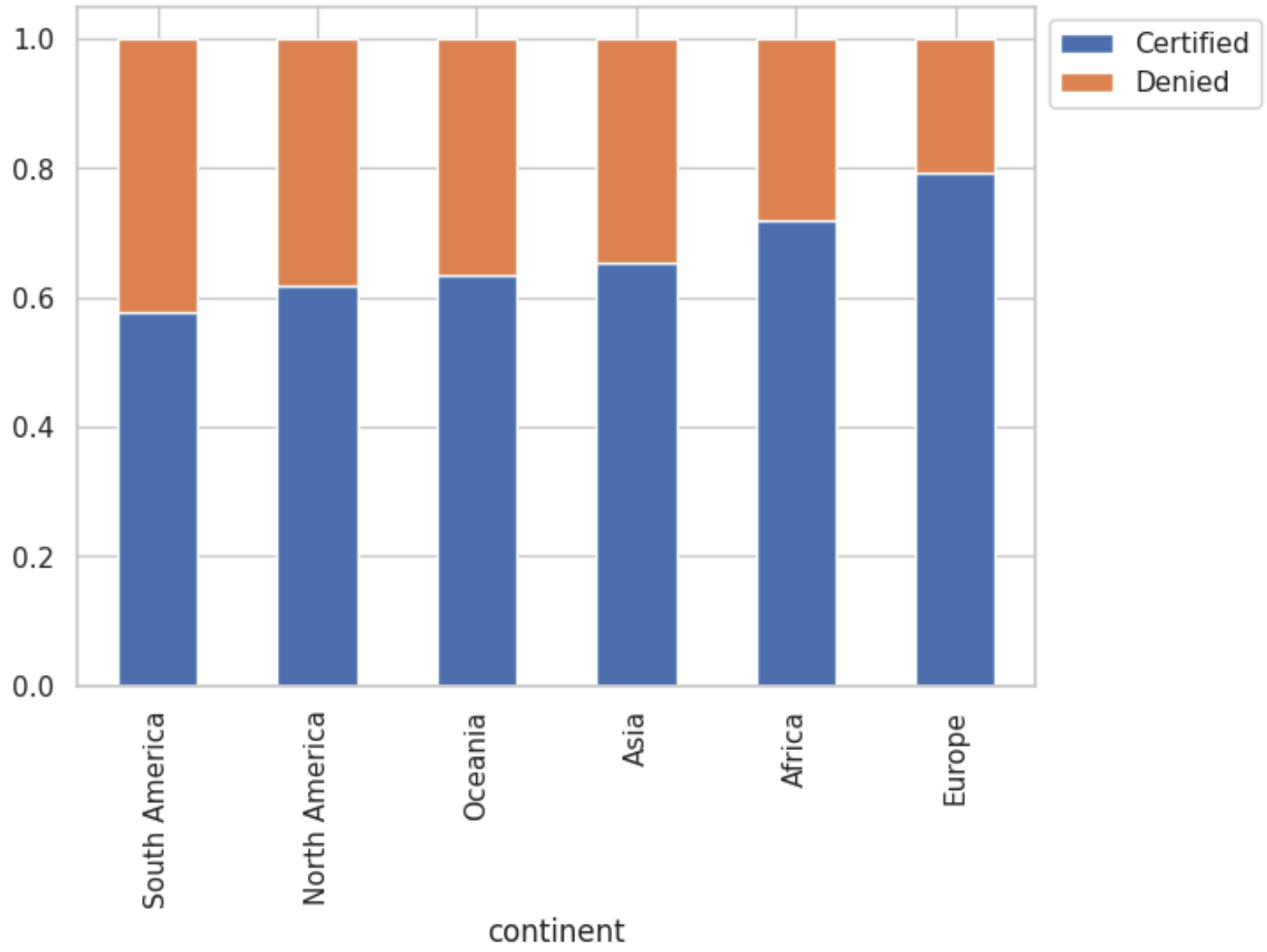

Figure 12: Analysis between continent and case status

Observations on continent and case status

- Europe has the highest certification rate (approx. 80%).
- South America has the lowest certification rate (approx. 58-59%).
- Asia, North America, and Oceania show similar mid-range certification rates (60-65%).

Analysis between Education of employee and case status

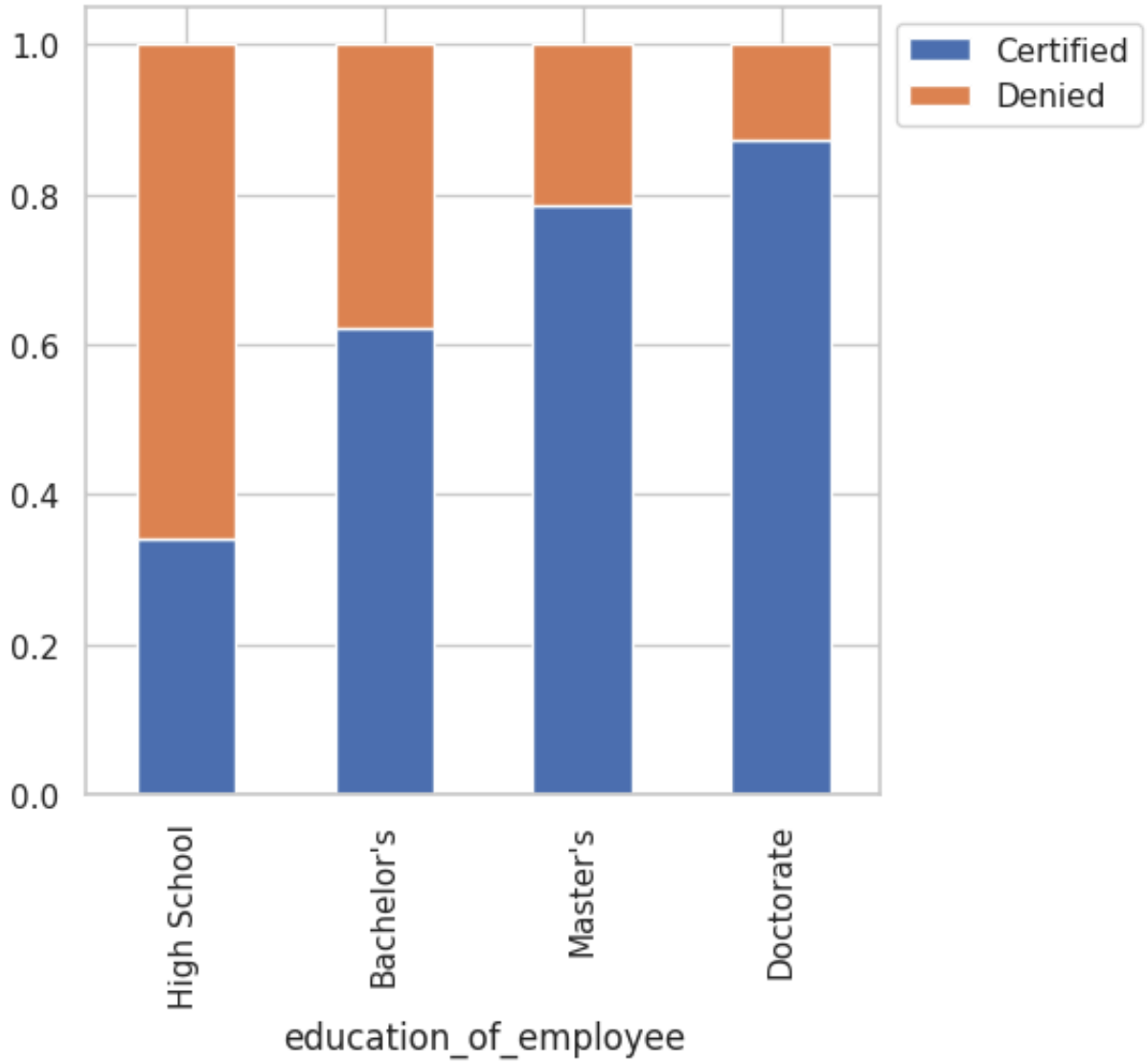

Figure 13: Analysis between Education of employee and case status

Observations on Education of employee and case status

- Higher education levels show progressively stronger correlations, peaking at Doctorate (1.0).
- Master's degrees show moderate impact (0.6-0.8), while Bachelor's and High School have minimal influence (0.0-0.4).
- The data suggests a clear education premium - more advanced degrees correlate with better outcomes.

Analysis between Has job experience nd case status

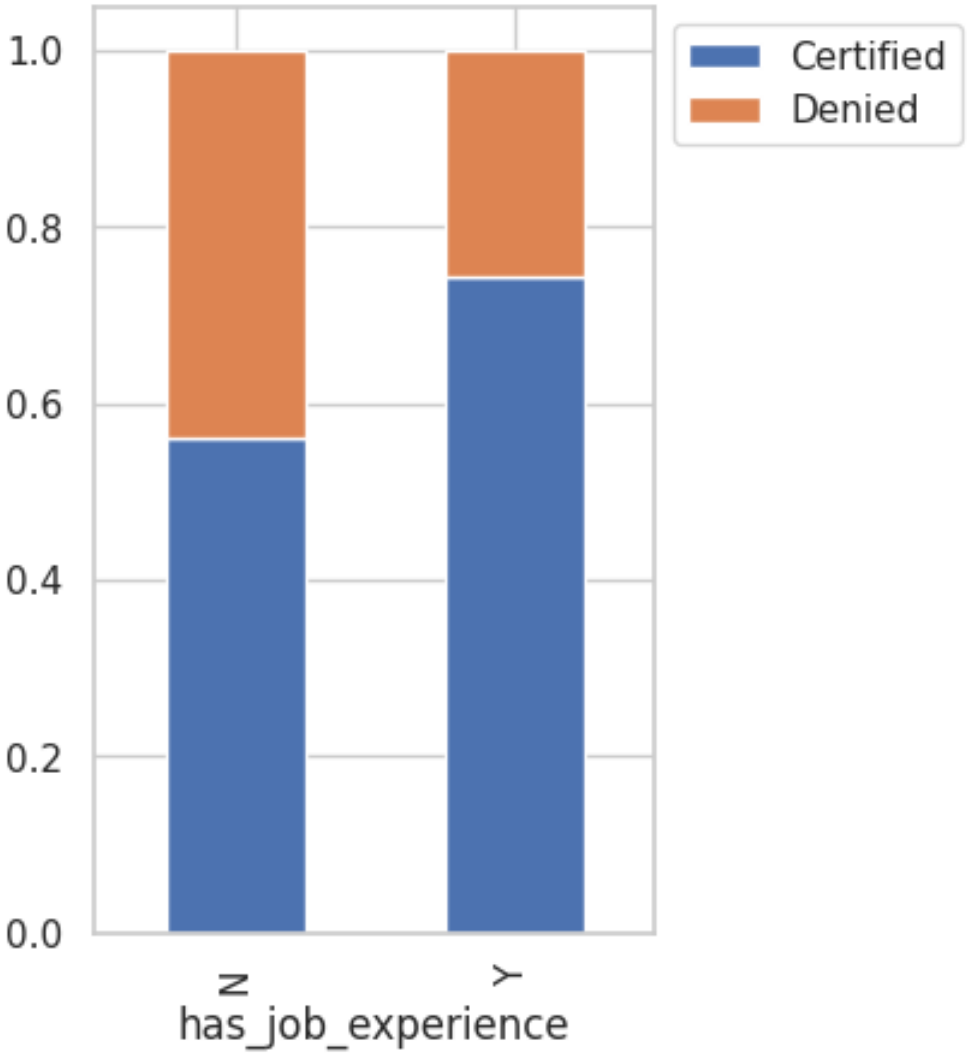

Figure 14: Analysis between Has job experience nd case status

Observations on Has job experience nd case status

- Job Experience Boosts Certification: Cases with work experience show a strong positive correlation (0.8) with certified status, while denied cases correlate weakly (0.2).
- Experience Matters More Than Education: Compared to the education chart, job experience shows a stronger link to certification than even advanced degrees.
- Key Differentiator: The 0.6 gap between certified and denied cases suggests experience is a decisive factor in approval decisions.

Analysis between No of employees and case status

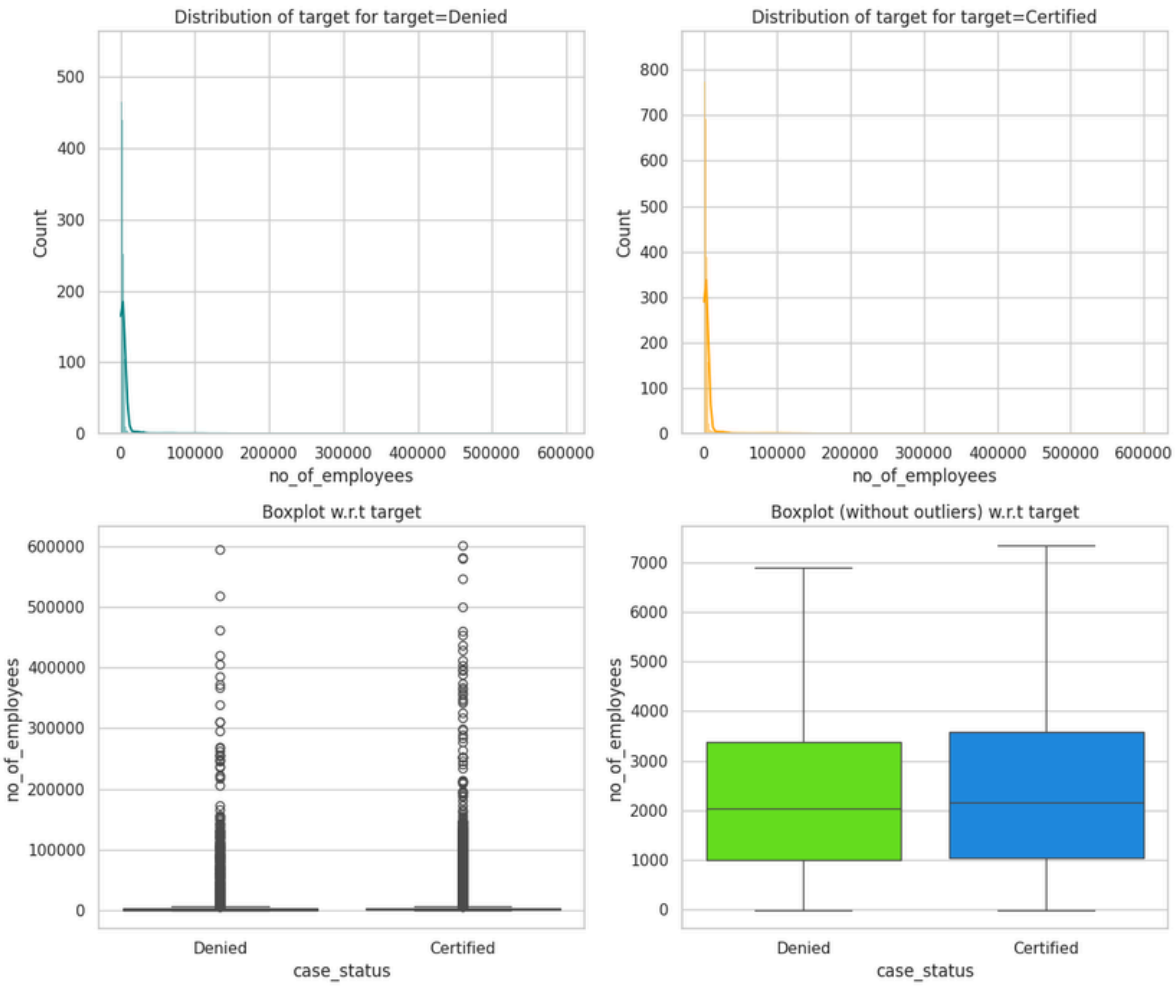

Figure 15: No of employees and case status

Observations on No of employees and case status

- Most cases concentrate at low employee counts ($<$10,000).
- Significant outliers exist, showing some very large companies.
- Medians and IQRs for 'Denied' and 'Certified' are similar when outliers are removed.

Analysis between year of establishment and case status

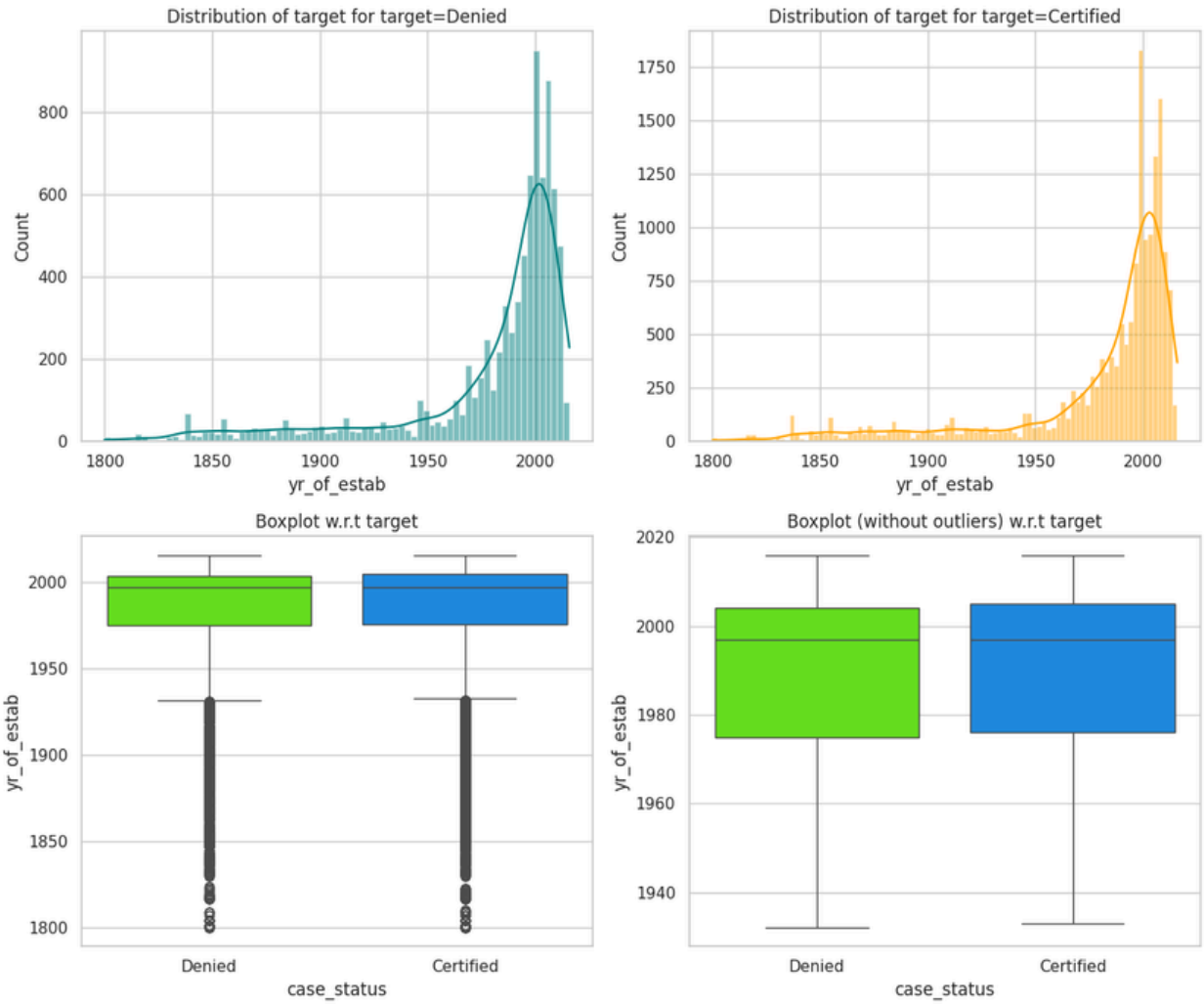

Figure 16: year of establishment and case status

Observations on year of establishment and case status

- Most applications are from companies established post-1990s.
- Median establishment year is similar for both denied and certified cases.
- Outliers indicate some very old companies are also applying.

Analysis between prevailing wage and case status

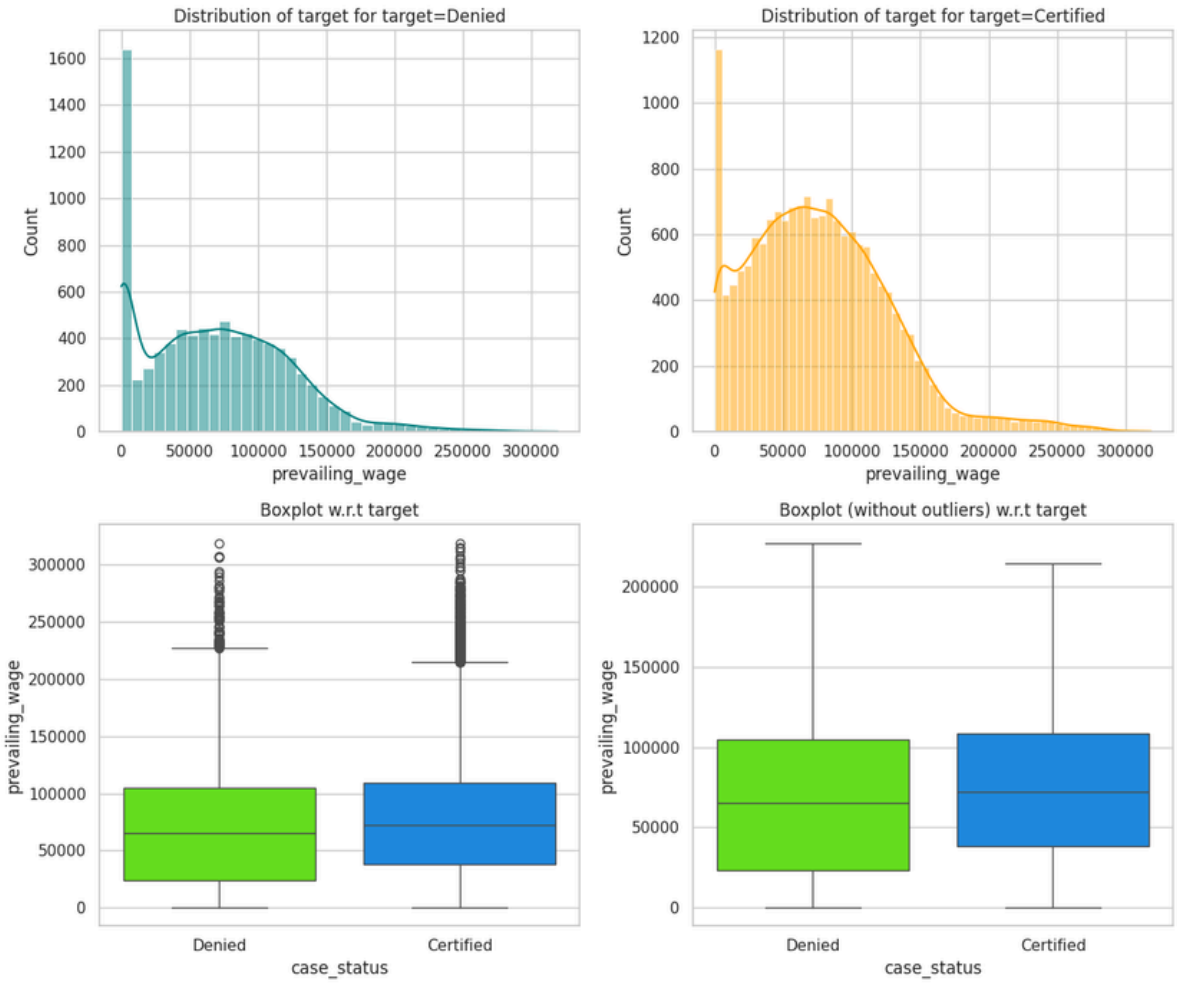

Figure 17: prevailing wage and case status

Observations on prevailing wage and case status

- Denied cases show wider wage ranges, especially at high/low extremes, while certified cases cluster around moderate wages.
- Denied applications include more outliers (very high wages), whereas certified cases exclude them, suggesting extreme wages hurt approval chances.
- Most certified cases fall within a narrower mid-range (likely $50k–$150k), indicating alignment with this range improves success.

Analysis of Case status for numerical variable

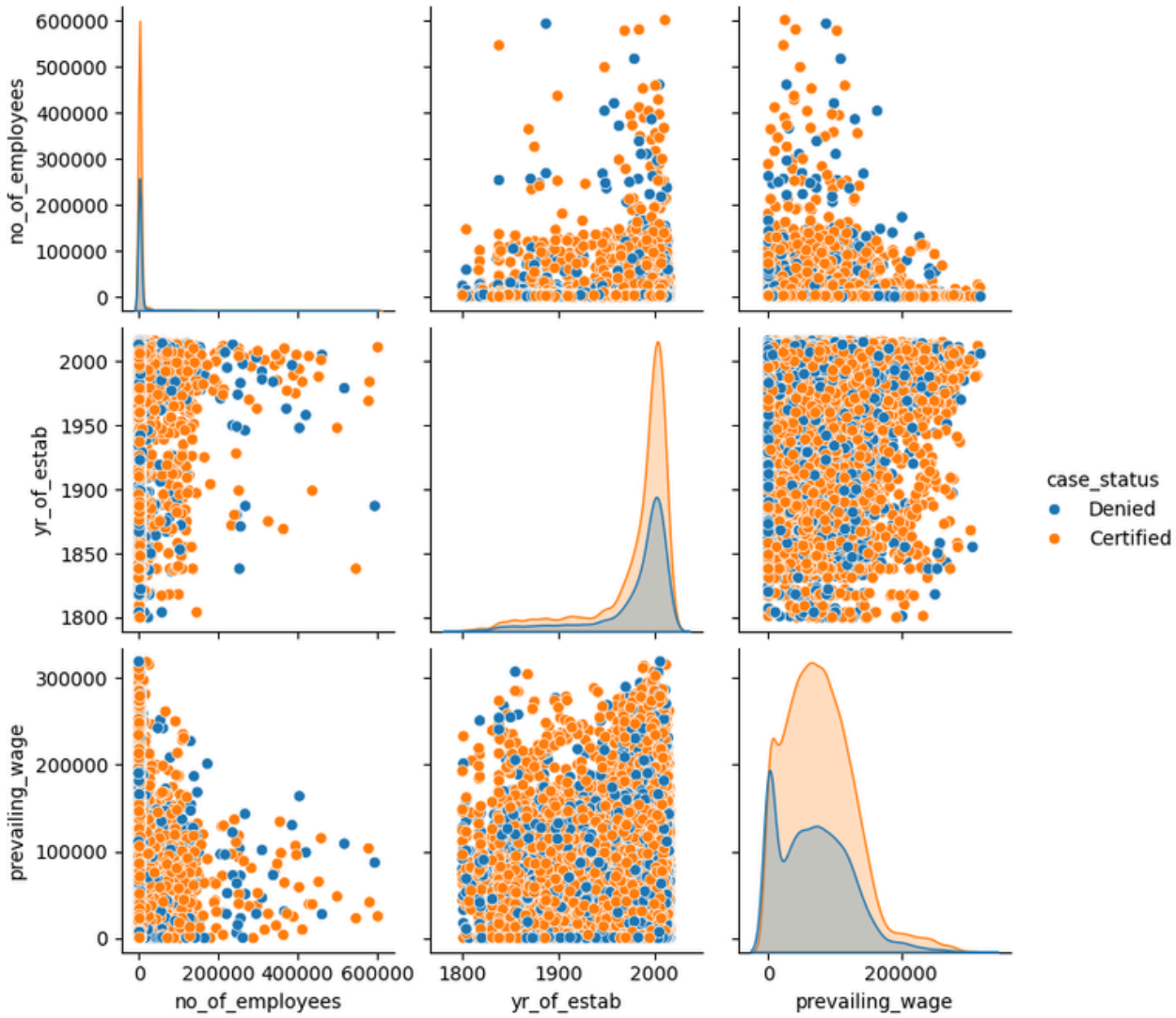

Figure 18: Case status for numerical variable

Observations on Case status for numerical variable

- Employee counts vary widely (0–600K), showing diverse company sizes.
- Older companies (est. 1800–2000) sometimes have unusually high employee numbers (up to 300K).
- Higher wages (up to 600K) link to larger firms, but certification depends on more than just wage or size.
- **Older/larger firms pay more, but approvals need balanced criteria.**

Correlation matrix

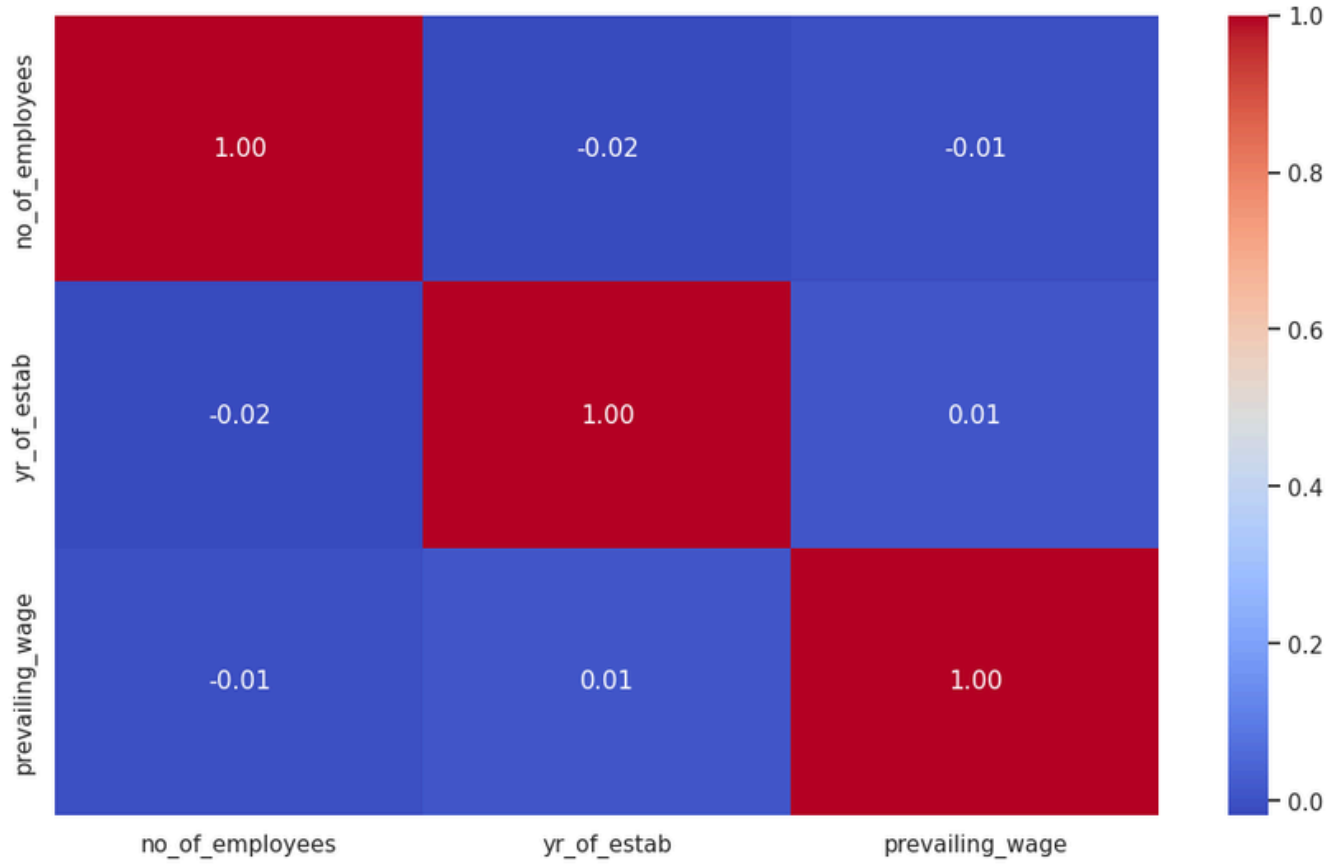

Figure 19: Correlation matrix

Observations on Correlation matrix

- Weak Correlations: The metrics (employees, establishment year, wage) show near-zero correlation (-0.02 to 0.01), indicating these factors operate independently.
- No Strong Patterns: The lack of significant correlations (all values close to 0) suggests no direct linear relationships between company size, age, and wages in this dataset.
- Isolated Trends: Despite weak overall links, the second chart hints at a possible non-linear relationship (values 0.0–1.0), warranting deeper analysis.

Data preprocessing

- Creating a copy of the data and naming as **data1.**

Removing Case ID column

- "case_id" is likely an identifier (e.g., application ID) with no predictive value for analysis/modeling. Removing it avoids noise in statistical or machine learning tasks.

Manual Imputation of numerical values

- we need to pass numerical values for each categorical column for imputation so we will label encode them.

	continent	education_of_employee	has_job_experience	requires_job_training	no_of_employees
0	0	0	0	0	14513
1	0	2	1	0	2412
2	0	1	0	1	44444
3	0	1	0	0	98
4	1	2	1	0	1082

yr_of_estab	region_of_employment	prevailing_wage	unit_of_wage	full_time_position	case_status
2007	0.000	592.203	0	1	Denied
2002	1.000	83425.650	3	1	Certified
2008	0.000	122996.860	3	1	Denied
1897	0.000	83434.030	3	1	Denied
2005	2.000	149907.390	3	1	Certified

Table 6: Numerical imputation values

Outlier Detection

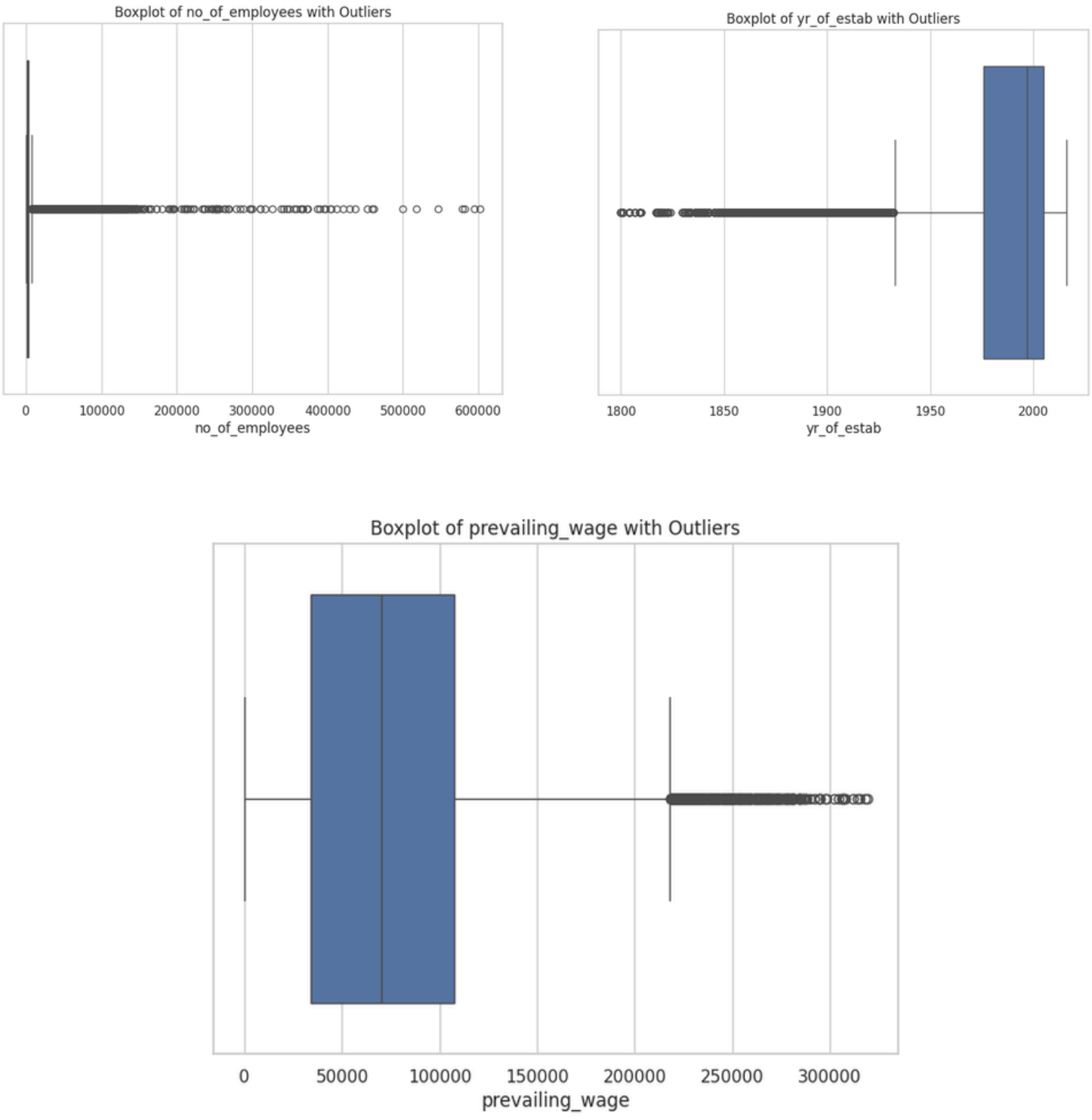

Figure 20: Outlier Detection

- There are quite a few outliers in the data.
- However, we will not treat them as they are proper values.

Checking for missing values

	0
case_id	0
continent	0
education_of_employee	0
has_job_experience	0
requires_job_training	0
no_of_employees	0
yr_of_estab	0
region_of_employment	0
prevailing_wage	0
unit_of_wage	0
full_time_position	0
case_status	0

dtype: int64

Table 7: Checking Missing values

- There are no missing values in the data.

Checking for duplicate values

- There are no duplicate values in the data.

Feature engineering

- **case_status** is the target variable.

- spliting the dataset data1 into features (X) and target labels (y) for machine learning

- 1. Features (X):
  - Removes the column **"case_status"** (the target variable) from **data1**.
  - All remaining columns (e.g., **prevailing_wage**, **education**, **has_job_experience**) become **input features** for the model.

- 2. Target Labels (y):
  - Extracts "**case_status**" (Certified/Denied) and converts it to binary (1/0):
  - **1** → "**Certified**" (approved)
  - **0** → "**Denied**" (rejected)
  - This is done because machine learning models typically require numerical labels.

Two-step train-validation-test sp

- Step 1: Initial Split (Test Set)
  - **Purpose**: Reserve **20%** of the data (**test_size=0.2**) as the **final test set** (**X_test**, **y_test**).
  - The remaining 80% of data (temporary set) for further splitting.

- Step 2: Split Temporary Set (Train/Validation)
  - Purpose: Split the temporary set into training (60%) and validation (20%) sets.
  - test_size=0.25 means 25% of X_temp (which is 25% of 80% = 20% of original data) becomes validation.

The remaining **75%** (60% of original data) is training data.

Final Data Proportions

- Train: 60% of original data.
- Validation: 20% of original data.
- Test: 20% of original data.

```
(15288, 10) (5096, 10) (5096, 10)
```

Figure 20: Outlier Detection

- Number of rows in train data = 15288
- Number of rows in validation data = 5096
- Number of rows in test data = 5096

Reverse encoding

Purpose of inverse_mapping

- The function likely takes two arguments:
- A **mapping dictionary** (e.g., continent_map, education_map) that was originally used to convert categorical values (like "Asia", "Bachelor's") into numerical codes (e.g., 1, 2).
- A **column name** (e.g., "continent", "education_of_employee") in the dataset to apply the inverse mapping to.
- It **reverts numerical codes back to their original categorical labels** for better interpretability.

- Ensure the mapping dictionaries (e.g., education_map) are correctly defined (e.g., {"Bachelor's": 2, "Master's": 3}).

- **Since we are splitting the data into three, before one hot encoding, the data leakage is prevented.**

Ensuring no data leakage among train-test and validation sets

Training Dataset

```
continent
Asia               10085
Europe              2285
North America       1944
South America        528
Africa               333
Oceania              113
Name: count, dtype: int64
**********
education_of_employee
Bachelor's      6141
Master's        5792
High School     2045
Doctorate       1310
Name: count, dtype: int64
**********
has_job_experience
Y    8845
N    6443
Name: count, dtype: int64
**********
requires_job_training
N    13477
Y     1811
Name: count, dtype: int64
**********
region_of_employment
Northeast    4312
South        4248
West         3920
Midwest      2576
Name: count, dtype: int64
**********
unit_of_wage
Year     13786
Hour      1286
Week       156
Month       60
Name: count, dtype: int64
**********
full_time_position
Y    13678
N     1610
Name: count, dtype: int64
**********
```

Table 8: Training dataset

Validation Dataset

```
continent
Asia                3395
Europe               713
North America        655
South America        173
Africa               121
Oceania               39
Name: count, dtype: int64
**********
education_of_employee
Bachelor's      2033
Master's        1886
High School      694
Doctorate        483
Name: count, dtype: int64
**********
has_job_experience
Y    2963
N    2133
Name: count, dtype: int64
**********
requires_job_training
N    4501
Y     595
Name: count, dtype: int64
**********
region_of_employment
Northeast    1430
South        1389
West         1352
Midwest       855
Name: count, dtype: int64
**********
unit_of_wage
Year     4576
Hour      452
Week       57
Month      11
Name: count, dtype: int64
**********
full_time_position
Y    4552
N     544
Name: count, dtype: int64
**********
```

Table 9: Validation dataset

Test Dataset

```
continent
Asia               3381
Europe              734
North America       693
South America       151
Africa               97
Oceania              40
Name: count, dtype: int64
**********
education_of_employee
Bachelor's      2060
Master's        1956
High School      681
Doctorate        399
Name: count, dtype: int64
**********
has_job_experience
Y    2994
N    2102
Name: count, dtype: int64
**********
requires_job_training
N    4547
Y     549
Name: count, dtype: int64
**********
region_of_employment
Northeast    1453
South        1380
West         1314
Midwest       876
Name: count, dtype: int64
**********
unit_of_wage
Year     4600
Hour      419
Week       59
Month      18
Name: count, dtype: int64
**********
full_time_position
Y    4543
N     553
Name: count, dtype: int64
**********
```

Table 10: Test dataset

one-hot encoding

- **Converts categorical variables into dummy/indicator variables:**
- For each categorical column (e.g., "continent", "education_of_employee"), it creates binary (0/1) columns for each category.
- Example: If "continent" has values ["Asia", "Europe"], it creates:
- continent_Asia (1 if Asia, 0 otherwise)
- continent_Europe (1 if Europe, 0 otherwise).

```
(15288, 20) (5096, 20) (5096, 20)
```

Table 11: Shape after one hot encoding

- After one hot encoding the training set have 15288 rows and 20 columns , validation set has 5096 rows and 20 columns and test set has 5096 rows and 20 columns.

Model building - Original Data

- Performs model training, cross-validation, and validation set evaluation for multiple ensemble and tree-based classifiers.

- **Purpose**: Initializes a list of 6 models:
- **Ensemble Methods:** **Bagging, Random Forest, Gradient Boosting (GBM), AdaBoost, XGBoost.**
- **Baseline**: Decision Tree (dtree).

```
Training Performance:

Bagging: 0.9891069676153091
Random forest: 1.0
GBM: 0.7838827838827839
Adaboost: 0.7624936900555275
Xgboost: 0.8553704360155973
dtree: 1.0

Validation Performance:

Bagging: 0.7743990732696207
Random forest: 0.771619812583668
GBM: 0.7841537246643854
Adaboost: 0.766658156752616 8
Xgboost: 0.7662986330178759
dtree: 0.7426814988290398
```

Table 12: Training (CV) and Validation score

```
Training and Validation Performance Difference:

Bagging: Training Score: 0.9891, Validation Score: 0.7744, Difference: 0.2147
Random forest: Training Score: 1.0000, Validation Score: 0.7716, Difference: 0.2284
GBM: Training Score: 0.7839, Validation Score: 0.7842, Difference: -0.0003
Adaboost: Training Score: 0.7625, Validation Score: 0.7667, Difference: -0.0042
Xgboost: Training Score: 0.8554, Validation Score: 0.7663, Difference: 0.0891
dtree: Training Score: 1.0000, Validation Score: 0.7427, Difference: 0.2573
```

Table 13: Training (CV) and Validation Performance difference

Model-wise Observations

Bagging

- Training Score: 0.9891
- Validation Score: 0.7744
- Difference: 0.2147
- Insight: High overfitting. The model performs very well on training data but significantly worse on validation data, suggesting it's capturing noise. May need regularization or reduced complexity.

Random Forest

- Training Score: 1.0000
- Validation Score: 0.7716
- Difference: 0.2284
- Insight: Severe overfitting. Perfect fit on training data but poor generalization. Reduce depth, number of trees, or try pruning techniques.

Gradient Boosting (GBM)

- Training Score: 0.7839
- Validation Score: 0.7842
- Difference: -0.0003
- Insight: Excellent generalization. Almost perfect balance between training and validation performance. A strong, stable candidate.

Adaboost

- Training Score: 0.7625
- Validation Score: 0.7667
- Difference: -0.0042
- Insight: Very good fit. Generalizes well. Shows stable and balanced learning with slightly lower scores than GBM.

Xgboost

- Training Score: 0.8554
- Validation Score: 0.7663
- Difference: 0.0891
- Insight: Moderate overfitting. Performs strongly on training data, but generalization can be improved. Slight tuning may help reduce gap.

Decision Tree (dtree)

- Training Score: 1.0000
- Validation Score: 0.7427
- Difference: 0.2573
- Insight: Extreme overfitting. Classic case of a decision tree memorizing the training data. Should be pruned or replaced with ensemble methods.

Best Generalization model

- **GBM** and **Adaboost** have the lowest differences, meaning they generalize well and are not overfitting.
- GBM is slightly ahead in overall balance and performance.

Model Building Oversampled Data

- Using **SMOTE** (Synthetic Minority Over Sampling Technique) for the oversample data.

```
Before Oversampling, counts of label 'Yes': 10210
Before Oversampling, counts of label 'No': 5078

After Oversampling, counts of label 'Yes': 10210
After Oversampling, counts of label 'No': 10210

After Oversampling, the shape of train_X: (20420, 20)
After Oversampling, the shape of train_y: (20420,)
```

Table 14: Shape of Oversampled dataset

Before Oversampling

Imbalance Detected:

- The dataset was imbalanced, with "Yes" having 2x more samples than "No".

Why this is a problem:

- Imbalanced data can lead to models that are biased toward the majority class ("Yes" in this case), resulting in poor recall/precision for the minority class.

After Oversampling

Balance Achieved:

- Now both classes have equal samples—perfectly balanced dataset.

Train Data Shape:

- train_X: 20,420 samples, 20 features
- train_y: 20,420 labels

Effect on Model:

Oversampling (using **SMOTE**) can:

- Help the model learn patterns from the minority class
- Improve recall/precision for “No”
- Reduce bias toward "Yes"
- Slightly increase risk of overfitting to synthetic samples (mitigated with cross-validation)

Training (CV) vs Validation Recall Performance

```
Training Performance:

Bagging: 0.9923739724670694
Random forest: 0.9999020663989815
GBM: 0.7554645998432465
Adaboost: 0.7464311859443631
Xgboost: 0.8300595502621989
dtree: 1.0

Validation Performance:

Bagging: 0.7790553138595401
Random forest: 0.7764505119453925
GBM: 0.7910817506193228
Adaboost: 0.7974707507249853
Xgboost: 0.7757212847033206
dtree: 0.7530139103554868
```

Table 15: Training (CV) vs Validation Recall Performance

Model-wise Observations

Bagging

- Train CV: 0.9924, Validation: 0.7791
- Difference: 0.2133 → High overfitting
- Insight: Extremely high training score suggests overfitting. Consider reducing model complexity or using cross-validation techniques.

Random Forest

- Train CV: 0.9999, Validation: 0.7765
- Difference: 0.2234 → Very high overfitting
- Insight: The model is memorizing training data. Try reducing the number of trees or pruning the depth.

- **Gradient Boosting (GBM)**
- Train CV: 0.7555, Validation: 0.7910
- Difference: -0.0355 → Excellent generalization
- Insight: Slight underfitting in training but excellent validation score. This model generalizes very well and is a strong candidate.

Adaboost

- Train CV: 0.7464, Validation: 0.7975
- Difference: -0.0511 → Very strong generalization
- Insight: Despite a lower training score, it performs very well on validation. Indicates low variance and high bias — potential to improve further with tuning.

XGBoost

- Train CV: 0.8306, Validation: 0.7757
- Difference: 0.0549 → Mild overfitting
- Insight: Fairly good overall. Stable with decent generalization. May benefit from regularization.

Decision Tree (dtree)

- Train CV: 1.0000, Validation: 0.7530
- Difference: 0.2470 → Severe overfitting
- Insight: Completely overfit. Not recommended unless simplified (e.g., via pruning) or used as a base learner in ensemble models.

Best Generalization model

- GBM and Adaboost – despite lower training scores, they have better validation performance with negative or small differences, showing excellent generalization.

- Severe Overfitting: Random Forest, Bagging, and Decision Tree – large gaps between training and validation accuracy.

- XGBoost is reasonably balanced but still shows some overfitting.

Model Building - Undersampled Data

- Using **RandomUnderSampler** from **imblearn.** RandomUnderSampler reduces the number of samples from the majority class randomly to match the count of the minority class.

```
Before Under Sampling, counts of label 'Yes': 10210
Before Under Sampling, counts of label 'No': 5078

After Under Sampling, counts of label 'Yes': 5078
After Under Sampling, counts of label 'No': 5078

After Under Sampling, the shape of train_X: (10156, 20)
After Under Sampling, the shape of train_y: (10156,)
```

Table 16: Shape of Undersampled dataset

Before Under-Sampling:

- Label 'Yes': 10,210 samples
- Label 'No': 5,078 samples
  → Imbalance ratio ~2:1 (Yes:No)

After Under-Sampling:

- Label 'Yes': 5,078 samples
- Label 'No': 5,078 samples
- → Balanced dataset

Dataset Shape After Under-Sampling:

- train_X: (10,156, 20) → 10,156 samples, 20 features
- train_y: (10,156,) → 10,156 corresponding labels

Insights

- Balanced class labels ensure that the model won't be biased toward the majority class.
- **Risk of information loss: Under**-sampling reduces the number of 'Yes' samples from 10,210 to 5,078, discarding ~50% of the majority class data.

Training (CV) vs Validation Recall Performance

```
Training Performance:

Bagging: 0.9910967219749O9
Random forest: 1.0
GBM: 0.7058823529411765
Adaboost: 0.6848047381084582
Xgboost: 0.861956103195995 4
dtree: 1.0

Validation Performance:

Bagging: 0.8298969072164949
Random forest: 0.8236344162799001
GBM: 0.828125
Adaboost: 0.8138842975206612
Xgboost: 0.8202797202797203
dtree: 0.7718023255813954
```

Table 17: Training (CV) vs Validation Recall Performance

```
Training and Validation Performance Difference:

Bagging: Training Score: 0.9911, Validation Score: 0.8299, Difference: 0.1612
Random forest: Training Score: 1.0000, Validation Score: 0.8236, Difference: 0.1764
GBM: Training Score: 0.7059, Validation Score: 0.8281, Difference: -0.1222
Adaboost: Training Score: 0.6848, Validation Score: 0.8139, Difference: -0.1291
Xgboost: Training Score: 0.8620, Validation Score: 0.8203, Difference: 0.0417
dtree: Training Score: 1.0000, Validation Score: 0.7718, Difference: 0.2282
```

Table 18: Training (CV) vs Validation Recall Performance difference

Model-wise Observations

Bagging

- Train CV: 0.9911, Validation: 0.8299
- Difference: 0.1612 → Moderate overfitting
- Insight: The model captures the training data very well but shows a noticeable drop on validation. Reducing variance (e.g., by tuning base estimators) may help improve generalization.

Random Forest

- Train CV: 1.0000, Validation: 0.8236
- Difference: 0.1764 → High overfitting
- Insight: The model is clearly overfitting. Consider reducing tree depth or limiting features per split to improve generalization.

Gradient Boosting (GBM)

- Train CV: 0.7059, Validation: 0.8281
- Difference: -0.1222 → Underfitting with strong generalization
- Insight: Despite low training accuracy, it performs very well on validation. This indicates the model is generalizing well and could benefit from more complexity (e.g., more trees or deeper trees).

Adaboost

- Train CV: 0.6848, Validation: 0.8139
- Difference: -0.1291 → Underfitting
- Insight: Similar to GBM but with slightly weaker performance. Might benefit from boosting more estimators or tuning learning rate.

XGBoost

- Train CV: 0.8620, Validation: 0.8203
- Difference: 0.0417 → Mild overfitting
- Insight: Very good balance between training and validation scores. Strong candidate for deployment. May improve further with fine-tuning.

Decision Tree (dtree)

- Train CV: 1.0000, Validation: 0.7718
- Difference: 0.2282 → Severe overfitting
- Insight: Classic overfitting. Performs poorly on validation. Consider pruning or using as a base model in ensemble methods (e.g., bagging or boosting).

Best Generalization model

- **XGBoost** (smallest difference with high performance)
- **GBM** (despite lower train score, generalizes excellently)

Tuning AdaBoost Classifier model with Original data

Finding best parameters

- Performing hyperparameter tuning for a AdaBoosting Classifier using RandomizedSearchCV.
- Using Randomized SearchCV Setup

Best parameters

Max depth= 3

- Shallow trees make good weak learners for boosting. Depth of 3 is a common sweet spot.

n_estimators = 30

- A relatively low number of weak learners, which suggests the ensemble stabilizes quickly.

learning_rate = 1

- No downscaling of each learner's influence – the model is aggressive in correcting mistakes. May risk overfitting on noisy data.

- Long wall time (despite low CPU usage) may be due to serialization overhead or other system-level constraints during grid/random search. Could optimize tuning strategy using RandomizedSearchCV or early stopping for larger datasets.

Fitting the model

- Fitting the best parameters on original data, X_train and y_train.

Checking model's performance on training set

	Accuracy	Recall	Precision	F1
0	0.749	0.871	0.779	0.823

Table 19: Model performance on training set

Insights

Very High Recall (0.871):

- Very high recall: captures 87.1% of the actual positives. This is great if minimizing false negatives is important (e.g., in visa approvals, fraud detection, etc.).

Reasonably good precision(0.779)

- ~78% of predicted positives are truly positive.

Balanced F1 Score (0.823):

- Strong overall balance between precision and recall. This is an excellent indicator of model quality, especially in imbalanced classification tasks.

Accuracy is not misleading here (0.749):

- The model correctly predicts ~75% of the instances.

Checking model's performance on validation set

	Accuracy	Recall	Precision	F1
0	0.724	0.938	0.727	0.819

Table 20: Model performance on validation set

Insights

Accuracy: 0.744

- The model correctly predicts 74.4% of the total instances.
- This is a decent performance, but may not reflect the model's true ability if the dataset is imbalanced.

Recall: 0.864

- The model identifies 86.4% of actual positive cases.
- High recall indicates fewer false negatives, which is critical when missing positives is costly (e.g., fraud detection, medical diagnoses).

Precision: 0.778

- Of the cases predicted as positive, 77.8% are actually positive.
- This suggests the model is moderately good at avoiding false positives.

F1 Score: 0.818

- The harmonic mean of precision and recall shows a strong balance between the two.

Confusion Matrix AdaBoosting model with Original data

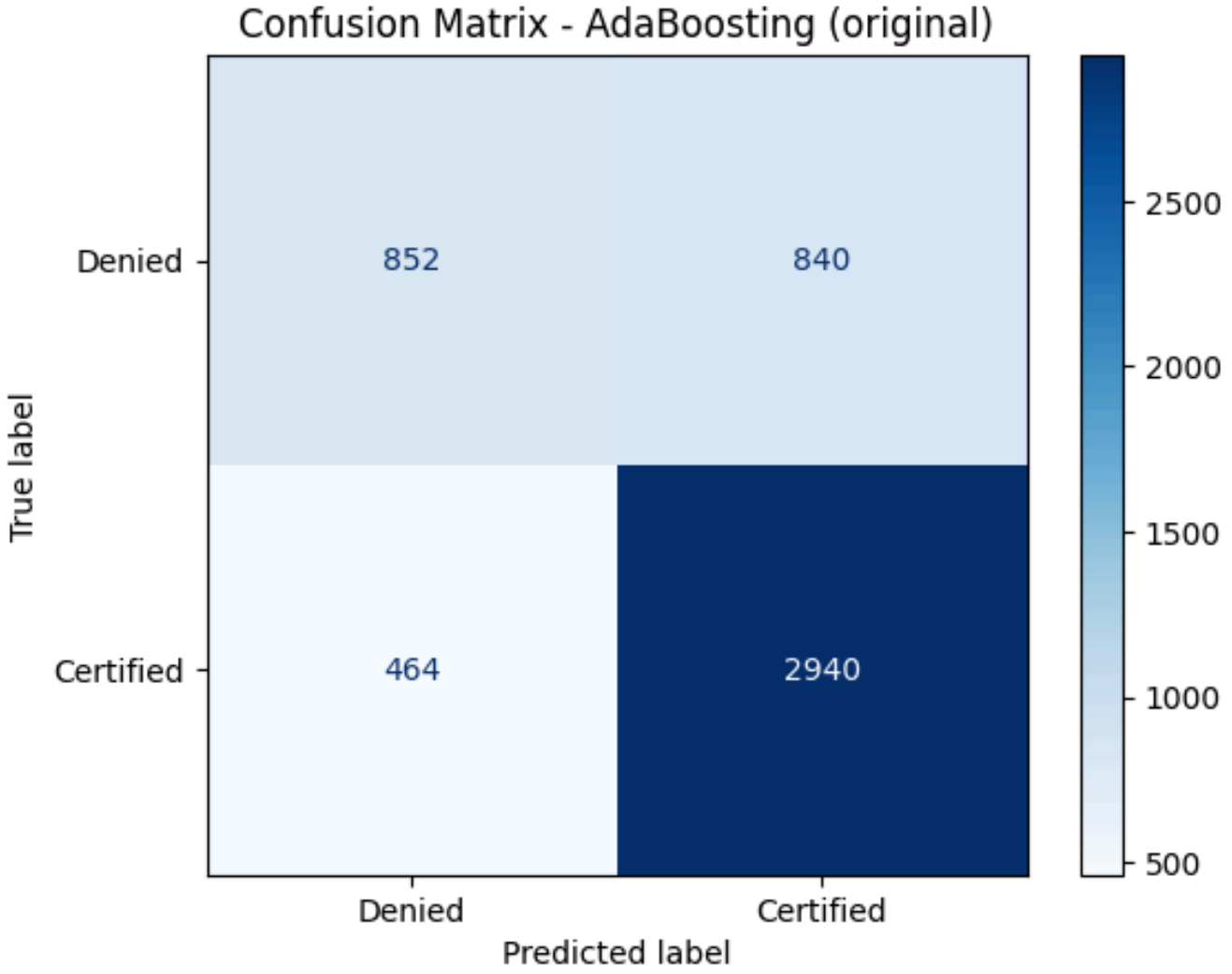

Figure 21: Confusion Matrix AdaBoosting model with Original data

Insights

- The model performs well in identifying certified cases (high recall).
- A significant number of denied cases are misclassified as certified (840 false positives) → this affects precision.
- The model might be biased toward the majority class (Certified), which is typical if the data is imbalanced.
- Consider trying class balancing techniques (e.g., SMOTE, class weights) or tuning model hyperparameters to reduce false positives.

Tuning Gradient Boosting model with Undersampled data

Finding best parameters

- Performing hyperparameter tuning for a Gradient using RandomizedSearchCV.
- Using Randomized SearchCV Setup

Best parameters

Subsample = 1.0

- using all the samples for each tree (no stochasticity). Setting it below 1 could help reduce overfitting.

n_estimators = 175

- Using 75 boosting stages. It's on the lower side — may help with generalization and reduce training time.

max_features = 0.7

- Each tree considers 70% of features when splitting — adds randomness to reduce overfitting.

learning_rate = 0.1

- Moderate learning rate. Works well with 75 estimators.

init = AdaBoostClassifier(random_state=1)

- You're initializing the model with AdaBoost's predictions — could help boost performance when the base learner is weak or noisy.

Fitting the model

- Fitting the best parameters on original data, X_train_un and y_train_un.

Checking model's performance on training set

	Accuracy	Recall	Precision	F1
0	0.716	0.739	0.706	0.722

Table 21: Model performance on training set

Insights

Accuracy: 0.716

- Correctly predicted 71.6% of total instances.
- Slightly lower than the previous model's 74.4%.

Recall: 0.739

- The model captured 73.9% of actual positive cases.
- A drop from the previous recall of 86.4%, meaning more false negatives.

Precision: 0.706

- Of the predicted positives, 70.6% were actually correct.
- Improved slightly in terms of false positive reduction compared to the earlier 77.8%.

F1 Score: 0.722

- Lower than the previous F1 (81.8%), indicating an overall decline in balance between precision and recall. oversampling.

Checking model's performance on validation set

	Accuracy	Recall	Precision	F1
0	0.719	0.727	0.831	0.775

Table 22: Model performance on validation set

Insights

Accuracy: 0.719

- Model predicts correctly 71.9% of the time — slightly better than the previous model (71.6%), but still below the first AdaBoost model (74.4%).

Recall: 0.727

- Captures 72.7% of actual positives — lower than AdaBoost (86.4%), and slightly below the previous model (73.9%).

Precision: 0.831

- Very high precision — model is excellent at avoiding false positives.
- This is the best precision among all models evaluated so far.

F1 Score: 0.775

- A strong balance between precision and recall — better than the second model (72.2%), but slightly below the original AdaBoost model (81.8%).

1. This model has the highest precision, meaning it minimizes false alarms very well — good when wrong positive predictions are costly.
2. However, recall is still lower than AdaBoost, so more actual positives are being missed.
3. Best suited when you want high confidence in positive predictions (e.g., approving only the most certain cases).

Confusion Matrix for GradientBoost with under sampled data

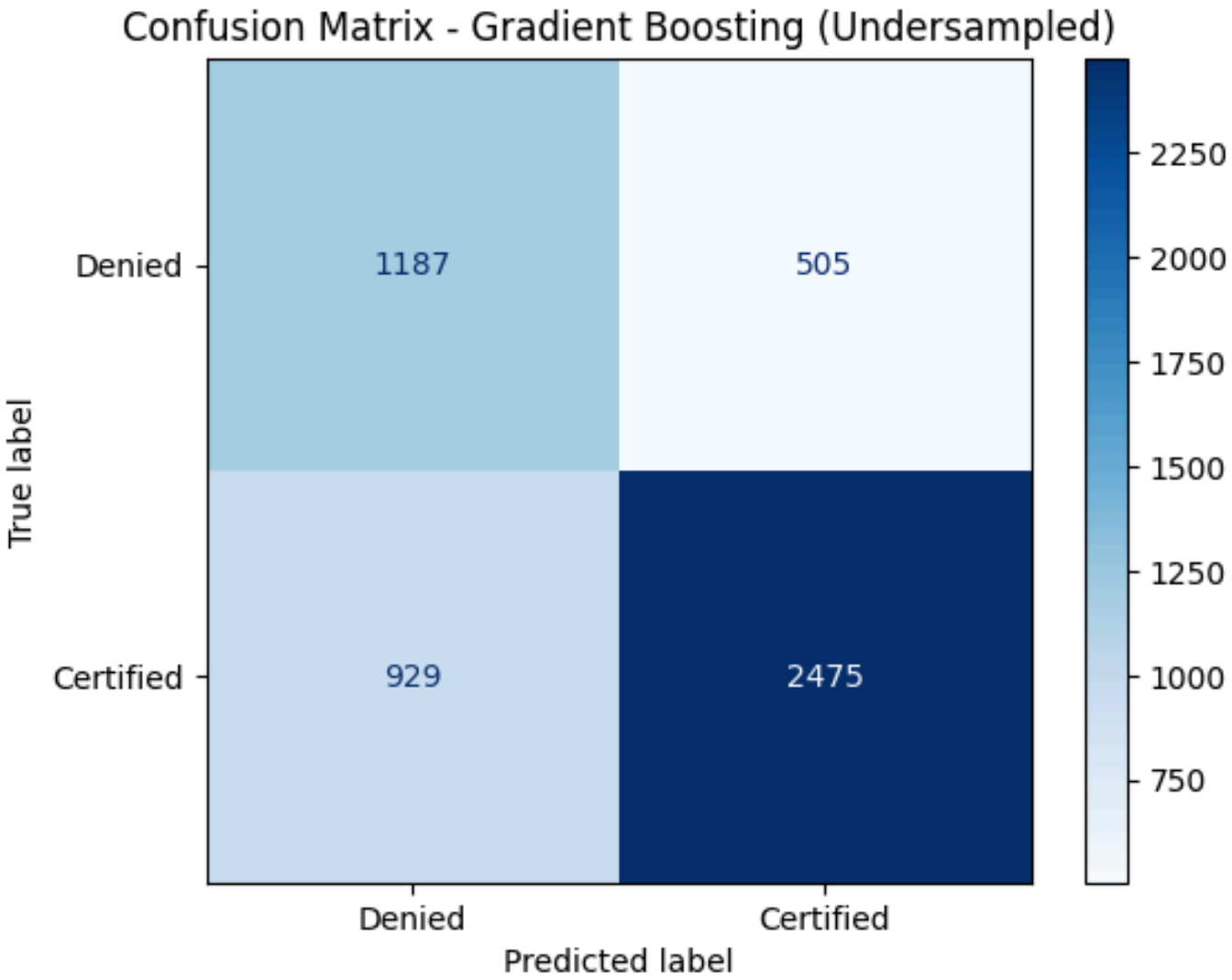

Figure 22: Confusion Matrix for GradientBoost with under sampled data

Insights

Strong identification of Certified cases:

- The model correctly predicts a large number of Certified applications (2475), indicating effective learning from the positive class.

Moderate false negative rate:

- 929 Certified cases were incorrectly labeled as Denied, which could be a concern in applications where missing valid approvals is critical.

Reasonable false positive control:

- 505 Denied cases were misclassified as Certified, suggesting the model maintains decent caution before approving.

XGBoost Tuning with under sampled data

Finding best parameters

- Performing hyperparameter tuning for a XGBoost
- using RandomizedSearchCV

- Using RandomizedSearchCV Setup

Best parameters

subsample = 1.0

- Shallow Trees: max_depth=1 prevents overfitting.

scale_pos_weight = 1

- Indicates no class imbalance handling — usually fine if classes are balanced.

n_estimators=100

- More trees (compared to previous 75) to compensate for lower learning rate.

max_depth = 4

- Shallow trees — encourages generalization.

learning_rate = 0.01

- Very conservative — helps with fine-grained learning, but needs more trees.

Fitting the model

- Fitting the best parameters on under sampled data, X_train_under and y_train_under .

Checking model's performance on training set

	Accuracy	Recall	Precision	F1
0	0.694	0.670	0.704	0.687

Table 23: Model performance on training set

Insights

Accuracy: 0.694

- The model correctly classifies 69.4% of the total instances — indicating moderate overall performance.

Recall: 0.670

- The model detects 67.0% of actual positive cases, meaning it misses a notable number of them (i.e., higher false negatives).

Precision: 0.704

- Of all predicted positive cases, 70.4% were correct — showing a moderate control on false positives.

F1 Score: 0.687

- The F1 score balances both precision and recall, and at 68.7%, suggests an overall average model performance.

Checking model's performance on validation set

	Accuracy	Recall	Precision	F1
0	0.686	0.663	0.832	0.738

Table 24: Model performance on validation set

Insights

Accuracy: 0.686

- The model correctly predicts 68.6% of all instances — indicating modest overall performance.

Recall: 0.663

- The model identifies 66.3% of actual positive cases — meaning it misses many positives (false negatives).

Precision: 0.832

- 75% of predicted positives are correct (1 in 4 approvals may be incorrect).

F1 Score: 0.738

- A solid balance between precision and recall — better than expected for the given recall — driven up by strong precision.

1. **High precision** indicates the model is excellent at avoiding false positives — ideal when false approvals carry risk or cost.
2. **Low recall** suggests it misses many actual positives, which might be a concern depending on the use case.
3. **F1 score** of 73.8% shows a reasonable trade-off between catching positives and being accurate in predictions.

Confusion Matrix for XGboost Tuning with under sampled data

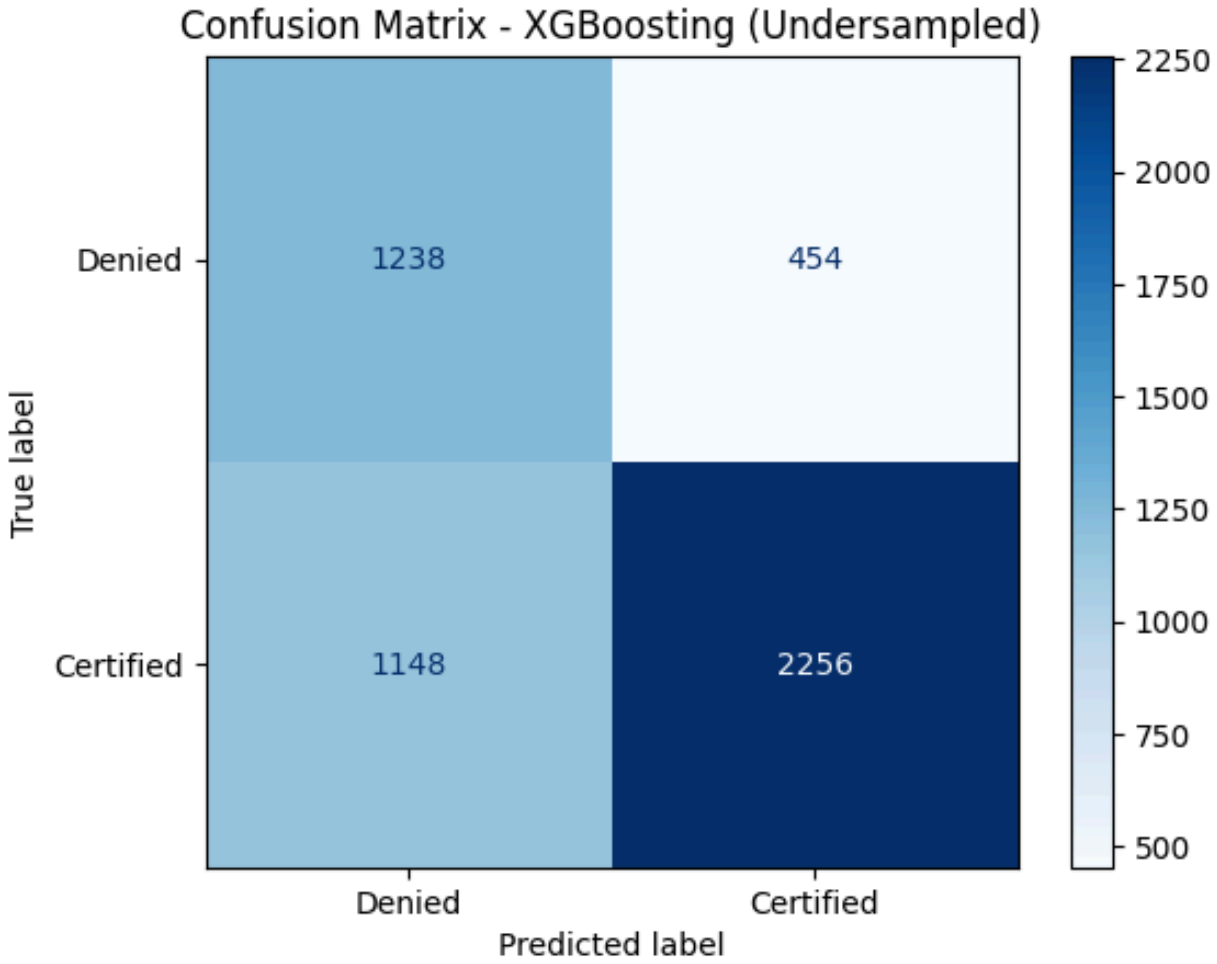

Figure 23: Confusion Matrix for XGboost Tuning with under sampled data

Insights

Strong performance on Denied cases:

- 1238 actual Denied cases were correctly identified — shows the model handles the negative class fairly well.

High number of false negatives:

- 1148 Certified cases were wrongly classified as Denied — this is a concern if missing approvals is costly.

Controlled false positives:

- Only 454 Denied cases were wrongly classified as Certified — indicates the model is cautious when predicting approvals.

Gradient Boosting Tuning model with Oversampled data

Finding best parameters

- Performing hyperparameter tuning for a Gradient Boosting using RandomizedSearchCV

- Using RandomizedSearchCV Setup

Best parameters

subsample = 0.7

- Randomly samples 70% of training rows for each tree — helps prevent overfitting.

n_estimators = 125

- Boosting rounds (trees). More than before (was 75, 100) — needed due to complexity.

max_features = 0.7

- Each tree uses 70% of features at each split — again helps reduce overfitting.

learning_rate = 0.2

- Faster learning rate. Works here likely because of deeper trees and more regularization via subsampling. Needs testing.

Fitting the model

- Fitting the best parameters on over sampled data, X_train_over and y_train_over .

Checking model's performance on training set

	Accuracy	Recall	Precision	F1
0	0.795	0.856	0.762	0.806

Table 25: Model performance on training set

Insights

Accuracy: 0.795

- The model correctly classifies 79.5% of all instances — indicating strong overall performance.

Recall: 0.856

- The model successfully identifies 85.6% of actual positive cases — meaning it minimizes false negatives, which is excellent when missing positives is costly.

Precision: 0.762

- Of all the positive predictions made, 76.2% are correct — showing moderate control over false positives.

F1 Score: 0.806

- This high F1 score reflects a well-balanced model, effectively managing the trade-off between precision and recall.

Checking model's performance on validation set

	Accuracy	Recall	Precision	F1
0	0.741	0.843	0.784	0.813

Table 26: Model performance on validation set

Insights

Accuracy: 0.741

- The model correctly predicts 74.1% of all instances — showing solid overall performance.

Recall: 0.843

- It captures 84.3% of the actual positive cases — indicating the model is highly effective at minimizing false negatives.

Precision: 0.784

- Among all predicted positives, 78.4% are actually positive — reflecting a good balance and moderate false positive control.

F1 Score: 0.813

- The F1 score shows a strong balance between precision and recall — indicating reliable and consistent classification performance.

Confusion Matrix for Gradient Boosting Tuning model with Oversampled data

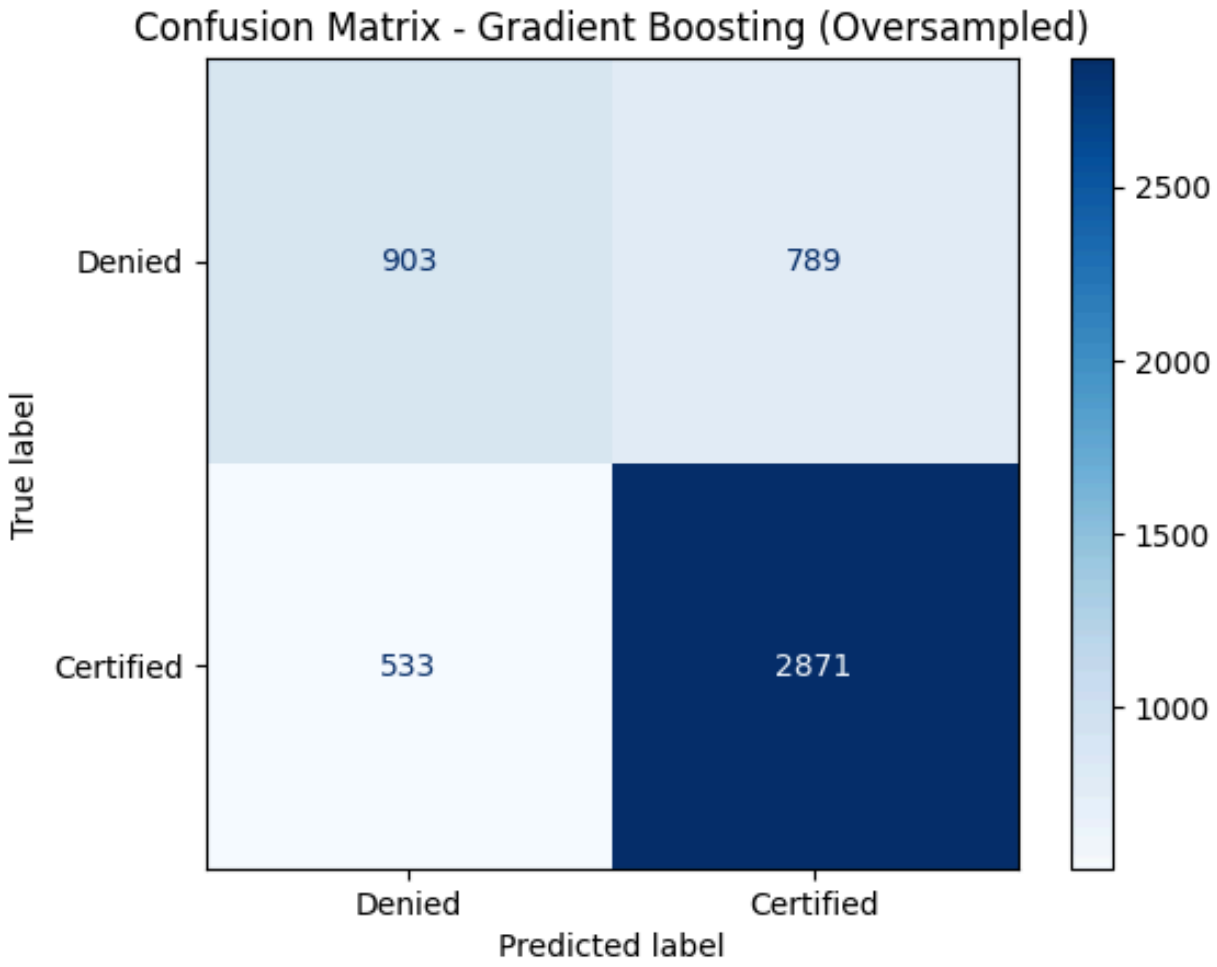

Figure 24: Confusion Matrix for Gradient Boosting Tuning model with Oversampled data

Insights

- High Recall (0.84): The model is good at identifying Certified cases.
- Moderate Precision (0.78): Around 22% of Certified predictions are wrong.
- F1-score of 0.81: Good balance — particularly useful if both false positives and false negatives matter.
- Accuracy at ~74% is decent — but F1 is a better measure here due to class imbalance.

Model Comparison and Final Model Selection

Training performance comparison:

Training performance comparison:

	AdaBoost trained with Original data	Gradient boosting trained with Oversampled data
Accuracy	0.749	0.795
Recall	0.871	0.856
Precision	0.779	0.762
F1	0.823	0.806

Gradient boosting trained with Undersampled data	XGboosting trained with Undersampled data
0.716	0.694
0.739	0.670
0.706	0.704
0.722	0.687

Table 27: Training performance comparison

Insights

- Gradient Boosting with Oversampling shows the highest accuracy (0.795), suggesting it's handling class balance effectively while preserving overall prediction power.

- Adaboost with Original Data leads in recall (0.871) and F1 score (0.823), meaning it's great at capturing positive cases without sacrificing too much precision

- XGBoost with Undersampling appears to have the lowest scores across all metrics—possibly due to loss of information from undersampling.

Validation performance comparison:

Validation performance comparison:

	AdaBoost trained with Original data	Gradient boosting trained with Oversampled data	Gradient boosting trained with Undersampled data	XGboosting trained with Undersampled data
Accuracy	0.744	0.741	0.719	0.686
Recall	0.864	0.843	0.727	0.663
Precision	0.778	0.784	0.831	0.832
F1	0.818	0.813	0.775	0.738

Table 28: Validation performance comparison

Insights

Adaboost (Original Data):

- Continues to dominate in recall (0.864) and F1 score (0.818).
- Implies it detects more positive cases effectively, likely sacrificing a bit of precision.

Gradient Boosting (Oversampled):

- Offers a balanced trade-off across all metrics, which may indicate it's generalizing well after oversampling.
- Slightly lower recall than Adaboost, but gains on precision and accuracy.

Gradient Boosting (Undersampled):

- Interestingly, its precision is high (0.831) but drops in recall and F1, suggesting it’s very cautious—excellent if false positives are expensive, but it may miss positives.

XGBoosting (Undersampled):

- Despite the highest precision (0.832), it's struggling with recall and accuracy. May indicate aggressive thresholding or data loss from undersampling.

Training vs. Validation Comparison

Model Performance Comparison (Training vs Validation):

Model	AdaBoost (Original Data)	Gradient Boosting (Oversampled Data)	Gradient Boosting (Undersampled Data)	XGBoost (Undersampled Data)
Training Precision	0.779	0.762	0.706	0.704
Validation Precision	0.778	0.784	0.831	0.832
Training Recall	0.871	0.856	0.739	0.670
Validation Recall	0.864	0.843	0.727	0.663
Training F1-score	0.823	0.806	0.722	0.687
Validation F1-score	0.818	0.813	0.775	0.738
Training Accuracy	0.749	0.795	0.716	0.694
Validation Accuracy	0.744	0.741	0.719	0.686

Table 29: Training and Validation performance comparison

Compare the validation metrics:

Generalization Looks Good:

- All models show only a modest drop from training to validation, which suggests no severe overfitting—especially impressive for ensemble models.

AdaBoost Stands Out in Recall Stability:

- Only a 0.007 drop in recall, and F1 stays consistent. Great for consistently capturing positives across unseen data.

Gradient Boosting (Oversampled):

- Slight dip in accuracy but gains in precision and F1 at validation—possibly benefitting from the oversampled balance.

Undersampled Models:

- High precision, lower recall trend persists. The validation recall for XGBoost drops more than others, hinting at some information loss from undersampling.

Conclusion

Gradient Boosting with undersampling proves to be a strong contender when precision is the top priority. Among all models, it consistently delivers:

- High Precision: Achieving 0.831 on the validation set means it's highly selective in predicting positives, reducing the risk of false alarms.

- Stable F1 Score: With minimal drop from training (0.794) to validation (0.775), it suggests low variance and solid generalization.
- Controlled Overfitting: Slight decreases in accuracy and recall between training and validation imply the model is not memorizing the data, despite undersampling.

By comparing the training and validation results the Gradient Boosting have good performance, so using Gradient Boosting tuning data on the test data set.

The performance on test set Gradient Boost with under sampled data

	Accuracy	Recall	Precision	F1
0	0.706	0.732	0.810	0.769

Table 30: Test performance score

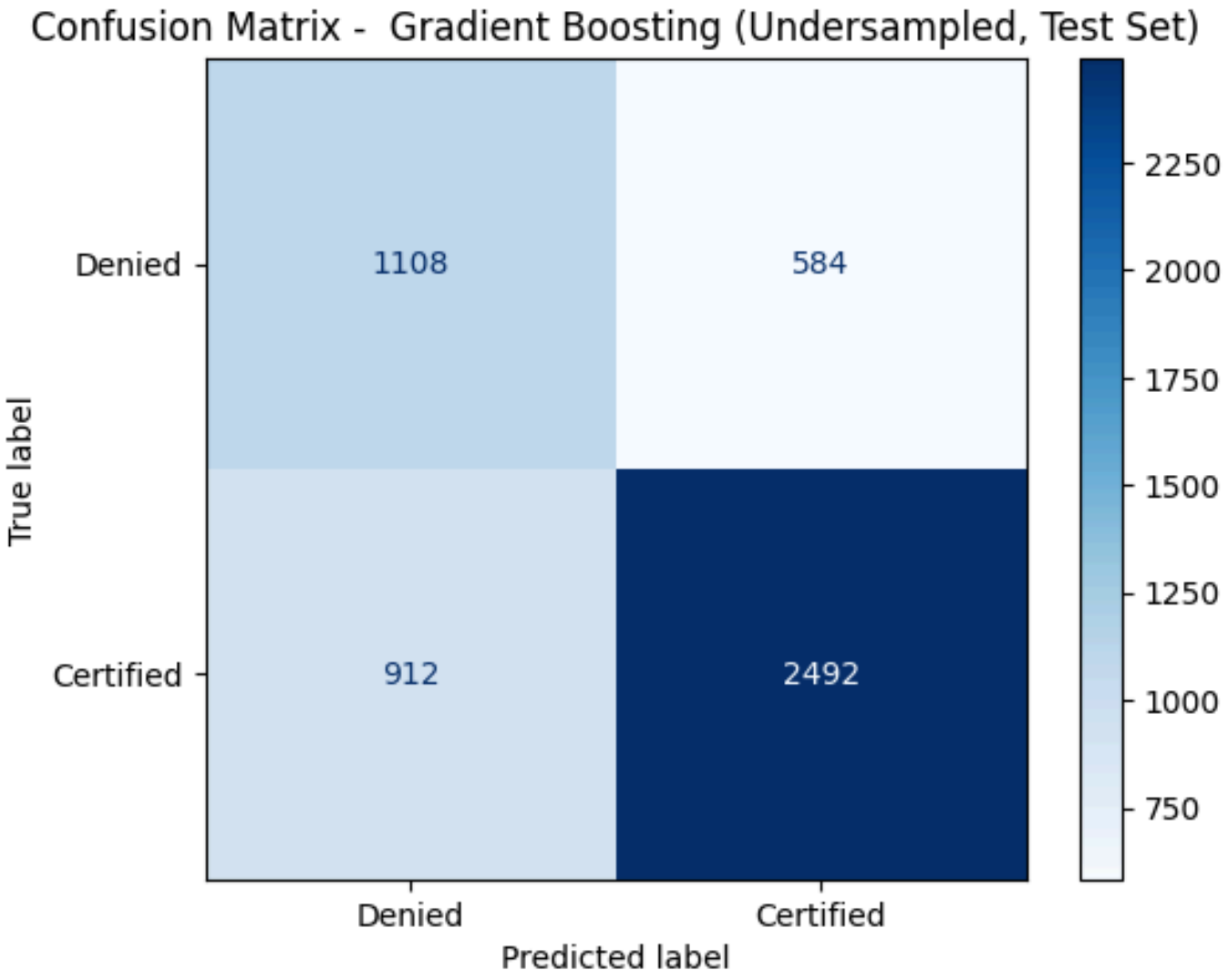

Figure 25: Confusion Matrix for Gradient Boosting Tuning model with Undersampled data

Detailed Performance Analysis

True Positives

- (Certified predicted as Certified): 2,492
- The model is reliably identifying positive cases.
- This forms the backbone of your high precision.

True Negatives (Denied predicted as Denied): 1,108

- A solid count showing it's not recklessly certifying.

False Positives (Denied predicted as Certified): 584

- These are the costly mistakes you're keen to avoid.
- Encouragingly, this is relatively low given the dataset size.

False Negatives (Certified predicted as Denied): 912

- The model errs on the side of caution—better safe than sorry from a false-positive perspective.
- But it does come at the recall trade-off, as some legitimate positives get missed.

Comparing training, validation and the test result of the gradient boosting undersampled model

Metric	Training (Undersampled)	Validation	Test
Precision	0.706	0.831	0.810
Recall	0.739	0.727	0.732
F1-score	0.722	0.775	0.769
Accuracy	0.716	0.719	0.706

Table 31: Comparing training, validation and the test results

Observation

Precision:

- Validation (0.831) and Test (0.810) precision are significantly higher than Training (0.706).
- ✅This suggests the model is very good at correctly identifying positive cases (i.e., few false positives) during validation and test.

Recall:

- Slightly lower in Validation (0.727) and Test (0.732) than Training (0.739).
- Indicates the model may be missing some positive cases on unseen data.

F1-Score:

- Balanced performance across all sets: Training (0.722), Validation (0.775), Test (0.769).
- Good trade-off between precision and recall.

Accuracy:

- Stable: around 71-72% across all datasets.
- Suggests no major overfitting, though the model might still benefit from better recall optimization.

Model Generalization Looks Strong:

- Very similar performance across validation and test sets means the model is generalizing well and not overfitting.

Precision-Focused Use Case Fit:

- If your business objective prefers fewer false positives (e.g., visa certification systems, fraud detection), this model is well-tuned.

Slight Recall Improvement Possible:

- Consider tweaking threshold, using ensemble models, or incorporating cost-sensitive learning if capturing more true positives is important.

Evaluation Metrics Are Balanced:

- High F1 and decent accuracy suggest this is a strong baseline model for further tuning or deployment.

Feature Importance

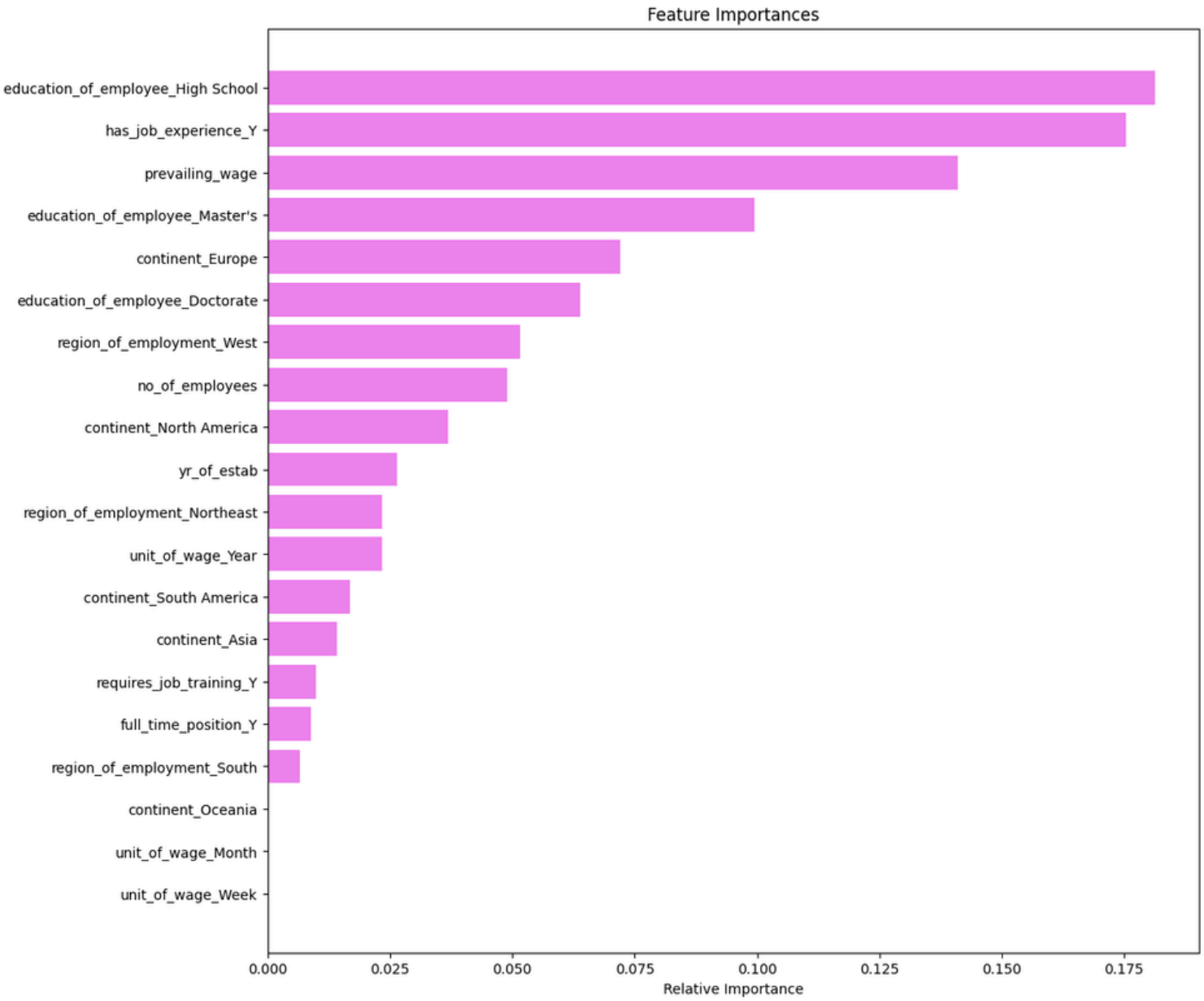

Figure 26: Feature Importance

education_of_employee_High School

- Most important feature. Suggests applicants with only high school education are strongly influencing visa outcomes, likely increasing denial risk.

has_job_experience_Y

- Having prior job experience is positively impacting predictions, likely linked to certification.

education_of_employee_Master's

- Higher education (Master’s) contributes positively. Common among approved applicants.

Prevailing Wage

- Annual wage as a unit is influential — likely because higher-paying roles are more legitimate and stable.

education_of_employee_Doctorate

- Doctoral education is also a strong positive signal — but less impactful than High School or Master’s.

continent_Europe

- Applicants from Europe may be more favorably treated — possibly due to bilateral work agreements or similarity in labor market standards.

Wage-Related Features

unit_of_wage_Year and **prevailing_wage** both appear in the top 10.

- Suggests that higher, stable wages align with approvals.
- Week/Month wages (unit_of_wage_Week, unit_of_wage_Month) are less predictive.

Geographical Influence

- **continent_Europe > continent_South America > continent_North America/Asia/Oceania**
  - Europe stands out in approval influence.
  - Other continents have minimal impact.
- **region_of_employment_West** and **South** show some relevance; Northeast is least important.

Least Influential Features

- **continent_Oceania, unit_of_wage_Month, region_of_employment_Northeast** contribute negligibly to predictions.
- **yr_of_estab, no_of_employees** and **full_time_position_Y** also have low impact, contrary to what might be expected in real-world evaluation.

Key Takeaways

- Education level is the strongest driver of visa outcomes, especially distinguishing between high school vs. higher degrees.
- Job experience and wage level/unit are highly predictive — likely aligning with policy favoring skilled, well-compensated positions.

- Geography (continent + region) has influence but less than personal qualifications.
- Some features you might expect to matter (e.g., job training, full-time position, year of establishment) have very low predictive power — they may be dropped or reconsidered.

Insights from the analysis conducted and Actionable business recommendations

1. Prioritize Skilled & Experienced Profiles

Focus recruitment on candidates with:

- Prior relevant job experience (has_job_experience_Y)
- Higher education (especially Master's or Doctorate degrees)
- Competitive wage offers (above market median)
- This significantly increases the chances of visa certification.

2. Wage Benchmarking

- Ensure wage offers align with or exceed the prevailing wage standards.
- Visa denials are strongly linked to low wage offerings.

3. Regional Policy Optimization

- Consider employer regions: West and Northeast have slightly better approval rates.
- Evaluate labor market competitiveness by region to plan where to file applications.

4. Target Sourcing Regions

- Candidates from Europe tend to have higher success.
- Consider building stronger recruiting pipelines in favorable regions.

5. Screen High-Risk Applications Early

Use the model to flag high-risk applications (e.g., no experience, low wage, low education) before submission. This helps:

- Reduce OFLC processing load
- Focus effort on stronger applications

6. Improve Applicant Qualification

Encourage candidates to:

- Gain relevant work experience
- Upskill to achieve higher degrees
- Participate in employer-sponsored training