Machine Learning-2
Business report

Context:

Business communities in the United States are facing high demand for human resources, but one of the constant challenges is identifying and attracting the right talent, which is perhaps the most important element in remaining competitive. Companies in the United States look for hard-working, talented, and qualified individuals both locally as well as abroad.

The Immigration and Nationality Act (INA) of the US permits foreign workers to come to the United States to work on either a temporary or permanent basis. The act also protects US workers against adverse impacts on their wages or working conditions by ensuring US employers' compliance with statutory requirements when they hire foreign workers to fill workforce shortages. The immigration programs are administered by the Office of Foreign Labor Certification (OFLC).

OFLC processes job certification applications for employers seeking to bring foreign workers into the United States and grants certifications in those cases where employers can demonstrate that there are not sufficient US workers available to perform the work at wages that meet or exceed the wage paid for the occupation in the area of intended employment.

Contents

Objective

In FY 2016, the OFLC processed 775,979 employer applications for 1,699,957 positions for temporary and permanent labor certifications. This was a nine percent increase in the overall number of processed applications from the previous year. The process of reviewing every case is becoming a tedious task as the number of applicants is increasing every year.

The increasing number of applicants every year calls for a Machine Learning based solution that can help in shortlisting the candidates having higher chances of VISA approval. OFLC has hired the firm EasyVisa for data-driven solutions. You as a data scientist at EasyVisa have to analyze the data provided and, with the help of a classification model:

1. Facilitate the process of visa approvals.
2. Recommend a suitable profile for the applicants for whom the visa should be certified or denied based on the drivers that significantly influence the case status.

Data Dictionary:

- case_id: ID of each visa application
- continent: Information of continent the employee
- education_of_employee: Information of education of the employee
- has_job_experience: Does the employee has any job experience? Y= Yes; N = No
- requires_job_training: Does the employee require any job training? Y = Yes; N = No
- no_of_employees: Number of employees in the employer's company
- yr_of_estab: Year in which the employer's company was established
- region_of_employment: Information of foreign worker's intended region of employment in the US.
- prevailing_wage: Average wage paid to similarly employed workers in a specific occupation in the area of intended employment. The purpose of the prevailing wage is to ensure that the foreign worker is not underpaid compared to other workers offering the same or similar service in the same area of employment.
- unit_of_wage: Unit of prevailing wage. Values include Hourly, Weekly, Monthly, and Yearly.
- full_time_position: Is the position of work full-time? Y = Full-Time Position; N = Part-Time Position
- case_status: Flag indicating if the Visa was certified or denied

Rubric:

Exploratory Data Analysis

- Problem definition - Univariate analysis - Bivariate analysis - Use appropriate visualizations to identify the patterns and insights - Key meaningful observations on individual variables and the relationship between variables

Data preprocessing

- Prepare the data for analysis - Feature Engineering - Missing value Treatment - Outlier Treatment - Ensure no data leakage among train-test and validation sets

Model Building - Original Data

- Choose the appropriate metric for model evaluation - Build 5 models (from decision trees, bagging and boosting methods) - Comment on the model performance * You can choose NOT to build XGBoost if you are facing issues with the installation

Model Building - Oversampled Data

- Oversample the train data - Build 5 models (from decision trees, bagging and boosting methods) - Comment on the model performance * You can choose NOT to build XGBoost if you are facing issues with the installation

Model Building - Undersampled Data

- Undersample the train data - Build 5 models (from decision trees, bagging and boosting methods) - Comment on the model performance * You can choose NOT to build XGBoost if you are facing issues with the installation

Model Performance Improvement using Hyperparameter Tuning

- Choose 3 models (at least) that might perform better after tuning with proper reasoning - Tune the 3 models (at least) obtained above using randomized search and metric of interest - Comment on the performance of 3 tuned models * You can choose NOT to tune XGBoost if you experience long runtimes

Model Performance Comparison and Final Model Selection

- Compare the performance of tuned models - Choose the best model - Comment on the performance of the best model on the test set

Actionable Insights & Recommendations

- Write down insights from the analysis conducted - Provide actionable business recommendations

Business Report Quality

- Adhere to the business report checklist

Exploratory Data Analysis:

– The Name of the dataset is “EasyVisa.csv”.

– The Dataset contain a total of 25480 rows and 12 columns.

– There are no Duplicates present in the Data.

– There are no missing values present in the dataset.

– There some amount of outliers present in the dataset. In no_of_employees, yr_of_estab, prevailing_wage. But those outliers seem to be legit and we can leave it as it is.

– There are 9 Categorical Columns ‘case_id', 'continent', 'education_of_employee', 'has_job_experience', 'requires_job_training', 'region_of_employment', 'unit_of_wage', 'full_time_position', 'case_status'.

– And there are 3 Numerical column: no_of_employees, yr_of_estab, prevailing_wage.

– The Datatypes present are :

- Int64.
- Object.
- Float64.

Univariate Analysis:

Numerical Variables:

1. **case_id:**

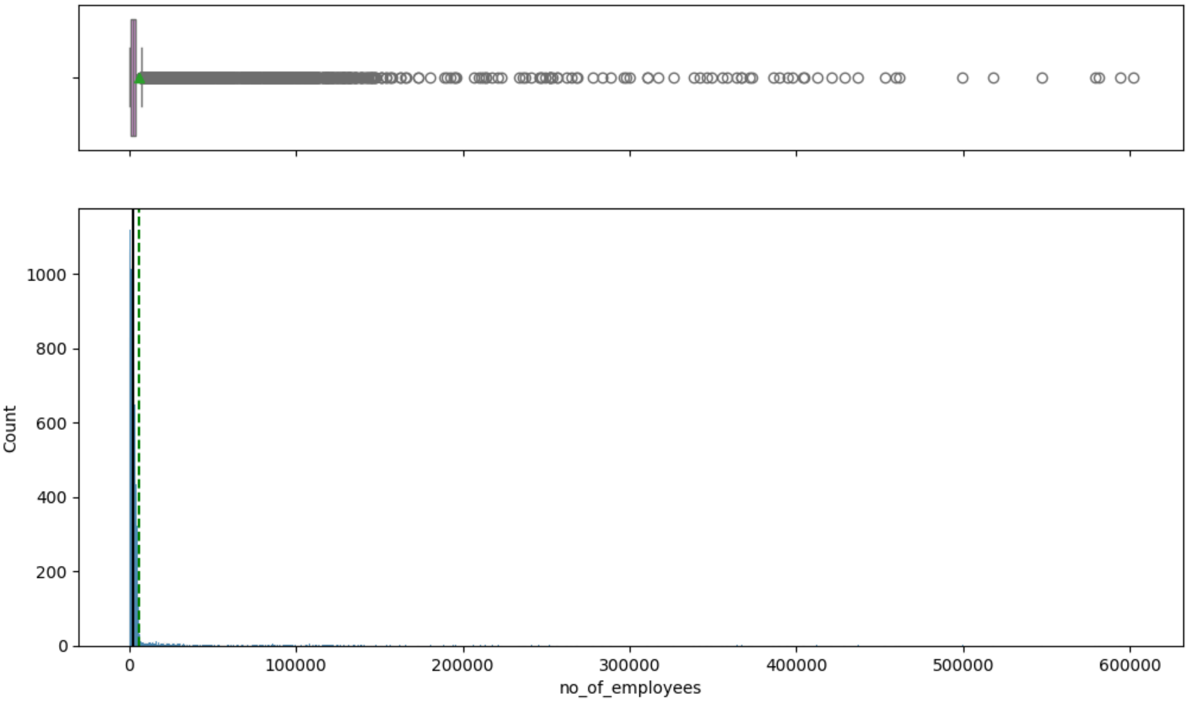


Figure 1: no_of_employees

- `no_of_employees` is highly skewed towards the right.

- We can see there are sme outliers present inthe plot which we need to have a look.

2. **yr_of_estab:**

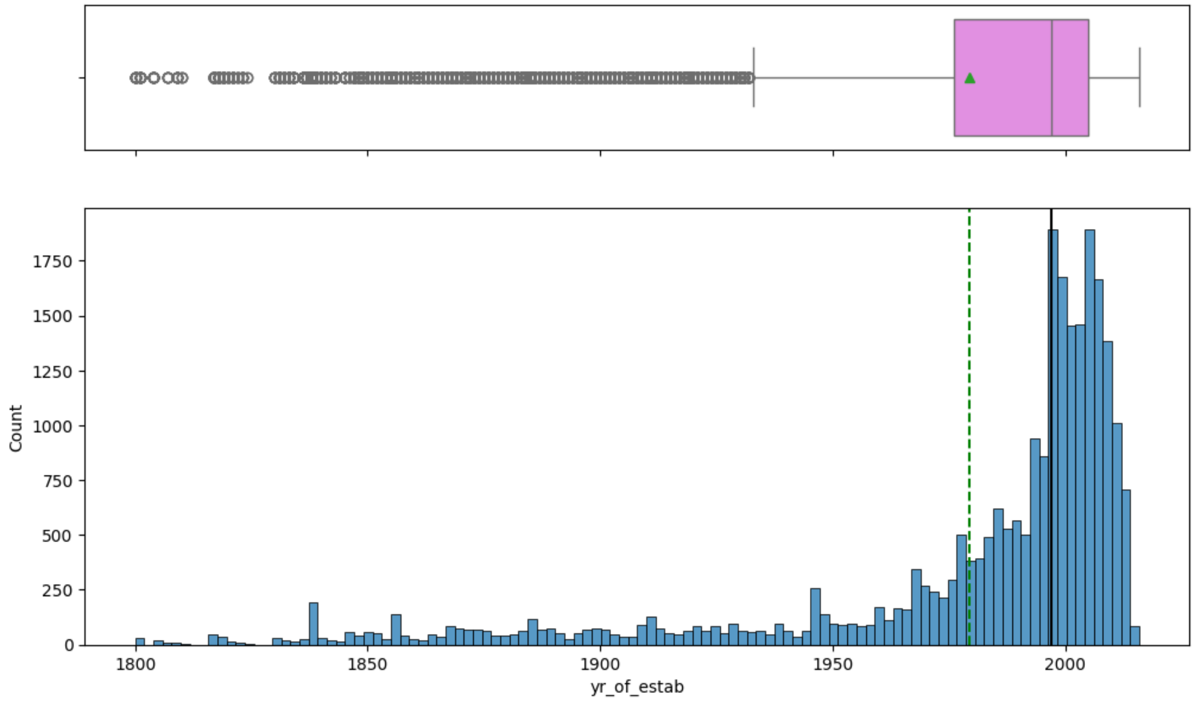


Figure 2: yr_of_estab

- We can clearly see that the `yr_of_estab` column is skewed towards left.

- Most of them are established post 1950 and then.

- The mean and median value are between 1970 to 2000.

3. prevailing_wage:

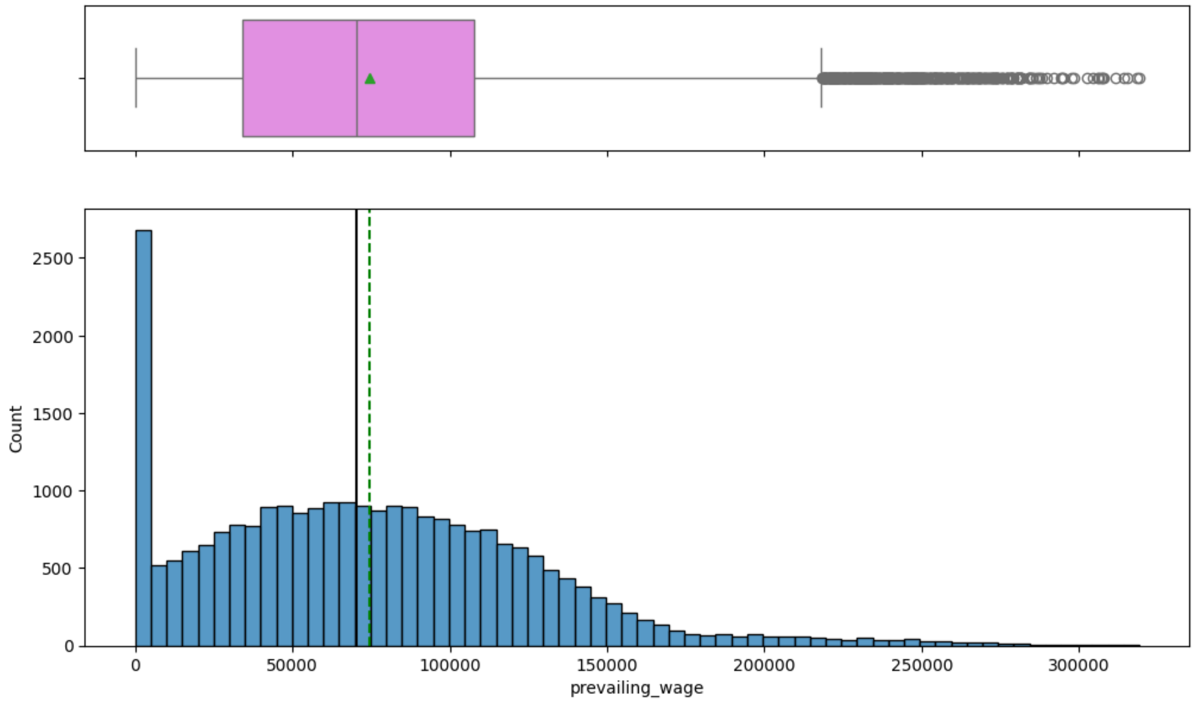
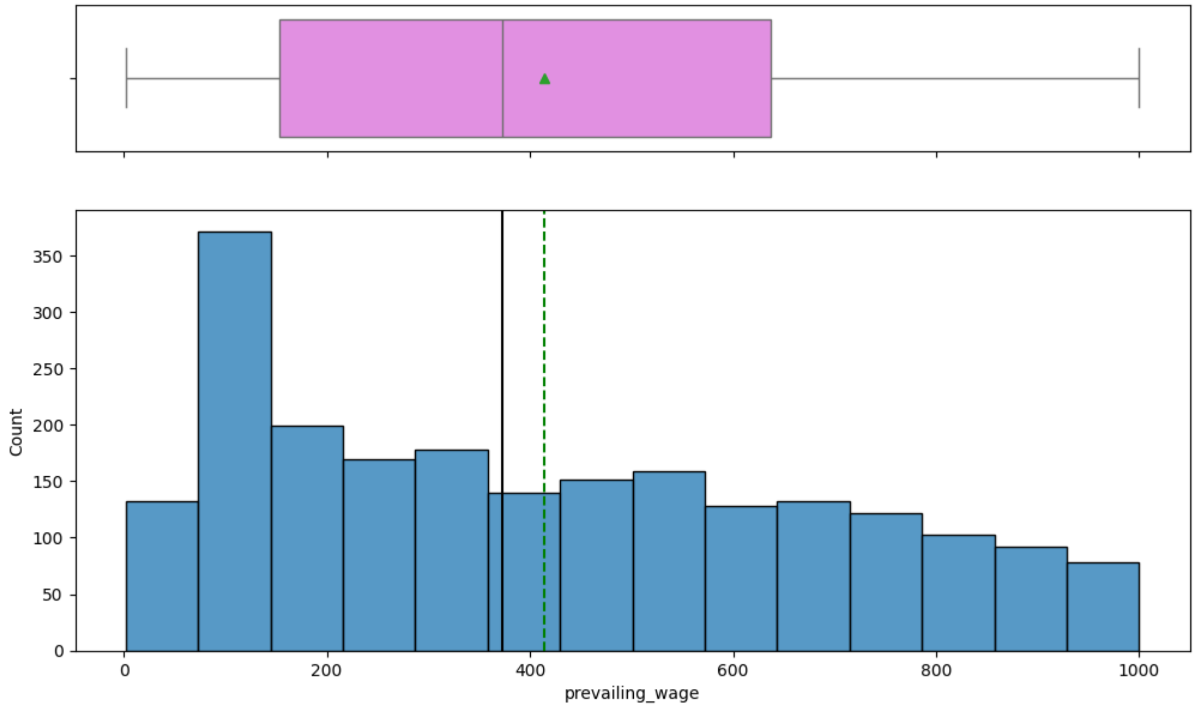


Figure 3: type_of_meal_plan

- `prevailing_wage` column has some values which is close to zero which we need have a look at.
- There are some outliers present but these values seems to be legit.

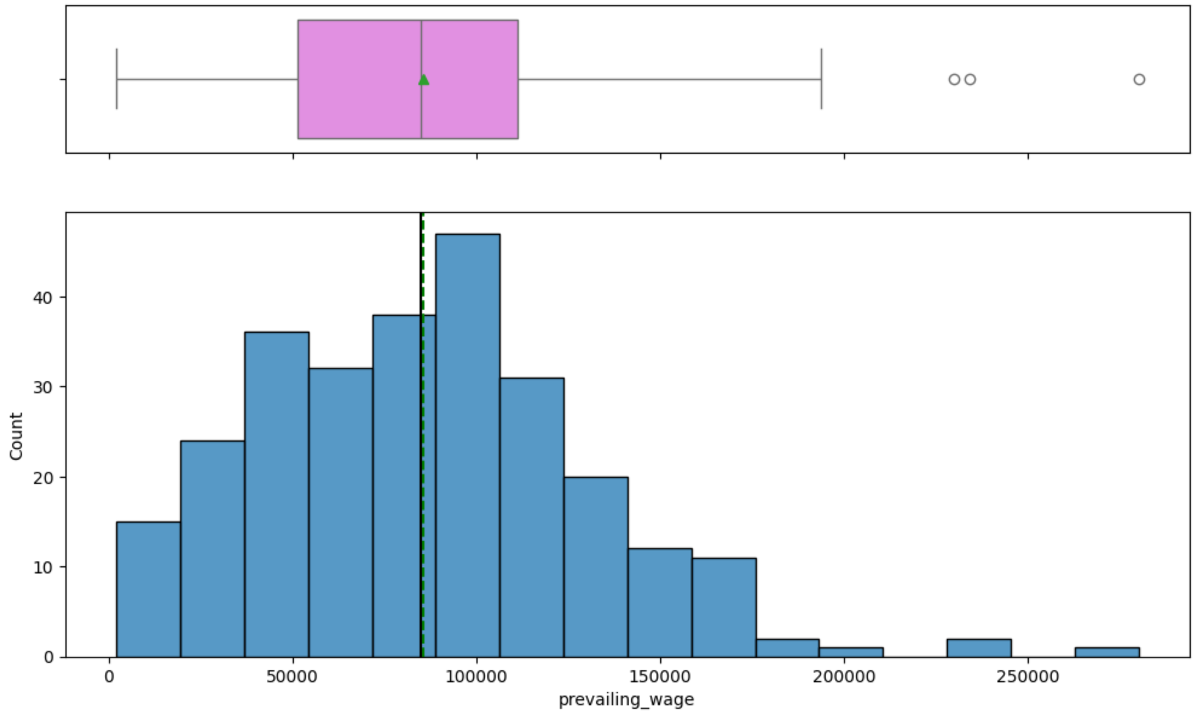
Employee having Hourly wages in Prevailing_wages:



- Above plot tells hourly wages range from 0 to 1000 which and the mean and median values are close to 400.

- We can assume that the above values indicates the salary the employees are getting is weekly wage since 400 Dollars per hour is not normal.

Employee having Weekly wages in Prevailing_wages:

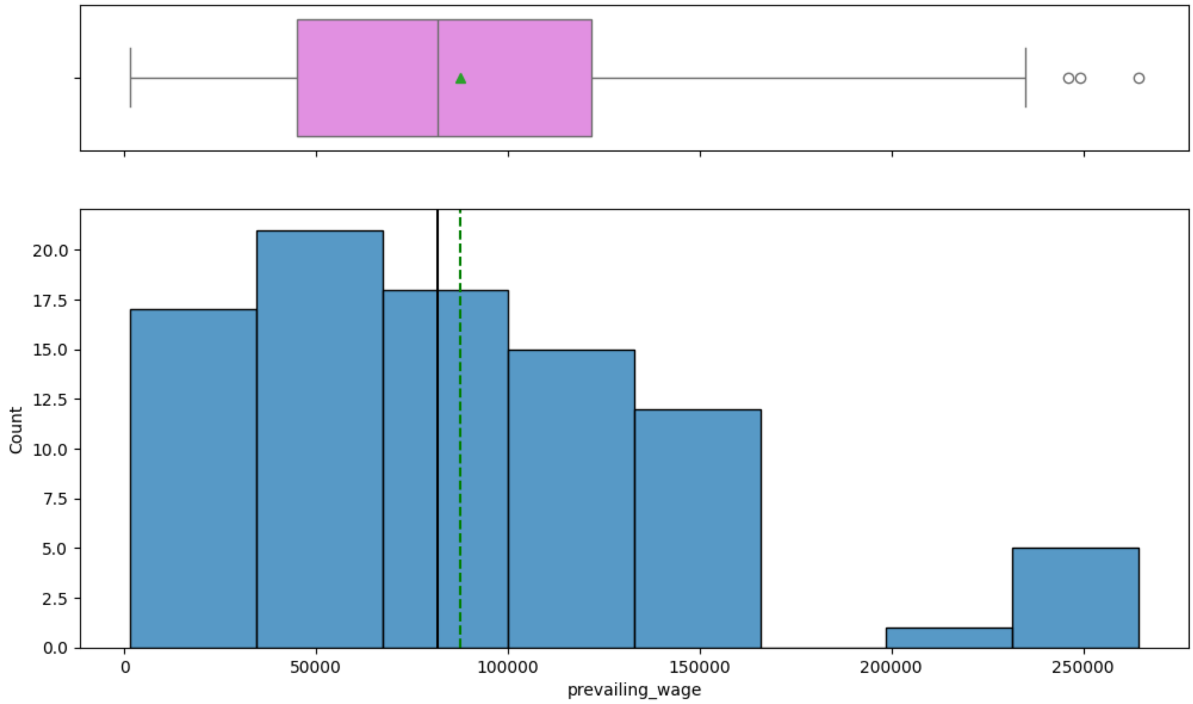


- The mean and median values lies somewhere near 80000 to 90000 which appears to be incorrect we need to look into it.

- There is slight skweness present towards the right.

- Most of the values are present between 0 and 150000.

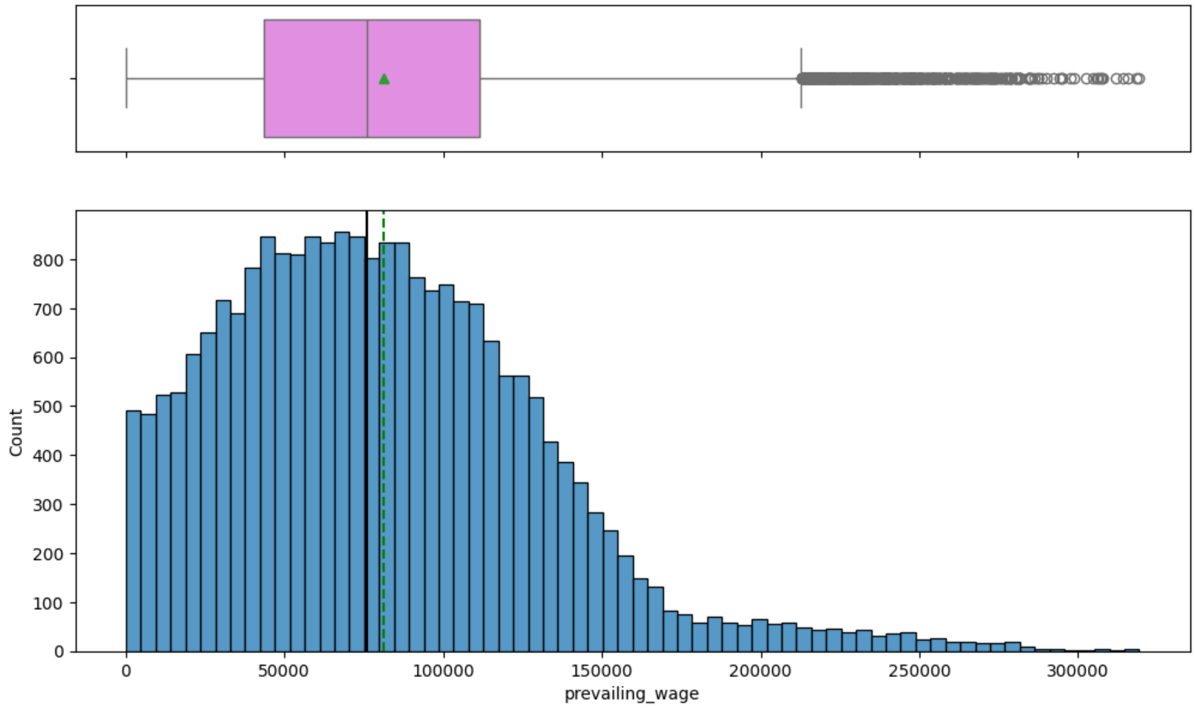
Employee having Monthly wages in Prevailing_wages:



- Similar to Weekly wages the Monthly wage mean and median value also lies near 80000 to 90000 USD which looks quite similar.

- Here also most of the values are present between 0 to 150000

Employee having Yearly wages in Prevailing_wages:



- Similar to Weekly wages and Monthly wage the Yearly wages mean and median value also lies near 80000 to 90000 USD.

- From the above observation we can conclude that the Weekly, Monthly and Yearly wages are all calculated annually.

Categorical Variables:

4. Continent:

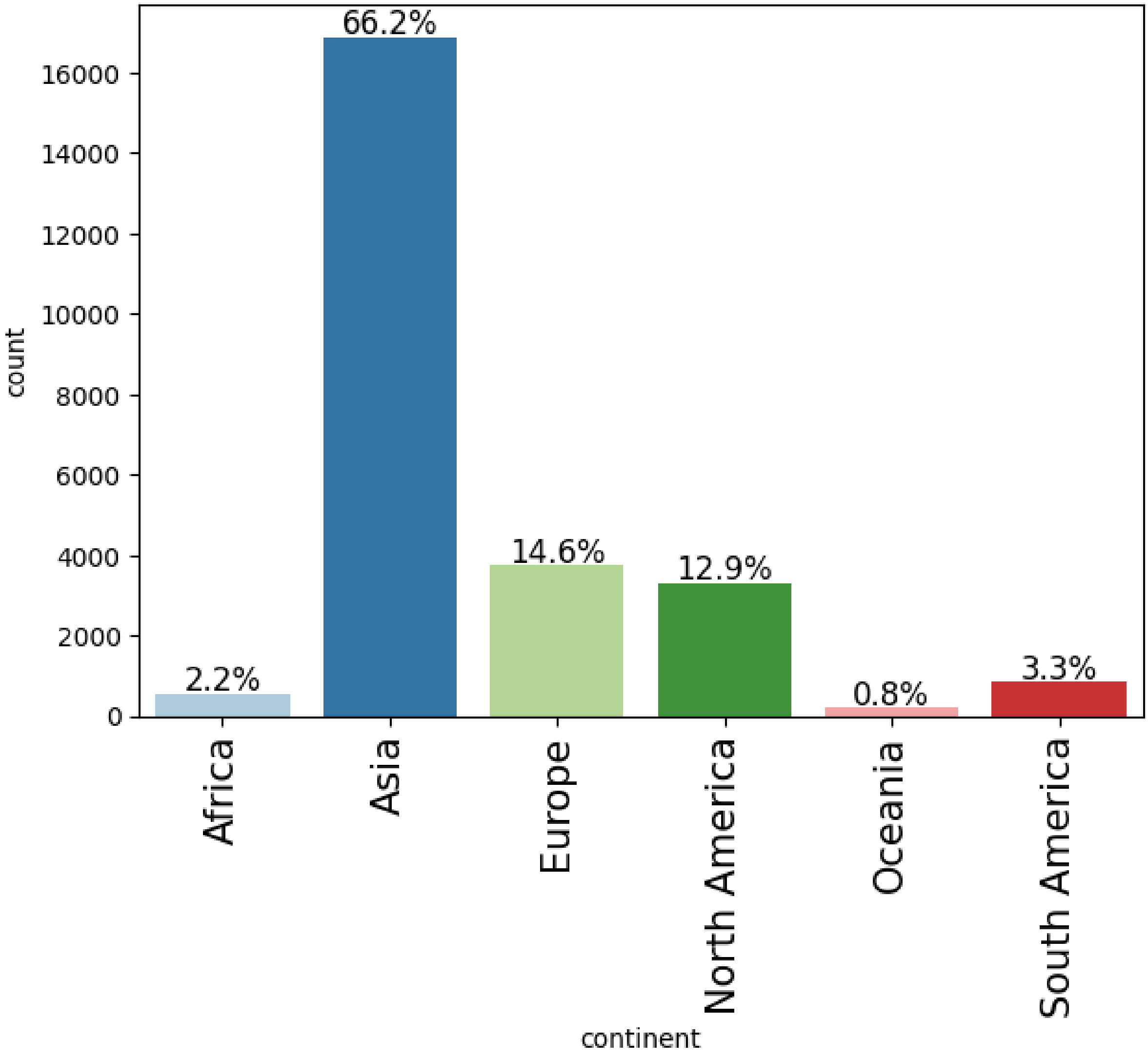


Figure 4: continent.

- From the above plot we can see that the `continent` column have 6 values Asia, Africa, North America, Oceania and South America.
- Among those Majority of the employees (> 50%) are from Asia.
- Europe has some 14.6%, North america has 12.9%, South america has 3.3%, Africca has 2.2% and oceania has 0.8%.

5. Market_segement_type:

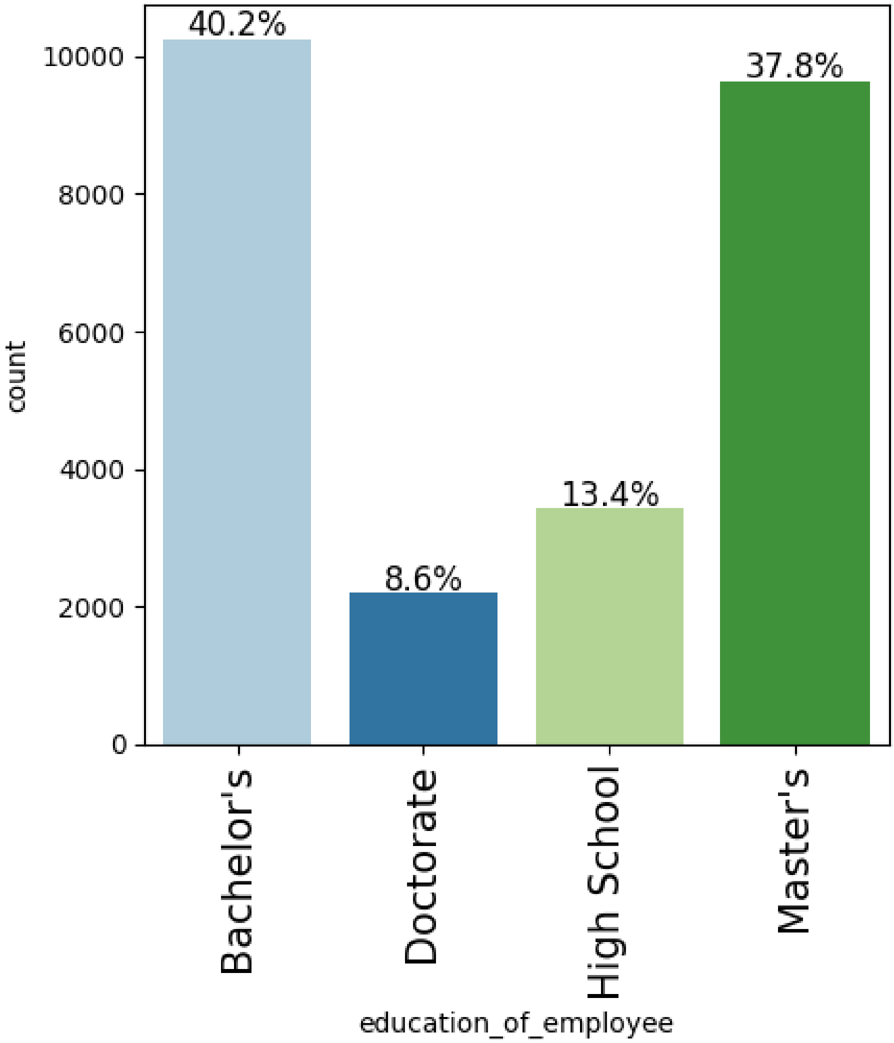


Figure 5: education_of_employee

- Majority of the employees have Bachelor's degree(40.2%) or Master's(37.8%), only minimum employees have Doctorate(8.6%) or High school(13.4%).

6. has_job_experience:

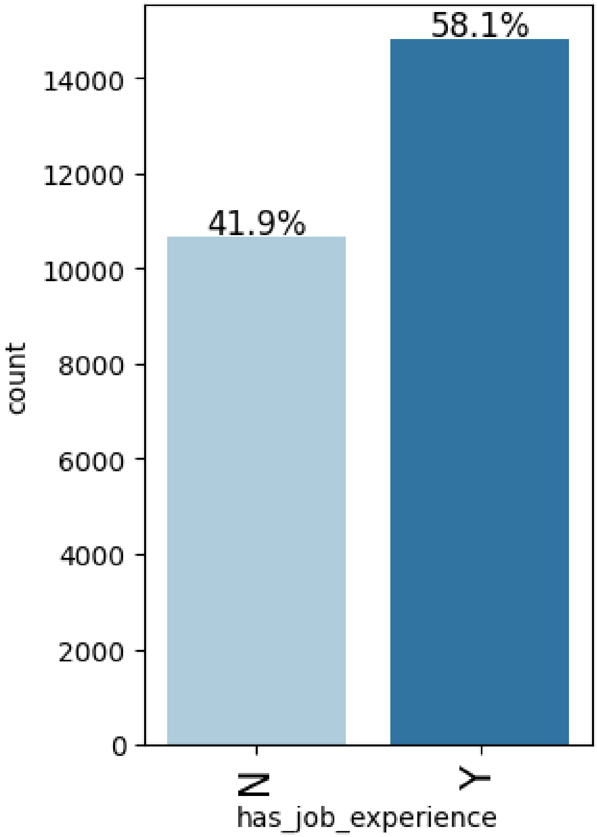


Figure 6: has_job_experience

- Most employees are experienced with 58.1% and 41.9% of employees are without experience.

7. requires_job_training:

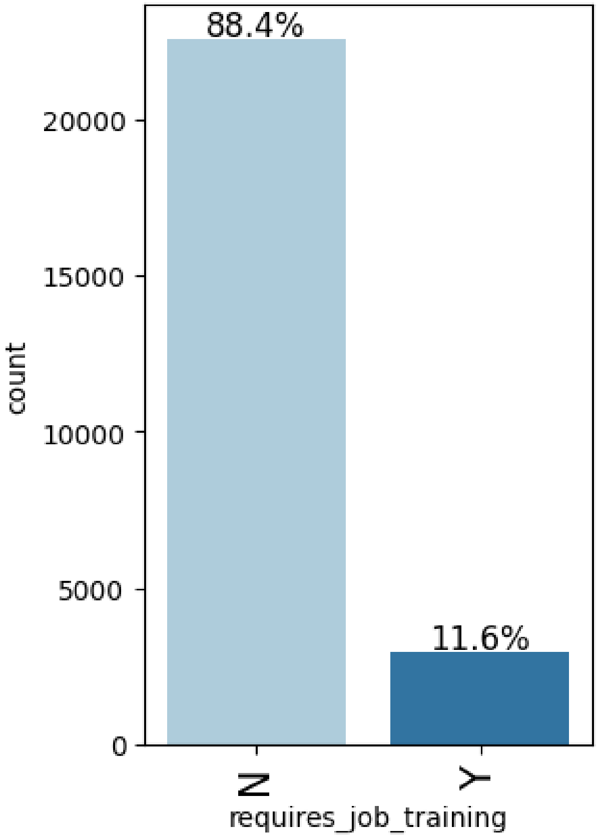


Figure 7: requires_job_training

- Its good to see that 88.4% of the employees do not require job training.

- Only 11.6% of the employees require job training.

8. region_of_employment:

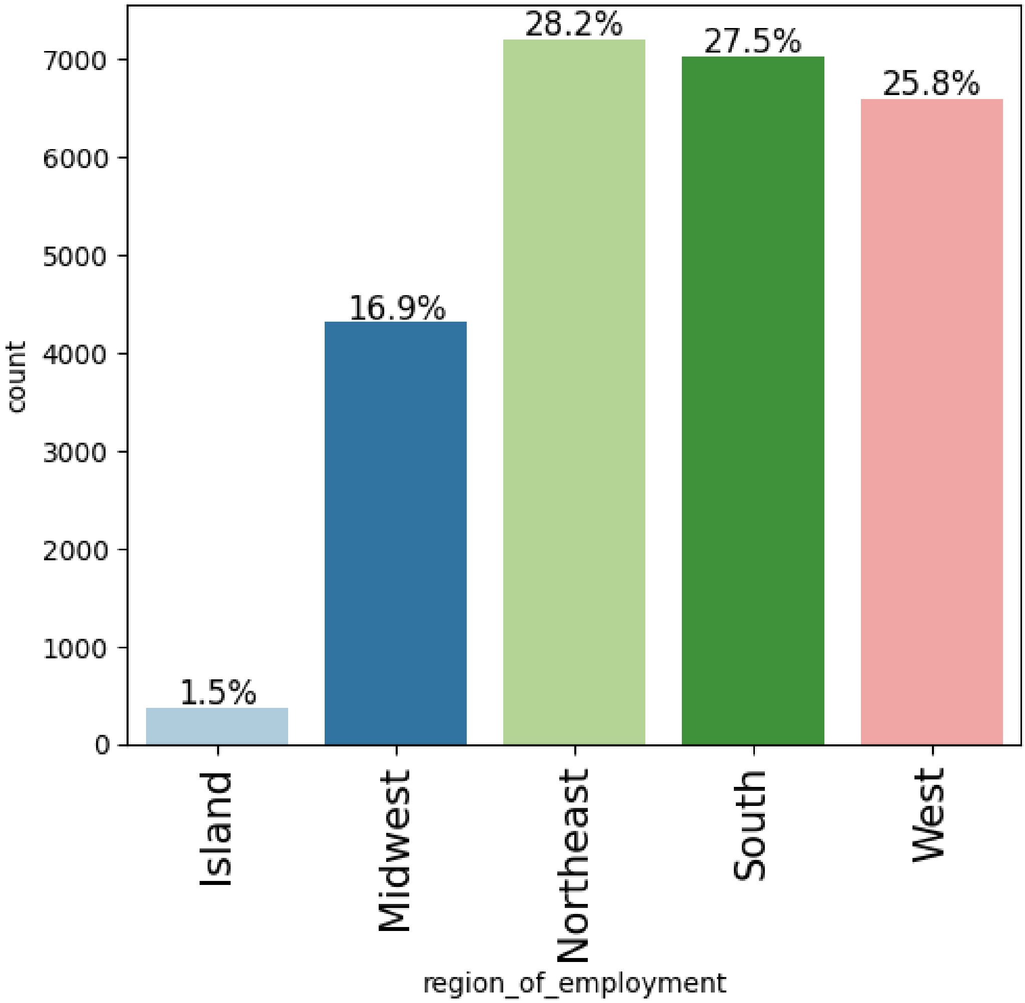


Figure 8: region_of_employment

- Northeast(28.2%), South(27.5%) and West(25.8%) have similar number of employees ranging from 25-29% applying for visa approval.

- Midwest(16.9%) and Island(1.5%) hase less employees compared to others.

9. unit_of_wage:

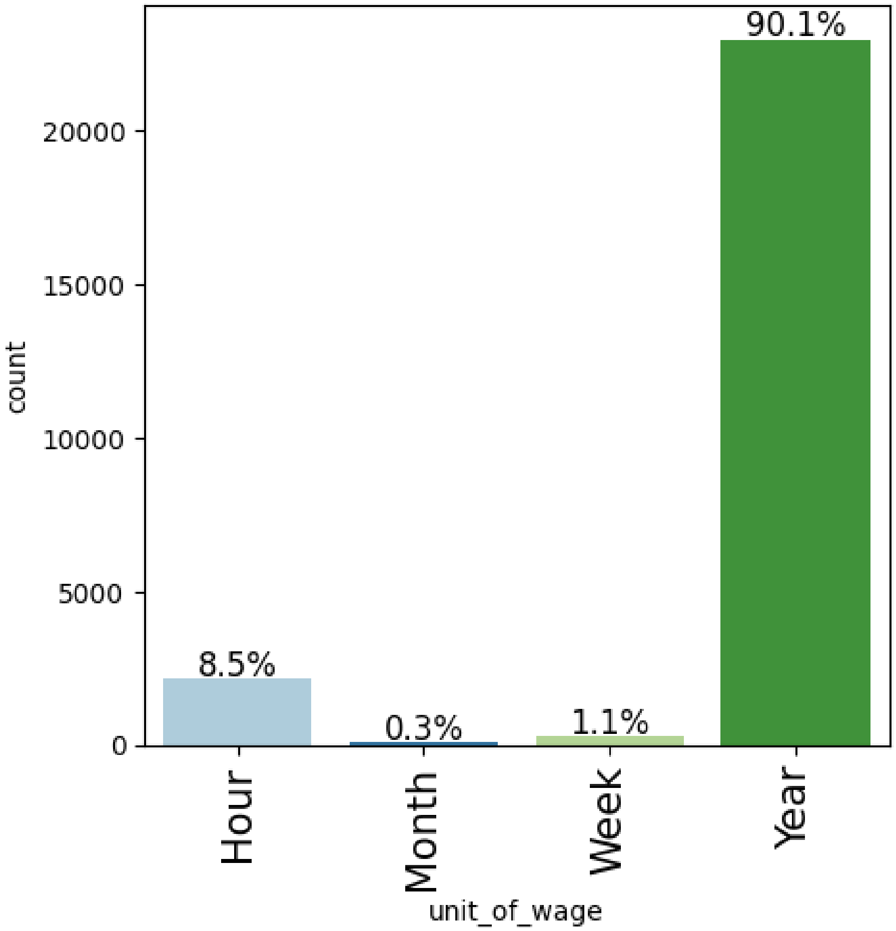


Figure 9: unit_of_wage

- 90.1% of employees are getting Yearly wages.

- 8.5% of the employees are getting Hourly wage.

- Minimum number of employees are getting wages on Monthly (0.3%) and Weekly (1.1%).

10. full_time_position:

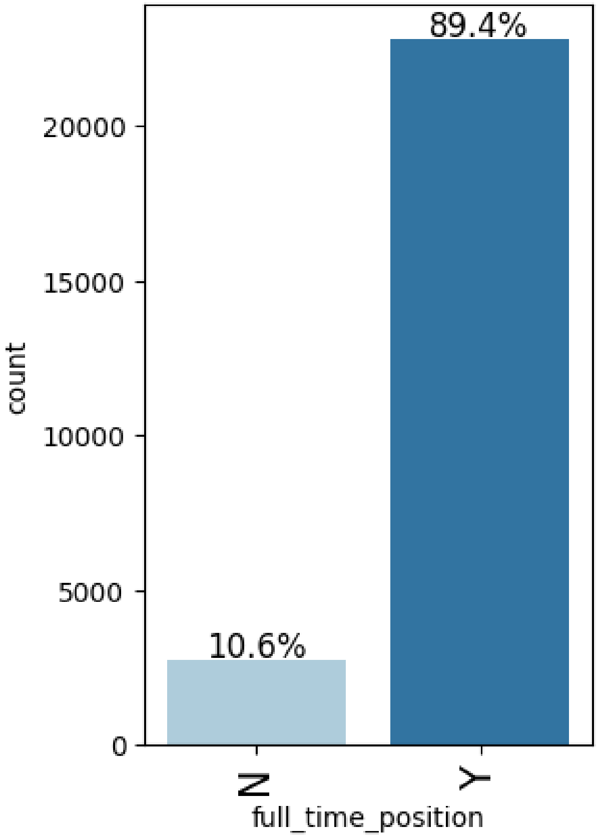


Figure 10: full_time_position

- Most of the employees applied visa are full time employees with 89.4% from overall.
- 10.6% of the applicants have applied for Part time jobs.

11. case_status:

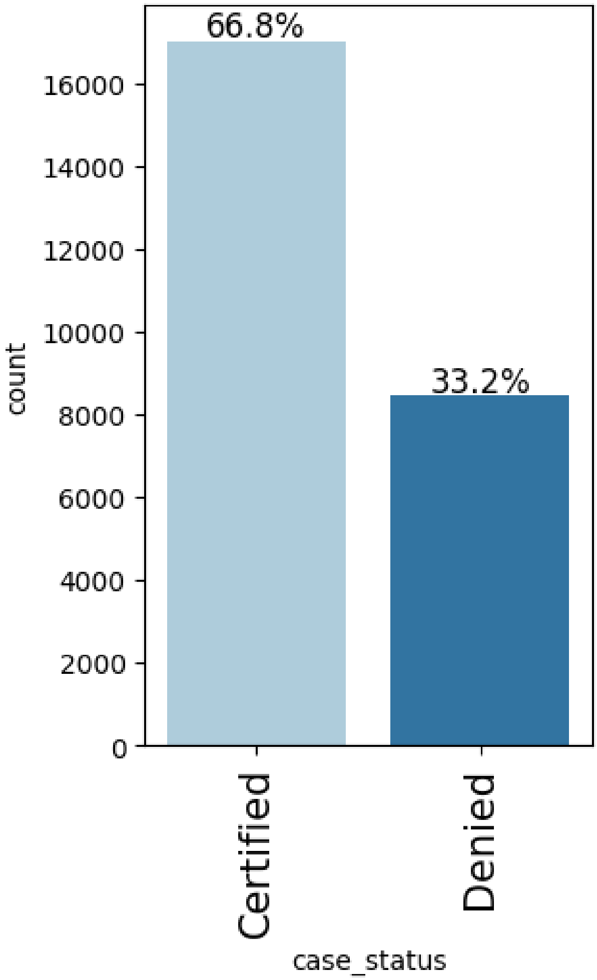


Figure 11: case_status

- From the above plot we can clearly see that most of the applications were certified(66.8%).
- There are 33.2% of the applications that are denied.

Bivariate Analysis:

Heatmap:

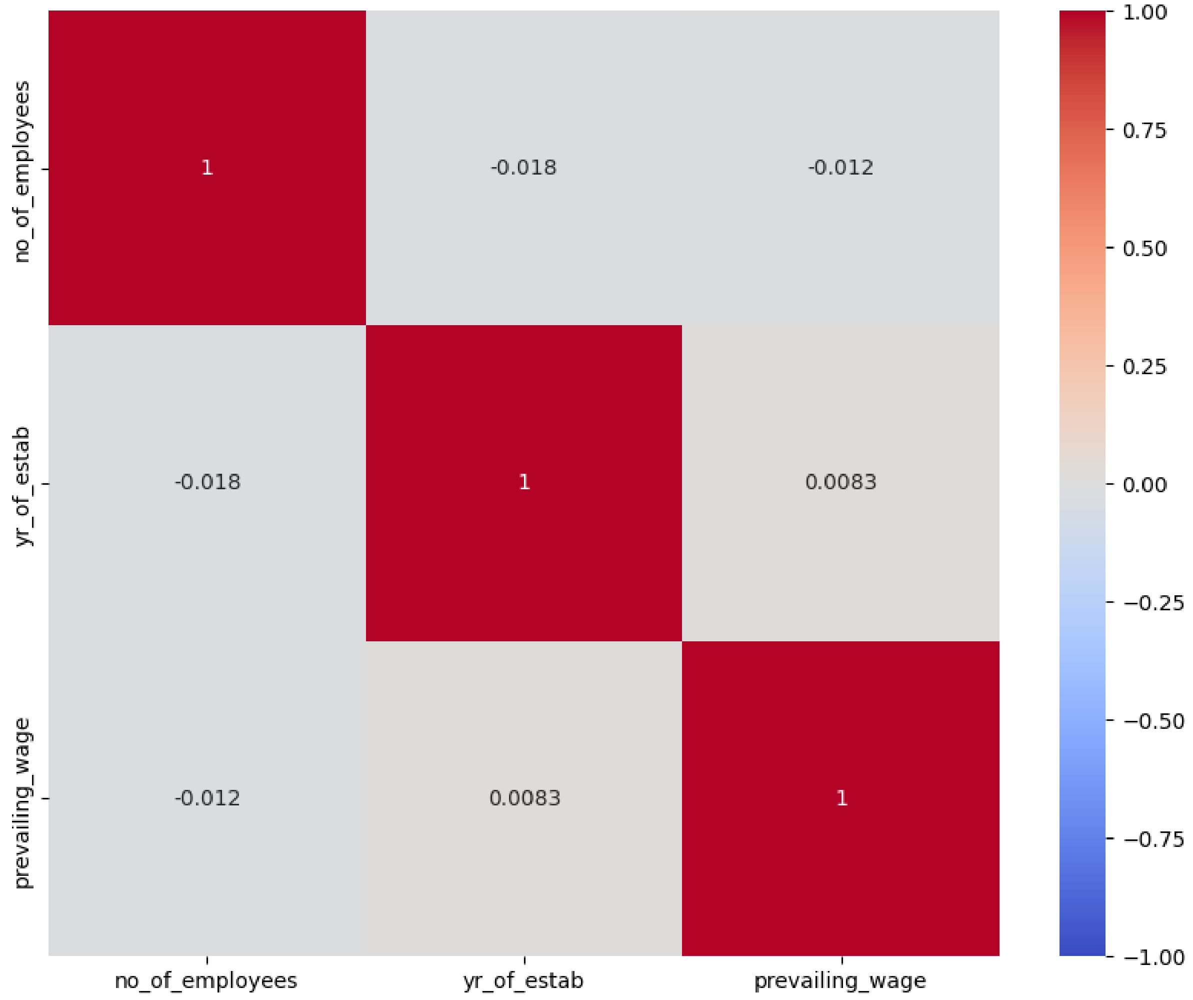


Figure 12: Heatmap

- From the above heatmap we can clearly see that there is no strong relationship between the numerical columns.'
- There is a weak negative relationship between `prevailing_wage` vs `no_of_emplooyees` and `yr_of_estab` vs `no_of_employees`.
- `prevailing_wage` and `yr_of_estab` has some positive correlations between them.

1. no_of_employees VS case_status:

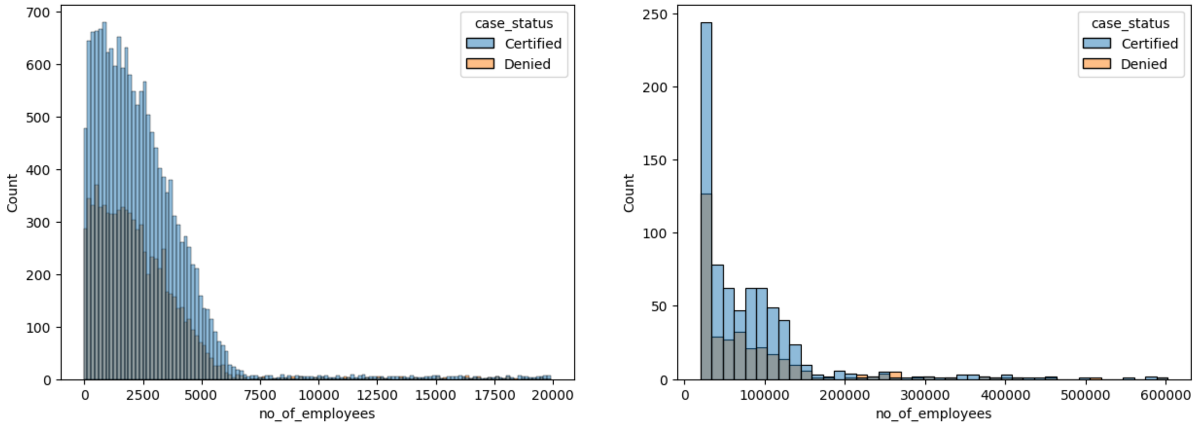


Figure 13: no_of_employees VS case_status.

- From the above plot we can see that the there is a huge difference in number of cases that are certified and number of cases that are denied.
- Since the `no_of_employees` column doesn't have much relationship to the target variable we can split the continous data in `no_of_column` into 43 different category.

2. continent vs case_status:

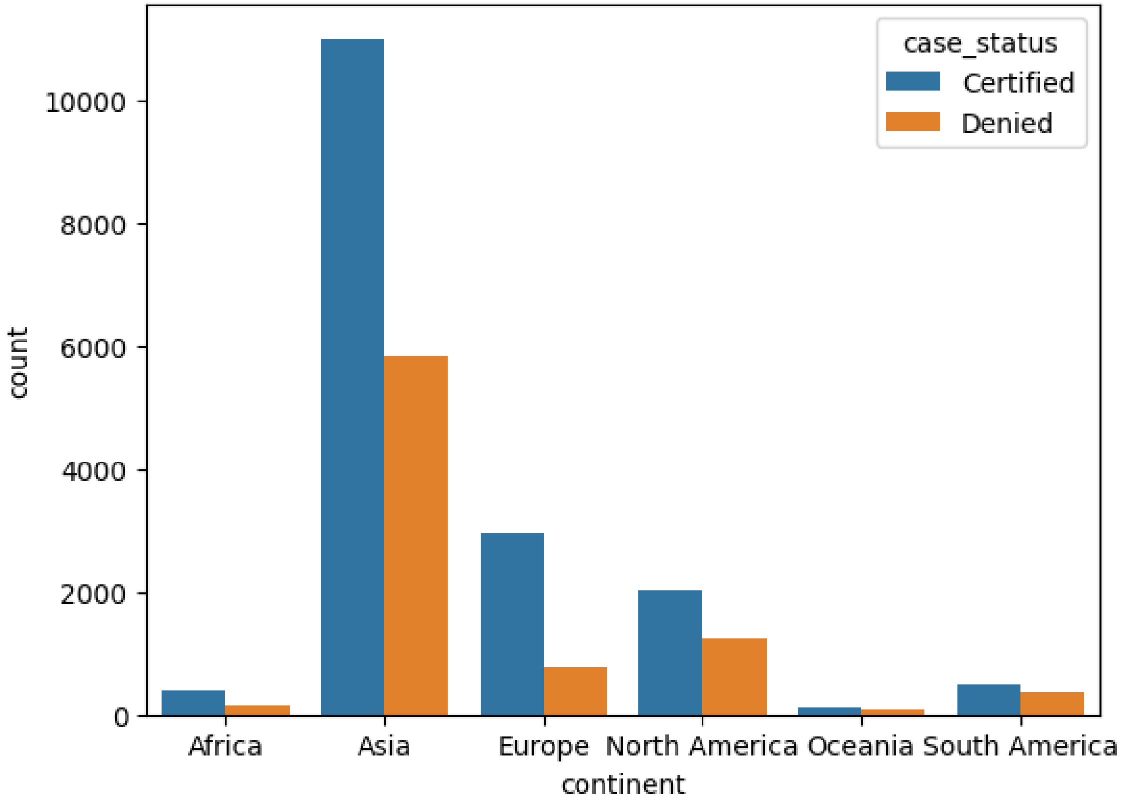
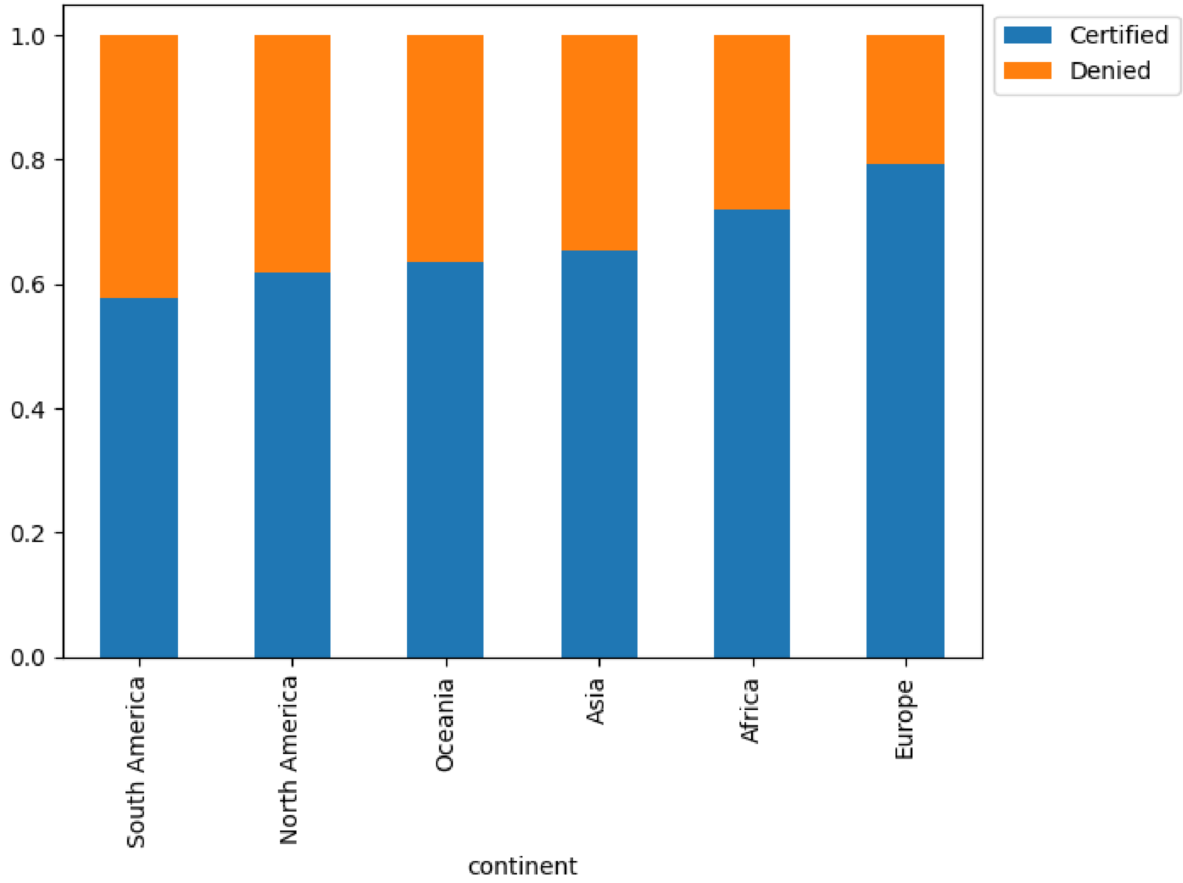
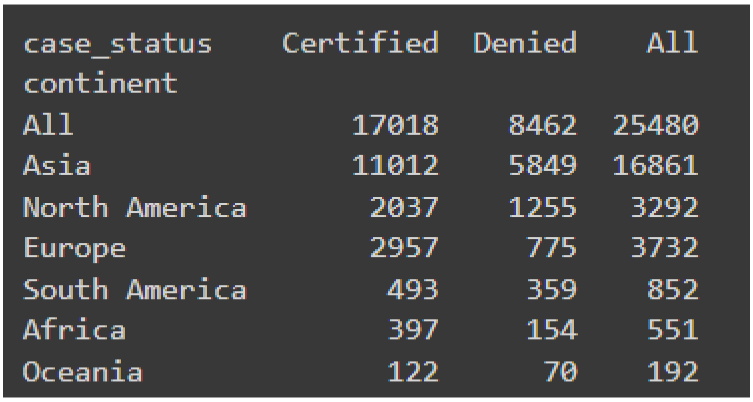


Figure 14: continent VS case_status

- From the above plot as expected the number of employees who are certified for visa are more compared to denied across all countries.





```
case_status    Certified  Denied    All
continent
All                17018    8462  25480
Asia               11012    5849  16861
North America       2037    1255   3292
Europe              2957     775   3732
South America        493     359    852
Africa               397     154    551
Oceania              122      70    192
```

- From the above plot as expected the number of employees who are certified for visa are more compared to denied across all countries.
- The Proportion with respect to certifications of the continents is Europe, Africa, Asia, Oceania, North america, South america.

3. education_of_employee vs case_status:

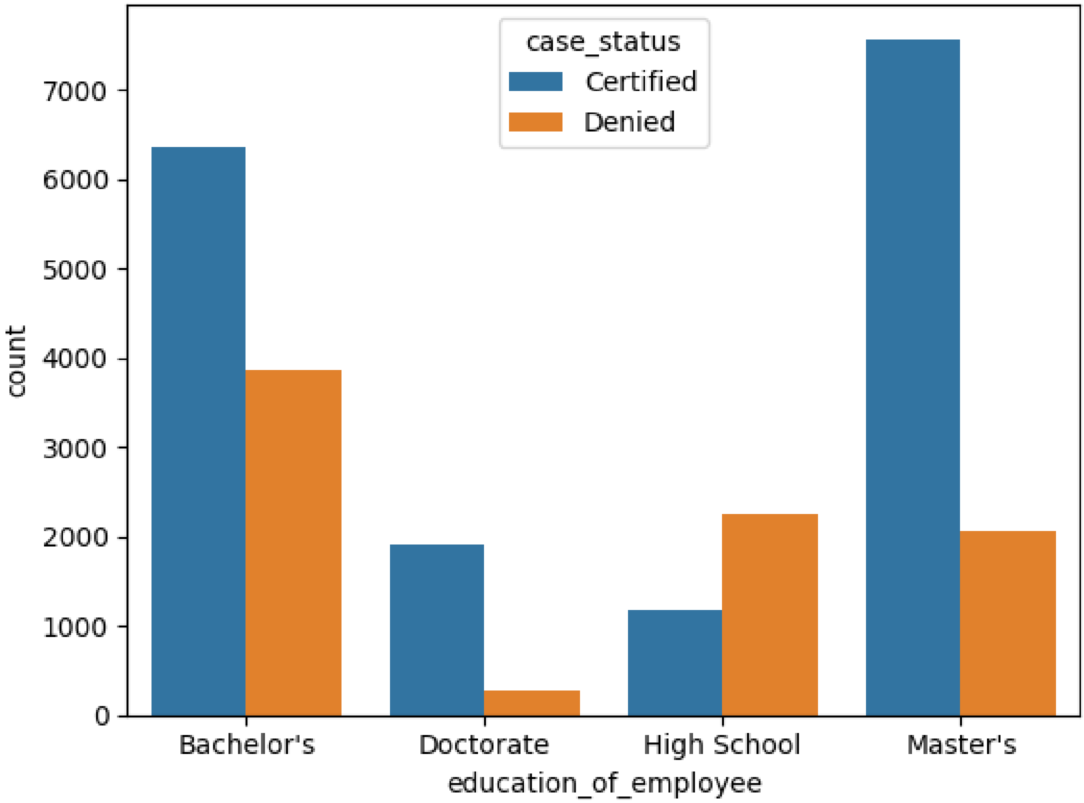
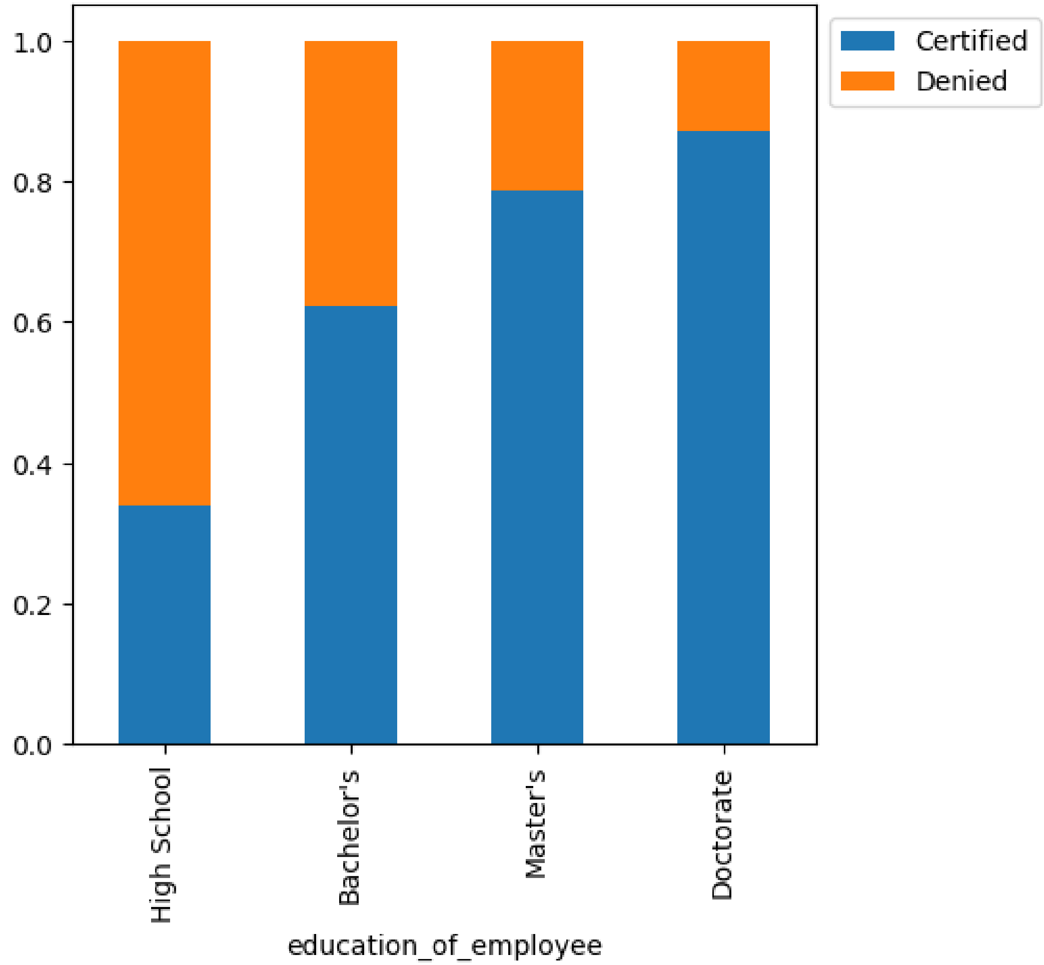
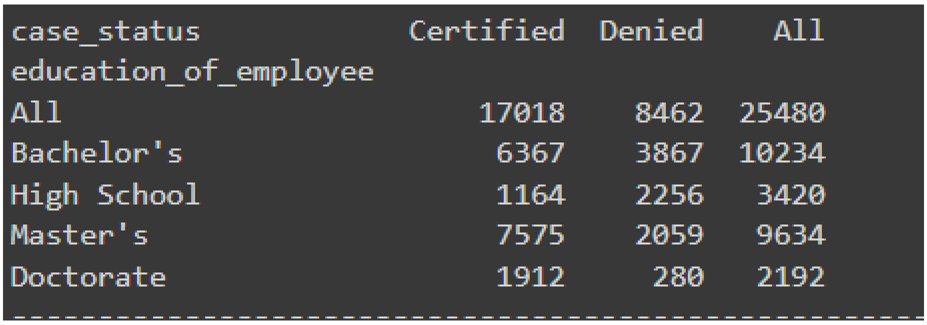


Figure 15: education_of_employee VS case_status

- From the above countplot the employees who have master's degree have higher certified values compared to denied.
- The employees with bachelor's degree also have more certified vallues compared to denied.
- People with High school degree have been denied for visa and only few cases it has been approved.
- Doctrate employees seem to have higher proportion with certified and denied.

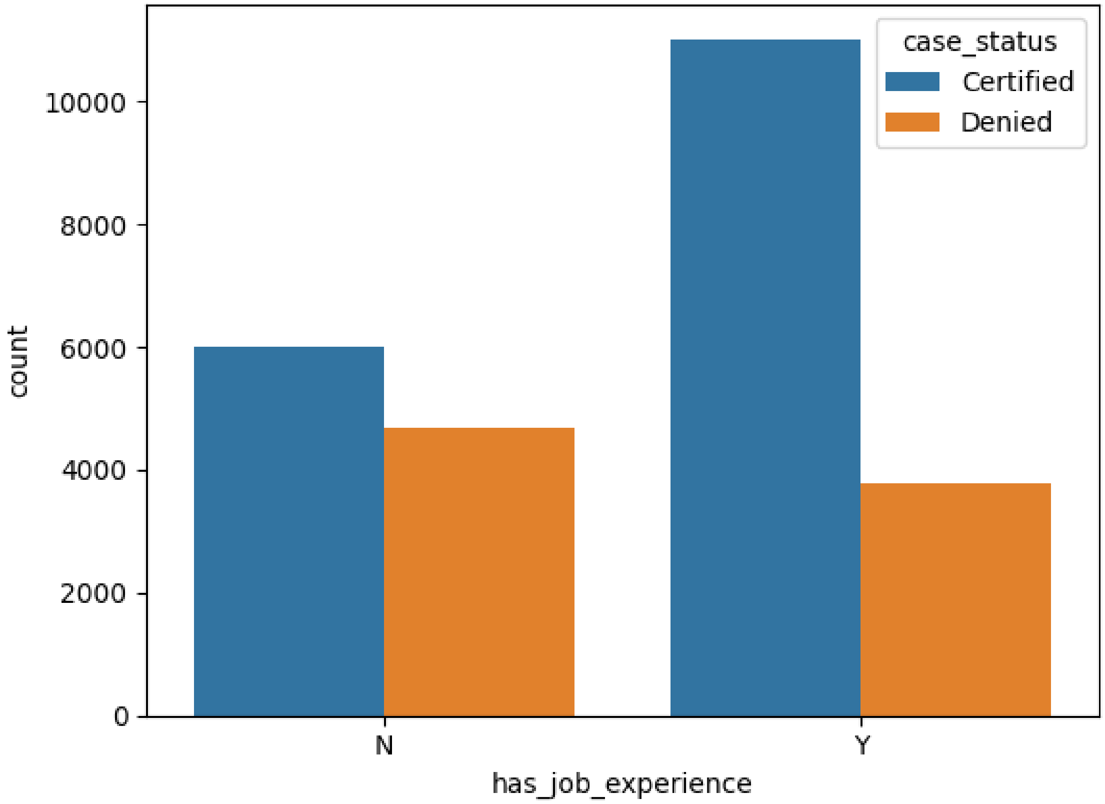




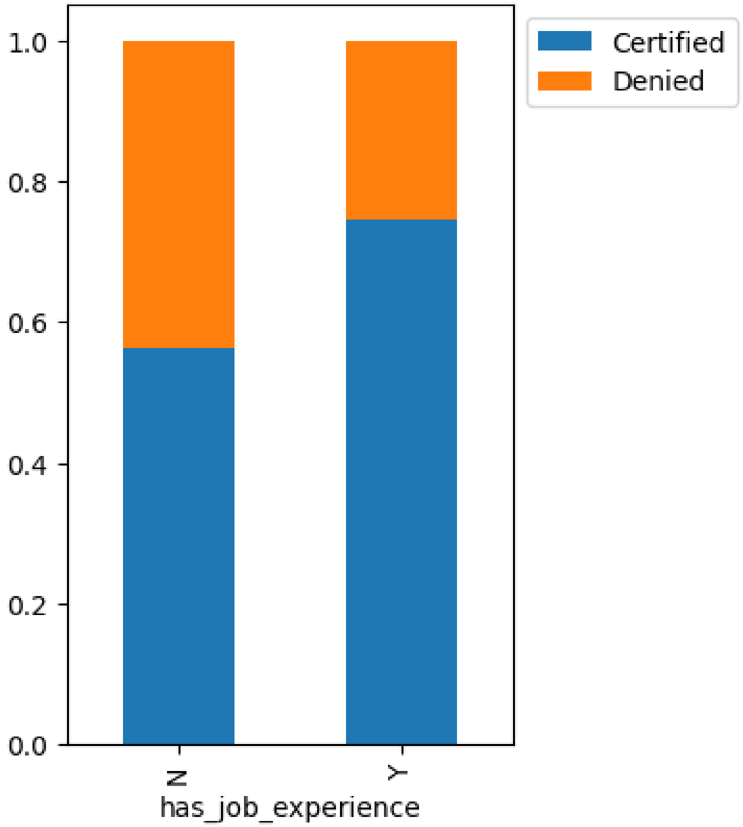
```
case_status            Certified  Denied    All
education_of_employee
All                        17018    8462  25480
Bachelor's                  6367    3867  10234
High School                 1164    2256   3420
Master's                    7575    2059   9634
Doctorate                   1912     280   2192
----------------------------------------------------
```

- The Proportion with respect to certifications of the `education_of_employee` is Doctrate > Master > Bachelor > High school.

4. has_job_experience VS case_status:

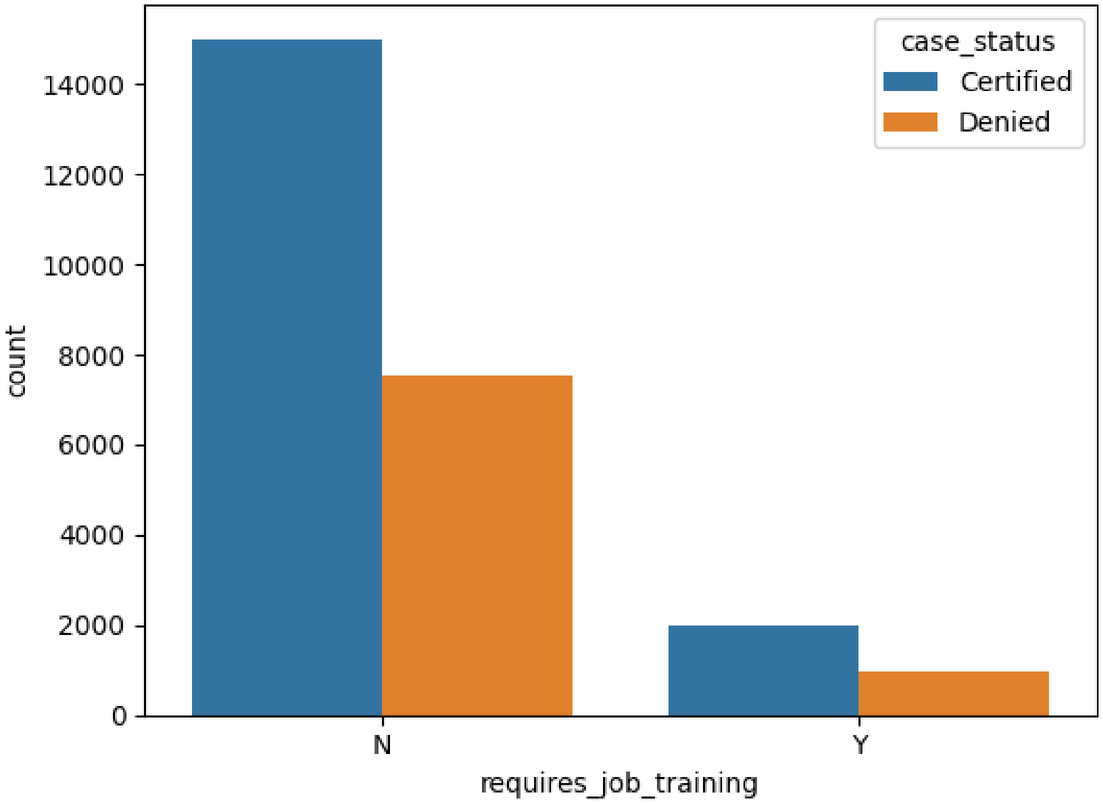


- As expected the people who have prior work experience have are more approved for visa.
- The proportion for certified and denied for employees witgout previous work experience seems to be equal.

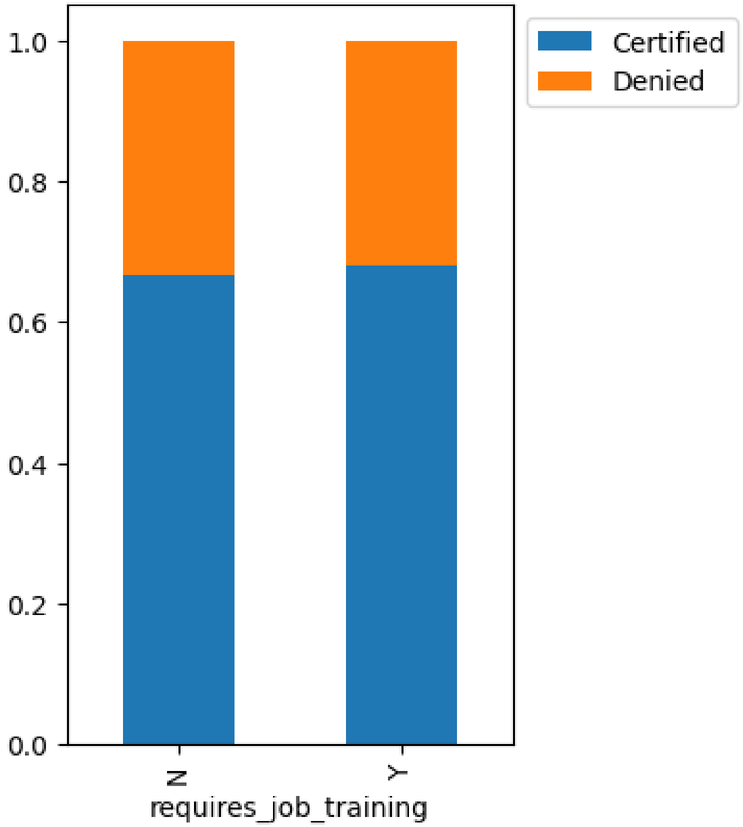


- From the above plot we can clearly see that the proportion of people having certified for visa have previous job work experience.

5. requires_job_training VS case_status:

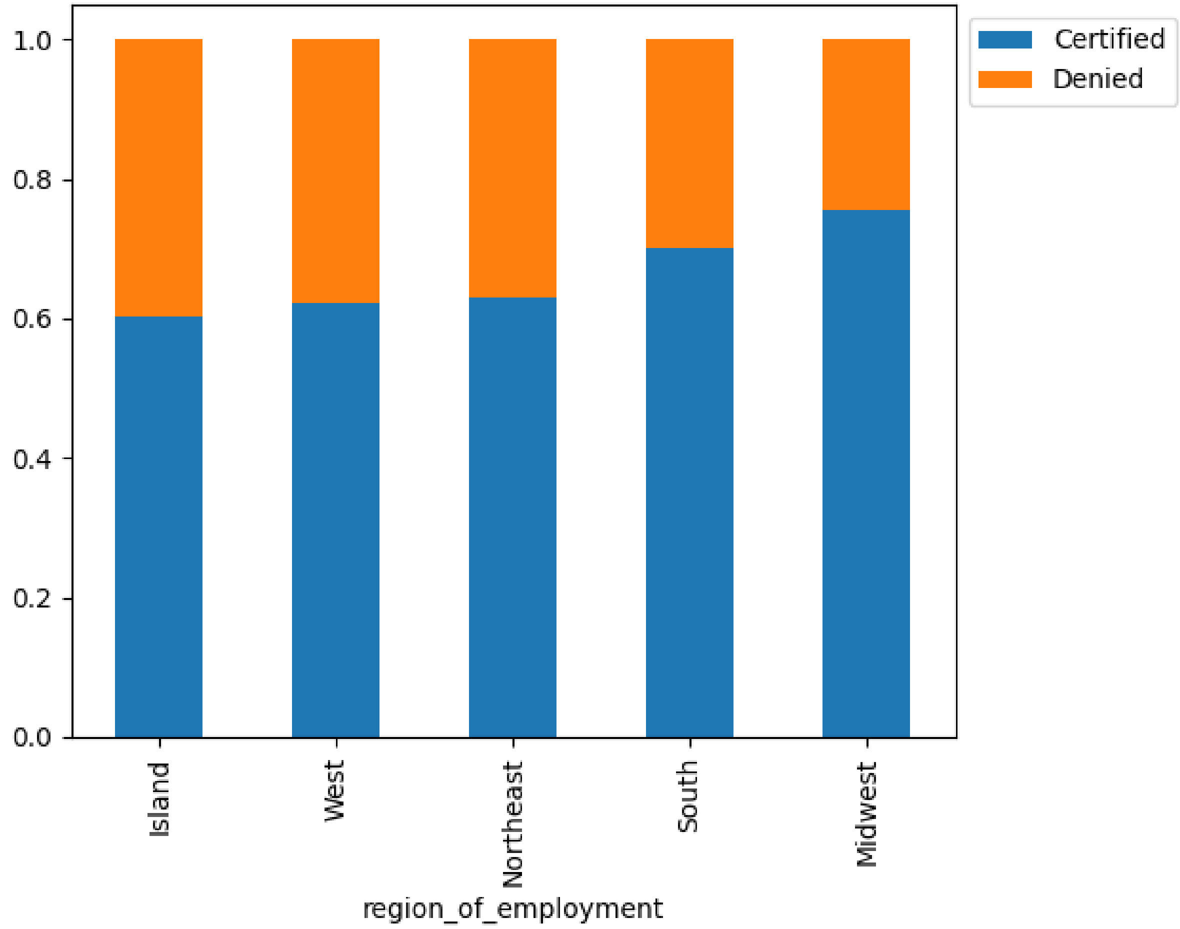


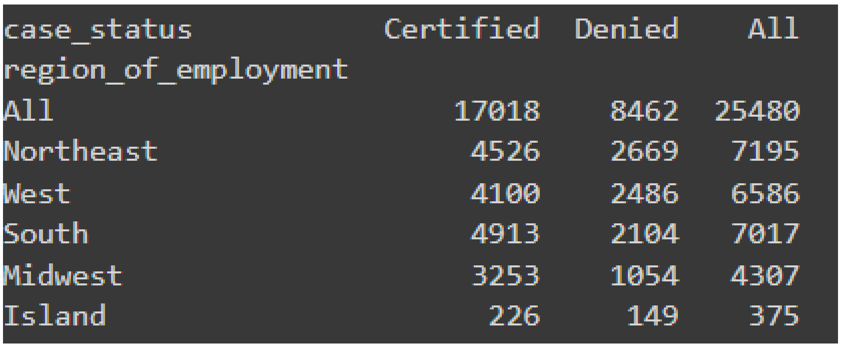
- From the above the graph we can tell that people who doesn't want job training.
- People who requires job training are less in number and the proportion is less.



- From the above plot we can clearly see that both people requiring job training and people who doesn't require job training have similar proprtion.

6. region_of_employment:

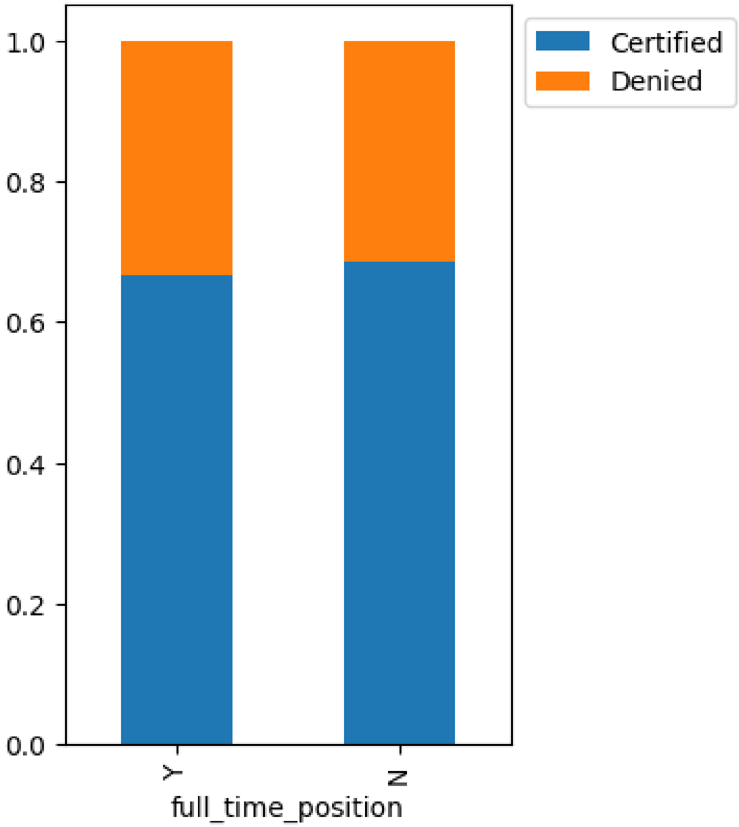




```
case_status            Certified  Denied    All
region_of_employment
All                        17018    8462  25480
Northeast                   4526    2669   7195
West                        4100    2486   6586
South                       4913    2104   7017
Midwest                     3253    1054   4307
Island                       226     149    375
```

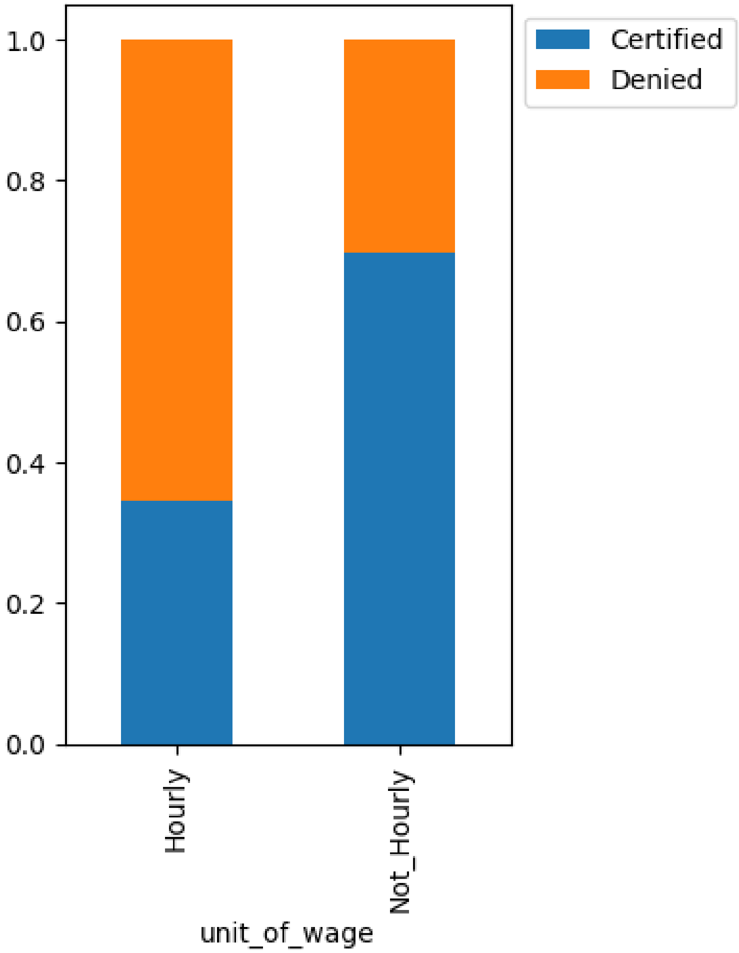
- The above plot tells us that the proportion between the region of employment is in order: Midwest > South > Northwest > Island.
- The proportion between certified and denied for Northwest, West and Island have similar proportions.

7. full_time_position VS case_status:



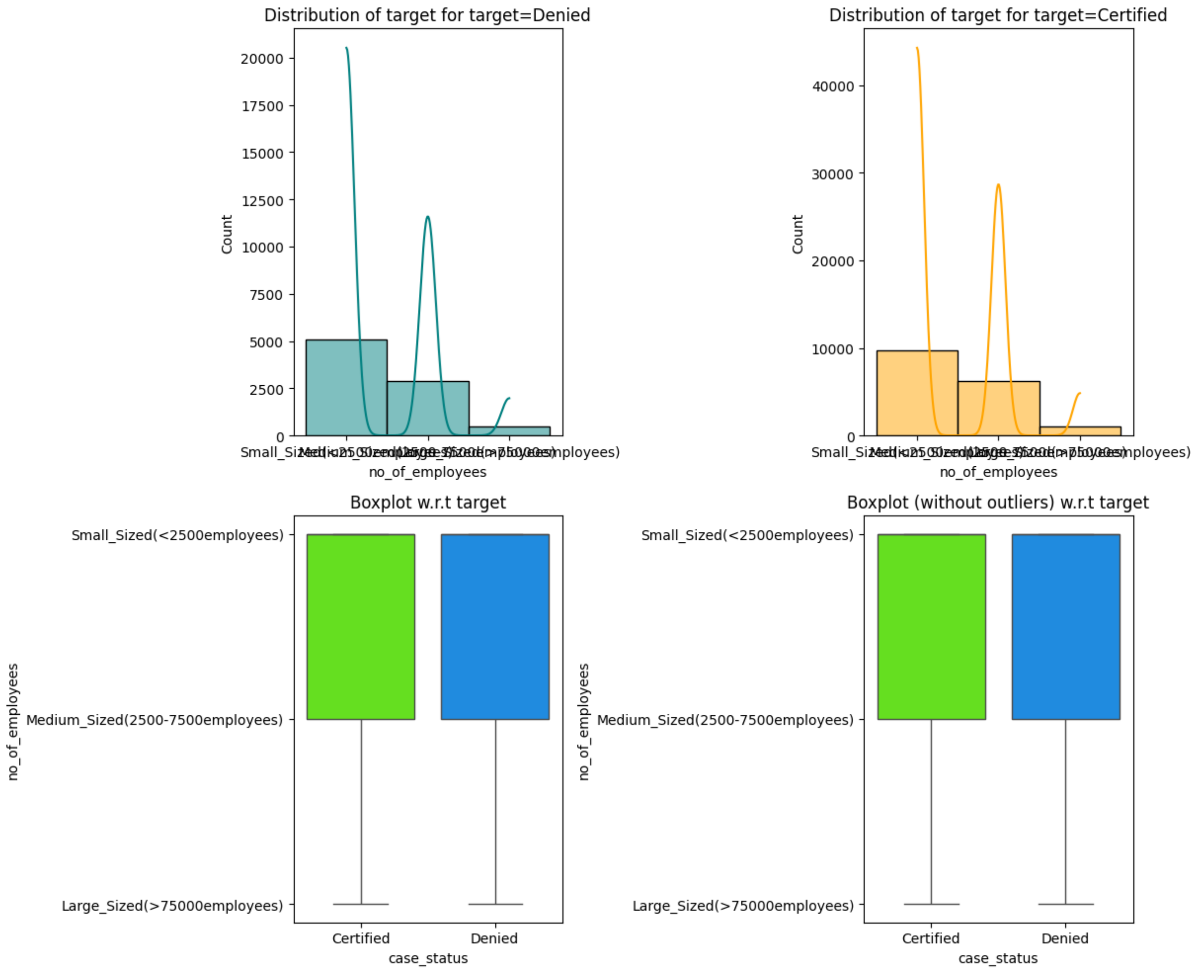
-The Proportion for the above plot with respect to visa certification full_time_position is Yes ~ No.

8. unit_of_wage VS case_status:



- From the above, the trend is with respect to visa certifications is Not_Hourly > Hourly.

9. no_of_employees VS case_status:



- From the above plots we can see the distribution of no_of_employees with respect to case_status.
- we can see that the median of the certified and denied have approximately similar and near to 2000 no_of_employees.

Outlier Treatment:

- We have some outliers in column `yr_of_estab` and `prevailing_wage`.
- Percentage for outlier present in the column `yr_of_estab` is 12.79%.
- Percentage for outlier present in the column `prevailing_wage` is 1.83%.

The outliers present in the data seems to be fine, we don't need to remove or impute those values we are leaving it as it is. We are not treating them.

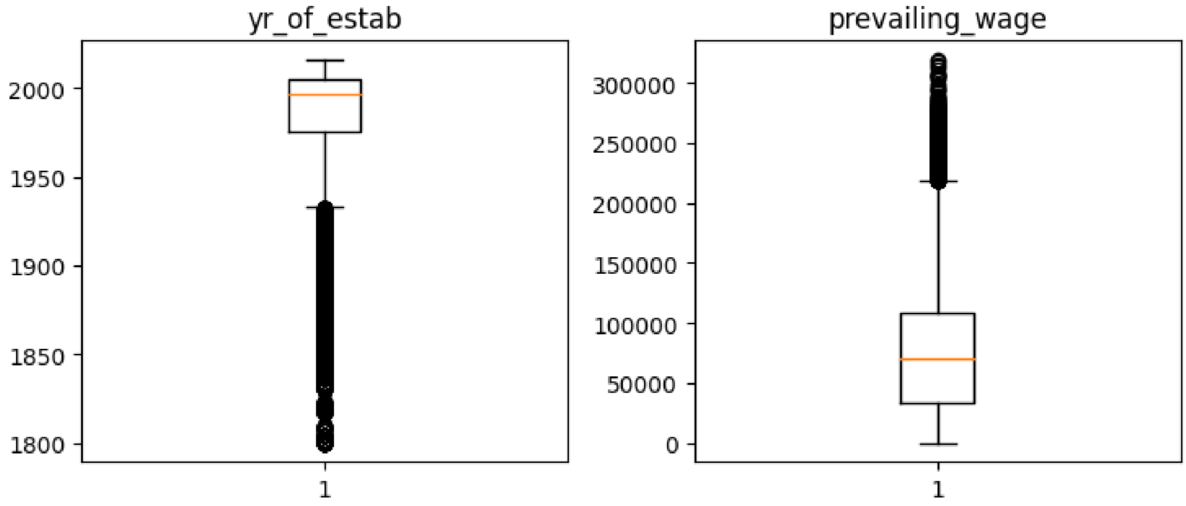


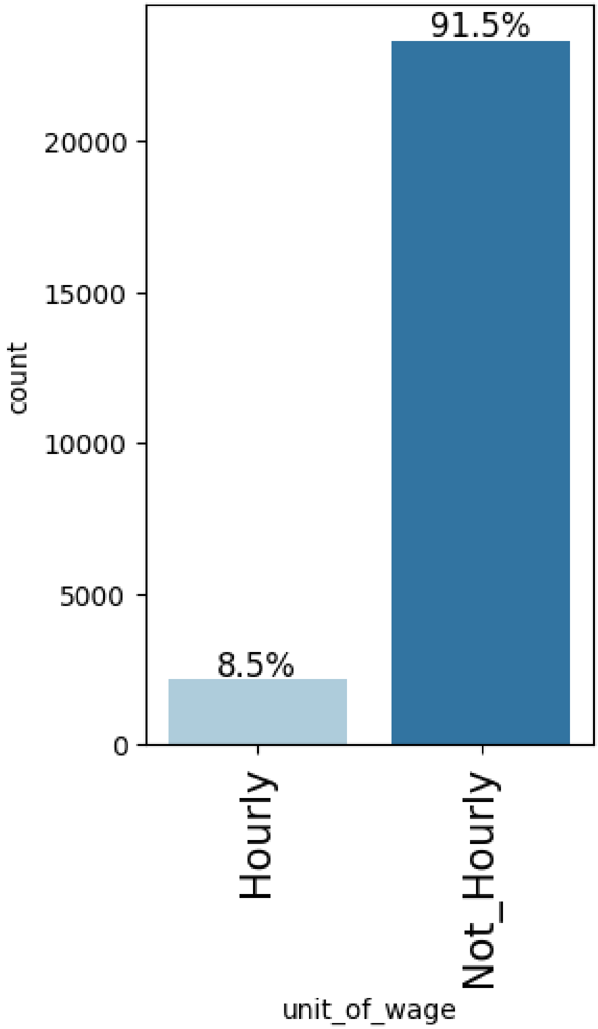
Figure 16: Outlier Treatment

Feature Engineering:

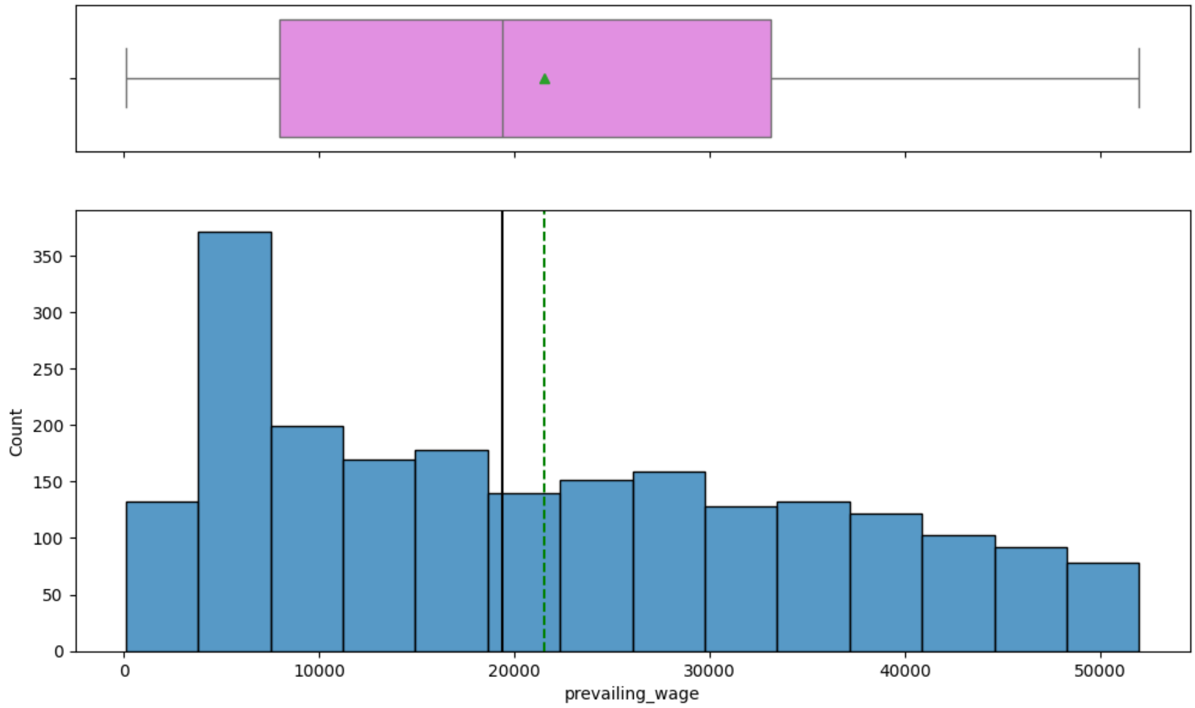
Prevailing_wage and unit_of_wage columns:

- In `unit_of_wages` column have 4 value ‘Hour’, ‘Week’, ‘Month’, ’Year’.

- Previously we saw in univariate analysis that the Weekly , Monthly and yearly wages of `prevailing_wage` column have are calculated annually and entered.
- So we can convert the Weekly , Monthly and yearly values in `unit_of_wage` colunmn as ‘Not_hourly’.



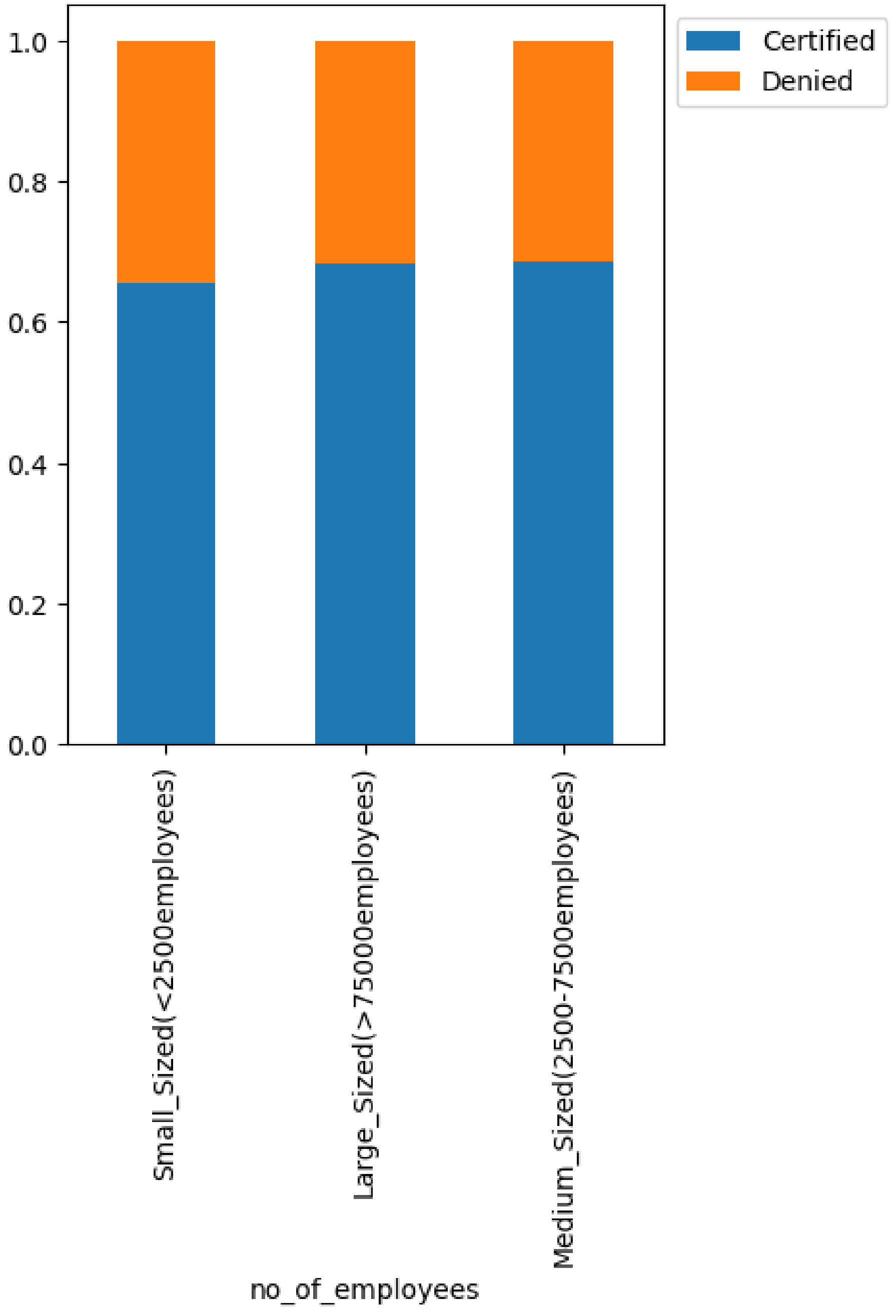
- After converting the ‘unit_of_wage’ column.
- The ‘Hour’ wage values are also seem to be calculated for weekly wage and entered.
- Since we consider the hourly wages are weekly wages got by the employees we are multiplying with 52 to get the yearly.
- So that all the units of the column `unit_of_wages` will be in years.



- After converting Hourly wages in ‘prevailing_wage’ * 52 we can see that the values are converted to annually.

No_of_employees column from continuous to categorical:

- From the above plot we can see that the there is a huge difference in number of cases that are certified and number of cases that are denied.
- Since the `no_of_employees` column doesn't have much relationship to the target variable we can split the continous data in `no_of_column` into 43 different category.

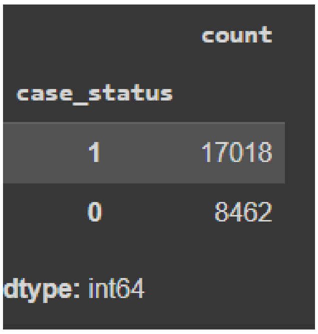


- After converting the numerical values to 3 different categorical values :
- Small_sized(<2500employees)
- Medium_sized(2500-7500employees)
- Large_sized(>7500employees)

Removing 'case_id' column:

- We are also removing the 'Case_id' column since there is no relation with target variable which is 'case_status'

Converting ‘case_status’ column into 0 and 1 from, ‘Certified’ and ‘Denied’:



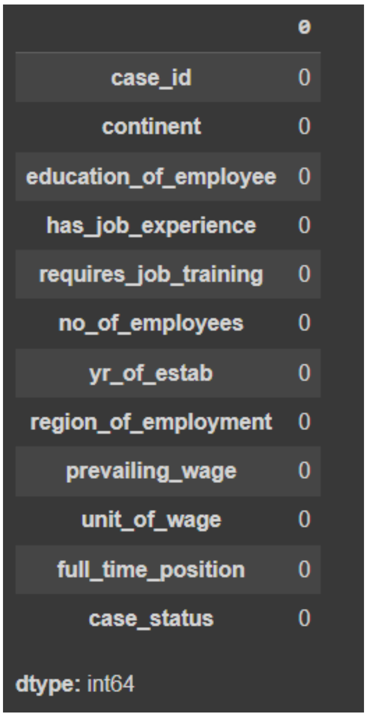
case_status	count
1	17018
0	8462

dtype: int64

- We are converting the Target variable to 0 and 1 because it will be simpler for the model to predict numerical variables.

Missing value Treatment:

- We don’t have any missing values or null values present in the data.



	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

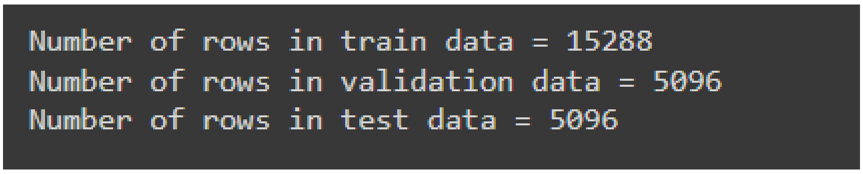
Duplicate value:

```
0
```

- Since, we don’t see any duplicate values we are good to go without any treatment for the duplicate.

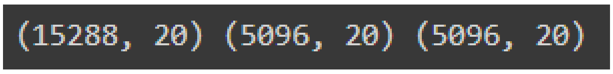
Checking and preventing data leakage:

- We are splitting data into Train set, Validation set and Test set. In order to avoid data leakage.
- We won’t be using the test data in testing the model we will be using it to do the final testing as production data.



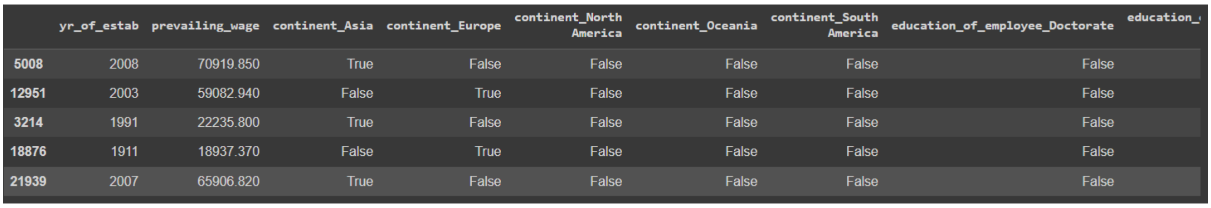
```
Number of rows in train data = 15288
Number of rows in validation data = 5096
Number of rows in test data = 5096
```

Preparing dummies for the categorical variables:



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

- We are preparing dummy variables for the categorical columns so that the model can understand the categorical variables.
- From the above image we can see that the columns size have been increased from 11 to 20 because for newly created dummy variables.



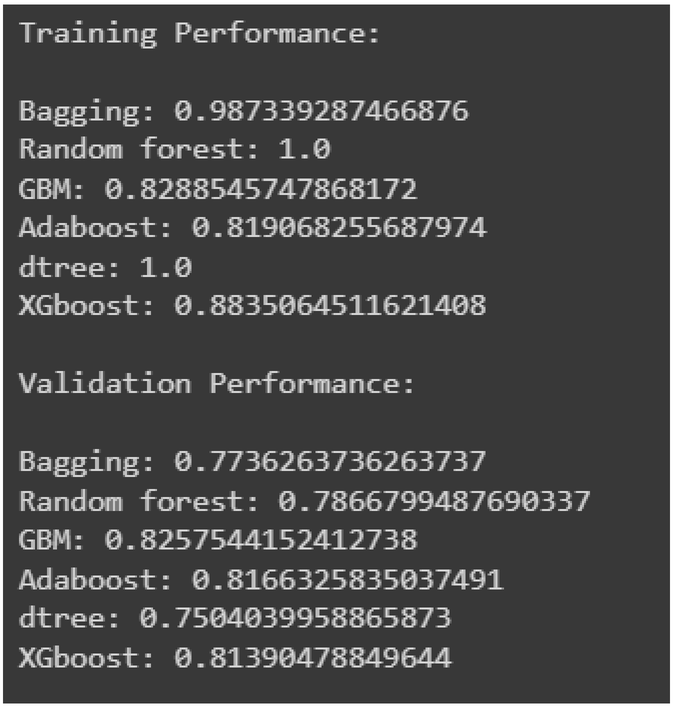
	yr_of_estab	prevailing_wage	continent_Asia	continent_Europe	continent_North America	continent_Oceania	continent_South America	education_of_employee_Doctorate	education_
5008	2008	70919.850	True	False	False	False	False	False	
12951	2003	59082.940	False	True	False	False	False	False	
3214	1991	22235.800	True	False	False	False	False	False	
18876	1911	18937.370	False	True	False	False	False	False	
21939	2007	65906.820	True	False	False	False	False	False	

Model Building:

- We are building Decision Tree, Bagging, Random Forest, Adaboost, Gradient Boost and XG Boost models here.
- These models belong to Ensemble models in which we create multiple weak machine learning models and getting their predictions and combine together to get one predictions, either we average (numerical) or get the mode (Categorical).

Evaluation metric we are choosing is 'F1 Score'.

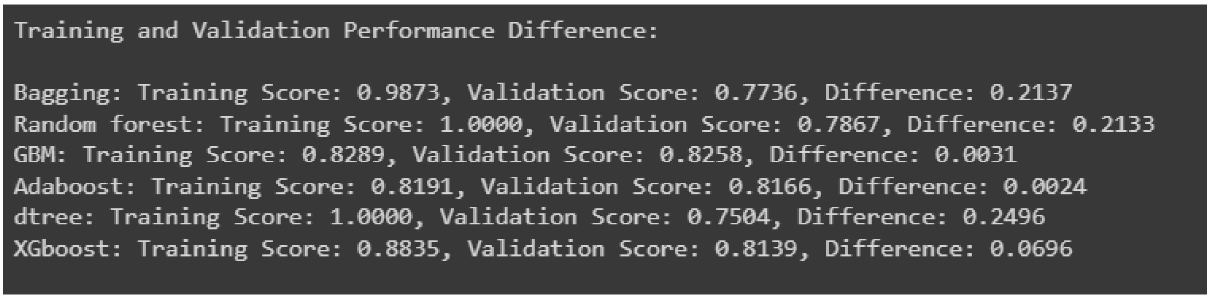
- The training performance of the models with f1_scrore as below:



```
Training Performance:

Bagging: 0.987339287466876
Random forest: 1.0
GBM: 0.8288545747868172
Adaboost: 0.819068255687974
dtree: 1.0
XGboost: 0.8835064511621408

Validation Performance:

Bagging: 0.7736263736263737
Random forest: 0.7866799487690337
GBM: 0.8257544152412738
Adaboost: 0.8166325835037491
dtree: 0.7504039958865873
XGboost: 0.81390478849644
```

- Since we are considering the prediction of both 'Certified' and 'Denied' values as a important we cannot be selecting recall or Precision as a metric we will be considering the f1_score as Evaluation metric.

We are checking the f1_ score for training and validation set and finding the difference between them to find which model is performing better:



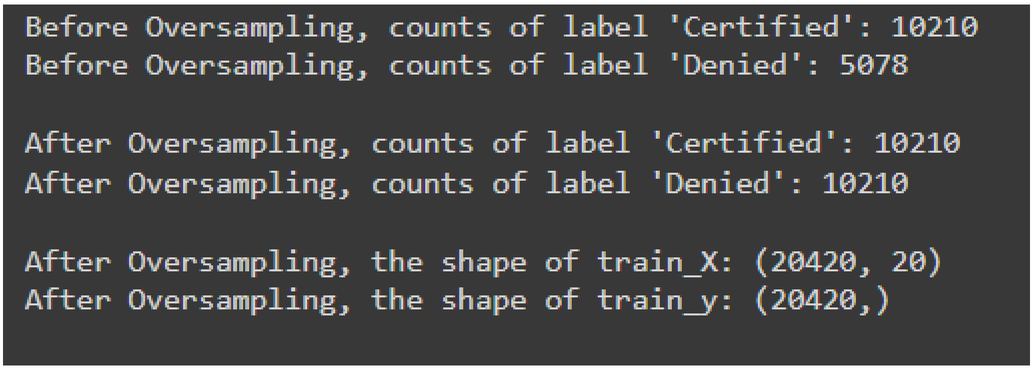
```
Training and Validation Performance Difference:

Bagging: Training Score: 0.9873, Validation Score: 0.7736, Difference: 0.2137
Random forest: Training Score: 1.0000, Validation Score: 0.7867, Difference: 0.2133
GBM: Training Score: 0.8289, Validation Score: 0.8258, Difference: 0.0031
Adaboost: Training Score: 0.8191, Validation Score: 0.8166, Difference: 0.0024
dtree: Training Score: 1.0000, Validation Score: 0.7504, Difference: 0.2496
XGboost: Training Score: 0.8835, Validation Score: 0.8139, Difference: 0.0696
```

From the above result we can clearly tell,
- GBM model is performing well with the Training Score: 0.8289, Validation Score: 0.8258, Difference: 0.0031.
- Followed by AdaBoost model Training Score: 0.8191, Validation Score: 0.8166, Difference: 0.0024.

- XGBoost also is performing well in train and validation data with Training Score: 0.8835, Validation Score: 0.8139, Difference: 0.0696.

Building model on Oversampled data:

- We are building the model on Oversampling data with SMOTE function.



```
Before Oversampling, counts of label 'Certified': 10210
Before Oversampling, counts of label 'Denied': 5078

After Oversampling, counts of label 'Certified': 10210
After Oversampling, counts of label 'Denied': 10210

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

From the above image we can see that the count before oversampling is:

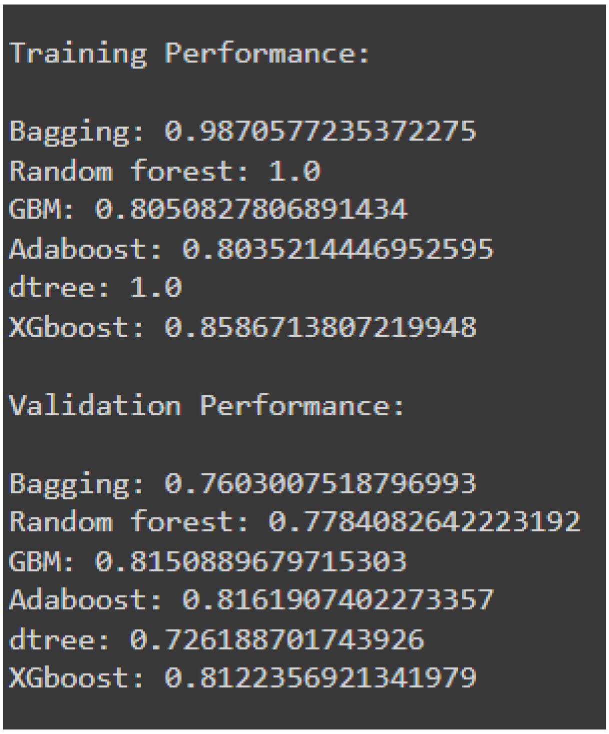
- Certified: 10210
- Denied: 5078

Count after Oversampling:

- Certified: 10210
- Denied: 10210
-

The function increased the data points of the minority column to the data points of majority column.

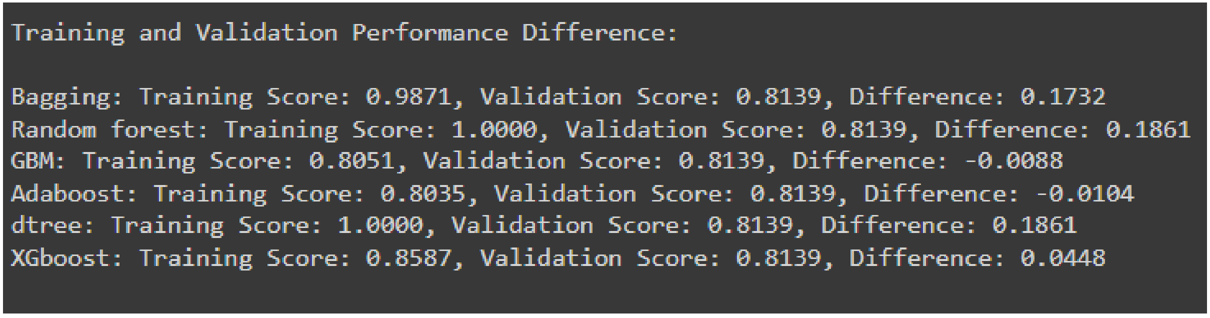
The F1 scores and performance for the models with oversampled data are as follows:



```
Training Performance:

Bagging: 0.9870577235372275
Random forest: 1.0
GBM: 0.8050827806891434
Adaboost: 0.8035214446952595
dtree: 1.0
XGboost: 0.8586713807219948

Validation Performance:

Bagging: 0.7603007518796993
Random forest: 0.7784082642223192
GBM: 0.8150889679715303
Adaboost: 0.8161907402273357
dtree: 0.726188701743926
XGboost: 0.8122356921341979
```

We can see there is slight difference in model performances after oversampling the data.

Let’s check the performance differences between training and validation:



```
Training and Validation Performance Difference:

Bagging: Training Score: 0.9871, Validation Score: 0.8139, Difference: 0.1732
Random forest: Training Score: 1.0000, Validation Score: 0.8139, Difference: 0.1861
GBM: Training Score: 0.8051, Validation Score: 0.8139, Difference: -0.0088
Adaboost: Training Score: 0.8035, Validation Score: 0.8139, Difference: -0.0104
dtree: Training Score: 1.0000, Validation Score: 0.8139, Difference: 0.1861
XGboost: Training Score: 0.8587, Validation Score: 0.8139, Difference: 0.0448
```

From the above image we can see that,

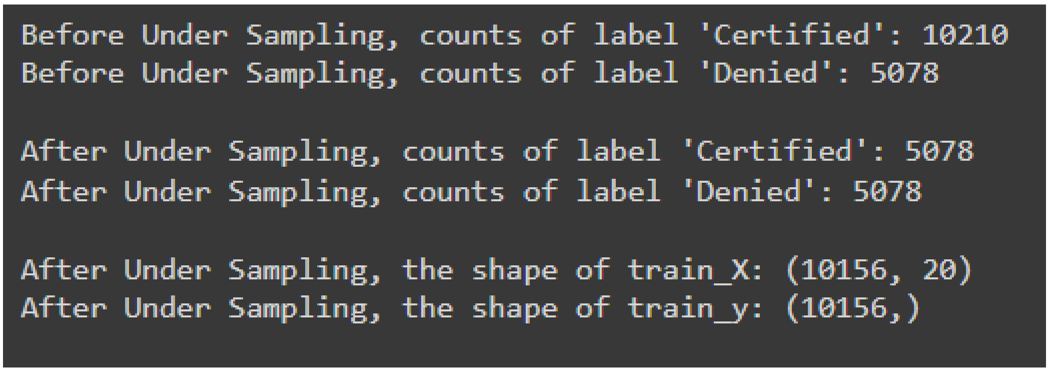
- After Oversampling of dataset, Adaboost model is performing very well with Training Score: 0.8035, Validation Score: 0.8139, Difference: -0.0104.

- Followed by GBM witrh the Training Score: 0.8051, Validation Score: 0.8139, Difference: -0.0088

- Next XGBoost have better performance with Training Score: 0.8587, Validation Score: 0.8139, Difference: 0.0448.

Building Model with Undersampled data:

- We are building the model on Underampling data with RandomUnderSampler function.



```
Before Under Sampling, counts of label 'Certified': 10210
Before Under Sampling, counts of label 'Denied': 5078

After Under Sampling, counts of label 'Certified': 5078
After Under Sampling, counts of label 'Denied': 5078

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

From the above image we can see that the count before undersampling is:

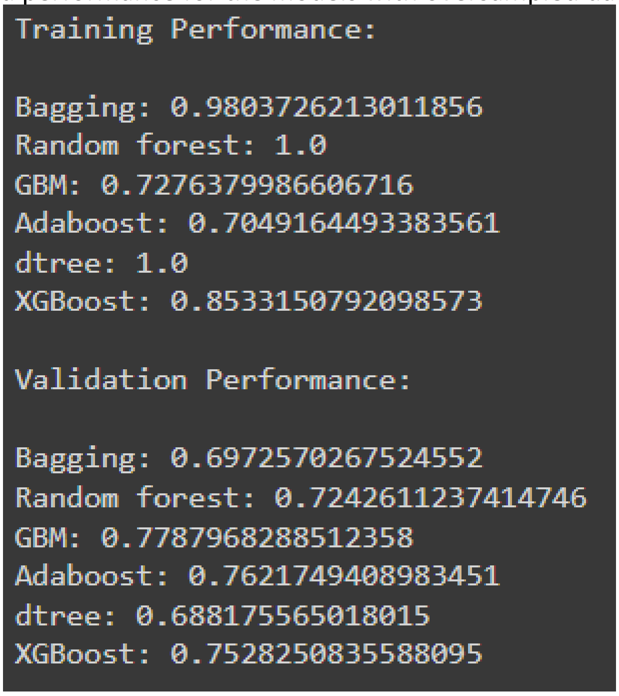
- Certified: 10210
- Denied: 5078

Count after Oversampling:

- Certified: 5078
- Denied: 5078

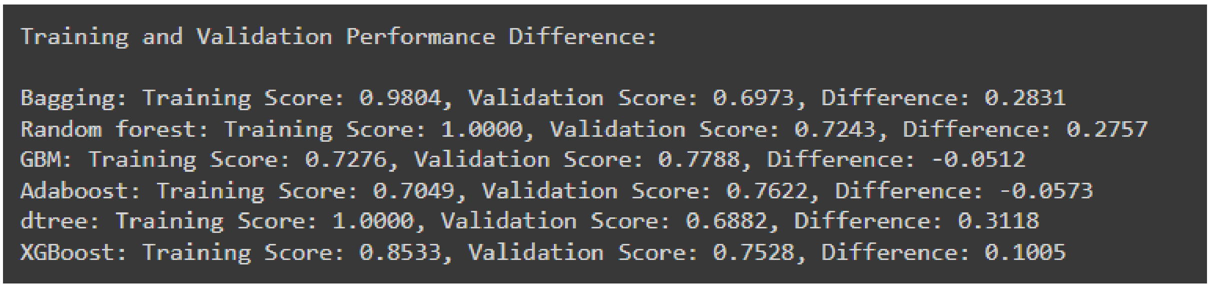
The function reduced the data points majority column to the data points of minority column.

The F1 scores and performance for the models with oversampled data are as follows:



```
Training Performance:

Bagging: 0.9803726213011856
Random forest: 1.0
GBM: 0.7276379986606716
Adaboost: 0.7049164493383561
dtree: 1.0
XGBoost: 0.8533150792098573

Validation Performance:

Bagging: 0.6972570267524552
Random forest: 0.7242611237414746
GBM: 0.7787968288512358
Adaboost: 0.7621749408983451
dtree: 0.688175565018015
XGBoost: 0.7528250835588095
```

We can see there is difference in model performances after undersampling the data.



```
Training and Validation Performance Difference:

Bagging: Training Score: 0.9804, Validation Score: 0.6973, Difference: 0.2831
Random forest: Training Score: 1.0000, Validation Score: 0.7243, Difference: 0.2757
GBM: Training Score: 0.7276, Validation Score: 0.7788, Difference: -0.0512
Adaboost: Training Score: 0.7049, Validation Score: 0.7622, Difference: -0.0573
dtree: Training Score: 1.0000, Validation Score: 0.6882, Difference: 0.3118
XGBoost: Training Score: 0.8533, Validation Score: 0.7528, Difference: 0.1005
```

From the above image we can see that,

- After undersampling the data, GBM seems to be a good model with the Training Score: 0.7276, Validation Score: 0.7788, Difference: -0.0512.
- Followed by adaboost with the Training Score: 0.7049, Validation Score: 0.7622, Difference: -0.0573.
- Which is followed by XGBoost with Training Score: 0.8533, Validation Score: 0.7528, Difference: 0.1005.

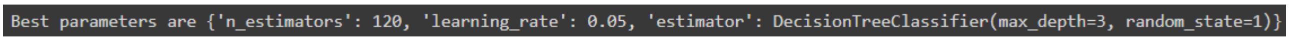
Commenting and Choosing the models after Oversampling and Undersampling:

- We have created 18 models till now with Original data, Oversapmled data and Underdsampled data
- After building 18 models we can clearly see that the GBM, adaboost and XGBoost model trained on Original data, Adaboost, GBM and XGB trained on Oversampled data and GBM, Adaboost and XGBoost model trained on undersampled data exhibit better performance on Training and validation dataset.
- Sometimes models might overfit after undersampling and oversampling, so it's better to tune the models to get a generalized performance
- We will tune these 3 models using the same data as we trained them before.

HyperTuning:

Hypertuning AdaBoost model with Original Dataset:

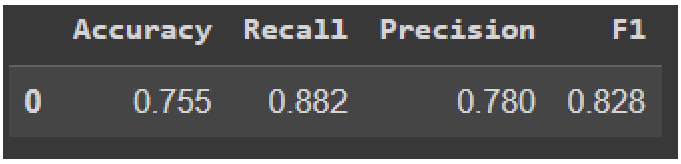
After Tuning our Adaboost model which we created using the Hyperparameters we got by using randomisedCV function the best hyperparameters we got are:



```
Best parameters are {'n_estimators': 120, 'learning_rate': 0.05, 'estimator': DecisionTreeClassifier(max_depth=3, random_state=1)}
```

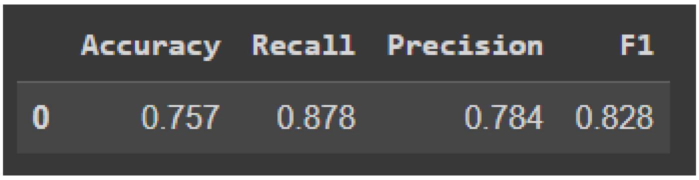
The performance of the model are as follows:

Performance of the model on Training dataset:



	Accuracy	Recall	Precision	F1
0	0.755	0.882	0.780	0.828

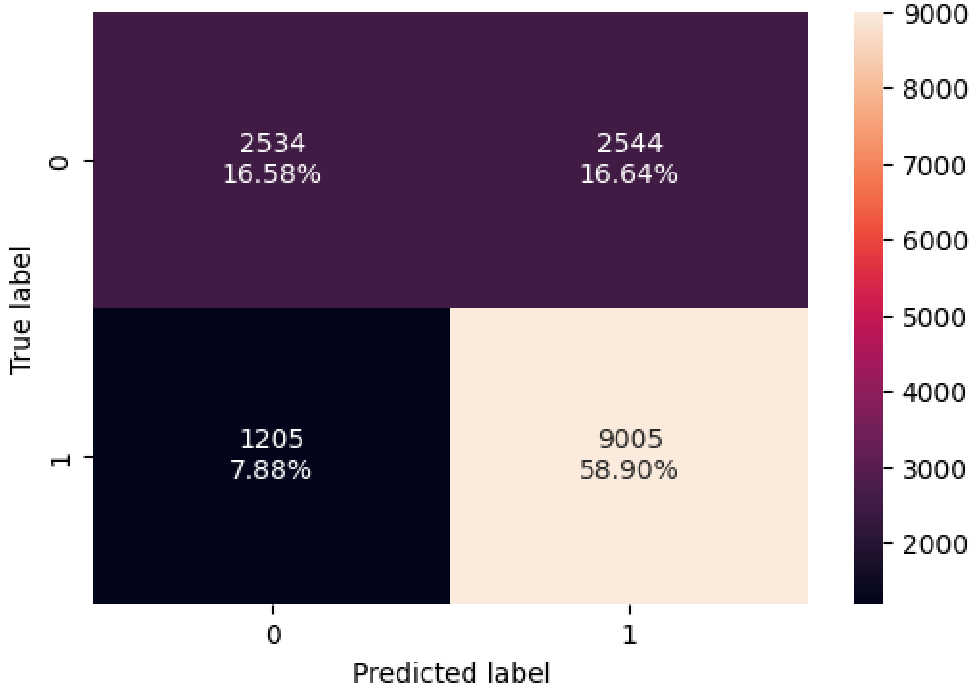
Performance of the model in Validation dataset:



	Accuracy	Recall	Precision	F1
0	0.757	0.878	0.784	0.828

Both the F1 score of train and validation dataset are similar which is 0.828 and the model seems to perform well on validation dataset on other metrics like recall and precision.

Confusion matrix for train data:



- We have created Confusion matrix for train data
- We have

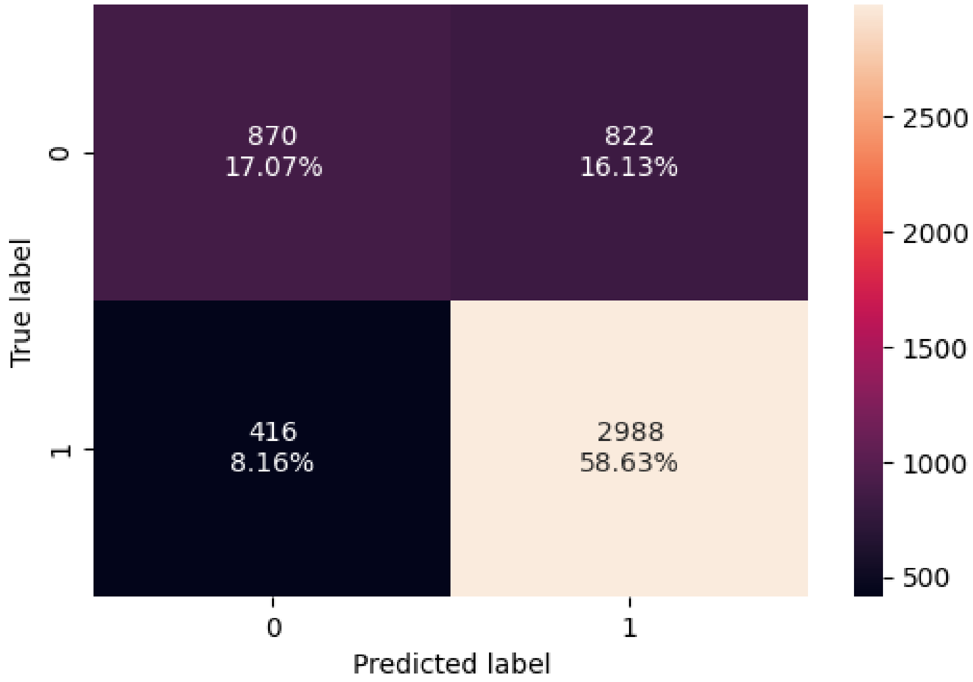
  TP = 9005(58.90%),

  TN = 2535(16.58%),

  FP = 2544(16.64%),

  FN = 1205(7.88%)

Confusion matrix for validation data:



- We have created Confusion matrix for validation data
- We have

    TP = 2988(58.63%),

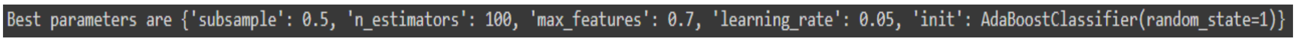
    TN = 870(17.07%),

    FP = 822(16.13%),

    FN = 416(8.16%)

Hypertuning Gradient Boosting model with Original dataset:

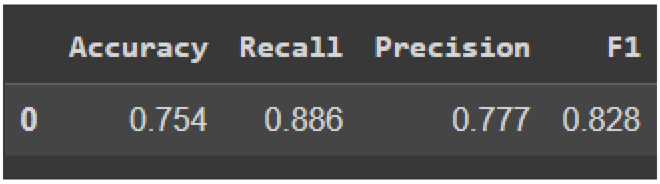
After Tuning our Gradient Boost model which we created using the Hyperparameters we got by using randomisedCV function the best Hyperparameters are:



```
Best parameters are {'subsample': 0.5, 'n_estimators': 100, 'max_features': 0.7, 'learning_rate': 0.05, 'init': AdaBoostClassifier(random_state=1)}
```

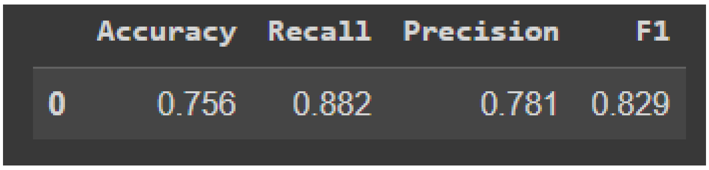
The performance of the model are as follows:

Performance of the model on Training dataset:



	Accuracy	Recall	Precision	F1
0	0.754	0.886	0.777	0.828

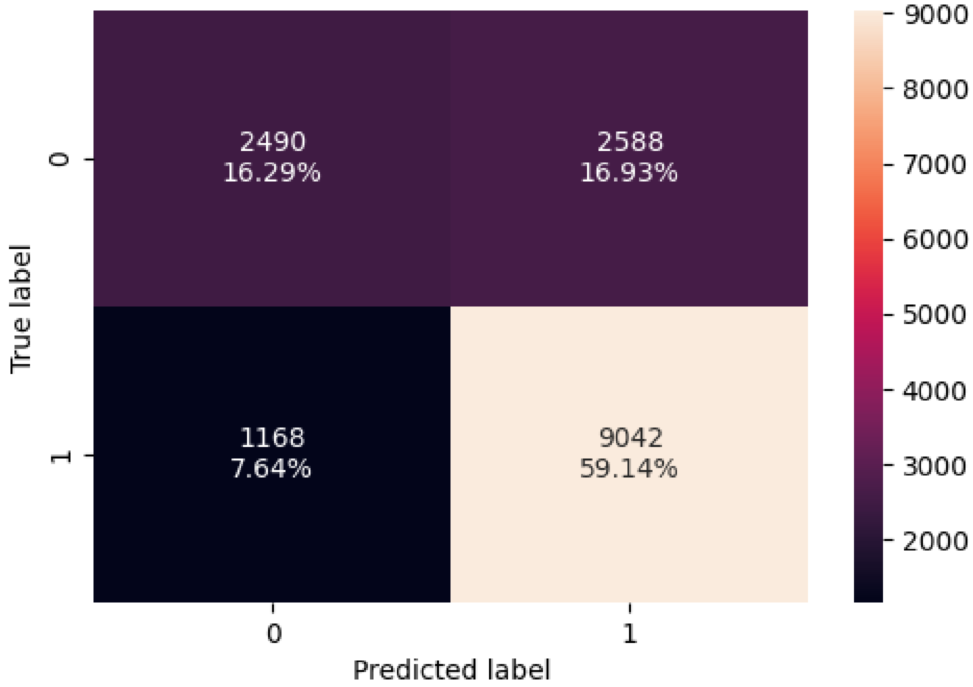
Performance of the model in Validation dataset:



	Accuracy	Recall	Precision	F1
0	0.756	0.882	0.781	0.829

Both the F1 score of train and validation dataset are good and the model seems to perform well on validation dataset on F1 Score: 0.829 which is more compare to training performance and on other metrics like recall and precision also perform well.

Confusion matrix for train data:



- We have created Confusion matrix for train data
- We have

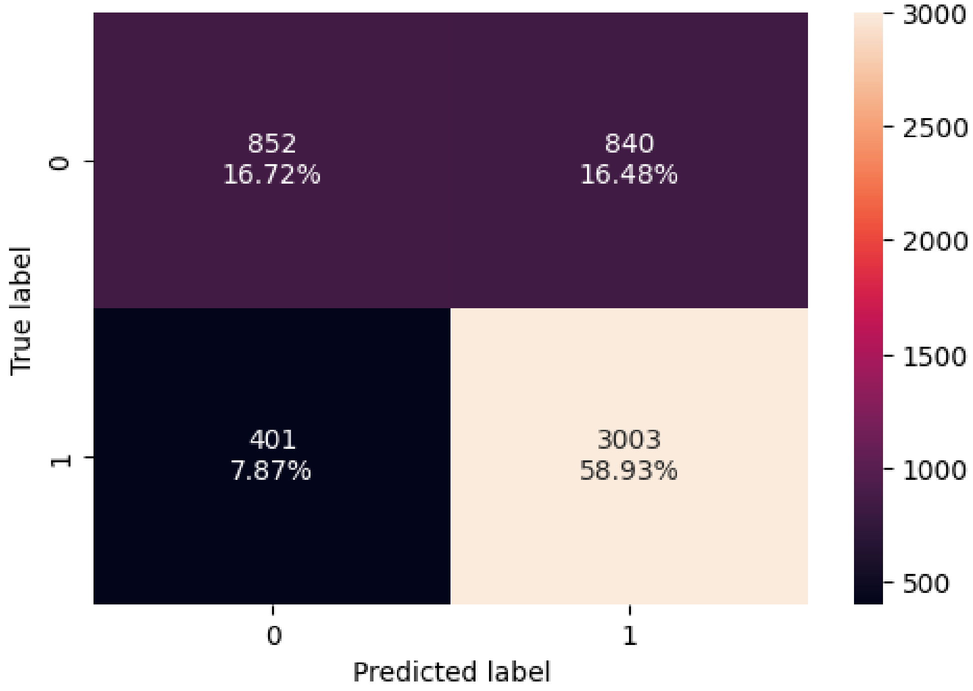
    TP = 9042(59.14%),

    TN = 2490(16.29%),

    FP = 2588(16.93%),

    FN = 1168(7.64%)

Confusion matrix for Validation data:



- We have created Confusion matrix for Validation data
- We have

  TP = 3003(58.93%),

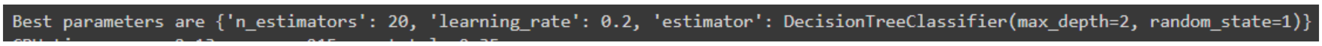
  TN = 852(16.72%),

  FP = 840(16.48%),

  FN = 401(7.87%)

Hypertuning Adaboost model on Oversampled data:

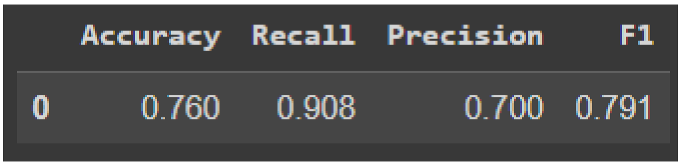
After Tuning our Adaboost model on Oversampled data which we created using the Hyperparameters we got by using randomisedCV function the best Hyperparameters are:



```
Best parameters are {'n_estimators': 20, 'learning_rate': 0.2, 'estimator': DecisionTreeClassifier(max_depth=2, random_state=1)}
```

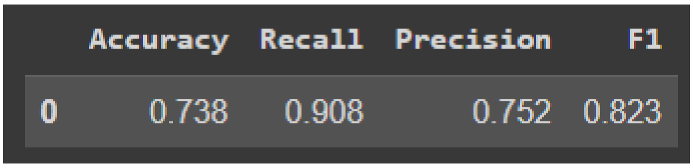
The performance of the model are as follows:

Performance of the model on Training dataset:



	Accuracy	Recall	Precision	F1
0	0.760	0.908	0.700	0.791

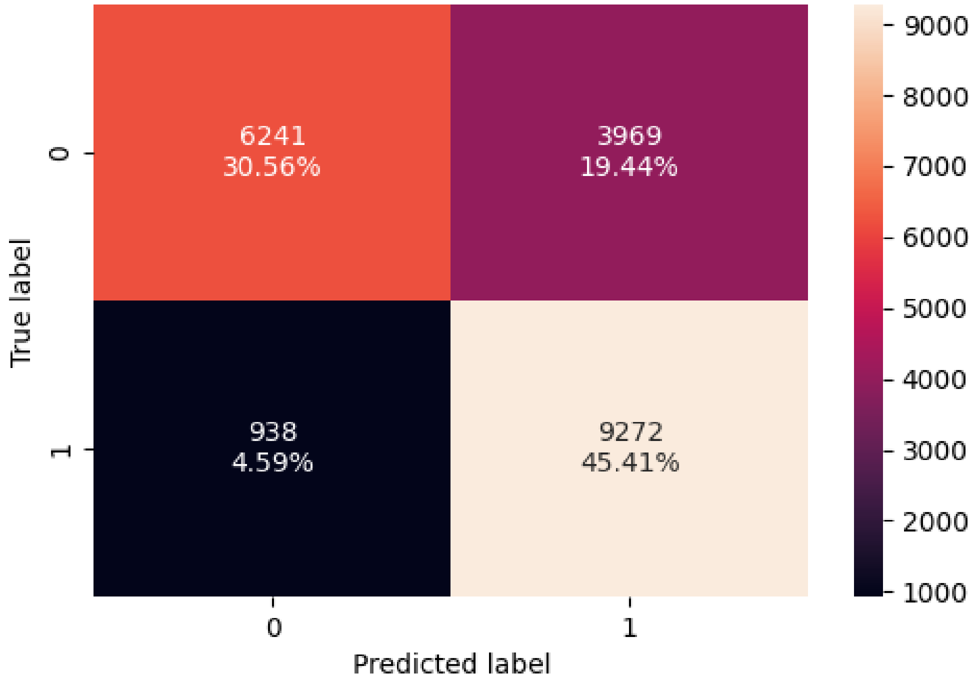
Performance of the model in Validation dataset:



	Accuracy	Recall	Precision	F1
0	0.738	0.908	0.752	0.823

Both the F1 score of train and validation dataset are good and the model seems to perform increased on validation dataset on F1 Score: 0.823 and on trainn set F1 score: 0.791 which is more compare to training performance and on other metrics like recall and precision also perform well but the accuracy value: 0.760(train) and 0.738(test) is less compared other models.

Confusion matrix for train data:



- We have created Confusion matrix for train data
- We have

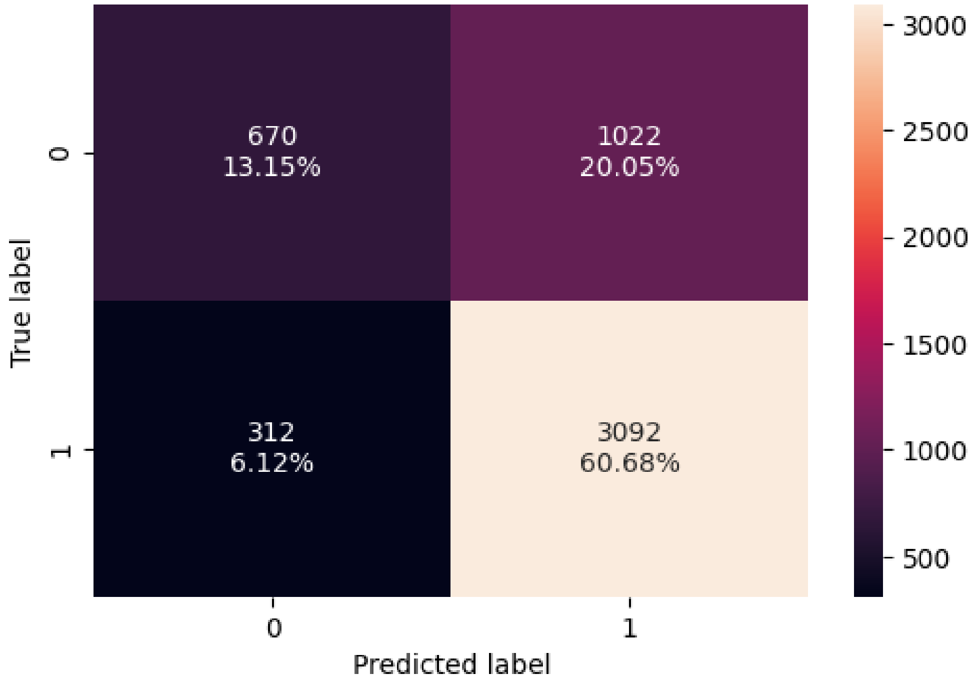
TP = 9272(45.41%),

TN = 6241(30.56%),

FP = 3969(19.44%),

FN = 938(4.59%)

Confusion matrix for validation data:



- We have created Confusion matrix for Validation data
- We have

  TP = 3092(60.68%),

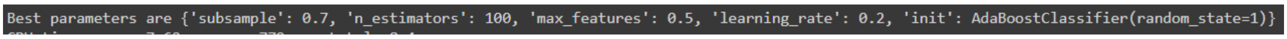
  TN = 670(13.15%),

  FP = 1022(20.05%),

  FN = 312(6.12%)

Hypertuning Gradient Boost model on Oversampled data:

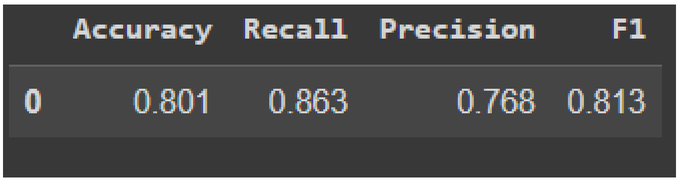
After Tuning our Gradient Boost model on Oversampled data which we created using the Hyperparameters we got by using randomisedCV function the best Hyperparameters are:



```
Best parameters are {'subsample': 0.7, 'n_estimators': 100, 'max_features': 0.5, 'learning_rate': 0.2, 'init': AdaBoostClassifier(random_state=1)}
```

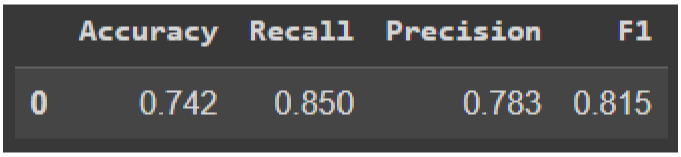
The performance of the model are as follows:

Performance of the model on Training dataset:



	Accuracy	Recall	Precision	F1
0	0.801	0.863	0.768	0.813

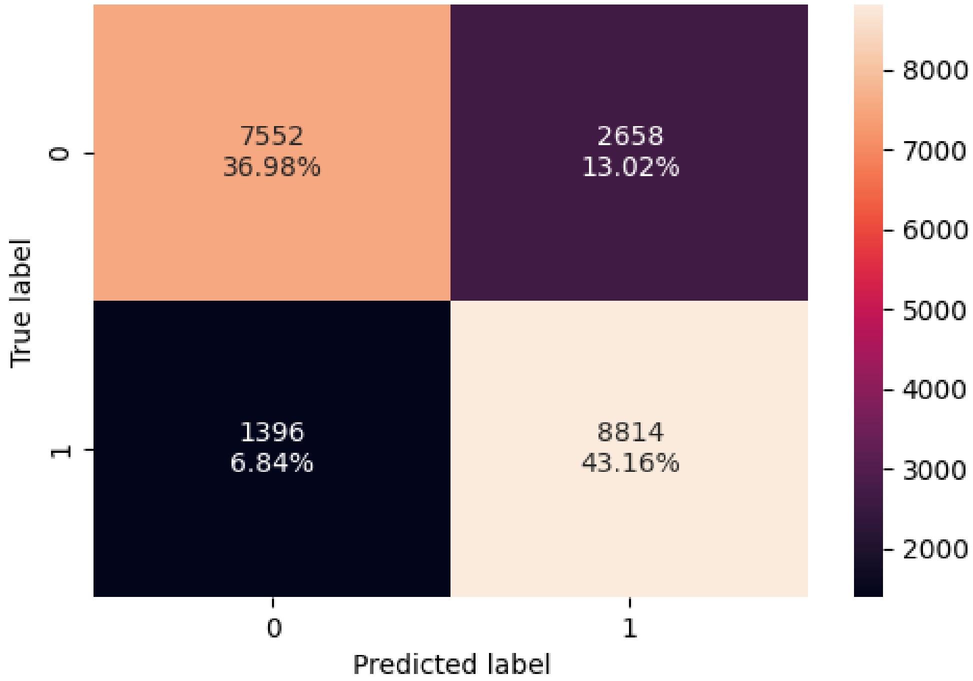
Performance of the model in Validation dataset:



	Accuracy	Recall	Precision	F1
0	0.742	0.850	0.783	0.815

Both the F1 score of train and validation dataset are good and the model seems to perform increased on validation dataset on F1 Score: 0.815 and on train set F1 score: 0.813 which is more compare to training performance and on other metrics like recall and precision also perform well but the accuracy value: 0.801(train) and 0.742(test) got reduced.

Confusion matrix for train data:



- We have created Confusion matrix for train data

- We have

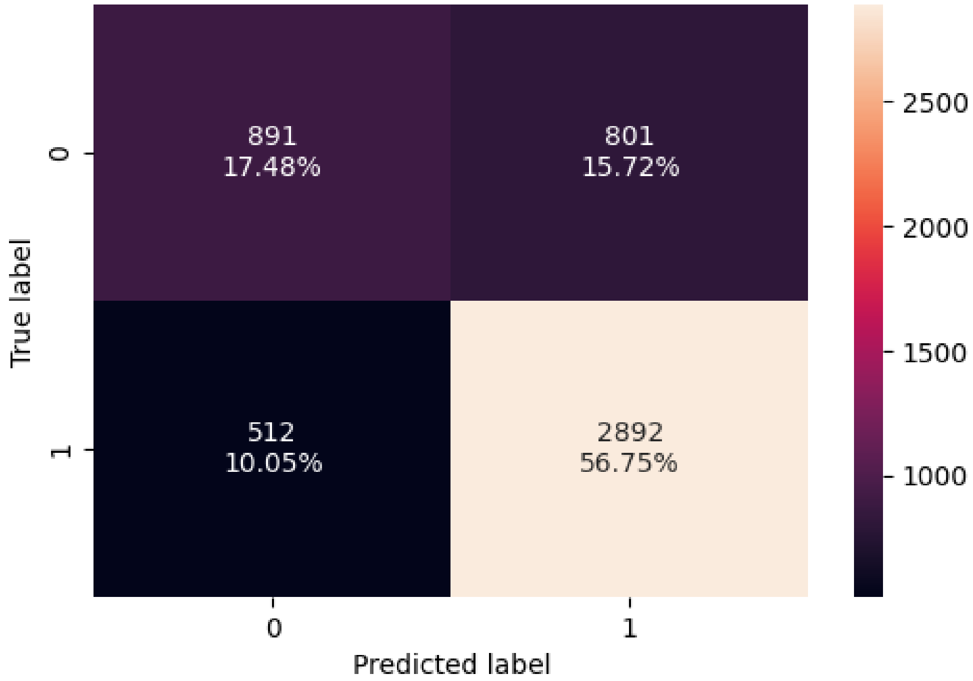
  TP = 8814(43.16%),

  TN = 7552(36.98%),

  FP = 2658(13.02%),

  FN = 1396(6.84%)

Confusion matrix for validation data:



- We have created Confusion matrix for Validation data
- We have

    TP = 2892(56.75%),

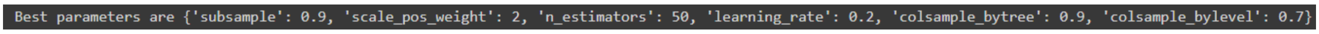
    TN = 891(17.48%),

    FP = 801(15.72%),

    FN = 512(10.05%)

Hypertuning XG Boost model on Original Dataset:

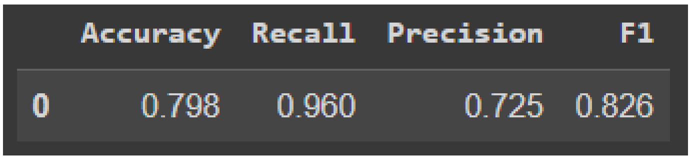
After Tuning our XG Boost model on Original data which we created using the Hyperparameters we got by using randomisedCV function the best Hyperparameters are:



```
Best parameters are {'subsample': 0.9, 'scale_pos_weight': 2, 'n_estimators': 50, 'learning_rate': 0.2, 'colsample_bytree': 0.9, 'colsample_bylevel': 0.7}
```

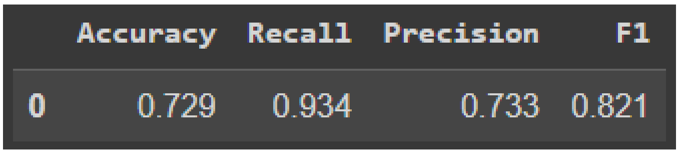
The performance of the model are as follows:

Performance of the model on Training dataset:



	Accuracy	Recall	Precision	F1
0	0.798	0.960	0.725	0.826

Performance of the model in Validation dataset:

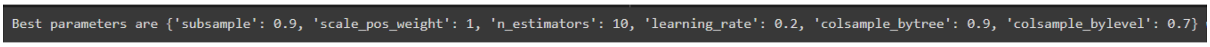


	Accuracy	Recall	Precision	F1
0	0.729	0.934	0.733	0.821

Both the F1 score of train and validation dataset are good and the model perfomance on trainin g dataset with F1 Score: 0.826 and on validation set F1 score: 0.821 which is good and on other metrics like recall and precision also perform well but the accuracy value: 0.798(train) and 0.729(test) got reduced.

Hypertuning XG Boost model on Oversampled Dataset:

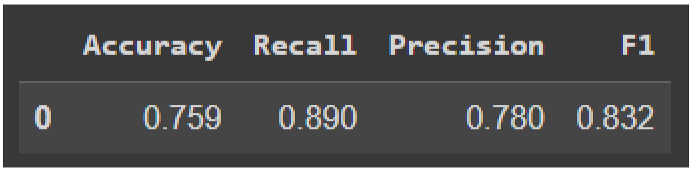
After Tuning our XG Boost model on Oversampled data which we created using the Hyperparameters we got by using randomisedCV function the best Hyperparameters are:



```
Best parameters are {'subsample': 0.9, 'scale_pos_weight': 1, 'n_estimators': 10, 'learning_rate': 0.2, 'colsample_bytree': 0.9, 'colsample_bylevel': 0.7}
```

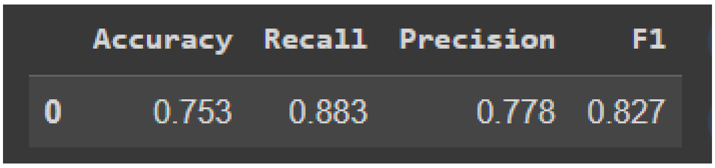
The performance of the model are as follows:

Performance of the model on Training dataset:



	Accuracy	Recall	Precision	F1
0	0.759	0.890	0.780	0.832

Performance of the model in Validation dataset:

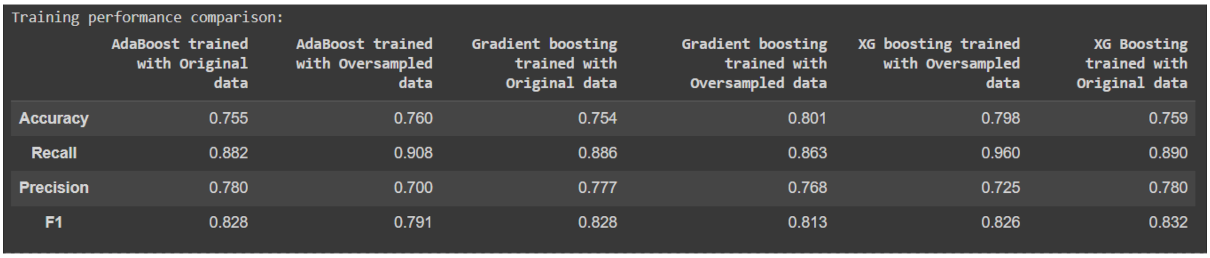


	Accuracy	Recall	Precision	F1
0	0.753	0.883	0.778	0.827

Both the F1 score of train and validation dataset are good and the model perfomance on trainin g dataset with F1 Score: 0.832 and on validation set F1 score: 0.827 which is good and on other metrics like recall and precision also perform well but the accuracy value: 0.759(train) and 0.753(test) is less when compared to other models.

Checking the Overall Model performance and Selecting the model:

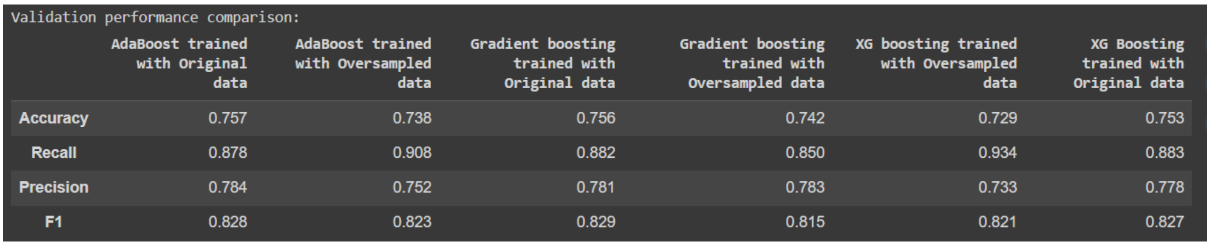
Performance comparision of model on Training dataset,



Training performance comparison:

	AdaBoost trained with Original data	AdaBoost trained with Oversampled data	Gradient boosting trained with Original data	Gradient boosting trained with Oversampled data	XG boosting trained with Oversampled data	XG Boosting trained with Original data
Accuracy	0.755	0.760	0.754	0.801	0.798	0.759
Recall	0.882	0.908	0.886	0.863	0.960	0.890
Precision	0.780	0.700	0.777	0.768	0.725	0.780
F1	0.828	0.791	0.828	0.813	0.826	0.832

Performance comparision of model on Validation dataset,



Validation performance comparison:

	AdaBoost trained with Original data	AdaBoost trained with Oversampled data	Gradient boosting trained with Original data	Gradient boosting trained with Oversampled data	XG boosting trained with Oversampled data	XG Boosting trained with Original data
Accuracy	0.757	0.738	0.756	0.742	0.729	0.753
Recall	0.878	0.908	0.882	0.850	0.934	0.883
Precision	0.784	0.752	0.781	0.783	0.733	0.778
F1	0.828	0.823	0.829	0.815	0.821	0.827

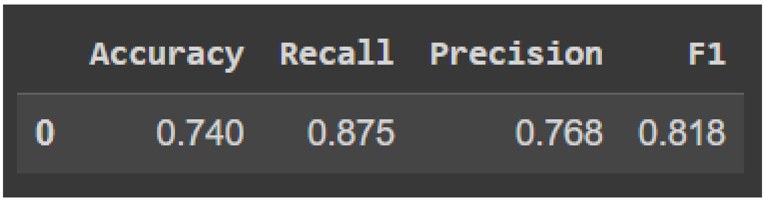
- From the Performance of Training and Validation we can clearly conclude that the Adaboost model for original data and Gradient Boosting for Original data. Are performing better and more suitable for our dataset.

- From scores got by Hypertuning models with Oversampled and undersampled in training and validation dataset we can conclude that Adaboost on Original data and Gradient boost on Original dataset is having better and similar performance and with better f1_score.

- So we are checking on the testing for these two models for Test dataset.

Testing the Adaboost model built with Original dataset:

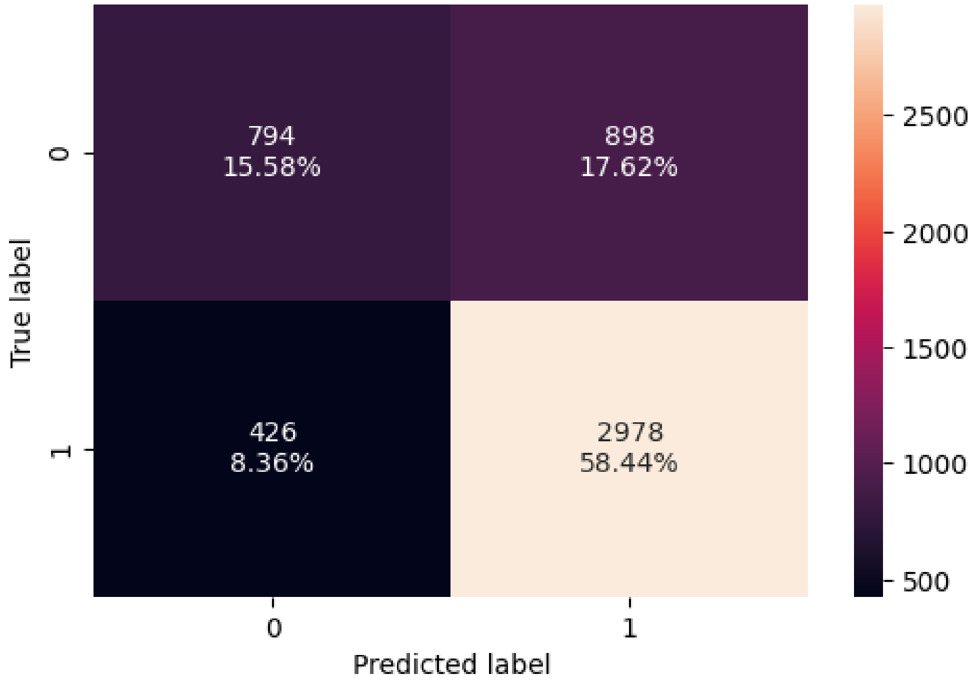
Performance of model on Testing data:



	Accuracy	Recall	Precision	F1
0	0.740	0.875	0.768	0.818

- Adaboost model on Original data has ~81% of f1_score on test data.
- This performance is in line with what we achieved with this model on the train and validation sets
- So, this is a generalized model

Confusion Matrix:



- We have created Confusion matrix for test data for Adaboost
- We have

  TP = 2678(58.44%),

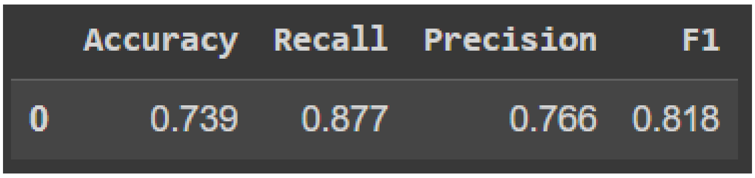
  TN = 794(15.58%),

  FP = 898(17.62%),

  FN = 426(8.36%)

Testing the Gradient Boost model built with Original dataset:

Performance of model on Testing data:



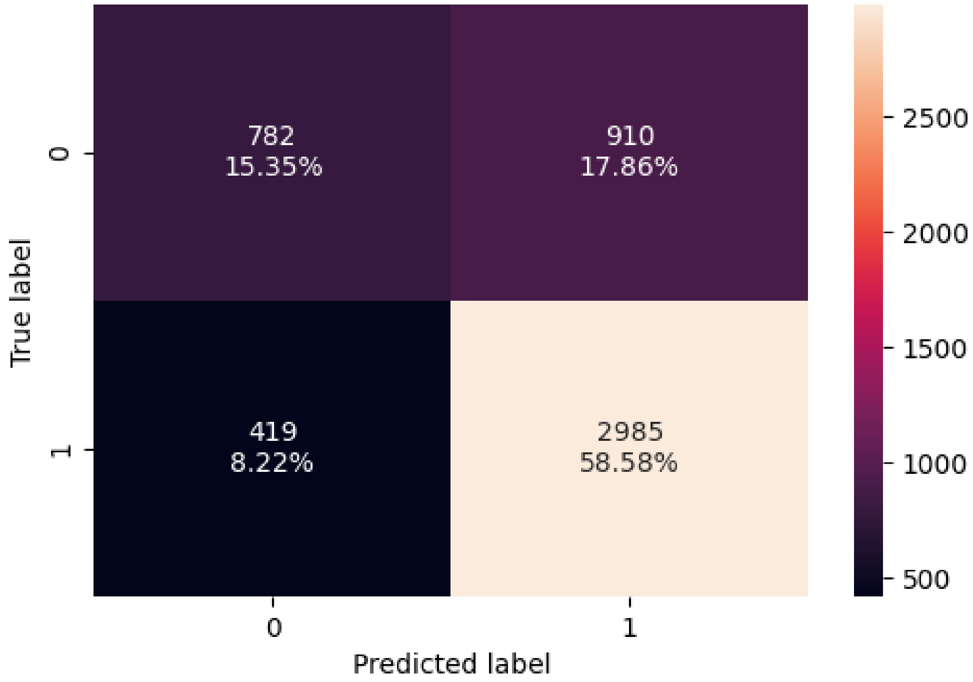
	Accuracy	Recall	Precision	F1
0	0.739	0.877	0.766	0.818

- Gradient boosting model on Original dataset also has ~81% of f1_score which is similar to what we got on adaboost.

- This performance is also in line with what we achieved with this model on the train and validation sets.

- So, we also consider this as an generalized model.

Confusion matrix:



- We have creeated Confusion matrix for test data

- We have

  TP = 2985(58.58%),

  TN = 782(115.35%),

  FP = 910(17.86%),

  FN = 419(8.22%)

Final model selection:

- The final model that we suggest is Adaboost model which we built for Original dataset. eventhough both model have similar F1 score, with accuracy, recall and precision we can conclude we are going with tuned_adb (Adaboost with Original data model).

Recommendation:

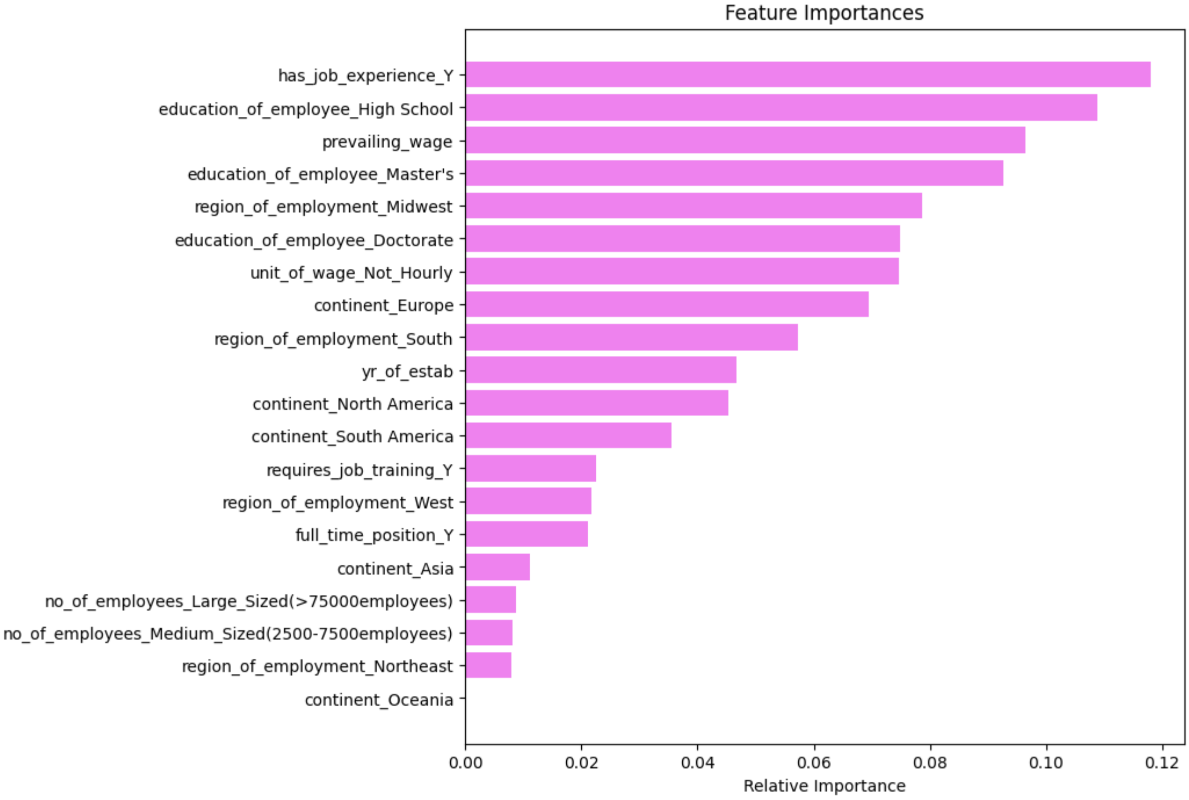
- Since the F1 score is the same, and there's no difference in this key performance metric, you can choose either model based on secondary factors like:
- Training speed or complexity (if one model is faster or easier to interpret).
- Business context or algorithm preference (if one model is more preferred due to certain constraints or familiarity).

Conclusion:

- Both models are equally good in terms of F1 score, so the choice can be made based on non-performance factors like efficiency or complexity.

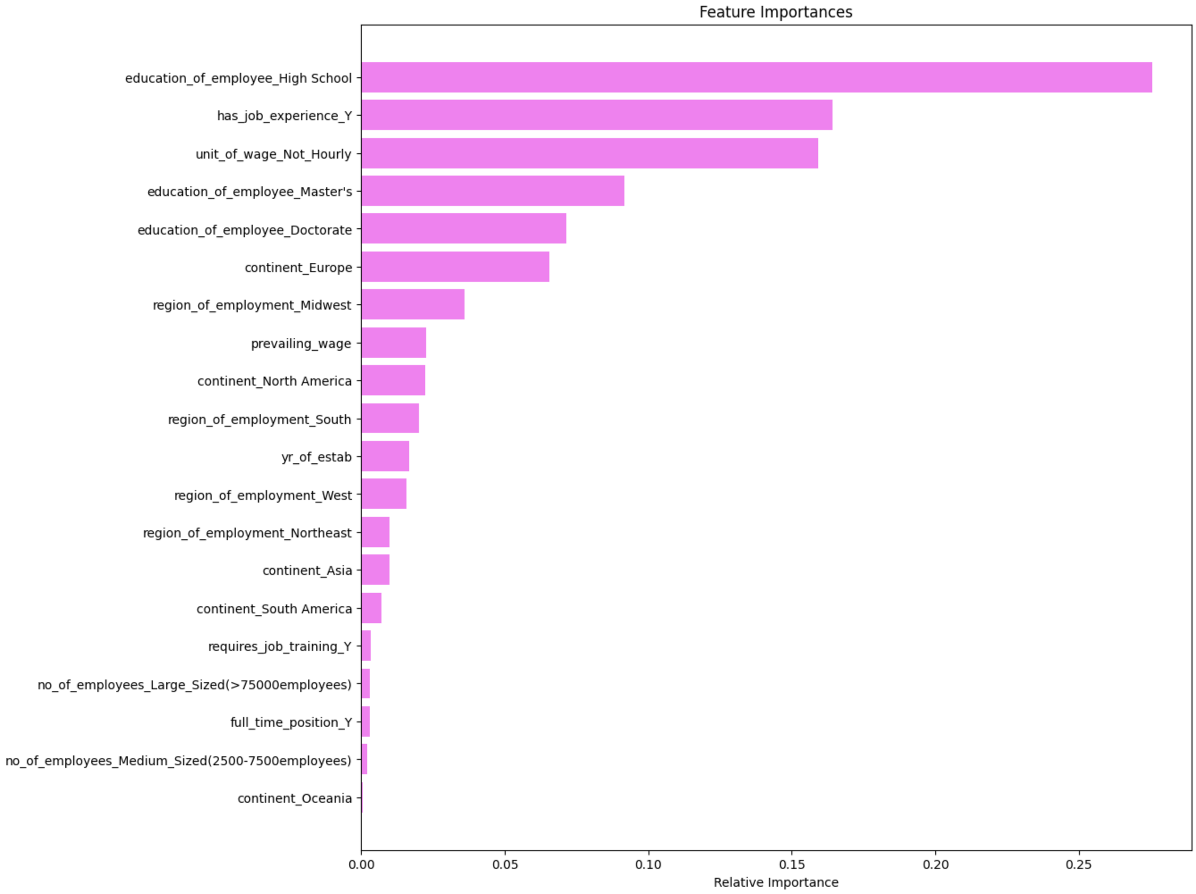
Feature Importance:

Important features for Adaboost model fo original dataset:



- We can see that `has_experience_Y`, `education_of_employee_high_school`, `prevailing_wage`, `education_of_employee_master's` are most important features which is important in model predictions for adaboost model.

Important features for Gradient Boost model fo original dataset:



- We can see that `education_of_employee_high_school`, `has_experience_Y`, `unit_of_wage_Not_Hourly`, `education_of_employee_master's` are most important features which is important in model predictions for Gradient Boost model.

Actionable Insights for OFLC and Business Communities in the US:

Education and Job Experience as Key Factors:

- Master's degree holders have a visa certification rate of 78.6%, while employees with a Doctorate see even better outcomes at 87.2%.

- Job experience also plays a critical role, with applicants who have prior experience seeing a 74.5% certification rate compared to only 56.1% for those without.

- Recommendation: US companies should focus on recruiting talent with higher educational qualifications and job experience, as these factors significantly improve the chances of visa certification.

Wage Structures Impact Certification:

- Employees with non-hourly wages (monthly or yearly) have a certification ra11te of 69.8%, compared to 34.6% for hourly wage earners. This discrepancy highlights the importance of stable, long-term compensation in visa approvals.

- Recommendation: US employers should offer non-hourly wage structures when possible to foreign workers. This aligns with the statutory requirements for fair wages and increases visa approval rates.

Geographical Focus for Better Visa Success:

- Applicants from Europe and Asia have higher success rates (79.2% and 65.3%, respectively). In contrast, regions like Africa and Oceania see lower approval rates.

- Recommendation: US businesses seeking foreign talent should target regions like Europe and Asia, where visa certifications are more likely, while refining applications for regions with lower success rates.

Company Size and Visa Approvals:

- Large companies (over 7,500 employees) have a 71.7% certification rate, significantly higher than smaller firms.

- Recommendation: Small and medium-sized enterprises (SMEs) can enhance their visa approval chances by emphasizing financial stability and growth potential in their applications. Partnering with larger firms or providing more robust support to applicants can further improve outcomes.

Business Recommendations for EasyVisa and OFLC:

Leverage Data to Streamline Approvals:

- Use Machine Learning models to identify key applicant profiles—such as those with advanced degrees and prior experience—that are more likely to receive visa certifications, streamlining the review process and reducing the burden on OFLC.

Focus on Wages and Stability:

- Encourage employers to offer competitive, non-hourly wage packages, as these are directly linked to higher visa approval rates. This supports compliance with INA's requirements for fair wages while also benefiting the employer's success in hiring foreign talent.

Optimize Recruitment Based on Visa Success Regions:

- Prioritize talent acquisition from regions like Asia and Europe, where historical data shows higher certification rates, while addressing the challenges of lower-certification regions through improved application processes.
- By implementing these insights, EasyVisa can enhance its ability to provide data-driven solutions, helping the OFLC and US businesses more effectively identify and certify qualified foreign talent.