```
import warnings
warnings.filterwarnings('ignore')
from statsmodels.tools.sm_exceptions import ConvergenceWarning
warnings.simplefilter('ignore', ConvergenceWarning)
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows',200)
import statsmodels.api as sms
from statsmodels.stats.outliers_influence import variance_inflation_factor
import statsmodels.api as sm
from statsmodels.tools.tools import add_constant
from sklearn.linear_model import LogisticRegression


from sklearn.metrics import(
    f1_score,
    accuracy_score,
    precision_score,
    recall_score,
    confusion_matrix,
    roc_auc_score,
    roc_curve,
    ConfusionMatrixDisplay,
    precision_recall_curve
)
```

```
who=pd.read_csv('/content/logistic_regression.csv')
data=who.copy()
data.head()
```

	age	workclass	fnlwgt	education	education_no_of_years	marital_status	occupation	race	sex	capital_gain	capital_loss	worki
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	White	Male	2174	0	
1	50	Self-emp-not-inc	83311	Bachelors	13	married	Exec-managerial	White	Male	0	0	
2	38	Private	215646	HS-grad	9	not_married	Handlers-cleaners	White	Male	0	0	
3	53	Private	234721	11th	7	married	Handlers-cleaners	Black	Male	0	0	

```
data.tail()
```

	age	workclass	fnlwgt	education	education_no_of_years	marital_status	occupation	race	sex	capital_gain	capital_loss	w
32526	27	Private	257302	Assoc-acdm	12	married	Tech-support	White	Female	0	0	
32527	40	Private	154374	HS-grad	9	married	Machine-op-inspct	White	Male	0	0	
32528	58	Private	151910	HS-grad	9	not_married	Adm-clerical	White	Female	0	0	
32529	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	White	Male	0	0	
32530	52	Self-emp-inc	287927	HS-grad	9	married	Exec-managerial	White	Female	15024	0	

```
data.shape
```

```
(32531, 14)
```

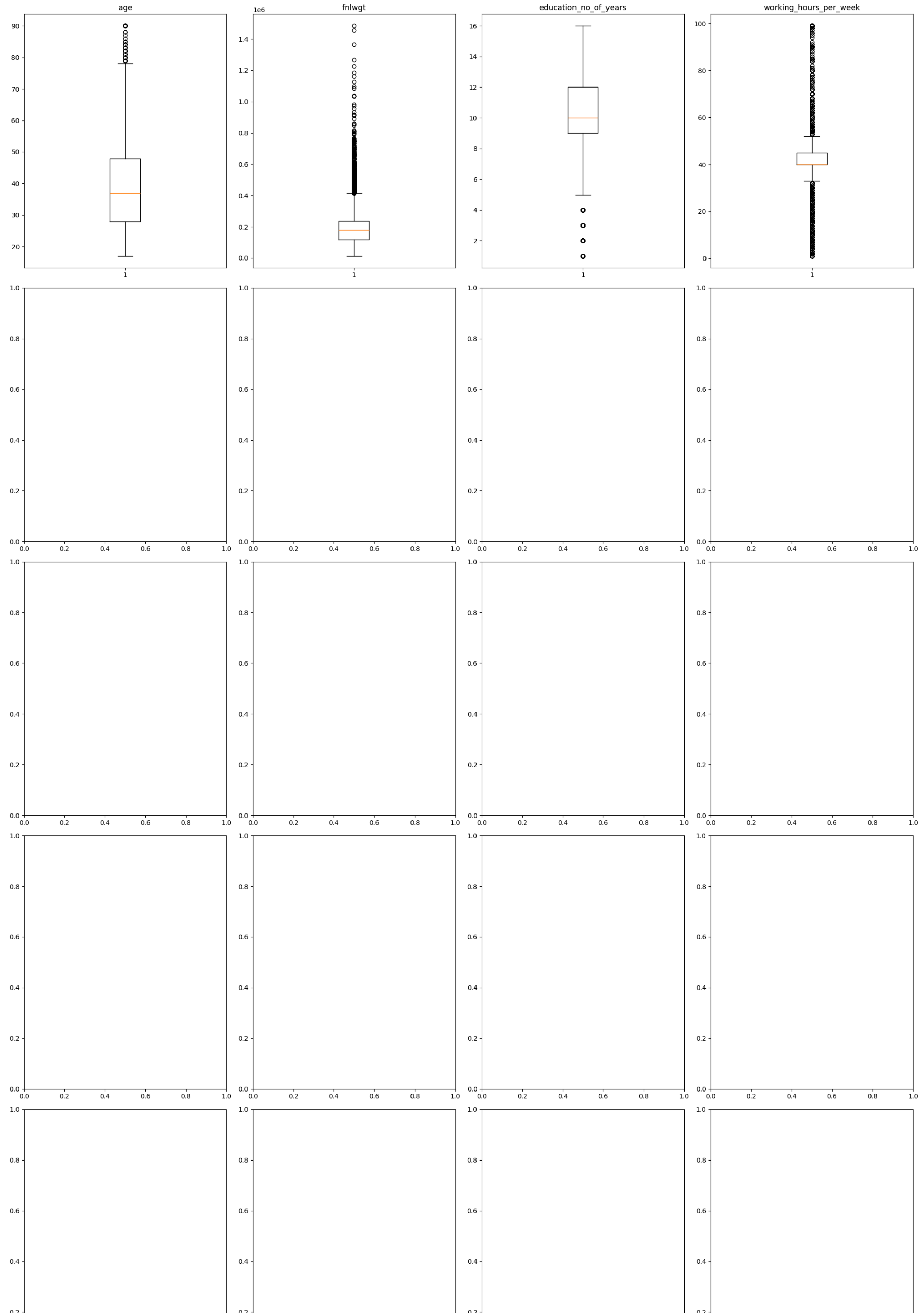
```
data.describe()
```

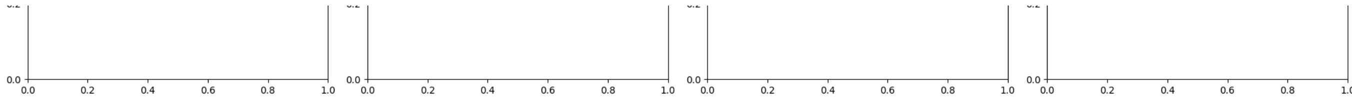
	age	fnlwgt	education_no_of_years	capital_gain	capital_loss	working_hours_per_week
count	32531.000000	3.253100e+04	32531.000000	32531.000000	32531.000000	32531.000000
mean	38.588362	1.897882e+05	10.081953	1078.642649	87.384341	40.441025
std	13.637644	1.055642e+05	2.571842	7388.624210	403.137260	12.347506
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.178150e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783700e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.370190e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.000000

```
data.drop(["capital_gain","capital_loss"],axis=1,inplace=True)
```

```
# numerical_col=data.select_dtypes(include=np.number).columns.tolist()
# plt.figure(figsize=(20, 30))
# for i, variable in enumerate(numerical_col):
#   plt.subplots(5, 4,i + 1)
#   plt.boxplot(data[variable], whis=1.5)
#   plt.tight_layout()
#   plt.title(variable)

# plt.show()
```

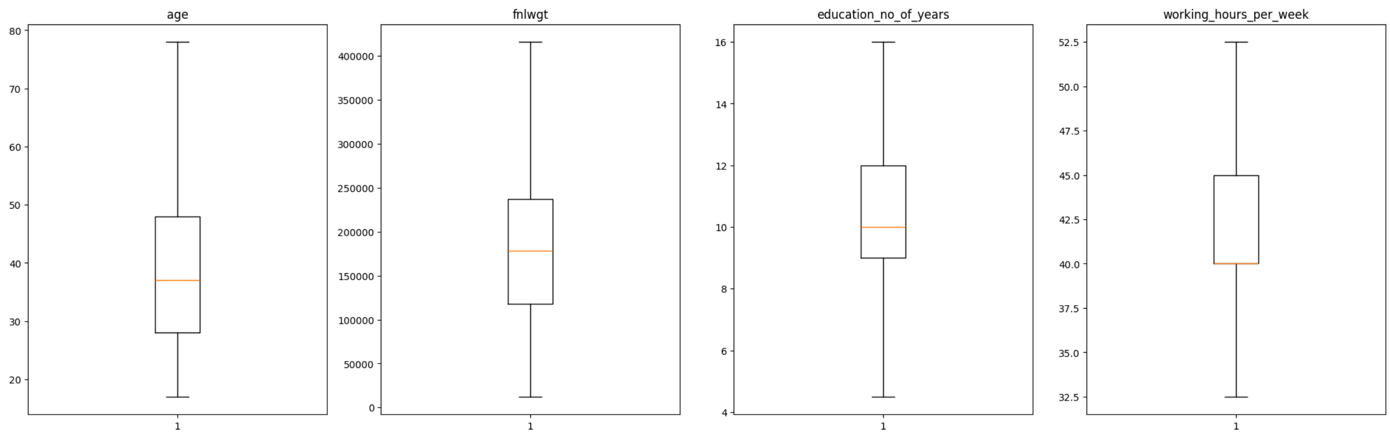
```
numerical_col = data.select_dtypes(include=np.number).columns.tolist()
fig, axes = plt.subplots(5, 4, figsize=(20, 30))

axes = axes.flatten()

for i, variable in enumerate(numerical_col):
    ax = axes[i]
    ax.boxplot(data[variable], whis=1.5)
    ax.set_title(variable)

plt.tight_layout()
plt.show()
```


age
fnlwgt
education_no_of_years
working_hours_per_week




```
# def treat_outliers(df,col):
#   q1=df[col].quantile(0.25)
#   q3=df[col].quantile(0.75)
#   iqr=q3-q1
#   lower_whisker=q1-1.5*iqr
#   upper_whisker=q3+1.5*iqr

#   df[col]=np.clip(df[col],lower_whisker,upper_whisker)
#   return df


#   def treat_outliers_all(df,col_list):
#     for c in col_list:
#       df=treat_outliers(df,c)
#     return df

#   numerical_col=data.select_dtypes(include=np.number).columns.tolist()
#   data=treat_outliers_all(data,numerical_col)

#   plt.figure(figsize=(20, 30))
#   for i, variable in enumerate(numerical_col):
#     plt.subplots(5, 4,i + 1)
#     plt.boxplot(data[variable], whis=1.5)
#     plt.tight_layout()
#     plt.title(variable)

# plt.show()


def treat_outliers(df, col):
    q1 = df[col].quantile(0.25)
    q3 = df[col].quantile(0.75)
    iqr = q3 - q1
    lower_whisker = q1 - 1.5 * iqr
    upper_whisker = q3 + 1.5 * iqr

    df[col] = np.clip(df[col], lower_whisker, upper_whisker)
    return df

def treat_outliers_all(df, col_list):
    for c in col_list:
        df = treat_outliers(df, c)
```

```
    return df

numerical_col = data.select_dtypes(include=np.number).columns.tolist()
data = treat_outliers_all(data, numerical_col)

plt.figure(figsize=(20, 30))
for i, variable in enumerate(numerical_col):
    plt.subplot(5, 4, i + 1)
    plt.boxplot(data[variable], whis=1.5)
    plt.title(variable)

plt.tight_layout()
plt.show()
```




```
data['salary']=data['salary'].apply(lambda x:1 if x==' <=50K' else 0)
```

```
x=data.drop(['salary'],axis=1)
y=data['salary']

x=sm.add_constant(x)
x=pd.get_dummies(x,drop_first=True)
x=x.astype(float)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=1)
```

```
logit=sm.Logit(y_train,x_train.astype(float))
lg=logit.fit()
print(lg.summary())
```

```
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------------
const                           9.5200   2.75e+05   3.46e-05      1.000   -5.39e+05     5.4e+05
age                            -0.0314      0.002    -17.126      0.000      -0.035      -0.028
fnlwgt                      -6.146e-07   2.13e-07     -2.881      0.004   -1.03e-06   -1.96e-07
education_no_of_years          -0.1961   4.59e+04  -4.27e-06      1.000   -8.99e+04    8.99e+04
working_hours_per_week         -0.0655      0.004    -18.686      0.000      -0.072      -0.059
workclass_Local-gov             0.7669      0.127      6.061      0.000       0.519       1.015
workclass_Never-worked         15.2788   1.47e+08   1.04e-07      1.000   -2.88e+08    2.88e+08
```

```
education_9th                        0.4116   4.59e+04   8.97e-06      1.000    -8.99e+04     8.99e+04
education_Assoc-acdm                -0.0897   2.75e+05  -3.26e-07      1.000    -5.39e+05     5.39e+05
education_Assoc-voc                 -0.3665   2.29e+05   -1.6e-06      1.000     -4.5e+05      4.5e+05
education_Bachelors                 -0.5915   3.21e+05  -1.84e-06      1.000    -6.29e+05     6.29e+05
education_Doctorate                 -0.9450   4.59e+05  -2.06e-06      1.000    -8.99e+05     8.99e+05
education_HS-grad                   -0.1775   1.38e+05  -1.29e-06      1.000     -2.7e+05      2.7e+05
education_Masters                   -0.7981   3.67e+05  -2.17e-06      1.000    -7.19e+05     7.19e+05
education_Preschool                 12.0908   6.88e+04      0.000      1.000    -1.35e+05     1.35e+05
education_Prof-school               -1.3568   4.13e+05  -3.29e-06      1.000    -8.09e+05     8.09e+05
education_Some-college              -0.2672   1.84e+05  -1.46e-06      1.000     -3.6e+05      3.6e+05
marital_status_married              -2.5742      0.069    -37.060      0.000       -2.710       -2.438
marital_status_not_married          -0.4045      0.084     -4.806      0.000       -0.569       -0.240
occupation_Armed-Forces              0.5322      1.529      0.348      0.728       -2.465        3.530
occupation_Craft-repair              0.0011      0.088      0.013      0.990       -0.172        0.175
occupation_Exec-managerial          -0.6952      0.085     -8.199      0.000       -0.861       -0.529
occupation_Farming-fishing           1.0593      0.157      6.763      0.000        0.752        1.366
occupation_Handlers-cleaners         0.8354      0.162      5.169      0.000        0.519        1.152
occupation_Machine-op-inspct         0.4118      0.115      3.573      0.000        0.186        0.638
occupation_Other-service             0.9811      0.135      7.294      0.000        0.718        1.245
occupation_Priv-house-serv           2.3843      1.091      2.185      0.029        0.245        4.524
occupation_Prof-specialty           -0.4583      0.089     -5.143      0.000       -0.633       -0.284
occupation_Protective-serv          -0.4979      0.143     -3.475      0.001       -0.779       -0.217
occupation_Sales                    -0.1410      0.091     -1.555      0.120       -0.319        0.037
occupation_Tech-support             -0.5943      0.123     -4.815      0.000       -0.836       -0.352
occupation_Transport-moving          0.1759      0.112      1.572      0.116       -0.043        0.395
occupation_Unknown                   8.2721   1.47e+08   5.63e-08      1.000    -2.88e+08     2.88e+08
race_Asian-Pac-Islander             -0.2499      0.287     -0.871      0.384       -0.812        0.312
race_Black                          -0.2403      0.251     -0.958      0.338       -0.732        0.251
race_Other                           0.4680      0.401      1.167      0.243       -0.318        1.254
race_White                          -0.3346      0.238     -1.403      0.161       -0.802        0.133
sex_Male                            -0.0661      0.059     -1.129      0.259       -0.181        0.049
native_contienent_europe            -0.2102      0.241     -0.871      0.384       -0.683        0.263
native_contienent_north_america     -0.2393      0.191     -1.251      0.211       -0.614        0.136
native_contienent_other              0.2389      0.218      1.097      0.272       -0.188        0.666
native_contienent_south_america      1.0479      0.526      1.992      0.046        0.017        2.079
====================================================================================================
```

```python
def model_performance_classification_statsmodels(
    model,
    predictors,
    target,
    threshold=0.5
):
  pred_temp=model.predict(predictors)>threshold
  pred=np.round(pred_temp)

  acc=accuracy_score(target,pred)
  f1=f1_score(target,pred)
  recall=recall_score(target,pred)
  precision=precision_score(target,pred)

  df_perf=pd.DataFrame(
      {
          'Accuracy':acc,
          'f1':f1,
          'Recall':recall,
          'Precision':precision,
      },
      index=[0]
  )
  return df_perf

print("Training Performance")
model_performance_classification_statsmodels(lg,x_train,y_train)
```

Training Performance

	Accuracy	f1	Recall	Precision
0	0.835624	0.894957	0.923759	0.867897

```python
vif_series=pd.Series(
    [variance_inflation_factor(x_train.values,i) for i in range(x_train.shape[1])],
    index=x_train.columns
)
print("VIF",vif_series)
```

```
VIF const                         0.000000
age                          1.531401
fnlwgt                       1.037491
```

```
education_no_of_years                  inf
working_hours_per_week            1.237162
workclass_Local-gov               3.041470
workclass_Never-worked                 inf
workclass_Private                 7.717533
workclass_Self-emp-inc            2.217152
workclass_Self-emp-not-inc        3.487046
workclass_State-gov               2.317984
workclass_Unknown                      inf
workclass_Without-pay             1.015562
education_11th                         inf
education_12th                         inf
education_1st-4th                      inf
education_5th-6th                      inf
education_7th-8th                      inf
education_9th                          inf
education_Assoc-acdm                   inf
education_Assoc-voc                    inf
education_Bachelors                    inf
education_Doctorate                    inf
education_HS-grad                      inf
education_Masters                      inf
education_Preschool                    inf
education_Prof-school                  inf
education_Some-college                 inf
marital_status_married            1.971883
marital_status_not_married        1.714786
occupation_Armed-Forces           1.010066
occupation_Craft-repair           2.133802
occupation_Exec-managerial        2.046469
occupation_Farming-fishing        1.401571
occupation_Handlers-cleaners      1.440475
occupation_Machine-op-inspct      1.577192
occupation_Other-service          1.750404
occupation_Priv-house-serv        1.054692
occupation_Prof-specialty         2.317960
occupation_Protective-serv        1.253107
occupation_Sales                  1.891884
occupation_Tech-support           1.239360
occupation_Transport-moving       1.509398
occupation_Unknown                     inf
race_Asian-Pac-Islander           5.305020
race_Black                       10.196674
race_Other                        1.878501
race_White                       13.494644
sex_Male                          1.520198
native_contienent_europe          2.424319
native_contienent_north_america   5.250971
native_contienent_other           2.331141
native_contienent_south_america   1.328881
dtype: float64
```

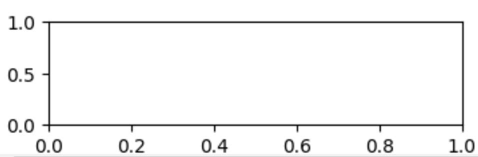
```
x_train=x_train.drop("occupation_Unknown",axis=1)

vif_series2=pd.Series(
    [variance_inflation_factor(x_train.values,i) for i in range(x_train.shape[1])],
    index=x_train.columns
)
print("VIF",vif_series2)
#
```

```
VIF const                             0.000000
age                               1.531401
fnlwgt                            1.037491
education_no_of_years                  inf
working_hours_per_week            1.237162
workclass_Local-gov               3.041470
workclass_Never-worked            1.012785
workclass_Private                 7.717533
workclass_Self-emp-inc            2.217152
workclass_Self-emp-not-inc        3.487046
workclass_State-gov               2.317984
workclass_Unknown                 3.039325
workclass_Without-pay             1.015562
education_11th                         inf
education_12th                         inf
education_1st-4th                      inf
education_5th-6th                      inf
education_7th-8th                      inf
education_9th                          inf
education_Assoc-acdm                   inf
education_Assoc-voc                    inf
education_Bachelors                    inf
```

```
education_Doctorate                   inf
education_HS-grad                     inf
education_Masters                     inf
education_Preschool                   inf
education_Prof-school                 inf
education_Some-college                inf
marital_status_married           1.971883
marital_status_not_married       1.714786
occupation_Armed-Forces          1.010066
occupation_Craft-repair          2.133802
occupation_Exec-managerial       2.046469
occupation_Farming-fishing       1.401571
occupation_Handlers-cleaners     1.440475
occupation_Machine-op-inspct     1.577192
occupation_Other-service         1.750404
occupation_Priv-house-serv       1.054692
occupation_Prof-specialty        2.317960
occupation_Protective-serv       1.253107
occupation_Sales                 1.891884
occupation_Tech-support          1.239360
occupation_Transport-moving      1.509398
race_Asian-Pac-Islander          5.305020
race_Black                      10.196674
race_Other                       1.878501
race_White                      13.494644
sex_Male                         1.520198
native_contienent_europe         2.424319
native_contienent_north_america  5.250971
native_contienent_other          2.331141
native_contienent_south_america  1.328881
dtype: float64
```

```
x_train2=x_train.drop("education_no_of_years",axis=1)

vif_series2=pd.Series(
    [variance_inflation_factor(x_train2.values,i) for i in range(x_train2.shape[1])],
    index=x_train2.columns
)
print("VIF",vif_series2)
```

```
VIF const                        326.271888
age                               1.531401
fnlwgt                            1.037491
working_hours_per_week            1.237162
workclass_Local-gov               3.041470
workclass_Never-worked            1.012785
workclass_Private                 7.717533
workclass_Self-emp-inc            2.217152
workclass_Self-emp-not-inc        3.487046
workclass_State-gov               2.317984
workclass_Unknown                 3.039325
workclass_Without-pay             1.015562
education_11th                    2.147415
education_12th                    1.422410
education_1st-4th                 1.188352
education_5th-6th                 1.338683
education_7th-8th                 1.655272
education_9th                     1.524978
education_Assoc-acdm              2.105382
education_Assoc-voc               2.374490
education_Bachelors               5.939193
education_Doctorate               1.564790
education_HS-grad                 8.112679
education_Masters                 2.978449
education_Preschool               1.064098
education_Prof-school             1.811978
education_Some-college            6.857574
marital_status_married            1.971883
marital_status_not_married        1.714786
occupation_Armed-Forces           1.010066
occupation_Craft-repair           2.133802
occupation_Exec-managerial        2.046469
occupation_Farming-fishing        1.401571
occupation_Handlers-cleaners      1.440475
occupation_Machine-op-inspct      1.577192
occupation_Other-service          1.750404
occupation_Priv-house-serv        1.054692
occupation_Prof-specialty         2.317960
occupation_Protective-serv        1.253107
occupation_Sales                  1.891884
occupation_Tech-support           1.239360
occupation_Transport-moving       1.509398
race_Asian-Pac-Islander           5.305020
race_Black                       10.196674
race_Other                        1.878501
race_White                       13.494644
sex_Male                          1.520198
native_contienent_europe          2.424319
native_contienent_north_america   5.250971
native_contienent_other           2.331141
native_contienent_south_america   1.328881
dtype: float64
```




```
logit2=sm.Logit(y_train,x_train2.astype(float))
lg2=logit2.fit(disp=False)
print("training performance")
model_performance_classification_statsmodels(lg2,x_train2,y_train)
```

training performance

	Accuracy	f1	Recall	Precision
0	0.835624	0.894957	0.923759	0.867897

```
cols=x_train2.columns.tolist()
max_p_value=1
while len(cols)>0:
  x_train_aux=x_train2[cols]
  model=sm.Logit(y_train,x_train_aux.astype(float))
  model_fit=model.fit(disp=False)
  p_values=model_fit.pvalues
  max_p_value=max(p_values)
  feature_with_p_max=p_values.idxmax()
  if max_p_value>0.05:
```

```
    cols.remove(feature_with_p_max)
  else:
    break
selected_features=cols
print(selected_features)
```

```
['const', 'age', 'fnlwgt', 'working_hours_per_week', 'workclass_Local-gov', 'workclass_Private', 'workclass_Self-emp-not-inc', 'workclas
```

```
x_train3=x_train2[selected_features]
logit3=sm.Logit(y_train,x_train3.astype(float))
lg3=logit3.fit(disp=False)
print(lg3.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                 salary   No. Observations:                22771
Model:                          Logit   Df Residuals:                    22735
Method:                           MLE   Df Model:                           35
Date:                Tue, 26 Nov 2024   Pseudo R-squ.:                  0.3666
Time:                        17:50:00   Log-Likelihood:                -7980.7
converged:                       True   LL-Null:                       -12600.
Covariance Type:            nonrobust   LLR p-value:                     0.000
==================================================================================================
                                      coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------------------------
const                               8.0557      0.233     34.544      0.000       7.599       8.513
age                                -0.0314      0.002    -17.178      0.000      -0.035      -0.028
fnlwgt                          -6.273e-07   2.12e-07     -2.964      0.003   -1.04e-06   -2.12e-07
working_hours_per_week             -0.0650      0.003    -19.014      0.000      -0.072      -0.058
workclass_Local-gov                 0.6101      0.103      5.930      0.000       0.408       0.812
workclass_Private                   0.3585      0.072      4.976      0.000       0.217       0.500
workclass_Self-emp-not-inc          0.8012      0.094      8.485      0.000       0.616       0.986
workclass_State-gov                 0.7793      0.121      6.466      0.000       0.543       1.016
workclass_Unknown                   1.0881      0.132      8.229      0.000       0.829       1.347
education_1st-4th                   1.5989      0.729      2.192      0.028       0.169       3.029
education_7th-8th                   0.6164      0.229      2.687      0.007       0.167       1.066
education_9th                       0.5931      0.279      2.123      0.034       0.046       1.141
education_Assoc-acdm               -1.2818      0.146     -8.791      0.000      -1.568      -0.996
education_Assoc-voc                -1.3671      0.134    -10.168      0.000      -1.631      -1.104
education_Bachelors                -1.9848      0.114    -17.449      0.000      -2.208      -1.762
education_Doctorate                -2.9087      0.196    -14.845      0.000      -3.293      -2.525
education_HS-grad                  -0.7808      0.108     -7.260      0.000      -0.992      -0.570
education_Masters                  -2.3806      0.131    -18.149      0.000      -2.638      -2.124
education_Prof-school              -3.1388      0.178    -17.616      0.000      -3.488      -2.790
education_Some-college             -1.0739      0.111     -9.666      0.000      -1.292      -0.856
marital_status_married             -2.5792      0.067    -38.449      0.000      -2.711      -2.448
marital_status_not_married         -0.3935      0.084     -4.711      0.000      -0.557      -0.230
occupation_Exec-managerial         -0.7135      0.063    -11.324      0.000      -0.837      -0.590
occupation_Farming-fishing          1.0603      0.142      7.450      0.000       0.781       1.339
occupation_Handlers-cleaners        0.7954      0.149      5.340      0.000       0.503       1.087
occupation_Machine-op-inspct        0.3845      0.097      3.967      0.000       0.195       0.575
occupation_Other-service            0.9771      0.122      8.026      0.000       0.738       1.216
occupation_Priv-house-serv          2.4240      1.082      2.240      0.025       0.303       4.545
occupation_Prof-specialty          -0.4775      0.072     -6.675      0.000      -0.618      -0.337
occupation_Protective-serv         -0.5487      0.130     -4.205      0.000      -0.804      -0.293
occupation_Sales                   -0.1563      0.068     -2.299      0.021      -0.290      -0.023
occupation_Tech-support            -0.6295      0.110     -5.700      0.000      -0.846      -0.413
race_Other                          0.6816      0.329      2.069      0.039       0.036       1.327
race_White                         -0.1545      0.067     -2.301      0.021      -0.286      -0.023
native_contienent_other             0.4532      0.149      3.041      0.002       0.161       0.745
native_contienent_south_america     1.2598      0.489      2.576      0.010       0.301       2.218
==================================================================================================
```

```
odds=np.exp(lg3.params)
perc_change_odds=(np.exp(lg3.params)-1)*100

pd.set_option('display.max_columns',None)
pd.DataFrame({"Odds":odds, "Change_Odds":perc_change_odds},index=x_train3.columns).T
```

	const	age	fnlwgt	working_hours_per_week	workclass_Local-gov	workclass_Private	workclass_Self-emp-not-inc	workclass
Odds	3151.769230	0.969130	0.999999	0.937106	1.840620	1.431156	2.228214	2
Change_Odds	315076.922967	-3.086978	-0.000063	-6.289395	84.062005	43.115618	122.821364	118

```
log_reg_model_train_perf=model_performance_classification_statsmodels(lg3,x_train3,y_train)
print("Training_performace")
log_reg_model_train_perf
```

Training_performace

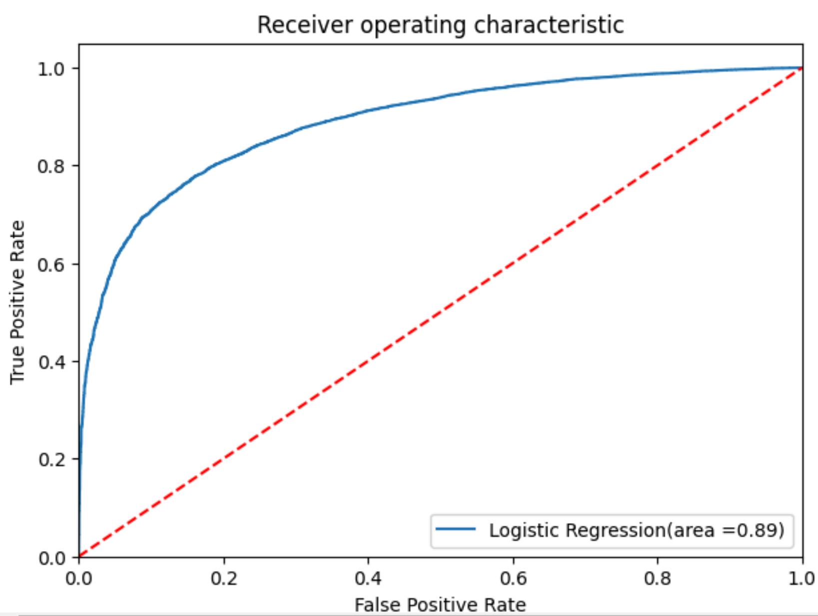
	Accuracy	f1	Recall	Precision
0	0.835536	0.894936	0.924048	0.867602

```
x_test3=x_test[list(x_train3.columns)]
log_reg_model_test_perf=model_performance_classification_statsmodels(lg3,x_test3,y_test)
print("Test_performace")
log_reg_model_test_perf
```

Test_performace

	Accuracy	f1	Recall	Precision
0	0.830635	0.891855	0.917239	0.867838

```
logit_roc_auc_train=roc_auc_score(y_train,lg3.predict(x_train3))
fpr,tpr,thresholds=roc_curve(y_train,lg3.predict(x_train3))
plt.figure(figsize=(7,5))
plt.plot(fpr,tpr,label="Logistic Regression(area =%0.2f)"%logit_roc_auc_train)
plt.plot([0,1],[0,1],'r--')
plt.xlim([0.0,1.0])
plt.ylim([0.0,1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```




```
fpr,tpr,thresholds=roc_curve(y_train,lg3.predict(x_train3))

optimal_idx=np.argmax(tpr-fpr)
optimal_threshold_auc_roc=thresholds[optimal_idx]
print(optimal_threshold_auc_roc)
```

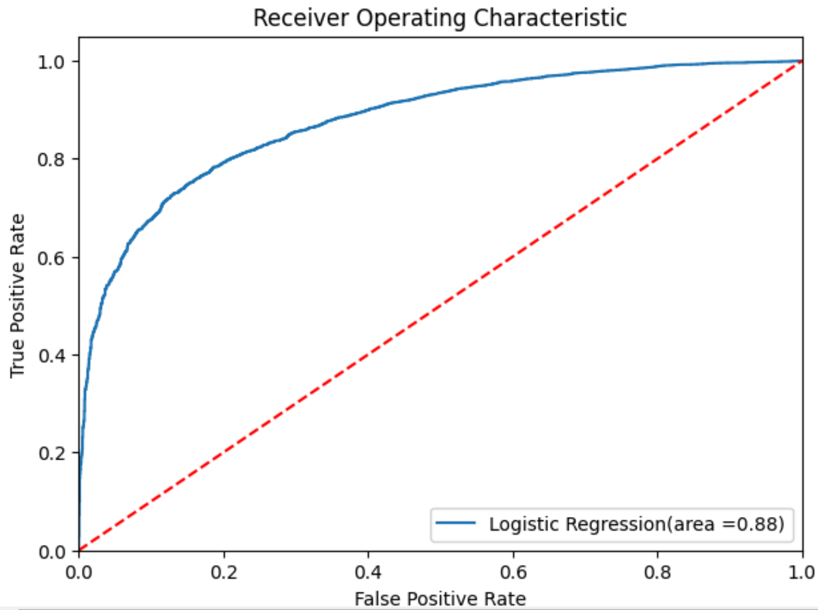
0.7579399407330087

```
log_reg_model_train_perf_threshold_auc_roc=model_performance_classification_statsmodels(lg3,x_train3,y_train,optimal_threshold_auc_roc)
print("Training_performace")
log_reg_model_train_perf_threshold_auc_roc
```

Training_performace

	Accuracy	f1	Recall	Precision
0	0.792499	0.85028	0.777301	0.938383

```
logit_roc_auc_train=roc_auc_score(y_test,lg3.predict(x_test3))
fpr,tpr,thresholds=roc_curve(y_test,lg3.predict(x_test3))
plt.figure(figsize=(7,5))
plt.plot(fpr,tpr,label="Logistic Regression(area =%0.2f)"%logit_roc_auc_train)
plt.plot([0,1],[0,1],'r--')
plt.xlim([0.0,1.0])
plt.ylim([0.0,1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic')
plt.legend(loc="lower right")
plt.show()
```




```
log_reg_model_test_perf_threshold_auc_roc=model_performance_classification_statsmodels(lg3,x_test3,y_test,optimal_threshold_auc_roc)
print("Test_performace")
log_reg_model_test_perf_threshold_auc_roc
```

Test_performace

	Accuracy	f1	Recall	Precision
0	0.783299	0.844336	0.771901	0.931774

```
y_scores=lg3.predict(x_train3)
prec,rec_val,thres=precision_recall_curve(y_train,y_scores)

def plot_precision_recall_vs_threshold(prec,rec_val,thres):
  plt.plot(thres,prec[:-1],"b--",label="Precision")
  plt.plot(thres,rec_val[:-1],"g-",label="Recall")
  plt.xlabel("Threshold")
  plt.legend(loc="upper left")
  plt.ylim([0,1])
plt.figure(figsize=(10,7))
plot_precision_recall_vs_threshold(prec,rec_val,thres)
plt.show()
```