GRANTLY EDUTECH Pvt Limited

PROJECT TITLE

EMPLOYEE DATA CLEANING

```
import pandas as pd
import numpy as np
from datetime import datetime
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import pickle
pdf = pd.read_csv('C:/Users/LENOVO/Downloads/Employee_Salaries.csv')
print("Columns in the dataset:", pdf.columns)
pdf['Base_Salary'] = pdf['Base_Salary'].fillna(pdf['Base_Salary'].mean())
print(pdf.duplicated().sum())
pdf = pdf.drop_duplicates()
sns.boxplot(pdf['Base_Salary'])
plt.show()
Q1 = pdf['Base_Salary'].quantile(0.25)
Q3 = pdf['Base_Salary'].quantile(0.75)
IQR = Q3 - Q1
pdf = pdf[~((pdf['Base_Salary'] < (Q1 - 1.5 * IQR)) | (pdf['Base_Salary'] > (Q3 + 1.5 * IQR)))]
pdf['Base_Salary'] = pdf['Base_Salary'].replace({r'[$,]': ''}, regex=True)
```

```
pdf['Base_Salary'] = pd.to_numeric(pdf['Base_Salary'], errors='coerce')
print(pdf.isnull().sum())
print(pdf.dtypes)
pdf = pd.get_dummies(pdf, columns=['Department', 'Division'])
print("Columns in the dataset:", pdf.columns)
scaler = StandardScaler()
columns_to_scale = ['Base_Salary', 'Overtime_Pay']
for col in columns_to_scale:
    if col not in pdf.columns:
        raise KeyError(f"Column '{col}' not found in the dataset.")
    if not np.issubdtype(pdf[col].dtype, np.number):
        raise ValueError(f"Column '{col}' is not numeric.")
pdf[columns_to_scale] = scaler.fit_transform(pdf[columns_to_scale])
X = pdf.drop(target_column, axis=1)
y = pdf[target_column]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 5, 10],
    'min_samples_split': [2, 5, 10]
}
```

grid_search = GridSearchCV(RandomForestClassifier(random_state=42), param_grid, cv=5, scoring='accuracy')

grid_search.fit(X_train, y_train)

print("Best Parameters:", grid_search.best_params_)

print("Best Accuracy:", grid_search.best_score_)

with open('employee_model.pkl', 'wb') as f:

pickle.dump(grid_search.best_estimator_, f)

with open('employee_model.pkl', 'rb') as f:

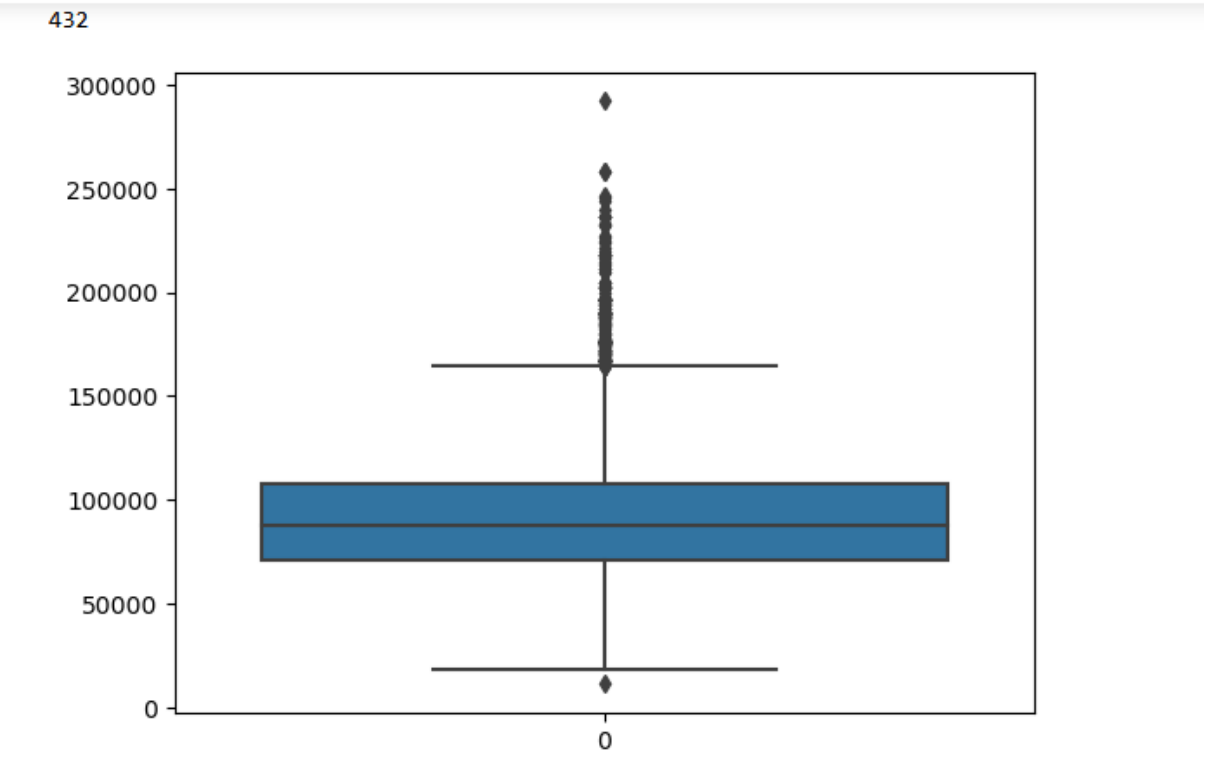
loaded_model = pickle.load(f)

predictions = loaded_model.predict(X_test)

print("Predictions:", predictions)

OUTPUT:-

```
Columns in the dataset: Index(['Department', 'Department_Name', 'Division',
'Gender', 'Base_Salary',
       'Overtime_Pay', 'Longevity_Pay', 'Grade', 'Target', 'Actual
Target'],
      dtype='object')
```




```
Department             0
Department_Name        0
Division               0
```

```
Gender                0
Base_Salary           0
Overtime_Pay          0
Longevity_Pay         0
Grade                20
Target             5713
Actual Target      5189
dtype: int64
Department          object
Department_Name     object
Division            object
Gender              object
Base_Salary        float64
Overtime_Pay       float64
Longevity_Pay      float64
Grade               object
Target              object
Actual Target       object
dtype: object
Columns in the dataset: Index(['Department_Name', 'Gender', 'Base_Salary',
'Overtime_Pay',
       'Longevity_Pay', 'Grade', 'Target', 'Actual Target',
'Department_ABS',
       'Department_BOA',
       ...
       'Division_TBS 34 OPS Public Safety Data Systems',
       'Division_TBS 34 OPS Radio Communications Services',
       'Division_TBS 34 OSP DevOps and Server Support',
       'Division_TBS 34 OSP Employee Productivity Services',
       'Division_TBS 34 OSP Enterprise Cloud Solutions',
       'Division_TBS 34 OSP Enterprise Services Team',
       'Division_TBS 34 OSP Infrastructure and Cloud Services',
       'Division_TBS 34 OSP Low Code Governance and Administration',
       'Division_ZAH 05 Office of Zoning and Administrative Hearings',
       'Division_ZAH 05 Zoning and Administrative Hearings'],
      dtype='object', length=635)
```