```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from scipy.stats import ttest_ind, pearsonr
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import LabelEncoder

# Load the dataset
file_path = '/content/employee_data.csv'  # Update path if needed
df = pd.read_csv(file_path)
```

Run cell (Ctrl+Enter)
cell executed since last change

executed by UDIT RANJAN
9:49 AM (0 minutes ago)
executed in 0.019s

```
# Display basic info
print(" Dataset Overview:")
print(df.info())
print("\n First 5 Records:")
print(df.head())

# Handle missing values intelligently
df.loc[:, 'Salary'] = df['Salary'].fillna(df['Salary'].median())
df.loc[:, 'Experience'] = df['Experience'].fillna(df['Experience'].median())
  # Fill missing experience
df.dropna(inplace=True)  # Drop rows where categorical fields are missing

# Detect Outliers using IQR method
Q1 = df['Salary'].quantile(0.25)
Q3 = df['Salary'].quantile(0.75)
IQR = Q3 - Q1
outliers = df[(df['Salary'] < (Q1 - 1.5 * IQR)) | (df['Salary'] > (Q3 + 1.5 * IQR))]
print(f"\n Outliers detected: {len(outliers)}")

# Salary Statistics
print("\n Salary Statistics:")
print(df['Salary'].describe())
```

```
 Dataset Overview:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   EmployeeID  20 non-null     int64
 1   Age         20 non-null     int64
 2   Department  20 non-null     object
 3   Experience  20 non-null     int64
 4   Salary      20 non-null     int64
dtypes: int64(4), object(1)
memory usage: 932.0+ bytes
None

 First 5 Records:
   EmployeeID  Age Department  Experience  Salary
0           1   23    Finance           8   93563
1           2   28    Finance           2   41742
2           3   37         HR           8   56905
3           4   23         HR          23  138397
4           5   55         IT          29   96879

 Outliers detected: 0
```

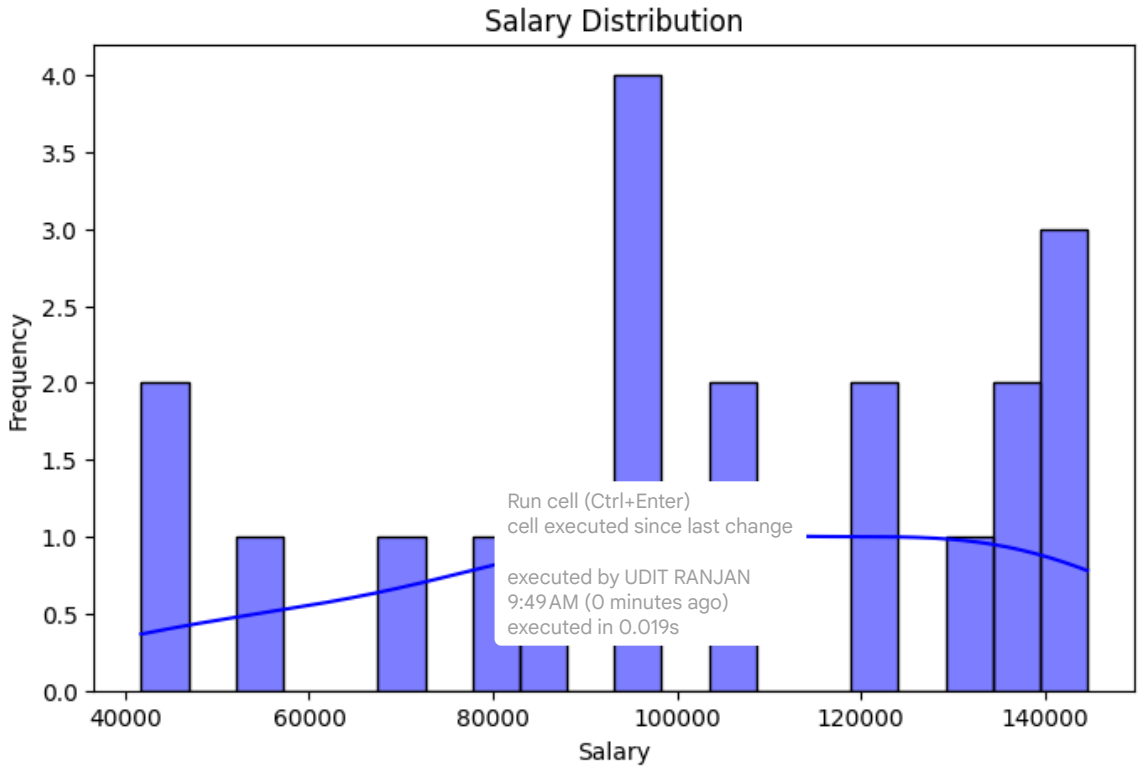
```
 Salary Statistics:
count        20.000000
mean     102503.150000
std       32459.740566
min       41742.000000
25%       82244.750000
50%      101315.500000
75%      132247.500000
max      144637.000000
Name: Salary, dtype: float64
```

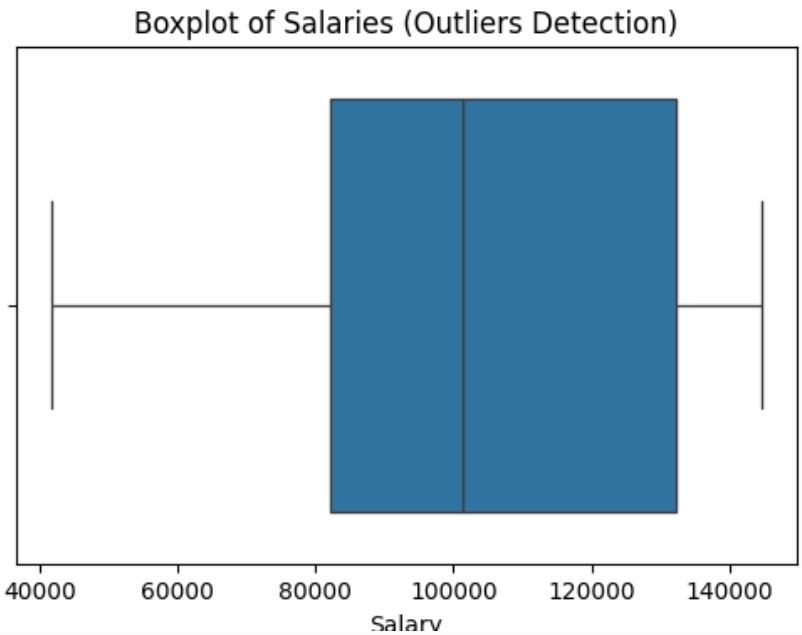
```
# Salary Distribution Visualization
plt.figure(figsize=(8,5))
sns.histplot(df['Salary'], bins=20, kde=True, color='blue')
plt.title('Salary Distribution')
plt.xlabel('Salary')
plt.ylabel('Frequency')
plt.show()

# Boxplot for Outlier Detection
plt.figure(figsize=(6,4))
sns.boxplot(x=df['Salary'])
plt.title("Boxplot of Salaries (Outliers Detection)")
plt.show()
```

Run cell (Ctrl+Enter)
cell executed since last change

executed by UDIT RANJAN
9:49 AM (0 minutes ago)
executed in 0.019s


Salary Distribution
Frequency
Salary
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
40000
60000
80000
100000
120000
140000
Run cell (Ctrl+Enter)
cell executed since last change
executed by UDIT RANJAN
9:49 AM (0 minutes ago)
executed in 0.019s

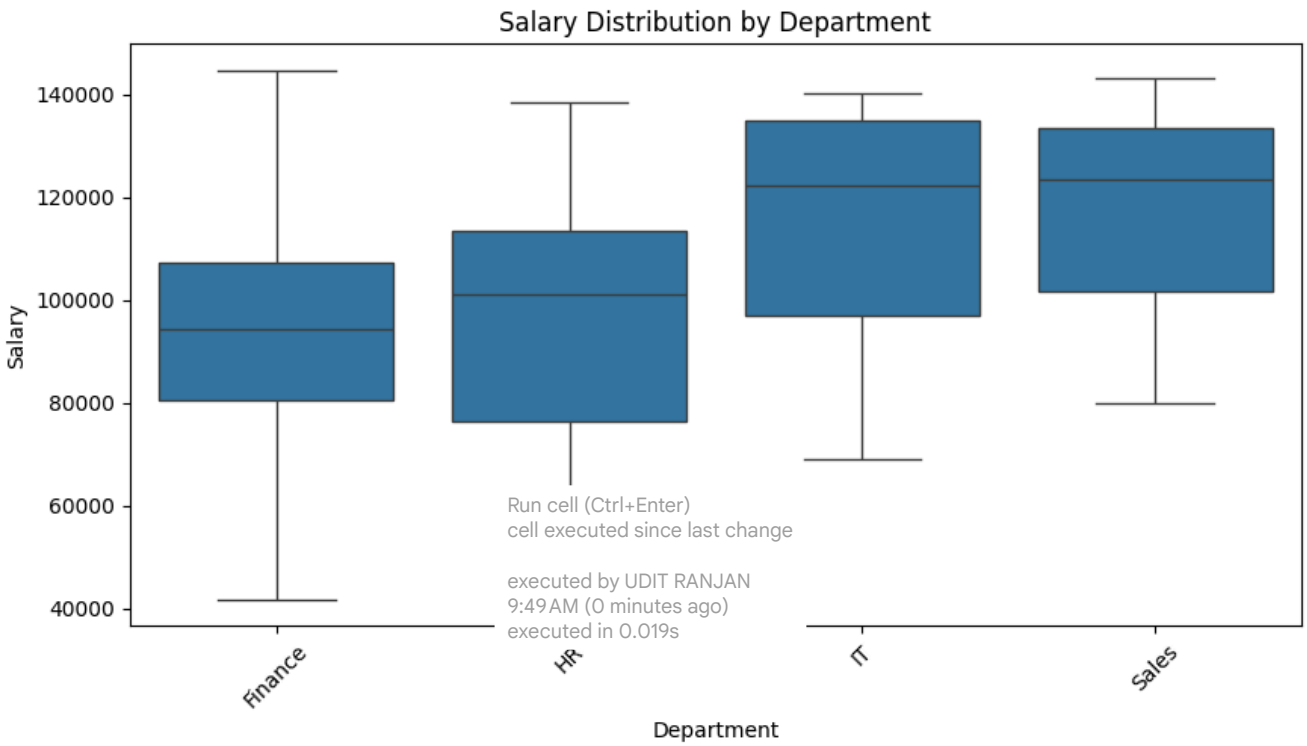

Boxplot of Salaries (Outliers Detection)
40000
60000
80000
100000
120000
140000
Salary

```
# Salary by Department
if 'Department' in df.columns:
    plt.figure(figsize=(10,5))
    sns.boxplot(x='Department', y='Salary', data=df)
    plt.xticks(rotation=45)
    plt.title('Salary Distribution by Department')
    plt.show()

# Salary by Experience
if 'Experience' in df.columns:
    plt.figure(figsize=(8,5))
    sns.scatterplot(x='Experience', y='Salary', data=df)
    plt.title('Salary vs Experience')
    plt.xlabel('Years of Experience')
    plt.ylabel('Salary')
    plt.show()
```

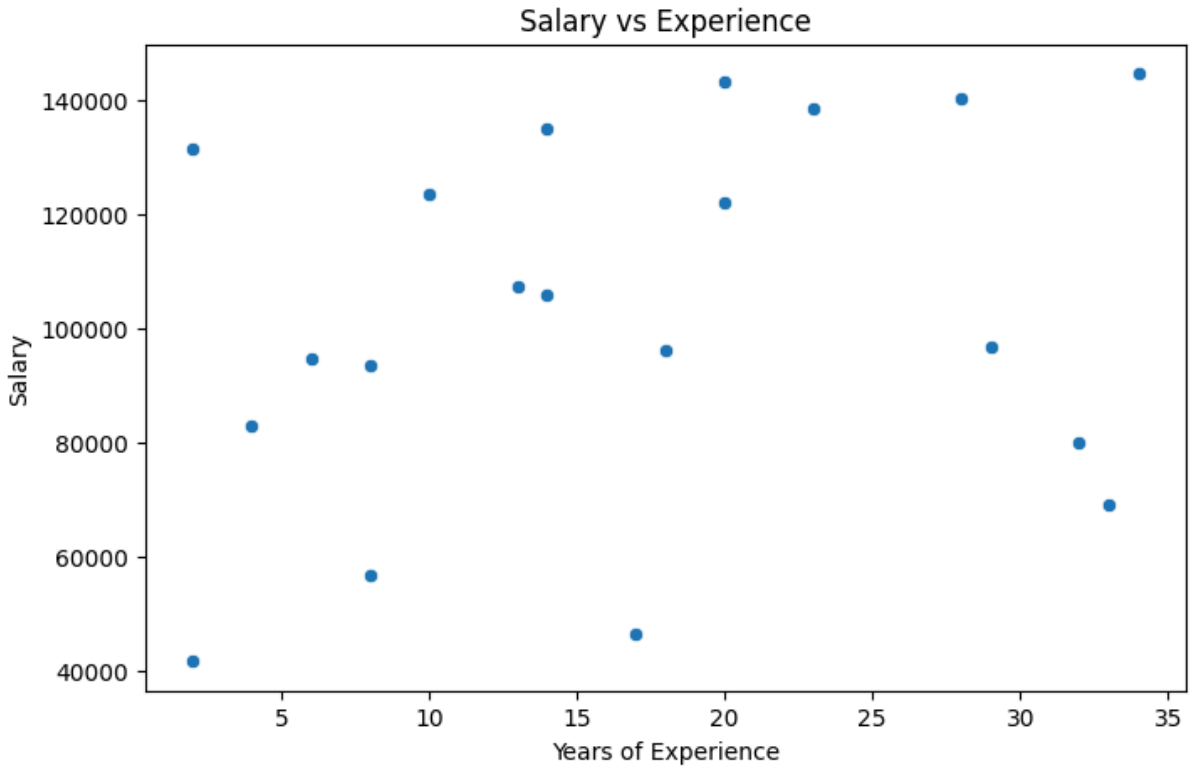
Run cell (Ctrl+Enter)
cell executed since last change

executed by UDIT RANJAN
9:49 AM (0 minutes ago)
executed in 0.019s



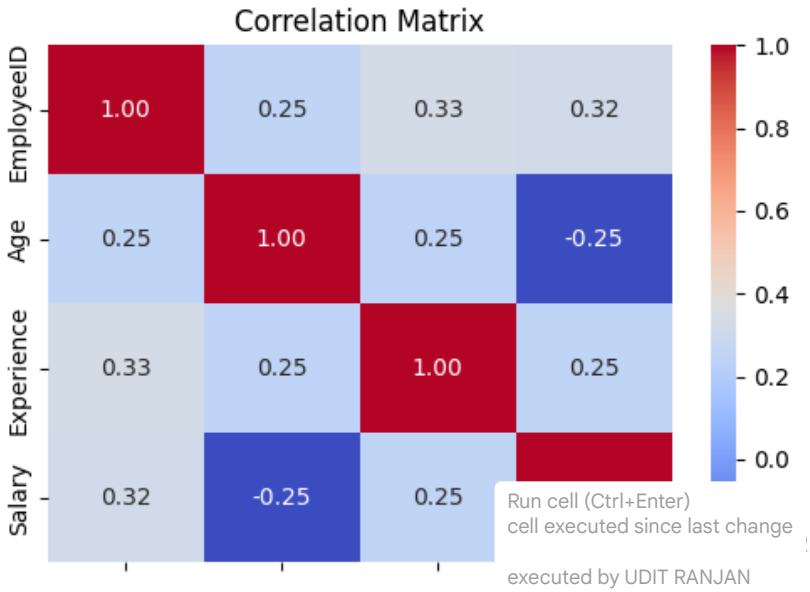
Run cell (Ctrl+Enter)
cell executed since last change

executed by UDIT RANJAN
9:49 AM (0 minutes ago)
executed in 0.019s




```
# Correlation Analysis (Exclude non-numeric columns)
numeric_df = df.select_dtypes(include=['number'])

if not numeric_df.empty:
    plt.figure(figsize=(6, 4))
    sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
    plt.title('Correlation Matrix')
    plt.show()
else:
    print("\nNo numeric columns found for correlation analysis.")
```



Run cell (Ctrl+Enter)
cell executed since last change

executed by UDIT RANJAN
9:49 AM (0 minutes ago)
executed in 0.019s