EDS_Minor Group Project..

Projected by :-

1) 717_Ruturaj Deshmukh.

2) 707_Om Bhutkar.

3) 716_Prathamesh Deshmukh.

4) 704_Shreyas Atkari.

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

```
data = pd.read_csv("/content/minor.csv")
print(data)
```

```
    sr.no    emp_id  age        Dept location education    recruitment_type  \
0       0    HR8270   28          HR   Suburb        PG            Referral
1       1  TECH1860   50  Technology   Suburb        PG             Walk-in
2       2  TECH6390   43  Technology   Suburb        UG            Referral
3       3   SAL6191   44       Sales     City        PG           On-Campus
4       4    HR6734   33          HR     City        UG  Recruitment Agency
..    ...       ...  ...         ...      ...       ...                 ...
94     94    HR4104   36          HR   Suburb        UG            Referral
95     95    HR8215   44          HR   Suburb        UG  Recruitment Agency
96     96    HR3454   33          HR     City        UG  Recruitment Agency
97     97   PUR9996   54  Purchasing     City        UG             Walk-in
98     98   SAL3731   49       Sales     City        PG            Referral

    job_level  rating  onsite  awards  certifications  salary  satisfied
0           5       2       0       1               0   86750          1
1           3       5       1       2               1   42419          0
2           4       1       0       2               0   65715          0
3           2       3       1       0               0   29805          1
4           2       1       0       5               0   29805          1
..        ...     ...     ...     ...             ...     ...        ...
94          1       1       0       0               1   24076          0
95          4       4       1       5               1   65715          0
96          4       5       1       3               1   65715          1
97          1       3       1       7               1   24076          0
98          3       5       0       8               0   42419          0

[99 rows x 14 columns]
```

```
# Display the first few rows of the dataset
print(data.head())
```

```
   sr.no    emp_id  age        Dept location education    recruitment_type  \
0      0    HR8270   28          HR   Suburb        PG            Referral
1      1  TECH1860   50  Technology   Suburb        PG             Walk-in
2      2  TECH6390   43  Technology   Suburb        UG            Referral
3      3   SAL6191   44       Sales     City        PG           On-Campus
```

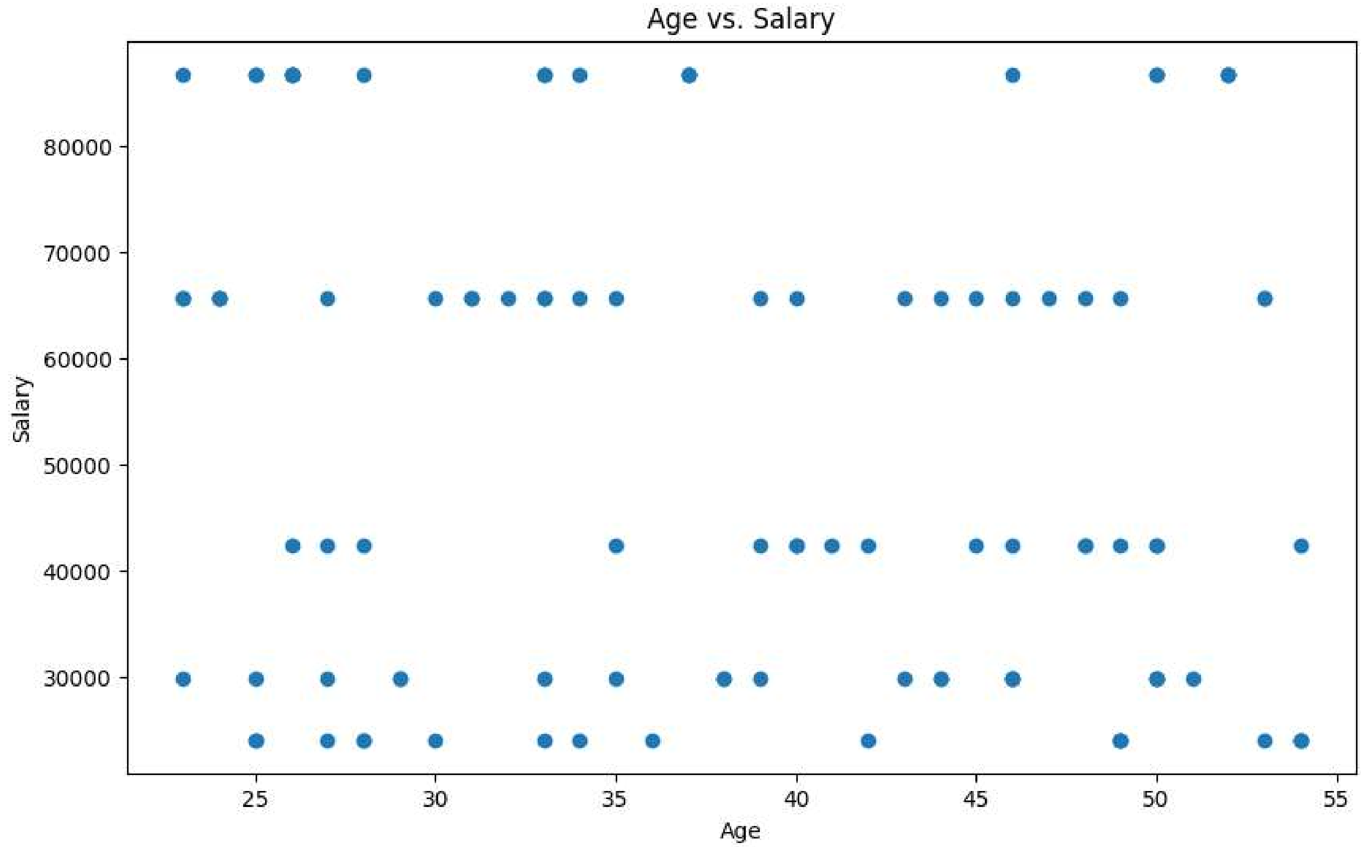
```
     4      4    HR6734   33          HR      City        UG  Recruitment Agency

        job_level  rating  onsite  awards  certifications  salary  satisfied
     0          5       2       0       1               0   86750          1
     1          3       5       1       2               1   42419          0
     2          4       1       0       2               0   65715          0
     3          2       3       1       0               0   29805          1
     4          2       1       0       5               0   29805          1
```

```
# Compute basic statistics
average_age = data['age'].mean()
max_salary = data['salary'].max()
print(average_age)
print(max_salary)
```

```
     38.313131313131315
     86750
```

```
# Perform data manipulation
data['total_awards'] = data['awards'] + data['certifications']
```

```
# Visualization
plt.figure(figsize=(10, 6))
plt.scatter(data['age'], data['salary'])
plt.xlabel('Age')
plt.ylabel('Salary')
plt.title('Age vs. Salary')
plt.show()
```



```
# Select the relevant columns for linear regression
X = data[['age', 'job_level', 'onsite', 'awards', 'certifications']]
y = data['salary']
```

```
print(X)
print(y)
```

```
         age  job_level  onsite  awards  certifications
     0    28          5       0       1               0
     1    50          3       1       2               1
     2    43          4       0       2               0
     3    44          2       1       0               0
     4    33          2       0       5               0
     ..  ...        ...     ...     ...             ...
     94   36          1       0       0               1
     95   44          4       1       5               1
     96   33          4       1       3               1
     97   54          1       1       7               1
     98   49          3       0       8               0

     [99 rows x 5 columns]
     0     86750
     1     42419
     2     65715
     3     29805
     4     29805
           ...
     94    24076
     95    65715
     96    65715
     97    24076
     98    42419
     Name: salary, Length: 99, dtype: int64
```

```
# Select the relevant columns for linear regression
X = data[['age', 'job_level', 'onsite', 'awards', 'certifications']]
y = data['salary']


# Create an instance of the Linear Regression model
model = LinearRegression()


# Fit the model to the data
model.fit(X, y)
```

```
      ▾ LinearRegression
      LinearRegression()
```

```
# Generate predictions
predictions = model.predict(X)


# Calculate the coefficient of determination (R-squared)
r_squared = model.score(X, y)


# Extract the coefficients and intercept
coefficients = model.coef_
intercept = model.intercept_


# Print the results
print("Coefficients:", coefficients)
print("Intercept:", intercept)
print("R-squared:", r_squared)
```

```
     Coefficients: [  -56.81411226 16304.25823398   747.36147054   -44.87473012
        290.24579805]
     Intercept: 2707.3308628208542
     R-squared: 0.9563559829427001
```

```
# Select the relevant features and target variable
X = data[['age', 'job_level', 'onsite', 'awards', 'certifications']]
y = data['satisfied']
```

```
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Create an instance of the KNN classifier
knn = KNeighborsClassifier(n_neighbors=5)
```

```
# Fit the classifier to the training data
knn.fit(X_train, y_train)
```

```
     ▾ KNeighborsClassifier
     KNeighborsClassifier()
```

```
# Generate predictions on the test set
predictions = knn.predict(X_test)
```

```
# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, predictions)
```

```
# Print the results
print("Accuracy:", accuracy)
```

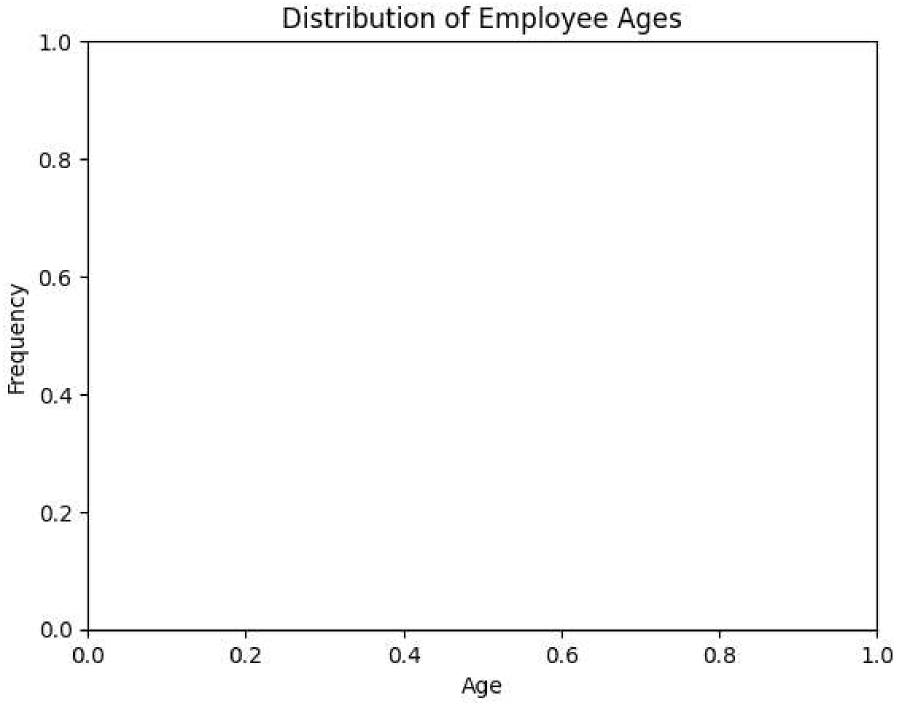
```
     Accuracy: 0.6
```

```
# Extract the 'age' column from the dataset
age_data = data['age']
```

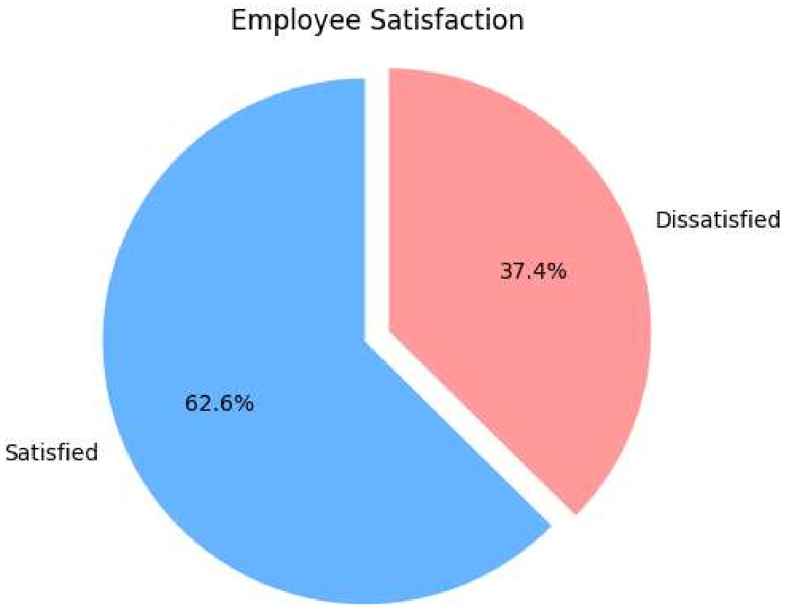
```
# Create a histogram
plt.hist(age_data, bins=10, edgecolor='black')
```

```
(array([17., 10.,  5., 12.,  6.,  7.,  7.,  9., 15., 11.]),
 array([23. , 26.1, 29.2, 32.3, 35.4, 38.5, 41.6, 44.7, 47.8, 50.9, 54. ]),
 <BarContainer object of 10 artists>)
```

```
# Set the labels and title
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Distribution of Employee Ages')
plt.show()
```



```
# Calculate the count of satisfied and dissatisfied employees
satisfied_count = data['satisfied'].sum()
dissatisfied_count = len(data) - satisfied_count


# Create a pie chart
labels = ['Satisfied', 'Dissatisfied']
sizes = [satisfied_count, dissatisfied_count]
colors = ['#66b3ff', '#ff9999']
explode = (0.1, 0)


plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle
plt.title('Employee Satisfaction')
plt.show()
```



✓ 0s completed at 10:26 AM