```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_csv('/content/Salary Data.csv')
```

```
df
```

	Age	Gender	Education Level	Job Title	Years of Experience	Salary
0	32.0	Male	Bachelor's	Software Engineer	5.0	90000.0
1	28.0	Female	Master's	Data Analyst	3.0	65000.0
2	45.0	Male	PhD	Senior Manager	15.0	150000.0
3	36.0	Female	Bachelor's	Sales Associate	7.0	60000.0
4	52.0	Male	Master's	Director	20.0	200000.0
...	...	...	...	...	...	...
370	35.0	Female	Bachelor's	Senior Marketing Analyst	8.0	85000.0
371	43.0	Male	Master's	Director of Operations	19.0	170000.0
372	29.0	Female	Bachelor's	Junior Project Manager	2.0	40000.0
373	34.0	Male	Bachelor's	Senior Operations Coordinator	7.0	90000.0
374	44.0	Female	PhD	Senior Business Analyst	15.0	150000.0

375 rows × 6 columns

```
df.head()
```

	Age	Gender	Education Level	Job Title	Years of Experience	Salary
0	32.0	Male	Bachelor's	Software Engineer	5.0	90000.0
1	28.0	Female	Master's	Data Analyst	3.0	65000.0
2	45.0	Male	PhD	Senior Manager	15.0	150000.0
3	36.0	Female	Bachelor's	Sales Associate	7.0	60000.0
4	52.0	Male	Master's	Director	20.0	200000.0

```
df.head(10)
```

	Age	Gender	Education Level	Job Title	Years of Experience	Salary
0	32.0	Male	Bachelor's	Software Engineer	5.0	90000.0
1	28.0	Female	Master's	Data Analyst	3.0	65000.0
2	45.0	Male	PhD	Senior Manager	15.0	150000.0
3	36.0	Female	Bachelor's	Sales Associate	7.0	60000.0
4	52.0	Male	Master's	Director	20.0	200000.0
5	29.0	Male	Bachelor's	Marketing Analyst	2.0	55000.0
6	42.0	Female	Master's	Product Manager	12.0	120000.0
7	31.0	Male	Bachelor's	Sales Manager	4.0	80000.0
8	26.0	Female	Bachelor's	Marketing Coordinator	1.0	45000.0
9	38.0	Male	PhD	Senior Scientist	10.0	110000.0

```
df.tail()
```

	Age	Gender	Education Level	Job Title	Years of Experience	Salary
370	35.0	Female	Bachelor's	Senior Marketing Analyst	8.0	85000.0
371	43.0	Male	Master's	Director of Operations	19.0	170000.0
372	29.0	Female	Bachelor's	Junior Project Manager	2.0	40000.0
373	34.0	Male	Bachelor's	Senior Operations Coordinator	7.0	90000.0
374	44.0	Female	PhD	Senior Business Analyst	15.0	150000.0

```
df.tail(10)
```

	Age	Gender	Education Level	Job Title	Years of Experience	Salary
365	43.0	Male	Master's	Director of Marketing	18.0	170000.0
366	31.0	Female	Bachelor's	Junior Financial Analyst	3.0	50000.0
367	41.0	Male	Bachelor's	Senior Product Manager	14.0	150000.0
368	44.0	Female	PhD	Senior Data Engineer	16.0	160000.0
369	33.0	Male	Bachelor's	Junior Business Analyst	4.0	60000.0
370	35.0	Female	Bachelor's	Senior Marketing Analyst	8.0	85000.0
371	43.0	Male	Master's	Director of Operations	19.0	170000.0
372	29.0	Female	Bachelor's	Junior Project Manager	2.0	40000.0
373	34.0	Male	Bachelor's	Senior Operations Coordinator	7.0	90000.0
374	44.0	Female	PhD	Senior Business Analyst	15.0	150000.0

```
df.isnull()
```

	Age	Gender	Education Level	Job Title	Years of Experience	Salary
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
370	False	False	False	False	False	False
371	False	False	False	False	False	False
372	False	False	False	False	False	False
373	False	False	False	False	False	False
374	False	False	False	False	False	False

375 rows × 6 columns

```
df.isnull().sum()
```

	0
Age	2
Gender	2
Education Level	2
Job Title	2
Years of Experience	2
Salary	2

```
df.shape
```

```
(375, 6)
```

```
df.size
```

```
2250
```

```
df.describe()
```

	Age	Years of Experience	Salary
count	373.000000	373.000000	373.000000
mean	37.431635	10.030831	100577.345845
std	7.069073	6.557007	48240.013482
min	23.000000	0.000000	350.000000
25%	31.000000	4.000000	55000.000000
50%	36.000000	9.000000	95000.000000
75%	44.000000	15.000000	140000.000000
max	53.000000	25.000000	250000.000000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 375 entries, 0 to 374
Data columns (total 6 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Age                  373 non-null    float64
 1   Gender               373 non-null    object 
 2   Education Level      373 non-null    object 
 3   Job Title            373 non-null    object 
 4   Years of Experience  373 non-null    float64
 5   Salary               373 non-null    float64
dtypes: float64(3), object(3)
memory usage: 17.7+ KB
```

```
df.fillna(df.mean(numeric_only=True), inplace=True)
```

```
from sklearn.preprocessing import LabelEncoder
for col in df.select_dtypes(include='object').columns:
    df[col].fillna("unknown", inplace=True)
    df[col] = LabelEncoder().fit_transform(df[col])
```

```
X = df.drop('Salary', axis=1)
```

```
X
```

	Age	Gender	Education Level	Job Title	Years of Experience
0	32.0	1	0	159	5.0
1	28.0	0	1	17	3.0
2	45.0	1	2	130	15.0
3	36.0	0	0	101	7.0
4	52.0	1	1	22	20.0
...	...	...	...	...	...
370	35.0	0	0	131	8.0
371	43.0	1	1	30	19.0
372	29.0	0	0	70	2.0
373	34.0	1	0	137	7.0
374	44.0	0	2	110	15.0

375 rows × 5 columns

```
y = df['Salary']
```

```
y
```

```
        Salary
  0    90000.0
  1    65000.0
  2   150000.0
  3    60000.0
  4   200000.0
 ...       ...
370    85000.0
371   170000.0
372    40000.0
373    90000.0
374   150000.0
375 rows × 1 columns
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```
#Train using Random Forest Regressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train_scaled, y_train)
```

```
▾    RandomForestRegressor    ⓘ ?
RandomForestRegressor(random_state=42)
```

```
predictions = model.predict(X_test_scaled)
```

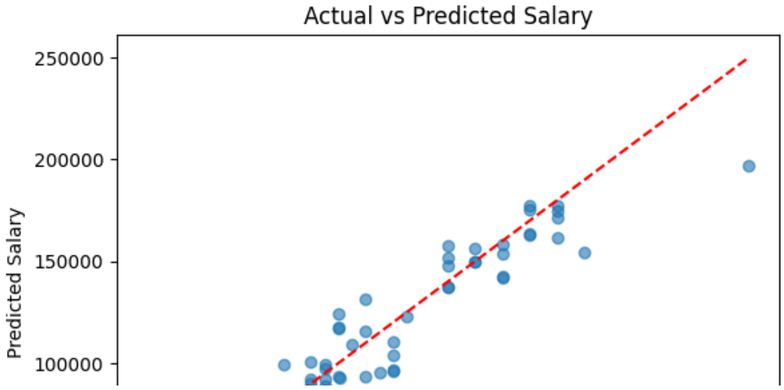
```
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(y_test, predictions)
r2 = r2_score(y_test, predictions)
rmse = np.sqrt(mse)

print("🎯 Improved Model Performance:")
print(f"Mean Squared Error: {mse:.2f}")
print(f"Root Mean Squared Error: {rmse:.2f}")
print(f"R² Score: {r2:.4f}")
```

```
🎯 Improved Model Performance:
Mean Squared Error: 175768619.24
Root Mean Squared Error: 13257.78
R² Score: 0.9274
```

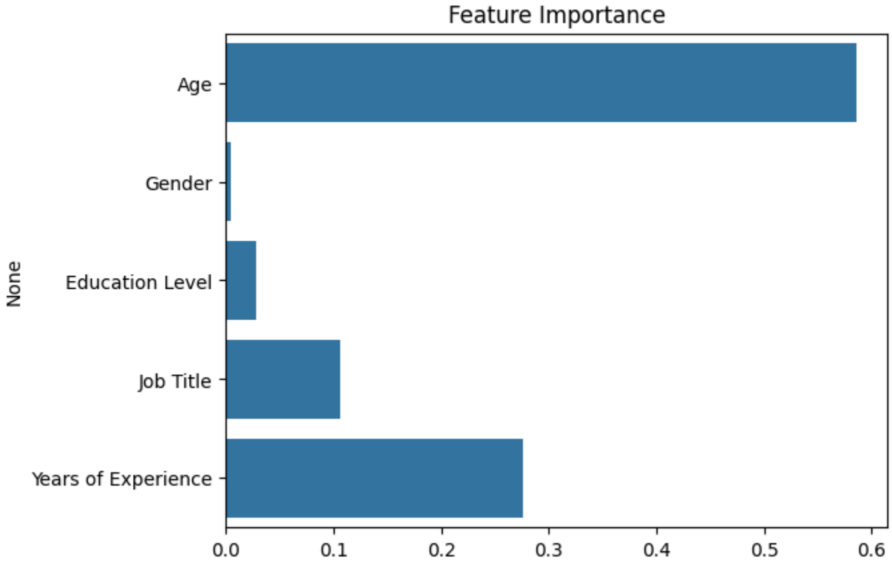
```
import matplotlib.pyplot as plt

plt.scatter(y_test, predictions, alpha=0.6)
plt.xlabel("Actual Salary")
plt.ylabel("Predicted Salary")
plt.title("Actual vs Predicted Salary")
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.show()
```



```
import seaborn as sns

feature_importance = model.feature_importances_
sns.barplot(x=feature_importance, y=X.columns)
plt.title("Feature Importance")
plt.show()
```



```
import joblib
joblib.dump(model, "salary_model.pkl")
joblib.dump(scaler, "scaler.pkl")  # if using StandardScaler
```

```
['scaler.pkl']
```