```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

```
df=pd.read_csv('/content/Lab_4_hiring.csv')
```

```
df.head
```

```
<bound method NDFrame.head of   experience  test_score(out of 10)  interview_score(out of 10)  salary($)
0        NaN                    8.0                           9      50000
1        NaN                    8.0                           6      45000
2       five                    6.0                           7      60000
3        two                   10.0                          10      65000
4      seven                    9.0                           6      70000
5      three                    7.0                          10      62000
6        ten                    NaN                           7      72000
7     eleven                    7.0                           8      80000>
```

```
df
```

	experience	test_score(out of 10)	interview_score(out of 10)	salary($)
0	NaN	8.0	9	50000
1	NaN	8.0	6	45000
2	five	6.0	7	60000
3	two	10.0	10	65000
4	seven	9.0	6	70000
5	three	7.0	10	62000
6	ten	NaN	7	72000
7	eleven	7.0	8	80000

```
df.dtypes
```

```
experience                      object
test_score(out of 10)          float64
interview_score(out of 10)       int64
salary($)                        int64
dtype: object
```

```
df.rename(columns={'test_score(out of 10)':'test_score'},inplace=True)
```

```
df.rename(columns={'interview_score(out of 10)':'interview_score'},inplace=True)
```

```
df.rename(columns={'salary($)':'salary'},inplace=True)
```

```
df['experience'] = pd.to_numeric(df['experience'], errors='coerce')
df['test_score'] = pd.to_numeric(df['test_score'], errors='coerce')
df['interview_score'] = pd.to_numeric(df['interview_score'], errors='coerce')
```

```
df.isna().sum()
```

```
experience         8
test_score         1
interview_score    0
salary             0
dtype: int64
```

```
for col in ['test_score']:
    df[col] = df[col].fillna(df[col].mean())
```

```
df['experience'] = df['experience'].fillna(0)
```

```
df.isna().sum()
```

```
experience         0
test_score         0
interview_score    0
salary             0
dtype: int64
```

```
X = df[['experience', 'test_score', 'interview_score']]
y = df['salary']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)



model = LinearRegression()



model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

```
Mean Squared Error: 350230496.6645901
```

```
new_applicants = pd.DataFrame({
    'experience': [3, 9],
    'test_score': [8, 10],
    'interview_score': [7, 9]
})
predicted_salaries = model.predict(new_applicants)
print(f"Predicted salaries: {predicted_salaries}")
```

```
Predicted salaries: [68516.19854362 65065.46292168]
```

```
applicant1 = np.array([3, 8, 7]).reshape(1, -1)
predicted_salary1 = model.predict(applicant1)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LinearRegression was
  warnings.warn(
```

```
predicted_salary1
```

```
array([68516.19854362])
```

```
applicant2 = np.array([9, 8, 9]).reshape(1, -1)
predicted_salary2 = model.predict(applicant2)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LinearRegression was
  warnings.warn(
```

```
predicted_salary2
```

```
array([62938.10373012])
```