```
import pandas as pd
```

```
#loading the dataset
df=pd.read_csv("hr_dashboard_data(1).csv")
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Name                  200 non-null    object 
 1   Age                   200 non-null    int64  
 2   Gender                200 non-null    object 
 3   Projects Completed    200 non-null    int64  
 4   Productivity (%)      200 non-null    int64  
 5   Satisfaction Rate (%) 200 non-null    int64  
 6   Feedback Score        200 non-null    float64
 7   Department            200 non-null    object 
 8   Position              200 non-null    object 
 9   Joining Date          200 non-null    object 
 10  Salary                200 non-null    int64  
dtypes: float64(1), int64(5), object(5)
memory usage: 17.3+ KB
```

```
df.head()
```

	Name	Age	Gender	Projects Completed	Productivity (%)	Satisfaction Rate (%)	Feedback Score	Department	Positi
0	Douglas Lindsey	25	Male	11	57	25	4.7	Marketing	Analy
1	Anthony Roberson	59	Female	19	55	76	2.8	IT	Manag
2	Thomas Miller	30	Male	8	87	10	2.4	IT	Analy
3	Joshua Lewis	26	Female	1	53	4	1.4	Marketing	Inte
4	Stephanie Bailey	43	Male	14	3	9	4.5	IT	Tea Lea

Next steps: Generate code with df New interactive sheet

```
df.tail()
```

	Name	Age	Gender	Projects Completed	Productivity (%)	Satisfaction Rate (%)	Feedback Score	Department	Pos
195	Stephanie Fisher	29	Female	9	32	87	3.5	HR	Deve
196	Jeremy Miller	26	Male	7	45	28	2.8	IT	Deve
197	Daniel Pierce	22	Male	3	36	77	1.6	Finance	
198	Michael Hernandez	36	Female	23	96	50	3.4	Marketing	Ma
199	Victor Gutierrez	43	Male	10	86	71	2.0	IT	

```
df.describe()
```

	Age	Projects Completed	Productivity (%)	Satisfaction Rate (%)	Feedback Score	Salary
count	200.000000	200.000000	200.000000	200.000000	200.000000	200.000000
mean	34.650000	11.455000	46.755000	49.935000	2.883000	76619.245000
std	9.797318	6.408849	28.530068	28.934353	1.123263	27082.299202
min	22.000000	0.000000	0.000000	0.000000	1.000000	30231.000000
25%	26.000000	6.000000	23.000000	25.750000	1.900000	53080.500000
50%	32.000000	11.000000	45.000000	50.500000	2.800000	80540.000000
75%	41.000000	17.000000	70.000000	75.250000	3.900000	101108.250000
max	60.000000	25.000000	98.000000	100.000000	4.900000	119895.000000

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

```
                           0
Name                       0
Age                        0
Gender                     0
Projects Completed         0
Productivity (%)           0
Satisfaction Rate (%)      0
Feedback Score             0
Department                 0
Position                   0
Joining Date               0
Salary                     0

dtype: int64
```

```
#datatypes checking
df.dtypes
```

```
                                0
Name                       object
Age                         int64
Gender                     object
Projects Completed          int64
Productivity (%)            int64
Satisfaction Rate (%)       int64
Feedback Score            float64
Department                 object
Position                   object
Joining Date               object
Salary                      int64

dtype: object
```

```
df.shape
```

```
(200, 11)
```

```
#importing necessary libraries
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score , mean_squared_error
```

```
df.head()
```

	Name	Age	Gender	Projects Completed	Productivity (%)	Satisfaction Rate (%)	Feedback Score	Department	Positi
0	Douglas Lindsey	25	Male	11	57	25	4.7	Marketing	Analy
1	Anthony Roberson	59	Female	19	55	76	2.8	IT	Manag
2	Thomas Miller	30	Male	8	87	10	2.4	IT	Analy
3	Joshua Lewis	26	Female	1	53	4	1.4	Marketing	Inte
4	Stephanie Bailey	43	Male	14	3	9	4.5	IT	Tea Le

Next steps: Generate code with df New interactive sheet

```
#feature enginerring
le=LabelEncoder()

df['Gender']=le.fit_transform(df['Gender'])
df['Department']=le.fit_transform(df['Department'])
df['Position']=le.fit_transform(df['Position'])
```

```
#additional feature engineering
le_gender=LabelEncoder()
le_dept=LabelEncoder()
le_pos=LabelEncoder()

df['Gender']=le_gender.fit_transform(df['Gender'])
df['Department']=le_dept.fit_transform(df['Department'])
df['Position']=le_pos.fit_transform(df['Position'])
```

```
#drop unwanted column
df_model=df.drop(columns=['Name','Joining Date'])
df_model.head()
```

	Age	Gender	Projects Completed	Productivity (%)	Satisfaction Rate (%)	Feedback Score	Department	Position	Salary
0	25	1	11	57	25	4.7	3	0	63596
1	59	0	19	55	76	2.8	2	3	112540
2	30	1	8	87	10	2.4	2	0	66292
3	26	0	1	53	4	1.4	3	1	38303
4	43	1	14	3	9	4.5	2	5	101133

Next steps: Generate code with `df_model` New interactive sheet

```
df.head()
```

	Name	Age	Gender	Projects Completed	Productivity (%)	Satisfaction Rate (%)	Feedback Score	Department	Positi
0	Douglas Lindsey	25	1	11	57	25	4.7	3	
1	Anthony Roberson	59	0	19	55	76	2.8	2	
2	Thomas Miller	30	1	8	87	10	2.4	2	
3	Joshua Lewis	26	0	1	53	4	1.4	3	
4	Stephanie Bailey	43	1	14	3	9	4.5	2	

Next steps: Generate code with `df` New interactive sheet

```
#define features(x) and target(y)

X=df_model.drop('Salary', axis=1)
y=df_model['Salary']
print(X.columns)
```

```
Index(['Age', 'Gender', 'Projects Completed', 'Productivity (%)',
       'Satisfaction Rate (%)', 'Feedback Score', 'Department', 'Position'],
      dtype='object')
```

```
#train_test_split the data
X_train,X_test,y_train,y_test=train_test_split(
    X,y, test_size=0.2,random_state=42
)
```

```
print(X_train.shape)
print(y_train.shape)
```

```
(160, 8)
(160,)
```

```
#create and train our linear regression model
model=LinearRegression()
model.fit(X_train,y_train)
```

```
▾ LinearRegression ⓘ ?
LinearRegression()
```

```
#make prediction
y_pred=model.predict(X_test)
```

```
#evaluate the model
r2=r2_score(y_test,y_pred)
mse=mean_squared_error(y_test,y_pred)
```

```
print("model Performance")
print("r2 score",r2)
print("MSE",mse)
```

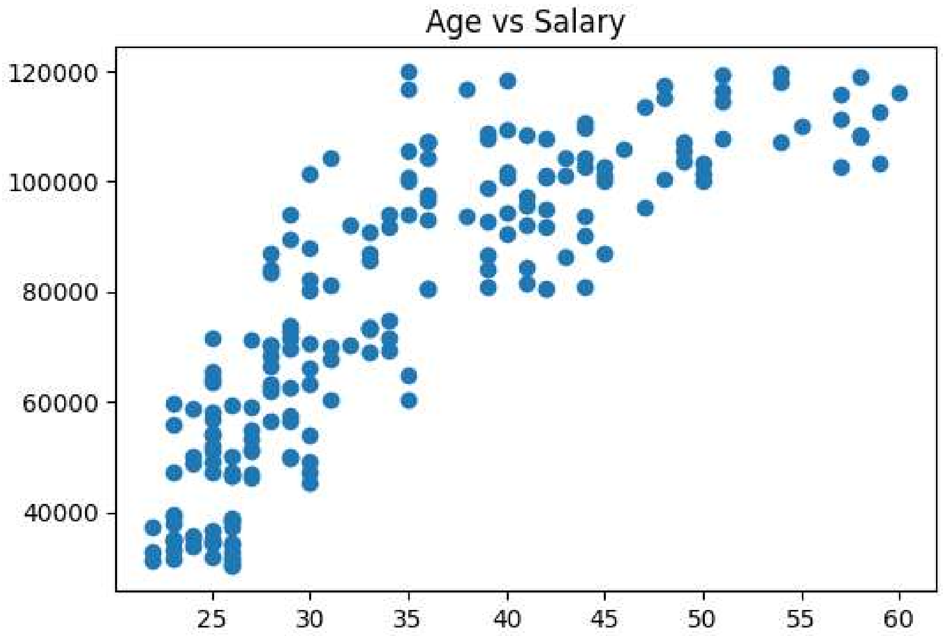
```
model Performance
r2 score 0.861580762838813
MSE 107995022.4628826
```

```
import matplotlib.pyplot as plt
```

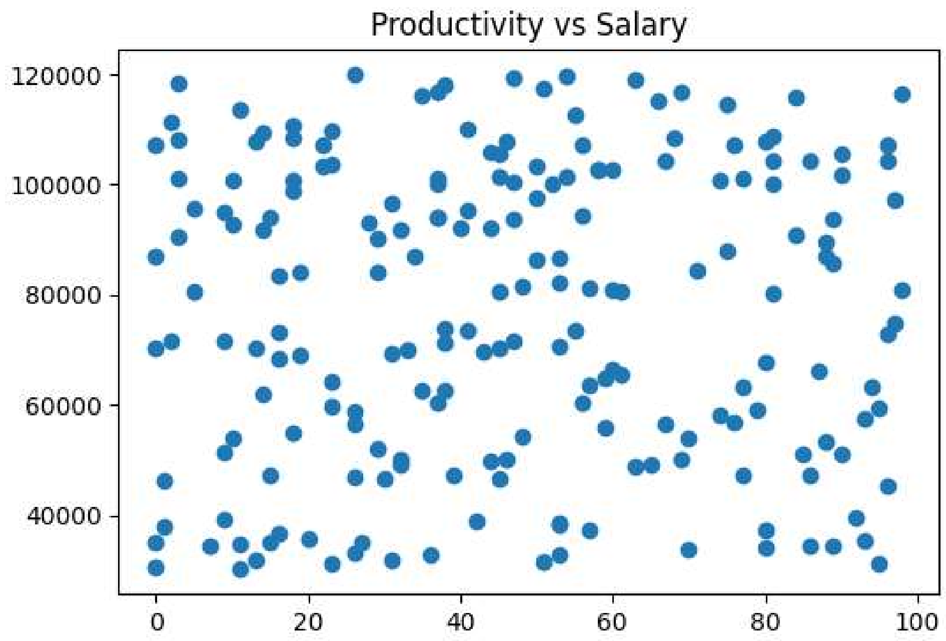
```
#salary Distribution
plt.figure(figsize=(6,4))
plt.hist(df['Salary'],bins=15)
plt.title("Salary Distribution")
plt.xlabel=("Salary")
plt.ylabel=("Frequency")
plt.show()
```

Salary Distribution

```
#Age vs Salary
plt.figure(figsize=(6,4))
plt.scatter(df['Age'],df['Salary'])
plt.title("Age vs Salary")
plt.xlabel=("Age")
plt.ylabel=("Salary")
plt.show()
```

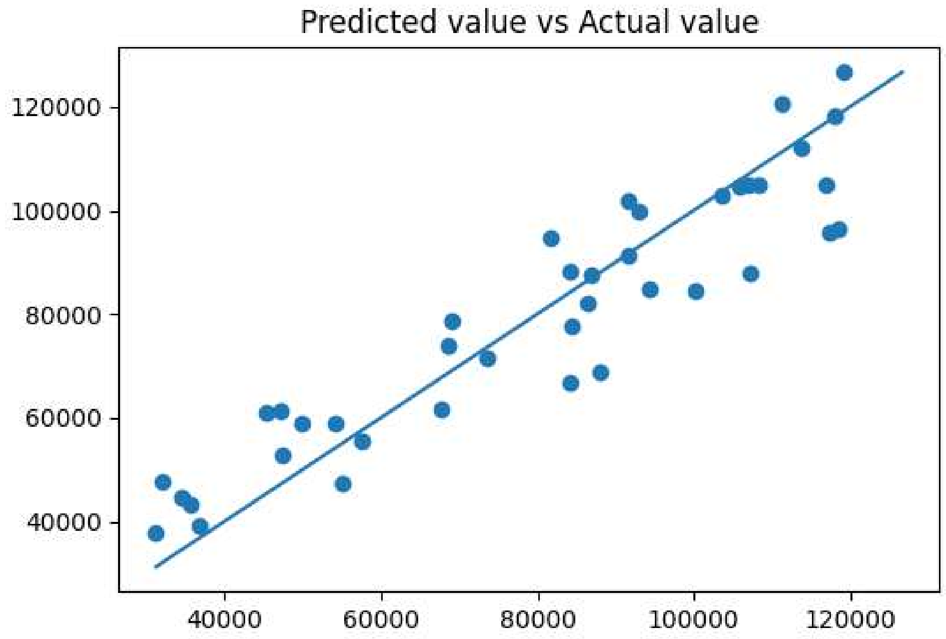


```
#Productivity vs salary
plt.figure(figsize=(6,4))
plt.scatter(df['Productivity (%)'],df['Salary'])
plt.title("Productivity vs Salary")
plt.xlabel=("Productivity")
plt.ylabel=("Salary")
plt.show()
```



```
#Actual vs Predicted
plt.figure(figsize=(6,4))
plt.scatter(y_test, y_pred)
plt.title("Predicted value vs Actual value")
plt.xlabel=("Actual Salary")
plt.ylabel=("Predicted Salary")


min_val=min(y_test.min(),y_pred.min())
max_val=max(y_test.min(),y_pred.max())
plt.plot([min_val, max_val],[min_val,max_val])
plt.show()
```



```
#importing gradio library
import gradio as gr
```

```
gender_options=list(le_gender.classes_)
dept_options=list(le_dept.classes_)
pos_options=list(le_pos.classes_)
```

```
def predict_salary_ui(age,gender,projects_completed,productivity,satisfaction,feedback_score,
    gender_encoded=le_gender.transform([gender])[0]
    dept_encoded=le_dept.transform([department])[0]
    pos_encoded= le_pos.transform([position])[0]
    input_df=pd.DataFrame({
        'Age':[age],
        'Gender':[gender_encoded],
        'Projects Completed':[projects_completed],
        'Productivity (%)':[productivity],
        'Satisfaction Rate (%)':[satisfaction],
        'Feedback Score':[feedback_score],
        'Department': [dept_encoded],
        'Position': [pos_encoded]
    })

      #to predict salary
    predict_salary=model.predict(input_df)[0]
    return f"Estimate Salary: ₹{predict_salary:.2f}"

age_input = gr.Slider(minimum=18, maximum=65, value=30, step=1, label="Age")
gender_input = gr.Dropdown(choices=gender_options, value=gender_options[0], label="Gender")
projects_input = gr.Slider(minimum=0, maximum=30, value=5, step=1, label="Projects Completed"
productivity_input = gr.Slider(minimum=0, maximum=100, value=50, step=1, label="Productivity
satisfaction_input = gr.Slider(minimum=0, maximum=100, value=50, step=1, label="Satisfaction
feedback_input = gr.Slider(minimum=0.0, maximum=5.0, value=3.0, step=0.1, label="Feedback Sco
department_input = gr.Dropdown(choices=dept_options, value=dept_options[0], label="Department"
position_input = gr.Dropdown(choices=pos_options, value=pos_options[0], label="Position")


output_box = gr.Textbox(label="Predicted Salary")

demo = gr.Interface(
    fn=predict_salary_ui,
    inputs=[
        age_input,
        gender_input,
        projects_input,
        productivity_input,
        satisfaction_input,
        feedback_input,
        department_input,
        position_input
    ],
    outputs=output_box,
    title="HR Salary Prediction App",
    description="Enter employee details to predict salary using a Linear Regression model."
)
```

```
demo.launch()
```

```
It looks like you are running Gradio on a hosted Jupyter notebook, which requires `share=True

Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://bddd0b017cb32b2c7e.gradio.live
```