Predicting Employee Attrition

Project ID - #CC69856

Name - Jyoti Ghaytadak

Project Level - Intermediate Level

Assigned By- CodeClause Internship

Project Details-

Aim -

Develop a model to predict the likelihood of employee attrition in a company.

Description -

Utilize HR data to build a classification model that predicts whether an employee is likely to leave the company.

Technologies -

Python, Pandas, Scikit-learn.

Import Libraries :

In [1]:
```
#Import Packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os
import re
import sys,traceback
```

In [2]:
```
'''Function to load the dataset'''
def data_init(data_filepath):
    try:
        hr = pd.read_csv(data_filepath,low_memory= False)

        col_list = list(hr)

        print("Loaded successfully.")

        return hr
    except:
        print("File Could not be loaded")
        print("Check your file or filepathname")
        return False
```

In [3]:
```
'''User interacive way to access the dataset'''
c = 1
while (c!=0):
    data_filepath = str(input("Enter data filepath:"))
    if os.path.isfile(data_filepath) :
        hr_data = data_init(data_filepath)
    else:
        '''Add double slash in filepath and try again!'''
        data_filepath = re.escape(data_filepath)
        hr_data = data_init(data_filepath)
    if type(hr_data) != str: c = 0
    else: print ("Check if file exists in the filepath and Let's try again ! \r
```

```
Enter data filepath:
File Could not be loaded
Check your file or filepathname
```

Load the data

In [20]:
```
df = pd.read_csv(r"C:\Users\C ZONE\Downloads\Dataset-Employee_Attrition.csv")
df.head()
```

Out[20]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
0	0.38	0.53	2	157	3
1	0.80	0.86	5	262	6
2	0.11	0.88	7	272	4
3	0.72	0.87	5	223	5
4	0.37	0.52	2	159	3

Dataset size

In [21]:
```
df.shape
```

Out[21]:
```
(14999, 10)
```

Data type of columns

In [22]:
```
df.dtypes
```

Out[22]:
```
satisfaction_level       float64
last_evaluation          float64
number_project             int64
average_montly_hours       int64
time_spend_company         int64
Work_accident              int64
left                       int64
promotion_last_5years      int64
Department                object
salary                    object
dtype: object
```

Data Information

In [23]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   satisfaction_level     14999 non-null  float64
 1   last_evaluation        14999 non-null  float64
 2   number_project         14999 non-null  int64  
 3   average_montly_hours   14999 non-null  int64  
 4   time_spend_company     14999 non-null  int64  
 5   Work_accident          14999 non-null  int64  
 6   left                   14999 non-null  int64  
 7   promotion_last_5years  14999 non-null  int64  
 8   Department             14999 non-null  object 
 9   salary                 14999 non-null  object 
dtypes: float64(2), int64(6), object(2)
memory usage: 1.1+ MB
```

Checking for duplicate record

In [24]:
```
df[df.duplicated()]
```

Out[24]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_com
396	0.46	0.57	2	139	
866	0.41	0.46	2	128	
1317	0.37	0.51	2	127	
1368	0.41	0.52	2	132	
1461	0.42	0.53	2	142	
...	...	...	...	...	
14994	0.40	0.57	2	151	
14995	0.37	0.48	2	160	
14996	0.37	0.53	2	143	
14997	0.11	0.96	6	280	
14998	0.37	0.52	2	158	

3008 rows × 10 columns

Drop duplicate Record/Rows

In [25]:
```
df=df.drop_duplicates()
df.shape
```

Out[25]:
```
(11991, 10)
```

Check For missing Values

In [26]:
```
df.isnull().sum()
```

Out[26]:
```
satisfaction_level       0
last_evaluation          0
number_project           0
average_montly_hours     0
time_spend_company       0
Work_accident            0
left                     0
promotion_last_5years    0
Department               0
salary                   0
dtype: int64
```

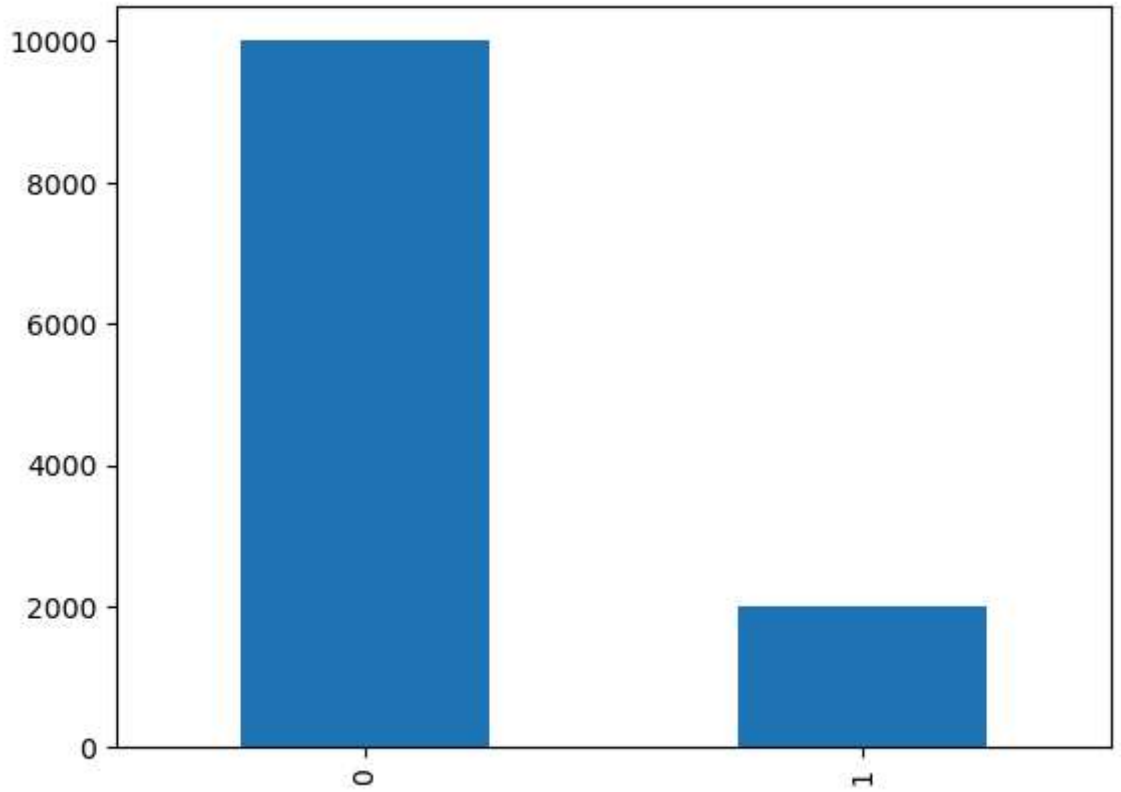
Data exploration and visualization

In [27]:
```
df['left'].value_counts()
```

Out[27]:
```
0    10000
1     1991
Name: left, dtype: int64
```

In [28]:
```
df['left'].value_counts().plot(kind = 'bar')
```

Out[28]:
```
<Axes: >
```



In [29]:
```
df.head()
```

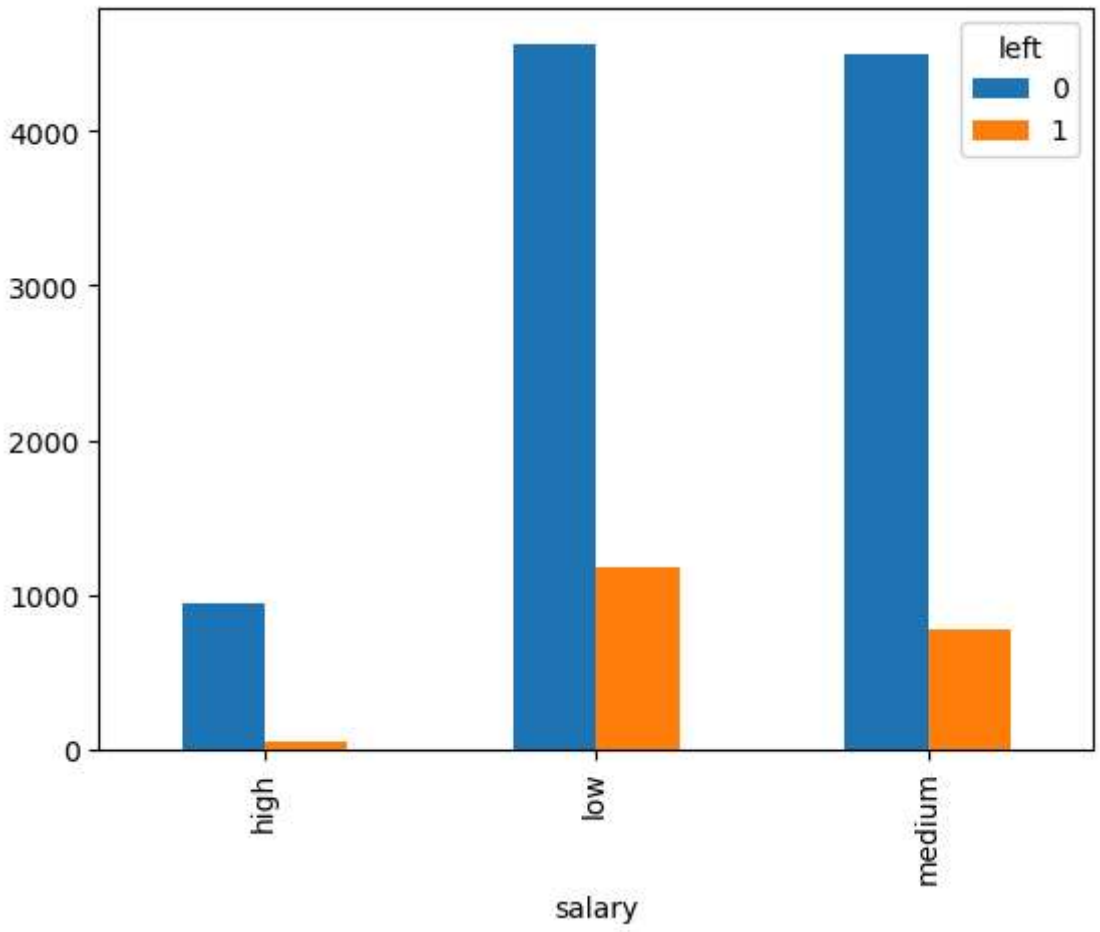
Out[29]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
0	0.38	0.53	2	157	3
1	0.80	0.86	5	262	6
2	0.11	0.88	7	272	4
3	0.72	0.87	5	223	5
4	0.37	0.52	2	159	3

Impact of salary on employee retention

In [30]:
```
pd.crosstab(df.salary,df.left).plot(kind='bar')
```

Out[30]:
```
<Axes: xlabel='salary'>
```



In [31]:
```
pd.crosstab(df.salary,df.left)
```

Out[31]:

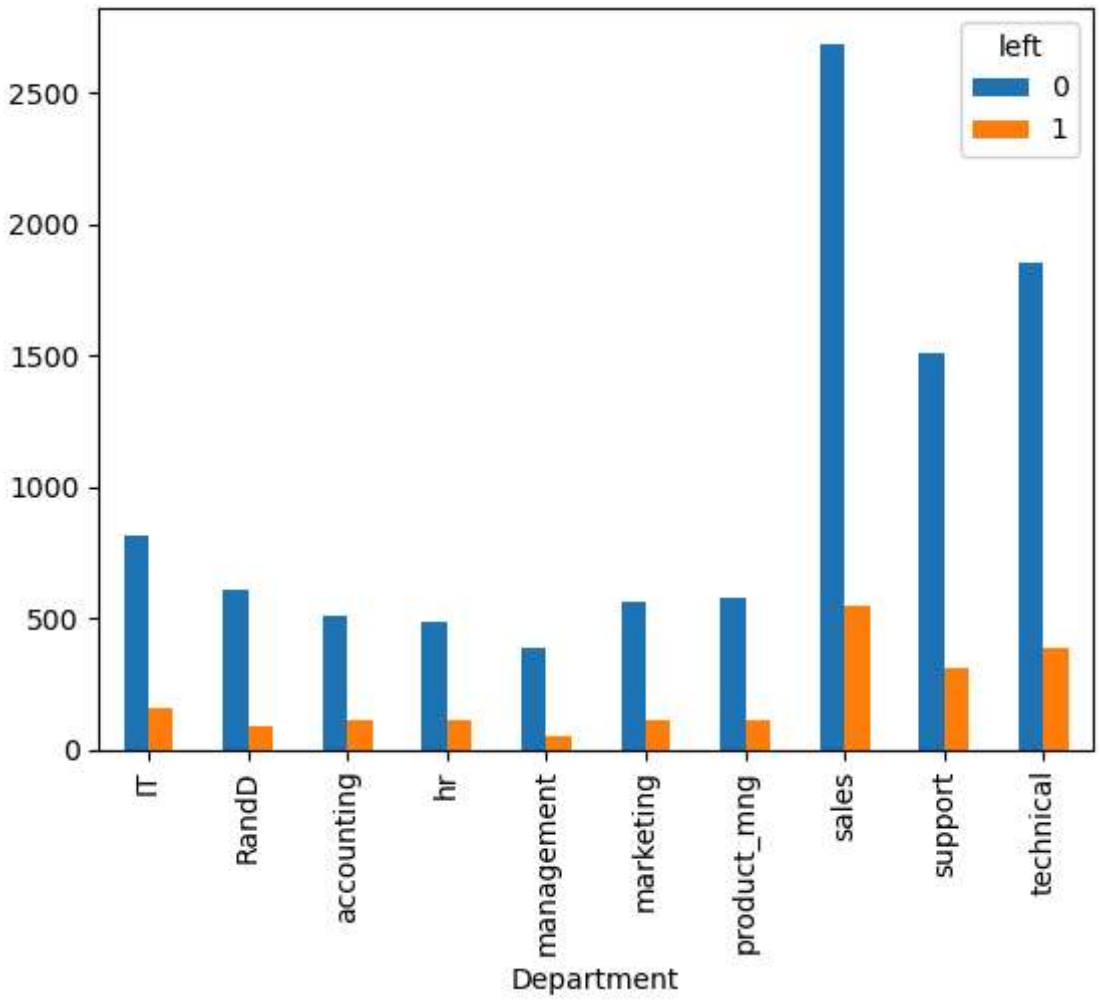
left	0	1
salary		
high	942	48
low	4566	1174
medium	4492	769

```
Above bar chart shows employeess with high salaries are likely to  not leave
the  company
```

Department wise employee retention rate

In [32]:
```
pd.crosstab(df.Department,df.left).plot(kind = 'bar')
```

Out[32]:
```
<Axes: xlabel='Department'>
```



In [33]:
```
pd.crosstab(df.Department,df.left)
```

Out[33]:

left	0	1
Department		
IT	818	158
RandD	609	85
accounting	512	109
hr	488	113
management	384	52
marketing	561	112
product_mng	576	110
sales	2689	550
support	1509	312
technical	1854	390

Distribution of each Numerical feature

In [34]:
```
num_feature_list1 =[f for f in df.columns  if df.dtypes[f]  =='float64' ]
num_feature_list1
```

Out[34]:
```
['satisfaction_level', 'last_evaluation']
```

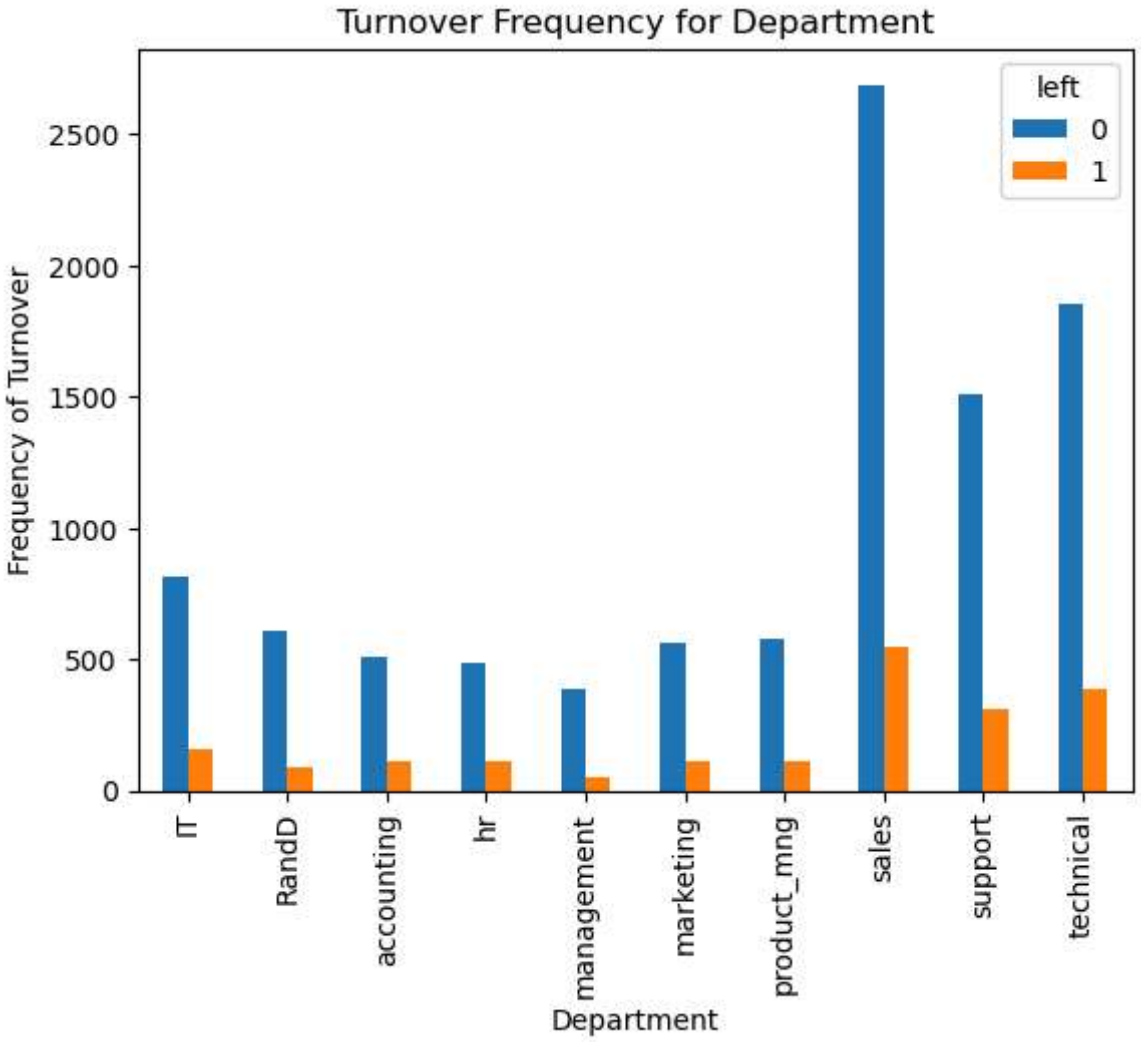
In [35]:
```
num_feature_list2 =[f for f in df.columns  if df.dtypes[f]  =='int64' ]
num_feature_list2
```

Out[35]:
```
['number_project',
 'average_montly_hours',
 'time_spend_company',
 'Work_accident',
 'left',
 'promotion_last_5years']
```

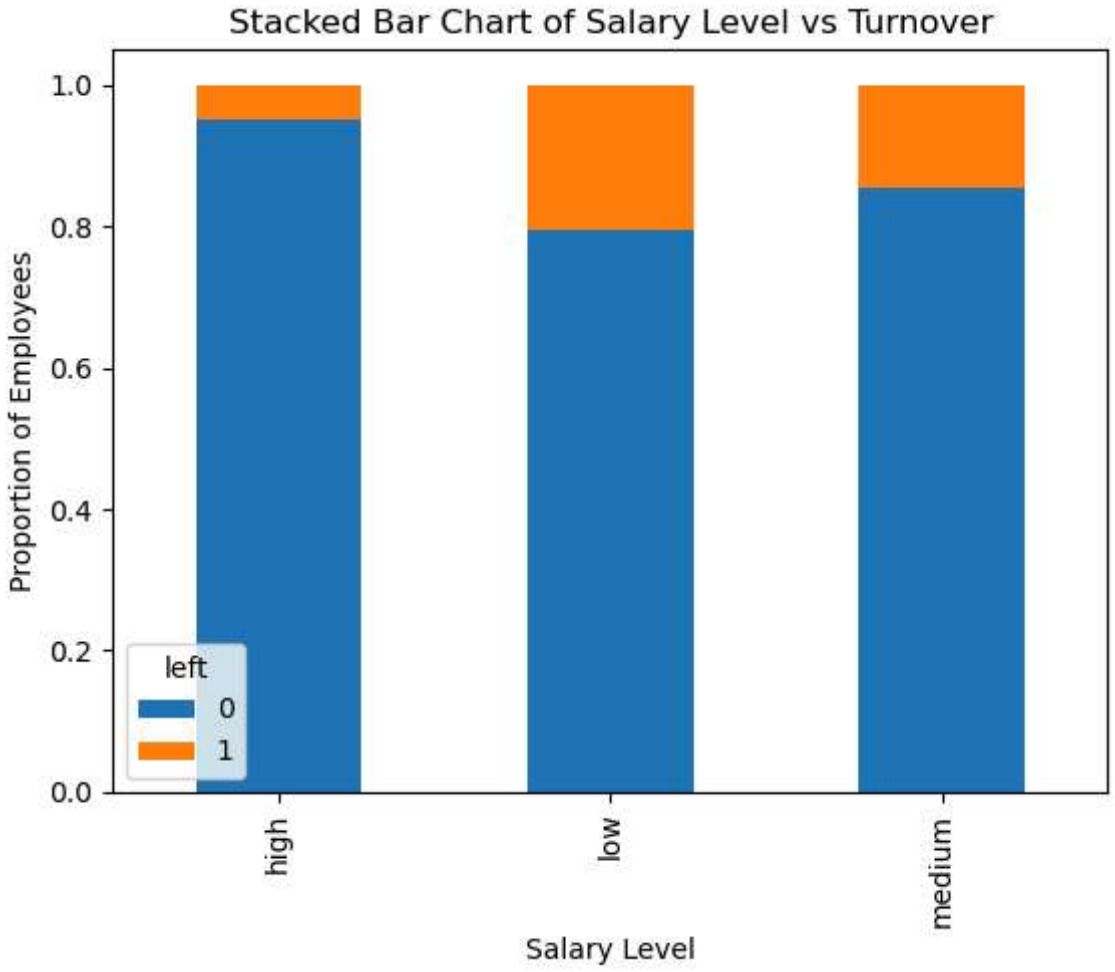
In [36]:
```
num_col_list =['number_project',
 'average_montly_hours',
 'time_spend_company',
 'Work_accident',
 'left',
 'promotion_last_5years','satisfaction_level', 'last_evaluation']
```

In [37]:
```
%matplotlib inline

#Bar chart for department employee work for and the frequency of turnover
pd.crosstab(df.Department,df.left).plot(kind='bar')
plt.title('Turnover Frequency for Department')
plt.xlabel('Department')
plt.ylabel('Frequency of Turnover')
plt.savefig('department_bar_chart')
```

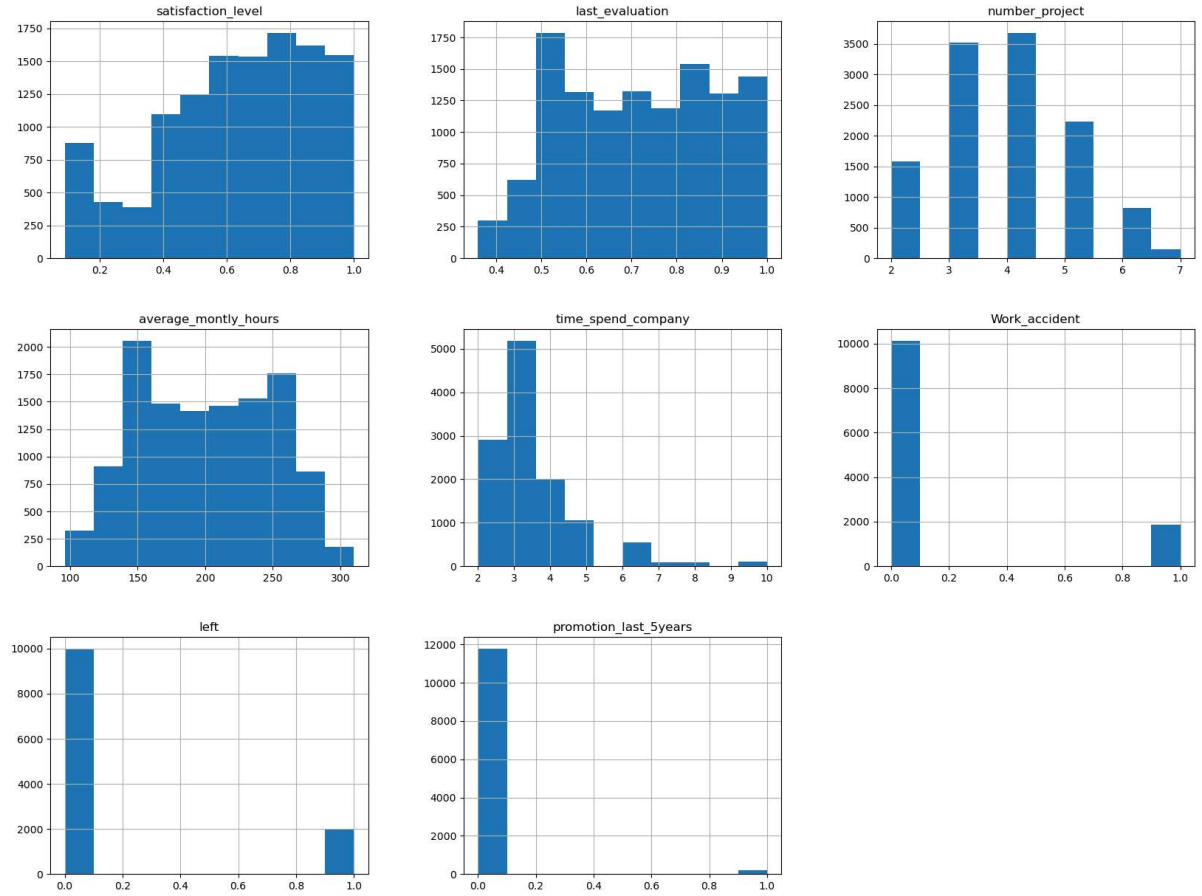


In [38]:
```
#Bar chart for employee salary level and the frequency of turnover
table=pd.crosstab(df.salary, df.left)
table.div(table.sum(1).astype(float), axis=0).plot(kind='bar', stacked=True)
plt.title('Stacked Bar Chart of Salary Level vs Turnover')
plt.xlabel('Salary Level')
plt.ylabel('Proportion of Employees')
plt.savefig('salary_bar_chart')
```



In [39]:
```
#Histogram of numeric variables
num_bins = 10

df.hist(bins=num_bins, figsize=(20,15))
plt.savefig("df_histogram_plots")
plt.show()
```



Create Dummy Variable for Categorical Variable

There are two categorical variables in the dataset and they need to be converted to dummy variables before they can be used for modelling

In [40]:
```
df.head()
```

Out[40]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
0	0.38	0.53	2	157	3
1	0.80	0.86	5	262	6
2	0.11	0.88	7	272	4
3	0.72	0.87	5	223	5
4	0.37	0.52	2	159	3

In [41]:
```
cat_vars=['Department','salary']
for var in cat_vars:
    cat_list='var'+'_'+var
    cat_list = pd.get_dummies(df[var], prefix=var)
    df1=df.join(cat_list)
    df=df1
```

In [42]:
```
df.drop(df.columns[[8, 9]], axis=1, inplace=True)
```

In [43]:
```
df.columns.values
```

Out[43]:
```
array(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_montly_hours', 'time_spend_company', 'Work_accident',
       'left', 'promotion_last_5years', 'Department_IT',
       'Department_RandD', 'Department_accounting', 'Department_hr',
       'Department_management', 'Department_marketing',
       'Department_product_mng', 'Department_sales', 'Department_support',
       'Department_technical', 'salary_high', 'salary_low',
       'salary_medium'], dtype=object)
```

In [44]:
```
df_vars=df.columns.values.tolist()
y=['left']
X=[i for i in df_vars if i not in y]
```

In [58]:
```
X
```

Out[58]:
```
['satisfaction_level',
 'last_evaluation',
 'number_project',
 'average_montly_hours',
 'time_spend_company',
 'Work_accident',
 'promotion_last_5years',
 'Department_IT',
 'Department_RandD',
 'Department_accounting',
 'Department_hr',
 'Department_management',
 'Department_marketing',
 'Department_product_mng',
 'Department_sales',
 'Department_support',
 'Department_technical',
 'salary_high',
 'salary_low',
 'salary_medium']
```

In [59]:
```
cols=['satisfaction_level',
 'last_evaluation',
 'number_project',
 'average_montly_hours',
 'time_spend_company',
 'Work_accident',
 'promotion_last_5years',
 'Department_IT',
 'Department_RandD',
 'Department_accounting',
 'Department_hr',
 'Department_management',
 'Department_marketing',
 'Department_product_mng',
 'Department_sales',
 'Department_support',
 'Department_technical',
 'salary_high',
 'salary_low',
 'salary_medium']
X=df[cols]
y=df['left']
```

Conclusion -

Random Forest is the best classfier for predicting employee attrition for our dataset. Some of the most important factors on which employee attrition depends are

Satisfaction Level

Tenure with organisation

Time since last evaluation

Work Accident

Salary

Department

Career Advancement (If Promoted in last five years or not)