In [2]:
```
import pandas as pd
```

In [3]:
```
a=pd.read_csv("HR_comma_sep.csv")
```

In [4]:
```
a.head(10)
```

Out[4]:

	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
5	0.41	0.50	2	153	3
6	0.10	0.77	6	247	4
7	0.92	0.85	5	259	5
8	0.89	1.00	5	224	5
9	0.42	0.53	2	142	3

In [5]:
```
a
```

Out[5]:

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

14999 rows × 10 columns

In [6]:
```
a.describe()
```

Out[6]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_compa
count	14999.000000	14999.000000	14999.000000	14999.000000	14999.0000
mean	0.612834	0.716102	3.803054	201.050337	3.498
std	0.248631	0.171169	1.232592	49.943099	1.460
min	0.090000	0.360000	2.000000	96.000000	2.000
25%	0.440000	0.560000	3.000000	156.000000	3.000
50%	0.640000	0.720000	4.000000	200.000000	3.000
75%	0.820000	0.870000	5.000000	245.000000	4.000
max	1.000000	1.000000	7.000000	310.000000	10.000

In [7]:

```
a.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
```

In [8]:

```
a.isnull().sum()
```

Out[8]:

```
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
```

In [9]:

```
b=a.drop(['last_evaluation','number_project','time_spend_company','Work_accident','
```

In [10]:

```
b.head(10)
```

Out[10]:

	satisfaction_level	average_montly_hours	left	promotion_last_5years	salary
0	0.38	157	1	0	low
1	0.80	262	1	0	medium
2	0.11	272	1	0	medium
3	0.72	223	1	0	low
4	0.37	159	1	0	low
5	0.41	153	1	0	low
6	0.10	247	1	0	low
7	0.92	259	1	0	low
8	0.89	224	1	0	low
9	0.42	142	1	0	low

In [11]:

```
b.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 5 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   satisfaction_level     14999 non-null  float64
 1   average_montly_hours   14999 non-null  int64  
 2   left                   14999 non-null  int64  
 3   promotion_last_5years  14999 non-null  int64  
 4   salary                 14999 non-null  object 
dtypes: float64(1), int64(3), object(1)
memory usage: 586.0+ KB
```

In [12]:

```
b.describe()
```

Out[12]:

	satisfaction_level	average_montly_hours	left	promotion_last_5years
count	14999.000000	14999.000000	14999.000000	14999.000000
mean	0.612834	201.050337	0.238083	0.021268
std	0.248631	49.943099	0.425924	0.144281
min	0.090000	96.000000	0.000000	0.000000
25%	0.440000	156.000000	0.000000	0.000000
50%	0.640000	200.000000	0.000000	0.000000
75%	0.820000	245.000000	0.000000	0.000000
max	1.000000	310.000000	1.000000	1.000000

In [13]:

```
b=pd.get_dummies(b,dtype=int)
```

In [14]:

```
b.head(10)
```

Out[14]:

	satisfaction_level	average_montly_hours	left	promotion_last_5years	salary_high	salary_low	sa
0	0.38	157	1	0	0	1	
1	0.80	262	1	0	0	0	
2	0.11	272	1	0	0	0	
3	0.72	223	1	0	0	1	
4	0.37	159	1	0	0	1	
5	0.41	153	1	0	0	1	
6	0.10	247	1	0	0	1	
7	0.92	259	1	0	0	1	
8	0.89	224	1	0	0	1	
9	0.42	142	1	0	0	1	

In [15]:

```
b.describe()
```

Out[15]:

	satisfaction_level	average_montly_hours	left	promotion_last_5years	salary_high
count	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000
mean	0.612834	201.050337	0.238083	0.021268	0.082472
std	0.248631	49.943099	0.425924	0.144281	0.275092
min	0.090000	96.000000	0.000000	0.000000	0.000000
25%	0.440000	156.000000	0.000000	0.000000	0.000000
50%	0.640000	200.000000	0.000000	0.000000	0.000000
75%	0.820000	245.000000	0.000000	0.000000	0.000000
max	1.000000	310.000000	1.000000	1.000000	1.000000

In [16]:

```
b.shape
```

Out[16]:

```
(14999, 7)
```

In [17]:

```
corr_mat=b.corr()
corr_mat
```

Out[17]:

	satisfaction_level	average_montly_hours	left	promotion_last_5years	s
satisfaction_level	1.000000	-0.020048	-0.388375	0.025605	
average_montly_hours	-0.020048	1.000000	0.071287	-0.003544	
left	-0.388375	0.071287	1.000000	-0.061788	
promotion_last_5years	0.025605	-0.003544	-0.061788	1.000000	
salary_high	0.029708	-0.007101	-0.120929	0.076756	
salary_low	-0.047415	-0.001050	0.134722	-0.082832	
salary_medium	0.031367	0.005007	-0.068833	0.040985	

In [ ]:

In [18]:

```
y=b['left']
x=b.drop(['left'],axis=1)
```

In [19]:

```
x
```

Out[19]:

	satisfaction_level	average_montly_hours	promotion_last_5years	salary_high	salary_low	sal
0	0.38	157	0	0	1	
1	0.80	262	0	0	0	
2	0.11	272	0	0	0	
3	0.72	223	0	0	1	
4	0.37	159	0	0	1	
...	...	...	...	...	...	
14994	0.40	151	0	0	1	
14995	0.37	160	0	0	1	
14996	0.37	143	0	0	1	
14997	0.11	280	0	0	1	
14998	0.37	158	0	0	1	

14999 rows × 6 columns

In [20]:

```
y
```

Out[20]:

```
0        1
1        1
2        1
3        1
4        1
        ..
14994    1
14995    1
14996    1
14997    1
14998    1
Name: left, Length: 14999, dtype: int64
```

In [21]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.36,random_state=45)
```

In [22]:
```
from sklearn.linear_model import LogisticRegression
log=LogisticRegression()
log.fit(x_train,y_train)
```

Out[22]:
```
LogisticRegression()
```

In [23]:
```
y_pred=log.predict(x_test)
```

In [24]:
```
y_pred
```

Out[24]:
```
array([1, 1, 0, ..., 0, 0, 0], dtype=int64)
```

In [25]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[25]:
```
array([[3838,  283],
       [ 933,  346]], dtype=int64)
```

In [26]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[26]:
```
0.7748148148148148
```

In [27]:
```
res=pd.DataFrame(columns=['left','predicted'])
res['left']=y_test
res['predicted']=y_pred
res=res.reset_index()
res['ID']=res.index
```

In [28]:
```
res.head(10)
```

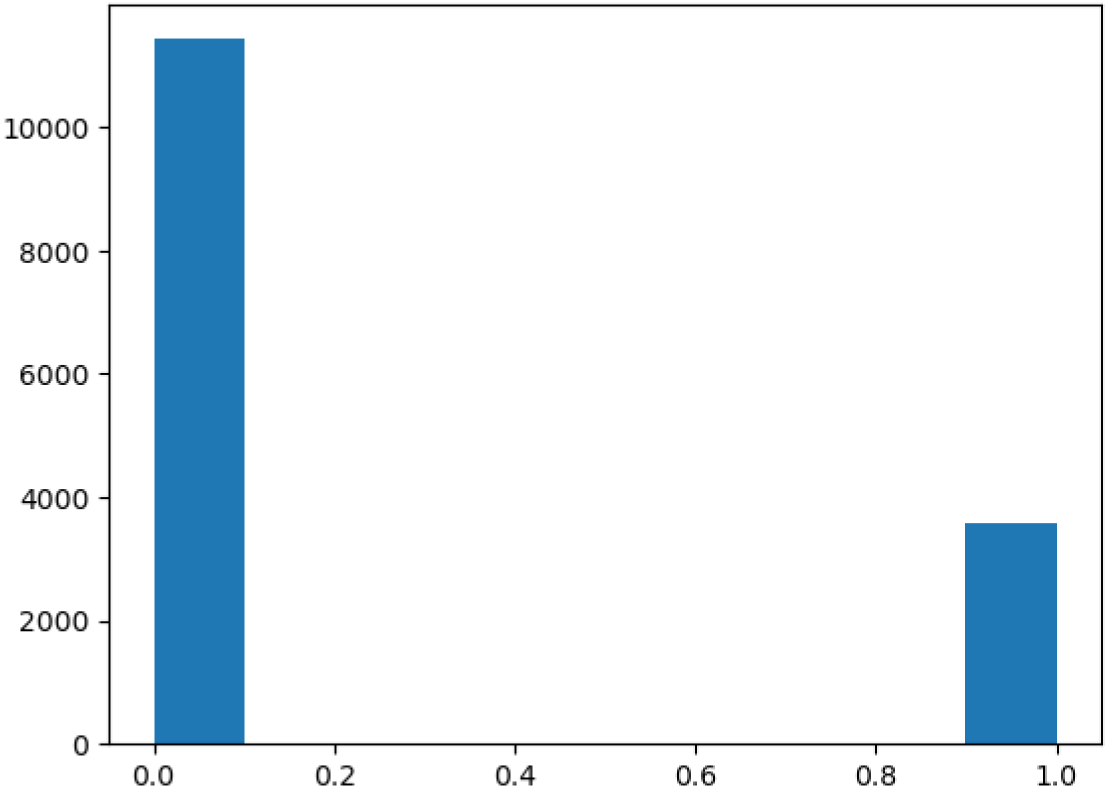
Out[28]:

	index	left	predicted	ID
0	3059	0	1	0
1	386	1	1	1
2	12830	0	0	2
3	4212	0	1	3
4	14609	1	0	4
5	11896	0	0	5
6	14839	1	0	6
7	33	1	0	7
8	14283	1	0	8
9	12337	1	0	9

In [29]:
```
import matplotlib.pyplot as plt
```

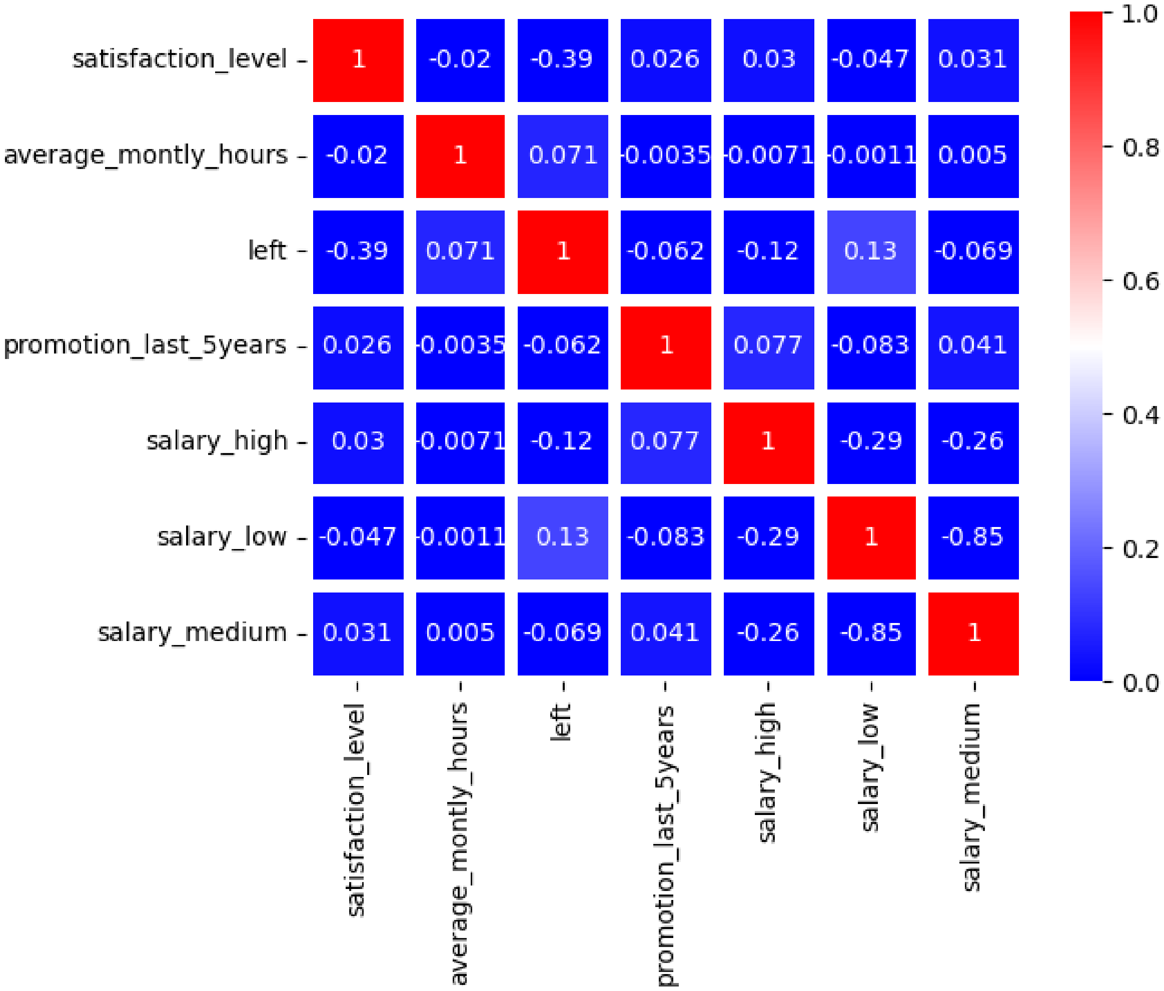
In [30]:
```
plt.hist(b['left'])
```

```
Out[30]: (array([11428.,     0.,     0.,     0.,     0.,     0.,     0.,     0.,
             0.,  3571.]),
          array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ]),
          <BarContainer object of 10 artists>)
```



```
In [31]: import seaborn as sns
         sns.heatmap(corr_mat,vmax=1,vmin=0,annot=True,linewidth=5,cmap='bwr')
```

```
Out[31]: <AxesSubplot:>
```



```
In [32]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [35]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```