Predicting employee attrition using Logistic Regression

In [129]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, f1_score
from sklearn.metrics import accuracy_score as acs
from sklearn.metrics import precision_recall_fscore_support as score

hr_data = pd.read_csv('HR_data.csv')
hr_data.head()
```

Out[129]:

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

In [130]:
```
X = hr_data.drop('left', axis=1)
y = hr_data['left'].values.ravel()
```

Preprocessing features

In [138]:
```
#Label encoding the salary variable

label_enc = LabelEncoder()
X['salary_level'] = label_enc.fit_transform(X[['salary']])
X_LE = X.drop('salary', axis=1)
X_LE.shape
```

Out[138]: (14999, 9)

In [132]:
```
#Dummy Encoding the department variable

dept_dummies = pd.get_dummies(X_LE['Department'])
merged_X = pd.concat([X_LE, dept_dummies], axis=1)
final_X = merged_X.drop(['Department'], axis=1)
final_X = final_X.drop(['RandD'], axis=1)

final_X.head()
```

Out[132]:

	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 [133]:
```
#Define my train and test sets

X_train, X_test, y_train, y_test = train_test_split(final_X, y, test_size=0.25, r
print(X_train.shape)
print(X_test.shape)
```

```
(11249, 17)
(3750, 17)
```

In [134]:
```
#Create and train my logistic regression model

logreg = LogisticRegression(max_iter=500).fit(X_train, y_train)

#predict using the model
y_pred = logreg.predict(X_test)
```

In [135]:
```
#Define my performance metrics

cm=confusion_matrix(y_pred,y_test)
class_label = ["Left", "Retained"]
df_cm = pd.DataFrame(cm, index=class_label,columns=class_label)

precision, recall, fscore, train_support = score(y_test, y_pred, average='binary'
print('Precision: {} \nRecall: {} \nF1-Score: {} \nAccuracy: {}'.format(
    round(precision, 3), round(recall, 3), round(fscore,3), round(acs(y_test,y_pr

sns.heatmap(df_cm,annot=True,cmap='Pastel1',linewidths=2,fmt='d')
plt.title("Confusion Matrix",fontsize=15)
plt.xlabel("Predicted")
plt.ylabel("True")
plt.show()
```

```
F1-Score: 0.333
Accuracy: 77%
```