SCI015 Mini-Project:

Employee Turnover Rate

Done By:
Nithin Raj Murali Babu (U2223360E)
Ke Nora (U2230249C)

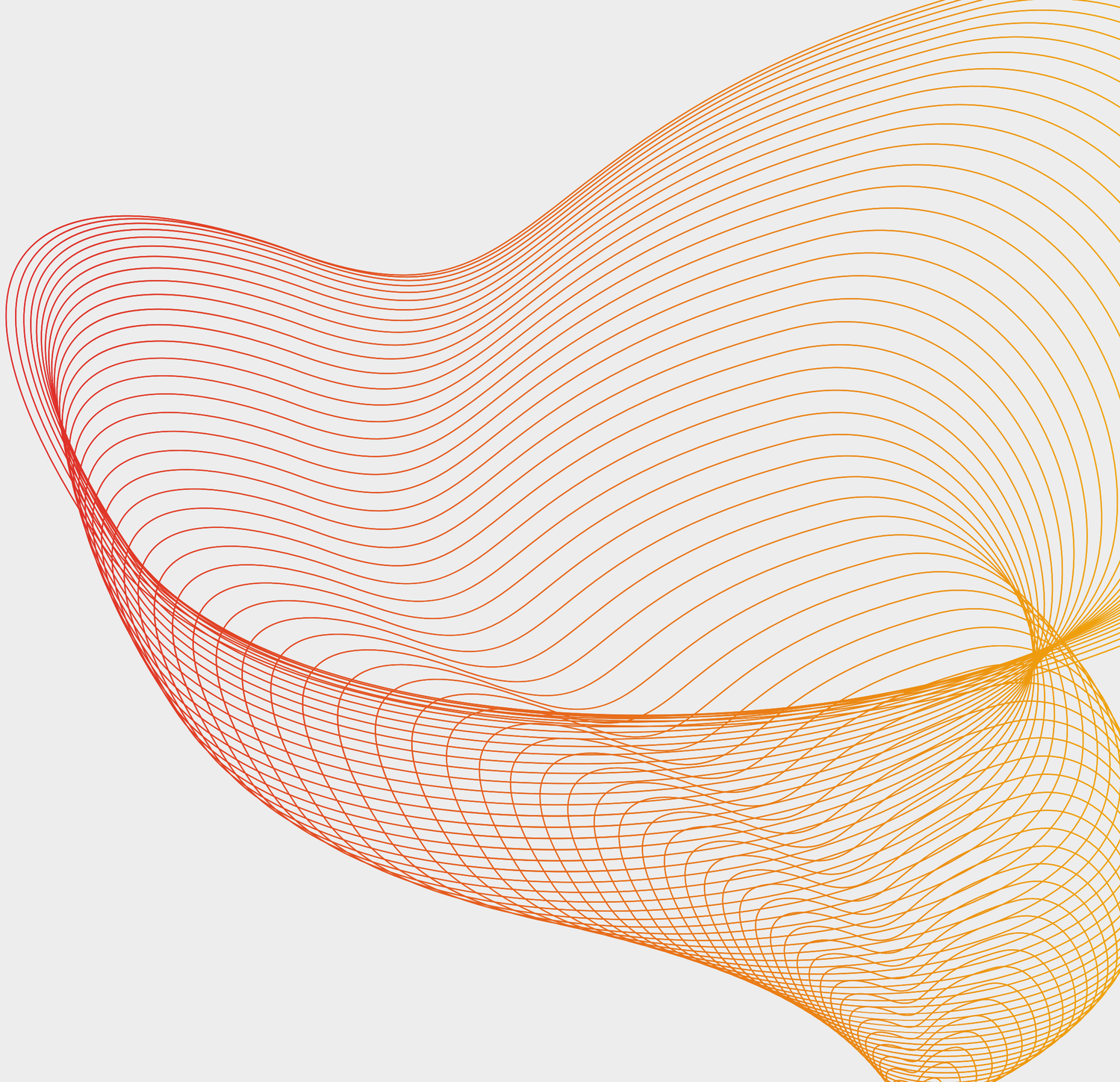


Table of Contents

Introduction

Employee Turnover Rate

Definition:

A measurement of how many employees leave an organisation over a specific period of time.

- could be caused by many factors
- could be voluntary or involuntary

Problem Definition

- What are the most effective variables in predicting employee turnover rate?
- Can we come up with a model to predict employee turnover using a combination of these factors?

Exploratory Analysis

Dataset taken from kaggle

Data Preparation

In [1]:
```
#Essential Libraries & Data File
import pandas as pd
import numpy as np
import seaborn as sb
sb.set()
import matplotlib.pyplot as plt

ETRData = pd.read_csv("IBM Employee data (train).csv")
print("Size of Data is ", ETRData.shape)
```

```
Size of Data is  (1058, 35)
```

Data Preparation

```
In [2]: # Brief Look into data set

        ETRData.head(n = 20)
```

Out[2]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipS
0	41	1	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	
1	49	0	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	
2	37	1	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	
3	33	0	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	
4	27	0	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	

Data Preparation

```
In [3]: #variables
        ETRData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1058 entries, 0 to 1057
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       1058 non-null   int64
 1   Attrition                 1058 non-null   int64
 2   BusinessTravel            1058 non-null   object
 3   DailyRate                 1058 non-null   int64
 4   Department                1058 non-null   object
 5   DistanceFromHome          1058 non-null   int64
 6   Education                 1058 non-null   int64
 7   EducationField            1058 non-null   object
 8   EmployeeCount             1058 non-null   int64
 9   EmployeeNumber            1058 non-null   int64
 10  EnvironmentSatisfaction   1058 non-null   int64
 11  Gender                    1058 non-null   object
 12  HourlyRate                1058 non-null   int64
 13  JobInvolvement            1058 non-null   int64
 14  JobLevel                  1058 non-null   int64
 15  JobRole                   1058 non-null   object
 16  JobSatisfaction           1058 non-null   int64
 17  MaritalStatus             1058 non-null   object
 18  MonthlyIncome             1058 non-null   int64
```

Data Preparation

In [4]:
```
#Quick analysis of Variables

ETRData.describe()
```

Out[4]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobI
count	1058.000000	1058.000000	1058.000000	1058.000000	1058.000000	1058.0	1058.000000	1058.000000	1058.000000	
mean	37.055766	0.169187	809.542533	8.978261	2.879017	1.0	731.753308	2.712665	65.643667	
std	9.410421	0.375094	408.478049	8.040608	1.031442	0.0	431.418209	1.092959	20.324861	
min	18.000000	0.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000	30.000000	
25%	30.000000	0.000000	465.250000	2.000000	2.000000	1.0	364.500000	2.000000	48.000000	
50%	36.000000	0.000000	817.500000	7.000000	3.000000	1.0	723.500000	3.000000	65.000000	
75%	43.000000	0.000000	1168.500000	13.000000	4.000000	1.0	1101.750000	4.000000	83.000000	
max	60.000000	1.000000	1499.000000	29.000000	5.000000	1.0	1487.000000	4.000000	100.000000	

8 rows × 27 columns

Data Cleaning

To clean this dataset, we have to first check if there any duplicate values or Null values that we need to remove.

In [5]:
```
#Finding Duplicates
print("Total duplicates in dataset is ", ETRData.duplicated().sum())
```

```
Total duplicates in dataset is  0
```

Data Cleaning

In [6]:
```
#Finding NULL Values
ETRData.isnull()
```

Out[6]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	Relati
0	False	False	False	False	False	False	False	False	False	False	...	
1	False	False	False	False	False	False	False	False	False	False	...	
2	False	False	False	False	False	False	False	False	False	False	...	
3	False	False	False	False	False	False	False	False	False	False	...	
4	False	False	False	False	False	False	False	False	False	False	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1053	False	False	False	False	False	False	False	False	False	False	...	
1054	False	False	False	False	False	False	False	False	False	False	...	
1055	False	False	False	False	False	False	False	False	False	False	...	
1056	False	False	False	False	False	False	False	False	False	False	...	
1057	False	False	False	False	False	False	False	False	False	False	...	

1058 rows × 35 columns

From a first glance, we can see there is no null values, but to really prove there is none we use the following code:

In [7]:
```
if ETRData.isnull().any().any():
    print("There are null values present.")
else:
    print("There are no null values present.")
```

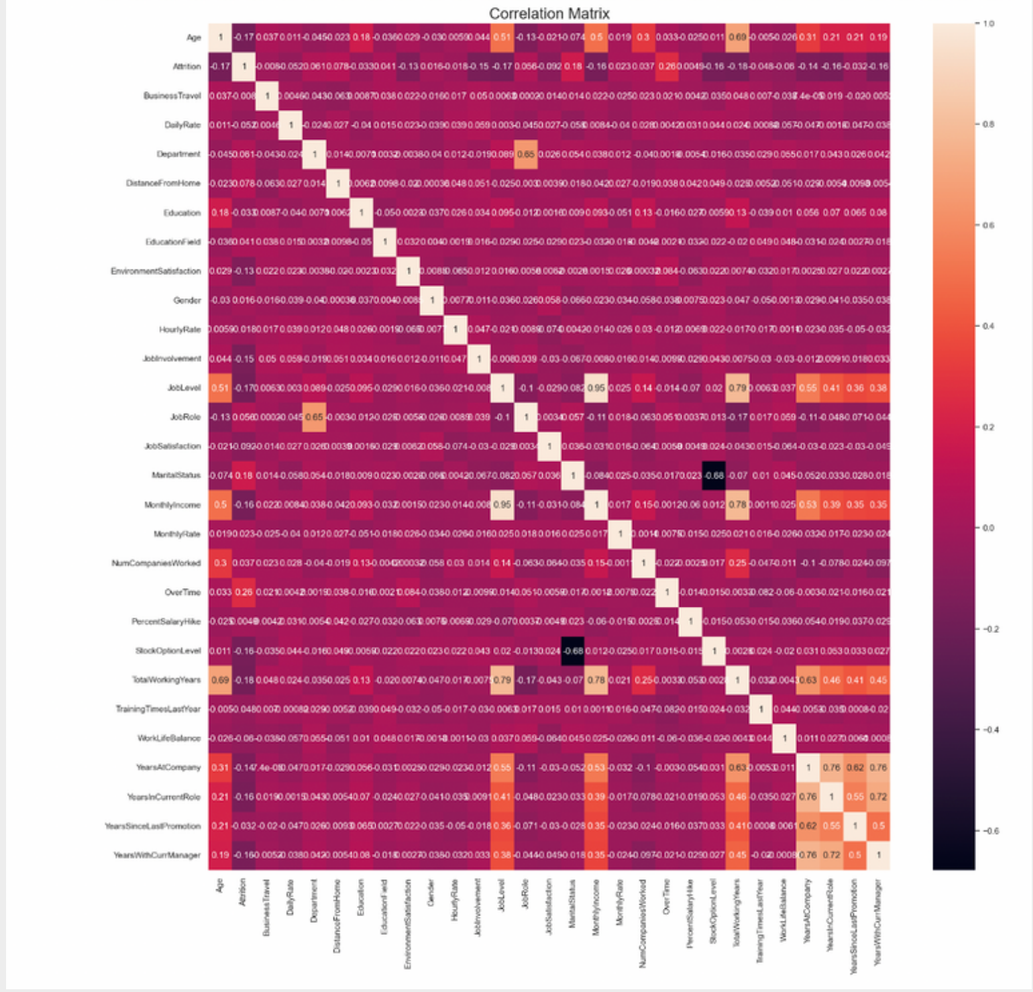
```
There are no null values present.
```

Exploratory Analysis

Age
Attrition
BusinessTravel
DailyRate
Department
DistanceFromHome
Education
EducationField
EnvironmentSatisfaction
Gender
HourlyRate
JobInvolvement
JobLevel
JobRole
JobSatisfaction
MaritalStatus
MonthlyIncome
MonthlyRate
NumCompaniesWorked
OverTime
PercentSalaryHike
StockOptionLevel
TotalWorkingYears
TrainingTimesLastYear
WorkLifeBalance
YearsAtCompany
YearsInCurrentRole
YearsSinceLastPromotion
YearsWithCurrManager

```
In [13]: ETRdf = pd.DataFrame(ETRData)

         f, axes = plt.subplots(29, 3, figsize=(90, 90))

         position = 0
         for i in ETRdf:
             sb.boxplot(data = ETRdf[i], orient = "h", ax = axes[position,0])
             sb.histplot(data = ETRdf[i], ax = axes[position,1])
             sb.violinplot(data = ETRdf[i], orient = "h", ax = axes[position,2])
             position += 1
```

```
In [34]: #correlation matrix
         plt.figure(figsize=(20, 20))
         sb.heatmap(ETRData.corr(), annot=True);
         plt.title('Correlation Matrix', fontsize=20);
```



Exploratory Analysis

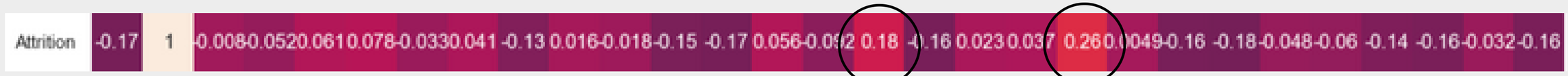
- the departure of employees
- categorical variable (either "Yes" or "No")

Correlation of Attrition with 29 variables.

Exploratory Analysis

Variables with highest positive correlation to Attrition

Over Time



Correlation of Attrition with 29 variables.

Marital Status

Exploratory Analysis

Variables with highest negative correlation to Attrition

Job Level

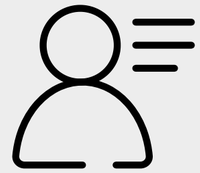
Correlation of Attrition with 29 variables.

Total Working Years

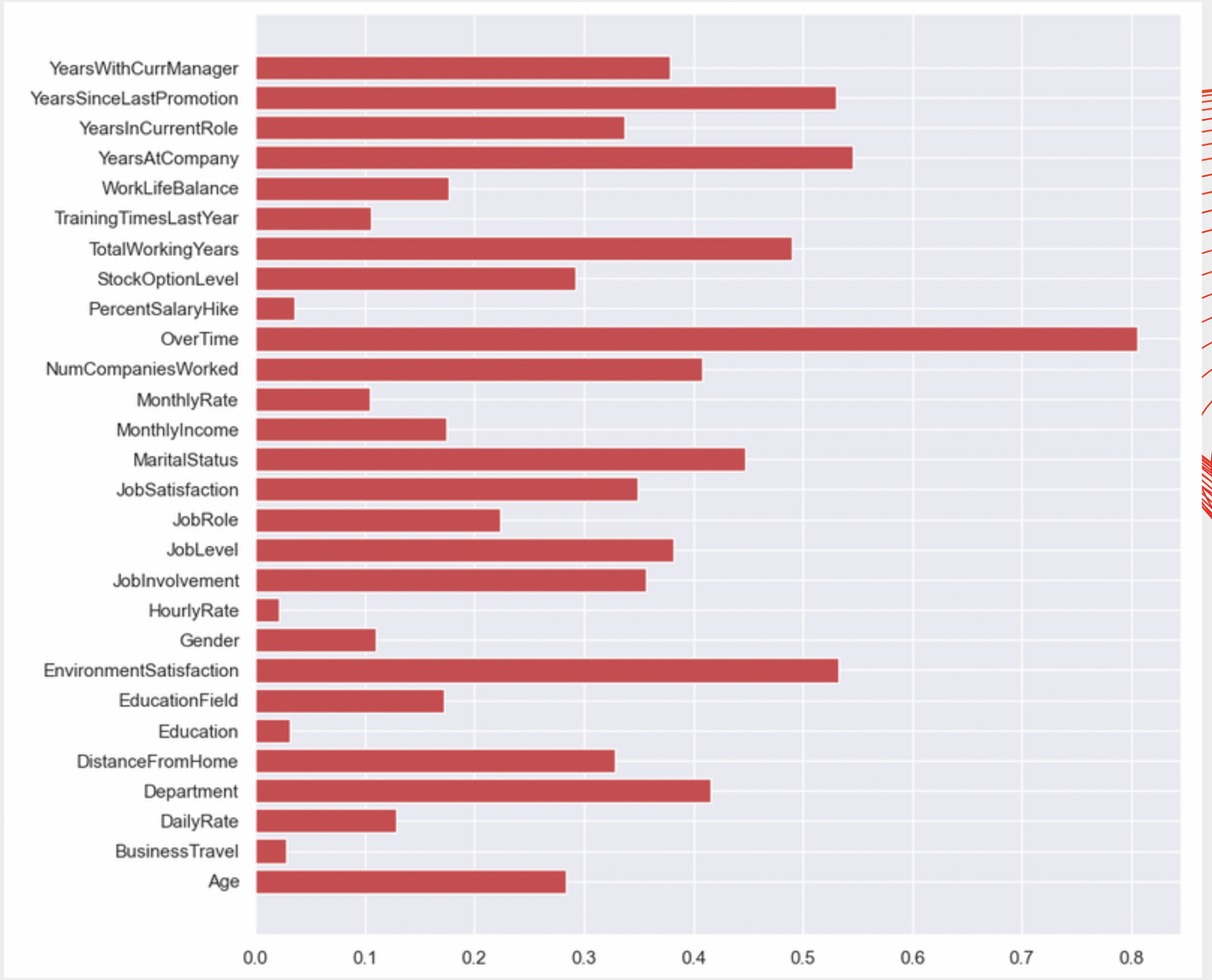
Machine Learning

Logistic Regression Model

In [17]:
```
ETRcopy = ETRData.copy()
ETRcopy.drop(['Attrition'], axis = 1, inplace = True)
X = ETRcopy
y = ETRData['Attrition'].values.ravel()

# Split train and test set into 80:20 ratio
X_train,X_test,y_train,y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
sc = StandardScaler()
sc.fit(X_train)
LRM = LogisticRegression()
LRM.fit(sc.transform(X_train), y_train)

print('Logistic Regression Model:')
print('Training Model accuracy: {:.10f}'.format(LRM.score(X_train,y_train)))
print('Test Model accruracy: {:.10f}'.format(LRM.score(X_test,y_test)))

#finding most effective variables that affect turnover rate
Effectiveness = LRM.coef_.flatten()
plt.rcParams['figure.figsize'] = (10, 10)
plt.barh(ETRcopy.columns, abs(Effectiveness), color = 'r')
plt.show()
```

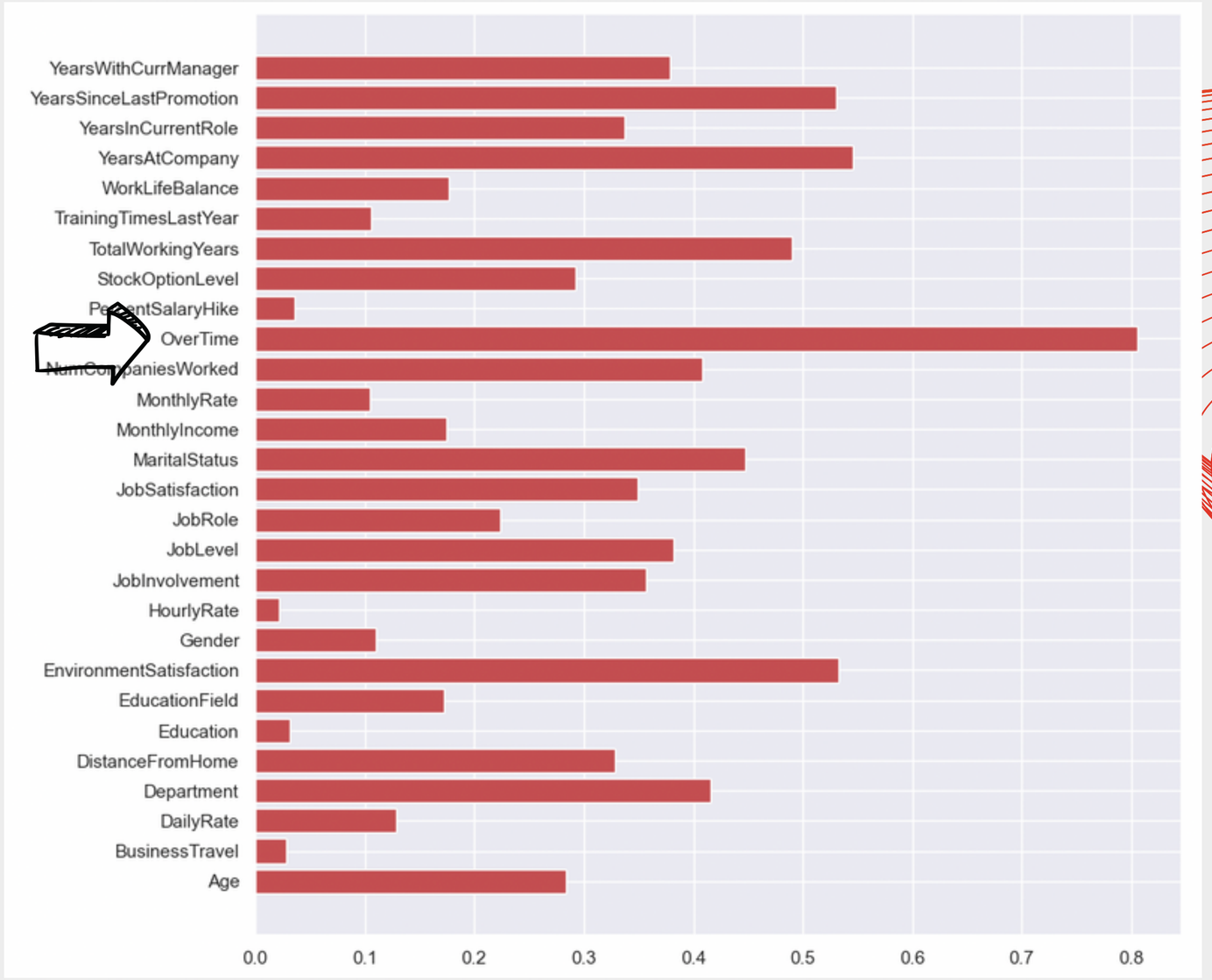


Logistic Regression Model



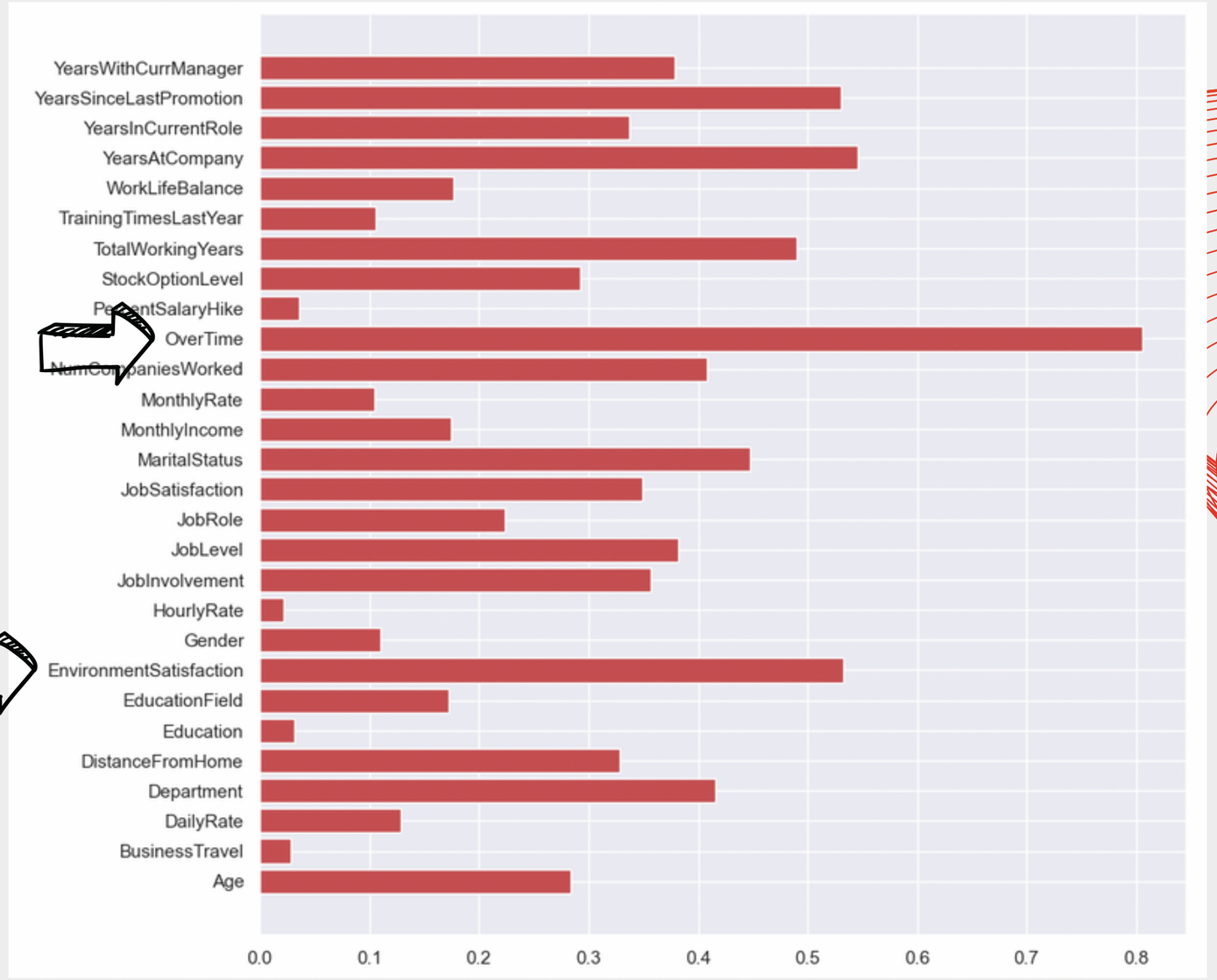

```
Logistic Regression Model:
Training Model accuracy: 0.4550827423
Test Model accruracy: 0.4150943396
```

Logistic Regression Model



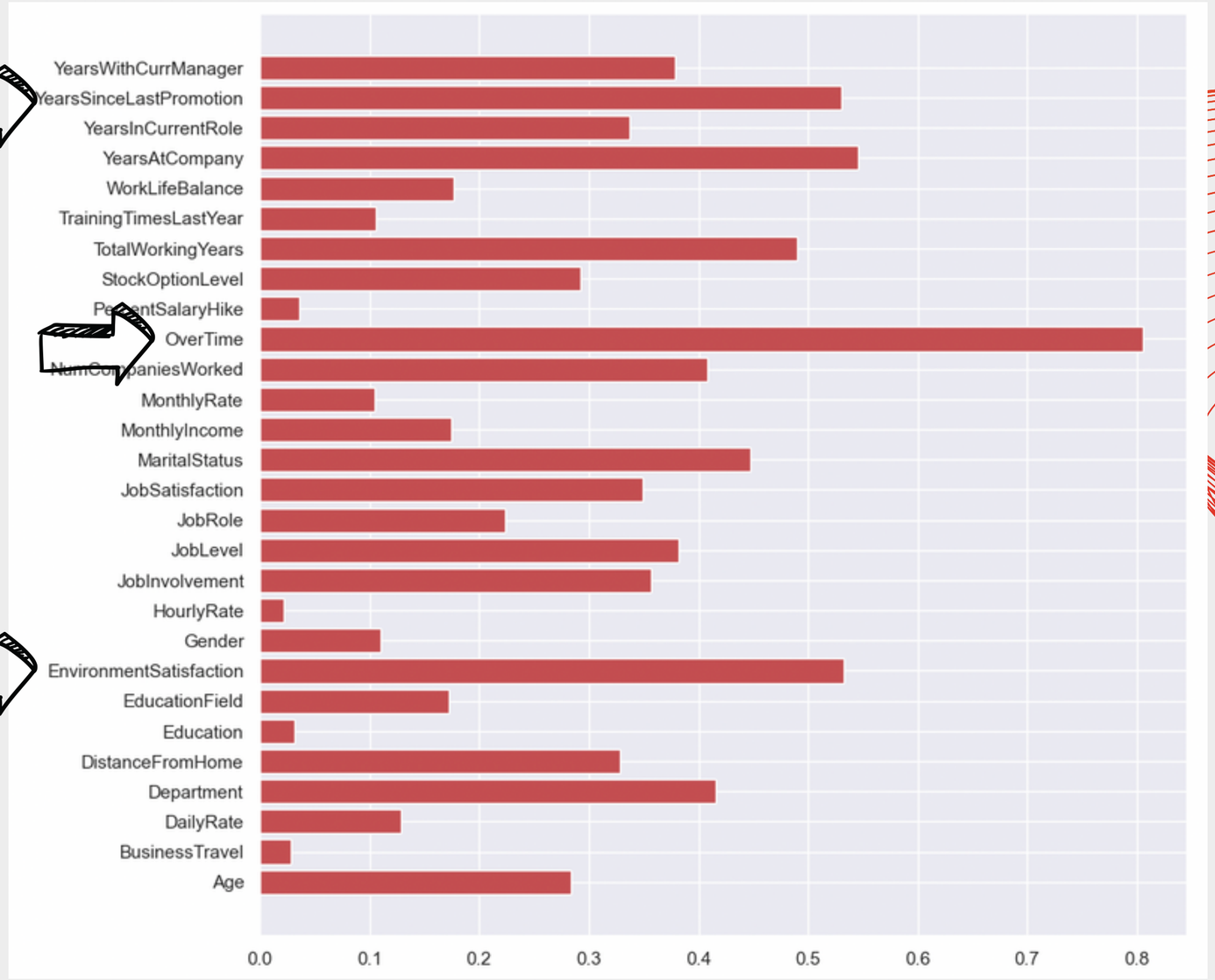

```
Logistic Regression Model:
Training Model accuracy: 0.4550827423
Test Model accruracy: 0.4150943396
```

Logistic Regression Model



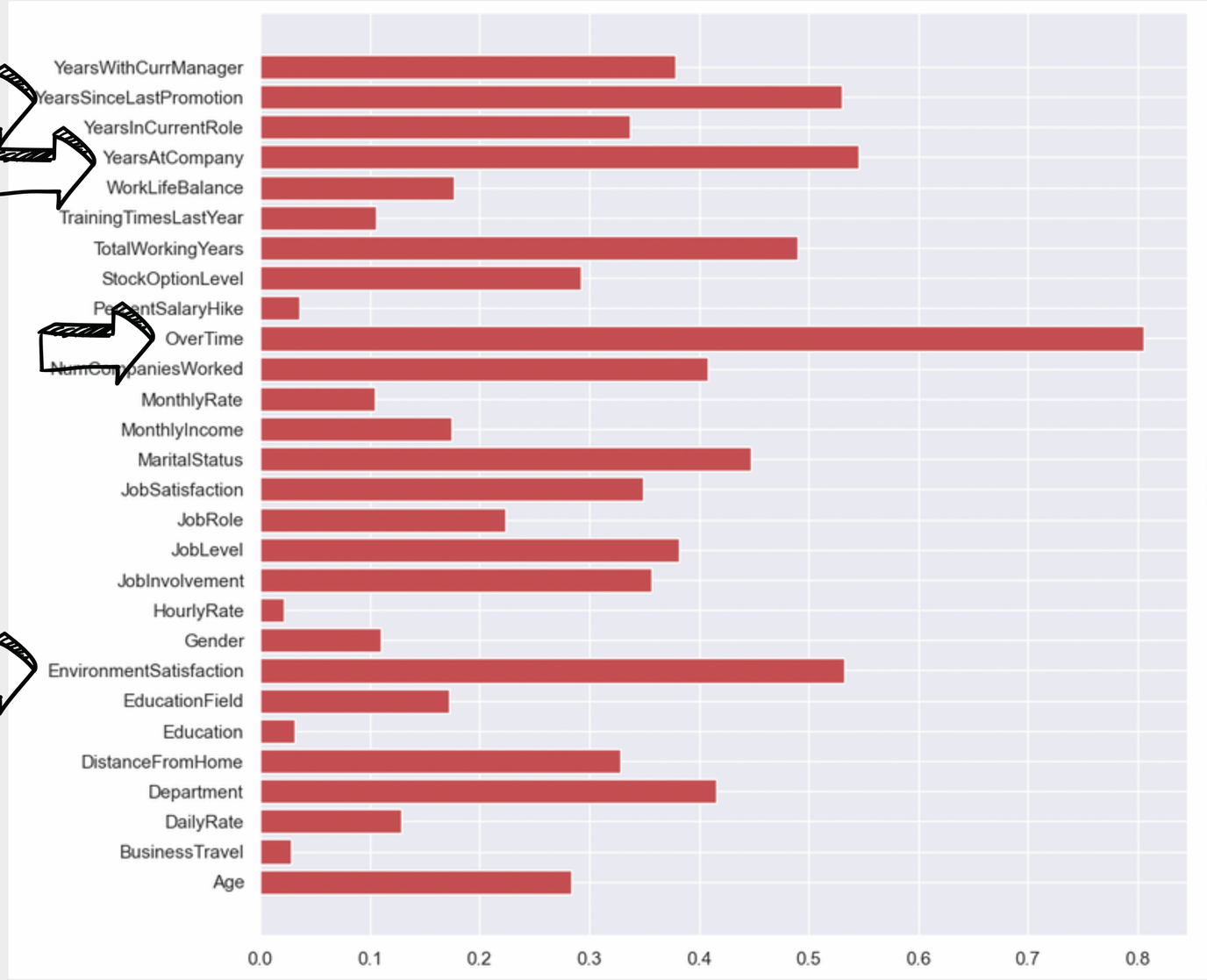

```
Logistic Regression Model:
Training Model accuracy: 0.4550827423
Test Model accruracy: 0.4150943396
```

Logistic Regression Model



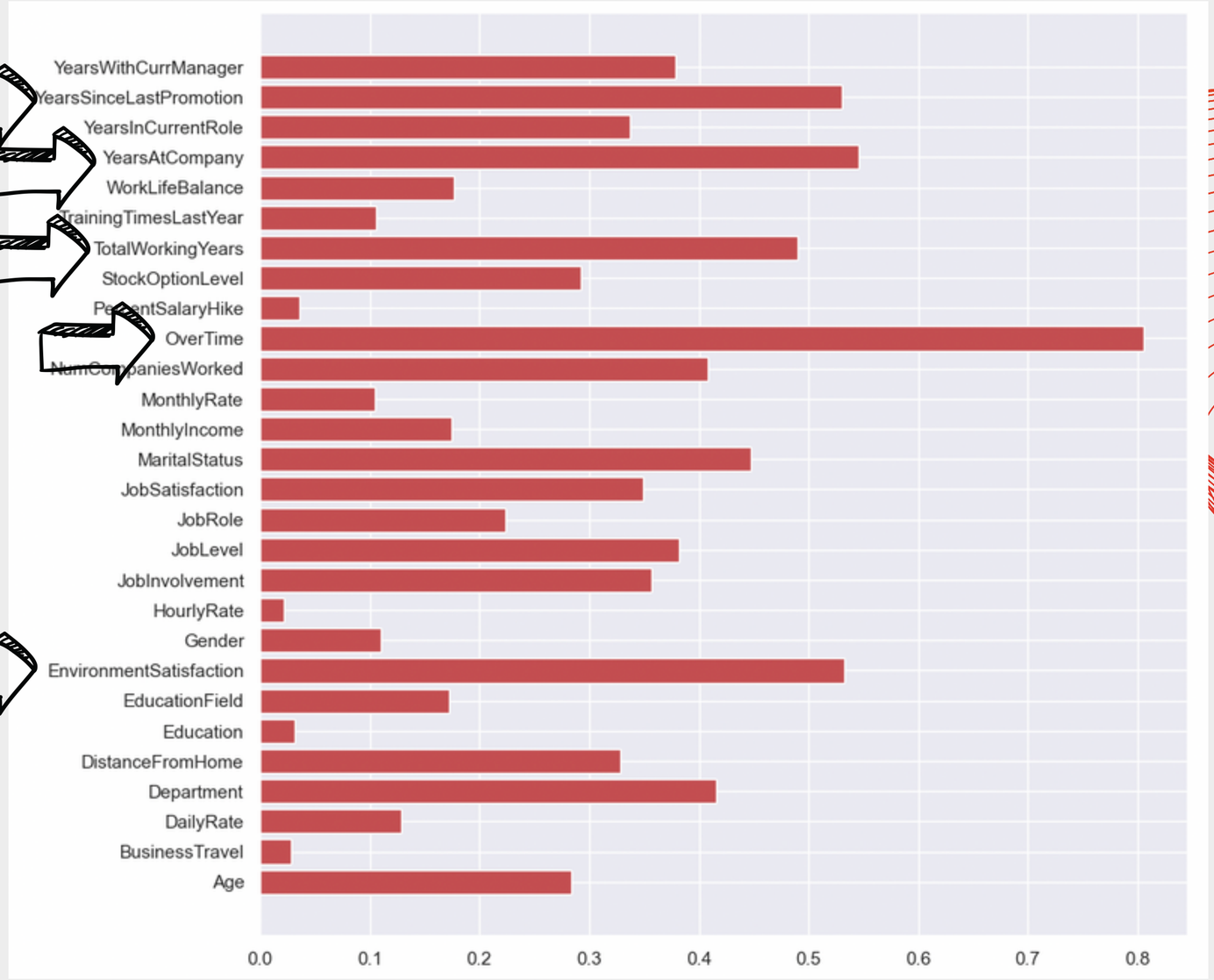

```
Logistic Regression Model:
Training Model accuracy: 0.4550827423
Test Model accruracy: 0.4150943396
```

Logistic Regression Model




```
Logistic Regression Model:
Training Model accuracy: 0.4550827423
Test Model accruracy: 0.4150943396
```

Logistic Regression Model




```
Logistic Regression Model:
Training Model accuracy: 0.4550827423
Test Model accruracy: 0.4150943396
```

Random Forest Model

In [25]:

```
accuracy_list = []
f1_list = []
roc_auc_list = []

def result(X, y, ts, rs, model):

    #train test split
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=ts, random_state=rs)

    #scaling
    sc = StandardScaler()
    X_train = sc.fit_transform(X_train)
    X_test = sc.transform(X_test)

    #fit on data
    model.fit(X_train, y_train)

    #prediction
    pred = model.predict(X_test)

    #performance of model
    print("Classification Report: \n", classification_report(y_test, pred))
    print("-" * 100)
    print()

    #accuracy of model
    acc = accuracy_score(y_test, pred)
    accuracy_list.append(acc)
    print("Accuracy Score: ", acc)
    print("-" * 100)
    print()

    #confusion matrix for model
    print("Confusion Matrix: ")
    plt.figure(figsize=(10, 5))
    sb.heatmap(confusion_matrix(y_test, pred), annot=True, fmt='g');
    plt.title('Confusion Matrix', fontsize=20)


rf = RandomForestClassifier()
x = ETRcopy
y = ETRData['Attrition'].values.ravel()
result(x, y, 0.25, 42, rf)
```

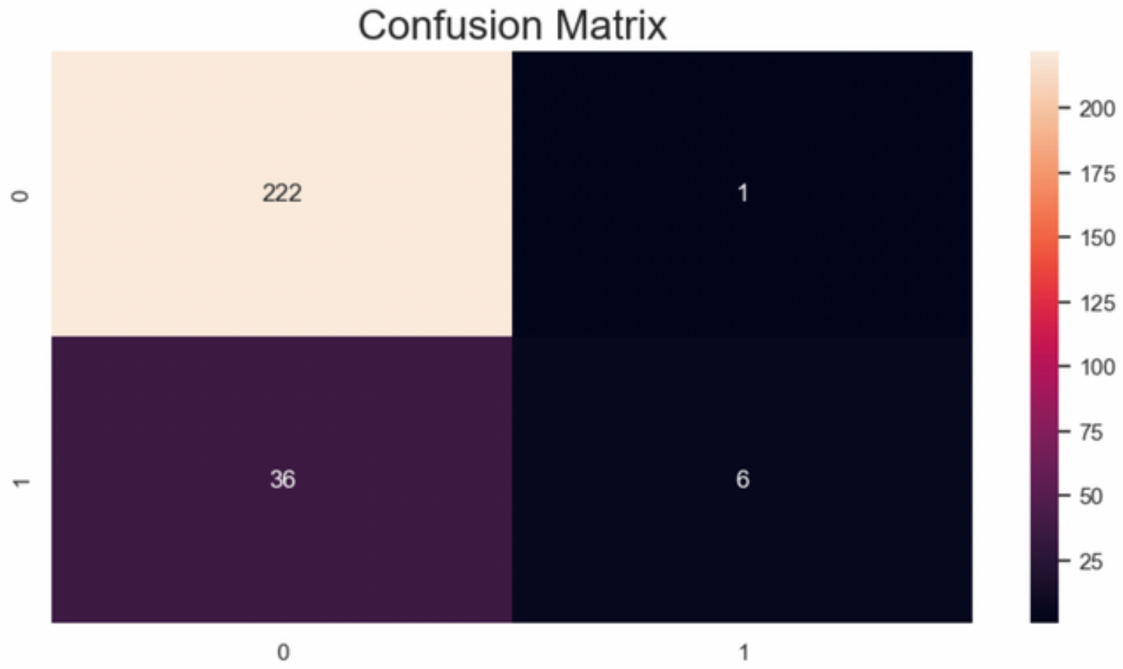
Random Forest Model

Classification Report:

	precision	recall	f1-score	support
0	0.86	1.00	0.92	223
1	0.86	0.14	0.24	42
accuracy			0.86	265
macro avg	0.86	0.57	0.58	265
weighted avg	0.86	0.86	0.82	265

Accuracy Score: 0.8603773584905661

Confusion Matrix:



- model could correctly predict employees unlikely to quit and chose to not quit. [True Negatives]
- TNR = [222 / (222+1)] * 100% = 99.55%
- failed to predict employees supposed to quit, but chose not to quit. [True Positives]
- TPR = [6 / (38+6)] * 100% = 13.73%

Conclusion

Knowing the voluntary employee turnover rate could be very helpful. Should it be higher than expected, companies and employers can quickly implement retention strategies and initiatives to get prevent their turnover rates from becmoing a reality.

Conclusion

Machine Learning Models:

1 . Logistic Regression

✓ Effective in prediction

✖ Accuracy

2. Random Forest

✓ More effective and accurate

Thank You