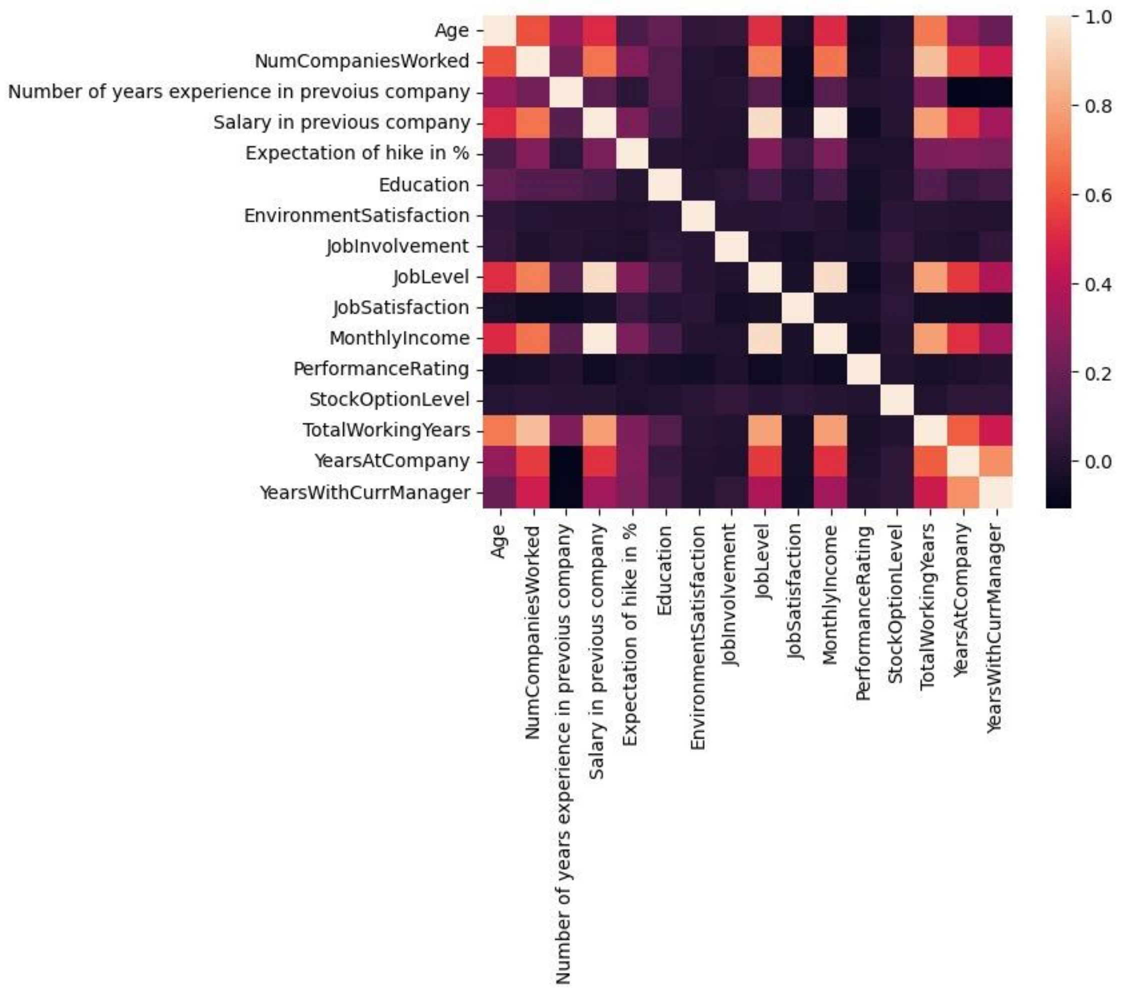
Employee Attrition Prediction Execution Results

Correlation between the columns

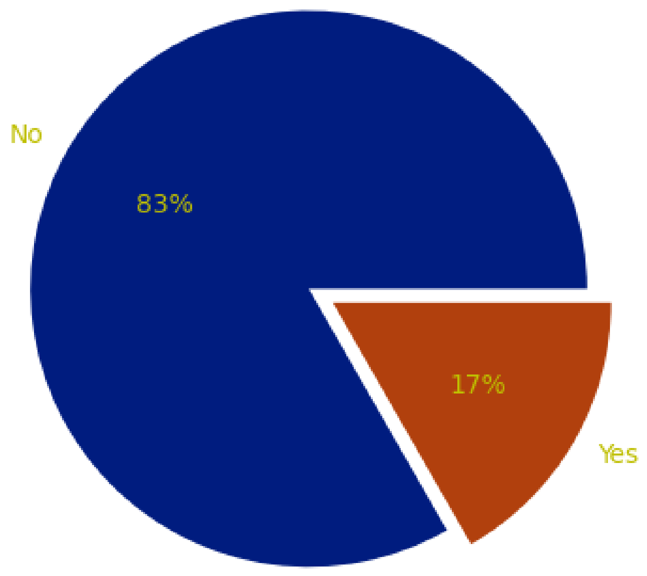
```
#finding the correlation between the columns
corr=numeric_df.corr()
```

```
sns.heatmap(corr)
plt.show()
```



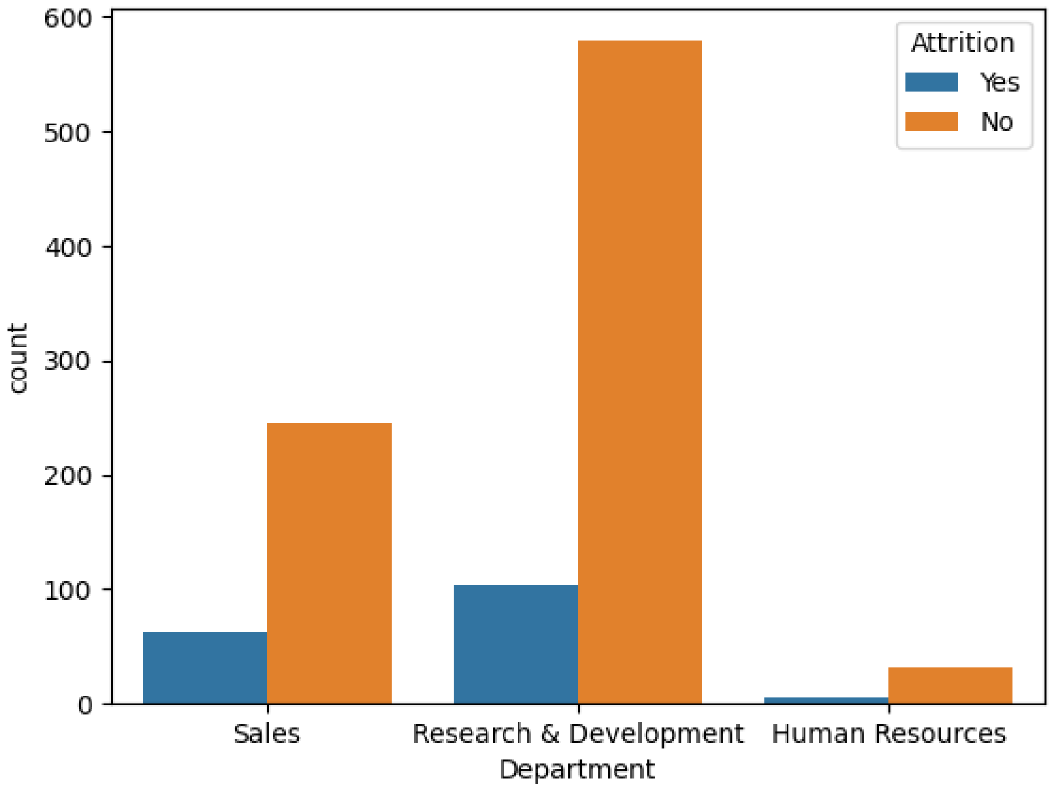
Distribution of targets

```
#distribution of targets

# declaring data
data = train_data['Attrition'].value_counts()
ratios = train_data['Attrition'].value_counts().tolist()
keys = ['No','Yes']

# declaring exploding pie
explode = [0, 0.1]
# define Seaborn color palette to use
\ = sns.color_palette('dark')

# plotting data on chart
plt.pie(ratios, labels=keys, colors=palette_color,
        explode=explode, autopct='%.0f%%',textprops={'color':"y"})

# displaying chart
plt.savefig('/content/drive/MyDrive/Results/targets')

plt.show()
```

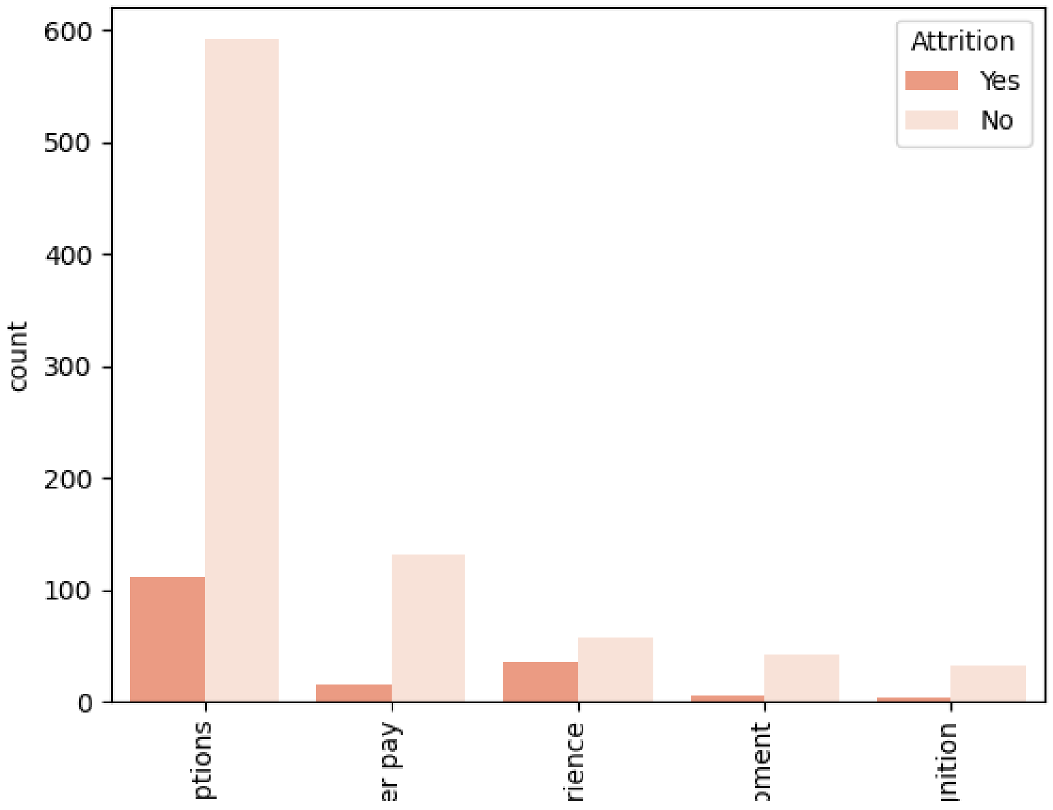


Attrition department wise

```
#attrition departmentwise
sns.countplot(data=train_data, x="Department", hue="Attrition")
plt.savefig('/content/drive/MyDrive/Results/department')

plt.show()
```

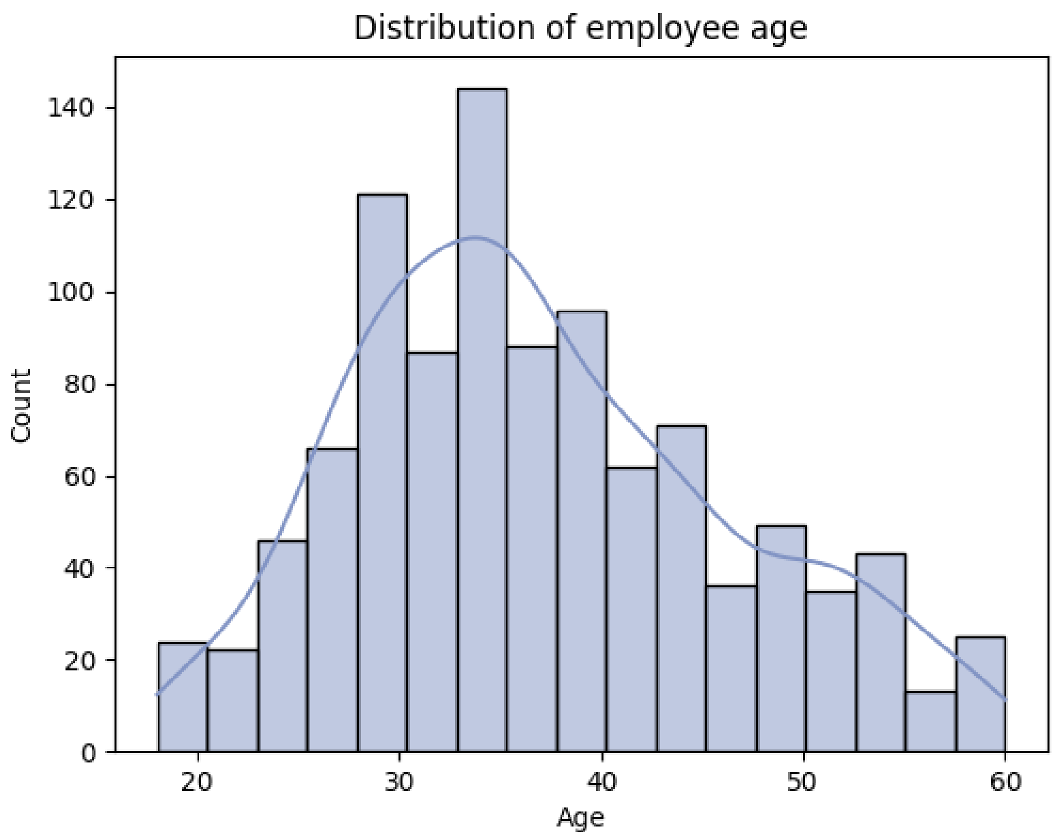


```
sns.dark_palette("#79C")
sns.countplot(data=train_data, x="Reason for leaving", hue="Attrition",palette=["#fc9272","#fee0d2"])
plt.xticks(rotation=90)
plt.savefig('/content/drive/MyDrive/Results/department')

plt.show()
```

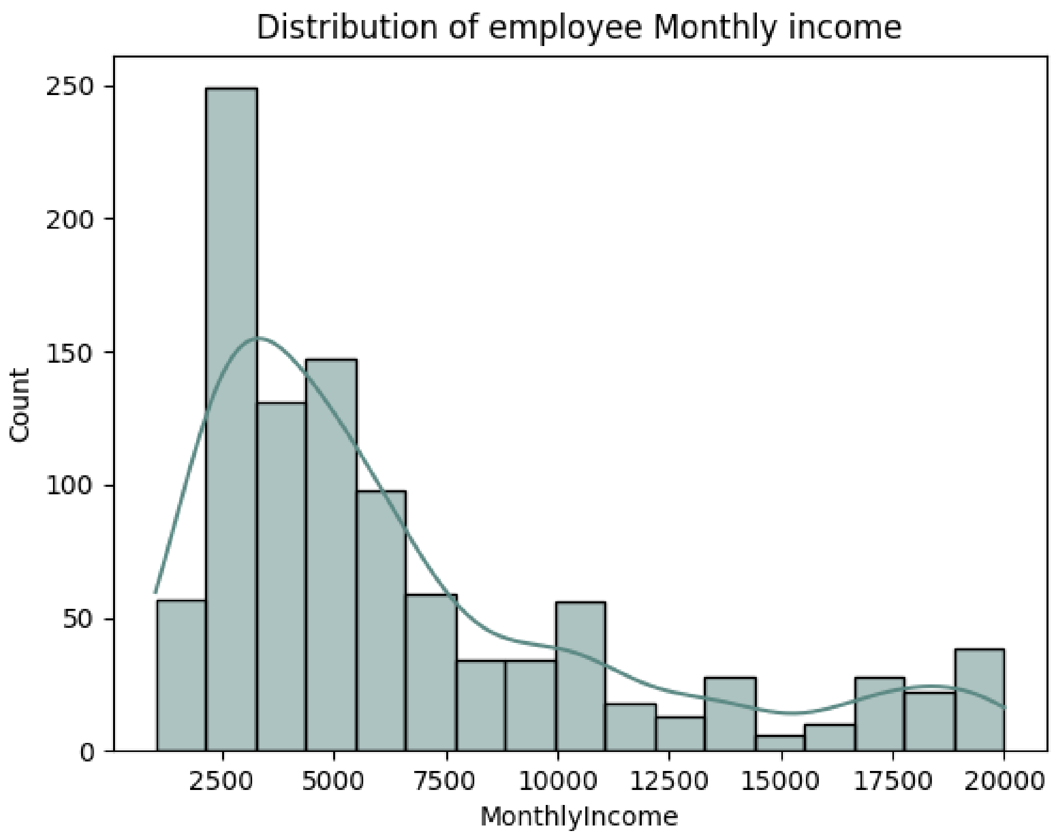


Distribution of age

```
#distribution of age
sns.dark_palette("seagreen")
sns.histplot(x=train_data['Age'],color='#8294C4',kde=True)
plt.title('Distribution of employee age')
plt.savefig('/content/drive/MyDrive/Results/age')
```

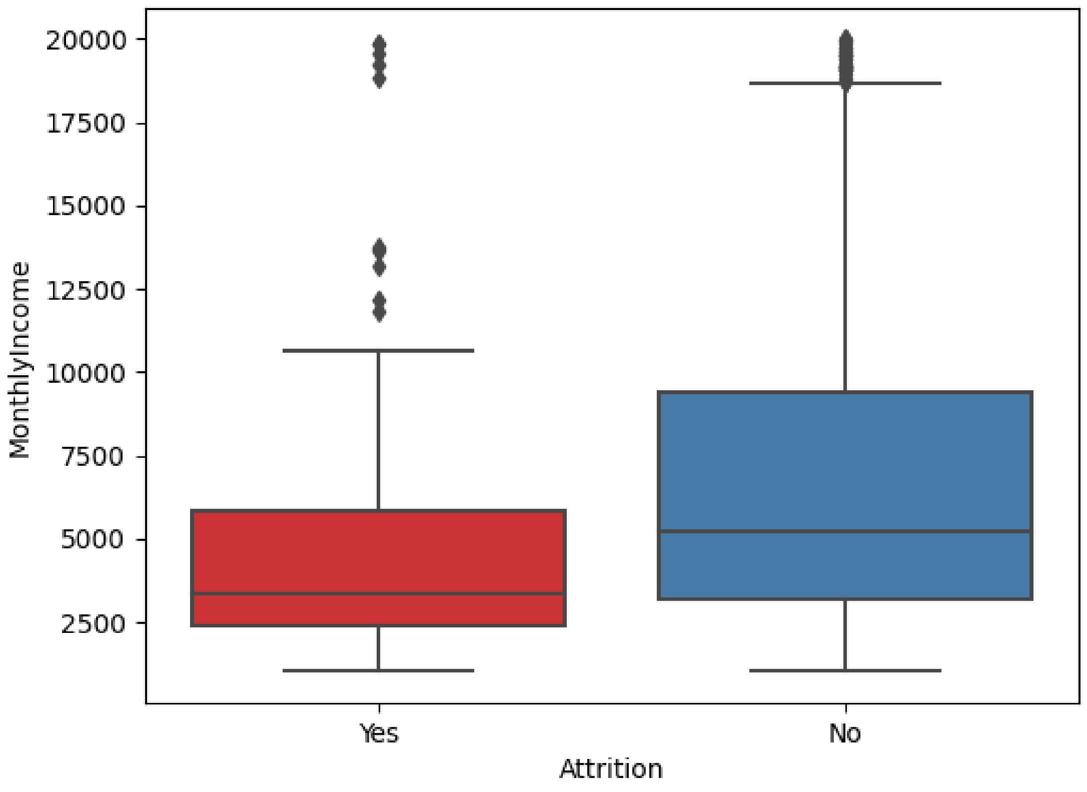


```
#distribution of age
sns.dark_palette("seagreen")
sns.histplot(x=train_data['MonthlyIncome'],color='#5C8984',kde=True)
plt.title('Distribution of employee Monthly income')
plt.savefig('/content/drive/MyDrive/Results/age')
plt.show()
```



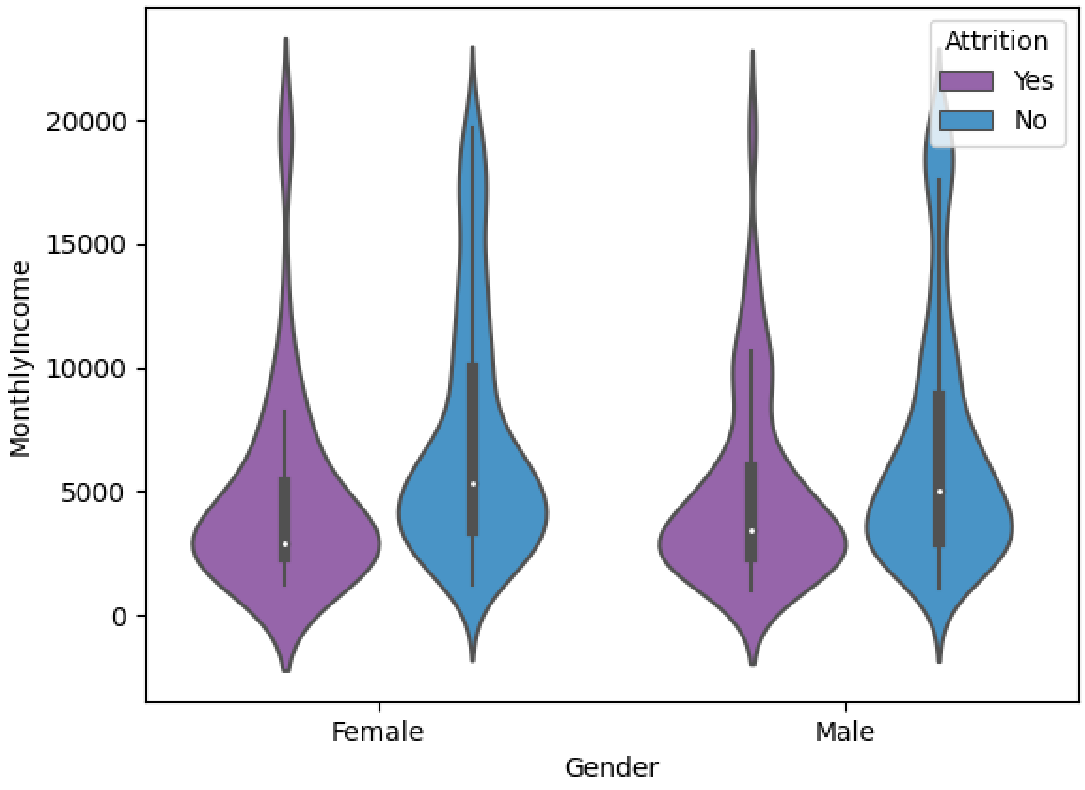
Attrition Monthly Income

```
sns.boxplot(x=train_data['Attrition'],y=train_data['MonthlyIncome'],palette="Set1")
plt.savefig('/content/drive/MyDrive/Results/income')
plt.show()
```



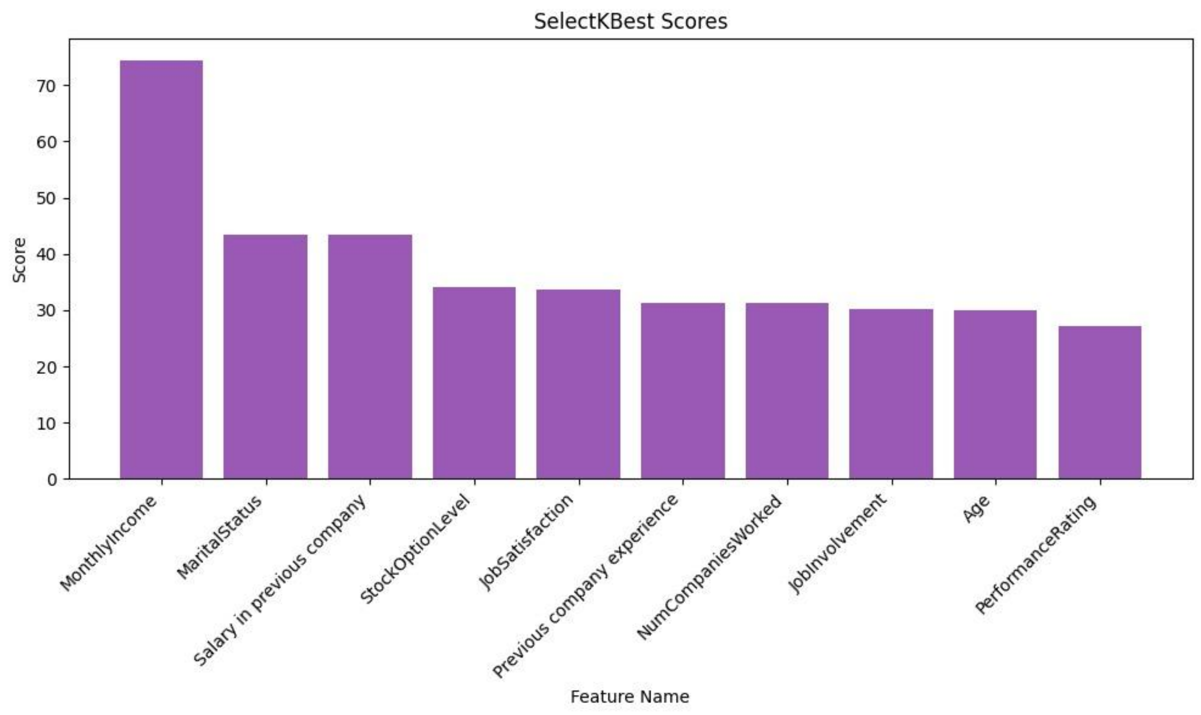
Attrition with Gender and Monthly Income

```
my_colors = ["#9b59b6", "#3498db",
             "#2ecc71", "#006a4e"]
sns.set_palette( my_colors )
sns.violinplot(x=train_data['Gender'],y=train_data['MonthlyIncome'],hue=train_data['Attrition'])
plt.savefig('/content/drive/MyDrive/Results/Business_travel')
plt.show()
```



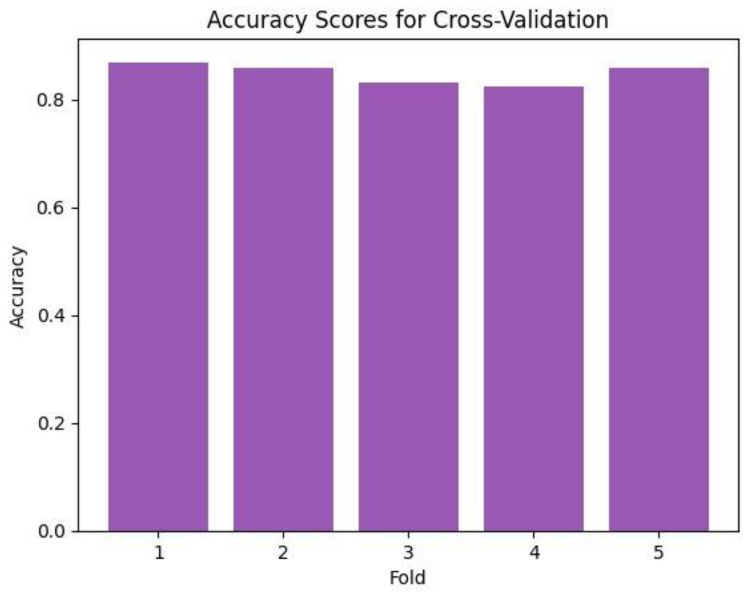
Plot of Feature Name

```
# Plot the scores
plt.figure(figsize=(10, 6))
plt.bar(range(k), sorted_scores[:k])
plt.xticks(range(k), top_feature_names, rotation=45, ha='right')
plt.xlabel('Feature Name')
plt.ylabel('Score')
plt.title('SelectKBest Scores')
plt.tight_layout()
plt.show()
```



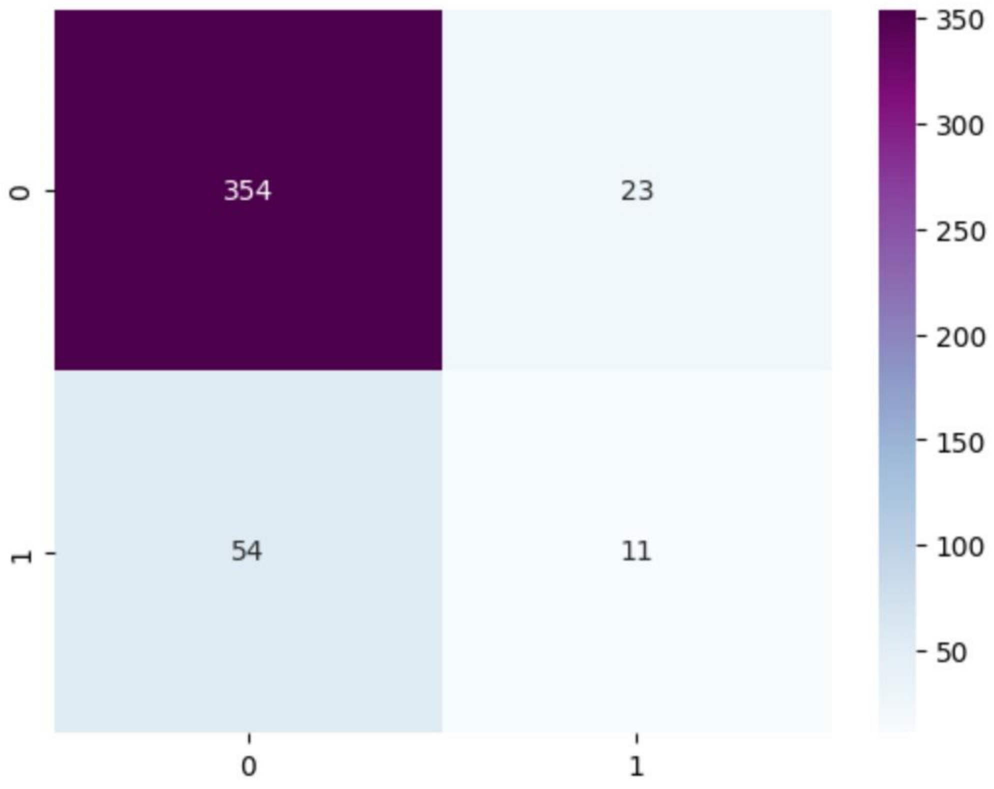
Plot of accuracy scores

```python
folds = list(range(1, len(scores) + 1))

# Plot the accuracy scores
plt.bar(folds, scores)
plt.xlabel('Fold')
plt.ylabel('Accuracy')
plt.title('Accuracy Scores for Cross-Validation')
plt.show()
```



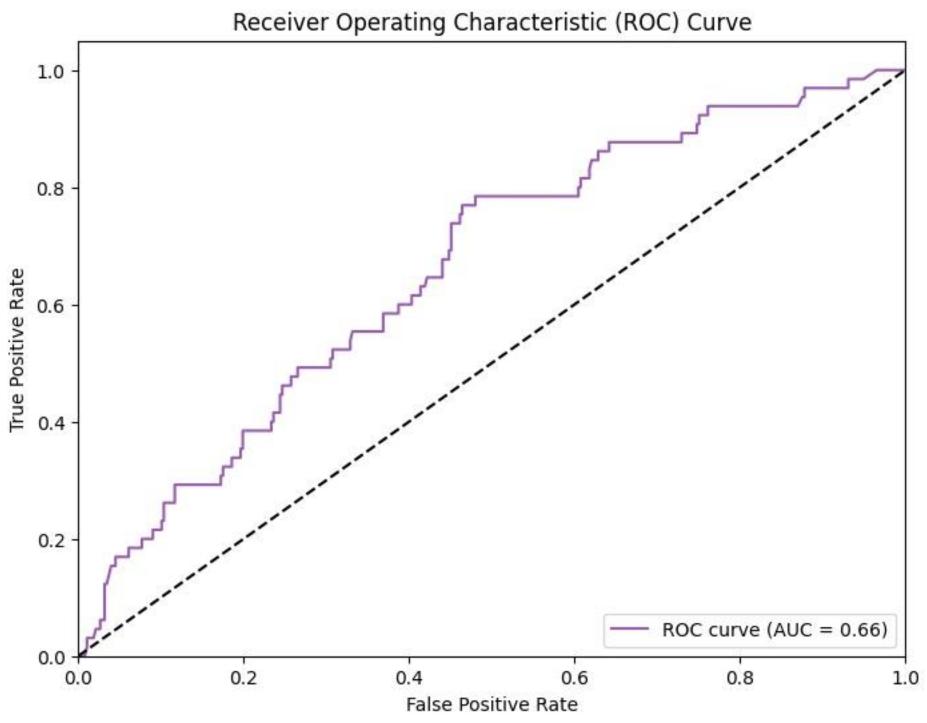
Confusion Matrix

```
cm = confusion_matrix(y_test,ada_preds)
sns.heatmap(cm,annot=True,fmt='g',cmap="BuPu")
plt.show()
```



ROC – Curve

```
y_pred_proba = ada.predict_proba(X_test)[:, 1]

# Calculate the false positive rate, true positive rate, and threshold values
fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)

# Calculate the AUC
auc = roc_auc_score(y_test, y_pred_proba)

# Plot the ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label='ROC curve (AUC = {:.2f})'.format(auc))
plt.plot([0, 1], [0, 1], 'k--')  # Plot the random guessing curve
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc='lower right')
plt.show()
```


Receiver Operating Characteristic (ROC) Curve
True Positive Rate
False Positive Rate
ROC curve (AUC = 0.66)