HR ATTRIBUTION

In [ ]:
```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer, f1_score
import numpy as np
from sklearn.metrics import confusion_matrix, roc_curve, roc_auc_score, auc
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import numpy as np
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer, roc_auc_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import accuracy_score
```

In [ ]:
```
import warnings
warnings.simplefilter(action="ignore", category=FutureWarning)
```

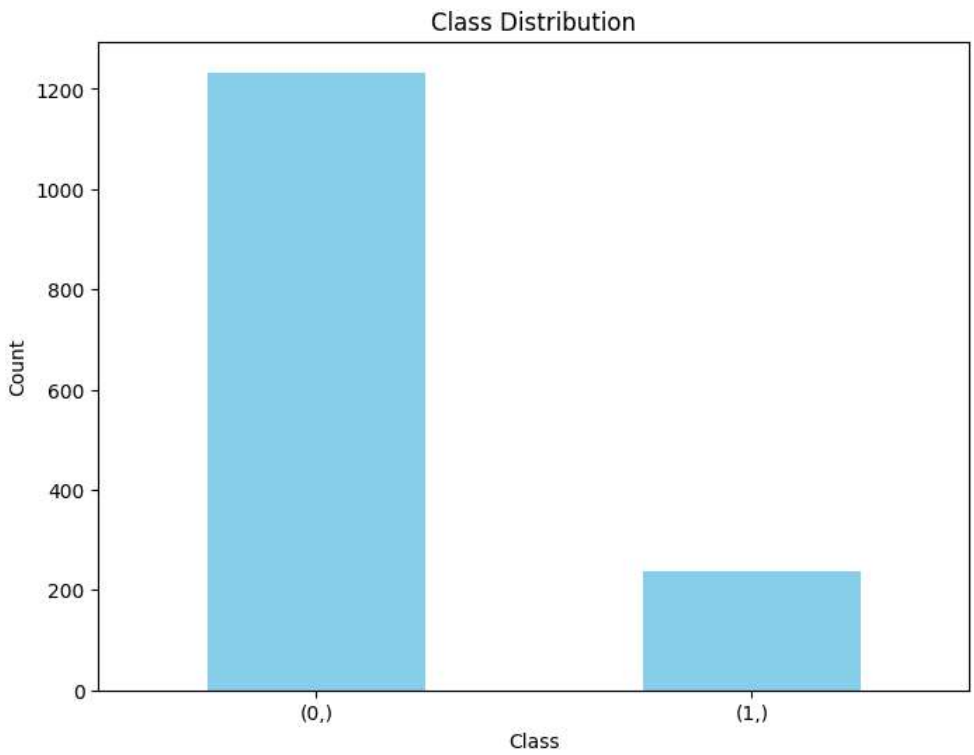
1.) Import, split data into X/y, plot y data as bar charts, turn X categorical variables binary and tts.

In [ ]:
```
df = pd.read_csv("HR_Analytics.csv")
```

In [ ]:
```
y = df[["Attrition"]].copy()
X = df.drop("Attrition", axis = 1)
```

In [ ]:
```
y["Attrition"] = [1 if i == "Yes" else 0 for i in y["Attrition"]]
```

In [ ]:
```
class_counts = y.value_counts()


plt.figure(figsize=(8, 6))
class_counts.plot(kind='bar', color='skyblue')
plt.xlabel('Class')
plt.ylabel('Count')
plt.title('Class Distribution')
plt.xticks(rotation=0)  # Remove rotation of x-axis labels
plt.show()
```



In [ ]:
```
# Step 1: Identify string columns
string_columns = X.columns[X.dtypes == 'object']

# Step 2: Convert string columns to categorical
for col in string_columns:
    X[col] = pd.Categorical(X[col])
```

```
# Step 3: Create dummy columns
X = pd.get_dummies(X, columns=string_columns, prefix=string_columns,drop_first=True)
```

In [ ]:
```
x_train,x_test,y_train,y_test=train_test_split(X,
 y, test_size=0.20, random_state=42)
```

2.) Using the default Decision Tree. What is the IN/Out of Sample accuracy?

In [ ]:
```
clf = DecisionTreeClassifier()
clf.fit(x_train,y_train)
y_pred=clf.predict(x_train)
acc=accuracy_score(y_train,y_pred)
print("IN SAMPLE ACCURACY : " , round(acc,2))

y_pred=clf.predict(x_test)
acc=accuracy_score(y_test,y_pred)
print("OUT OF SAMPLE ACCURACY : " , round(acc,2))
```

```
IN SAMPLE ACCURACY :  1.0
OUT OF SAMPLE ACCURACY :  0.78
```

3.) Run a grid search cross validation using F1 score to find the best metrics. What is the In and Out of Sample now?

In [ ]:
```
# Define the hyperparameter grid to search through
param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': np.arange(1, 11),  # Range of max_depth values to try
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}


dt_classifier = DecisionTreeClassifier(random_state=42)

scoring = make_scorer(f1_score, average='weighted')

grid_search = GridSearchCV(estimator=dt_classifier, param_grid=param_grid, scoring=scoring, cv=5)

grid_search.fit(x_train, y_train)

# Get the best parameters and the best score
best_params = grid_search.best_params_
best_score = grid_search.best_score_

print("Best Parameters:", best_params)
print("Best F1-Score:", best_score)
```

```
Best Parameters: {'criterion': 'gini', 'max_depth': 6, 'min_samples_leaf': 2, 'min_samples_split': 2}
Best F1-Score: 0.8214764475510983
```

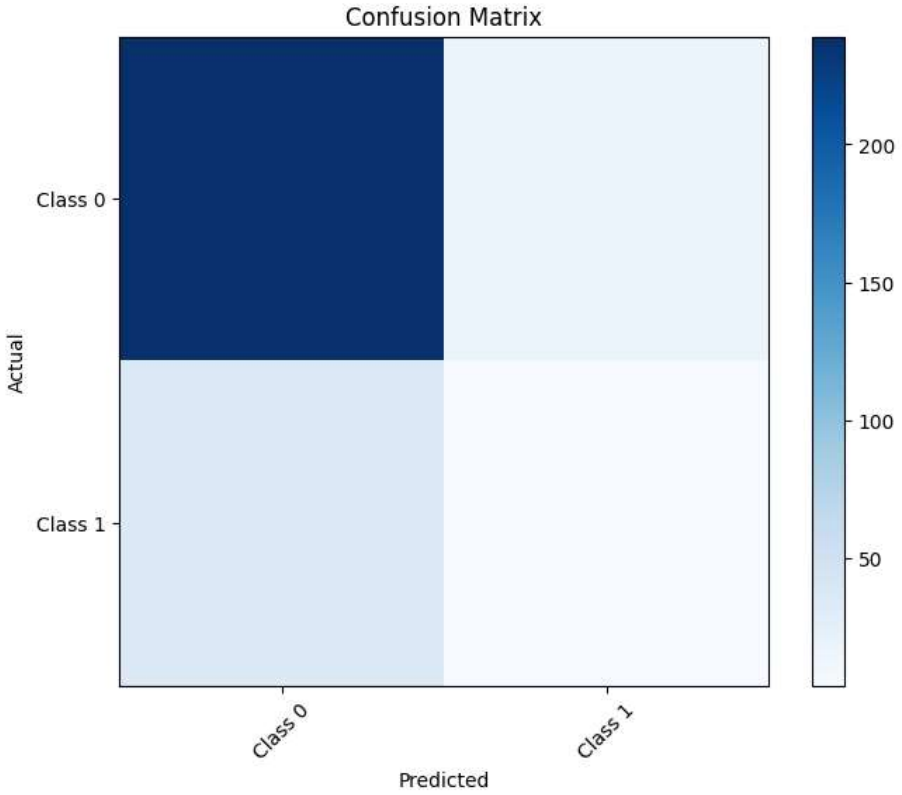
In [ ]:
```
clf = tree.DecisionTreeClassifier(**best_params, random_state =42)
clf.fit(x_train,y_train)
y_pred=clf.predict(x_train)
acc=accuracy_score(y_train,y_pred)
print("IN SAMPLE ACCURACY : " , round(acc,2))

y_pred=clf.predict(x_test)
acc=accuracy_score(y_test,y_pred)
print("OUT OF SAMPLE ACCURACY : " , round(acc,2))
```

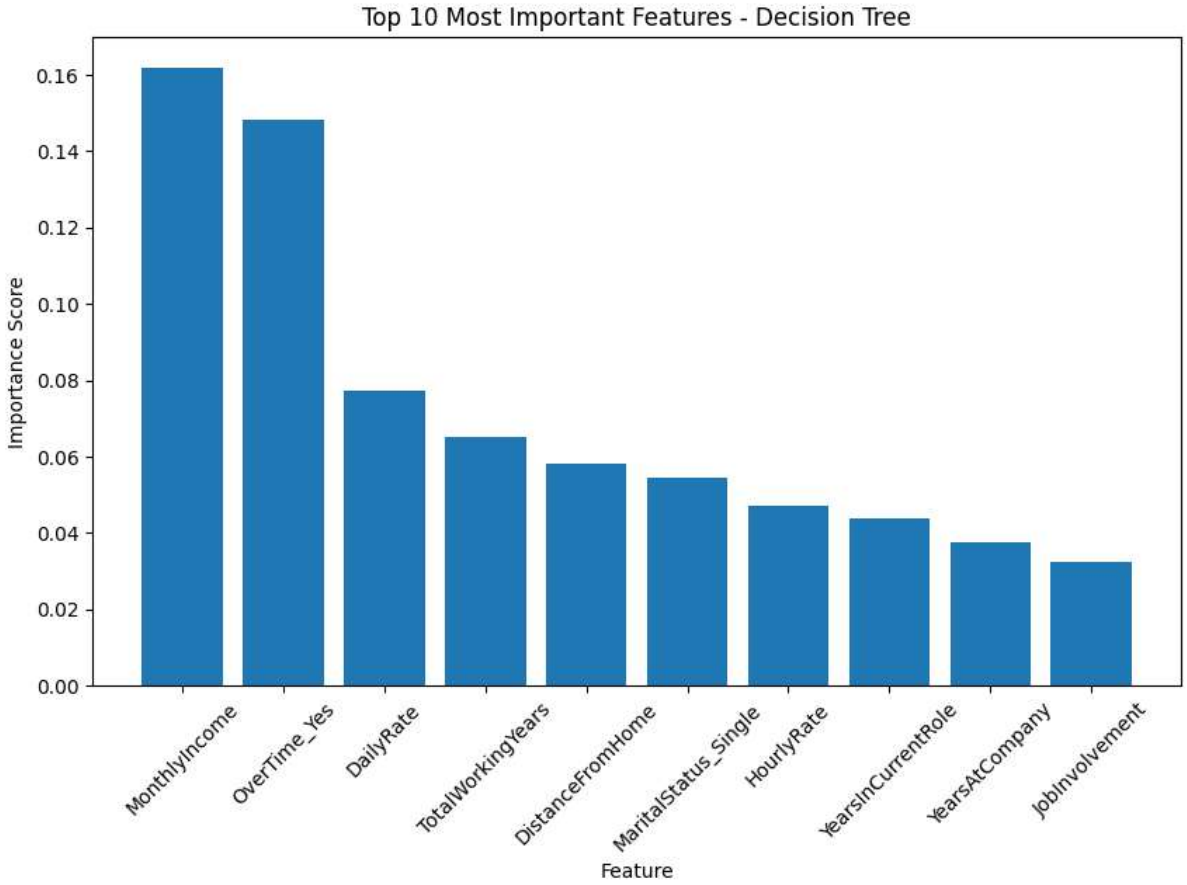
```
IN SAMPLE ACCURACY :  0.91
OUT OF SAMPLE ACCURACY :  0.83
```

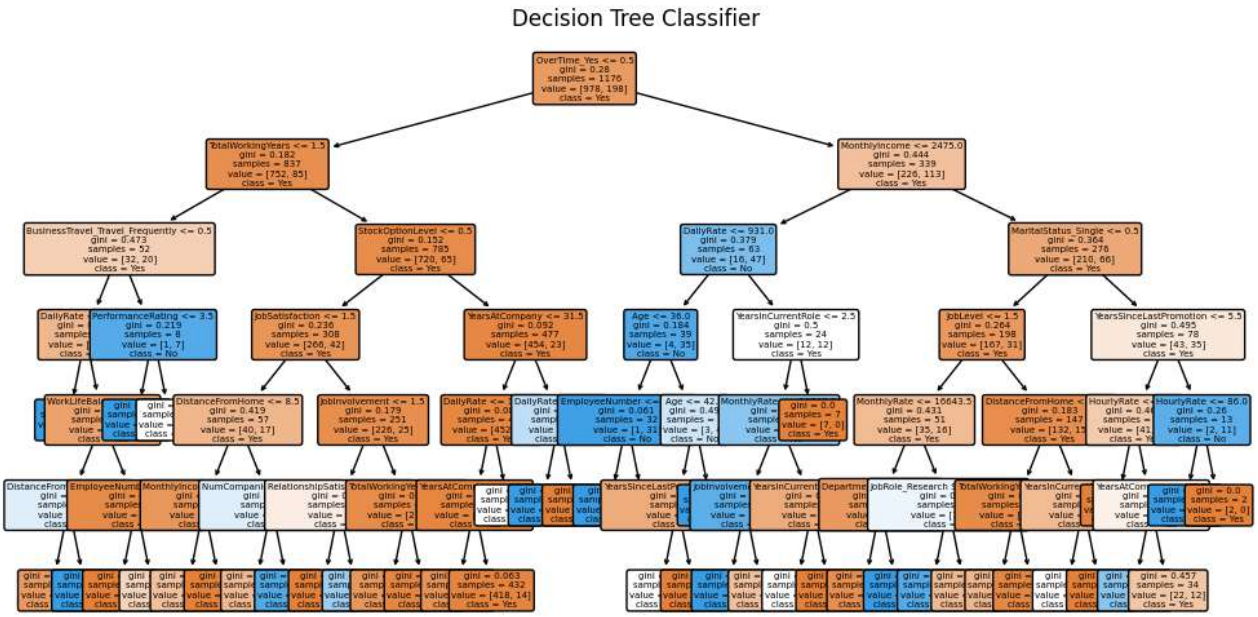
4.) Plot .....

In [ ]:
```
# Make predictions on the test data
y_pred = clf.predict(x_test)
y_prob = clf.predict_proba(x_test)[:, 1]

# Calculate the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix
plt.figure(figsize=(8, 6))
plt.imshow(conf_matrix, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
tick_marks = np.arange(len(conf_matrix))
plt.xticks(tick_marks, ['Class 0', 'Class 1'], rotation=45)
plt.yticks(tick_marks, ['Class 0', 'Class 1'])
```

```
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()




feature_importance = clf.feature_importances_

# Sort features by importance and select the top 10
top_n = 10
top_feature_indices = np.argsort(feature_importance)[::-1][:top_n]
top_feature_names = X.columns[top_feature_indices]
top_feature_importance = feature_importance[top_feature_indices]

# Plot the top 10 most important features
plt.figure(figsize=(10, 6))
plt.bar(top_feature_names, top_feature_importance)
plt.xlabel('Feature')
plt.ylabel('Importance Score')
plt.title('Top 10 Most Important Features - Decision Tree')
plt.xticks(rotation=45)
plt.show()

# Plot the Decision Tree for better visualization of the selected features
plt.figure(figsize=(12, 6))
plot_tree(clf, filled=True, feature_names=X.columns, class_names=["Yes", "No"], rounded=True, fontsize=5)
plt.title('Decision Tree Classifier')
plt.show()
```







5.) Looking at the graphs. what would be your suggestions to try to improve customer retention? What additional information would you need for a better plan. Plot anything you think would assist in your assessment.

ANSWER :

We can reduce attrition by adjusting pay and overtime hours. But the direction of the impact of certain features maybe uncertain. For example, in some cases, people may be more willing to work overtime for double pay. Therefore, we need to further calculate the correlation coefficient to determine the impact.

In [ ]:
```
from scipy.stats import pearsonr
```

In [ ]:
```
np.corrcoef(np.array(X["OverTime_Yes"]), np.array(y["Attrition"]))
```

Out[ ]:
```
array([[1.        , 0.24611799],
       [0.24611799, 1.        ]])
```

Therefore, the correlation between the OverTime_Yes feature and the Attrition target is positive, which means that as the OverTime_Yes feature increases, the Attrition target also increases. This is consistent with the feature importance plot, which shows that the more overtime an employee works, the more likely they are to leave the company.

6.) Using the Training Data, if they made everyone work overtime. What would have been the expected difference in client retention?

In [ ]:
```
x_train_experiment = x_train.copy()
```

In [ ]:
```
x_train_experiment["OverTime_Yes"] = 0
```

In [ ]:
```
y_pred = clf.predict(x_train_experiment)
y_pred_experiment = clf.predict(x_train_experiment)
```

In [ ]:
```
diff = sum(y_pred - y_pred_experiment)
print("Change fro, ....",diff)
```

```
Change fro, .... 59
```

7.) If they company loses an employee, there is a cost to train a new employee for a role ~2.8 * their monthly income.

To make someone not work overtime costs the company 2K per person.

Is it profitable for the company to remove overtime? If so/not by how much?

What do you suggest to maximize company profits?

In [ ]:
```
x_train_experiment["Y"] = y_pred
x_train_experiment["Y_experiment"] = y_pred_experiment
```

In [ ]:
```
x_train_experiment["RetChange"] = x_train_experiment["Y_experiment"] - x_train_experiment["Y"]
```

In [ ]:
```
sav = sum(-2.8 * x_train_experiment["RetChange"] * x_train_experiment["MonthlyIncome"])
```

In [ ]:
```
cost = len(x_train[x_train["OverTime_Yes"] == 1]) * 2000
```

In [ ]:
```
sav - cost
```

```
-117593.99999999977
```

ANSWER :

It's not profitable to cancel overtime. And I will not suggest eliminate overtime from the perspective of maximize profit. But in reality, it is very unwise to make all employees work overtime in order to increase profits.Because forced overtime costs companies more than just the cost of training new employees.

8.) Use your model and get the expected change in retention for raising and lowering peoples income. Plot the outcome of the experiment. Comment on the outcome of the experiment and your suggestions to maximize profit.

In [ ]:
```
raise_amount = 100
```

In [ ]:
```
x_train_experiment = x_train.copy()

x_train_experiment["MonthlyIncome"] = x_train_experiment["MonthlyIncome"]* raise_amount

y_pred = clf.predict(x_train)
y_pred_experiment = clf.predict(x_train_experiment)
```

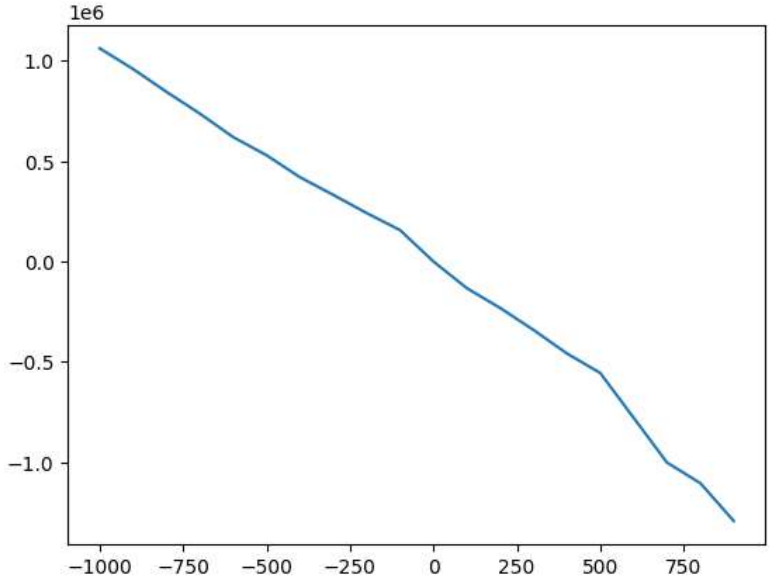
```
diff = sum(y_pred - y_pred_experiment)
print("Change in attrition", diff)
```

```
Change in attrition 22
```

In [ ]:
```
profits = []
for raise_amount in range(-1000,1000,100):
    x_train_experiment = x_train.copy()

    x_train_experiment["MonthlyIncome"] = x_train_experiment["MonthlyIncome"] + raise_amount

    y_pred = clf.predict(x_train)
    y_pred_experiment = clf.predict(x_train_experiment)

    diff = sum(y_pred - y_pred_experiment)
    print("Change in attrition", diff)

    x_train_experiment["Y"] = y_pred
    x_train_experiment["Y_exp"] = y_pred_experiment

    x_train_experiment["RetChange"] = x_train_experiment["Y_exp"]-x_train_experiment["Y"]

    sav = sum(-2.8 * x_train_experiment["RetChange"] * x_train_experiment["MonthlyIncome"])

    cost = len(x_train)*raise_amount

    print("Profit,",sav - cost)
    profits.append(sav- cost)
```

```
Change in attrition -36
Profit, 1060038.0
Change in attrition -32
Profit, 957250.0
Change in attrition -30
Profit, 843780.0
Change in attrition -27
Profit, 735375.2
Change in attrition -25
Profit, 618595.6
Change in attrition -20
Profit, 528206.0
Change in attrition -18
Profit, 419185.2
Change in attrition -13
Profit, 330876.0
Change in attrition -8
Profit, 240693.6
Change in attrition -3
Profit, 155901.2
Change in attrition 0
Profit, 0.0
Change in attrition 6
Profit, -132602.4
Change in attrition 15
Profit, -231392.0
Change in attrition 16
Profit, -340771.2
Change in attrition 22
Profit, -457480.80000000005
Change in attrition 24
Profit, -554859.2
Change in attrition 22
Profit, -777344.4
Change in attrition 20
Profit, -1000319.6000000001
Change in attrition 21
Profit, -1102908.8
Change in attrition 19
Profit, -1290690.8
```

In [ ]:
```
plt.plot(range(-1000,1000,100),profits)
plt.show()
```



ANSWER :

According to the model, in order to maximize profits, we should reduce the salary of all employees by more than $1,000. However, I still think that this is not reasonable in reality, because many negative effects of the pay cut are not estimated by the model, e.g., decrease in employee morale, damage to the corporate image, etc.