HW 4

February 2, 2024

1 HR ATTRIBUTION

[1]:
```
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
```

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

[2]:
```
df = pd.read_csv("HR_Analytics.csv")
df.head()
```

[2]:

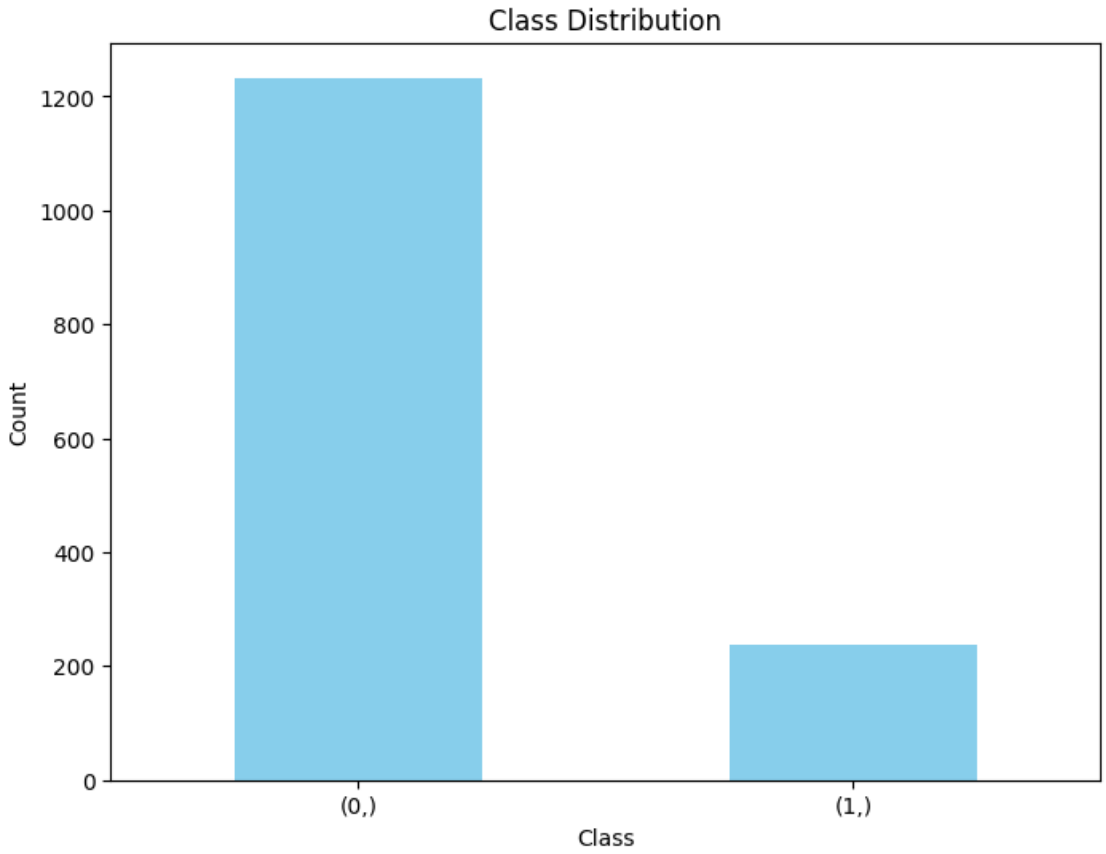
	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5
4	27	No	Travel_Rarely	591	Research & Development					

```
4                  2          1        Medical              1                7

   …  RelationshipSatisfaction StandardHours  StockOptionLevel  \
0  …                         1            80                 0
1  …                         4            80                 1
2  …                         2            80                 0
3  …                         3            80                 0
4  …                         4            80                 1

   TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  YearsAtCompany  \
0                  8                      0               1               6
1                 10                      3               3              10
2                  7                      3               3               0
3                  8                      3               3               8
4                  6                      3               3               2

  YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                  4                        0                     5
1                  7                        1                     7
2                  0                        0                     0
3                  7                        3                     0
4                  2                        2                     2

[5 rows x 35 columns]
```

[3]:
```
y = df[["Attrition"]].copy()
X = df.drop("Attrition", axis = 1)
```

[4]:
```
y["Attrition"] = [1 if i == "Yes" else 0 for i in y["Attrition"]]
```

[5]:
```
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()
```



[6]:
```
# 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)
```

[7]:
```
x_train,x_test,y_train,y_test=train_test_split(X,
 y, test_size=0.20, random_state=42)
```

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

[8]:
```
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 :  0.77
```

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

[9]:
```
# 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
```

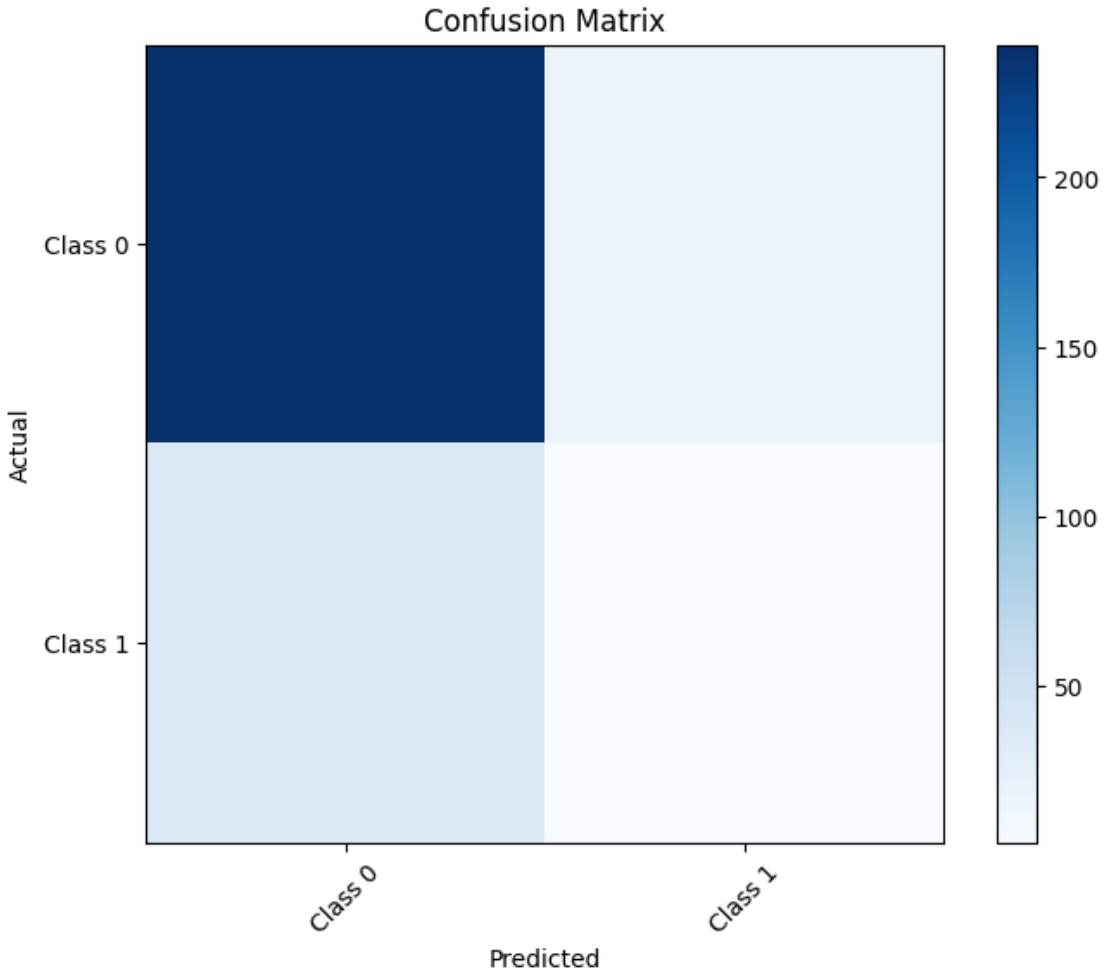
[10]:
```
clf = tree.DecisionTreeClassifier(**best_params, random_state =42)
clf.fit(x_train,y_train)
y_pred=clf.predict(x_train)
acc=accuracy_score(y_train,y_pred)
print("IN SAMPLE ACCURACY : " , round(acc,2))

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

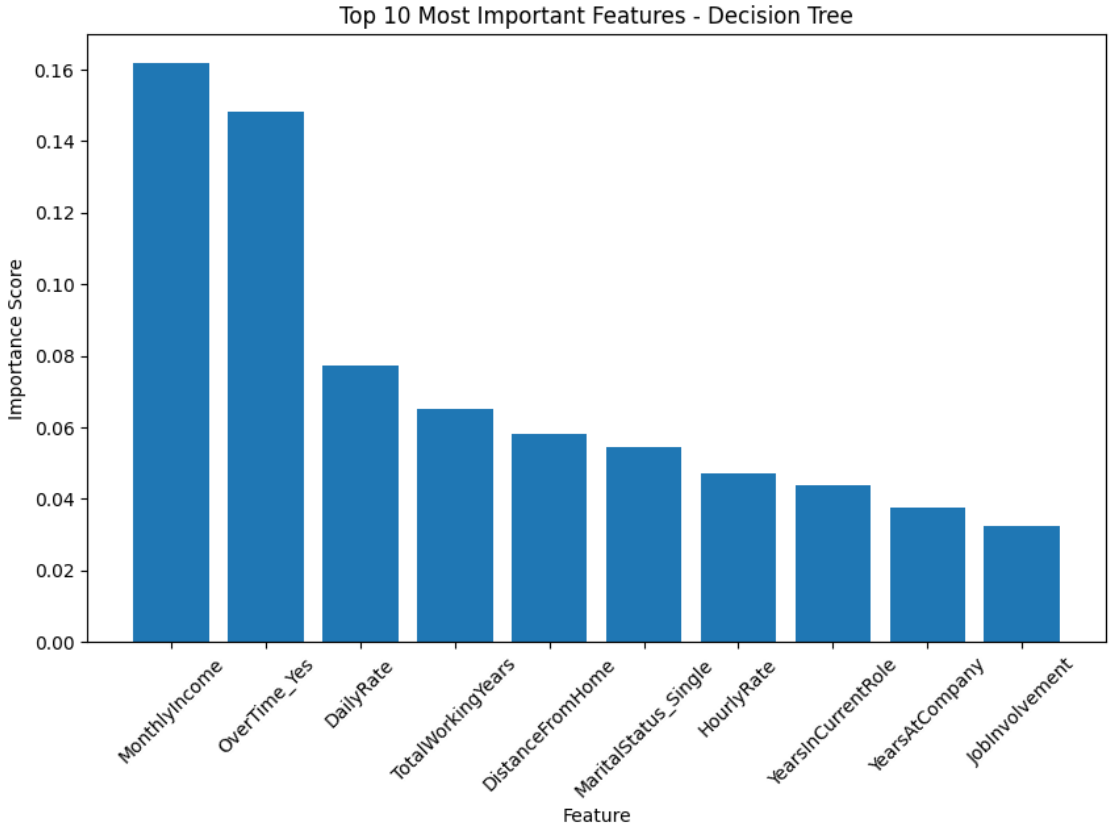
```
IN SAMPLE ACCURACY :  0.91
OUT OF SAMPLE ACCURACY :  0.83
```

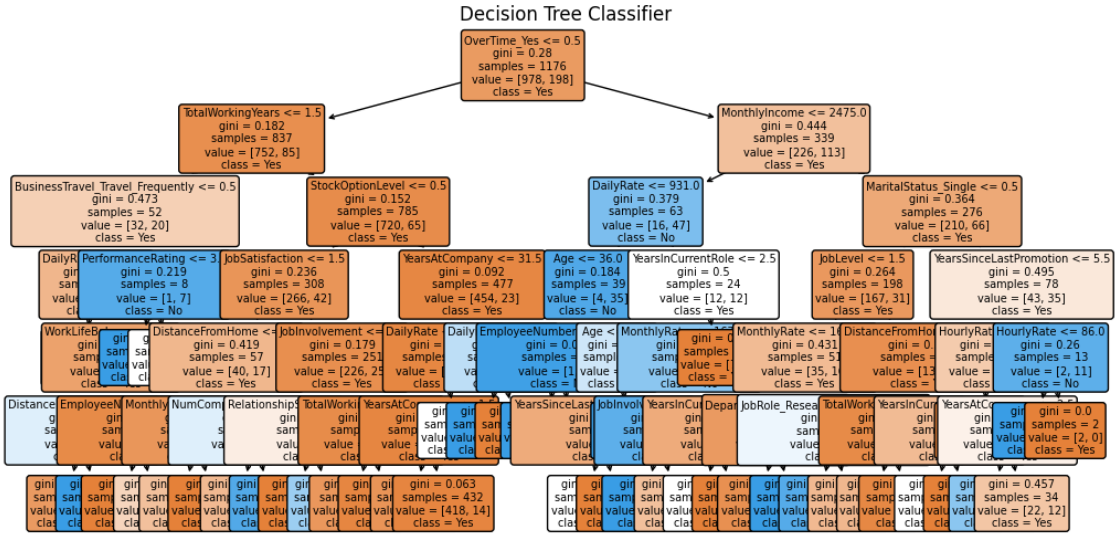
5 4.) Plot

[11]:
```
# 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=list(X.columns), class_names=["Yes",␣
↪"No"], rounded=True, fontsize=7)
plt.title('Decision Tree Classifier')
plt.show()
```




Top 10 Most Important Features - Decision Tree
Importance Score
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
MonthlyIncome
OverTime_Yes
DailyRate
TotalWorkingYears
DistanceFromHome
MaritalStatus_Single
HourlyRate
YearsInCurrentRole
YearsAtCompany
JobInvolvement
Feature


Decision Tree Classifier
OverTime_Yes <= 0.5
gini = 0.28
samples = 1176
value = [978, 198]
class = Yes
TotalWorkingYears <= 1.5
gini = 0.182
samples = 837
value = [752, 85]
class = Yes
MonthlyIncome <= 2475.0
gini = 0.444
samples = 339
value = [226, 113]
class = Yes
BusinessTravel_Travel_Frequently <= 0.5
gini = 0.473
samples = 52
value = [32, 20]
class = Yes
StockOptionLevel <= 0.5
gini = 0.152
samples = 785
value = [720, 65]
class = Yes
DailyRate <= 931.0
gini = 0.379
samples = 63
value = [16, 47]
class = No
MaritalStatus_Single <= 0.5
gini = 0.364
samples = 276
value = [210, 66]
class = Yes
PerformanceRating <= 3.
gini = 0.219
samples = 8
value = [1, 7]
class = No
JobSatisfaction <= 1.5
gini = 0.236
samples = 308
value = [266, 42]
class = Yes
YearsAtCompany <= 31.5
gini = 0.092
samples = 477
value = [454, 23]
class = Yes
Age <= 36.0
gini = 0.184
samples = 39
value = [4, 35]
class = No
YearsInCurrentRole <= 2.5
gini = 0.5
samples = 24
value = [12, 12]
class = Yes
JobLevel <= 1.5
gini = 0.264
samples = 198
value = [167, 31]
class = Yes
YearsSinceLastPromotion <= 5.5
gini = 0.495
samples = 78
value = [43, 35]
class = Yes
DistanceFromHome <=
gini = 0.419
samples = 57
value = [40, 17]
class = Yes
JobInvolvement <=
gini = 0.179
samples = 251
value = [226, 25
class = Yes
EmployeeNumber
gini = 0.0
samples = [1
class =
MonthlyRate <= 1
gini = 0.431
samples = 51
value = [35, 1
class = Yes
HourlyRate <= 86.0
gini = 0.26
samples = 13
value = [2, 11]
class = No
gini = 0.0
samples = 2
value = [2, 0]
class = Yes
gini = 0.063
samples = 432
value = [418, 14]
class = Yes
gini = 0.457
samples = 34
value = [22, 12]
class = Yes

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

6.1 ANSWER :

[12]:
```
from scipy.stats import pearsonr
def calculate_correlation(X, feature_name, y):
        feature= X[feature_name]
        coef, _= pearsonr(feature,y)
        return coef
```

Case 1: Overtime

[13]:
```
np.corrcoef(np.array(X["OverTime_Yes"]), np.array(y["Attrition"]))
```

[13]:
```
array([[1.        , 0.24611799],
       [0.24611799, 1.        ]])
```

The correlation between doing overtime and attrition is 0.24611799.

1. Work-Life Balance Initiatives:

- Encourage and promote a healthy work-life balance. Implement policies that discourage excessive overtime and provide employees with opportunities for flexible working hours or remote work when possible.

2. Employee Well-Being Programs:

- Invest in employee well-being programs that focus on physical and mental health. Provide resources, workshops, and support for stress management, mindfulness, and overall wellness to help employees cope with workplace demands.

3. Overtime Management Policies:

- Establish clear and transparent policies regarding overtime. Ensure that overtime is necessary and well-compensated. Communicate the reasons for overtime when it occurs and strive to minimize it, particularly if it becomes a consistent trend.

Case 2: Marital Status

[14]:
```
np.corrcoef(np.array(X["MaritalStatus_Single"]), np.array(y["Attrition"]))
```

[14]:
```
array([[1.        , 0.17541855],
       [0.17541855, 1.        ]])
```

1. Career Development Opportunities:

- Provide clear pathways for career development and advancement within the organization. Offer training, mentorship programs, and opportunities for skill enhancement to all employees, ensuring that career growth is not limited by marital status.

2. Social and Networking Events:

- Organize social and networking events specifically designed to engage single employees. These activities can help build a sense of community, foster connections, and create a supportive workplace environment.

3. Inclusive Benefits Packages:

- Ensure that benefits packages are inclusive and consider the needs of single employees. This may involve providing flexible healthcare options, financial wellness programs, or benefits that cater to individual lifestyle preferences.

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

Case 1: Noone works overtime

[15]:
```
x_train_experiment = x_train.copy()
```

[16]:
```
x_train_experiment["OverTime_Yes"] = 0
```

[17]:
```
y_pred_experiment = clf.predict(x_train_experiment)
y_pred = clf.predict(x_train)
```

[18]:
```
diff = sum(y_pred - y_pred_experiment)

print ("If noone was made to work overtime, we'd retain", diff , "customers")
```

```
If noone was made to work overtime, we'd retain 59 customers
```

Case 2: Everyone works overtime

[19]:
```
x_train_experiment["OverTime_Yes"] = 1
```

[20]:
```
y_pred_experiment = clf.predict(x_train_experiment)
y_pred = clf.predict(x_train)
```

[21]:
```
diff = sum(y_pred - y_pred_experiment)

print ("If everyone was made to work overtime, we'd lose", diff , "customers")
```

```
If everyone was made to work overtime, we'd lose -141 customers
```

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

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

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

11 What do you suggest to maximize company profits?

[22]:
```
x_train_experiment["Y"] = y_pred
x_train_experiment["Y_exp"]= y_pred_experiment
x_train_experiment["Ret_change"] = x_train_experiment["Y_exp"]-␣
↪x_train_experiment["Y"]
```

[23]:
```
sav = sum(-2.8 * x_train_experiment["Ret_change"]*␣
↪x_train_experiment["MonthlyIncome"])
```

[24]:
```
cost = len(x_train[x_train["OverTime_Yes"]==1])*2000
```

[25]:
```
sav-cost
```

[25]:
```
-2111037.2
```

This suggests that eliminating overtime would result in a net loss of around 117,594 dollars for the company. The expenses associated with removing overtime, considering a cost of $2,000 per employee, outweighs the potential savings derived from reduced attrition costs.

1. Flexible Work Arrangements:

- Introduce flexible work arrangements, such as compressed workweeks or remote work options, to accommodate employee preferences. This can enhance work-life balance and job satisfaction without incurring the full cost of overtime.

2. Efficiency Training Programs:

- Invest in training programs aimed at improving overall efficiency. Enhancing employees' skills and productivity can potentially reduce the need for excessive overtime, leading to cost savings.

3. Employee Engagement and Recognition:

- Foster a positive work environment through employee engagement initiatives and recognition programs. Satisfied and motivated employees may be more willing to contribute voluntarily, minimizing the need for enforced overtime.

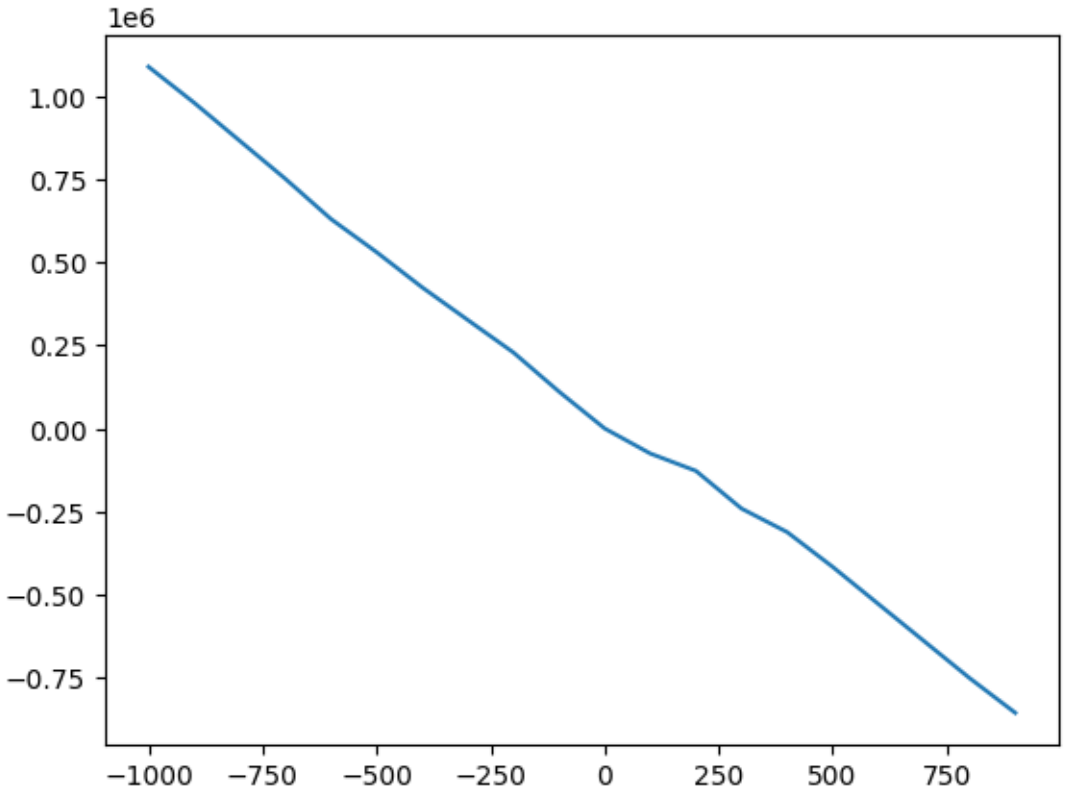
12 8.) Use your model and get the expected change in retention for raising and lowering peoples income. Plot the outcome of the experiment. Comment on the outcome of the experiment and your suggestions to maximize profit.

[26]:
```
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("profits,", sav-cost)
    profits.append(sav-cost)
```

```
Change in attrition -16
profits, 1087584.4
Change in attrition -14
profits, 979524.0
Change in attrition -13
profits, 864992.8
Change in attrition -12
profits, 750738.8
Change in attrition -12
profits, 629778.8
Change in attrition -9
profits, 530138.0
Change in attrition -7
profits, 424200.0
Change in attrition -4
profits, 326096.4
Change in attrition -1
profits, 228440.8
Change in attrition -1
profits, 110714.8
```

```
Change in attrition 0
profits, 0.0
Change in attrition 6
profits, -75328.40000000001
Change in attrition 15
profits, -127503.60000000002
Change in attrition 15
profits, -240914.8
Change in attrition 21
profits, -311586.80000000005
Change in attrition 22
profits, -416449.6000000001
Change in attrition 22
profits, -527889.6000000001
Change in attrition 22
profits, -639329.6000000001
Change in attrition 22
profits, -750769.6000000001
Change in attrition 23
profits, -854999.6000000001
```

[27]:
```
plt.plot(range(-1000, 1000, 100), profits)
plt.show()
```



This graph signifies a diminishing return associated with income variations. The declining trend suggests that as income levels change, there is a corresponding decrease in the expected retention outcomes. In the context of profit maximization, this implies that manipulating income levels may not be yielding proportional gains or, at some point, the costs associated with these changes outweigh the benefits.

1. Give bonuses in the range on the graph where the diminishing trend stabilizes or reaches an optimal value. This could indicate the income conditions under which profit is maximized.
2. Regularly assess the impact of bonus structures on both employee retention and company profits. Utilize this ongoing evaluation to make informed, data-driven adjustments to the bonus strategy.
3. Conduct a comprehensive cost-benefit analysis to understand the financial implications of income variations. Consider not only the direct costs but also the long-term benefits and potential risks associated with changes in income levels.
4. Investigate alternative strategies that might lead to profit maximization. This could involve adjusting income-related parameters, introducing new compensation models, or optimizing existing HR practices.