## Decision Trees

מצורף הקובץ votersdata.csv המכיל נתונים על בוחרים בארה״ב , כולל עמודת תוצאת הבחירה (רפובליקני/ דמוקרטי).

1. תחילה הגדירו random seed עם הערך 123 והשתמשו בו בפיצול הנתונים **ובאתחול המודל.**
2. הכרת הדאטה: נסו להכיר את הקשרים בין המשתנים השונים לבין עמודת המטרה vote.
   2. א. שרטטו stacked bar plots עבור המשתנים הקטגוריים.
   2. ב. שרטטו multivariate boxplot עבור הנומריים.
3. תקנו את הנתונים במידת הצורך. זה כולל טיפול בערכים חסרים, ערכים לא תקינים, נרמול והמרות נדרשות.
4. חלקו את הדאטה באופן רנדומי ל-70% training set ו-30% test set בשימוש ב seed שהגדרתם.
5. בנו מודל עץ בעזרת train set לחיזוי ערך המשתנה vote וענו על השאלות הבאות:

זכרו להציב Random seed

## Model Evaluation

6. בנו Confusion matrix עבור חיזוי על ה- test set בעזרת המודל שבניתם בשאלה הקודמת. הניחו כי ״דמוקרטי״ = Positive וחשבו את המדדים הבאים:
   א. Accuracy
   ב. Precision
   ג. Recall
7. השוו את המדדים עבור ה train set. האם מתקיימת תופעת ה-overfitting במודל החיזוי שבניתם? נמקו.
8. צרו מודל חדש משופר על סמך המסקנות מ 7. הגבילו את גובה העץ ל5 ואת כמות הרשומות לחיתוך ל40 וענו:

א. מהו עומק העץ ?

ב. כמה עלים יש בעץ ?

ג. מהו פיצ'ר החלוקה הטוב ביותר בעץ ?

ד. האם יש פיצ'רים שלא נכללו במודל ? מהם?

ה. האם תצפית מס׳ 68 (בדאטה סט המקורי) סווגה נכונה במודל ? נמקו.

9. בצעו שוב חיזוי על ה train set וה test sets ובנו מטריצות חדשות.

10. סטודנטים הריצו מודל משופר ויצאו להם המדדים הבאים:

Test set result:

Accuracy: 0.7946428571428571

Precision: 0.7142857142857143

recall: 0.9433962264150944

Train set results:

Accuracy: 0.7961538461538461

Precision: 0.7261904761904762

recall: 0.9457364341085271

מה אפשר להסיק מהתוצאות הנ"ל לגבי ביצועי המודל על הדאטה ? יש לפרט במילים.

**Decision tree - Multiclass**

שנו את עמודת המטרה להיות "status" ובנו עץ החלטה לזיהוי הסטטוס.

בנו confusion matrix והדפיסו את מדד ה **accuracy**לאחר בדיקה על הtest set.

כתבו את התוצאה גם בקוד כהערה וענו:

11. האם המודל יחזה טוב את הסטטוס המשפחתי על דאטה חדש ? מדוע?

12. האם קיים חשד ל overfitting ?

13. חשבו את מדד ה **precision** עבור הקטגוריה single.

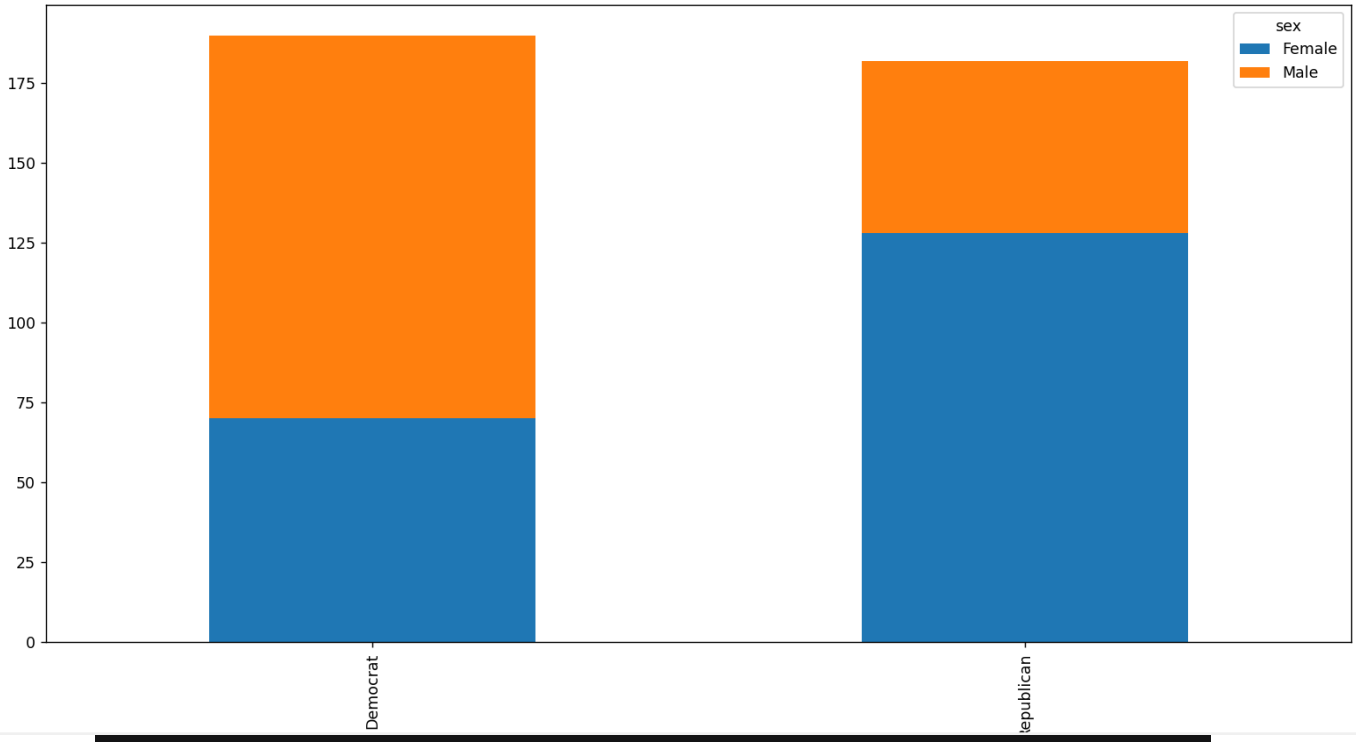
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import
train_test_split
from sklearn import metrics, tree
from sklearn import preprocessing
import scipy.stats as stats
from sklearn.tree import
DecisionTreeClassifier, plot_tree
df = pd.read_csv('voters_hm4.csv')
```

שאלה 1.

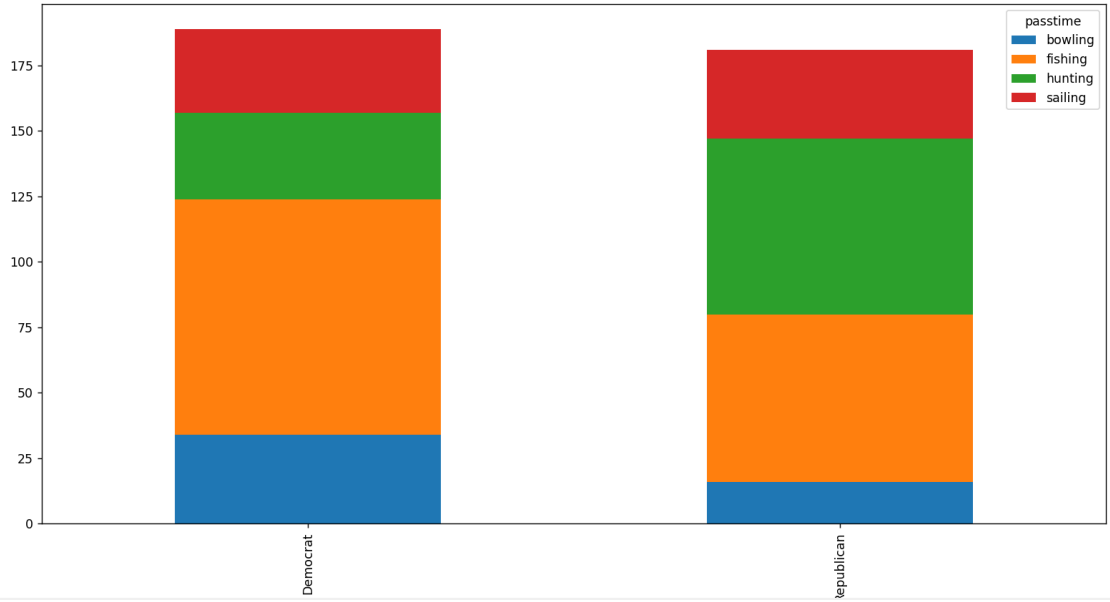
```
# Q1
r_seed = 123
```

שאלה 2א.

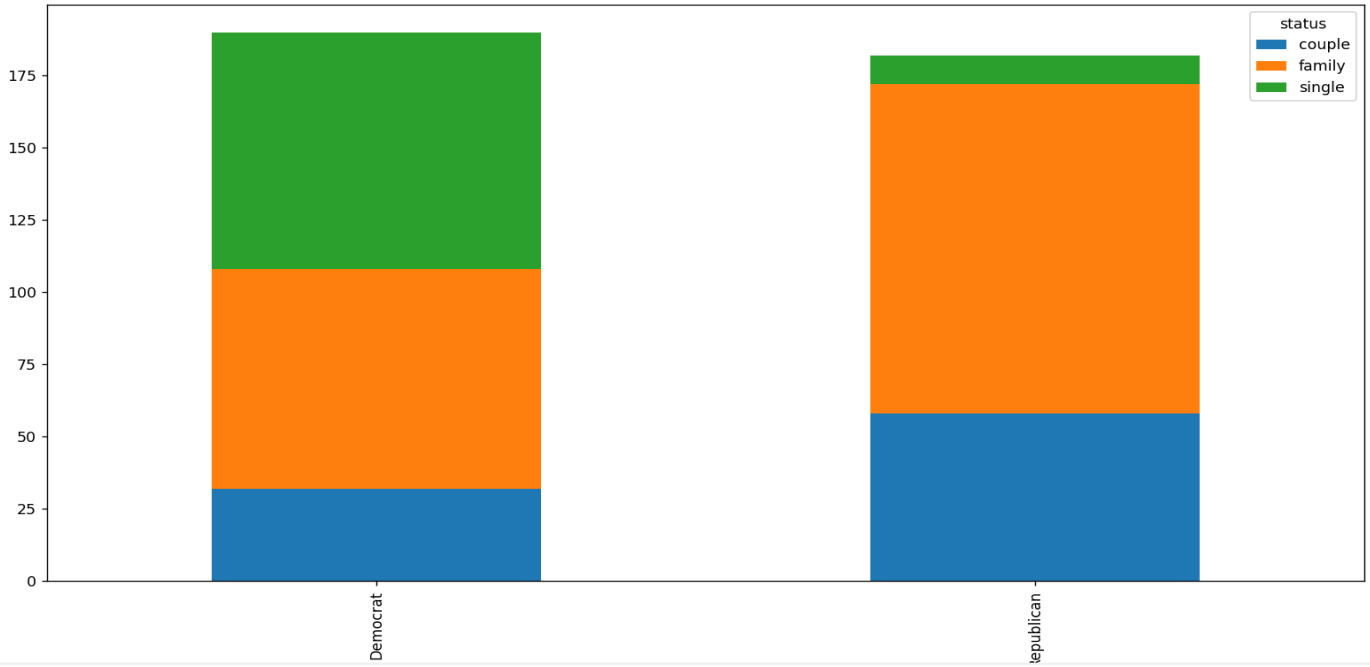
```
# Q2.a
Crosstab = pd.crosstab(index= df.vote,
columns= df.sex)
Crosstab.plot.bar(stacked= True)
plt.show()
```




```
Crosstab = pd.crosstab(index= df.vote,
columns= df.passtime)
Crosstab.plot.bar(stacked= True)
plt.show()
```

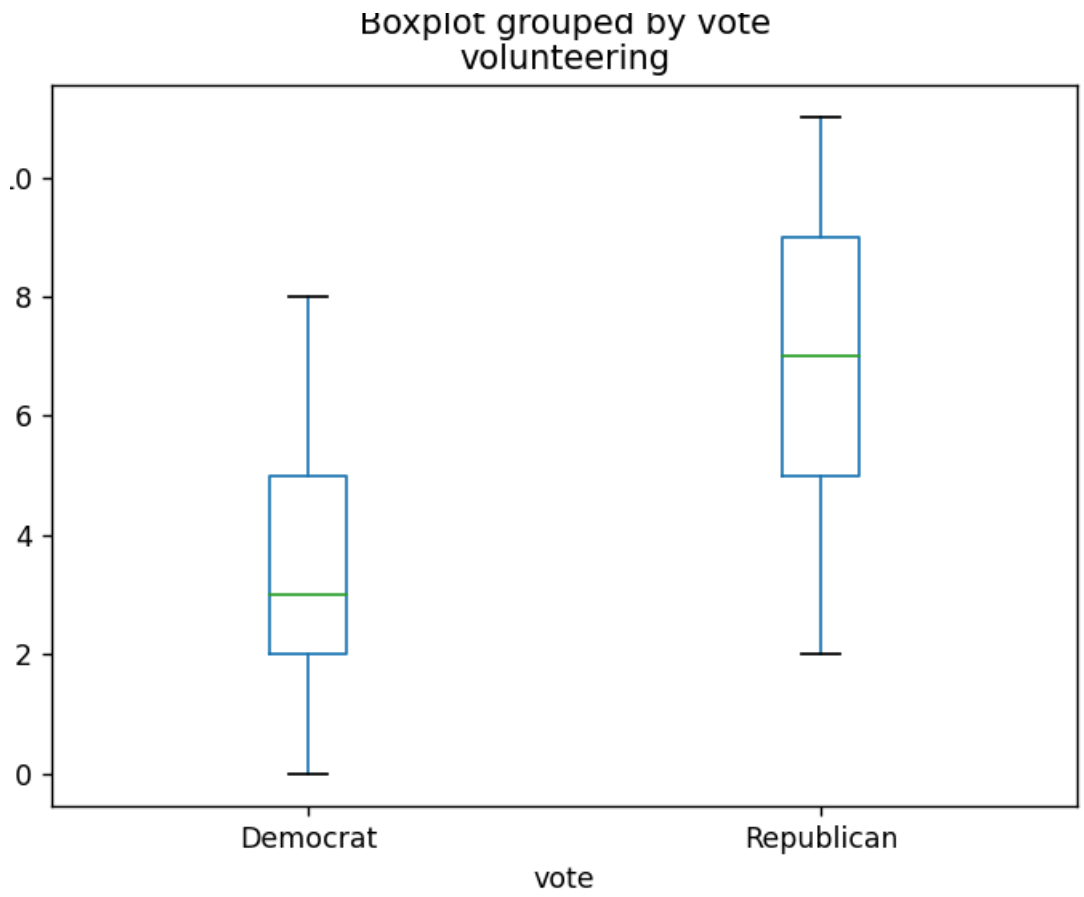


```
Crosstab = pd.crosstab(index= df.vote,
columns= df.status)
Crosstab.plot.bar(stacked= True)
plt.show()
```

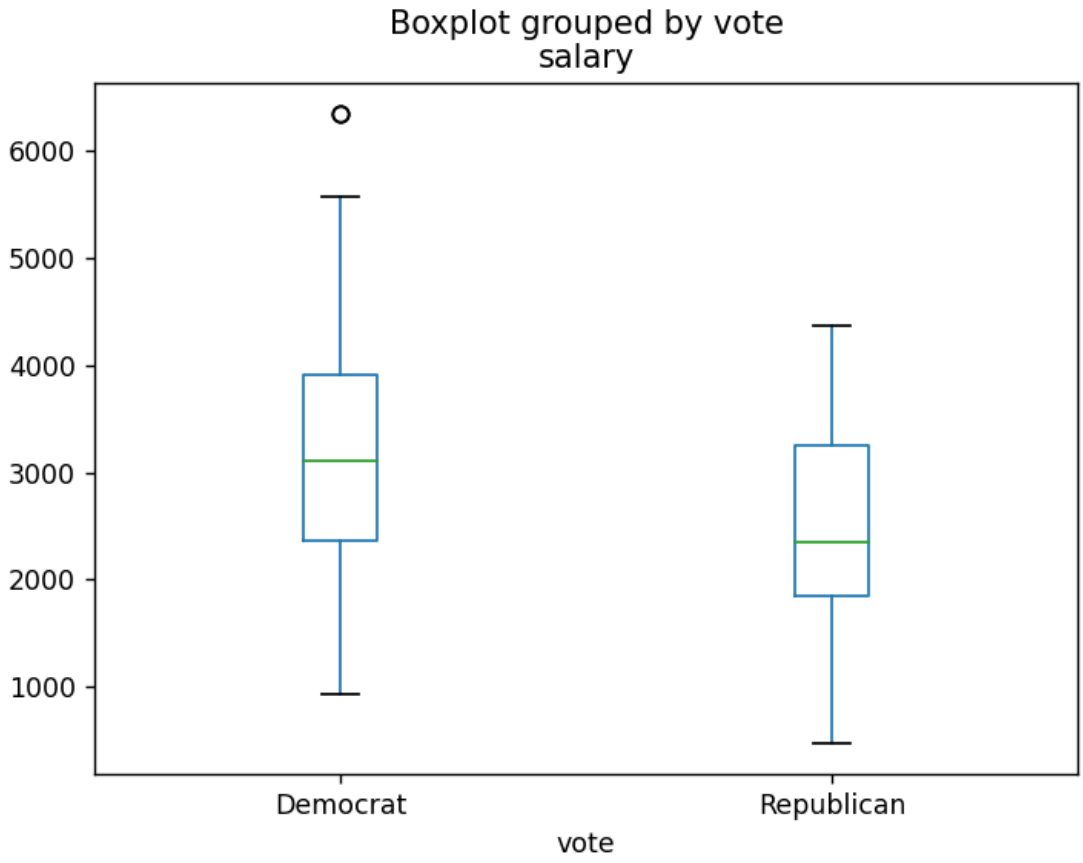



שאלה 2ב.

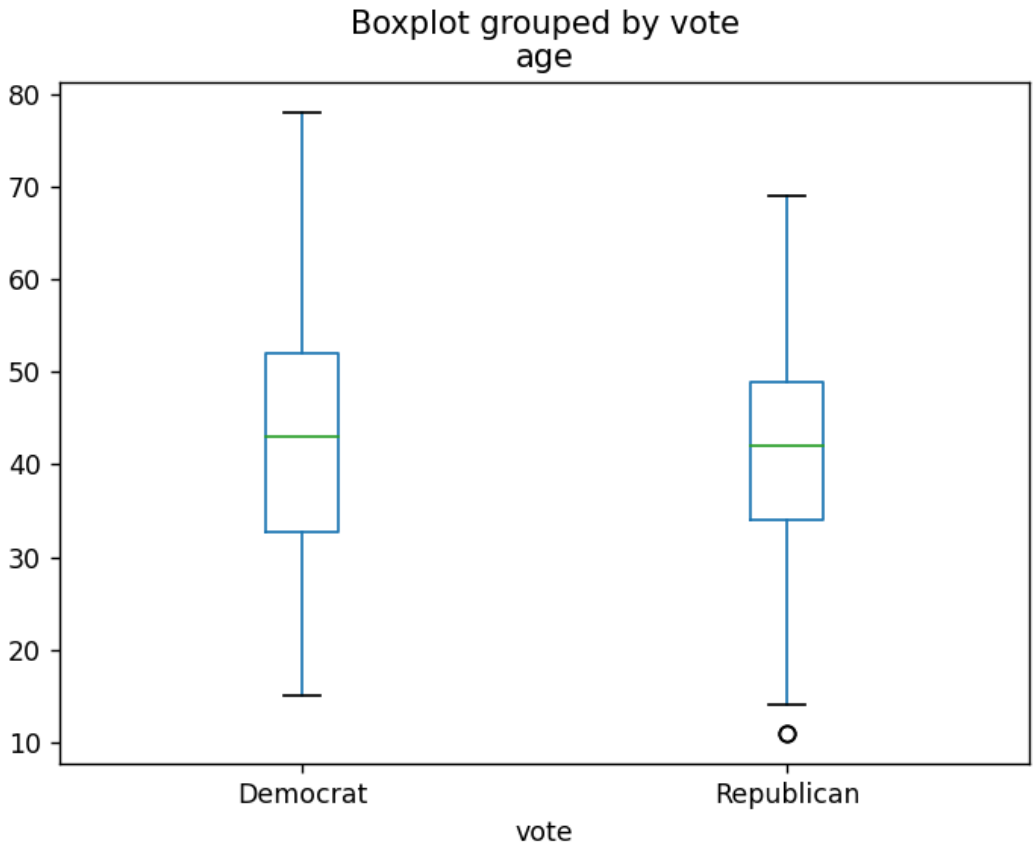
```
# Q2.b
df.boxplot(column=['volunteering'], by=
'vote', grid=False)
plt.show()
```




```
df.boxplot(column=['salary'], by= 'vote',
grid=False)
plt.show()
```




```
df.boxplot(column=['age'], by= 'vote',
grid=False)
plt.show()
```




שאלה 3.

```
# Q3
# we check how many nulls are in the data
print(df.isnull().sum())
df.dropna(subset=['passtime'],
inplace=True)
df['age'] =
df['age'].replace(to_replace=np.nan,value=d
f.age.mean())
print('****\n',df.isnull().sum())
df['salary'] =
df['salary'].replace(to_replace=np.nan,valu
e=df.salary.mean())
print('****\n',df.isnull().sum())
```

הדפסה:

```
sex              0
age             10
salary          36
volunteering     0
passtime         2
status           0
vote             0
dtype: int64
****
 sex
age              0
salary          36
volunteering     0
passtime         0
status           0
vote             0
dtype: int64
****
 sex             0
age             0
salary          0
volunteering    0
passtime        0
status          0
vote            0
dtype: int64
```

שאלה 4.

```
# Q4
le_vote = preprocessing.LabelEncoder()
le_sex = preprocessing.LabelEncoder()
le_pass = preprocessing.LabelEncoder()
le_status = preprocessing.LabelEncoder()
le_vote.fit(df['vote'])
df['target'] =
le_vote.transform(df['vote'])
df['sex_s'] =
le_sex.fit_transform(df['sex'])
df['passtime_p'] =
```

```
le_pass.fit_transform(df['passtime'])
df['status_s'] =
le_status.fit_transform(df['status'])
x = df.drop(['vote', 'target', 'sex',
'passtime', 'status'], axis= 1)
y = df['target']
x_train, x_test, y_train, y_test =
train_test_split(x, y, test_size=0.3,
random_state=r_seed)
```

שאלה 5.

```
# Q5
model =
DecisionTreeClassifier(random_state=r_seed)
model.fit(x_train, y_train)
```

שאלה 6.

```
# Q6
# bulding a confusion matrix
y_pred_test = model.predict(x_test)
cm_test = pd.crosstab(y_test, y_pred_test,
colnames=["pred"], margins=True)
print(cm_test)
print("Test set result:")
print("Accuracy:",
metrics.accuracy_score(y_test,y_pred_test))
print("Precision: ",
metrics.precision_score(y_test,
y_pred_test))
print("Recall: ",
metrics.recall_score(y_test, y_pred_test))
```

הדפסה:

```
pred     0   1  All
target
0       56   0   56
1        6  49   55
All     62  49  111
Test set result:
Accuracy: 0.9459459459459459
Precision:  1.0
Recall:  0.8909090909090909
```

שאלה7.

```
# Q7
# matrix for train set
y_pred_train = model.predict(x_train)
cm_train = pd.crosstab(y_train,
y_pred_train, colnames = ["pred"], margins
= True)
print(cm_train)
print("Train set result:")
print("Accuracy:",
metrics.accuracy_score(y_train,y_pred_train
))
print("Precision: ",
metrics.precision_score(y_train,
y_pred_train))
print("Recall: ",
metrics.recall_score(y_train,
y_pred_train))
```

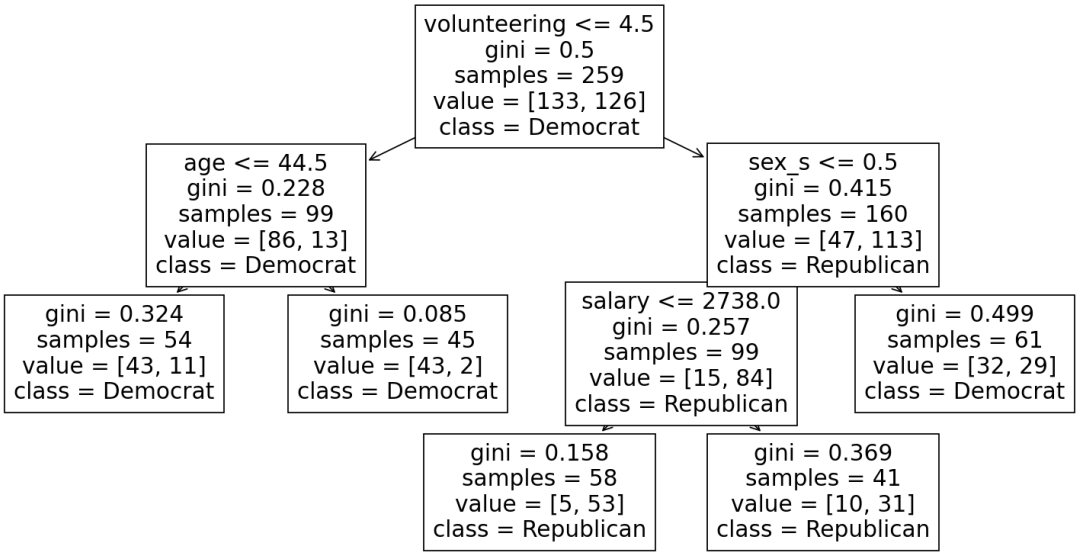
הדפסה :

```
pred      0    1  All
target
0       133    0  133
1         0  126  126
All     133  126  259
Train set result:
Accuracy: 1.0
Precision:  1.0
Recall:  1.0
```

```
# The model is over fitted to the train set
# because the train results are almost
```

```
perfect while the test set is providing
less high results
```

שאלה 8.




```
# Q8
re_model =
DecisionTreeClassifier(max_depth=5,
min_samples_leaf=40, random_state=r_seed)
re_model.fit(x_train, y_train)
fig = plt.figure(figsize=(14, 8))
tree.plot_tree(re_model, feature_names=
x_train.columns, class_names=
le_vote.classes_)
plt.show()
```

שאלה 8א.

```
# Q8a
# Tree depth is 3
```

שאלה 8ב.

```
# Q8b
# There are 5 leaves on the tree
```

שאלה 8ג.

```
# Q8c
# Best splitting feature is "volunteering"
```

שאלה 8ד.

```
# Q8d
# Not all features were included in the
tree: passtime and status
```

שאלה 8ה.

```
rr = x.iloc[[68]]
print("Sample number 68 predicted to be ",
le_vote.inverse_transform(re_model.predict(
rr)), "and in reality its 'Republican'")
# Sample number 68 was correctly classified
as "republican" as it is in the original
data frame
```

הדפסה :

```
Sample number 68 predicted to be  ['Republican'] and in reality its 'Republican'
```

שאלה 9.

```
# Q9
# Matrix for test set
y_pred_test = re_model.predict(x_test)
cm_test = pd.crosstab(y_test, y_pred_test,
colnames = ["pred"], margins = True)
print("*************\nREVISED MODEL")
print(cm_test)
print("Test set result:")
```

```
print("Accuracy:",
metrics.accuracy_score(y_test,y_pred_test))
print("Precision: ",
metrics.precision_score(y_test,
y_pred_test))
print("Recall: ",
metrics.recall_score(y_test, y_pred_test))
# Matrix for train set
y_pred_train = re_model.predict(x_train)
cm_train = pd.crosstab(y_train,
y_pred_train, colnames = ["pred"], margins
= True)
print(cm_train)
print("Train set result:")
print("Accuracy:",
metrics.accuracy_score(y_train,y_pred_train
))
print("Precision: ",
metrics.precision_score(y_train,
y_pred_train))
print("Recall: ",
metrics.recall_score(y_train,
y_pred_train))
```

הדפסה :

```
*************
REVISED MODEL
pred     0   1  All
target
0       47   9   56
1       22  33   55
All     69  42  111
Test set result:
Accuracy: 0.7207207207207207
Precision:  0.7857142857142857
Recall:  0.6
pred      0   1  All
target
0       118  15  133
1        42  84  126
All     160  99  259
Train set result:
Accuracy: 0.7799227799227799
Precision:  0.8484848484848485
Recall:  0.6666666666666666
```

שאלה 10.

```
# Q10
'''
Can conclude from those results that the
model is not overfitted might be
underfitted
and spot relatively well most of the sample
that are positive and predict
them as true(recall).
On the other hand, the model preform
relatively weak results in accuracy and
precision
'''
```

הדפסה:

```
*************
STATUS MODEL
pred       0   1   2  All
status_s
0         20   6   0   26
1          5  49   2   56
2          2   5  22   29
All       27  60  24  111
Test set result:
Accuracy: 0.8198198198198198
Precision:  [0.74074074 0.81666667 0.91666667]
Recall:  [0.76923077 0.875      0.75862069]
pred       0    1   2  All
status_s
0         64    0   0   64
1          0  133   0  133
2          0    0  62   62
All       64  133  62  259
Train set result:
Accuracy: 1.0
Precision:  [1. 1. 1.]
Recall:  [1. 1. 1.]
```

# Decision tree - Multiclass

```
z = df.drop(['vote', 'sex', 'passtime',
'status', 'status_s'], axis= 1)
w = df['status_s']
z_train, z_test, w_train, w_test =
train_test_split(z, w, test_size=0.3,
random_state=r_seed)
st_model =
DecisionTreeClassifier(random_state=r_seed)
st_model.fit(z_train, w_train)
w_pred_test = st_model.predict(z_test)
cm_test = pd.crosstab(w_test, w_pred_test,
colnames = ["pred"], margins = True)
print("*************\nSTATUS MODEL")
print(cm_test)
print("Test set result:")
```

```
print("Accuracy:",
metrics.accuracy_score(w_test,
w_pred_test))
print("Precision: ",
metrics.precision_score(w_test,
w_pred_test, average=None))
print("Recall: ",
metrics.recall_score(w_test, w_pred_test,
average=None))
# Matrix for train set
w_pred_train = st_model.predict(z_train)
cm_train = pd.crosstab(w_train,
w_pred_train, colnames=["pred"],
margins=True)
print(cm_train)
print("Train set result:")
print("Accuracy:",
metrics.accuracy_score(w_train,
w_pred_train))
print("Precision: ",
metrics.precision_score(w_train,
w_pred_train, average=None))
print("Recall: ",
metrics.recall_score(w_train, w_pred_train,
average=None))
```

שאלה 11.

```
# Q11 Accuracy is: 0.82
# The model won't predict the status that
well, we are looking for accuracy higher
than what we got
```

שאלה 12.

```
# Q12
# Yes, the model is suspected to be
```

```
overfitted
# when comparing test results and train
results, we get perfect train results
# and relatively bad test results
```

שאלה 13.

```
# Q13
# Precision for the "single" category in
Status: 0.91666667
```