; Decision Tree מודלים

.1

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

model_decision_tree = DecisionTreeClassifier(criterion = 'gini', max_depth = 8, min_samples_split = 6, min_samples_leaf = 5)
model_decision_tree.fit(X_train, Y_train)

model_decision_tree_predictions = model_decision_tree.predict(X_test)
print(classification_report(Y_test, model_decision_tree_predictions))
confusion_matrix(Y_test, model_decision_tree_predictions)
```

```
              precision    recall  f1-score   support

           0       0.95      0.85      0.90        41
           1       0.92      0.97      0.95        73

    accuracy                           0.93       114
   macro avg       0.93      0.91      0.92       114
weighted avg       0.93      0.93      0.93       114

array([[35,  6],
       [ 2, 71]])
```

.2

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

model_decision_tree = DecisionTreeClassifier(criterion = 'gini', max_depth = 5, min_samples_split = 2, min_samples_leaf = 2)
model_decision_tree.fit(X_train, Y_train)

model_decision_tree_predictions = model_decision_tree.predict(X_test)
print(classification_report(Y_test, model_decision_tree_predictions))
confusion_matrix(Y_test, model_decision_tree_predictions)
```

```
              precision    recall  f1-score   support

           0       0.95      0.93      0.94        44
           1       0.96      0.97      0.96        70

    accuracy                           0.96       114
   macro avg       0.96      0.95      0.95       114
weighted avg       0.96      0.96      0.96       114

array([[41,  3],
       [ 2, 68]])
```

.3

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

model_decision_tree = DecisionTreeClassifier(criterion = 'entropy', max_depth = 10, min_samples_split = 2, min_samples_leaf = 5)
model_decision_tree.fit(X_train, Y_train)

model_decision_tree_predictions = model_decision_tree.predict(X_test)
print(classification_report(Y_test, model_decision_tree_predictions))
confusion_matrix(Y_test, model_decision_tree_predictions)
```

```
              precision    recall  f1-score   support

           0       0.92      1.00      0.96        44
           1       1.00      0.94      0.97        70

    accuracy                           0.96       114
   macro avg       0.96      0.97      0.96       114
weighted avg       0.97      0.96      0.97       114

array([[44,  0],
       [ 4, 66]])
```

.4

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

model_decision_tree = DecisionTreeClassifier(criterion = 'gini', max_depth = 15, min_samples_split = 5, min_samples_leaf = 10)
model_decision_tree.fit(X_train, Y_train)

model_decision_tree_predictions = model_decision_tree.predict(X_test)
print(classification_report(Y_test, model_decision_tree_predictions))
confusion_matrix(Y_test, model_decision_tree_predictions)
```

```
              precision    recall  f1-score   support

           0       0.94      0.77      0.85        44
           1       0.87      0.97      0.92        70

    accuracy                           0.89       114
   macro avg       0.91      0.87      0.88       114
weighted avg       0.90      0.89      0.89       114

array([[34, 10],
       [ 2, 68]])
```

.5

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

model_decision_tree = DecisionTreeClassifier(criterion = 'gini', max_depth = None, min_samples_split = 2, min_samples_leaf = 2)
model_decision_tree.fit(X_train, Y_train)

model_decision_tree_predictions = model_decision_tree.predict(X_test)
print(classification_report(Y_test, model_decision_tree_predictions))
confusion_matrix(Y_test, model_decision_tree_predictions)
```

```
              precision    recall  f1-score   support

           0       0.95      0.95      0.95        44
           1       0.97      0.97      0.97        70

    accuracy                           0.96       114
   macro avg       0.96      0.96      0.96       114
weighted avg       0.96      0.96      0.96       114

array([[42,  2],
       [ 2, 68]])
```

.6

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

model_decision_tree = DecisionTreeClassifier(criterion = 'gini', max_depth = 10, min_samples_split = 3, min_samples_leaf = 5)
model_decision_tree.fit(X_train, Y_train)

model_decision_tree_predictions = model_decision_tree.predict(X_test)
print(classification_report(Y_test, model_decision_tree_predictions))
confusion_matrix(Y_test, model_decision_tree_predictions)
```

```
              precision    recall  f1-score   support

           0       0.89      0.75      0.81        44
           1       0.86      0.94      0.90        70

    accuracy                           0.87       114
   macro avg       0.87      0.85      0.86       114
weighted avg       0.87      0.87      0.87       114

array([[33, 11],
       [ 4, 66]])
```

: Random forest 사용

.1

```python
from sklearn.ensemble import RandomForestClassifier

model_random_forest = RandomForestClassifier(n_estimators = 50, max_depth = 10, min_samples_split = 2, min_samples_leaf = 2, criterion = 'gini')
model_random_forest.fit(X_train, Y_train)

model_random_forest_predictions = model_random_forest.predict(X_test)
print(classification_report(Y_test, model_random_forest_predictions))
confusion_matrix(Y_test, model_random_forest_predictions)
```

```
              precision    recall  f1-score   support

           0       0.96      0.98      0.97        44
           1       0.99      0.97      0.98        70

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114

array([[43,  1],
       [ 2, 68]])
```

.2

```python
[102] from sklearn.ensemble import RandomForestClassifier

model_random_forest = RandomForestClassifier(n_estimators = 300, max_depth = 20, min_samples_split = 2, min_samples_leaf = 3, criterion = 'entropy')
model_random_forest.fit(X_train, Y_train)

model_random_forest_predictions = model_random_forest.predict(X_test)
print(classification_report(Y_test, model_random_forest_predictions))
confusion_matrix(Y_test, model_random_forest_predictions)
```

```
              precision    recall  f1-score   support

           0       0.98      0.94      0.96        48
           1       0.96      0.98      0.97        66

    accuracy                           0.96       114
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114

array([[45,  3],
       [ 1, 65]])
```

.3

```python
from sklearn.ensemble import RandomForestClassifier

model_random_forest = RandomForestClassifier(n_estimators = 250, max_depth = 50, min_samples_split = 5, min_samples_leaf = 5, criterion = 'gini')
model_random_forest.fit(X_train, Y_train)

model_random_forest_predictions = model_random_forest.predict(X_test)
print(classification_report(Y_test, model_random_forest_predictions))
confusion_matrix(Y_test, model_random_forest_predictions)
```

```
              precision    recall  f1-score   support

           0       0.98      0.96      0.97        48
           1       0.97      0.98      0.98        66

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114

array([[46,  2],
       [ 1, 65]])
```

.4

```python
from sklearn.ensemble import RandomForestClassifier

model_random_forest = RandomForestClassifier(n_estimators = 50, max_depth = 2, min_samples_split = 5, min_samples_leaf = 5, criterion = 'entropy')
model_random_forest.fit(X_train, Y_train)

model_random_forest_predictions = model_random_forest.predict(X_test)
print(classification_report(Y_test, model_random_forest_predictions))
confusion_matrix(Y_test, model_random_forest_predictions)
```

```
              precision    recall  f1-score   support

           0       0.98      0.92      0.95        48
           1       0.94      0.98      0.96        66

    accuracy                           0.96       114
   macro avg       0.96      0.95      0.95       114
weighted avg       0.96      0.96      0.96       114

array([[44,  4],
       [ 1, 65]])
```

.5

```python
from sklearn.ensemble import RandomForestClassifier

model_random_forest = RandomForestClassifier(n_estimators = 75, max_depth = 25, min_samples_split = 10, min_samples_leaf = 3, criterion = 'gini')
model_random_forest.fit(X_train, Y_train)

model_random_forest_predictions = model_random_forest.predict(X_test)
print(classification_report(Y_test, model_random_forest_predictions))
confusion_matrix(Y_test, model_random_forest_predictions)
```

```
              precision    recall  f1-score   support

           0       0.98      0.94      0.96        48
           1       0.96      0.98      0.97        66

    accuracy                           0.96       114
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114

array([[45,  3],
       [ 1, 65]])
```

.6

```python
71] from sklearn.ensemble import RandomForestClassifier

model_random_forest = RandomForestClassifier(n_estimators = 100, max_depth = 10, min_samples_split = 10, min_samples_leaf = 2, criterion="gini")
model_random_forest.fit(X_train, Y_train)

model_random_forest_predictions = model_random_forest.predict(X_test)
print(classification_report(Y_test, model_random_forest_predictions))
confusion_matrix(Y_test, model_random_forest_predictions)
```

```
              precision    recall  f1-score   support

           0       0.98      0.98      0.98        43
           1       0.99      0.99      0.99        71

    accuracy                           0.98       114
   macro avg       0.98      0.98      0.98       114
weighted avg       0.98      0.98      0.98       114

array([[42,  1],
       [ 1, 70]])
```

: Adaboost 의 경우

.7

```python
[64] from sklearn.ensemble import AdaBoostClassifier

model_adaboost = AdaBoostClassifier(n_estimators = 50, learning_rate = 0.1, estimator = DecisionTreeClassifier(max_depth = 1), algorithm="SAMME.R")
model_adaboost.fit(X_train, Y_train)

model_ada_boost_predictions = model_adaboost.predict(X_test)
print(classification_report(Y_test, model_ada_boost_predictions))
confusion_matrix(Y_test, model_ada_boost_predictions)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_weight_boosting.py:527: FutureWarning: The SAMME.R algorithm (the default) is deprecated and will be remo
  warnings.warn(
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        43
           1       0.97      0.99      0.98        71

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114

array([[41,  2],
       [ 1, 70]])
```

.2

```python
from sklearn.ensemble import AdaBoostClassifier

model_adaboost = AdaBoostClassifier(n_estimators = 100, learning_rate = 0.1, estimator = DecisionTreeClassifier(max_depth = 2), algorithm="SAMME")
model_adaboost.fit(X_train, Y_train)

model_ada_boost_predictions = model_adaboost.predict(X_test)
print(classification_report(Y_test, model_ada_boost_predictions))
confusion_matrix(Y_test, model_ada_boost_predictions)
```

```
              precision    recall  f1-score   support

           0       1.00      0.87      0.93        52
           1       0.90      1.00      0.95        62

    accuracy                           0.94       114
   macro avg       0.95      0.93      0.94       114
weighted avg       0.94      0.94      0.94       114

array([[45,  7],
       [ 0, 62]])
```

3.

```python
from sklearn.ensemble import AdaBoostClassifier

model_adaboost = AdaBoostClassifier(n_estimators = 200, learning_rate = 0.01, estimator = DecisionTreeClassifier(max_depth = 3), algorithm="SAMME.R")
model_adaboost.fit(X_train, Y_train)

model_ada_boost_predictions = model_adaboost.predict(X_test)
print(classification_report(Y_test, model_ada_boost_predictions))
confusion_matrix(Y_test, model_ada_boost_predictions)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_weight_boosting.py:527: FutureWarning: The SAMME.R algorithm (the default) is deprecated and
  warnings.warn(
              precision    recall  f1-score   support

           0       1.00      0.85      0.92        52
           1       0.89      1.00      0.94        62

    accuracy                           0.93       114
   macro avg       0.94      0.92      0.93       114
weighted avg       0.94      0.93      0.93       114

array([[44,  8],
       [ 0, 62]])
```

4.

```python
from sklearn.ensemble import AdaBoostClassifier

model_adaboost = AdaBoostClassifier(n_estimators = 75, learning_rate = 0.5, estimator = DecisionTreeClassifier(max_depth = 2), algorithm="SAMME")
model_adaboost.fit(X_train, Y_train)

model_ada_boost_predictions = model_adaboost.predict(X_test)
print(classification_report(Y_test, model_ada_boost_predictions))
confusion_matrix(Y_test, model_ada_boost_predictions)
```

```
              precision    recall  f1-score   support

           0       1.00      0.87      0.93        52
           1       0.90      1.00      0.95        62

    accuracy                           0.94       114
   macro avg       0.95      0.93      0.94       114
weighted avg       0.94      0.94      0.94       114

array([[45,  7],
       [ 0, 62]])
```

5.

```python
from sklearn.ensemble import AdaBoostClassifier

model_adaboost = AdaBoostClassifier(n_estimators = 200, learning_rate = 0.1, estimator = DecisionTreeClassifier(max_depth = 3), algorithm="SAMME.R")
model_adaboost.fit(X_train, Y_train)

model_ada_boost_predictions = model_adaboost.predict(X_test)
print(classification_report(Y_test, model_ada_boost_predictions))
confusion_matrix(Y_test, model_ada_boost_predictions)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_weight_boosting.py:527: FutureWarning: The SAMME.R algorithm (the default) is deprecated and w
  warnings.warn(
              precision    recall  f1-score   support

           0       0.98      0.87      0.92        52
           1       0.90      0.98      0.94        62

    accuracy                           0.93       114
   macro avg       0.94      0.92      0.93       114
weighted avg       0.93      0.93      0.93       114

array([[45,  7],
       [ 1, 61]])
```

6.

```python
from sklearn.ensemble import AdaBoostClassifier

model_adaboost = AdaBoostClassifier(n_estimators = 300, learning_rate = 0.05, estimator = DecisionTreeClassifier(max_depth = 2), algorithm="SAMME.R")
model_adaboost.fit(X_train, Y_train)

model_ada_boost_predictions = model_adaboost.predict(X_test)
print(classification_report(Y_test, model_ada_boost_predictions))
confusion_matrix(Y_test, model_ada_boost_predictions)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_weight_boosting.py:527: FutureWarning: The SAMME.R algorithm (the default) is deprecated and wi
  warnings.warn(
              precision    recall  f1-score   support

           0       1.00      0.88      0.94        52
           1       0.91      1.00      0.95        62

    accuracy                           0.95       114
   macro avg       0.96      0.94      0.95       114
weighted avg       0.95      0.95      0.95       114

array([[46,  6],
       [ 0, 62]])
```

סיכום:

*כפי שניתן לראות מן הנתונים הבנו שהדיוק עבור כל מודל נתון נמצא
מעל F1 score לאחר הקלה ״ריה, מכיוון שבוצע מדגם על
כמה כי המודל יכול לפגוע מקרים של סרטן, ולאחות

מצורפים לקבצים 1 (False negative) שגיאות למודל

מסוכן מאוד.

תוצאות לומדים מיותר עבור כל מודל:

Decision Tree:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

model_decision_tree = DecisionTreeClassifier(criterion = 'gini', max_depth = None, min_samples_split = 2, min_samples_leaf = 2)
model_decision_tree.fit(X_train, Y_train)

model_decision_tree_predictions = model_decision_tree.predict(X_test)
print(classification_report(Y_test, model_decision_tree_predictions))
confusion_matrix(Y_test, model_decision_tree_predictions)
```

```
              precision    recall  f1-score   support

           0       0.95      0.95      0.95        44
           1       0.97      0.97      0.97        70

    accuracy                           0.96       114
   macro avg       0.96      0.96      0.96       114
weighted avg       0.96      0.96      0.96       114

array([[42,  2],
       [ 2, 68]])
```

נשים לב: המודל לא יודע recall, כי F1 (קבצים 1) שגיאות מגדרגה

5 ו 3, recall תשומת לב מ precision במודלים טוב:

1. recall (False negative): כמה מה לא ומה לא לוגיקה מלאים

כמודלים יקר.

2. precision (false positive): מאוד למעניין מאוד מאוד

אבל recall מאמת מודלים תשומת לב

Random Forest:

```
71] from sklearn.ensemble import RandomForestClassifier

model_random_forest = RandomForestClassifier(n_estimators = 100, max_depth = 10, min_samples_split = 10, min_samples_leaf = 2, criterion="gini")
model_random_forest.fit(X_train, Y_train)

model_random_forest_predictions = model_random_forest.predict(X_test)
print(classification_report(Y_test, model_random_forest_predictions))
confusion_matrix(Y_test, model_random_forest_predictions)
```

```
              precision    recall  f1-score   support

           0       0.98      0.98      0.98        43
           1       0.99      0.99      0.99        71

    accuracy                           0.98       114
   macro avg       0.98      0.98      0.98       114
weighted avg       0.98      0.98      0.98       114

array([[42,  1],
       [ 1, 70]])
```

Ada boost :

```
[64] from sklearn.ensemble import AdaBoostClassifier

     model_adaboost = AdaBoostClassifier(n_estimators = 50, learning_rate = 0.1, estimator = DecisionTreeClassifier(max_depth = 1), algorithm="SAMME.R")
     model_adaboost.fit(X_train, Y_train)

     model_ada_boost_predictions = model_adaboost.predict(X_test)
     print(classification_report(Y_test, model_ada_boost_predictions))
     confusion_matrix(Y_test, model_ada_boost_predictions)

/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_weight_boosting.py:527: FutureWarning: The SAMME.R algorithm (the default) is deprecated and will be remo
  warnings.warn(
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        43
           1       0.97      0.99      0.98        71

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114

array([[41,  2],
       [ 1, 70]])
```

best result :

```
71] from sklearn.ensemble import RandomForestClassifier

    model_random_forest = RandomForestClassifier(n_estimators = 100, max_depth = 10, min_samples_split = 10, min_samples_leaf = 2, criterion="gini")
    model_random_forest.fit(X_train, Y_train)

    model_random_forest_predictions = model_random_forest.predict(X_test)
    print(classification_report(Y_test, model_random_forest_predictions))
    confusion_matrix(Y_test, model_random_forest_predictions)

              precision    recall  f1-score   support

           0       0.98      0.98      0.98        43
           1       0.99      0.99      0.99        71

    accuracy                           0.98       114
   macro avg       0.98      0.98      0.98       114
weighted avg       0.98      0.98      0.98       114

array([[42,  1],
       [ 1, 70]])
```

```
Evaluation Metrics For Each Classifier

Decision Tree:
  Accuracy: 0.9649122807017544
  Precisin: 0.971830985915493
  Recall: 0.971830985915493
  F1 Score: 0.971830985915493
  Confusion Matrix: [[41  2]
 [ 2 69]]


Random Forest:
  Accuracy: 0.9824561403508771
  Precisin: 0.9859154929577465
  Recall: 0.9859154929577465
  F1 Score: 0.9859154929577465
  Confusion Matrix: [[42  1]
 [ 1 70]]


Adaboost:
  Accuracy: 0.9736842105263158
  Precisin: 0.9722222222222222
  Recall: 0.9859154929577465
  F1 Score: 0.9790209790209791
  Confusion Matrix: [[41  2]
 [ 1 70]]


The best classifer based on F1 Sccore (Class 1) is Random Forest with score of 0.9859
```