נמשיך את התרגיל שבו בנינו עץ החלטה לומד, ונרחיב עץ החלטה זה ל- Random Forest.
הערה: גם מי שלא מסיים את התרגיל, ממומלץ לקפוץ לסעיף 6 בו אנו מכירים ומשחקים עם מימוש קיים של Random Forest ב scikit-learn
דוגמה למימוש אפשר למצוא גם בלינק הבא:
http://machinelearningmastery.com/implement-random-forest-scratch-python/

כדי להפוך את עץ ההחלטה ל- Random Forests נעשה את השלבים הבאים:

1. בשביל ליישם bootstrapping נוסיף פונקציה בשם "create_subsample" שבוחרת מתוך dataset שלם שיש בו n שורות subsample של ה- data ע"י בחירה של 'n שורות של הdata (בשביל הפשטות ניקח n='n, בהמשך אפשר להגדיר n'=a*n ולבחור a<1). נזכיר שניתן לבחור באקראיות מספרים בעזרת הפונקציה randrange של פייטון, או randint של numpy:
https://docs.python.org/3/library/random.html#random.randrange
https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.random.randint.html

2. נייצר גרסה נוספת לפונקציה שמוצאת את ה- split הטוב ביותר שבה במקום לבדוק את כל הפיצ'רים האפשריים - בודקת רק חלק מהם שהיא בוחרת בצורה רנדומלית, מקובל לבחור שורש של מספר הפיצ'רים.
3. נייצר פונקציה בשם random_forests שקוראת m פעמים (לדוג' 5) לפונקציה שבונה את עץ ההחלטה - ומחזירה m עצים, כלומר m אובייקטים של root. אובייקט לכל עץ.
4. נייצר פונקציה בשם bagging_predict שמקבלת את העצים שייצרנו ווקטור קלט יחיד (סט יחיד של פיצ'רים) ומייצרת prediction ע"י בחירת הרוב מהprediction-ים של כל עץ יחיד.
5. למתקדמים: נבדוק את האלגוריתם ע"י K-Fold Cross Validation. נחלק את המידע לK חלקים שווים (לדוג' 4) נוציא כל פעם חלק אחד, נאמן את העץ על K-1 החלקים שנותרו ונבדוק את החיזוי עבור החלק שהוצאנו.

בסעיף זה נשחק עם מימוש קיים של Random Forest:

6. מימוש של Random Forest ניתן למצוא ספריית scikit-learn:
http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
אנו ניקח אותו ונריץ אותו על ה-data שלנו, ננסה גם להבין ולשחק עם הפרמטרים השונים ולראות איך זה משפיע על התוצאות