ML and BD in Healthcare, Homework #5

1. Answer homework questions below
2. When ready, submit your answers online as instructed

# Answer the questions

**Question 1:** (1 pt)

Machine learning models can improve clinical decision making because

1. They can be easily programmed in many programming languages
2. They do not require iterative computations
3. They use decision trees
4. They identify data-driven, objective decision logic

**Question 2:** (1 pt)

By limiting the depth of the decision tree we make sure that

1. It does not overfit the data
2. It uses as many features as possible
3. It provides the best fit into the data
4. It can be used with random forest models

**Question 3:** (1 pt)

One of the main advantages of decision tree model is:

1. Stability
2. Using linear regression
3. Representing interpretable decision logic
4. Not overfitting the data

**Question 4:** (1 pt)

Forest models are built by

1. Averaging linear regressions
2. Averaging decision trees
3. Averaging randomly-chosen leaves from the same large decision tree
4. Selecting the largest tree from a randomly generated set of decision trees

# Solve data problems

In this part, we will take a bit more time to explore the Wisconsin Breast Cancer dataset, described in the class. Load the dataset using the file provided with this homework[[1]](#footnote-1). Use ‘diagnosis’ as your target variable Y to be predicted (you can replace Malignant by 1, and Benign by 0, thus predicting cancer malignancy). Make sure you remove ‘diagnosis’ and ‘id’ from the feature variables X.

Note that there are many types of model accuracy metrics, they all can be found in sklearn.metrics.

**Question 5:** (5 pts)

**Model A**

Using LogisticRegression model and our greedy model selection algorithm that we have used previously (see the insert below), find the **best 3-feature model**. Use sklearn.metrics.accuracy\_score[[2]](#footnote-2) to select the model with the highest accuracy. Which one of the features below appears in this model?

1. Find the best one-feature model (try all one-feature models, and select the one with the lowest error e). This is our best feature F1.
2. Using F1 from the first step, try adding one more feature to it (from all features you have left), to find the best 2-feature model (F1, F2)
3. Similarly, keep adding more features: F3, F4, F5 – to the features from the previous step

Greedy stepwise feature selection algorithm

1. compactness\_worst
2. fractal\_dimension\_se
3. smoothness\_se
4. concave points\_mean
5. perimeter\_worst

*Checkpoint: Model* ***accuracy*** *score should be close to 0.95606*

**Question 6:** (3 pts)

**Model B**

Predicting cancer malignancy, one would definitely want to avoid false negatives: malignant cancer cases erroneously classified by the model as benign.

*Recall* (or *Sensitivity*, <https://en.wikipedia.org/wiki/Precision_and_recall>) is another quality metric, which determines how well a model can avoid false negatives. So let’s use the above greedy selection, and change model selection metric from accuracy to recall (sklearn.metrics.recall\_score).

Rerun the greedy code with this metric to select the **2-feature** model with the highest recall.

How many features in this new model are the same as in the model A above?

1. None
2. One
3. Two
4. Three

*Checkpoint: Model* ***recall*** *score should be close to 0.919811*

*Another interesting checkpoint – try running this algorithm to select the best 3-, 4-, 5-feature model, and you will see, that the best recall value is not changing; and going higher will definitely result in an overfit. So we have an interesting case when 2 features get all the job done.*

**Question 7:** (6 pts)

**Model C**

Regression is great, but we want to find a very simple model any physician can use. So let’s consider building a decision tree model. To develop something very interpretable that can be used by humans, let’s also limit the number of decision tree leaves to **3** by using max\_leaf\_nodes parameter in DecisionTreeClassifier()[[3]](#footnote-3).

As always, train the model on the entire dataset. Make a plot of the resulting model tree:

fig = plt.figure(figsize=(15,10))

\_ = tree.plot\_tree(model,

feature\_names=list(X.columns),

class\_names=['B','M'],

rounded = True,

fontsize = 12,

filled=True)

We have a patient with the following features

radius\_worst<12   
radius\_mean>9  
concave\_points\_worst<0.1

Looking at this tree plot only, answer the following question:

What does this model tell us about the patient cancer classification?

1. Benign
2. Malignant
3. Equally probable to be Benign or Malignant

*Checkpoint: Model accuracy score should be close to 0.94024*

**Question 8:** (4 pts)

**Model D**

Now, let’s build a really small random forest, with only **10 trees**, setting **tree maximum depth[[4]](#footnote-4)** to **3**. Compute the accuracy score of this forest (Model D) to the models A and C above. Which of the three models produced the most accurate result?

1. A
2. C
3. D
4. All three models have the same accuracy

*Note: Random forest uses random seeding to create its trees, so it may return slightly different results with each model run. Run it a few times comparing to the other models, and report most frequent result.*

1. This dataset has many versions, so please use the file provided [↑](#footnote-ref-1)
2. For instance, you can do this:

   from sklearn import metrics as mtr

   model.fit(X, Y)

   accuracy = mtr.accuracy\_score(Y, model.predict(xWithCurrentFeatures)) [↑](#footnote-ref-2)
3. Do not use any other parameters [↑](#footnote-ref-3)
4. Set these parameter values in RandomForestClassifier(); [↑](#footnote-ref-4)