CM146, Fall 2019
Problem Set 1: Decision trees and k-Nearest Neighbors
Due Oct. 21, 2019 at 11:59 PM

Submission instructions

- Submit your written answer on the Gradescope website as a PDF file. Submit your Python code on CCLE.
- If you submit a handwritten solution, please use a high-quality scanner app. If you typeset your solution, please use LaTeX .

1 Splitting Heuristic for Decision Trees [20 pts]

The ID3 algorithm iteratively grows a decision tree from the root downwards. On each iteration, the algorithm replaces one leaf node with an internal node that splits the data based on one decision attribute (or feature). In particular, the ID3 algorithm chooses the split that reduces the entropy the most, but there are other choices. For example, since our goal in the end is to have the lowest error, why not instead choose the split that reduces error the most? In this problem, we will explore one reason why reducing entropy is a better criterion.

Consider the following simple setting. Let us suppose each example is described by n boolean features: $X = \langle X_1, \ldots, X_n \rangle$, where $X_i \in \{0, 1\}$, and where $n \geq 4$. Furthermore, the target function to be learned is $f : X \to Y$, where $Y = X_1 \vee X_2 \vee X_3$. That is, $Y = 1$ if $X_1 = 1$ or $X_2 = 1$ or $X_3 = 1$, and $Y = 0$ otherwise. Suppose that your training data contains all of the 2^n possible examples, each labeled by f. For example, when $n = 4$, the data set would be

X_1	X_2	X_3	X_4	Y
0	0	0	0	0
1	0	0	0	1
0	1	0	0	1
1	1	0	0	1
0	0	1	0	1
1	0	1	0	1
0	1	1	0	1
1	1	1	0	1

X_1	X_2	X_3	X_4	Y
0	0	0	1	0
1	0	0	1	1
0	1	0	1	1
1	1	0	1	1
0	0	1	1	1
1	0	1	1	1
0	1	1	1	1
1	1	1	1	1

(a) **(5 pts)** The 1-leaf decision tree does not split the data at all; to be more precise, this tree assigns the majority vote which is $Y = 1$ to all possible choices of the input X in the table above. How many mistakes does the best 1-leaf decision tree make over the 2^n training examples? Make sure you also answer the general case for any value of $n \geq 4$.

Parts of this assignment are adapted from course material by Andrea Danyluk (Williams), Tom Mitchell and Maria-Florina Balcan (CMU), Stuart Russell (UC Berkeley), Carlos Guestrin (UW) and Jessica Wu (Harvey Mudd).

(b) **(5 pts)** Is there one single split that reduces the number of mistakes by at least one? (That is, is there a feature X_i that can be create a decision tree with fewer mistakes than your answer to part (a)?) Explain your answer.

(c) **(2 pts)** In Table 1, what is the entropy of the output label Y on the entire data without any split?

(d) **(3 pts)** In Table 1, you know that variable X_4 is unimportant for the label Y. What is the conditional entropy of Y given the split on X_4 in Table 1? What is the gain in entropy for this split?

(e) **(5 pts)** Is there a split that reduces the entropy of the output Y in Table 1 by a non-zero amount? If so, what is it, and what is the resulting conditional entropy of Y given this split? Notice, unlike (b) which asks about reducing accuracy, here, we are asking about reducing entropy.

2 Entropy and Information [9 pts]

The entropy of a Bernoulli (Boolean 0/1) random variable X with $p(X = 1) = q$ is given by

$$B(q) = -q \log q - (1 - q) \log(1 - q).$$

Notice, this form looks very similar to your quiz 0 question 10 on CCLE. Suppose that a set S of examples contains p positive examples and n negative examples. The entropy of S is defined as $H(S) = B\left(\frac{p}{p+n}\right)$.

(a) **(3 pts)** What is the value of q that maximizes $B(q)$? You need to show some equations, do not solve by simply plotting the function $B(q)$.

(b) **(6 pts)** This problem is a generalization of question 1d, where you observe that splitting the data into 2 groups based on the unimportant X_4 yields the same fraction of $Y = 0$ and $Y = 1$ for each group. Now, based on an attribute X_j, we split our examples into k disjoint subsets S_k, with p_k positive and n_k negative examples in each. If the ratio $\frac{p_k}{p_k+n_k}$ is the same for all k, show that the information gain of this attribute is 0.

3 k-Nearest Neighbors and Cross-validation [15 pts]

In the following questions you will consider a k-nearest neighbor classifier using Euclidean distance metric on a binary classification task. We assign the class of the test point to be the class of the majority of the k nearest neighbors. Note that a point can be its own neighbor.

(a) **(3 pts)** What value of k minimizes the training set error for this dataset? What is the resulting training error? Hint: We are not asking about classifying an unknown sample, and for a given training data point, you know its label.

(b) **(3 pts)** Why might using too large values k be bad in this dataset? Why might too small values of k also be bad?

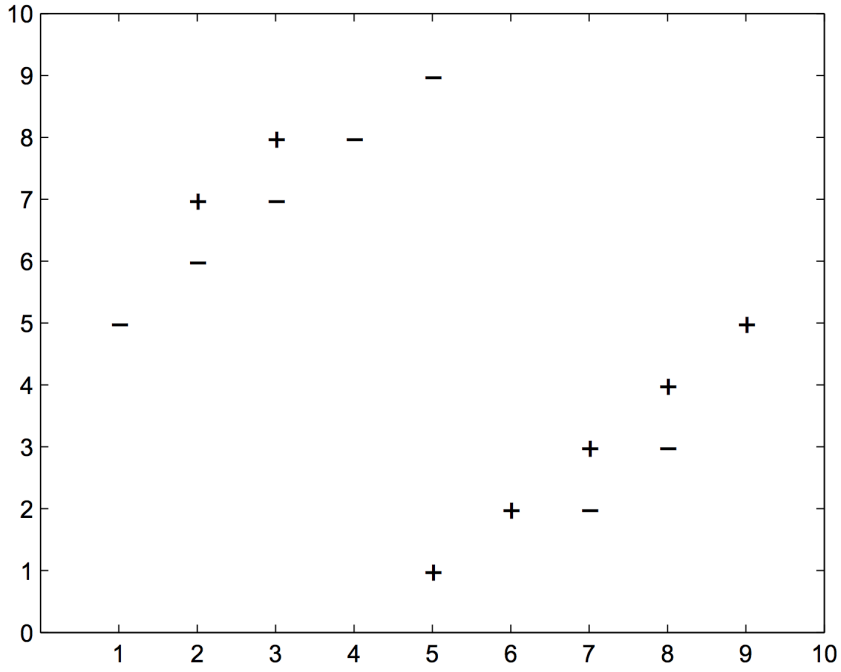

Figure 1: Dataset for KNN binary classification task.

(c) **(9 pts)** What value of k minimizes leave-one-out cross-validation error for this dataset? What is the resulting error? I strongly recommend you code this problem, and not compute Euclidean distance by hand. Do not submit code for this problem, submit only answer.

4 Programming exercise : Applying decision trees and k-nearest neighbors [56 pts]

Submission instructions

- Submit answers and plots for this problem together with the answers for question 1-3 on Gradescope. Submit your code for this problem on CCLE.

Introduction[1]

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

[1]This assignment is adapted from the Kaggle Titanic competition, available at https://www.kaggle.com/c/titanic. Some parts of the problem are copied verbatim from Kaggle.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

In this problem, we ask you to complete the analysis of what sorts of people were likely to survive. In particular, we ask you to apply the tools of machine learning to predict which passengers survived the tragedy.

Getting started

Code and data https://ccle.ucla.edu/course/view/19F-COMSCIM146-1

- Code : `titanic.py`
- Data : `titanic_train.csv`

Functions to use

- Decision Tree Classifier:
  http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
- K-Nearest Neighbor Classifier:
  http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
- Cross-Validation:
  https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
- Metrics:
  http://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

---

Download the code and data sets from CCLE website. For more information on the data set, see the Kaggle description: https://www.kaggle.com/c/titanic/data. The provided data sets are modified versions of the data available from Kaggle.[2]

Note that any portions of the code that you must modify have been indicated with `TODO`. Do not change any code outside of these blocks.

4.1 Visualization [5 pts]

One of the first things to do before trying any formal machine learning technique is to dive into the data. This can include looking for funny values in the data, looking for outliers, looking at the range of feature values, what features seem important, etc.

(a) **(5 pts)** Run the code (`titanic.py`) to make histograms for each feature, separating the examples by class (e.g. survival). This produces seven plots, one for each feature, and each plot should have two overlapping histograms, with the color of the histogram indicating the class. For each feature, what trends do you observe in the data?

[2]Passengers with missing values for any feature have been removed. Also, the categorical feature `Sex` has been mapped to `{'female': 0, 'male': 1}` and `Embarked` to `{'C': 0, 'Q': 1, 'S': 2}`.

4.2 Evaluation [51 pts]

Now, let us use `scikit-learn` to train a `DecisionTreeClassifier` and `KNeighborsClassifier` on the data. Using the predictive capabilities of the `scikit-learn` package is very simple. In fact, it can be carried out in three simple steps: initializing the model, fitting it to the training data, and predicting new values.[3]

(b) **(5 pts)** Before trying out any classifier, it is often useful to establish a *baseline*. We have implemented one simple baseline classifier, `MajorityVoteClassifier`, that always predicts the majority class from the training set. Read through the `MajorityVoteClassifier` and its usage and make sure you understand how it works.

Your goal is to implement and evaluate another baseline classifier, `RandomClassifier`, that predicts a target class according to the distribution of classes in the training data set. For example, if 60% of the examples in the training set have `Survived = 0` and 40% have `Survived = 1`, then, when applied to a test set, `RandomClassifier` should randomly predict 60% of the examples as `Survived = 0` and 40% as `Survived = 1`.

Implement the missing portions of `RandomClassifier` according to the provided specifications. Then train your `RandomClassifier` on the entire training data set, and evaluate its training error. If you implemented everything correctly, you should have an error about 0.40 to 0.50 range.

(c) **(5 pts)** Now that we have a baseline, train and evaluate a `DecisionTreeClassifier` (using the class from `scikit-learn` and referring to the documentation as needed). Make sure you initialize your classifier with the appropriate parameters; in particular, use the 'entropy' criterion discussed in class. What is the training error of this classifier?

(d) **(5 pts)** Similar to the previous question, train and evaluate a `KNeighborsClassifier` (using the class from `scikit-learn` and referring to the documentation as needed). Use k=3, 5 and 7 as the number of neighbors and report the training error of this classifier.

(e) **(9 pts)** So far, we have looked only at training error, but as we learned in class, training error is a poor metric for evaluating classifiers. Let us use cross-validation instead.

Implement the missing portions of `error(...)` according to the provided specifications. You may find it helpful to use `train_test_split(...)` from `scikit-learn`. To ensure that we always get the same splits across different runs (and thus can compare the classifier results), set the `random_state` parameter to be the trial number.

Next, use your `error(...)` function to evaluate the training error and (cross-validation) test error of each of your four models (for the `KNeighborsClassifier`, use k=5). To do this, manually create a random 80/20 split of the training data, train each model on the 80% fraction, evaluate the error on either the 80% or the 20% fraction, and repeat this 100 times to get an average result. What are the average training and test error of each of your classifiers on the Titanic data set?

(f) **(9 pts)** One way to find out the best value of k for `KNeighborsClassifier` is n-fold cross validation. Find out the best value of k using 10-fold cross validation. You may find the

[3] Almost all of the model techniques in `scikit-learn` share a few common named functions, once they are initialized. You can always find out more about them in the documentation for each model. These are `some_model_name.fit(...)`, `some_model_name.predict(...)`, and `some_model_name.score(...)`.

`cross_val_score(...)` from `scikit-learn` helpful. Run 10-fold cross validation for all odd numbers ranging from 1 to 50 as the number of neighbors. Unlike part (e), the function `cross_val_score(...)` will automatically split the data for you, so that you do not need to manually create the partitions. Plot the validation error against the number of neighbors, k. Include this plot in your answer, and provide a 1-2 sentence description of your observations. What is the best value of k?

(g) **(9 pts)** One problem with decision trees is that they can *overfit* to training data, yielding complex classifiers that do not generalize well to new data. Let us see whether this is the case for the Titanic data.

One way to prevent decision trees from overfitting is to limit their depth. Repeat your cross-validation experiments but for increasing depth limits, specifically, $1, 2, \ldots, 20$. Then plot the average training error and test error against the depth limit. Include this plot in your answer, making sure to label all axes and include a legend for your classifiers. What is the best depth limit to use for this data? Do you see overfitting? Justify your answers using the plot.

(h) **(9 pts)** Another useful tool for evaluating classifiers is *learning curves*, which show how classifier performance (e.g. error) relates to experience (e.g. amount of training data). For this experiment, first generate a random 90/10 split of the training data and do the following experiments considering the 90% fraction as training and 10% for testing.

Run experiments for the decision tree and k-nearest neighbors classifier with the best depth limit and k value you found above. This time, vary the amount of training data by starting with splits of 0.10 (10% of the data from 90% fraction) and working up to full size 1.00 (100% of the data from 90% fraction) in increments of 0.10. Then plot the decision tree and k-nearest neighbors training and test error against the amount of training data. Include this plot in your answer, and provide a 1-2 sentence description of your observations.