ECE M146 Homework 3 Solution
Introduction to Machine Learning Monday, April 15, 2019
Instructor: Lara Dolecek
TA: Zehui (Alex) Chen, Ruiyi (John) Wu

Note: All log in this homework is base 2.

1. You are trying to choose between two restaurants (sample 9 and sample 10) to eat at. To do this, you will use a decision tree that will be trained based on your past experiences (sample 1-8). The features for each restaurants and your judgment on the goodness of sample 1-8 are summarized by the following chart.

Sample #	HasOutdoorSeating	HasBar	IsClean	HasGoodAtmosphere	IsGoodRestaurant
1	0	0	1	1	1
2	1	0	0	0	0
3	0	1	1	1	1
4	0	0	0	0	0
5	1	1	0	0	0
6	1	0	1	0	1
7	1	0	0	1	1
8	0	0	1	1	1
9	0	1	0	1	?
10	1	1	1	1	?

(a) What is the entropy of IsGoodRestaurant, i.e., $H(IsGoodRestaurant)$?
Solution: Define the binary entropy function as follows:

$$H_b(p) = -p\log(p) - (1-p)\log(1-p).$$

$$H(IsGoodRestaurant) = H_b(\frac{3}{8}) = -(\frac{5}{8}\log(\frac{5}{8}) + \frac{3}{8}\log(\frac{3}{8})) \approx 0.9544$$

(b) Calculate the conditional entropy of IsGoodRestaurant conditioning on HasOutdoorSeating. To do this, first compute $H(IsGoodRestaurant|HasOutdoorSeating = 0)$ and $H(IsGoodRestaurant|HasOutdoorSeating = 1)$, then weigh each term by the probabilities $P(HasOutdoorSeating = 0)$ and $P(HasOutdoorSeating = 1$, respectively. Namely, calculate the following:

$$\begin{aligned}&H(IsGoodRestaurant|HasOutdoorSeating)\\ =&P(HasOutdoorSeating = 0)H(IsGoodRestaurant|HasOutdoorSeating = 0)\\ +&P(HasOutdoorSeating = 1)H(IsGoodRestaurant|HasOutdoorSeating = 1).\end{aligned}$$

Solution: Use the given equation, we get:

$$\begin{aligned}&H(IsGoodRestaurant|HasOutdoorSeating)\\ =&\frac{1}{2}H_b(\frac{3}{4}) + \frac{1}{2}H_b(\frac{1}{2}) \approx \frac{0.81127}{2} + \frac{1}{2} = 0.905635.\end{aligned}$$

(c) Similarly, calculate

$$H(IsGoodRestaurant|X), \text{for } X \in \{HasBar, IsClean, HasGoodAtmosphere\},$$

i.e., the conditional entropy of IsGoodRestaurant conditioning on the other three features.

Solution:

$$H(IsGoodRestaurant|HasBar) = \frac{2}{8}H_b(\frac{1}{2}) + \frac{6}{8}H_b(\frac{2}{3}) = \frac{2}{8} + 0.91829\frac{3}{4} \approx 0.9387.$$

$$H(IsGoodRestaurant|IsClean) = \frac{1}{2}H_b(\frac{3}{4}) + \frac{1}{2}H_b(1) \approx \frac{0.81127}{2} + 0 = 0.405635.$$

$$H(IsGoodRestaurant|HasGoodAtmosphere) = 0 + \frac{1}{2}H_b(\frac{1}{4}) \approx \frac{0.81127}{2} = 0.405635.$$

(d) Calculate the information gain:

$$I(IsGoodRestaurant; X) = H(IsGoodRestaurant) - H(IsGoodRestaurant|X),$$

for

$$X \in \{HasOutdoorSeating, HasBar, IsClean, HasGoodAtmosphere\}.$$

Solution:

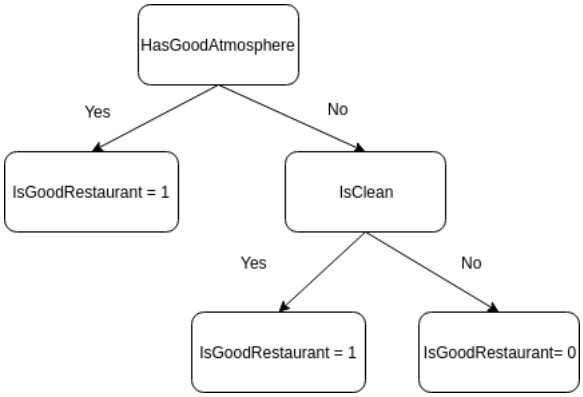
$$I(IsGoodRestaurant; HasOutdoorSeating) = 0.9544 - 0.9056 = 0.0488;$$
$$I(IsGoodRestaurant; HasBar) = 0.9544 - 0.9387 = 0.0157;$$
$$I(IsGoodRestaurant; IsClean) = 0.9544 - 0.4056 = 0.5488;$$
$$I(IsGoodRestaurant; HasGoodAtmosphere) = 0.9544 - 0.4056 = 0.5488.$$

(e) Based on the information gain, determine the first attribute to split on.

Solution: We choose HasGoodAtmosphere or IsClean which has the largest information gain.

(f) Make the full decision tree. After each split, treat the sets of samples with $X = 0$ and $X = 1$ as two separate sets and redo (b), (c), (d) and (e) on each of them. X is the feature for previous split and is thus excluded from the available features which can be split on next. Terminate splitting if after the previous split, the entropy of IsGoodRetaurant in the current set is 0. For example, if we choose HasGoodAtmosphere as our first feature to split, we get $H(IsGoodRetaurant|HasGoodAtmosphere = 1) = 0$. We thus stop splitting the tree in this branch. Draw the tree and indicate the split at each node.

Solution: Below show the decision tree if we choose HasGoodAtmosphere as the first splitting feature. If one choose IsClean as the first splitting feature, the place of HasGoodAtmosphere and IsClean will be interchanged.



After the first split, $H(IsGoodRetaurant|HasGoodAtmosphere = 1) = 0$ so the tree stops growing on that branch. We are left with the samples that have $HasGoodAtmosphere = 0$ which is summarized in the following table.

Sample #	HasOutdoorSeating	HasBar	IsClean	IsGoodRestaurant
2	1	0	0	0
4	0	0	0	0
5	1	1	0	0
6	1	0	1	1

The entropy for the current set is $H_b(\frac{1}{4}) = 0.8113$. The information gain by splitting on HasOutdoorSeating, HasBar and IsClean are $0.8113 - 0.6887 = 0.1226$, $0.8113 - 0.6887 = 0.1226$ and $0.8113 - 0 = 0.8113$, respectively. We then split using IsClean. After this split, every leaf is pure, i.e., IsGoodRetaurant is either 0 or 1. Therefore, we stop growing the tree.

(g) Now, determine if restaurants 9 and 10 are good or not.

Solution:

Restaurant 9: Good

Restaurant 10: Good

2. Consider again the linear regression problem in which we want to "weigh" different training instances differently because some of the instances are more important than others. Specifically, suppose we want to minimize

$$J(w_0, w_1) = \sum_{n=1}^{N} \alpha_n (w_0 + w_1 x_{n,1} - y_n)^2. \tag{1}$$

Here $\alpha_n > 0$. In HW2, you worked out the gradient of J with respect to $[w_0, w_1]$. Prove that Equation (1) has a global optimal solution. Hint: show the Hessian matrix for the function J is positive semi-definite. A symmetric real matrix M is said to be positive semi-definite if the scalar $z^T M z$ is nonnegative for every non-zero column vector z.
Solution: We already know:

$$\nabla_{\boldsymbol{w}} J(w_0, w_1) = \begin{bmatrix} \frac{\partial J(w_0,w_1)}{\partial w_0} \\ \frac{\partial J(w_0,w_1)}{\partial w_1} \end{bmatrix} = \begin{bmatrix} \sum_{n=1}^{N} 2\alpha_n (w_0 + w_1 x_{n,1} - y_n) \\ \sum_{n=1}^{N} 2\alpha_n x_{n,1} (w_0 + w_1 x_{n,1} - y_n) \end{bmatrix}.$$

We find the Hessian matrix:

$$H = \begin{bmatrix} \frac{\partial^2 J(w_0,w_1)}{\partial^2 w_0} & \frac{\partial^2 J(w_0,w_1)}{\partial w_0 \partial w_1} \\ \frac{\partial^2 J(w_0,w_1)}{\partial w_1 \partial w_0} & \frac{\partial^2 J(w_0,w_1)}{\partial^2 w_1} \end{bmatrix} = \begin{bmatrix} \sum_{n=1}^{N} 2\alpha_n & \sum_{n=1}^{N} 2\alpha_n x_{n,1} \\ \sum_{n=1}^{N} 2\alpha_n x_{n,1} & \sum_{n=1}^{N} 2\alpha_n x_{n,1}^2 \end{bmatrix} = 2 \sum_{n=1}^{N} z_n z_n^T,$$

where $z_n = [\sqrt{\alpha_n}, \sqrt{\alpha_n} x_{n,1}]^T$.
Next we show that $\forall \beta \in \mathbb{R}^2, \beta^T H \beta = 2 \sum_{n=1}^{N} \beta^T z_n z_n^T \beta = 2 \sum_{n=1}^{N} (\beta^T z_n)^2 \geq 0$. We therefore conclude H is positive semi-definite and Equation (1) has a global optimal solution.

3. When training decision trees for classification, we generally use entropy or gini index to help determine good splits. These functions are examples of impurity measures. We can formally define impurity measures as follows.

   Assume we are performing binary classification. Let V be a set of data points and let $\{y_i \in \{+1, -1\} \ , \ \forall i \in V\}$ be the set of labels. Let $V_1 \subseteq V$. We define p and q as follows:

$$p(V_1, V) = \frac{|V_1|}{|V|} \quad q(V_1) = \frac{|\{i : i \in V_1, y_i = 1\}|}{|V_1|}$$

   where $|\cdot|$ is the cardinality of a set.
   Let $i(q(V))$ measure the impurity of a set V. Two desired properties of $i(q(V))$ are:

   - $i(q(V)) = 0$ when the set has only one class
   - $i(q(V))$ reaches its maximum value for sets where the two classes are perfectly balanced.

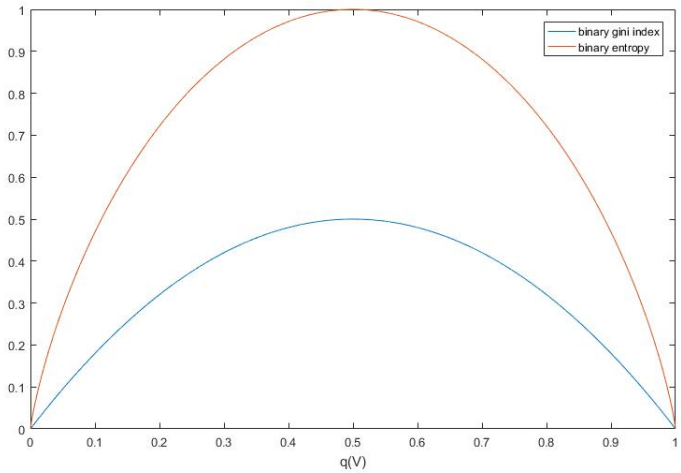
   When a split is performed of a set V, the result is two sets $V_1 \cup V_2 = V$ such that $V_1 \cap V_2 = \emptyset$. The information gain of this split is defined as

$$I(V_1, V_2, V) = i(q(V) - (p(V_1, V)i(q(V_1)) + p(V_2, V)i(q(V_2))).$$

   Using the previous notation, we define the binary gini Index and binary entropy as:

$$gini(q(V)) = 2q(V)(1 - q(V))$$
$$H(q(V)) = -(q(V)\log(q(V)) + (1 - q(V))\log(1 - q(V)))$$

   (a) Like binary entropy, gini index is an alternative impurity measure that is commonly used. Plot both the Gini index and the binary entropy function for $q(V)$ from 0 to 1. Comment on their similarities.
   Solution:



   They both attains the maximum value at $q(V) = \frac{1}{2}$. They are both 0 when $q(V) = 0$ or $q(V)1$.

(b) Show that if $i(q(V))$ is concave in $q(V)$ then $I(V_1, V_2, V) \geq 0 \; \forall \; V_1 \cup V_2 = V, V_1 \cap V_2 = \emptyset$. This means that every split does not lose any information. Recall that a function is concave if $\forall \lambda \in [0, 1]$ and $\forall q_1, q_2$ then $i(\lambda q_1 + (1 - \lambda) q_2) \geq \lambda i(q_1) + (1 - \lambda) i(q_2)$.

Solution: For simplicity, let $q = q(V), q_1 = q(V_1), q_2 = q(V_2), p_1 = p(V_1, V), p_2 = p(V_2, V)$.

$$\begin{aligned} q &= \frac{|\{i : i \in V, y_i = 1\}|}{|V|} \\ &= \frac{|\{i : i \in V_1, y_i = 1\}|}{|V|} + \frac{|\{i : i \in V_2, y_i = 1\}|}{|V|} \\ &= \frac{|V_1|}{|V|} \frac{|\{i : i \in V_1, y_i = 1\}|}{|V_1|} + \frac{|V_2|}{|V|} \frac{|\{i : i \in V_2, y_i = 1\}|}{|V_2|} \\ &= p_1 * q_1 + p_2 * q_2 \end{aligned}$$

Due to concavity, we have

$$i(q) = i(p_1 q_1 + p_2 q_2) \geq p_1 i(q_1) + p_2 i(q_2).$$

Hence, $I(V_1, V_2, V) \geq 0$

(c) Show that binary entropy is concave. Hint: Show that the 2nd derivative is always non-positive.

Solution:

$$\begin{aligned} \frac{dH(q)}{dq} &= \log(\frac{1}{q} - 1) \\ \frac{d^2 H(q)}{dq^2} &= -\frac{1}{q - q^2} \end{aligned}$$

Since $q \in [0, 1]$, then $q \geq q^2$. As such, the second derivative is always non-positive.

(d) Show that the binary Gini index is concave.

Solution: Similarly,

$$\begin{aligned} \frac{dgini(q)}{dq} &= 2(1 - 2q) \\ \frac{d^2 gini(q)}{dq^2} &= -4 \end{aligned}$$

4. In this section, 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.

   Consider the following two datasets:

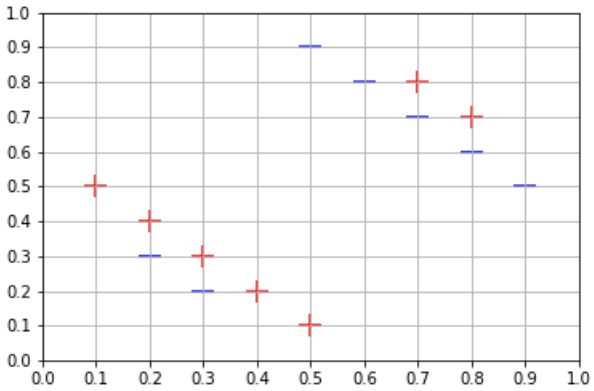


Table 1: KNN Example 1

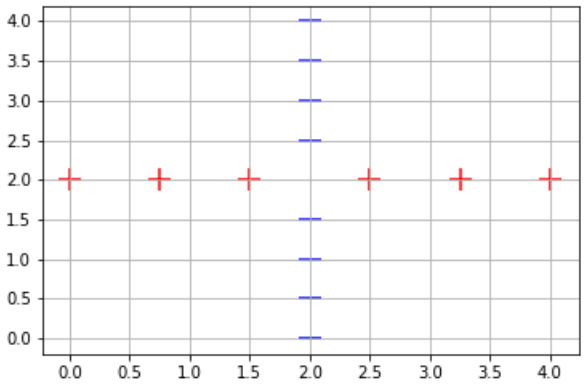


Table 2: KNN Example 2

   (a) For $k \in \{1, 3, 5, 7\}$, what values of k minimize leave-one-out cross-validation error for each dataset? What is the resulting validation error?

   Solution:

   For the first example, the best k are 5 and 7 with a validation error of $\frac{4}{14}$. If $k = 1$, with leave-one-out cross-validation, 10 points (despite the four points at $(0.1, 0.5), (0.5, 0.1), (0.9, 0.5)\text{and}(0.5, 0.9)$) are classified erroneously, resulting in a validation error of $\frac{10}{14}$. If $k = 3$, in addition to the 4 points that are seemingly out of place, the two points at $(0.3, 0.3)$ and $(0.7, 0.7)$ are also classified erroneously, resulting in a validation error of $\frac{6}{14}$.

For the second example, the best k are 1 with a validation error of $\frac{2}{14}$. The two misclassified points are $(1.5, 2)$ and $(2.5, 2)$. For $k = 3, 5$ and 7, there are at least two additional points that are classified erroneously. For $k = 3$, these two points are $(2, 1.5)$ and $(2, 2.5)$. For $k = 5$, these two points are $(1.75, 2)$ and $(1.75, 2)$. For $k = 7$, these two points are $(0, 2)$ and $(4, 2)$.

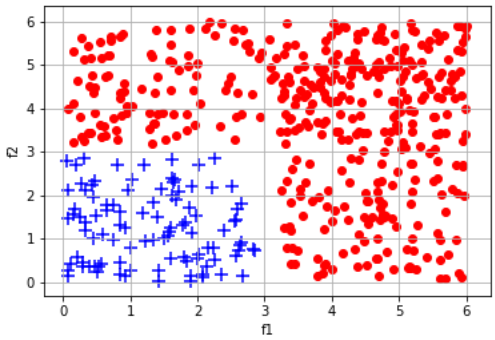
(b) In general, what are the drawbacks of using too big a k? What about too small a k? To see this, calculate the leave-one-out cross-validation error for the dataset in Figure 1 using $k = 1$ and $k = 13$.

Solution:

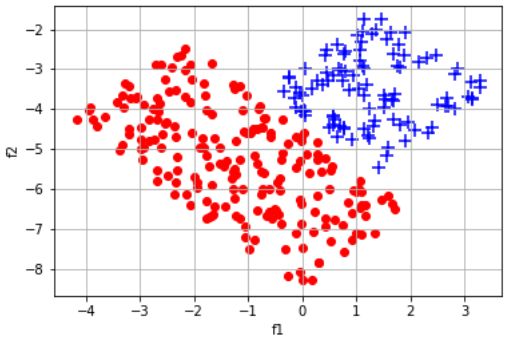
In general, too big causes underfitting since the largest class would dominate and too small causes overfitting. For Example 1, the validation error for $k = 1$ is $\frac{10}{14}$. The validation error for $k = 13$ is $\frac{14}{14}$ which is really bad.

Note although this statement holds true in general, for certain cases, e.g, our synthetic example 2, small k or large k may be preferred. Moreover, the nearest neighbor classifier, i.e., $k = 1$, is one commonly used calssifeir.

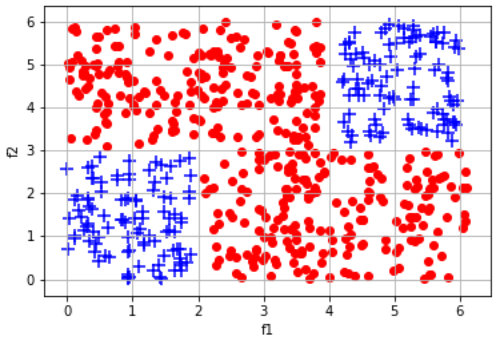
5. Consider the following plots:



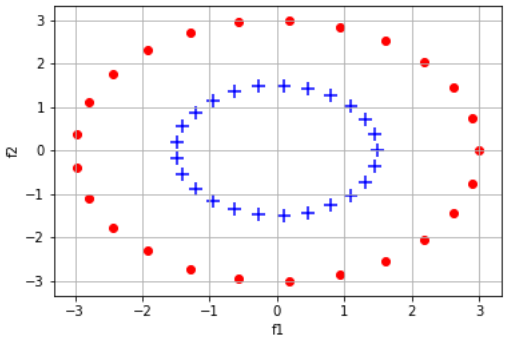
(1) Decision Tree Example 1



(2) Decision Tree Example 2



(3) Decision Tree Example 3



(4) Decision Tree Example 4

Assume that you are using a binary split decision tree to classify the points. At each node, you may ask questions like: "Is $f_1 \geq 3$?" or "Is $f_2 \leq 1.8$?". Here, $f1$ and $f2$ are the variables on the horizontal axis and vertical axis of the examples, respectively.

(a) Which examples can be fully separated using a depth 2 decision tree?

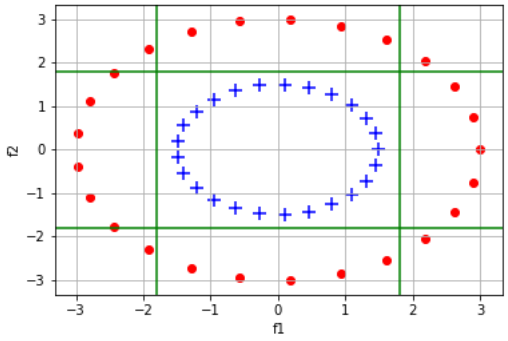
Solution: Examples 1 and 3. For example 1, the first question to ask is "Is $f1 \geq 3$?". If "No", ask the second question "Is $f2 \geq 3$?". For example 2, ask the first question "Is $f2 \geq 3$?" . If "No", ask the second question "Is $f1 \geq 2$?". If "Yes", ask the second question "Is $f1 \geq 4$?".

(b) Which example would have the most complex decision tree in terms of the number of splits? Explain why.

Solution: Example 2 because the points are separated by a non-axis aligned hyperplane which would require a lot of binary splits to model. Decision tree works well only when the decision boundaries are axis-aligned.

(c) If you used a depth 4 decision tree, is one more example now separable? If so, show how you can separate it using a depth 4 decision tree.

Solution: Example 4. You may ask these four questions: "Is $f1 \geq -1.8$?", "Is $f1 \leq 1.8$?", "Is $f1 \geq -1.8$?" and "Is $f1 \leq 1.8$?". If all answers are "Yes", then we decide the class is the blue one. See illustration in the next figure.


f2
f1
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3

6. On April 15, 1912, the largest passenger liner ever made collided with an iceberg during her maiden voyage. When the Titanic sank it killed 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships. One of the reasons that the shipwreck resulted in 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.

   In this section, you will use decision trees and K-nearest neighbors to predict whether a person survives or not. You will be using the data set provided in *Titanic.mat* which contains feature values and labels for both training and testing data. The data is normalized so that all feature values are between 0 and 1. If you are interested in the actual meaning of each feature, we provide the original data in *titanic.csv* where we use the mapping (Female:0, Male:1). (For python, feel free to use the following libraries: math, collections, numpy, matplotlib and sklearn. The corresponding data set are *dataTesting_X.csv*,*dataTesting_Y.csv*, *dataTraining_X.csv* and *dataTraining_Y.csv*.)

   (a) Before starting, it is useful to have a baseline accuracy to compare against. A simple baseline to consider is majority voting where the classifier always outputs the class with the majority in the training set. Provide the accuracy of this classifier on both the training and the testing set. To find the accuracy, calculate the fraction in both the training and the testing set that has the same class as the majority class in the training set.

   Solution: Baseline by predicting everyone dies for training set is 0.6155. Baseline by predicting everyone dies for training set is 0.6102.

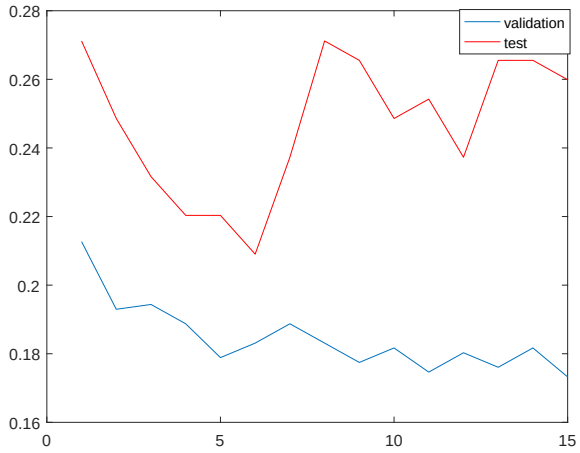
   (b) Now, use **fitctree** (**sklearn.tree.DecisionTreeClassifier** for python user) to train a decision tree classifier on the training data. Make sure to set the splitting criteria to "deviance" ("entropy" for python user) to make it use cross entropy. What is the training and testing accuracy of this model?

   Solution: Training error:0.0789 (python: 0.0141). Validation error:0.1915 (python: 0.1968). Testing error:0.2768 (python: 0.2938).

   (c) Now, use **fitcknn** (**sklearn.neighbors.KNeighborsClassifier** for python user) to train a K-nn classifier the training data. Try it out using $k = 1, 3, 5$. What is the training and testing accuracy of this model?
   Solution:
   $k = 1$:
   Training error:0.0155 (python: 0.0141). Validation error:0.2155 (python: 0.2251). Testing error:0.2712 (python: 0.2712).
   $k = 3$:
   Training error:0.1042 (python: 0.1028). Validation error:0.1986 (python: 0.1800). Testing error:0.2316 (python: 0.2316).
   $k = 5$:
   Training error:0.1282 (python: 0.1310). Validation error:0.1775 (python: 0.1673). Testing error:0.2203 (python: 0.2203).

(d) So far, we have only looked at training error which is not a good enough measure for how well the model generalizes. You will now use cross-validation error to measure the performance of your classifier. Familiarize yourself with **crossval** (**sklearn.model_selection.cross_val_score** for python user) in both class *ClassificationTree* and class *ClassificationKNN*. Using 10fold-cross validation, give the validation accuracy for the already trained models.
Solution: Provided above.

(e) You can also use cross-validation to help optimize for k in Knn classification. Find the 10-fold cross-validation error on Knn classifiers trained with "K" set to $1, 2, \ldots, 15$. Provide a plot of the validation error and testing error for each k. Try to explain you observation on how validation error and testing error change with respect to k.



Solution:
As k increases, both validation error and testing error goes down at beginning because small k tend to under-fit the data. As k goes too large, validation error keep decreasing but testing error start to increase. This is due to over-fitting. The zigzag behavior is due to even ks. When k is even, the error tends to go up because of ties.

(f) To predict the outcome of the Titanic dataset, which classifier will you use, the k-nearest neighbor or the decision tree? **Solution:** Use k-nearest neighbor classifier with $k = 5$ because it has lower validation error and testing error.