University of California, Los Angeles
Cyrus Asasi
506014946

ECE M146 **Prof. Suhas Diggavi**

Homework

Due Saturday, May 4th, 2024 11:59pm **via Gradescope**

1. In each of these problems, give a clear explanation for your choice.
   (a) In the ID3 algorithm, the resulting decision tree may sometimes be suboptimal.
   (b) A node in a decision tree cannot have more than two children.
   (c) At any point in time, the ID3 algorithms splits the data in the decision tree by finding the feature X that minimizes the mutual information $I(X;Y)$, where Y is the label.
   (d) You cannot use nearest neighbor classifier when there are categorical features in your dataset. TRUE / FALSE.
   (e) A k-NN classifier is more likely to overfit for larger values of k.
   (f) Consider a k-NN with $k = 1$ and only two samples in the training dataset. If we use an ℓ_2 distance metric, then the classifier is equivalent to a linear classifier.
   (g) Training k-NN classifiers involves only saving the training dataset into memory. TRUE / FALSE.
   (h) If we set $k = N$ where N is the number of samples in the training data, then k-NN will output the majority class among the training samples.
   (i) Deep decision trees are less likely to overfit.
   (j) A regularized solution which minimizes $J(\mathbf{w}) + \lambda R(\mathbf{w})$ leads to a model with a higher empirical training loss $J(\mathbf{w})$. Here $R(\mathbf{w})$ is a regularizer e.g. $\|w\|_2^2$.
   (k) You trained a binary classifier which has very high accuracy on training data but much lower accuracy on test data. Which of the following statements can be true? Give 1-2 lines of reasoning along with your answer.
      i. This is an example of underfitting.
      ii. The model may not have been regularized.
      iii. The training and test dataset are sampled from different distributions.
      iv. There are too many samples in the training set hence the model is overfitting to training set which causes poor performance in the test set.

2. You get the following data set:

V	W	X	Y
0	0	0	0
0	1	0	1
1	0	0	1
1	1	0	0
1	1	1	0

   Your task is to build a decision tree for classifying variable Y.
   (a) Write down the entire decision tree constructed by ID3. What is the training error of this classifier.
   (b) Can you find a tree with smaller height than the tree returned by ID3 in b), which also has zero training error?
   (c) Consider the following process: we start at the root and prune splits for which the information gain is less than some small number ϵ. This is called top-down pruning. What is the decision tree returned whenever $\epsilon = 10^{-4}$? What is the training set error for this tree?
   (d) What trees do we obtain as we vary $\epsilon > 0$?

3. State [YES/NO] for the following being valid positive semi-definite kernels or not. To receive full credit, you must provide an associated linear mapping for the [YES] case, or construct a counterexample for the [NO] case. No points would be awarded for answers without justification.

   (a) The function $k : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}_+$ defined as $k\left(x, x'\right) = \left(x - x'\right)^4$ for all $x, x' \in \mathbb{R}$. Is this a valid kernel?

   (b) Let $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^D, k_0\left(\mathbf{x}, \mathbf{x}'\right)$ be any valid kernel. Let

   $$k\left(\mathbf{x}, \mathbf{x}'\right) = \|\mathbf{x}\|^2 K_0\left(\mathbf{x}, \mathbf{x}'\right)\left\|\mathbf{x}'\right\|^2$$

   Is this a valid kernel?

4. Suppose we are looking for a maximum-margin linear classifier through the origin, (i.e. bias $b = 0$) for the hard margin SVM formulation, (i.e., no slack variables). In other words,

   $$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ s.t. } y^{(i)} \mathbf{w}^T \mathbf{x}^{(i)} \geq 1, i = 1, \ldots, n$$

   (a) Given a single training vector $\mathbf{x} = (1,1)^T \in \mathbb{R}^2$ with label $y = -1$, what is the $\mathbf{w}^*$ that satisfies the above constrained minimization?

   (b) Suppose we have two training examples, $\mathbf{x}^{(1)} = (1,1)^T \in \mathbb{R}^2$ and $\mathbf{x}^{(2)} = (1,0)^T \in \mathbb{R}^2$ with labels $y^{(1)} = 1$ and $y^{(2)} = -1$. What is $\mathbf{w}^*$ in this case?

   (c) Suppose we now allow the bias b to be non-zero. In other words, we now adopt the hard margin SVM formulation from lecture, where $\mathbf{w} = \boldsymbol{\theta}_{1:d}$ are the parameters excluding the bias:

   $$\min_{\boldsymbol{\theta}} \frac{1}{2}\|\mathbf{w}\|^2 \text{ s.t. } y^{(i)} \boldsymbol{\theta}^T \mathbf{x}^{(i)} \geq 1, i = 1, \ldots, n$$

   How would the classifier and the margin change in the previous question? What are $\left(\mathbf{w}^*, b^*\right)$? Compare your solutions with and without bias.

5. In this problem, we ask you to compare the classification models we have studied till now i.e., Decision trees, K-nearest neighbors, and Logistic regression.

   Introduction

   This data was extracted from the 1994 Census bureau database by Ronny Kohavi and Barry Becker. For computational reasons, we have already extracted a relatively clean subset of the data for this homework. The prediction task is to determine whether a person makes over $50K a year.

   In this problem, we ask you to complete the analysis of what sorts of people were likely to earn more than $50K a year. In particular, we ask you to apply the tools of machine learning to predict which individuals are more likely to have high income.

   Visualization

   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.

   Note: We have already converted all the categorical features to numerical ones. The target column is the last one: " $>$ 50k”, where 1 and 0 indicate $>$ 50k or $\leq$ 50k respectively. The feature "fnlwgt" describes the number of people the census believes the entry represents. All the other feature names should be self-explanatory.

   Evaluation

   Now, let us use `scikit-learn` to train a `DecisionTreeClassifier`, `KNeighborsClassifier`, and `LogisticRegression` 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]

(a) Make histograms for each feature, separating the examples by class (e.g. income greater than 50k or smaller than or equal to 50k). This should produce fourteen 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? (Please only describe the general trend. No need for more than two sentences per feature.)

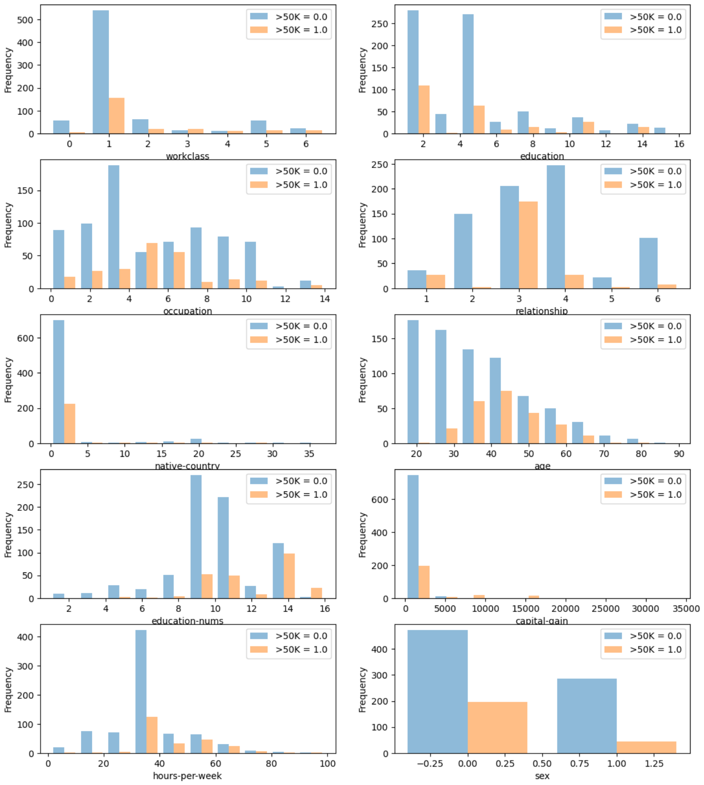

Figure: Histograms of each feature separated by class

i. For the workclass feature the histogram shows that the majority of data points lie within workclass 1. Workclass 1 contains both the most people who make less than 50K, but also the most people who make more than 50K in the dataset.

ii. For the education feature, we can once again see most of the data points within classes 1 through 5, which includes individuals who's highest level of education is either a high school or college bachelors degree.

iii. Regarding the marital status feature, we can see that the majority of our data set represents people who are in status 1 or 3, which correspond to being either divorced or separated. However, it is interesting to note that almost all the people in our dataset that make more than 50K a year are divorced.

iv. The occupation feature, shows how the majority of people in our dataset are making less than 50K regardless of occupation. The only exception is the exec-managerial occupation, in which most of the people in our data set would make more than 50K.

v. One interesting observation about relationship status within the dataset is that almost all of the people in our data who make more than 50K are husbands (category 3).

vi. Most of the data comes from people who identify as white. Whites hav both the most number of people who make less than 50K, but also the most number of people who make more than 50K.

vii. Unsurprisingly for a UCI census, almost all the people who we have data on are from the United States.

viii. This feature corresponds to age, and we can see how as people get older and reach their peak earnings throughout their 40s-50s, the number of people who are making more than 50K increases, before decreasing again throughout retirement age.

ix. TODO

x. The education number feature shows how as the number gets higher and people get more and more educated, their earning increases. This is shown by illustrating that most 50K+ earners are highly educated.

xi. The data for capital gain shows that nearly everyone who makes less than 50K shows low capital gain, whereas all the people who have significant capital gain make more than 50K.

xii. For capital loss, it shows that the majority of people, both high and low earners are not experiencing capital loss because they likely do not participate.

xiii. The hours-per-week feature shows clearly that people who do not work very many hours do not earn more than 50K. Among people who do work 40+ hours a week, we see many people making both less than and more than 50K.

xiv. The sex feature shows that many more females have filled out the census than males.

(b) 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 85% of the examples in the training set have $> 50\text{k} = 0$ and 15% have $> 50\text{k} = 1$, then, when applied to a test set, `RandomClassifier` should randomly predict 85% of the examples as $> 50\text{k} = 0$ and 15% as $> 50\text{k} = 1$.

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 of 0.385 or 0.374, depending on an implementation detail; both error values are correct.

(c) 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) 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 respectively.

(e) Similar to the previous question, train and evaluate a `LogisticRegression` (using the class from `scikit-learn` and referring to the documentation as needed). Use $\lambda = 0.1, 1$ and 10 as the regularization hyperparameter and report the training error of this classifier respectively. Make sure you initialize your classifier with the appropriate parameters; `random_state=0` and `max_iter=1000`. (Hint: function argument C is the inverse of regularization strength, therefore $\mathrm{C} = 1/\lambda$.)

(f) 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 `StratifiedShuffleSplit(...)` from `scikit-learn`. To ensure that we always get the same splits across different runs (and thus can compare the classifier results), set the $\text{random}_{\text{state}}$ parameter to be the same (e.g., 0).

Next, use your `error(...)` function to evaluate the training error and (cross-validation) validation error and validation micro averaged F1 (if you don't know what is F1 click here) Score of the `RandomClassifier`, `DecisionTreeClassifier`, `KNeighborsClassifier`, and `LogisticRegression` models (for the `DecisionTreeClassifier`, use 'entropy' criterion, for the
`KNeighborsClassifier`, use $k = 5$, for the `LogisticRegression`, use $\lambda = 1$, `random_state = 0` and `max_iter = 1000`). To do this, generate a random 85/15 split of the training data, train each model on the 85% fraction, evaluate the error on both the 85% and the 15% fraction, and repeat this 100 times to get an average result. What are the average training error, validation error, and validation F1 score of each of your classifiers on the `adult_subsample` data set?

(g) 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 5 -fold cross validation. You may find the `cross_val_score(...)` from `scikit-learn` helpful. Run 5-fold cross validation for all odd numbers ranging from 1 to 50 as the number of neighbors. Then plot the validation score against the number of neighbors, k. Include this plot in your writeup, and provide a 12 sentence description of your observations. What is the best value of k and what is the corresponding score?

(h) 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 `adult_subsample` data.

One way to prevent decision trees from overfitting is to limit their depth. Repeat your crossvalidation experiments but for increasing depth limits, specifically, $1, 2, \ldots, 20$. Then plot the average training error and validation error against the depth limit. Include this plot in your writeup, 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.