CS260, Winter 2017
Problem Set 2: Bayes, Logistics, and Decision Trees
Due 2/1/2017

Ray Zhang

Jan 21, 2017

1 Problem 1

(a) Problem 1a

Solution: Solution to problem 1a

We start off by stating some givens:

$x \in \Re^D$ and $y \in \{0, 1\}$

$P(X_j|y_k)$ $N(\mu_{jk}, \sigma_j)$

$P(y_i)$ $Ber(\theta)$, and $P(y_i = 1) = \pi$

Stating the bayes theorem:

$P(Y = 1|X) = \frac{P(X|Y=1)P(Y=1)}{P(X)}$ where

$P(X) = P(X|Y = 1)P(Y = 1) + P(X|Y = 0)P(Y = 0)$ from the total probability theorem's conditional analog.

The gaussian pdf equation is: $\frac{1}{\sqrt{2\pi}\sigma_d} exp(-(X_d - \mu_{d1})^2/(2\sigma_d)^2)$

We know that $P(X|Y = 1) = \Pi_d^D P(X_d|Y = 1)$, where D is the number of dimensions of features. We denote this probability in short as P_1, and for $P(X|Y = 0)$, it's denoted as P_0.

Because multiplying is the same as addition in the exponential term, we can define P_1 as follows:

$P_1 = exp(\sum_d^D -log\sqrt{2\pi}\sigma_d - \frac{(X_d-\mu_{d1})^2}{2\sigma_d^2})$

Plugging into the bayes theorem equation:

$\frac{P_1\pi}{P_1\pi+P_0(1-\pi)}$

We can divide both sides by the numerator to get:

$\frac{1}{1+P_0(1-\pi)/(P_1\pi)}$

The $P_0(1 - \pi)/(P_1\pi)$ term is equal to this once it's expanded, after cancellation of the square root pi term:

$exp(log(\frac{1-\pi}{\pi}) + \sum_d^D \frac{(X_d-\mu_{d1})^2-(X_d-\mu_{d0})^2}{2\sigma_d^2})$

Then we can start to see an ω_0 term(not finished yet), and we can see the $\omega^T X$ term(also not finished yet) as the summation.

To expand the summation, we see we can cancel the quadratic terms:

$\sum_d^D \frac{(X_d-\mu_{d1})^2-(X_d-\mu_{d0})^2}{2\sigma_d^2}) = \sum_d^D \frac{2\mu_{d0}X_d-2\mu_{d1}X_d+\mu_{d1}^2-\mu_{d0}^2}{2\sigma_d^2})$

We see that there are some constants we can take out, namely:

$\sum_d^D \frac{\mu_{d1}^2-\mu_{d0}^2}{2\sigma_d^2}$, which are absorbed by the ω_0 term.

The expression $\omega^T X$ can be satisfied if for some i-th index, $\omega_i = \frac{\mu_{i0}-\mu_{i1}}{\sigma_d^2}$

The answer is:

$$\omega = \begin{pmatrix} \frac{\mu_{10}-\mu_{11}}{\sigma_1^2} \\ \frac{\mu_{20}-\mu_{21}}{\sigma_2^2} \\ \dots \\ \frac{\mu_{D0}-\mu_{D1}}{\sigma_D^2} \end{pmatrix}$$

and

$\omega_0 = log(\frac{1-\pi}{\pi}) + \sum_d^D \frac{\mu_{d1}^2-\mu_{d0}^2}{2\sigma_d^2}$

(b) Problem 1b

Solution: Solution to problem 1b

This is essentially solving:

$argmax_{\mu,\sigma,\pi} P(X \cap Y) = argmax_{\mu,\sigma,\pi} P(X|Y)P(Y)$ The probability to optimize can be taken as the log probability since it's a monotonic function.

$log(P(X|Y)P(Y)) = logP(X|Y) + logP(Y)$

Since the two probabilities do not share variables, where $P(X|Y)$ uses μ, σ for the gaussian, $P(Y)$ uses π for the bernoulli, we can then optimize them separately:

$argmax_{\mu,\sigma} logP(X|Y) + argmax_{\pi} logP(Y)$

To maximize the bernoulli one(the one where π is involved), we plug in $P(Y)$ as a function of π:

$argmax_{\pi} log\Pi_n^N \pi_{y_n} = \sum_n^N log\pi_{y_n} = \sum_{n:y_n=0} log(1-\pi) + \sum_{n:y_n=1} log(\pi)$

To find the argmax, try to find the critical point of the function:

$\frac{\partial}{\partial x}(\sum_{n:y_n=0} log(1-\pi) + \frac{\partial}{\partial x}(\sum_{n:y_n=1} log(\pi))$

$= \sum_{n:y_n=0} \frac{1}{1-\pi} + \sum_{n:y_n=1} \frac{1}{\pi}$

Denote Z_0 as the number of elements with their corresponding labels to be 0, Z_1 for the labels to be 1, and Z as the ratio of Z_1/Z_0 then we get:

$\pi = Z_1/Z_0(1-\pi) = Z(1-\pi) = Z - Z\pi$

$\pi + Z\pi = Z$

$\pi = Z/(Z+1) = Z_1/(Z_1+Z_0)$ after expansion.

Now, to solve the second part, where μ, σ are involved, we also plug in the probability:

$argmax_{\mu,\sigma} logP(X|Y) = log(\Pi_n^N \Pi_d^D P(X_{nd}|y_n)) = \sum_n^N \sum_d^D logP(X_{nd}|y_n)$

We can separate by class:

$\sum_{n:y_n=0} \sum_d^D logP(X_{nd}|y=0) + \sum_{n:y_n=1} \sum_d^D logP(X_{nd}|y=1)$

Plugging in for $logP(X_{nd}|y=0) = -\frac{1}{2}log2\pi - log\sigma_d - \frac{(X_{nd}-\mu_{d0})^2}{2\sigma_d^2}$

and $logP(X_{nd}|y=1) = -\frac{1}{2}log2\pi - log\sigma_d - \frac{(X_{nd}-\mu_{d1})^2}{2\sigma_d^2}$,

we can get the following summation:

$\sum_{n:y_n=0} \sum_d^D -\frac{1}{2}log2\pi - log\sigma_d - \frac{(X_{nd}-\mu_{d0})^2}{2\sigma_d^2} + \sum_{n:y_n=1} \sum_d^D -\frac{1}{2}log2\pi - log\sigma_d - \frac{(X_{nd}-\mu_{d1})^2}{2\sigma_d^2}$

And then following, we can then take the derivative with respect to μ and get the following:

$\sum_{n:y_n=0} \sum_d^D \frac{X_{nd}-\mu_{d0}}{\sigma_d^2} + \sum_{n:y_n=1} \sum_d^D \frac{X_{nd}-\mu_{d1}}{2\sigma_d^2}$

We arrive at, for each of the summations above, the following expression:

for any d, for class 0: $\sum_{n:y_n=0} \frac{X_{nd}-\mu_{d0}}{\sigma_d^2} = \sum_{n:y_n=0} \frac{X_{nd}}{\sigma_d^2} - \sum_{n:y_n=0} \frac{\mu_{d0}}{\sigma_d^2}$.

We set the above summation equation to 0 and solve for μ_{d0}:

$Z_0 \frac{\mu_{d0}}{\sigma_d^2} = \sum_{n:y_n=0} \frac{X_{nd}}{\sigma_d^2}$

$\mu_{d0} = \frac{\sum_{n:y_n=0} X_{nd}}{Z_0}$

Similarly, we can do the same for μ_{d1}:

$\mu_{d1} = \frac{\sum_{n:y_n=1} X_{nd}}{Z_1}$

Now that we have solved the μ portion, we now also need to optimize with respect to σ.

Going back to the original expanded argmax function:

$\sum_{n:y_n=0} \sum_d^D -\frac{1}{2} log 2\pi - log\sigma_d - \frac{(X_{nd}-\mu_{d0})^2}{2\sigma_d^2} + \sum_{n:y_n=1} \sum_d^D -\frac{1}{2} log 2\pi - log\sigma_d - \frac{(X_{nd}-\mu_{d1})^2}{2\sigma_d^2}$

Taking derivative with respect to σ, we get:

$\sum_{n:y_n=0} \sum_d^D -\frac{1}{\sigma_d} + \frac{(X_{nd}-\mu_{d0})^2}{\sigma_d^3} + \sum_{n:y_n=1} \sum_d^D -\frac{1}{\sigma_d} + \frac{(X_{nd}-\mu_{d1})^2}{\sigma_d^3}$

We arrive at, for each of the summations above, the following expression:

for any d, for class 0:

$\sum_{n:y_n=0} -\frac{1}{\sigma_d} + \frac{(X_{nd}-\mu_{d0})^2}{\sigma_d^3} = -\frac{Z_0}{\sigma_d} + \sum_{n:y_n=0} \frac{(X_{nd}-\mu_{d0})^2}{\sigma_d^3}$

We set the above summation equation to 0 and solve for σ_d:

$\frac{Z_0\sigma_d^2}{2} = \sum_{n:y_n=0} (X_{nd} - \mu_{d0})^2$

Similarly, for class 1:

$Z_1\sigma_d^2 = \sum_{n:y_n=1} (X_{nd} - \mu_{d1})^2$

Appending both sides by both classes:

$Z_0\sigma_d^2 + Z_1\sigma_d^2 = \sum_{n:y_n=0} (X_{nd} - \mu_{d0})^2 + \sum_{n:y_n=1} (X_{nd} - \mu_{d1})^2$

The left hand side is equal to:

$(Z_0 + Z_1)\sigma_d^2$

Thus, for some σ_d, we have:

$\sigma_d = \sqrt{\frac{(\sum_{n:y_n=0} (X_{nd}-\mu_{d0})^2 + \sum_{n:y_n=1} (X_{nd}-\mu_{d1})^2)}{Z_0+Z_1}}$

This can be expressed in the shorthand sum:

$\sigma_d = \sqrt{\frac{\sum_c^C \sum_{n:y_n=c} (X_{nd}-\mu_{dc})^2}{Z}}$ for all d, where Z is the total count of all classes.

2 Problem 2

(a) Problem 2a

Solution: Solution to problem 2a

Our data is given in the following form: $X = \{(x_i, y_i)_{n=1}^N\}$

In logistic regression, we want to maximize the conditional probability, $P(y|X, w, b) = \Pi_{n=1}^N P(y_i|X_i, w, b)$

We know that for a given:

$P(y_i|X_i, w, b) = \sigma(w^T x + b)^{y_i}(1 - \sigma(w^T x + b))^{1-y_i}$

For the log probability:

$logP(y_i|X_i, w, b) = y_i log\sigma(w^T x + b) + (1 - y_i)log(1 - \sigma(w^T x + b))$

We plug this into our original equation, and multiply by -1 to get the answer:

$logP(y|X, w, b) = -\sum_{n=1}^N y_i log\sigma(w^T x+b)+(1-y_i)log(1-\sigma(w^T x+b))$

(b) Problem 2b

Solution: Solution to problem 2b

First to compute the gradient:

$f(w) = -\sum_n^N y_n log\sigma(w^T x_n) + (1 - y_n)log(1 - \sigma(w^T x_n))$

Denote $a = w^T x_n$

For any w_i, for a given data point x_n and y_n:

$\frac{\partial f}{\partial w_i} = -(\frac{y_n}{\sigma(a)}\sigma(a)(1 - \sigma(a))x_{ni} + \frac{1-y_n}{1-\sigma(a)}(-\sigma(a)(1 - \sigma(a)))x_{ni})$

$\frac{\partial f}{\partial w_i} = -((y_n(1 - \sigma(a)) + (1 - y_n)(-\sigma(a)))x_{ni})$

$\frac{\partial f}{\partial w_i} = (y_n - \sigma(a))x_{ni}$

Then for the sum:

$\frac{\partial f}{\partial w_i} = \sum_n^N (y_n - \sigma(w^T x_n))x_{ni}$

Now to compute the second derivative, for some w_i, w_j:

$\frac{\partial^2 f}{\partial w_i \partial w_j} = \frac{\partial}{\partial w_j}(\sum_n^N (y_n - \sigma(w^T x_n))x_{ni})$

Upon expanding, we see that the y_n term is insignificant. We take the derivative to get:

$\frac{\partial^2 f}{\partial w_i \partial w_j} = \sum_n^N \sigma(w^T x_n)(1 - \sigma(w^T x_n))x_{ni}x_{nj}$

Then for our hessian, we have some matrix that looks like:

$$H(f_w) = \begin{pmatrix} \sum_n^N \sigma(w^T x_n)(1 - \sigma(w^T x_n))x_{n1}^2 & \sum_n^N \sigma(w^T x_n)(1 - \sigma(w^T x_n))x_{n1}x_{n2} & ... \\ \sum_n^N \sigma(w^T x_n)(1 - \sigma(w^T x_n))x_{n2}x_{n1} & \sum_n^N \sigma(w^T x_n)(1 - \sigma(w^T x_n))x_{n2}^2 & ... \\ ... & ... & ... \end{pmatrix}$$

Which is for any index of the Hessian matrix:

$H(f_w)_{ij} = \sum_n^N \sigma(w^T x_n)(1 - \sigma(w^T x_n))x_{ni}x_{nj}$

We need to make sure that the hessian is positive semidefinite for it to be convex. However, we see that if we take out the common term $a = \sigma(w^T x_n)(1 - \sigma(w^T x_n))$(which is always positive because $\sigma(x)$'s range is bounded between 0 and 1, and thus $1 - \sigma(x)$ is also bounded between 0 and 1):

$H(f_w)_{ij} = \sum_n^N a x_{ni}x_{nj}$

This hessian is equal to:

$H(f_w) = \sum_n^N a x_n x_n^T$

We know that for any outer product matrix, it is positive semidefinite, and thus the summation of all outer product matrices with non-negative coefficients of a is positive semidefinite as well.

Thus, because $H(w)$ is positive semidefinite, the loss function is convex.

(c) Problem 2c

Solution: Solution to problem 2c

Found in part a and part b, we see that the gradient of f is:

$(\nabla_w f)_i = \sum_n^N (y_n - \sigma(w^T x_n))x_{ni}$

There are 2 cases:

case 1: $y = 1$, and linear separated.

By linearly separated, we have that $\sigma(w^T x_n) > 0.5$.

Then if we plug some value $\phi \in (0.5, 1)$ into the gradient (since the sigmoid function has a natural bound between $(0, 1)$, and we just specified that $x > 0.5$):

$(\nabla_w f)_i = \sum_n^N (y_n - \phi)x_{ni}$

This increases w_i if $x_{ni} > 0$, and decreases w_i if $x_{ni} < 0$. This means the dot product will only get greater.

By definition of the dot product:

$w^T x = w \cdot x = ||w||||x||cos\theta$

We can see that $||x||$ stays constant, and $cos\theta$ will only move closer to 1(if $x_{ni} > 0$), or -1(if $x_{ni} < 0$), and so $||w||$ will be the main contributing factor to the increase of the dot product, as the gradient suggests.

This means $||w||$ will continuously increase as we optimize on gradient descent.

case 2: $y = 0$, and linear separated.

By linearly separated, we have that $\sigma(w^T x_n) < 0.5$.

Then if we plug some value $\phi \in (0, 0.5)$ into the gradient (since the sigmoid function has a natural bound between $(0, 1)$, and we just specified that $x < 0.5$):

$(\nabla_w f)_i = \sum_n^N (y_n - \phi) x_{ni}$

This decreases w_i if $x_{ni} > 0$, and increases w_i if $x_{ni} < 0$. This is the reversed case of case 1. A similar argument in case 1 can be given here.

Thus, in both disjoint and exhaustive cases, $||w||$ will go to infinity as the function is being optimized.

(d) Problem 2d

Solution: Solution to problem 2d

We can use the addition rule for derivatives(and thus gradients) for the gradient of $L(w)$ with an extra L2 regularizer:

$L(w) = f(w) + ||w||_2^2$

$\nabla L(w) = \nabla f(w) + \nabla(||w||_2^2)$

$(\nabla_w f)_i = \frac{\partial f}{\partial w_i} = \sum_n^N (y_n - \sigma(w^T x_n)) x_{ni}$

Calculating $\nabla(||w||_2^2)$ is simple:

$\nabla(||w||_2^2)_i = \frac{\partial ||w||_2^2}{\partial w_i} = 2w_i$ from the polynomial rule in derivatives.

Thus:

$\frac{\partial L}{\partial w_i} = (y - \sigma(w^T x_n))x_{ni} + 2w_i$

(e) Problem 2e

Solution: Solution to problem 2e

To prove that the solution is unique, we can build upon the proof in problem 2c. If we can prove that the optima is not when the vector w is going to infinity, and that the function is **strictly convex**, we have restricted a space for the optima to reside in, and with the strictly convex condition, we can guarantee that there exists a unique solution within the restricted space we will define below.

Before we proved that $||w||$ can go to infinity as gradient descent continues. We can disprove this:

$\lim_{||w||\to\infty} L(w) = (y - \sigma(||w||||x_n||cos\theta))x_{ni} + 2w_i$

where $L(w)$ is the lagrangian in part 2d.

It is obvious that the growth of $\frac{\partial L(w)}{\partial w_i} = (y - \sigma(||w||||x_n||cos\theta))x_{ni}$ will be less than $2w$ because $\sigma(x)$ "squashes" a function to the range $(0, 1)$, and thus as $||w||$ approaches infinity, $2w$ will dominate and diverge to infinity.

Thus:

$\lim_{||w||\to\infty} L(w) =$

$-\sum_{n=1}^{N} y_i log\sigma(w^T x) + (1 - y_i)log(1 - \sigma(w^T x)) + ||w||_2^2 = \infty$

The cost function to minimize has diverged to infinity. For some trivial w, $w = 0$, as in the zero vector, we can plug into the equation:

$-\sum_{n=1}^{N} y_i log\sigma(0) + (1 - y_i)log(1 - \sigma(0)) + ||0||_2^2 = -log(0.5) + 0$ for $y_i = 0$ or $y_i = 1$.

This shows us that there exists a cost function value lower than ∞, and thus we prove that $||w||$ cannot be infinity.

However, one more condition has to be met for us to prove that there exists a unique minimum: **the function now has to be strictly convex.**

Consider answer 2b):

$H(f_w) = \sum_n^N a x_n x_n^T$, this is without the L2 regularizer.

Because addition is homogenous for differentiation, we can add the hessian of the L2 regularizer into the hessian of $f(x)$.

Denote $g(w) = ||w||_2^2$. Then $H(g_w) = 2I$

Quick explanation:

$\frac{\partial g}{\partial w_i} = 2w_i$

$\frac{\partial^2 g}{\partial w_i^2} = 2$

$\frac{\partial^2 g}{\partial w_i w_j} = 0$ where $i \neq j$.

Thus, the hessian is $2I$. This is a positive definite matrix. Adding this to a positive semidefinite matrix allows the sum to be positive definite.

A positive definite matrix defines a strictly convex function's second derivative(or hessian). Proof that a strictly convex function with minima not at the extremums has a single unique minimizer:

Suppose there is a function $f(x)$ that is strictly convex:

$f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta)f(y)$

Now suppose there are two unique solutions to the graph, x_1, x_2.

Then, plugging into strictly convex equation:

$f(\theta x_1 + (1-\theta)x_2) < \theta f(x_1) + (1-\theta)f(x_2)$

We get that there is a new minima at $\theta x_1 + (1-\theta)x_2$. Then that means that x_1 and x_2 are not the unique minima.

Thus, for a strictly convex function where the solutions are not at the boundaries, we know there exists only one unique solution.

Applying the proof to our problem, we can say that the loss function with an L2 regularizer will give us a single unique minimizer, which is not at the boundary.

3 Problem 3

(a) Problem 3a

Solution: Solution to problem 3a

Information gain is defined as:

$H[Y] - H[Y|X]$, where X in our case is satisfying an inequality created by the parallel axis splits of either of the two categorical features.

$H[Y] = p_{high} * logp_{high} + p_{low} * logp_{low}$.

Since we are using maximum likelihood estimate as the prior, we can plug it in as $N_{feature}/N_{total}$ where N is the size of the data.

$H[Y] = -(73/100 * log(73/100) + 27/100 * log(27/100)) = 0.5833$

Choosing weather, we get:

$H[Y|Sunny] = -(23/28 * log(23/28) + 5/28 * log(5/28)) = 0.4692$

$H[Y|Rainy] = -(50/72 * log(50/72) + 22/72 * log(22/72)) = 0.6155$

$H[Y|Weather] = P(Sunny)*H[Y|Sunny]+P(Rainy)*H[Y|Rainy] = 28/100 * 0.4692 + 72/100 * 0.6155 = 0.5745$

$H[Y] - H[Y|Weather] = 0.5833 - 0.5745 = 0.0088$.

Our information gain from weather is 0.0088, which is not a lot.

Choosing traffic, we get(trivially):

$H[Y|Heavy] = 73/73 * log(73/73) + 0/73 * log(0/73) = 0$

$H[Y|Light] = 0/27 * log(0/27) + 27/27 * log(27/27) = 0$

$H[Y|Traffic] = 0$

$H[Y] - H[Y|Traffic] = 0.5833 - 0 = 0.5833$.

Our information gain is maximized here. Thus, we split on traffic because the information gain is the greatest. It's also obvious because it separates the labels purely.

(b) Problem 3b

Solution: Solution to problem 3b

Answer: The students trees, T1 and T2 are both equivalent in their final cost and their tree structure. In fact, T2's tree's decision bounds are just T1's scaled monotonically.

Recall that a decision tree's splits are based on:

$argmin_{j=1,\ldots,D} min_{t \in T_j} cost((x_i, y_i)|x_i > t) + cost((x_i, y_i)|x_i \leq t)$

The algorithm for determining the $argmin$ is sorting the data points x_i for $i \in [N]$ on the j-th axis. There exists N-1 possible parallel axis splits to minimize the entropy on the j-th axis, which is evaluated, and the split resulting in the greatest information gain is chosen. After

subtracting by the mean and dividing by the standard deviation of the dataset, we get the following argmin function:

$argmin_{j=1,\ldots,D} min_{t \in T_j} cost(((x_i - \overline{x})/\sigma, y_i) | (x_i - \overline{x})/\sigma > t) +$

$cost(((x_i - \overline{x})/\sigma, y_i) | (x_i - \overline{x})/\sigma \leq t)$

This is just a monotonic transformation on the feature space, and thus the thresholds would also be a monotonic transformation on feature space.

The reason it's a monotonic transformation on the feature space is that the ordering of elements are invariant under addition and multiplication of positive numbers. Since $\sigma \geq 0$, we can say for sure that multiplying $\frac{1}{\sigma}$ gives us a positive value as well.

The minimum cost of the transformed space, as a result, would be the exact same as the original space, as a result of creating parallel axis splits in the same order(thus creating the same structure of the tree).

(c) Problem 3c

Solution: Solution to problem 3c

Proving that $Gini(X) \leq H[X]$

$Gini(X) = \sum_{k=1}^{K} p_k(1 - p_k)$ where p_k is the maximum likelihood of the k-th class from our dataset.

$H[X] = -\sum_{k=1}^{K} p_k log p_k$

Elementwise, we can prove that $H[X] - Gini(X) \geq 0$, by stating that:

$-log p_k - (1 - p_k) \geq 0$ for all $k \in [K]$.

We first prove that the function is convex, in that there exists a single, global minima.

$\frac{\partial^2}{\partial x^2}(-log p_k - (1 - p_k)) = 1/p_k^2$

Thus, we can see that for all p_k we have that the second derivative is positive.

Finding the global minimum of the given inequality, and seeing that it is ≥ 0 will confirm that for every k, the summation will be of positive numbers.

$\frac{\partial}{\partial x}(-log p_k - (1 - p_k)) = -1/p_k + 1$

Solving for extremum:

$1/p_k = 1$

$p_k = 1$

The local minima is at $p_k = 1$. We can then plug it into the equation:

$-log(1) - (1 - 1) = 0 \geq 0$

And thus we confirmed that for all terms within the summation we have that it will be positive, then:

$H[X] - Gini(X) = \sum_{k=1}^{K} p_k(-logp_k - (1 - p_k)) \geq 0$

This proves that the value of the Gini index is less than or equal to the corresponding value of the cross-entropy.

4 Problem 4

(a) Problem 4b

Solution: Solution to problem 4b

KNN accuracies from hw1:

The value of K: 1, new accuracy: 7.557841e-01, train accuracy: 7.778947e-01

The value of K: 3, new accuracy: 8.380463e-01, train accuracy: 8.578947e-01

The value of K: 5, new accuracy: 8.586118e-01, train accuracy: 8.894737e-01

The value of K: 7, new accuracy: 8.688946e-01, train accuracy: 9.052632e-01

The value of K: 9, new accuracy: 8.740360e-01, train accuracy: 9.031579e-01

The value of K: 11, new accuracy: 8.894602e-01, train accuracy: 9.031579e-01

The value of K: 13, new accuracy: 8.740360e-01, train accuracy: 8.947368e-01

The value of K: 15, new accuracy: 8.560411e-01, train accuracy: 8.926316e-01

The value of K: 17, new accuracy: 8.560411e-01, train accuracy: 8.789474e-01

The value of K: 19, new accuracy: 8.560411e-01, train accuracy: 8.673684e-01

The value of K: 21, new accuracy: 8.431877e-01, train accuracy: 8.631579e-01

The value of K: 23, new accuracy: 8.380463e-01, train accuracy: 8.568421e-01

The value of K: 25, new accuracy: 8.406170e-01, train accuracy: 8.578947e-01

Best k = 11

The value of the test accuracy is : 8.894602e-01

Decision tree accuracies:

The value of leaf size: 1, using criterion: gdi, new accuracy: 9.460154e-01, train accuracy: 9.705263e-01

The value of leaf size: 1, using criterion: deviance, new accuracy: 9.280206e-01, train accuracy: 9.663158e-01

The value of leaf size: 2, using criterion: gdi, new accuracy: 9.460154e-01, train accuracy: 9.705263e-01

The value of leaf size: 2, using criterion: deviance, new accuracy: 9.280206e-01, train accuracy: 9.663158e-01

The value of leaf size: 3, using criterion: gdi, new accuracy: 9.357326e-01, train accuracy: 9.684211e-01

The value of leaf size: 3, using criterion: deviance, new accuracy: 9.177378e-01, train accuracy: 9.642105e-01

The value of leaf size: 4, using criterion: gdi, new accuracy: 9.280206e-01, train accuracy: 9.631579e-01

The value of leaf size: 4, using criterion: deviance, new accuracy: 9.177378e-01, train accuracy: 9.642105e-01

The value of leaf size: 5, using criterion: gdi, new accuracy: 9.228792e-01, train accuracy: 9.600000e-01

The value of leaf size: 5, using criterion: deviance, new accuracy: 9.177378e-01, train accuracy: 9.642105e-01

The value of leaf size: 6, using criterion: gdi, new accuracy: 9.383033e-01, train accuracy: 9.536842e-01

The value of leaf size: 6, using criterion: deviance, new accuracy: 9.305913e-01, train accuracy: 9.621053e-01

The value of leaf size: 7, using criterion: gdi, new accuracy: 9.280206e-01, train accuracy: 9.452632e-01

The value of leaf size: 7, using criterion: deviance, new accuracy: 9.254499e-01, train accuracy: 9.515789e-01

The value of leaf size: 8, using criterion: gdi, new accuracy: 9.254499e-01, train accuracy: 9.410526e-01

The value of leaf size: 8, using criterion: deviance, new accuracy: 9.177378e-01, train accuracy: 9.442105e-01

The value of leaf size: 9, using criterion: gdi, new accuracy: 9.228792e-01, train accuracy: 9.305263e-01

The value of leaf size: 9, using criterion: deviance, new accuracy: 9.203085e-01, train accuracy: 9.336842e-01

The value of leaf size: 10, using criterion: gdi, new accuracy: 9.048843e-01, train accuracy: 9.273684e-01

The value of leaf size: 10, using criterion: deviance, new accuracy: 9.023136e-01, train accuracy: 9.284211e-01

The best new accuracy: 9.460154e-01, value of leaf size: 1, using criterion: gdi

Using it to train on test set accuracy: 9.460154e-01

Naive bayes accuracy:

The value of test accuracy: 8.406170e-01, validation accuracy: 8.329049e-01, train accuracy: 8.831579e-01

Logistic regression accuracy:

The value of test accuracy: 9.280206e-01, validation accuracy: 9.203085e-01, train accuracy: 9.536842e-01

5 Discussion

I have discussed the problem set with the TA's of the class, Denali Molitor, and Frank Chen.