1 Comparison of Machine Learning Algorithms [Jayant, 20 points]

In this problem, you will review the important aspects of the algorithms we have learned about in class. For every algorithm listed in the two tables on the next pages, fill out the entries under each column according to the following guidelines. Turn in your completed table with your problem set. [$\approx \frac{1}{2}$ **point per entry**]

Guidelines:

1. **Generative or Discriminative** – Choose either "generative" or "discriminative"; you may write "G" and "D" respectively to save some writing.
2. **Loss Function** – Write either the name or the form of the loss function optimized by the algorithm (e.g., "exponential loss").
3. **Decision Boundary / Regression Function Shape** – Describe the shape of the decision surface or regression function, e.g., "linear". If necessary, enumerate conditions under which the decision boundary has different forms.
4. **Parameter Estimation Algorithm / Prediction Algorithm** – Name or concisely describe an algorithm for estimating the parameters or predicting the value of a new instance. Your answer should fit in the provided box.
5. **Model Complexity Reduction** – Name a technique for limiting model complexity and preventing overfitting.

Learning Method	Generative or Discriminative?	Loss Function	Decision Boundary	Parameter Estimation Algorithm	Model Complexity Reduction
Gaussian Naïve Bayes					
Logistic Regression					
Decision Trees					
K-Nearest Neighbors					
Support Vector Machines (with slack variables, no kernel)					
Boosting (with decision stumps)					

Table 1: Comparison of Classification Algorithms

Learning Method	Loss Function	Regression Function Shape	Parameter Estimation Algorithm	Prediction Algorithm
Linear Regression (assuming Gaussian noise model)				
Nadaraya-Watson Kernel Regression				
Regression Trees				

Table 2: Comparison of Regression Algorithms

2 Programming – Adaboost [Rob Hall, 20 points]

The goal of this question is to implement the adaboost algorithm for a classification problem, where the "weak learners" are decision stumps. Our data consist of $X \in \mathbb{R}^{n \times p}$ matrix of covariates, and a response vector $y \in \{-1, +1\}^n$.

A decision stump is defined by:

$$h_{(a,d,j)}(x) = \begin{cases} d & \text{if } x_j \leq a \\ -d & \text{o/w} \end{cases}$$

Where $a \in \mathbb{R}, j \in \{1 \dots p\}, d \in \{-1, +1\}$. Here $x \in \mathbb{R}^p$ is a vector, and x_j is the j^{th} coordinate.

The data for this assignment may be found at: `http://www.cs.cmu.edu/~aarti/Class/10701/hws/hw3-data.tar.gz` . It consists of both a training and testing set of data. Each consists of 1000 examples. There are 25 real valued features for each example, and a corresponding y label.

1. **[10 points]** Give a matlab program which takes as input: the data along with a set of weights (i.e., $\{(x_i, y_i, w_i)\}_{i=1}^n$, where $w_i \geq 0$ and $\sum_{i=1}^n w_i = 1$), and returns the decision stump which minimizes the weighted training error. Note that this requires selecting both the optimal a, d of the stump, and also the optimal coordinate j.

   The output should be a pair $(a^\star, d^\star, j^\star)$ with:

   $$\ell(a^\star, d^\star, j^\star) = \min_{a,d,j} \ell(a, d, j) = \min_{a,d,j} \sum_{i=1}^n w_i \mathbf{1}\{h_{a,d,j}(x_i) \neq y_i\}$$

   For full points, your approach should run in time $O(pn \log n)$ or better (Hint: you may assume that built in sorting routines will sort a list of length m in time $O(m \log m)$). Remember to include a proof of correctness of your algorithm as well as an analysis of its running time.

2. **[8 points]** Give a matlab program which implements AdaBoost, and uses decision stumps as weak learners. Use your decision stump program from above as a subroutine.

3. **[2 points]** Run your AdaBoost loop for 250 iterations on the data set, and plot the training error and testing error as a function of iteration number.

3 SVM [Min Chi, 20 points]

1. **[5 points]** Figure 1 (at the end of this problem) plots SVM decision boundries resulting from using different kernels and/or different slack penalties. In Figure 1, there are two classes of training data, with labels $y_i \in \{-1, 1\}$, represented by circles and squares respectively. The SOLID circles and squares represent the Support Vectors. Determine which plot in Figure 1 was generated by each of the following optimization problems: (Note that there are 6 plots, but only 5 problems, so one plot does not match any of the problems)

   (a)

   $$\min \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^n \xi_i$$

   s.t. $\forall i = 1, \cdots, n$:
   $\xi_i \geq 0$

$(\mathbf{w} \cdot \mathbf{x_i} + b)y_i - (1 - \xi_i) \geq 0$
and $C = 0.1$.

(b)

$$\min \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{n} \xi_i \tag{1}$$

s.t. $\forall i = 1, \cdots, n$:
$\xi_i \geq 0$
$(\mathbf{w} \cdot \mathbf{x_i} + b)y_i - (1 - \xi_i) \geq 0$
and $C = 1$.

(c)

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) \tag{2}$$

s.t. $\sum_{i=1}^{n} \alpha_i y_i = 0$;
$\alpha_i \geq 0, \forall i = 1, \cdots, n$;
where $\mathbf{K}(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \mathbf{v} + (\mathbf{u} \cdot \mathbf{v})^2$.

(d)

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) \tag{3}$$

s.t. $\sum_{i=1}^{n} \alpha_i y_i = 0$;
$\alpha_i \geq 0, \forall i = 1, \cdots, n$;
where $\mathbf{K}(\mathbf{u}, \mathbf{v}) = \exp(-\frac{\|\mathbf{u}-\mathbf{v}\|^2}{2})$.

(e)

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) \tag{4}$$

s.t. $\sum_{i=1}^{n} \alpha_i y_i = 0$;
$\alpha_i \geq 0, \forall i = 1, \cdots, n$;
where $\mathbf{K}(\mathbf{u}, \mathbf{v}) = \exp(- \parallel \mathbf{u} - \mathbf{v} \parallel^2)$.

2. **[3 points]** Consider the linear SVM with slack penalties:

$$\min \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{n} \xi_i \tag{5}$$

s.t. $\xi_i \geq 0$ and $(\mathbf{w} \cdot \mathbf{x_i} + b)y_i - (1 - \xi_i) \geq 0$ for all $i = 1, \cdots, n$;

Indicate which of the following statements will hold as we increase the constant C from some starting value. Use 'True' if the statement holds in all circumstances; 'False' if the statement never holds; and 'Possible' if the statement holds in some cases but not others.

(a) **[1 point]** b will not increase.

(b) **[1 point]** more points will be misclassified.

(c) **[1 point]** the margin will not increase.

3. **[12 points]** Consider the problem of separating the set of training vectors belonging to two separate classes. Our training data is of the form $\{(x_i, y_i)\}$ where the feature vectors $x_i \in \mathbb{R}^m$ and the class label $y_i \in \{-1, 1\}$.

   As mentioned in the lecture, if the training data is not linearly separable (i.e., by a decision rule $\text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$ for some $\mathbf{w}, b$), we need to formulate the problem with slack variables $\{\xi_i\}, 1 \leq i \leq n$. Moreover, the SVM classifier with the largest margin is obtained by solving the dual problem:

$$L(\mathbf{w}, b, \alpha, \xi, \beta) = \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i[(\mathbf{w} \cdot \mathbf{x_i} + b)y_i - (1 - \xi_i)] - \sum_{i=1}^{n} \beta_i \xi_i \tag{6}$$

   where C is a constant, $\alpha_i, \beta_i \geq 0, \forall i$ are the Lagrange multipliers, and $\xi_i \geq 0$ are the slack variables.

   (a) **[2 points]** Assume that $n = 4$ and $\mathbf{x}$ is two-dimensional $< x_i^1, x_i^2 >$: $< 2, 2 >, < 2.5, 2.5 >, < 5, 5 >, < 7, 7 >$. We now train SVM with equation 6. Show that for any labeling y of the four training examples, the optimal parameter vector $\hat{\mathbf{w}} = (\hat{w}^1, \hat{w}^2)$ has the property that $\hat{w}^1 = \hat{w}^2$.

   (b) **[5 points]** Consider training an SVM with slack variables, but with no bias variable (which means $b = 0$). We will use a kernel $\mathbf{K}(\mathbf{u}, \mathbf{v})$ with the property that for any two points $\mathbf{u}$ and $\mathbf{v}$ in the training set, $-1 < \mathbf{K}(\mathbf{u}, \mathbf{v}) < 1$. Furthermore, $\mathbf{K}(\mathbf{u}, \mathbf{u}) < 1$. There are n points in the training set, Show that if the constant C is chosen such that $C < \frac{1}{n-1}$, then all of the dual variables α_i are non-zero (i.e., all points in the training set become support vectors).

   (c) **[5 points]** Consider the kernel:

$$\mathbf{K}(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \mathbf{v} + 4(\mathbf{u} \cdot \mathbf{v})^2 \tag{7}$$

   where the vectors $\mathbf{u}$ and $\mathbf{v}$ are 2-dimensional. This kernel is equal to an inner product $\phi(u) \cdot \phi(v)$ for some definition of ϕ. What is the function ϕ?

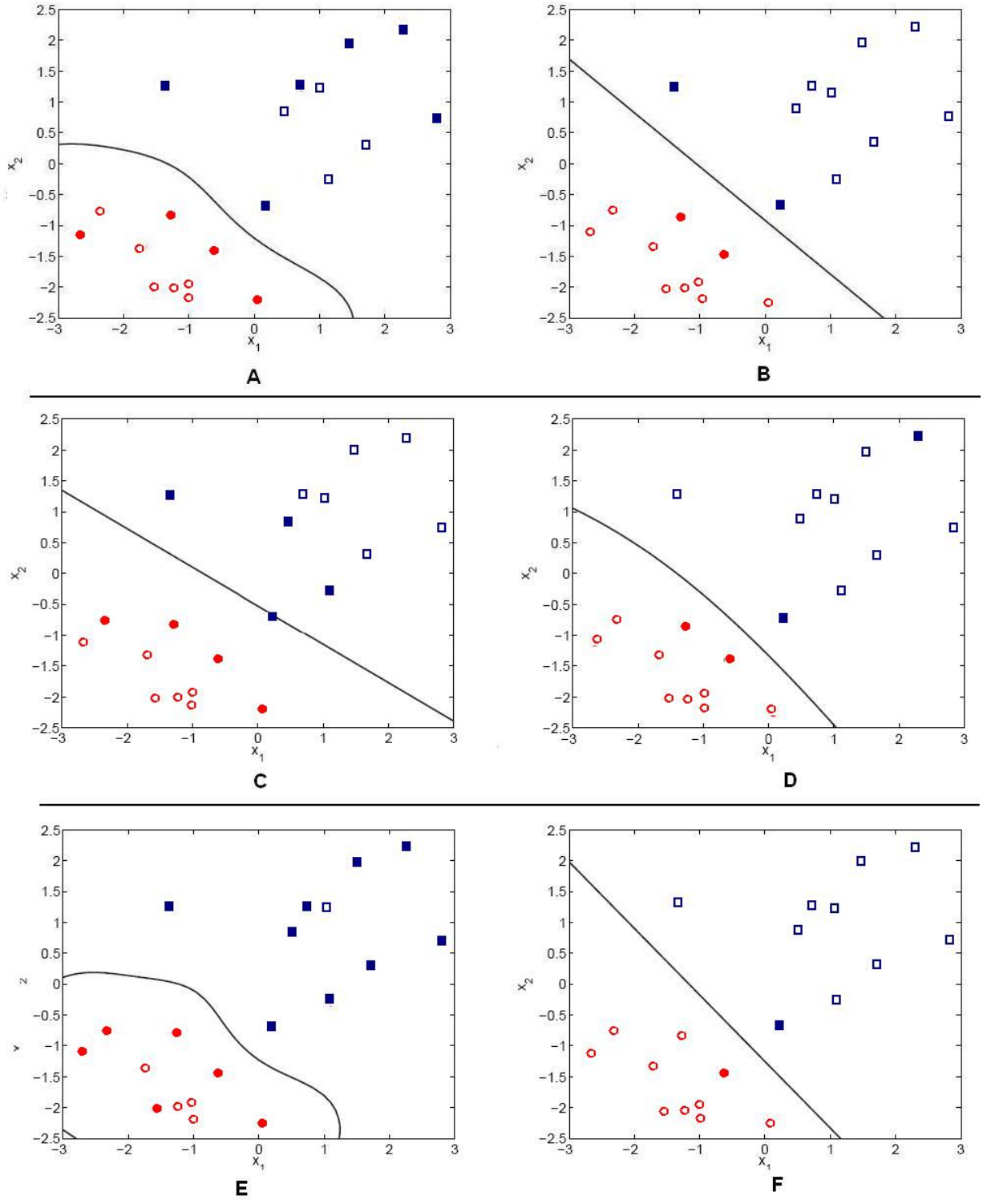

Figure 1: Induced Decision Boundaries

4 Hierarchical Clustering [Leman, 20 points]

In class you learned about bottom-up hierarchical clustering (a.k.a. agglomerative clustering), the basic algorithm of which is:

1. Start with each point in a cluster of its own
2. Until there is only one cluster
   (a) Find the closest pair of clusters
   (b) Merge them
3. Return the tree of cluster-mergers

To turn the above algorithm into a definite procedure, one needs to be able to *quantify* how *close* two clusters are. In class, you also learned several methods that define the distance between two clusters, such as Single-Link, Complete-Link, Average-Link, etc.

4.1 An alternative distance metric

In this problem you will anlayze an alternative approach to quantify the distance between two disjoint clusters, proposed by Joe H. Ward in 1963. We will call it **Ward's metric**.

Ward's metric simply says that the distance between two disjoint clusters, X and Y, is how much the sum of squares will increase when we merge them. More formally,

$$\Delta(X,Y) = \sum_{i \in X \cup Y} \|\vec{x_i} - \mu_{\vec{X \cup Y}}\|^2 - \sum_{i \in X} \|\vec{x_i} - \mu_{\vec{X}}\|^2 - \sum_{i \in Y} \|\vec{x_i} - \mu_{\vec{Y}}\|^2 \tag{8}$$

where μ_i is the centroid of cluster i and x_i is a data point in a give cluster. Here, $\Delta(X,Y)$ can be thought as the *merging cost* of combining clusters X and Y into one cluster. That is, in agglomerative clustering those two clusters with the lowest *merging cost* is merged using the Ward's metric as a *closeness* measure.

1. **[5 points]** Can you reduce the formula in Equation 1 for $\Delta(X,Y)$ to a simpler form? Give the simplified formula. (Please show all your work including the intermediate steps.) *Hint:* Your formula should be in terms of the cluster sizes (lets denote them as n_X and n_Y) and the distance $\|\vec{\mu_X} - \vec{\mu_Y}\|^2$ between cluster centroids μ_X and μ_Y **only**.
2. **[3 points]** Give an interpretation for Ward's metric. What do you think it is trying to achieve? *Hint:* The simplified formula from above will be helpful to answer this part.
3. **[5 points]** Assume that you are given two *pairs* of clusters P_1 and P_2. The centers of the two clusters in P_1 is farther apart than the centers of the two clusters in P_2. Using Ward's metric, does agglomerative clustering **always** choose to merge the two clusters in P_2 (those with less 'distance' between their centers)? Why (not)? Justify your answer with a simple example.

Extra credit: [3 points] In clustering it is usually not trivial to decide what is the right number of clusters the data falls into. Using Ward's metric for agglomerative clustering, can you come up with a simple heuristic to pick the number of clusters k?

4.2 Efficient updates of the closeness matrix

In class we discussed how the *closeness* matrix for hierarchical agglomerative clustering needs to be updated after each step. The naïve way of doing this would be to compute the distances between the to-be-merged two clusters X and Y and the other clusters from scratch based on the individual data points in the clusters.

1. **[4 points]** Suppose we want to come up with a more efficient update algorithm that can compute the new closeness matrix using **only** the entries from the current closeness matrix, that is, without having to access the individual data points in the clusters. For which of the following methods can we achieve this goal? 1) Single-Link, 2) Complete-Link, 3) Average-Link, 4) Ward's metric. Explain your reasoning.
2. **[3 points]** If the computation is not possible based on the current closeness matrix **alone** for some of the methods above, could you make it possible by storing a little extra information about each cluster? Please state clearly what extra information should be stored for which method.

5 K-means [TK, 20 points]

Let $X := \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be our sample points, and K denote the number of clusters to use. We represent the cluster assignments of the data points by an indicator matrix $\gamma \in \{0,1\}^{n \times K}$ such that $\gamma_{ik} = 1$ means $\mathbf{x}_i$ belongs to cluster k. We require that each point belongs to exactly one cluster, so $\sum_{k=1}^{K} \gamma_{ik} = 1$. The K-means method estimates γ by minimizing the following measure of distortion:

$$J(\gamma, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K) \;:=\; \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2,$$

where $\|\cdot\|$ denotes the vector 2-norm. The most popular algorithm for minimizing J is due to Lloyd[1] (1957), which alternates between estimating γ and re-computing $\boldsymbol{\mu}_k$'s:

- Initialize $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K$, and let $C := \{1, \ldots, K\}$.
- While the value of J is still decreasing[2], repeat the following:
  1. Determine γ by
  $$\gamma_{ik} \;\leftarrow\; \begin{cases} 1, & \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \le \|\mathbf{x}_i - \boldsymbol{\mu}_{k'}\|^2,\ \forall k' \in C, \\ 0, & \text{otherwise.} \end{cases}$$
  Break ties arbitrarily.
  2. Recompute $\boldsymbol{\mu}_k$ using the updated γ:
  For each $k \in C$, if $\sum_{i=1}^{n} \gamma_{ik} > 0$ set
  $$\boldsymbol{\mu}_k \;\leftarrow\; \frac{\sum_{i=1}^{n} \gamma_{ik} \mathbf{x}_i}{\sum_{i=1}^{n} \gamma_{ik}}.$$
  Otherwise, remove k from C.

1. [5 points] Show that Lloyd's algorithm stops in a finite number of iterations. (Hint: How many different values can γ take?)

[1] Lloyd, S. P. (1957). "Least square quantization in PCM". Bell Telephone Laboratories Paper.
[2] This stopping condition is slightly different from the one in Lloyd's algorithm: stop when γ remains the same.

2. [5 points] Let $\bar{\mathbf{x}}$ denote the sample mean. Consider the following three quantities:

$$\text{Total variation:} \qquad V(X) \;:=\; \frac{\sum_{i=1}^{n} \|\mathbf{x}_i - \bar{\mathbf{x}}\|^2}{n}.$$

$$\text{Within-cluster variation:} \qquad V_k(X) \;:=\; \frac{\sum_{i=1}^{n} \gamma_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2}{\sum_{i=1}^{n} \gamma_{ik}}.$$

$$\text{Between-cluster variation:} \qquad \widetilde{V}(X) \;:=\; \sum_{k=1}^{K} \left(\frac{\sum_{i=1}^{n} \gamma_{ik}}{n} \right) \|\boldsymbol{\mu}_k - \bar{\mathbf{x}}\|^2.$$

What is the relation between these three quantities? Based on this relation, show that K-means can be interpreted as minimizing a weighted average of with-in cluster variations while approximately maximizing the between-cluster variation. Note that the relation may contain an extra term that does not appear above.

3. [5 points] Instead of the squared Euclidean distance, we now use the Manhattan distance, denoted by $\|\cdot\|_1$, as the measure of distortion:

$$J_1(\gamma, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K) \;:=\; \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_1.$$

We minimize J_1 by a variant of the Lloyd's algorithm:

- Initialize $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K$, and let $C := \{1, \ldots, K\}$.
- While the value of J_1 is still decreasing, repeat the following:
  1. Determine γ by
  $$\gamma_{ik} \;\leftarrow\; \begin{cases} 1, & \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_1 \le \|\mathbf{x}_i - \boldsymbol{\mu}_{k'}\|_1, \ \forall k' \in C, \\ 0, & \text{otherwise.} \end{cases}$$
  Break ties arbitrarily.
  2. Recompute $\boldsymbol{\mu}_k$ using the updated γ:
  For each $k \in C$, if $\sum_{i=1}^{n} \gamma_{ik} > 0$ set
  $$\boldsymbol{\mu}_k \;\leftarrow\; ?$$
  Otherwise, remove k from C.

Fill in the missing update rule for $\boldsymbol{\mu}_k$ such that the algorithm produces a sequence of decreasing objective values unless a local minimum is reached. Note that

- Answers may not be unique.
- The following fact may be useful. Let $\{r_1, r_2, \ldots, r_n\}$ be n real numbers. The solution to the minimization problem
$$\min_{x} \sum_{i=1}^{n} |x - r_i|$$
is the median of $\{r_1, r_2, \ldots, r_n\}$.

4. [5 points] Show that the minimum of J is a non-increasing function of K, thereby conclude that it makes no sense to choose the number of clusters K by minimizing J.