CPSC 340 Assignment 3 (due Friday October 11 at 11:55pm)

f4r1b 42252965

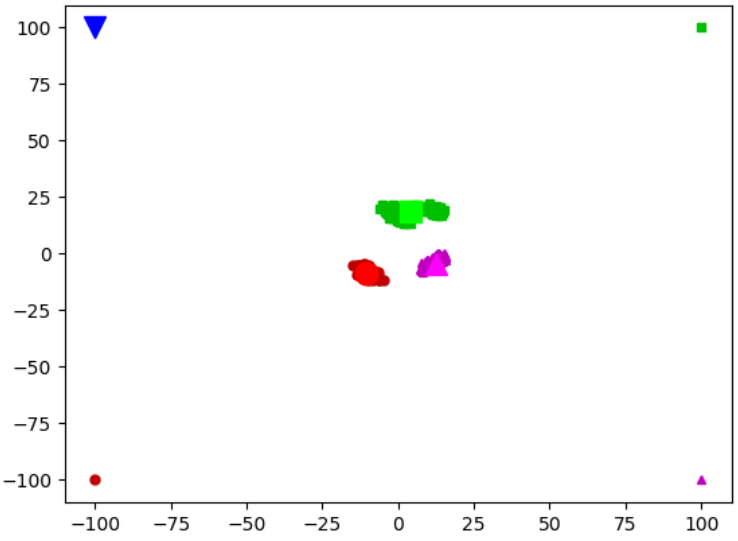
On this assignment, we are only going to allow a maximum of 1 "late class" to be used. This is because the midterm is on October 17th, and we want to give you more than 1 day to look at the solutions before the actual exam.

1 More Unsupervised Learning

1.1 k-Medians

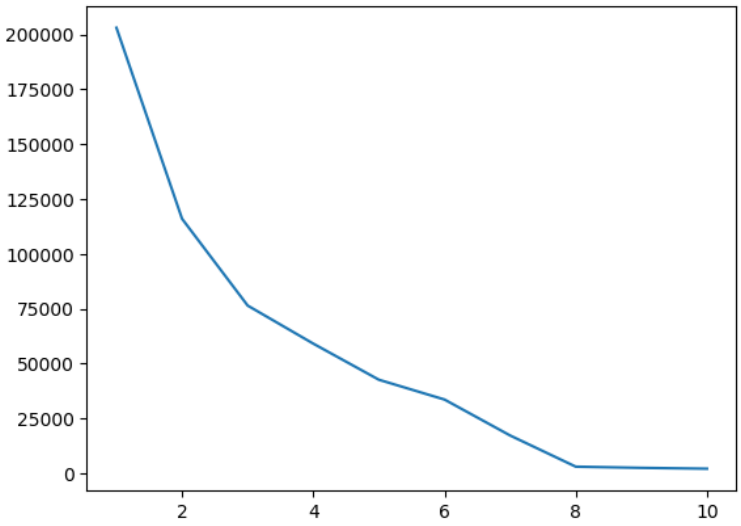
The data in *clusterData2.jl* is the exact same as *clusterData.jl* from a previous question, except it has 4 outliers that are far away from the data.

1. Using the *clustering2Dplot* function, output the clustering obtained by running k-means 50 times (with $k = 4$) on *clusterData2.mat* and taking the one with the lowest error. Are you satisfied with the result?



Answer: No, the algorithm have detected 3 Classed in the middle, but in the previous assignments, it should ideally identify 4 classed in the middle. But in this condition, an outlier have taken away one of the class.

2. What values of k might be chosen by the elbow method for this dataset?



Answer: 8

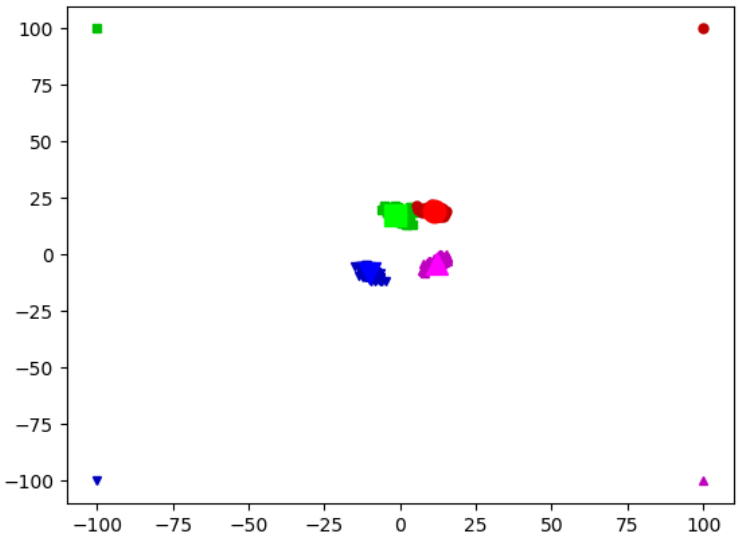
3. Implement the *k-medians* algorithm, which assigns examples to the nearest w_c in the L1-norm and then updates the w_c by setting them to the "median" of the points assigned to the cluster (we define the d-dimensional median as the concatenation of the medians along each dimension). For this algorithm it makes sense to use the L1-norm version of the error (where y_i now represents the closest median in the L1-norm),

$$f(w_1, w_2, \ldots, w_k, y_1, y_2, \ldots, y_n) = \sum_{i=1}^{n} \|x_i - w_{y_i}\|_1 = \sum_{i=1}^{n} \sum_{j=1}^{d} |x_{ij} - w_{y_i j}|,$$

Hand in your code and plot obtained by taking the clustering with the lowest L1-norm after using 50 random initializations for $k = 4$.

Answer:
Code on next page.

```julia
using Printf
using Statistics
using Random
include("misc.jl")
include("clustering2Dplot.jl")

mutable struct PartitionModel
    predict # Function for clustering new points
    y # Cluster assignments
    W # Prototype points
end

function kMedians(X,k;doPlot=false)
# K-means clustering

(n,d) = size(X)

# Choos random points to initialize means
W = zeros(k,d)
perm = randperm(n)
for c = 1:k
    W[c,:] = X[perm[c],:]
end

# Initialize cluster assignment vector
y = zeros(Int64, n)
changes = n
while changes ≠ 0

    # Compute (squared) Euclidean distance between each point and each mean
    D = distancesSquared(X,W)

    # Degenerate clusters will distance NaN, change to Inf
    # (since Julia thinks NaN is smaller than all other numbers)
    D[findall(isnan.(D))] .= Inf

    # Assign each data point to closest mean (track number of changes labels)
    changes = 0
    for i in 1:n
        (~,y_new) = findmin(D[i,:])
        changes += (y_new ≠ y[i])
        y[i] = y_new
    end

    # Optionally visualize the algorithm steps
    if doPlot && d == 2
        clustering2Dplot(X,y,W)
        sleep(.1)
    end

    # Find median of each cluster
    for c in 1:k
        W[c,:] = median(X[y.==c,:],dims=1)
    end

    # Optionally visualize the algorithm steps
    if doPlot && d == 2
        clustering2Dplot(X,y,W)
        sleep(.1)
    end

    #@printf("Running k-means, changes = %d\n",changes)
end

function predict(Xhat)
    (t,d) = size(Xhat)

    D = distancesSquared(Xhat,W)

    yhat = zeros(Int64,t)
    for i in 1:t
        (~,yhat[i]) = findmin(D[i,:])
    end
    return yhat
end

return PartitionModel(predict,y,W)
end

function kMediansError(X,y,W)
    (n,d) = size(X)

    f = 0
    for i in 1:n
        for j = 1:d
            f += (X[i,j] - W[y[i],j])
        end
    end
    return f
end
```



4. What value of k would be chosen by the elbow method if you k-medians and the L1-norm? Are you satisfied with this result?

   Answer: 3, yes

1.2 Density-Based Clustering

If you run the function *example_dbCluster*, it will apply the basic density-based clustering algorithm to the dataset from the previous part. The final output should look like this:
(The right plot is zoomed in to show the non-outlier part of the data.) Even though we know that each object was generated from one of four clusters (and we have 4 outliers), the algorithm finds 6 clusters and does not assign some of the original non-outlier objects to any cluster. However, the clusters will change if we change the parameters of the algorithm. Find and report values for the two parameters (*radius* and *minPts*) such that the density-based clustering method finds:

1. The 4 "true" clusters.

   Answer: radius = 2 minPts = 2

2. 3 clusters (merging the top two, which also seems like a reasonable interpretaition).

   Answer: radius = 5 minPts = 2

3. 2 clusters.

   Answer: radius = 15 minPts = 2

4. 1 cluster (consisting of the non-outlier points).

   Answer: radius = 20 minPts = 2

2 Matrix Notation and Linear Regression

2.1 Converting to Matrix/Vector/Norm Notation

Using our standard supervised learning notation (X, y, w) express the following functions in terms of vectors, matrices, and norms (there should be no summations or maximums).

1. $\sum_{i=1}^{n} |w^T x_i - y_i| + \lambda \sum_{j=1}^{d} |w_j|$.

   Answer: $\|w^T X - y\|_1 + \lambda \|w\|_1$

2. $\sum_{i=1}^{n} v_i (w^T x_i - y_i)^2 + \sum_{j=1}^{d} \lambda_j w_j^2$.

   Answer: $(w^T X - y)V \cdot (w^T X - y) + (\Lambda w) \cdot w$

3. $\left(\max_{i \in \{1,2,\ldots,n\}} |w^T x_i - y_i|\right)^2 + \frac{1}{2} \sum_{j=1}^{d} \lambda_j |w_j|$.

   Answer: $(\|w^T x - y\|_\infty)^2 + \frac{1}{2} \|\Lambda w\|_1$

You can use V to denote a diagonal matrix that has the (non-negative) "weights" v_i along the diagonal. The value λ (the "regularization parameter") is a non-negative scalar. You can Λ as a diagonal matrix that has the (non-negative) λ_j values along the diagonal.

2.2 Minimizing Quadratic Functions as Linear Systems

Write finding a minimizer w of the functions below as a system of linear equations (using vector/matrix notation and simplifying as much as possible). Note that all the functions below are convex so finding a w with $\nabla f(w) = 0$ is sufficient to minimize the functions (but show your work in getting to this point).

1. $f(w) = \frac{1}{2} \|w - u\|^2$ (projection of v onto real space).

   Answer: $f(w) = \frac{1}{2} \sum_{i=0}^{d} (w_i - u_i)^2$
   $\nabla f(w) = ((w_1 - u_i), (w_2 - u_i), \ldots)^T$
   Therefore Optimized $w = u$

2. $f(w) = \frac{1}{2} \sum_{i=1}^{n} v_i (w^T x_i - y_i)^2 + \lambda w^T u$ (weighted and tilted least squares).

   Answer: $f(w) = \frac{1}{2} \sum_{i=1}^{n} v_i ((\sum_{j=i}^{d} x_i^d * w_d) - y_i)^2 + \lambda \sum_{j=1}^{d} w_d u_d$
   $f(w) = \frac{1}{2} \sum_{i=1}^{n} v_i (\hat{y_i} - y_i)^2 + \lambda \sum_{j=1}^{d} w_d u_d = \frac{1}{2} \sum_{i=1}^{n} v_i (\hat{y_i}^2 - 2\hat{y_i} y_i + y_i^2) + \lambda \sum_{j=1}^{d} w_d u_d$
   $f(w) = \frac{1}{2} \sum_{i=1}^{n} v_i ((\sum_{j=1}^{d} w_j x_i^j)^2 - 2(\sum_{j=1}^{d} w_j x_i^j) y_i + y_i^2) + \lambda \sum_{j=1}^{d} w_d u_d$
   $f(w) = \frac{1}{2} (w^T X^T V X w) - W^T X^T V y + \frac{1}{2} y^T V y + \lambda w^T u$
   $\nabla f(w) = \frac{1}{2} w^T X^T V X + \frac{1}{2} w^T (X^T V X)^T - y^T V X + 0 + \lambda u$
   $\nabla f(w) = w^T X^T V X - y^T V X + \lambda u$ Therefore, the $w = (X^T V X) \setminus (X^T V y - \lambda u)$

3. $f(w) = \frac{1}{2} \|Xw - y\|^2 + \frac{\lambda}{2} \|w - w^0\|^2$ (least squares shrunk towards non-zero w^0).

   Answer: $f(w) = \frac{1}{2} w^T X^T X w - w^T X^T y + 1/2 y^T y + \frac{\lambda}{2} w^T w - \lambda w^T w^0 + \frac{\lambda}{2} w^{0T} w^0$
   $\nabla f(w) = X^T X w - X^T y + \lambda I w - \lambda w^0$
   $\nabla f(w) = (X^T X + \lambda I) w - X^T y - \lambda w^0$
   solve for w when $\nabla f(w) = 0$
   $w = (X^T X + \lambda I) \setminus (X^T y + \lambda w^0)$

Above we assume that u and w^0 are d by 1 vectors, that v is a n by 1 vector. You can use V as a diagonal matrix containing the v_i values along the diagonal.

Hint: Once you convert to vector/matrix notation, you can use the results from class to quickly compute these quantities term-wise. As a sanity check for your derivation, make sure that your results have the right dimensions. As a sanity check, make that the dimensions match for all quantities/operations. In order to make the dimensions match you may need to introduce an identity matrix. For example, $X^T X w + \lambda w$ can be re-written as $(X^T X + \lambda I) w$.

2.3 Convex Functions

Recall that convex loss functions are typically easier to minimize than non-convex functions, so it's important to be able to identify whether a function is convex.

Show that the following functions are convex:

1. $f(w) = \alpha w^2 - \beta w + \gamma$ with $w \in \mathbb{R}, \alpha \geq 0, \beta \in \mathbb{R}, \gamma \in \mathbb{R}$ (1D quadratic).

   Answer: $f''(x) = 2\alpha$ Because $\alpha \geq 0$ so $f''(x) \geq 0$ Therefore it is convex.

2. $f(w) = w \log(w)$ with $w > 0$ ("neg-entropy")

   Answer: $f'(w) = \log(w) + w * 1/w$
   $f''(w) = 1/w$ Because $w \geq 0$, then $f''(w) \geq 0$ So it is convex.

3. $f(w) = \|Xw - y\|^2 + \lambda \|w\|_1$ with $w \in \mathbb{R}^d, \lambda \geq 0$ (L1-regularized least squares).

   Answer: Because sum of convex functions are convex, and Norms and Square Norms are convex functions. Therefore, f(w) is convex.

4. $f(w) = \sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i))$ with $w \in \mathbb{R}^d$ (logistic regression).

   Answer: $g(z) = log(1 + \exp(z))$
   $g''(z) = \frac{\exp(z)^2}{(1+\exp(z))^2} \geq 0$ Therefore for any z, it is an convex function.
   So f(x) is convex.

5. $f(w) = \sum_{i=1}^{n} [\max\{0, |w^T x_i - y_i|\} - \epsilon] + \frac{\lambda}{2} \|w\|_2^2$ with $w \in \mathbb{R}^d, \epsilon \geq 0, \lambda \geq 0$ (support vector regression).

   Answer: Because $|w^T x_i - y_i| \geq 0$
   So $f(w) = \sum_{i=1}^{n} [|w^T x_i - y_i| - \epsilon] + \frac{\lambda}{2} \|w\|_2^2$
   We know that $\frac{\lambda}{2} \|w\|_2^2$ is convex, so we just need to show the previos part is convex.
   $g(w) = \sum_{i=1}^{n} [|w^T x_i - y_i| - \epsilon]$ We also know that $|w^T x - y|$ is convex for any x and y as the second derivative is always 0. So g(x) is convex as it is the sum of convex functions. So f(x) is convex.

General hint: for the first two you can check that the second derivative is non-negative since they are one-dimensional. For the last 3 you'll have to use some of the results regarding how combining convex functions can yield convex functions which can be found in the lecture slides.

Hint for part 4 (logistic regression): this function may seem non-convex since it contains $\log(z)$ and log is concave, but there is a flaw in that reasoning: for example $\log(\exp(z)) = z$ is convex despite containing a log. To show convexity, you can reduce the problem to showing that $\log(1 + \exp(z))$ is convex, which can be done by computing the second derivative. It may simplify matters to note that $\frac{\exp(z)}{1+\exp(z)} = \frac{1}{1+\exp(-z)}$.

3 Linear and Nonlinear Regression

If you run the script *example_nonLinear*, it will:

1. Load a one-dimensional regression dataset.
2. Fit a least-squares linear regression model.
3. Report the training error.
4. Report the test error (on a dataset not used for training).
5. Draw a figure showing the training data and what the linear model looks like.

Unfortunately, this is an awful model of the data. The average squared training error on the data set is over 28000 (as is the test error), and the figure produced by the demo confirms that the predictions are usually nowhere near the training data:

3.1 Linear Regresion with Bias Variable

The y-intercept of this data is clearly not zero (it looks like it's closer to 200), so we should expect to improve performance by adding a *bias* variable, so that our model is

$$y_i = w^T x_i + w_0.$$

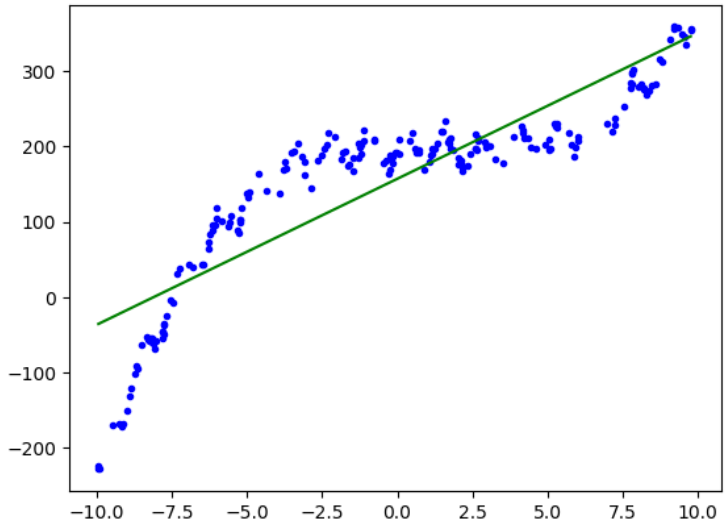
instead of

$$y_i = w^T x_i.$$

Write a new function, *leastSquaresBias*, that has the same input/model/predict format as the *leastSquares* function, but that adds a *bias* variable w_0. Hand in your new function, the updated plot, and the updated training/test error.

Hint: recall that adding a bias w_0 is equivalent to adding a column of ones to the matrix X. Don't forget that you need to do the same transformation in the *predict* function.

Answer:




```
Squared train Error with least squares: 3551.346
Squared test Error with least squares: 3393.869
1-element Array{PyCall.PyObject,1}:
 PyObject <matplotlib.lines.Line2D object at 0x00000000020D37F0>
```

```
include("misc.jl")

function leastSquares(X,y)
    n = size(X,1)
    v = ones(n,1)
    R = [X v]
    # Find regression weights minimizing squared error
    w = (R'R)\(R'y)

    # Make linear prediction function
    function predict(Xhat)
        n1 = size(Xhat,1)
        v = ones(n1,1)
        R = [Xhat v]
        return R*w
    end
    # Return model
    return GenericModel(predict)
end
```

3.2 Linear Regression with Polynomial Basis

Adding a bias variable improves the prediction substantially, but the model is still problematic because the target seems to be a *non-linear* function of the input. Write a new function, *leastSquaresBasis(x,y,p)*, that takes a data vector x (i.e., assuming we only have one feature) and the polynomial order p. The function should perform a least squares fit based on a matrix Z where each of its rows contains the values $(x_i)^j$ for $j = 0$ up to p. E.g., *leastSquaresBasis(x,y,3)* should form the matrix

$$Z = \begin{bmatrix} 1 & x_1 & (x_1)^2 & (x_1)^3 \\ 1 & x_2 & (x_2)^2 & (x_2)^3 \\ \vdots & & & \\ 1 & x_n & (x_n)^2 & (x_N)^3 \end{bmatrix},$$

and fit a least squares model based on it. Hand in the new function, and report the training and test error for $p = 0$ through $p = 10$. Explain the effect of p on the training error and on the test error.

Answer:

```
Squared train Error with p = 0: 15480.520
Squared test Error with p = 0: 14390.763
Squared train Error with p = 1: 3551.346
Squared test Error with p = 1: 3393.869
Squared train Error with p = 2: 2167.992
Squared test Error with p = 2: 2480.725
Squared train Error with p = 3: 252.046
Squared test Error with p = 3: 242.805
Squared train Error with p = 4: 251.462
Squared test Error with p = 4: 242.126
Squared train Error with p = 5: 251.143
Squared test Error with p = 5: 239.545
Squared train Error with p = 6: 248.583
Squared test Error with p = 6: 246.005
Squared train Error with p = 7: 247.011
Squared test Error with p = 7: 242.888
Squared train Error with p = 8: 241.306
Squared test Error with p = 8: 245.966
Squared train Error with p = 9: 235.762
Squared test Error with p = 9: 259.296
Squared train Error with p = 10: 235.074
Squared test Error with p = 10: 256.300
```

```
function leastSquares(X,y,d)
    n = size(X,1)
    R = pow(X,d)
    # Find regression weights minimizing squared error
    w = (R'R)\(R'y)

    # Make linear prediction function
    function predict(Xhat)
        n1 = size(Xhat,1)
        R = pow(Xhat,d)
        return R*w
    end
    # Return model
    return GenericModel(predict)
end

function pow(x,d)
    r = x[:,1]
    ans = ones(size(x))
    for i in 1:d
        g =r.^i
        ans = [ans g]
    end
    return ans
end
```

Note: for this question we'll assume $d = 1$ (we'll discuss polynomial bases with more input features later in the course).

Hints: To keep the code simple and reduce the chance of having errors, you may want to write a new function *polyBasis* that you can use for transforming both the training and testing data.

3.3 Manual Search for Optimal Basis

Polynomials are a flexible class of functions, but there is structure in this data that is not well-modelled by polynomials. Try to find a nonlinear basis that gives the best performance on this dataset in terms of test error. Report the basis that you use and the training/test score that you achieve.

Hint: the data seems to have periodic behaviour, and it's possible to obtain training and test errors below 60.

Answer: The minimum Test Error i get is 51.061613323917456.
The bases are $x, x^2, x^3, sin(x/0.2), sin(x^2/0.2), sin(x^3/0.2)$

4 Robust Regression and Gradient Descent

The script *example_outliers* loads a one-dimensional regression dataset that has a non-trivial number of 'outlier' data points. These points do not fit the general trend of the rest of the data, and pull the least squares model away from the main downward trend that most data points exhibit:

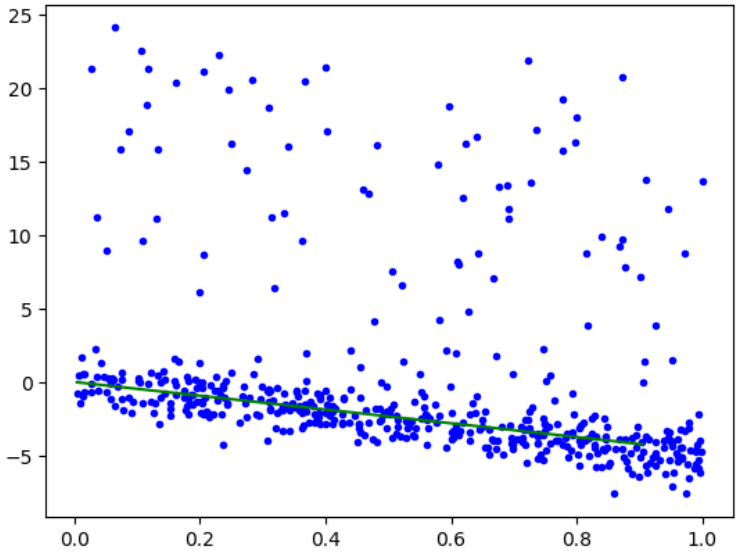
4.1 Weighted Least Squares in One Dimension

One of the most common variations on least squares is *weighted* least squares. In this formulation, we have a weight v_i for every training example. To fit the model, we minimize the weighted squared error,

$$f(w) = \frac{1}{2}\sum_{i=1}^{n} v_i(w^T x_i - y_i)^2.$$

In this formulation, the model focuses on making the error small for examples i where v_i is high. Similarly, if v_i is low then the model allows a larger error.

Write a model function, *weightedLeastSquares(X,y,v)*, that implements this model (note that this can be solved as a linear system). Apply this model to the data containing outliers, setting $v_i = 1$ for the first 400 data points and $v_i = 0.1$ for the last 100 data points (which are the outliers). Hand in your function and the updated plot.



Answer:

```
function weightedLeastSquares(X,y,v)
    X = X.*v
    # Find regression weights minimizing squared error
    w = (X'X)\(X'y)

    # Make linear prediction function
    predict(Xhat) = Xhat*w

    # Return model
    return GenericModel(predict)
end
```

4.2 Smooth Approximation to the L1-Norm

Unfortunately, we typically do not know the identities of the outliers. In situations where we suspect that there are outliers, but we do not know which examples are outliers, it makes sense to use a loss function that is more robust to outliers. In class, we discussed using the Huber loss,

$$f(w) = \sum_{i=1}^{n} h(w^T x_i - y_i),$$

where

$$h(r_i) = \begin{cases} \frac{1}{2} r_i^2 & \text{for } |r_i| \leq \epsilon \\ \epsilon(|r_i| - \frac{1}{2}\epsilon) & \text{otherwise} \end{cases}.$$

This is less sensitive to outliers than least squares, although it can no longer be minimized by solving a linear system. Derive the gradient ∇f of this function with respect to w. You should show your work but you do not have to express the final result in matrix notation. Hint: you can start by computing the derivative of h with respect to r_i and then get the gradient using the chain rule. You can use $\text{sgn}(r_i)$ as a function that returns 1 if r_i is positive and -1 if it is negative.

Answer: first, compute the derivative of function h.

$$h'(r_i) = \begin{cases} r_i & \text{for } |r_i| \leq \epsilon \\ sgn(r_i)\epsilon & \text{otherwise} \end{cases}$$

$$h'(r_i) = sgn(r_i)\min\{\epsilon, r_i\}$$

Then, calculate the gradiant of f.

$$\begin{aligned}\nabla f(w) &= \sum_{i=0}^{n} sgn(r_i)\min\{\epsilon, r_i\}(W^T x_i - y_i)' \\ &= \sum_{i=0}^{n} sgn(W^T x_i - y_i)\min\{\epsilon, W^T x_i - y_i\}x_i\end{aligned}$$

4.3 Robust Regression

The function *example_gradient* is the same as *example_outlier*, except that it fits the least squares model using a *gradient descent* method. You'll see that it produces the same fit as we obtained using the normal equations.

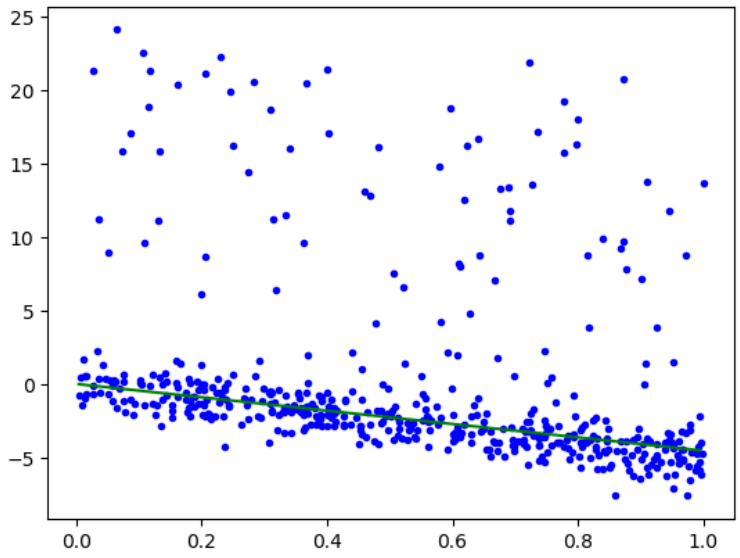
The typical input to a gradient method is a function that, given w, returns $f(w)$ and $\nabla f(w)$. See *funObj* in the *leastSquaresGradient* function for an example. Note that *leastSquaresGradient* also has a numerical check that the gradient code is approximately correct, since implementing gradients is often error-prone.[1]

An advantage of gradient-based strategies is that they are able to solve problems that do not have closed-form solutions, such as the formulation from the previous section. The function *robustRegression* has most of the implementation of a gradient-based strategy for fitting the Huber regression model. The only part missing is the function and gradient calculation inside the *funObj* code. Modify this function to implement the objective function and gradient based on the Huber loss (from the previous section). Hand in your code, as well as the plot obtained using this robust regression approach with $\epsilon = 1$.

Answer:

```
function robustRegressionObj(w,X,y)
    n = size(X)[1]

    f=0
    g = zeros(size(w))
    for i in 1:n
        r = dot(X[i,:],w) - y[i]
        if abs(r) > 1
            f = f + (abs(r) - 0.5)
        else
            f = f + 0.5*r*r
        end
        g = g + sign(r) *min(abs(r),1)* transpose(X[i,:])
    end

    return (f,g)
end
```

[1] Though sometimes the numerical gradient checker itself can be wrong. For a lot more on numerical differentiation you can take CPSC 303.



5 Very-Short Answer Questions

1. Describe a dataset with k clusters where k-means cannot find the true clusters.

   Answer: Non-convex clusters. Two U shaped clusters biting together.

2. Why do we need random restarts for k-means but not for density-based clustering?

   Answer: Because there is a chance that the random staer drop into a local minimum, and will not be the true clusters.

3. Can hierarchical clustering find non-convex clusters?

   Answer: Yes, if we are able to say seperate the child claster from the parent, then the remaining parent is a non-convex claster.

4. For each outlier detection method below, list an example method and a problem with identifying outliers using this method:

   - Model-based outlier detection.

     Answer: Mechod: the one with a z-score higher than 5 or lower than -5 is an outlier.
     Problem: Assumes it is unimodel.

   - Graphical-based outlier detection.

     Answer: Plot a scatter plot of the data, and spot the outlier visually.
     Problem: It is hard to do if number of features is large.

   - Supervised outlier detection.

     Answer: Use supervised learning to train the detector on the training set and try to perfict the outlier.
     Problem: only able to detect certain known outliers in the training set.

5. In linear regression, why do we compute the squared error $(y_i - \hat{y}_i)^2$ and not test the equality $(y_i = \hat{y}_i)$?

   Answer: Because the euqlity is hard to achive. And we can calculate the derivative of the function of squared error, but not with the equality which is useful to find local minimum.

6. Describe a simple 2-feature ($d = 2$) case where the least squares estimate would not be unique.

   Answer: $X = (1, 1) y = (1)$. The possible w is (1,0), (0,1) ...

7. Why do we typically add a column of 1 values to X when we do linear regression? Should we do this if we're using decision trees?

   Answer: When we have an offset on y, that is when x = 0 but y != 0. No because in decision tree, the added column 1 is not related to the y at all. If we have a decision stump on that new column, all examples will go to one branch which is useless.

8. When should we consider using gradient descent to approximate the solution to the least squares problem instead of exactly solving it with the closed form solution?

   Answer: When the d is large, solving the equation is slow, we need to consider the gradiant decent solution.

9. If a function is convex, what does that say about stationary points of the function? Does convexity imply that a stationary points exists?

   Answer: The convex function means that the stationary point is also the global minimum, but it doesn't make sure that there exists a stationary point.

10. For robust regression based on the L1-norm error, why can't we just set the gradient to 0 and solve a linear system? In this setting, why we would want to use a smooth approximation to the absolute value?

    Answer: Because there is a point in the function that the deravitive of the function does not exist, which means, the gradiant decent will not work there. So we use sooth approximation to make sure that the gradiant is continuous.

11. What is the problem with having too small of a learning rate in gradient descent?

    Answer: The steps takes too long to solve the problem.

12. What is the problem with having too large of a learning rate in gradient descent?

    Answer: We might step over the solution.

Project Proposal (OPTIONAL FOR 340 STUDENTS)

For 532M students, there is a project component to the course that will be worth 20% of your final grade. For 340 students, there is no requirement to do a project. However, 340 students have the option to do a project anyway for the possibility of obtaining a higher grade: your project grade can replace either your 2 lowest assignment scores or your midterm score (whicheve helps you more).[2]

This semester you will have the option of doing two styles of projects:

1. **Small Projects**: These projects are done in groups of 2-3. The final deliverable will be a 6-page report that is due near the end of the exam period (something like December 16th). It is expected that this project will be a literature survey, but research projects are also ok.
2. **Big Projects**: These projects are done in groups of approximately 5 (4 or 6 is ok if needed but I would prefer you aim for 5). The final deliverable will be a poster to be presented December 8th from 4-7pm at a hotel downtown (probably the Fairmont Waterfront). Students from a similar class at SFU will also be presenting their posters during this time, and over 50 companies (looking to potentially hire people like you) will also be there.

[2]The course is *not* graded on a curve, so 340 students are not hurt by choosing the to skip the project.

There aren't really any restrictions on the group compositions: 340 students can work with 532M students, auditors can work with registered students, and you can combine this project with a project from another one of your classes (assuming you get the other instructor's permission, and even if not all students in the other class are registered in this class). The only combinations I really want to avoid are having students do projects with the TAs (due to the obvious conflict of interest), project groups that have no students enrolled in 340 or 532M, or projects that contain people taking no CPSC classes.

If you are in 532M, or in 340 and want to do a project, for the final part of this assignment you must a submit a project proposal for your course project. The proposal should be a maximum of 2 pages (and 1 page or half of a page is ok if you can describe your plan concisely). The proposal should be written for the instructors and the TAs, so you don't need to introduce any ML background but you will need to introduce non-ML topics.

You should submit this question as a group on Gradescope, separate from the other assignment questions.

There is quite a bit of flexibility in terms of the type of project you do, as I believe there are many ways that people can make valuable contributions to research. However, note that ultimately the final deliverable for the project will be a report/poster that emphasizes a particular "contribution" (i.e., what doing the project has added to the world). The reason for this, even though it's strange for some possible projects, is that this is the standard way that results are communicated to the research community.

The three mains ingredients of the project proposal are:

1. What problem you are focusing on.
2. What you plan to do.
3. What will be the "contribution".

Also, for the course project note that negative results (i.e., we tried something that we thought we would work in a particular setting but it didn't work) are acceptable (and often unavoidable).

If you are doing a small project, the default "template" for the project is the following:

1. **Literature review**: you pick a specific topic in ML, read at least 10 papers on the topic, then write a report summarizing what has been done on the topic and what are the most promising directions of future work. In this case, the contribution would be your summary of the relationships between the existing works, and your insights about where the field is going.

Here are some standard "templates" for a big project (you also have the option of using these for small projects):

2. **Application bake-off**: you pick a specific application (from your research, personal interests, or maybe from Kaggle) or a small number of related applications, and try out a bunch of techniques (e.g., random forests vs. logistic regression vs. generative models). In this case, the contribution would be showing that some methods work better than others for this specific application (or your contribution could be that everything works equally well/badly).
3. **New application**: you pick an application where people aren't using ML, and you test out whether ML methods are effective for the task. In this case, the contribution would be knowing whether ML is suitable for the task.
4. **Scaling up**: you pick a specific machine learning technique, and you try to figure out how to make it run faster or on larger datasets. In this case, the contribution would be the new technique and an evaluation of its performance, or could be a comparison of different ways to address the problem.
5. **Improving performance**: you pick a specific machine learning technique, and try to extend it in some way to improve its performance. In this case, the contribution would be the new technique and an evaluation of its performance.

6. **Generalization to new setting**: you pick a specific machine learning technique, and try to extend it to a new setting (for example, making a multi-label version of random forests). In this case, the contribution would be the new technique and an evaluation of its performance, or could be a comparison of different ways to address the problem.
7. **Coding project**: you pick a specific method or set of methods, and build an implementation of them. In this case, the contribution could be the implementation itself or a comparison of different ways to solve the problem.
8. **Theory**: you pick a theoretical topic (like the variance of cross-validation), read what has been done about it, and try to prove a new result (usually by relaxing existing assumptions or adding new assumptions). The contribution could be a new analysis of an existing method, or why some approaches to analyzing the method will not work.
9. **Reproducibility Challenge**: you take part in the 2019 NeurIPS reproducibility challenge, where you try to reproduce the results of a recently-submitted machine learning paper. Information on the challenge is available here (watch out for the early deadline if you want to officially take part): `https://reproducibility-challenge.github.io/neurips2019`

The above are just suggestions, and many projects will mix several of these templates together, but if you are having trouble getting going then it's best to stick with one of the above templates. Also note that the project can focus on topics not covered in the course (like RNNs), so there is flexibility in the topic, but the topic should be closely-related to ML.

This question is mandatory but will not be formally marked: it's just a sanity check that you have at least one project idea that has an appropriate topic and scope, that you find a group early, and that you allocate some time to thinking about the project. Also, there is flexibility in the choice of project topics even after the proposal: if you want to explore different topics you can ultimately choose to do a project that is unrelated to the one in your proposal (and changing groups is ok too). If you aren't sure what to do, go bug the TAs in office hours (which is a good idea even if you are sure what you want to).