Mini Project 3

ECE4530J - Decision Making in Smart Cities Summer 2022

∗ Name: Huang Yucheng ID: 519021910885

Problem 1

Suppose that the response y is generated by

$$y = f(x) + \epsilon,$$

where ϵ is a zero-mean Gaussian noise with variance 1 .

a) Suppose that $f(x) = x$. Randomly generate $10x$'s and generate the corresponding y 's; you need to generate two random numbers (i.e., x and ϵ for each of the 10 points). Fit the data with linear regression and plot the scatter points.

b) Suppose that $f(x) = x^2$. Randomly generate $10(x, y)$ pairs. Fit the data with linear regression and plot the scatter points.

c) Suppose that $f(x) = 1/x$. Randomly generate $10(x, y)$ pairs. Fit the data with linear regression and plot the scatter points.

Answer:

a) We do the linear regression and find that $y = 1.1022x - 1.096$, at this condition, the error is very low and the points are scattered nearby the reference line.

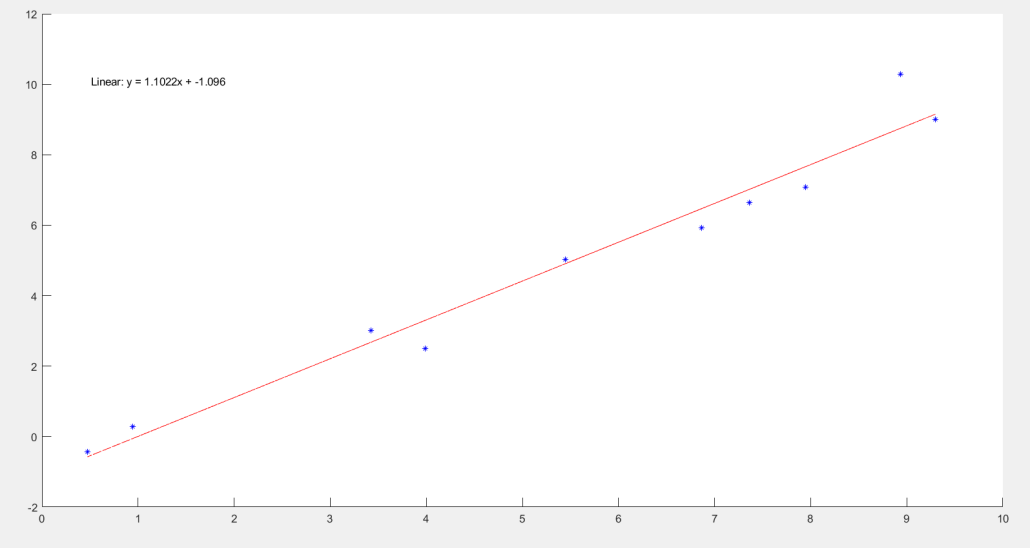

Figure 1: Figure for Problem 1a

```
noise = randn(1, 10);
x = 10*rand(1, 10);
y1 = x + noise;
scatter(x, y1, 25, 'b', '*')
P = polyfit(x, y1, 1);
yfit = polyval(P, x);
hold on;
plot(x, yfit, 'r-.');
eqn = string(" Linear: y = " + P(1)) + "x + " + string(P(2));
text(min(x), max(y1), eqn, "HorizontalAlignment", "left", "VerticalAlignment",
```

Figure 2: Code for Problem 1a

b) We do the linear regression and find that $y = 9.8431x - 13.0824$, at this condition, we found that some small numbers are above the line, and some medium numbers are below the line. This is because it should be a quadratic curve, and there will be deviations when fitting a straight line.

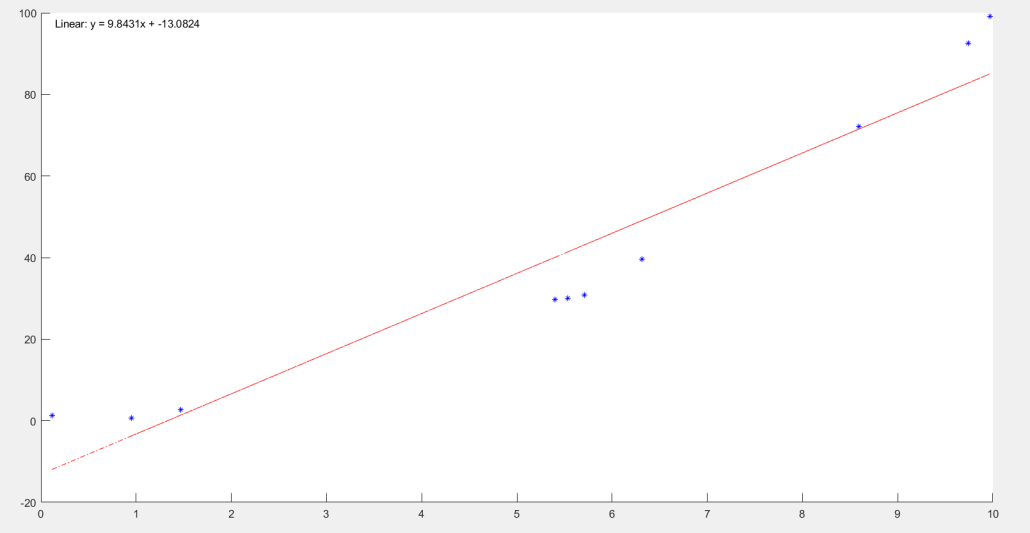

Figure 3: Figure for Problem 2a

```
noise = randn(1, 10);
x = 10*rand(1, 10);
y1 = x.^2 + noise;
scatter(x, y1, 25, 'b', '*')
P = polyfit(x, y1, 1);
yfit = polyval(P, x);
hold on;
plot(x, yfit, 'r-.');
eqn = string(" Linear: y = " + P(1)) + "x + " + string(P(2));
text(min(x), max(y1), eqn, "HorizontalAlignment", "left", "VerticalAlignment",
```

Figure 4: Code for Problem 2a

c) We do the linear regression and find that $y = -0.46523x + 3.3288$, at this condition, we found that it deviates completely from the original image, this is because this is a hyperbolic function, it does not conform to the law of linear fitting at all, it is meaningless if linear fitting is performed.

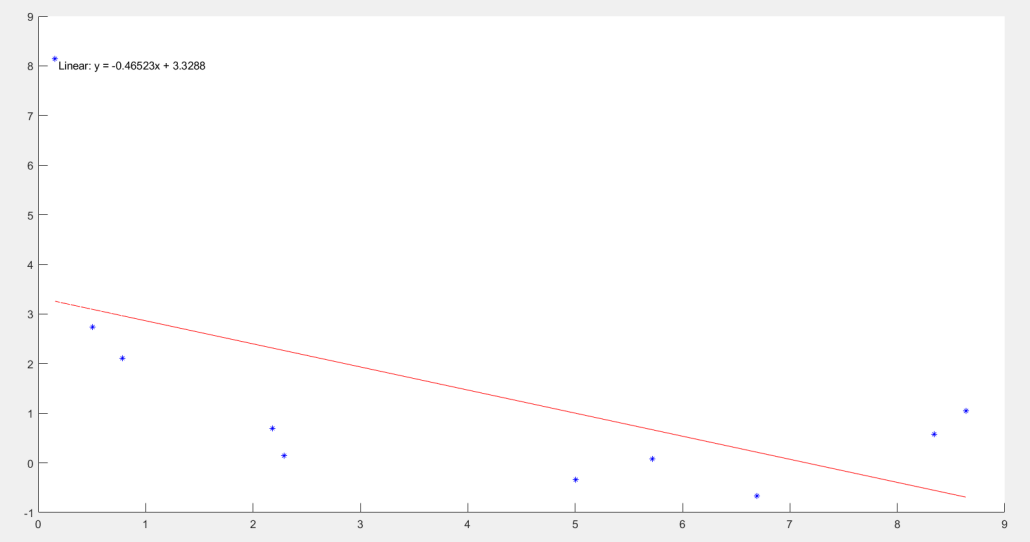

Figure 5: Figure for Problem 3a

```
noise = randn(1, 10);
x = 10*rand(1, 10);
y1 = 1./x + noise;
scatter(x, y1, 25, 'b', '*')
P = polyfit(x, y1, 1);
yfit = polyval(P, x);
hold on;
plot(x, yfit, 'r-.');
eqn = string(" Linear: y = " + P(1)) + "x + " + string(P(2));
text(min(x), max(y1), eqn, "HorizontalAlignment", "left", "VerticalAlignment",
```

Figure 6: Code for Problem 3a

Problem 2

Consider the data in the attachment "MP3P2.xlsx".

a) Show that linear regression does not work for this classification problem.

b) Use the logit function to train a linear classifier. Indicate the boundaries that you find. Report the misclassification rate.

Answer:

a) We do the linear regression and find that $y = 0.89573x + 0.16718$, linear regression is pointless at this point, because the line divides the image into two parts, but it passes through the three types of points and does not separate the three types.

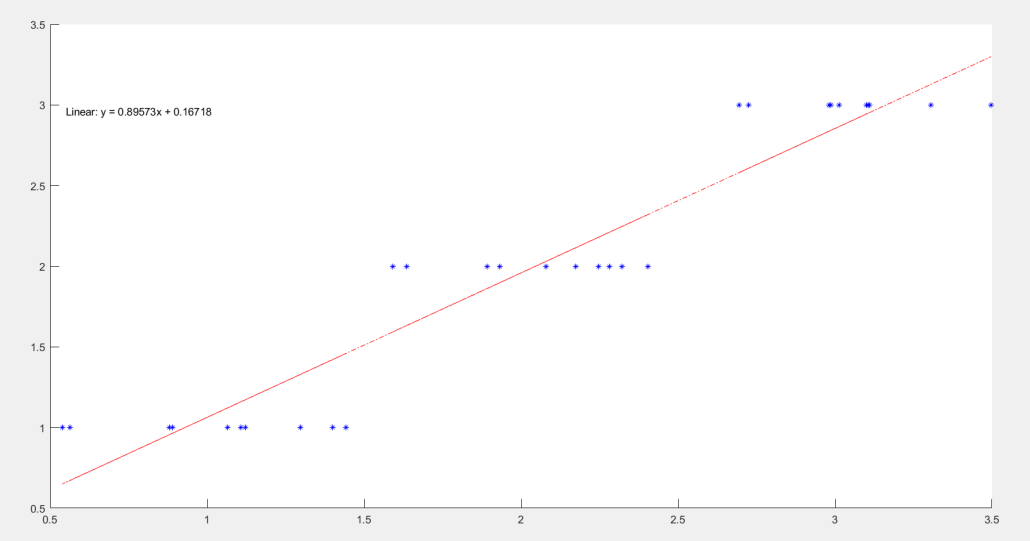

Figure 7: Figure for Problem 2a

```
noise = randn(1, 10);
x=[1.10831869000000 1.06556218700000     1.29700806300000     0.5395750530(
y1=[1   1   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2
scatter(x, y1, 25, 'b', '*')
P = polyfit(x, y1, 1);
yfit = polyval(P, x);
hold on;
plot(x, yfit, 'r-.');
eqn = string(" Linear: y = " + P(1)) + "x + " + string(P(2));
text(min(x), max(y1), eqn, "HorizontalAlignment", "left", "VerticalAlignment",
```

Figure 8: Code for Problem 2a

b) We do the linear classification and find that $y = 1.1022x - 1.096$, at this condition, we found that the linear classification draws two lines, which perfectly divide the three point sets into three parts, and the misclassification rate is 0 because the two lines do not pass through any point.

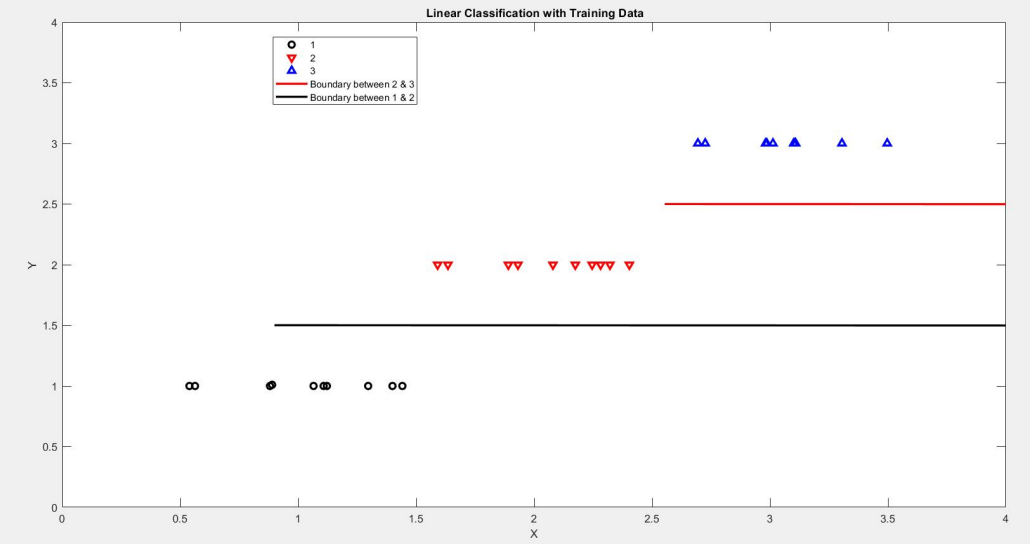

Figure 9: Figure for Problem 2b

```
h1 = gscatter(PL,PW,Z,'krb','ov^',[],'off');
h1(1).LineWidth = 2;
h1(2).LineWidth = 2;
h1(3).LineWidth = 2;
legend('1','2','3','Location','best')
hold on
X = [PL,PW];
MdlLinear = fitcdiscr(X,Z);
MdlLinear.ClassNames([2 3])
K = MdlLinear.Coeffs(2,3).Const;
L = MdlLinear.Coeffs(2,3).Linear;
f = @(x1,x2) K + L(1)*x1 + L(2)*x2;
h2 = fimplicit(f,[.9 7.1 0 2.5]);
h2.Color = 'r';
h2.LineWidth = 2;
h2.DisplayName = 'Boundary between 2 & 3';
MdlLinear.ClassNames([1 2])
K = MdlLinear.Coeffs(1,2).Const;
L = MdlLinear.Coeffs(1,2).Linear;
f = @(x1,x2) K + L(1)*x1 + L(2)*x2;
h3 = fimplicit(f,[.9 7.1 0 2.5]);
h3.Color = 'k';
h3.LineWidth = 2;
h3.DisplayName = 'Boundary between 1 & 2';
axis([0 4 0 4])
xlabel('X')
ylabel('Y')
title('{\bf Linear Classification with Training Data}')
```

Figure 10: Code for Problem 2b