به نام خدا

دانشگاه صنعتی امیرکبیر

دانشکده مهندسی کامپیوتر

پاسخ تمرین سری سوم شناسایی آماری الگو

استاد:

دکتر رحمتی

دانشجو:

حلیمه رحیمی

شماره دانشجویی:

۹۹۱۳۱۰۴۳

زمستان ۱۳۹۹

اصلاحات صورت گرفته:

در تمرین های پیاده سازی، تصاویر مربوط به کدها و نتایج را نگذاشته بودم. همچنین مطالب کوتاهی در مورد کد ها اضافه کردم و برخی کد ها را تا حدودی کامنت گذاری کردم.

Assignment III

1. Is Oxford/AstraZeneca Vaccine Safe?

Volunteer	SBP (mmHg)		Blood pH	
1	134.11 ± 5.67	128.44 – 139.78	7.38 ± 0.01	7.37 – 7.39
2	129.53 ± 3.44	126.09 – 132.97	7.43 ± 0.00	7.43
3	142.81 ± 4.92	137.89 – 147.73	7.42 ± 0.01	7.41 – 7.43
4	130.26 ± 4.88	125.38 – 135.14	7.40 ± 0.02	7.38 – 7.42
5	118.43 ± 2.83	115.6 – 121.26	7.38 ± 0.02	7.36 – 7.40
6	144.40 ± 2.51	141.89 – 146.91	7.36 ± 0.00	7.36
7	126.20 ± 4.37	121.83 – 130.57	7.41 ± 0.03	7.38 – 7.44
8	137.05 ± 1.17	135.88 – 138.22	7.40 ± 0.01	7.39 – 7.41
9	114.63 ± 7.21	107.42 – 121.84	7.39 ± 0.02	7.37 – 7.41
10	151.42 ± 3.10	148.32 – 154.52	7.37 ± 0.01	7.36 – 7.38
11	132.29 ± 8.67	123.62 – 140.96	7.41 ± 0.00	7.41
12	164.05 ± 6.12	157.93 – 170.17	7.40 ± 0.01	7.39 – 7.41

a. Draw a histogram for both measurements using appropriate bins centres with the width of one.

SBP: very low-normal (<120 mmHg), low-normal (between 120–130 mmHg), high-normal (between 130–140 mmHg), high (between 140–150 mmHg), very high (>150 mmHg) ⊛

(I've circled the highest and lowest value in each column.)

Based on the fact about SBP ⊛ I've considered each bin width equal to 10 mmHg and 1 unit.

In order to face the error values in each cell, I've considered a continuos uniform distribution to get the probability of each volunteer having a specific SBP. So basically what I'm doing to calculate the number of samples in each bin is that : If a volunteer's range of SBP is compeletely in that bin, 1 number of samples is added to the amount of samples in the bin else a portion is added.

Based on the lowest and highest values and bin width, the edges are : ...100 _ 110 - 120 _ 130 _ 140 _ 150 _ 160 _ 170 - 180 - ... mmHg

#1 : $128.44 \xrightarrow[1.58]{} 130 \xrightarrow[9.78]{} 139.78$ #2 : $126.09 \xrightarrow[3.91]{} 130 \xrightarrow[2.97]{} 132.97$

#3 : $137.89 \xrightarrow[2.11]{} 140 \xrightarrow[7.73]{} 147.73$ #4 : $125.38 \xrightarrow[4.62]{} 130 \xrightarrow[5.14]{} 135.14$

#5 : $115.6 \xrightarrow[4.4]{} 120 \xrightarrow[1.26]{} 121.26$ #6 : $141.89 \xrightarrow[5.02]{} 146.91$

#7 : $121.83 \xrightarrow[8.17]{} 130 \xrightarrow[0.57]{} 130.57$ #8 : $135.88 \xrightarrow[2.34]{} 138.22$

#9: 107.42 $\xrightarrow[2.58]{}$ 110 $\xrightarrow[10]{}$ 120 $\xrightarrow[1.84]{}$ 121.84 #10: 148.32 $\xrightarrow[1.68]{}$ 150 $\xrightarrow[4.52]{}$ 154.52

#11: 123.62 $\xrightarrow[6.38]{}$ 130 $\xrightarrow[10]{}$ 140 $\xrightarrow[0.96]{}$ 140.96 #12: 157.93 $\xrightarrow[2.07]{}$ 160 $\xrightarrow[10]{}$ 170 $\xrightarrow[0.17]{}$ 170.17

bins:

<100 : 0

100 - 110: $\frac{2.58}{2\times 7.21}$

110 - 120: $\frac{4.4}{2\times 2.83} + \frac{10}{2\times 7.21}$

120 - 130: $\frac{1.58}{2\times 5.67} + \frac{3.91}{2\times 3.44} + \frac{4.62}{2\times 4.88} + \frac{8.17}{2\times 4.37} + \frac{6.38}{2\times 8.67} + \frac{1.26}{2\times 2.83} + \frac{1.84}{2\times 7.21}$

130 - 140: $\frac{2.11}{2\times 4.92} + \frac{10}{2\times 8.67} + \frac{2.34}{2.34} + \frac{0.57}{2\times 4.37} + \frac{5.14}{2\times 4.88} + \frac{2.97}{2\times 3.44} + \frac{9.78}{2\times 5.67}$

140 - 150: $\frac{1.68}{2\times 3.10} + \frac{0.96}{2\times 8.67} + \frac{5.02}{5.02} + \frac{7.73}{2\times 4.92}$

150 - 160: $\frac{2.07}{2\times 6.12} + \frac{4.52}{2\times 3.10}$

160 - 170: $\frac{10}{2\times 6.12}$

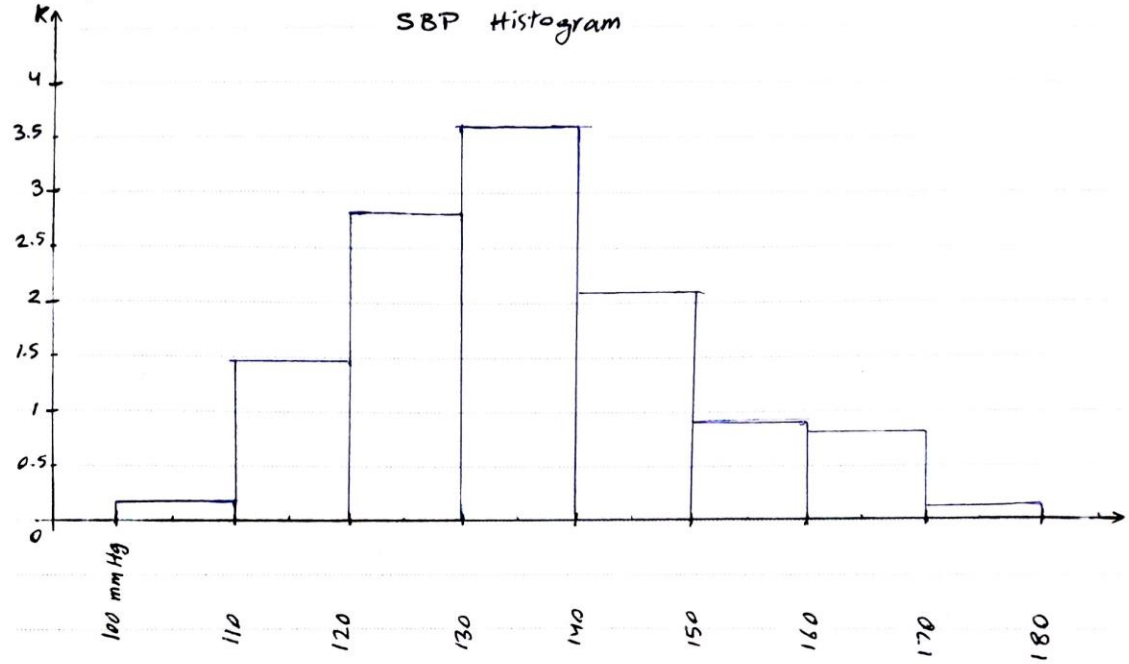
170 - 180: $\frac{0.17}{2\times 6.12}$

180< : 0

<100	100-110	110-120	120-130	130-140	140-150	150-160	160-170	170-180	180<
0	0.1789	1.4708	2.8339	3.6771	2.1119	0.8981	0.8169	0.0138	0

Since I have used rounding in my calculations these numbers are not exact, so I'm rounding sum of these numbers and as expected it is equal to 12 (we have 12 volunteers).

SBP Histogram



Blood pH in human body takes values between 7.35 and 7.45.

Based on the fact about Blood pH I've considered each bin width equal to 0.01 and 1 unit.

$< 7.36 : 0$

$7.36 - 7.37 : 1 + \frac{1}{4} + \frac{1}{2}$

$7.37 - 7.38 : \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{2}$

$7.38 - 7.39: \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{6} + \frac{1}{4}$

$7.39 - 7.40: \frac{1}{4} + \frac{1}{4} + \frac{1}{6} + \frac{1}{2} + \frac{1}{4} + \frac{1}{2}$

$7.40 - 7.41: \frac{1}{4} + \frac{1}{6} + \frac{1}{2} + \frac{1}{4} + \frac{1}{2}$

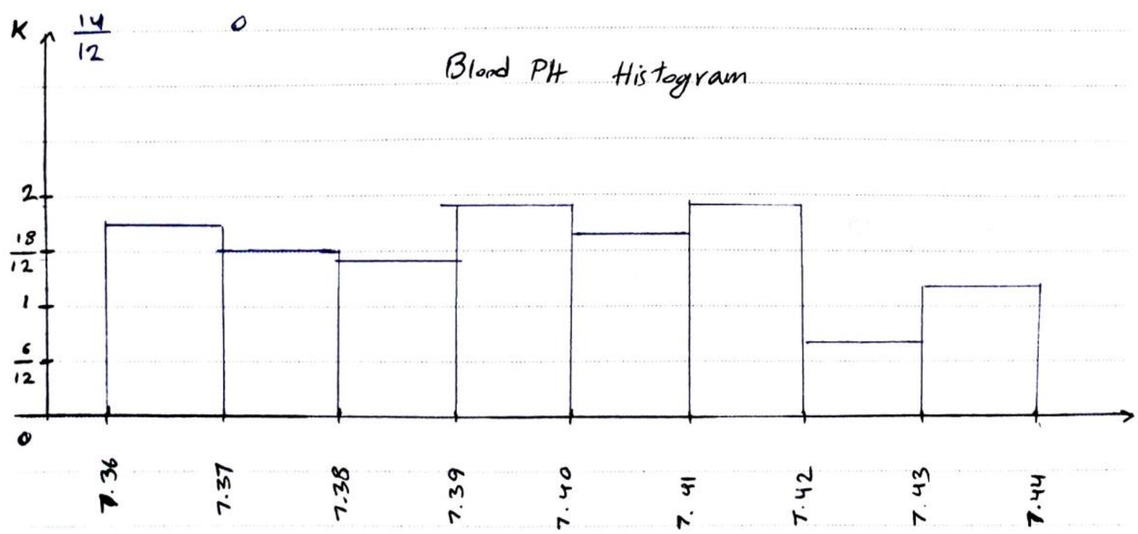
$7.41 - 7.42: \frac{1}{2} + \frac{1}{4} + \frac{1}{6} + 1$

$7.42 - 7.43: \frac{1}{2} + \frac{1}{6}$

$7.43 - 7.44: 1 + \frac{1}{6}$

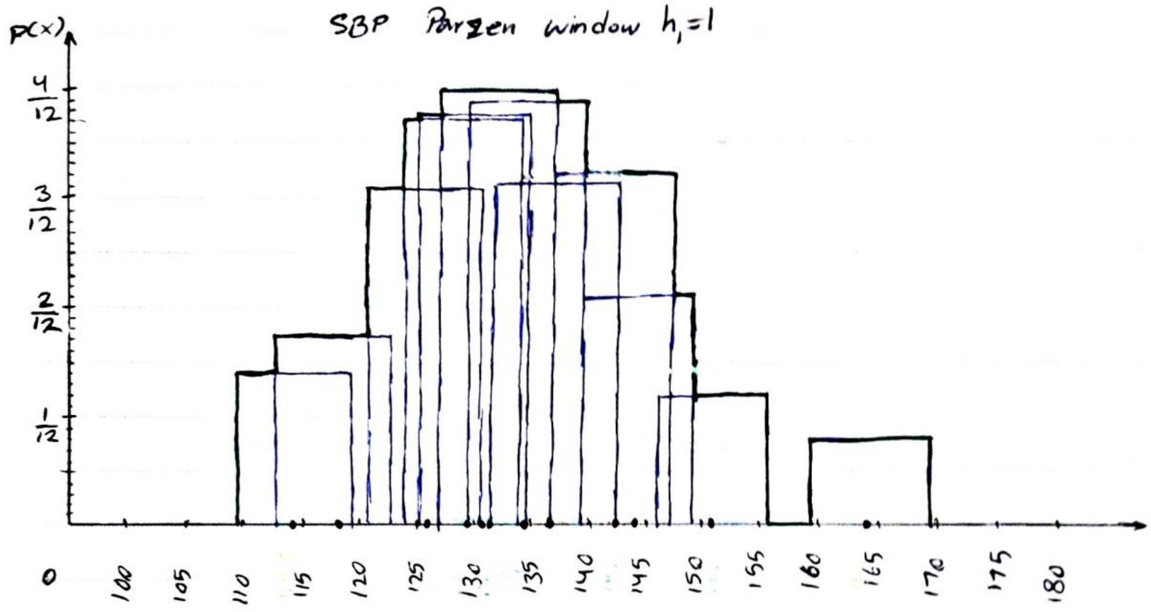
$7.44 < : 0$

<7.36	7.36 - 7.37	7.37 - 7.38	7.38 - 7.39	7.39 - 7.40	7.40 - 7.41	7.41 - 7.42	7.42 - 7.43	7.43 - 7.44	7.44<
0	$\frac{21}{12}$	$\frac{18}{12}$	$\frac{17}{12}$	$\frac{23}{12}$	$\frac{20}{12}$	$\frac{23}{12}$	$\frac{8}{12}$	$\frac{14}{12}$	0



b. Sketch the Parzen window estimate of the two unknown density functions for $h_1 = 1$ and $h_2 = 3$.

Parzen window $h_1 = 1$ for SBP:



$114.63 : \{109.63 - 119.63\}$:

$$\frac{10}{2 \times 7.21} + \frac{119.63 - 115.6}{2 \times 2.83}$$

$118.43 : \{113.43 - 123.43\}$:

$$\frac{121.84 - 113.43}{2 \times 7.21} + \underbrace{\frac{121.26 - 115.6}{2 \times 2.83}}_{1} + \frac{123.43 - 121.83}{2 \times 4.37}$$

$126.20 : \{121.20 - 131.20\}$:

$$\frac{131.20 - 128.44}{2 \times 5.67} + \frac{131.20 - 126.09}{2 \times 3.44} + \frac{131.20 - 125.38}{2 \times 4.88} + \frac{121.26 - 121.20}{2 \times 2.83} +$$

→ next page

$$\underbrace{\frac{130.57-121.83}{2\times 4.37}}_{1}+\frac{121.84-121.20}{2\times 7.21}+\frac{131.20-123.62}{2\times 8.67}$$

$129.53:\{129.53-134.53\}:$

$$\frac{134.53-128.44}{2\times 5.67}+\underbrace{\frac{132.97-126.09}{2\times 3.44}}_{1}+\frac{134.53-125.38}{2\times 4.88}+\frac{130.57-124.53}{2\times 4.37}+$$

$$\frac{134.53-124.53}{2\times 8.67}$$

$130.26:\{125.26-135.26\}:$

$$\frac{135.26-128.44}{2\times 5.67}+\underbrace{\frac{132.97-126.09}{2\times 3.44}}_{1}+\underbrace{\frac{135.14-125.38}{2\times 4.88}}_{1}+\frac{130.57-125.26}{2\times 4.37}+$$

$$\frac{135.26-125.26}{2\times 8.67}$$

$132.29:\{127.29-137.29\}:$

$$\frac{137.29-128.44}{2\times 5.67}+\frac{132.97-127.29}{2\times 3.44}+\frac{135.14-127.29}{2\times 4.88}+\frac{130.57-127.29}{2\times 4.37}+$$

$$\frac{137.29-135.88}{2\times 1.17}+\frac{137.29-127.29}{2\times 8.67}$$

$134.11:\{129.11-139.11\}:$

$$\frac{139.11-129.11}{2\times 5.67}+\frac{132.97-129.11}{2\times 3.44}+\frac{139.11-137.89}{2\times 4.92}+\frac{135.14-129.11}{2\times 4.88}+$$

$$\frac{130.57-129.11}{2\times 4.37}+\underbrace{\frac{138.22-135.88}{2\times 1.17}}_{1}+\frac{139.11-129.11}{2\times 8.67}$$

$137.05:\{132.05-142.05\}:$

$$\frac{139.78-132.05}{2\times 5.67}+\frac{132.97-132.05}{2\times 3.44}+\frac{142.05-137.89}{2\times 4.92}+\frac{135.14-132.05}{2\times 4.88}+$$

$$\frac{142.05-141.89}{2\times 2.51}+\underbrace{\frac{138.22-135.88}{2\times 1.17}}_{1}+\frac{140.96-132.05}{2\times 8.67}$$

$142.81 : \{137.81 - 147.81\}$:

$$\frac{139.78-137.81}{2\times 5.67}+\underbrace{\frac{147.73-137.89}{2\times 4.92}}_{1}+\underbrace{\frac{146.91-141.89}{2\times 2.51}}_{1}+\frac{138.22-137.81}{2\times 1.17}+$$

$$\frac{140.96-137.81}{2\times 8.67}$$

$144.40 : \{139.40 - 149.40\}$:

$$\frac{147.73-139.40}{2\times 4.92}+\underbrace{\frac{146.91-141.89}{2\times 2.51}}_{1}+\frac{149.40-148.32}{2\times 3.1}+\frac{140.96-139.40}{2\times 8.67}$$

$151.42 : \{146.42 - 156.42\}$:

$$\frac{147.73-146.42}{2\times 4.92}+\frac{146.91-146.42}{2\times 2.51}+\underbrace{\frac{154.52-148.32}{2\times 3.1}}_{1}$$

$164.05 : \{159.05 - 169.05\}$

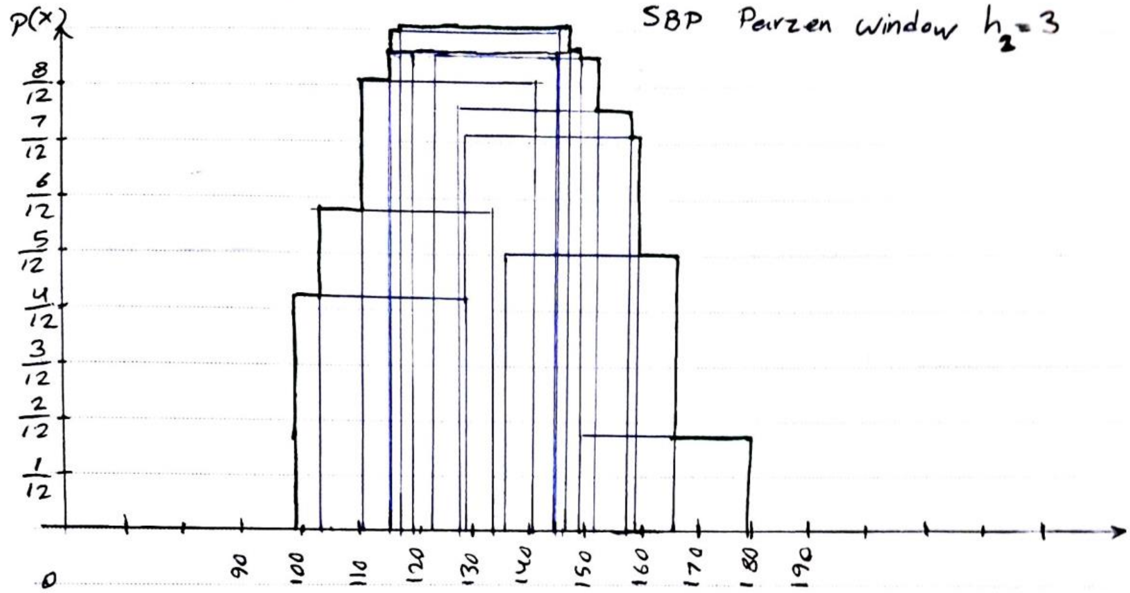
$$\frac{169.05-159.05}{2\times 6.12}$$

	{109.63 – 119.63}	{113.43 – 123.43}	{121.20 – 131.20}	{124.53 – 134.53}
	1.4	1.76	3.07	3.74
p(x)	0.116	0.146	0.2558	0.3116

	{125.26 – 135.26}	{127.29 – 137.29}	{129.11 – 139.11}	{132.05 – 142.05}
	3.78	3.96	3.92	3.1
	0.315	0.33	0.326	0.2583

{137.81 – 147.81}	{139.40 – 149.40}	{146.42 – 156.42}	{159.05 – 169.05}
3.23	2.11	1.23	0.81
0.26916	0.17583	0.1025	0.0675

Parzen window $h_2 = 3$ for SBP:



114.63 : {99.63 – 129.63} :

$$\frac{129.63-128.44}{2\times 5.67}+\frac{129.63-126.09}{2\times 3.44}+\frac{129.63-125.38}{2\times 4.88}+\underbrace{\frac{121.26-115.6}{2\times 2.83}}_{1}+$$

$$\frac{129.63-121.83}{2\times 4.37}+\underbrace{\frac{121.84-107.42}{2\times 7.21}}_{1}+\frac{129.63-123.62}{2\times 8.67}$$

118.43 : {103.43 – 133.43} :

$$\frac{133.43-128.44}{2\times 5.67}+1+\frac{133.43-125.38}{2\times 4.88}+1+1+1+\frac{133.43-123.62}{2\times 8.67}$$

126.20 : {111.20 – 141.20} :

$$1+1+\frac{141.20-137.89}{2\times 4.92}+1+1+1+1+\frac{121.84-111.20}{2\times 7.21}+1$$

$129.53:\{114.53-144.53\}:$

$$1+1+\frac{144.53-137.89}{2\times 4.92}+1+1+\frac{144.53-141.89}{2\times 2.51}+1+1+\frac{121.84-114.53}{2\times 7.21}+1$$

$130.26:\{115.26-145.26\}:$

$$1+1+\frac{145.26-137.89}{2\times 4.92}+1+\frac{121.26-115.26}{2\times 2.83}+\frac{145.26-141.89}{2\times 2.51}+1+1+$$

$$\frac{121.84-115.26}{2\times 7.21}+1$$

$132.29:\{117.29-147.29\}:$

$$1+1+\frac{147.29-137.89}{2\times 4.92}+1+\frac{121.26-117.29}{2\times 2.83}+1+1+1+\frac{121.84-117.29}{2\times 7.21}+1$$

$134.11:\{119.11-149.11\}:$

$$1+1+1+1+\frac{121.26-119.11}{2\times 2.83}+1+1+1+\frac{121.84-119.11}{2\times 7.21}+\frac{149.11-148.32}{2\times 3.1}+1$$

$137.05:\{122.05-152.05\}:$

$$1+1+1+1+1+\frac{130.57-122.05}{2\times 4.37}+1+\frac{152.05-148.32}{2\times 3.1}+1$$

$142.81:\{127.81-157.81\}:$

$$1+\frac{132.97-127.81}{2\times 3.44}+1+\frac{135.14-127.81}{2\times 4.88}+1+\frac{130.57-127.81}{2\times 4.37}+1+1+$$

$$\frac{140.96-127.81}{2\times 8.67}$$

$144.40 : \{129.40 - 159.40\}$:

$$\frac{139.78-129.40}{2\times 5.67}+\frac{132.97-129.40}{2\times 3.44}+1+\frac{135.14-129.40}{2\times 4.88}+1+\frac{130.57-129.40}{2\times 4.37}$$

$$+1+1+\frac{140.96-129.40}{2\times 8.67}+\frac{159.40-157.93}{2\times 6.12}$$

$151.42 : \{136.42 - 166.42\}$:

$$\frac{139.78-136.42}{2\times 5.67}+1+1+\frac{138.22-136.42}{2\times 1.17}+1+\frac{140.96-136.42}{2\times 8.67}+\frac{166.42-157.93}{2\times 6.12}$$

$164.05 : \{149.05 - 179.05\}$:

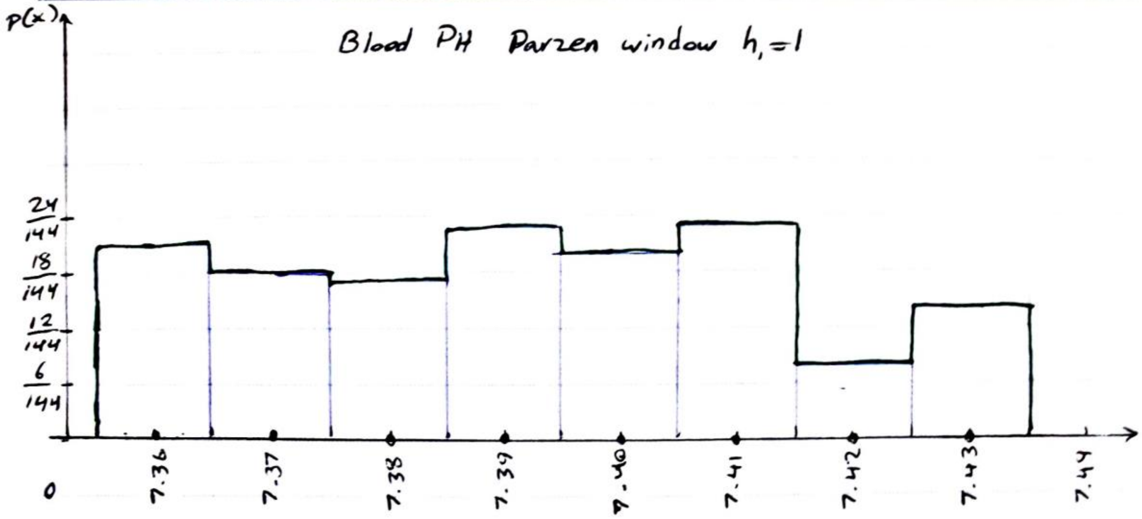
$$\frac{154.52-149.05}{2\times 3.1}+1$$

{99.63 - 129.63}	{103.43 - 133.43}	{111.20 - 141.20}	{114.53 - 144.53}
4.29	5.83	8.07	8.70
0.3575	0.48583	0.6725	0.725

{115.26 - 145.26}	{117.29 - 147.29}	{119.11 - 149.11}	{122.05 - 152.05}
8.93	8.97	8.69	8.57
0.744167	0.7475	0.724167	0.714167

{127.81 - 157.81}	{129.40 - 159.40}	{136.42 - 166.42}	{149.05 - 179.05}
7.57	7.11	5.02	1.88
0.63083	0.5925	0.4183	0.15667

Parzen window $h_1 = 1$ for Blood pH:



7.36: {7.355 – 7.365}:

$\frac{1}{4}+1+\frac{1}{2}=\frac{21}{12} \rightarrow \frac{21}{144}$ (p(x))

7.37: {7.365 – 7.375}:

$\frac{1}{2}+\frac{1}{4}+\frac{1}{4}+\frac{1}{2}=\frac{18}{12} \rightarrow \frac{18}{144}$

7.38: {7.375 – 7.385}:

$\frac{1}{2}+\frac{1}{4}+\frac{1}{4}+\frac{1}{6}+\frac{1}{4}=\frac{17}{12} \rightarrow \frac{17}{144}$

7.39: {7.385 – 7.395}:

$\frac{1}{4}+\frac{1}{4}+\frac{1}{6}+\frac{1}{2}+\frac{1}{4}+\frac{1}{2}=\frac{23}{12} \rightarrow \frac{23}{144}$

7.40: {7.395 – 7.405}:

$\frac{1}{4}+\frac{1}{6}+\frac{1}{2}+\frac{1}{4}+\frac{1}{2}=\frac{20}{12} \rightarrow \frac{20}{144}$ (p(x))

7.41: {7.405 – 7.415}:

$\frac{1}{2}+\frac{1}{4}+\frac{1}{6}+1=\frac{23}{12} \rightarrow \frac{23}{144}$

7.42: {7.415 – 7.425}:

$\frac{1}{2}+\frac{1}{6}=\frac{8}{12} \rightarrow \frac{8}{144}$

7.43: {7.425 – 7.435}:

$1+\frac{1}{6}=\frac{14}{12} \rightarrow \frac{14}{144}$

Parzen window $h_2=3$ for Blood pH:

7.36: {7.345 – 7.375}:

$\frac{1}{4}+1+\frac{1}{2}+\frac{1}{2}+\frac{1}{4}+\frac{1}{4}+\frac{1}{2}=\frac{39}{12} \rightarrow \frac{39}{144}$ (p(x))

7.40: {7.385 – 7.415}:

$\frac{23}{12}+\frac{20}{12}+\frac{23}{12}=\frac{66}{12} \rightarrow \frac{66}{144}$ (p(x))

7.37: {7.355 – 7.385}:

7.41: {7.395 – 7.425}:

$\frac{1}{4}+1+1+\frac{1}{4}+\frac{1}{4}+1+\frac{1}{4}+\frac{1}{4}+\frac{1}{6}+\frac{1}{4}=\frac{56}{12}\rightarrow\frac{56}{144}$ $\frac{20}{12}+\frac{23}{12}+\frac{8}{12}=\frac{51}{12}\rightarrow\frac{51}{144}$

7.38: $\{7.365 - 7.395\}$:

$\frac{18}{12}+\frac{17}{12}+\frac{23}{12}=\frac{58}{12} \rightarrow \frac{58}{144}$

7.42: $\{7.405 - 7.435\}$

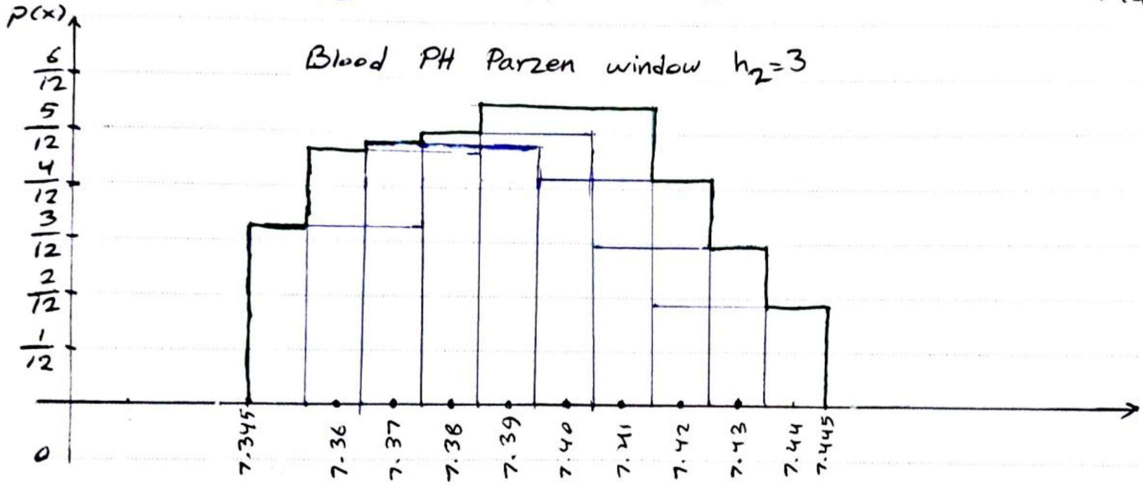
$\frac{23}{12}+\frac{8}{12}+\frac{14}{12}=\frac{35}{12} \rightarrow \frac{35}{144}$

7.39: $\{7.375 - 7.405\}$:

$\frac{17}{12}+\frac{23}{12}+\frac{20}{12}=\frac{60}{12} \rightarrow \frac{60}{144}$

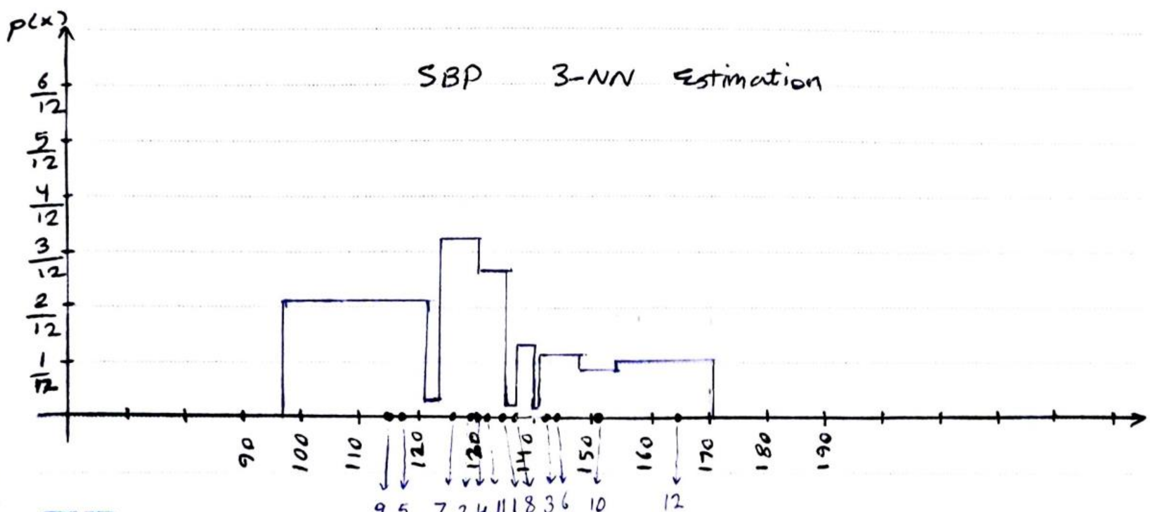
7.43: $\{7.415 - 7.445\}$:

$\frac{8}{12}+\frac{14}{12}=\frac{22}{12} \rightarrow \frac{22}{144}$



c. Sketch the 3-NN estimate of the density functions.

3-NN for SBP:



The assumption that I made :

I considered 3 nearest neighbors based on the mean of SBP values for each sample. I calculated the area based on the whole area a sample is possibly in.

Point 9: 9, 5, 7 ⇒ 107.42 – 130.57 ⇒ $\frac{1}{23.15}$

point 5: 9, 5, 7 ⇒ 107.42 – 130.57 ⇒ $\frac{1}{23.15}$

Point 7: 7, 2, 4 ⇒ 121.83 – 135.14 ⇒ $\frac{1}{13.31}$

point 2: 2, 4, 11 ⇒ 123.62 – 140.96 ⇒ $\frac{1}{17.34}$

point 4: 2, 4, 11 ⇒ 123.62 – 140.96 ⇒ $\frac{1}{17.34}$

point 11: 4, 11, 1 ⇒ 123.62 – 140.96 ⇒ $\frac{1}{17.34}$

point 1: 11, 1, 8 ⇒ 123.62 – 140.96 ⇒ $\frac{1}{17.34}$

point 8: 11, 1, 8 ⇒ 123.62 – 140.96 ⇒ $\frac{1}{17.34}$

point 3: 8, 3, 6 ⇒ 135.88 – 147.73 ⇒ $\frac{1}{11.85}$

point 6: 3, 6, 10 ⇒ 137.89 – 154.52 ⇒ $\frac{1}{16.63}$

point 10: 6, 10, 12 $\Rightarrow$ 141.89 - 170.17 $\Rightarrow \frac{1}{28.28}$

point 12: 6, 10, 12 $\Rightarrow$ 141.89 - 170.17 $\Rightarrow \frac{1}{28.28}$

107.42 - 121.83 : $\frac{2\times 14.41}{23.15} = 1.24 \xrightarrow{p(x)} \frac{1.24}{12}$

121.83 - 123.63 : $\frac{2\times 1.8}{23.15} + \frac{1.8}{13.31} \simeq 0.155 + 0.135 = 0.29 \rightarrow \frac{0.29}{12}$

123.63 - 130.57 : $\frac{5\times 6.94}{17.34} + \frac{6.94}{13.31} + \frac{2\times 6.94}{23.15} \simeq 2 + 0.521 + 0.599 = 3.12$

$\frac{3.12}{12}$

130.57 - 135.14 : $\frac{4.57}{13.31} + \frac{5\times 4.57}{17.34} \simeq 0.3433 + 1.3177 = 1.661 \rightarrow \frac{1.661}{12}$

135.14 - 135.88 : $\frac{5\times 0.74}{17.34} \simeq 0.2134 \rightarrow \frac{0.2134}{12}$

135.88 - 137.89 : $\frac{2.01}{11.85} \simeq 0.1696 \rightarrow \frac{0.1696}{12}$

137.89 - 140.96 : $\frac{5\times 3.07}{17.34} + \frac{3.07}{11.85} + \frac{3.07}{16.63} \simeq 0.8852 + 0.259 + 0.1846$
$= 1.3288 \rightarrow \frac{1.3288}{12}$

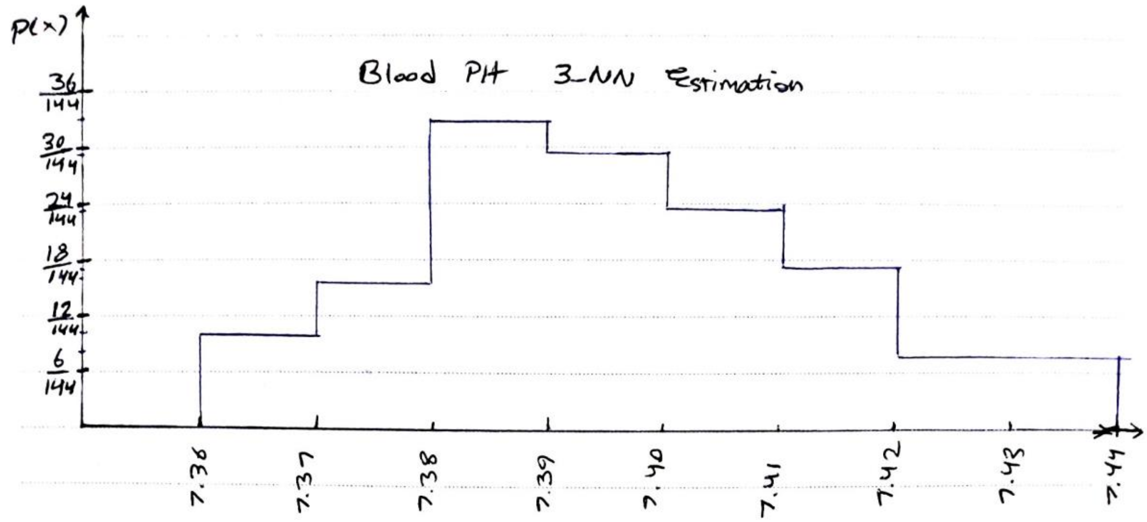
140.96 - 141.89 : $\frac{0.93}{11.85} + \frac{0.93}{16.63} = 0.07848 + 0.05592 = 0.1344 \rightarrow \frac{0.1344}{12}$

141.89 - 147.73 : $\frac{5.84}{11.85} + \frac{5.84}{16.63} + \frac{2\times 5.84}{28.28} \simeq 0.4928 + 0.3511 + 0.413$
$= 1.2569 \rightarrow \frac{1.2569}{12}$

147.73 - 154.52 : $\frac{6.79}{16.63} + \frac{2\times 6.79}{28.28} \simeq 0.4083 + 0.4801 = 0.8884 \rightarrow \frac{0.8884}{12}$

154.52 - 170.17 : $\frac{2\times 15.65}{28.28} \simeq 1.1068 \rightarrow \frac{1.1068}{12}$

3-NN for Blood pH :



point 6 : 6, 10, 1 ⇒ 7.36 – 7.39 ⇒ $\frac{1}{3}$

point 10 : 5, 10, 1 ⇒ 7.36 – 7.40 ⇒ $\frac{1}{4}$

point 1 : 1, 10, 5 ⇒ 7.36 – 7.40 ⇒ $\frac{1}{4}$

point 5 : 5, 1, 9 ⇒ 7.37 – 7.41 ⇒ $\frac{1}{4}$

point 9 : 9, 5, 4 ⇒ 7.37 – 7.41 ⇒ $\frac{1}{4}$

point 8 : 8, 12, 4 ⇒ 7.38 – 7.42 ⇒ $\frac{1}{4}$

point 12 : 12, 4, 8 ⇒ 7.38 – 7.42 ⇒ $\frac{1}{4}$

point 4 : 4, 8, 12 ⇒ 7.38 – 7.42 ⇒ $\frac{1}{4}$

point 11 : 11, 7, 4 ⇒ 7.38 – 7.44 ⇒ $\frac{1}{6}$

point 7 : 11, 7, 4 $\Rightarrow$ 7.38 – 7.44 $\Rightarrow$ $\frac{1}{6}$

point 3 : 7, 3, 2 $\Rightarrow$ 7.38 – 7.44 $\Rightarrow$ $\frac{1}{6}$

point 2 : 7, 3, 2 $\Rightarrow$ 7.38 – 7.44 $\Rightarrow$ $\frac{1}{6}$

7.36 – 7.37 : $\frac{1}{3} + \frac{1}{4} + \frac{1}{4} = \frac{4}{12} + \frac{6}{12} = \frac{10}{12} \xrightarrow{P(x)} \frac{10}{144}$

7.37 – 7.38 : $\frac{1}{3} + 4(\frac{1}{4}) = \frac{4}{12} + \frac{12}{12} = \frac{16}{12} \longrightarrow \frac{16}{144}$

7.38 – 7.39 : $\frac{1}{3} + 4(\frac{1}{4}) + 3(\frac{1}{4}) + 4(\frac{1}{6}) = \frac{4}{12} + \frac{21}{12} + \frac{8}{12} = \frac{33}{12} \rightarrow \frac{33}{144}$

7.39 – 7.40 : $7(\frac{1}{4}) + 4(\frac{1}{6}) = \frac{21}{12} + \frac{8}{12} = \frac{29}{12} \longrightarrow \frac{29}{144}$

7.40 – 7.41 : $5(\frac{1}{4}) + 4(\frac{1}{6}) = \frac{15}{12} + \frac{8}{12} = \frac{23}{12} \longrightarrow \frac{23}{144}$

7.41 – 7.42 : $3(\frac{1}{4}) + 4(\frac{1}{6}) = \frac{9}{12} + \frac{8}{12} = \frac{17}{12} \longrightarrow \frac{17}{144}$

7.42 – 7.43 : $4(\frac{1}{6}) = \frac{8}{12} \longrightarrow \frac{8}{144}$

7.43 – 7.44 : $4(\frac{1}{6}) = \frac{8}{12} \longrightarrow \frac{8}{144}$

d. Use a Gaussian kernel to estimate the probability of the following states:

– SBP = 131. One could use $h = \sigma^2$ or $h = \sigma$ or ... What I'm doing:

$$p(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\sqrt{2\pi h}} \exp\left(- \frac{(x - x_i)^2}{2h}\right)$$

In a uniform distribution $\text{Variance} = \frac{(b-a)^2}{12}$

for each sample we would have:

	SBP variance	Blood plt variance
1	$\simeq 10.71$	$\simeq 3.3 \times e^{-5}$
2	$\simeq 3.94$	0
3	$\simeq 8.07$	$\simeq 3.3 \times e^{-5}$
4	$\simeq 7.93$	$\simeq 1.3 \times e^{-4}$
5	$\simeq 2.67$	$\simeq 1.3 \times e^{-4}$
6	$\simeq 2.1$	0
7	$\simeq 6.36$	$3 \times e^{-4}$
8	$\simeq 0.45$	$\simeq 3.3 \times e^{-5}$
9	$\simeq 17.32$	$\simeq 1.3 \times e^{-4}$
10	$\simeq 3.2$	$\simeq 3.3 \times e^{-5}$
11	$\simeq 25.05$	0
12	$\simeq 12.48$	$\simeq 3.3 \times e^{-5}$

since it was hard to calculate with these numbers I had to multiply them by 10000. also couldn't put $\sigma^2 = 0$ in my calculations so instead I put 0.0005 for ± 0.00 in given table.

$$p(131) = \frac{1}{12-1} \sum_{i=1}^{12} \frac{1}{\sqrt{2\pi h_i}} \exp\left(-\frac{(x-x_i)^2}{2h_i}\right)$$

I'm considering a bandwidth which differs for each sample.

$h = \sigma^2$.

$$p(131) = \frac{1}{11}\left[\frac{1}{\sqrt{2\pi \times 10.71}} \exp\left(-\frac{(131-134.11)^2}{2 \times 10.71}\right) + \cdots + \frac{1}{\sqrt{2\pi \times 12.48}} \exp\left(-\frac{(131-164.0)^2}{2 \times 12.48}\right)\right]$$

$\simeq 0.001735$

- SBP falls into a safe range, or $125 < SBP < 145$.

$$p(125 < SBP < 145) = \int_{125}^{145} \frac{1}{11} \sum_{i=1}^{12} \frac{1}{\sqrt{2\pi h_i}} \exp\left(-\frac{(x-x_i)^2}{2h_i}\right) dx$$

$$P(125<x<145)=\frac{1}{11}\sum_{i=1}^{12}\int_{125}^{145}\frac{1}{\sqrt{2\pi h_i}}\exp\left(-\frac{(x-x_i)^2}{2h_i}\right)dx$$

$$P(125<x<145)=\frac{1}{11}\sum_{i=1}^{12}P_i\left(\frac{125-x_i}{\sqrt{h_i}}<Z<\frac{145-x_i}{\sqrt{h_i}}\right)$$

$$=\frac{1}{11}\Big[P(-2.7<Z<3.23)+P(-1.72<Z<5.9)$$

$$+P(-5.68<Z<0.69)+P(-1.68<Z<4.72)$$

$$+P(2.76<Z<11.17)+P(-8.66<Z<0.26)$$

$$+P(-0.4<Z<6.36)+P(-7.87<Z<5.19)$$

$$+P(2.73<Z<7.99)+P(-10.61<Z<-2.57)$$

$$+P(-1.75<Z<3.05)+P(-11.16<Z<-5.44)\Big]$$

$$P(125<x<145)=\frac{1}{11}\Big[(0.9994-0.0035)+(1-0.0427)$$

$$+(0.7549-0)+(1-0.0465)+(1-0.9971)$$

$$+(0.6026-0)+(1-0.3446)+(1-0)+(1-0.9968)$$

$$+(0.0051-0)+(0.9989-0.0401)+(0-0)=\frac{6.8896}{11}$$

$$=0.6263$$

$-\mathrm{pH}=7.42$ $\quad P(7.42)=\frac{1}{11}\sum_{i=1}^{12}\frac{1}{\sqrt{2\pi h}}\exp\left(-\frac{(x-x_i)^2}{2h_i}\right)$

$$P(7.42)=\frac{1}{11}\left[\frac{1}{2\pi\times\frac{1}{3}}\exp\left(-\frac{(7.42-7.38)^2}{2\times\frac{1}{3}}\right)+\dots\right]=0.8$$

- Blood becomes too acidy or pH < 7.38.

$$p(pH < 7.38) = \int_0^{7.38} \frac{1}{11} \sum_{i=1}^{12} \frac{1}{\sqrt{2\pi h_i}} \exp\left(\frac{x - x_i}{h_i}\right) dx$$

$$= \frac{1}{11} \sum_{i=1}^{12} \int_0^{7.38} \frac{1}{\sqrt{2\pi h_i}} \exp\left(\frac{x - x_i}{h_i}\right) dx$$

$$= \frac{1}{11} \sum_{i=1}^{12} p_i \left(Z < \frac{7.38 - x_i}{\sqrt{h_i}}\right)$$

$$p(pH < 7.38) = \frac{1}{11}\Big[p\Big(Z < \underbrace{\frac{7.38 - 7.38}{\sqrt{\frac{1}{3}}}}_{0}\Big) + p(Z < -5) + p(Z < -2.82)$$

$$+ p(Z < -1) + p(Z < 0) + p(Z < 2) + p(Z < -1.22) + p(Z < -1.41)$$

$$+ p(Z < -0.5) + p(Z < 0.7) + p(Z < -3) + p(Z < -1.41) =$$

$$\frac{1}{11}(0.5 + 0 + 0.0024 + 0.1587 + 0.5 + 0.9772 + 0.1112 + 0.0793$$

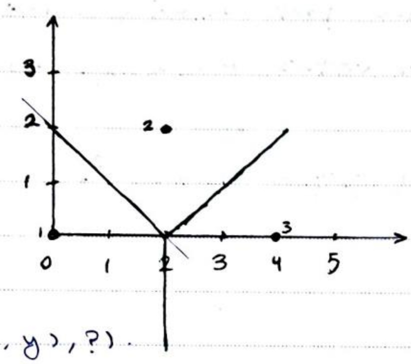
$$+ 0.3085 + 0.7580 + 0.0013 + 0.0793) = \frac{3.4759}{11} = 0.316$$

2. Beyond Density Estimation : K-NN is Jack of All Trades!

$D_1 = \{((0,0),1), ((2,2),2), ((4,0),3)\}$

$D_2 = \{((0,0),1), ((1,1),1), ((-1,1),2)\}$

a. Consider the samples in D_1 and draw the decision boundaries for a 1-NN classifier with respect to Euclidean distance. Write down the equations of the decision boundaries.



Assume a point $((x, y), ?)$.

Euclidean Distance $d(v_i, v_j) = \sqrt{(x_i - x_j)^2 + (y_i + y_j)^2}$

$d(v_1, v_?) = \sqrt{(0-x)^2 + (0-y)^2} = \sqrt{x^2 + y^2}$

$d(v_2, v_?) = \sqrt{(2-x)^2 + (2-y)^2}$

$d(v_3, v_?) = \sqrt{(4-x)^2 + (0-y)^2} = \sqrt{(4-x)^2 + y^2}$

(Square root is similar, so,)

$g_{12} = d(v_1, v_?) - d(v_2, v_?) = x^2 + y^2 - (2-x)^2 - (2-y)^2 = x^2 + y^2 - 4 + 4x$

$-x^2-4+4y+y^2 = 4x+4y-8 = x+y-2$ $\boxed{x+y-2=0}$

$g_{13} = x^2+y^2-(4-x)^2-y^2 = x^2+y^2-16+8x-x^2-y^2 = 8x-16 = x-2$

$\boxed{x-2=0}$

$g_{23} = (2-x)^2+(2-y)^2-(4-x)^2-y^2 = 4-4x+x^2+4-4y+y^2-16+8x-x^2-y^2$

$= -8+4x-4y = x-y-2$ $\boxed{x-y-2=0}$

b. Now repeat the previous part considering a new variant of Euclidean distance d_M. Given two vectors $v_1=(x_1,y_1)$ and $v_2=(x_2,y_2)$, d_M is defined as:

$$d_M(v_1,v_2) = \sqrt{\frac{1}{2}(x_1-x_2)^2+(y_1-y_2)^2}$$

$d_M(v_1,v_?) = \sqrt{\frac{1}{2}(0-x)^2+(0-y)^2} = \sqrt{\frac{1}{2}x^2+y^2}$

$d_M(v_2,v_?) = \sqrt{\frac{1}{2}(2-x)^2+(2-y)^2}$

$d_M(v_3,v_?) = \sqrt{\frac{1}{2}(4-x)^2+(0-y)^2} = \sqrt{\frac{1}{2}(4-x)^2+y^2}$

$g_{12} = \frac{1}{2}x^2+y^2-\frac{1}{2}(2-x)^2-(2-y)^2 = \frac{1}{2}x^2+y^2-2-\frac{1}{2}x^2+2x$

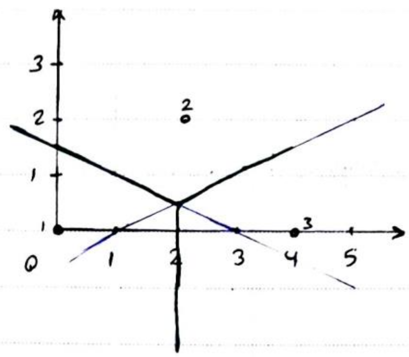
$-4-y^2+4y = -6+2x+4y = 2y+x-3$ $\boxed{2y+x-3=0}$

$g_{13} = \frac{1}{2}x^2+y^2-\frac{1}{2}(4-x)^2-y^2 = \frac{1}{2}x^2-8-\frac{1}{2}x^2+4x = -8+4x$

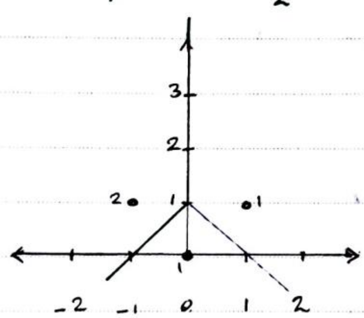
$= x-2$ $\boxed{x-2=0}$

$g_{23} = \frac{1}{2}(2-x)^2+(2-y)^2-\frac{1}{2}(4-x)^2-y^2 = 2+\frac{1}{2}x^2-2x+4+y^2-4y$

$-8-\frac{1}{2}x^2+4x-y^2=-2+2x-4y=x-2y-1$ $\boxed{x-2y-1=0}$



c. Repeat part (a) for samples in D_2.



$d(v_{10}, v_?)=\sqrt{(0-x)^2+(0-y)^2}=\sqrt{x^2+y^2}$

$d(v_{11}, v_?)=\sqrt{(1-x)^2+(1-y)^2}$

$d(v_2, v_?)=\sqrt{(-1-x)^2+(1-y)^2}$

$g_{10,2}=x^2+y^2-(-1-x)^2+(1-y)^2=x^2+y^2-1-x^2-2x-1-y^2+2y$

$=-2-2x+2y=y-x+1 \Rightarrow \boxed{y-x-1=0}$

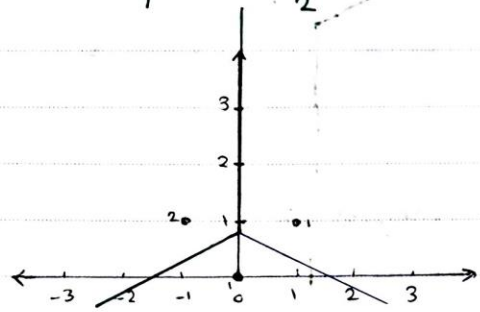
$g_{11,2}=(1-x)^2+(1-y)^2-(-1-x)^2-(1-y)^2=1+x^2-2x-1-x^2-2x$

$=-4x=x \Rightarrow \boxed{x=0}$

$$g_{10,11} = x^2+y^2-(1-x)^2-(1-y)^2 = x^2+y^2-1-x^2+2x-1-y^2+2y$$

$$= x+y-1 \qquad \boxed{x+y-1=0}$$

d. Repeat part (b) for samples in. D_2.



$$d_M(v_{10}, v_?) = \sqrt{\frac{1}{2}(0-x)^2+(0-y)^2} = \sqrt{\frac{1}{2}x^2+y^2}$$

$$d_M(v_{11}, v_?) = \sqrt{\frac{1}{2}(1-x)^2+(1-y)^2}$$

$$d_M(v_2, v_?) = \sqrt{\frac{1}{2}(-1-x)^2+(1-y)^2}$$

$$g_{10,2} = \frac{1}{2}x^2+y^2-\frac{1}{2}(-1-x)^2-(1-y)^2 = \frac{1}{2}x^2+y^2-\frac{1}{2}x^2-\frac{1}{2}-x-1-y^2+2y$$

$$= -\frac{3}{2}-x+2y = 4y-2x-3 \qquad \boxed{4y-2x-3=0}$$

$$g_{11,2} = \frac{1}{2}(1-x)^2+(1-y)^2-\frac{1}{2}(-1-x)^2-(1-y)^2 = \frac{1}{2}+\frac{1}{2}x^2-x-\frac{1}{2}-\frac{1}{2}x^2-x$$

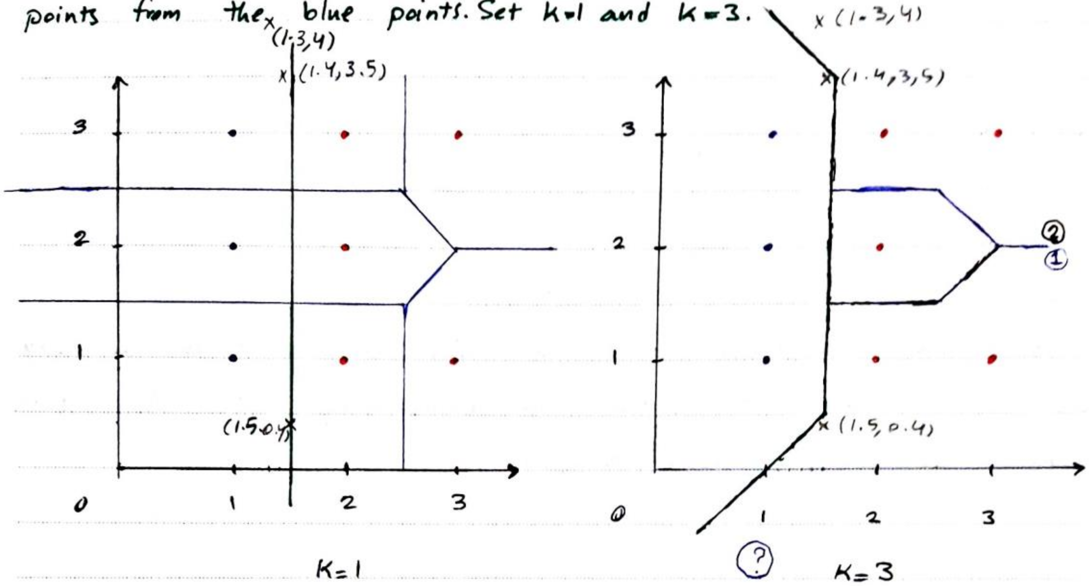
$$= -2x = x \qquad \boxed{x=0}$$

$$g_{10,11} = \frac{1}{2}x^2+y^2-\frac{1}{2}(1-x)^2-(1-y)^2 = \frac{1}{2}x^2+y^2-\frac{1}{2}-\frac{1}{2}x^2+x-1-y^2+2y$$

$$= -\frac{3}{2}+x+2y = 4y+2x-3 \qquad \boxed{4y-2x-3=0}$$

Now consider the dataset given in figure 2, part (a).

e. Draw the decision boundary of a K-NN classifier that separates the red points from the blue points. Set K=1 and K=3.



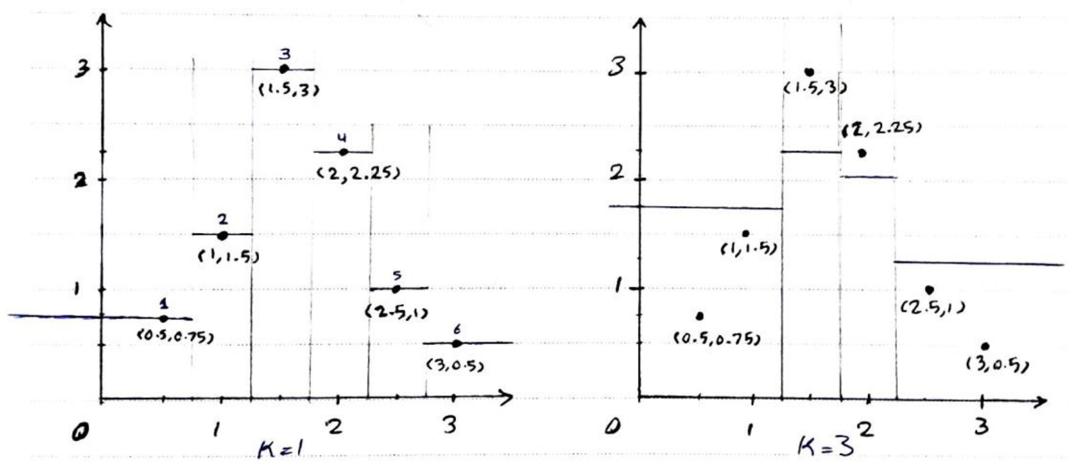
I don't wanna crowd diagram for K=3 so I'm just gonna draw the necessary lines that show why there's a difference between K=1 and K=3 decision boundary.

It's easy to understand how boundary ① in the diagram comes to be. For example consider point ?; The three closest points are the ones at the bottom and since two of those are red, ? gets to be red. The same thing happens with line ②.

f. Determine the labels of the following points using the above classifiers.

	K=1	K=3
(1.3,4)	blue	red
(1.5, 0.4)	(on the line, so I'm assuming random) blue	red
(1.4,3.5)	blue	blue

Finally, consider a regression problem in which the goal is to apply k-NN in order to predict the outcome of a test sample given its feature. Samples are illustrated in Figure 2, part (b).



g. Find the training error when K=1 and k=3.

If all of these six points are our training set, then for K=1, training error would be equal to zero, since we would have the questioned point in the training set. But since we use the average of three nearest neighbor's values for our prediction,

error would not be zero.

Computing Predictions:

point 1 (0.5, 0.75):

$|x_2 - x_1| = 1 - 0.5 = 0.5$ $|x_3 - x_1| = 1.5 - 0.5 = 1$ $|x_4 - x_1| = 2 - 0.5 = 1.5$

$|x_5 - x_1| = 2.5 - 0.5 = 2$ $|x_6 - x_1| = 3 - 0.5 = 2.5$ $|x_1 - x_1| = 0$

Three nearest neighbors: 1, 2, 3.

prediction for point 1 $= \frac{0.75 + 1.5 + 3}{3} = \frac{5.25}{3} = 1.75$

point 2 (1, 1.5):

$|x_1 - x_2| = 0.5$ $|x_2 - x_2| = 0$ $|x_3 - x_2| = 1.5 - 1 = 0.5$

$|x_4 - x_2| = 2 - 1 = 1$ $|x_5 - x_2| = 2.5 - 1 = 1.5$ $|x_6 - x_2| = 3 - 1 = 2$

Three nearest neighbors: 2, 1, 3

prediction for point 2 $= \frac{1.5 + 0.75 + 3}{3} = \frac{5.25}{3} = 1.75$

point 3 (1.5, 3):

3-NN: 3, 2, 4

prediction for point 3 $= \frac{3 + 1.5 + 2.25}{3} = \frac{6.75}{3} = 2.25$

point 4 (2, 2.25):

3-NN: 4, 3, 5

prediction for point 4 $= \frac{2.25 + 3 + 1}{3} = \frac{6.25}{3} = 2.083$

point 5 (2.5, 1):

3-NN: 5, 4, 6

prediction for point 5 $= \frac{1 + 2.25 + 0.5}{3} = \frac{3.75}{3} = 1.25$

point 6 (3, 0.5)

3-NN: 6, 5, 4

prediction for point 6 $= \frac{0.5 + 1 + 2.25}{3} = \frac{3.75}{3} = 1.25$

I'm assuming MSE as our error criterion!

Computing MSE:

$$Error_{MSE} = \frac{1}{6}\left[(0.75 - 1.75)^2 + (1.5 - 1.75)^2 + (3 - 2.25)^2 + (2.25 - 2.083)^2 + (1 - 1.25)^2 + (0.5 - 1.25)^2\right] = \frac{1}{6}(1 + 0.0625 + 0.5625 + 0.027889 + 0.0625 + 0.5625) \simeq 0.3796$$

h. Given the following test samples, predict their corresponding outcomes using 1-NN and 3-NN models: (0.75, ?), (1.75, ?), (2.25, ?)

(0.75, ?):

$|x_1 - x_?| = |0.5 - 0.75| = 0.25$ $\quad |x_2 - x_?| = |1 - 0.75| = 0.25$

$|x_3 - x_?| = |1.5 - 0.75| = 0.75$ $\quad |x_4 - x_?| = |2 - 0.75| = 1.25$

$|x_5 - x_?| = |2.5 - 0.75| = 1.75$ $\quad |x_6 - x_?| = |3 - 0.75| = 2.25$

1-NN: 1 OR 2, I'm choosing (randomly) 2.

So, prediction = 1.5

3-NN: Points 1,2,3

So, prediction $= \frac{0.75 + 1.5 + 3}{3} = 1.75$

$(1.75, ?)$:

$|x_1 - x_?| = |0.5 - 1.75| = 1.25$ $|x_2 - x_?| = |1 - 1.75| = 0.75$

$|x_3 - x_?| = |1.5 - 1.75| = 0.25$ $|x_4 - x_?| = |2 - 1.75| = 0.25$

$|x_5 - x_?| = |2.5 - 1.75| = 0.75$ $|x_6 - x_?| = |3 - 1.75| = 1.25$

1-NN: Point 3 OR 4, I'm choosing 3.

So, prediction = 3

3-NN: Points 3, 4, 2,

So, prediction $= \frac{3 + 2.25 + 1.5}{3} = 2.25$

$(2.25, ?)$:

$|x_1 - x_?| = |0.5 - 2.25| = 1.75$ $|x_2 - x_?| = |1 - 2.25| = 1.25$

$|x_3 - x_?| = |1.5 - 2.25| = 0.75$ $|x_4 - x_?| = |2 - 2.25| = 0.25$

$|x_5 - x_?| = |2.5 - 2.25| = 0.25$ $|x_6 - x_?| = |3 - 2.25| = 0.75$

1-NN: Point 4 OR 5, I'm choosing 4.

So, prediction = 2.25

3-NN: Points 4,5, 3,

So, prediction $= \frac{2.25 + 1 + 3}{3} \approx 2.083$

i. Plot the regression line considering 1-NN and 3-NN models.

You can find the regression lines on the diagrams in page "26".

3. Never Get Fooled Again: Fake Bill Detection System

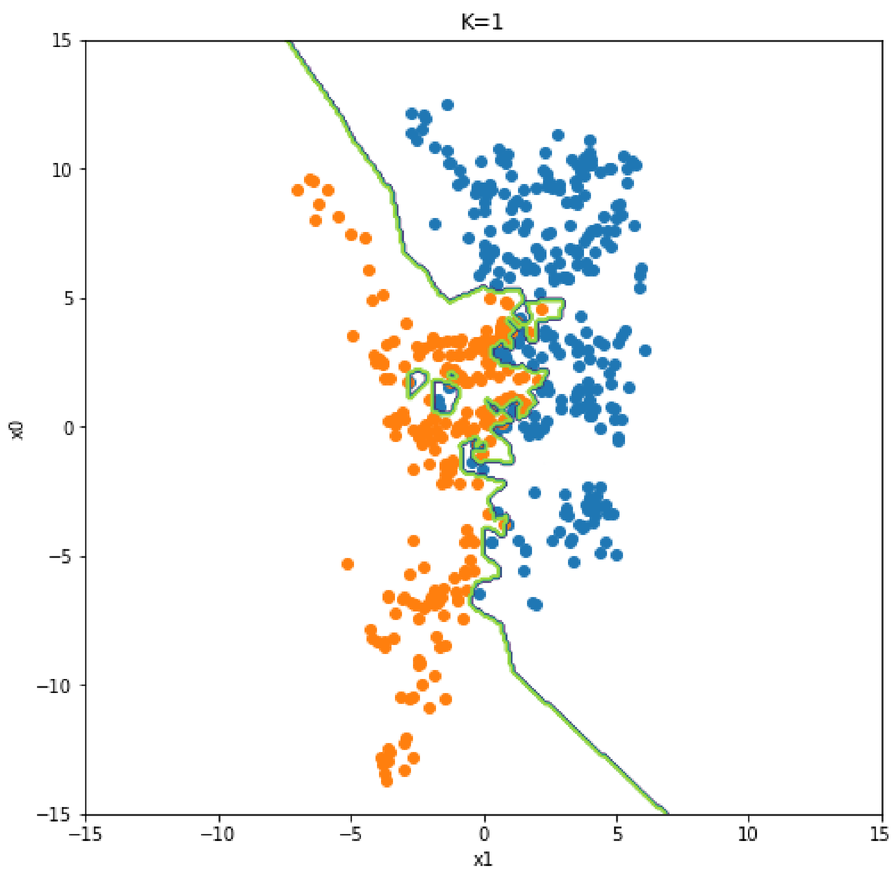
I implemented several functions to calculate Euclidean distance, calculate the nearest neighbors to a data point, predict the test data and create KNN model. I also implemented a function for calculating Accuracy.

a. Which of the two features are more suitable for a 1-NN classifier?

Feature 1, 2: Variance and Skewness of wavelet transformed image.

```
K=1
(0, 1) accuracy:  0.9311926605504587
(0, 2) accuracy:  0.8853211009174312
(0, 3) accuracy:  0.8910550458715596
(1, 2) accuracy:  0.8830275229357798
(1, 3) accuracy:  0.8704128440366973
(2, 3) accuracy:  0.7247706422018348
```

b. Plot the Voronoi diagram for these two features assuming a 1-NN classifier.

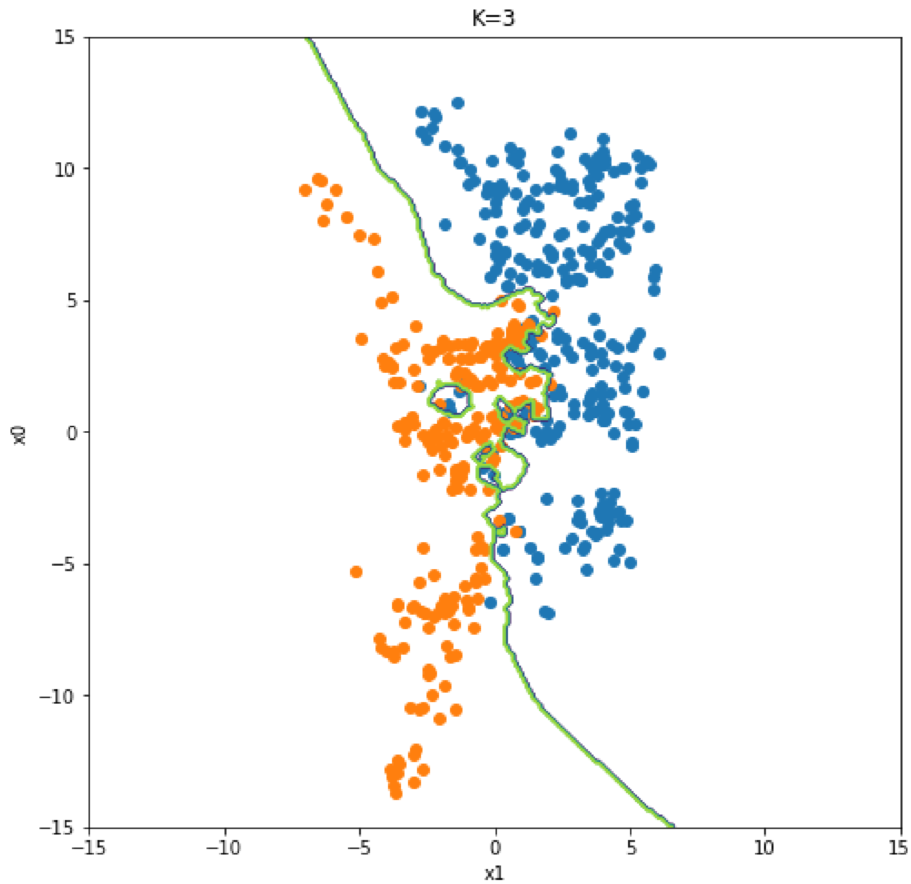


c. Repeat part (a) for a 3-NN classifier.

Again, Feature 1,2: Variance and Skewness of wavelet transformed image.

```
K=3
(0, 1) accuracy:  0.9311926605504587
(0, 2) accuracy:  0.8853211009174312
(0, 3) accuracy:  0.9105504587155964
(1, 2) accuracy:  0.8818807339449541
(1, 3) accuracy:  0.875
(2, 3) accuracy:  0.7144495412844036
```

d. Repeat part (b) for a 3-NN classifier.



As you can see 3-NN has more generalization therefore unlike 1-NN The decision boundary does not completely discriminate the two classes and there are some data points that lie in the other class's region.

4. Help Trump Fight Against Fake News

a. Implement a function `prepare_data()` which loads the data and preprocesses it using a `CountVectorizer` (look here). It must then split the dataset randomly into 70% training, 15% validation and 15% test samples.

I implemented the function just like it was asked in the question. the function takes two parameters; paths to the text data. I used countvectorizer how to count the text words and created dataframes for fake news and for real news, Then I concatenated the two data frames then I used sklearn to split the dataset. You can see the code below:

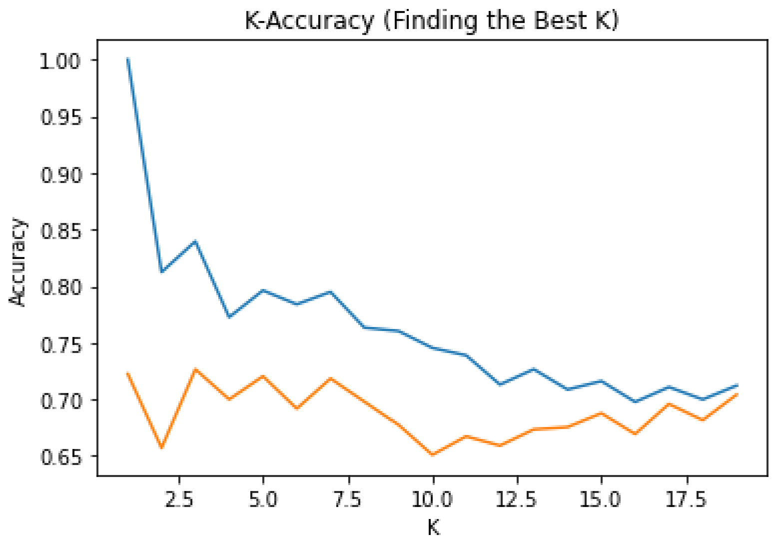
```python
def prepare_data(path1, path2):
    #creating fake news dataframe
    with open(path1) as f:
        fake_list = [line.rstrip('\n') for line in f]
    fake_vectorizer = text.CountVectorizer()
    fake = fake_vectorizer.fit_transform(fake_list)
    fake = fake.toarray()
    columns=fake_vectorizer.get_feature_names()
    df_fake = pd.DataFrame(fake, columns=fake_vectorizer.get_feature_names())
    d = {'class': np.ones(fake.shape[0])}
    y_fake = pd.DataFrame(data=d)
    #creating real news dataframe
    with open(path2) as f:
        real_list = [line.rstrip('\n') for line in f]
    real_vectorizer = text.CountVectorizer()
    real = real_vectorizer.fit_transform(real_list)
    real = real.toarray()
    columns=real_vectorizer.get_feature_names()
    df_real = pd.DataFrame(real, columns=real_vectorizer.get_feature_names())
    d = {'class': np.zeros(real.shape[0])}
    y_real = pd.DataFrame(data=d)
    #concatenating the dataframes
    y_df = pd.concat([y_fake, y_real])
    x_df = df_fake.append(df_real, ignore_index=True, sort=False)
    #filling missing values with mean of column
    x_df = x_df.fillna(x_df.mean())
    #dataset splitting
    x_train, x_test, y_train, y_test = train_test_split(x_df, y_df, test_size=0.3, random_state=1, shuffle=True)
    x_test, x_val, y_test, y_val = train_test_split(x_test, y_test, test_size=0.5, random_state=1)
    return x_train, x_test, x_val, y_train, y_test, y_val
```

b. Implement another function `knn_model_selection()` which utilizes a k-NN classifier to separate real news from the fake ones. Set the values of k to be between 1 and 20, and find the training and validation errors. Display a plot of training and validation accuracy for different values of k. Also determine the best model based on your calculations.

I implemented a function as asked the function gets the metric in order to use in the sklearn KNN model so I didn't have to to implement different functions for part b. and c.
I kept the accuracy for each KNN in a list then I got the index for the maximum accuracy.

```python
def knn_model_selection(met):
    #creating lists to save the accuracy for each knn
    #since sklearn knn gets parameter metric, one can change the metric and use this function for both part b and c
    acc_train = list()
    acc_val = list()
    ks = np.arange(1, 20, 1)
    for k in ks:
        #defining and fitting the data
        neigh = KNeighborsClassifier(n_neighbors=k, metric=met)
        neigh.fit(x_train, y_train)
        #calculating the accuracy
        y_pred = neigh.predict(x_val)
        acc_val.append(accuracy_score(y_val, y_pred))
        y_pred = neigh.predict(x_train)
        acc_train.append(accuracy_score(y_train, y_pred))
    plt.plot(np.arange(1, 20, 1),acc_train)
    plt.plot(np.arange(1, 20, 1),acc_val)
    plt.xlabel('K')
    plt.ylabel('Accuracy')
    plt.title('K-Accuracy (Finding the Best K)')
    plt.show()
    #finding the k that has maximum accuracy
    index = acc_val.index(max(acc_val))
    return ks[index]
```

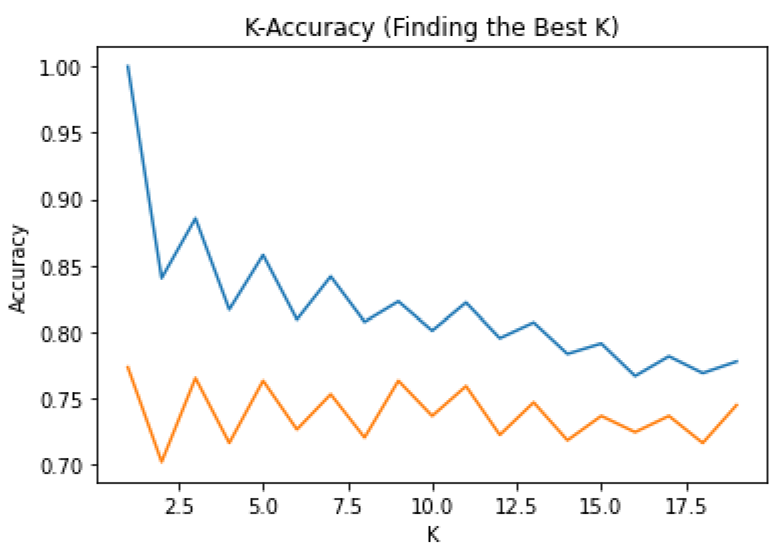
K=3 gave the best answer.




```
Best K: 3
```

c. Repeat the previous part with the distance metric set to cosine. Why might the cosine metric perform better than the Euclidean metric here?
Hint: Consider the set {'dog', 'CR7, 'dog dog dog'}.

K=1 gave the best answer.




```
Best K: 1
```

Cosine will work better when the orientation of the vectors are more important, not the magnitude.

Consider a situation where one text has more of a specific word and another text has a lot less of that word. So these two to might be far from each other when we use Euclidean distance, they would have similarities in there orientation. If the similarity equals 1, it means the two texts are in the same direction and therefore the same group. If it equals 0, there might be some similarities (perpendicular: 0). If it equals -1 they are in completely different directions and therefore no similarities.

5. A Glance At the World of Linear Discriminant Analysis

First, assume a two-category classification problem, where there are two sets of data with normal distribution such that:

$$P_1 = 0.3\,P_2, \quad \mu_1 = \begin{bmatrix} -3 \\ 0 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} 3 \\ 0 \end{bmatrix}, \quad \Sigma_1 = \begin{bmatrix} 1.5 & 1 \\ 1 & 1.5 \end{bmatrix}, \quad \Sigma_2 = \begin{bmatrix} 1.5 & -1 \\ -1 & 1.5 \end{bmatrix}$$

a. Find the linear discriminant function which maximises the Fisher criterion.

$$\vec{w} = S_w^{-1}(\vec{\mu}_1 - \vec{\mu}_2) \propto (\Sigma_1 + \Sigma_2)^{-1}(\vec{\mu}_1 - \vec{\mu}_2) = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}^{-1} \begin{bmatrix} -6 \\ 0 \end{bmatrix}$$

$$\vec{w} \propto \frac{1}{3^2} \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} -6 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{3} \end{bmatrix} \begin{bmatrix} -6 \\ 0 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

b. Minimize the error by adjusting the threshold.

$$g_{12} = w^T(x - x_0)$$

$$w^T = [-2 \quad 0] \qquad x_0 = \frac{1}{2}(\mu_2 + \mu_1) - \frac{(\mu_1 - \mu_2)}{(\mu_1 - \mu_2)^T \Sigma^{-1} (\mu_1 - \mu_2)} \ln \frac{P(\omega_1)}{P(\omega_2)}$$

$$x_0 = \frac{1}{2}\begin{bmatrix} 0 \\ 0 \end{bmatrix} - \frac{\begin{bmatrix} -6 \\ 0 \end{bmatrix}}{[-2 \quad 0]\begin{bmatrix} -6 \\ 0 \end{bmatrix}} \ln(0.3) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} -0.5 \\ 0 \end{bmatrix} \ln(0.3) = \begin{bmatrix} 0.5 \ln(0.3) \\ 0 \end{bmatrix}$$

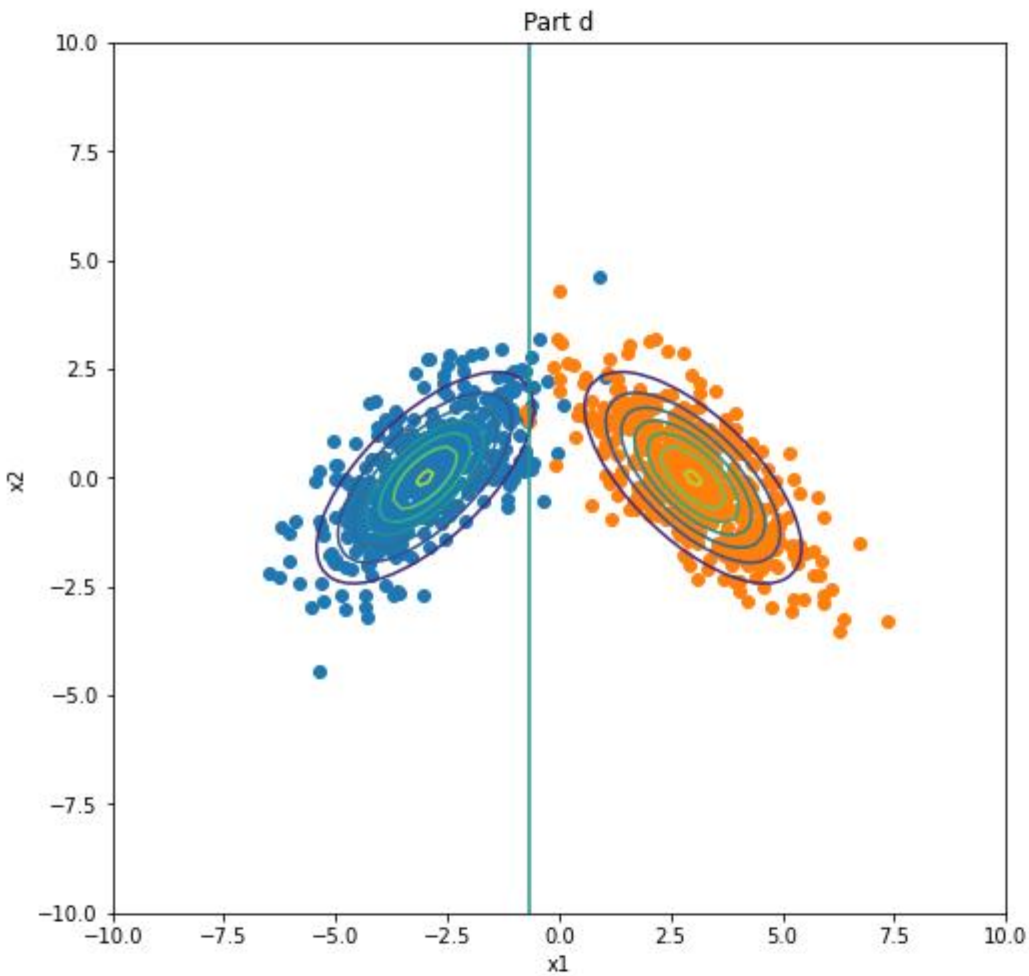
$$g_{12} = [-2 \quad 0]\left(x - \begin{bmatrix} 0.5 \ln(0.3) \\ 0 \end{bmatrix}\right) = [-2 \quad 0]\left(x - \begin{bmatrix} 0.6019864 \\ 0 \end{bmatrix}\right)$$

c. Use the linear discriminant function to classify the point $x = [-0.5, 0.25]^T$.

$$[-2 \quad 0]\left(\begin{bmatrix}-0.5\\0.25\end{bmatrix}-\begin{bmatrix}0.5\ln(0.3)\\0\end{bmatrix}\right)=[-2 \quad 0]\begin{bmatrix}-0.5(1+\ln(0.3))\\0.25\end{bmatrix}$$

$$= 1+\ln(0.3) = -0.2039728 < 0 \Rightarrow \omega_2$$

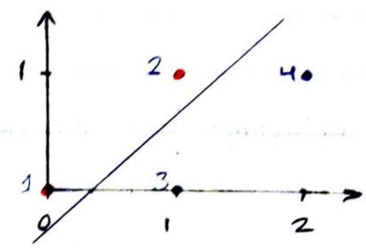
d. Generate 500 samples for each class (1000 in total) and sketch the separating line obtained in part (a).



The line won't be exactly in the middle because of $p_1 = 0.3p_2$.

Next, consider a 2-class classification problem as below:

e. Find the line separating the classes using the Perceptron algorithm in its reward and punishment form, with $\rho = 1$ and $\omega(0) = [0,0]^T$.



1: $[1\ 0\ 0]^T$ 2: $[1\ 1\ 1]^T$ 3: $[-1\ -1\ 0]^T$ 4: $[-1\ -2\ -1]^T$

$$w(K+1) = w(K) + \rho\, Y_M$$

$$w^T y_i = \sum_{k=0}^{3} w_k y_i^{(k)} \leq 0 \quad \xleftarrow{\text{if}} \quad \text{misclassified}$$

$$[0\ 0\ 0]\begin{bmatrix}1\\0\\0\end{bmatrix} = 0 \leq 0 \Longrightarrow w(1) = [0\ 0\ 0] + [1\ 0\ 0] = [1\ 0\ 0]$$

$$[1\ 0\ 0]\begin{bmatrix}1\\1\\1\end{bmatrix} = 1 > 0 \quad \checkmark$$

$$[1\ 0\ 0]\begin{bmatrix}-1\\-1\\0\end{bmatrix} = -1 \leq 0 \Rightarrow w(2) = [1\ 0\ 0] + [-1\ -1\ 0] = [0\ -1\ 0]$$

$$[0\ -1\ 0]\begin{bmatrix}-1\\-2\\-1\end{bmatrix} = 2 > 0 \Rightarrow \checkmark$$

$$[0\ -1\ 0]\begin{bmatrix}1\\0\\0\end{bmatrix} = 0 \leq 0 \Rightarrow w(3) = [0\ -1\ 0] + [1\ 0\ 0] = [1\ -1\ 0]$$

$$[1\ -1\ 0]\begin{bmatrix}1\\1\\1\end{bmatrix} = 0 \leq 0 \Rightarrow w(4) = [1\ -1\ 0] + [1\ 1\ 1] = [2\ 0\ 1]$$

$$[2\ 0\ 1]\begin{bmatrix}-1\\-1\\0\end{bmatrix} = -2 \leq 0 \Rightarrow w(5) = [2\ 0\ 1] + [-1\ -1\ 0] = [1\ -1\ 1]$$

$$[1\ -1\ 1]\begin{bmatrix}-1\\-2\\-1\end{bmatrix} = 0 \leq 0 \Rightarrow w(6) = [1\ -1\ 1] + [-1\ -2\ -1] = [0\ -3\ 0]$$

$$[0\ \ -3\ \ 0]\begin{bmatrix}1\\0\\0\end{bmatrix}=0\leq 0\Rightarrow w(7)=[0\ \ -3\ \ 0]+[1\ \ 0\ \ 0]=[1\ \ -3\ \ 0]$$

$$[1\ \ -3\ \ 0]\begin{bmatrix}1\\1\\1\end{bmatrix}=-2\leq 0\Rightarrow w(8)=[1\ \ -3\ \ 0]+[1\ \ 1\ \ 1]=[2\ \ -2\ \ 1]$$

$$[2\ \ -2\ \ 1]\begin{bmatrix}-1\\-1\\0\end{bmatrix}=0\leq 0\Rightarrow w(9)=[2\ \ -2\ \ 1]+[-1\ \ -1\ \ 0]=[1\ \ -3\ \ 1]$$

$$[1\ \ -3\ \ 1]\begin{bmatrix}-1\\-2\\-1\end{bmatrix}=4>0\Rightarrow\checkmark$$

$$[1\ \ -3\ \ 1]\begin{bmatrix}1\\0\\0\end{bmatrix}=1>0\Rightarrow\checkmark$$

$$[1\ \ -3\ \ 1]\begin{bmatrix}1\\1\\1\end{bmatrix}=-1\leq 0\Rightarrow w(10)=[1\ \ -3\ \ 1]+[1\ \ 1\ \ 1]=[2\ \ -2\ \ 2]$$

$$[2\ \ -2\ \ 2]\begin{bmatrix}-1\\-1\\0\end{bmatrix}=0\leq 0\Rightarrow w(11)=[2\ \ -2\ \ 2]+[-1\ \ -1\ \ 0]=[1\ \ -3\ \ 2]$$

$$[1\ \ -3\ \ 2]\begin{bmatrix}-1\\-2\\-1\end{bmatrix}=3>0\Rightarrow\checkmark$$

$$[1\ \ -3\ \ 2]\begin{bmatrix}1\\0\\0\end{bmatrix}=1>0\Rightarrow\checkmark$$

$$[1\ \ -3\ \ 2]\begin{bmatrix}1\\1\\1\end{bmatrix}=0\leq 0\Rightarrow w(12)=[1\ \ -3\ \ 2]+[1\ \ 1\ \ 1]=[2\ \ -2\ \ 3]$$

$$[2\ \ -2\ \ 3]\begin{bmatrix}-1\\-1\\0\end{bmatrix}=0\leq 0\Rightarrow w(13)=[2\ \ -2\ \ 3]+[-1\ \ -1\ \ 0]=[1\ \ -3\ \ 3]$$

$$[1\ \ -3\ \ 3]\begin{bmatrix}-1\\-2\\-1\end{bmatrix}=2>0\Rightarrow\checkmark$$

$$[1\ \ -3\ \ 3]\begin{bmatrix}1\\0\\0\end{bmatrix}=1>0\Rightarrow\checkmark$$

$$[1 \quad -3 \quad 3]\begin{bmatrix}1\\1\\1\end{bmatrix} = 1 > 0 \Rightarrow \checkmark$$

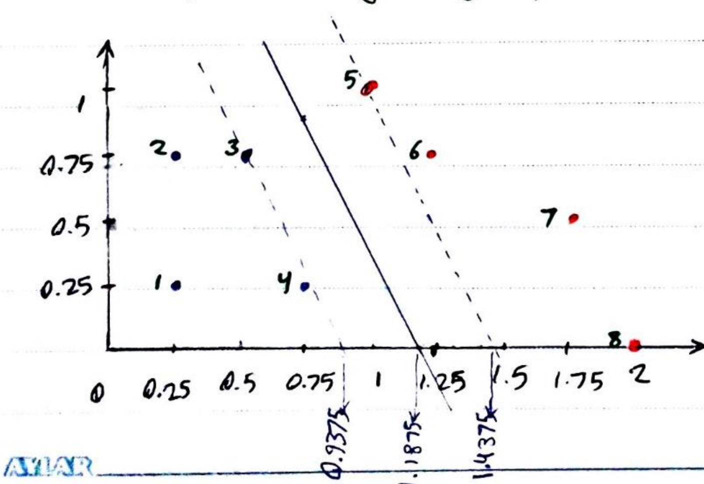
$$[1 \quad -3 \quad 3]\begin{bmatrix}-1\\-1\\0\end{bmatrix} = 2 > 0 \Rightarrow \checkmark$$

$$[1 \quad -3 \quad 3]\begin{bmatrix}-1\\-2\\-1\end{bmatrix} = 2 > 0 \Rightarrow \checkmark$$

Finally, consider the dataset in the following table:

f. Plot the sample points, and try to construct the weight vector for the optimal hyperplane and the optimal margin by inspection.

	Class	Point
1	ω_1	(0.25, 0.25)
2	ω_1	(0.25, 0.75)
3	ω_1	(0.5, 0.75)
4	ω_1	(0.75, 0.25)
5	ω_2	(1, 1)
6	ω_2	(1.25, 0.75)
7	ω_2	(1.75, 0.5)
8	ω_2	(2, 0)



I need to include that at first I did consider 3,4,5 as support vectors. The issue was when calculating lagrange coefficients, α_4 became equel to zero but in the end using quadprog in matlab you can see that α_4 becomes 0.0002 and not zero.

Anyways, I wrote the answer considering 3,5 as the support vectors.

Because I found 3, 5 (each from a different class than other) as two closest points, by inspection I considered the hyperplane to be:

$$y = \left(-\frac{1}{\frac{1-0.75}{1-0.5}}\right)\left(x - \frac{1+0.5}{2}\right) + \frac{1+0.75}{2} = -2x + 1.5 + \frac{7}{8} = -2x + \frac{19}{8}$$

$$\Downarrow$$

$$w = \begin{bmatrix} 2c & c \end{bmatrix} \qquad b = -\frac{19}{8}c$$

$$\text{margin} = \frac{2}{\|w\|} = \frac{2}{\sqrt{2^2+1^2}} = \frac{2}{\sqrt{5}\,c}$$

$$\text{margin} = d(v_3, v_5) \Rightarrow \frac{2}{\sqrt{5}\,c} = \sqrt{\frac{5}{16}} \Rightarrow \frac{4}{5c^2} = \frac{5}{16}$$

$$\frac{1}{c^2} = \frac{5}{16} \times \frac{5}{4} \Rightarrow c^2 = \frac{16\times 4}{5\times 5} = \frac{64}{25} \Rightarrow c = \frac{8}{5}$$

$$\Rightarrow \quad w = \begin{bmatrix} \frac{16}{5} & \frac{8}{5} \end{bmatrix} \qquad b = -\frac{19}{5} \qquad \Rightarrow \frac{16}{5}x + \frac{8}{5}y - \frac{19}{5} = 0$$

g. Determine the support vectors.

The vectors that have $\frac{1}{\|w\|}$ distance from the hyperplane are support vectors.

$$\frac{1}{\|w\|} = \frac{1}{\sqrt{(\frac{16}{5})^2 + (\frac{8}{5})^2}} = \frac{1}{\sqrt{\frac{320}{25}}} = \frac{5}{\sqrt{64\times 5}} = \frac{\sqrt{5}}{8}$$

or I could say the margin lines are:

$$\frac{16}{5}x + \frac{8}{5}y - \underset{\downarrow\; -\frac{19}{5} - \frac{1}{4}}{\frac{81}{20}} = 0 \qquad \frac{16}{5}x + \frac{8}{5}y - \underset{\downarrow\; -\frac{19}{5} + \frac{1}{4}}{\frac{71}{20}} = 0 \qquad \Rightarrow \text{support vectors: } 3, 5$$

h. Find the solution in the dual space by finding the Lagrange multipliers α_i. Compare the result with part (f).

$$L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^T x_j$$

we know our support vectors are points 3,5, so α for other points would be zero.

$$\Rightarrow L(\alpha) = \alpha_3 + \alpha_5 - \frac{1}{2} (\alpha_3 \alpha_3 z_3 z_3 x_3^T x_3 + \alpha_3 \alpha_5 z_3 z_5 x_3^T x_5 + \alpha_5 \alpha_3 z_5 z_3 x_5^T x_3 +$$

$$\alpha_5 \alpha_5 z_5 z_5 x_5^T x_5) = \alpha_3 + \alpha_5 - \frac{1}{2} (\alpha_3^2 [\tfrac{1}{2} \ \ \tfrac{3}{4}] \begin{bmatrix} \frac{1}{2} \\ \frac{3}{4} \end{bmatrix} - 2\alpha_3 \alpha_5 [\tfrac{1}{2} \ \ \tfrac{3}{4}] \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \alpha_5^2 [1 \ \ 1] \begin{bmatrix} 1 \\ 1 \end{bmatrix})$$

$$= \alpha_3 + \alpha_5 - \frac{1}{2} (\frac{13}{16} \alpha_3^2 - \frac{5}{2} \alpha_3 \alpha_5 + 2 \alpha_5^2) = \alpha_3 + \alpha_5 - \frac{13}{32} \alpha_3^2 + \frac{5}{4} \alpha_3 \alpha_5 - \alpha_5^2$$

we know that $\sum_{i=1}^{n} \alpha_i z_i = 0$.

So $\Rightarrow \alpha_3 - \alpha_5 = 0 \Rightarrow \alpha_3 = \alpha_5$

$$L(\alpha) = 2\alpha_3 - \frac{13}{32} \alpha_3^2 + \frac{5}{4} \alpha_3^2 - \alpha_3^2 = 2\alpha_3 - \frac{5}{32} \alpha_3^2$$

we want to maximize $L(\alpha) \Rightarrow \dfrac{\partial L(\alpha)}{\partial \alpha_3} = 2 - \dfrac{10}{32} \alpha_3 = 0$

$$\Rightarrow \frac{10}{32} \alpha_3 = 2 \Rightarrow \alpha_3 = 2 \times \frac{32}{10} = \frac{64}{10}$$

$$\Rightarrow \alpha_5 = \frac{64}{10}$$

I used quadprog in matlab to compute α_i.

Turns out $\alpha_3 = 6.3998$ $\alpha_4 = 0.0002$ $\alpha_5 = 6.4$ so support vectors are indeed points 3, 4, 5.

You can find this by the name of " P5h " in the files

$$\omega = \sum_{i=1}^{n} \alpha_i z_i x_i = (\alpha .* z)^T x = ([0, \dots, 6.3998, 0.0002, 6.4, \dots 0]) x = [-3.1999 \ -1.6001]$$

$$b = \frac{1}{z_3} - \omega^T x_3 = 3.8.001$$

6. Categorizing Different Antarctic Penguin Species.

a. Apply basic gradient descent and Newton's algorithm to the samples in Gentoo and Adélie categories. Set $\eta(k) = 0.1$. Display the criterion function as function of the iteration number. Also display the data distribution as well as the decision boundaries, and report the prediction accuracy on test samples.

Gradient Descent:

I tried this with $\eta(k) = 0.1$ and #Iteration: 500. I also used normalization. The reason I put #Iteration = 500 was that I could show to what iteration this learning rate would do well.

Then I got the theta for best number of iteration and calculated MSE and Accuracy for the test

MSE for test set ≃ 0.84 Accuracy = 1 for # Iter. = 17 (start: 0)

While after 500 iterations MSE was 1.16 and Accuracy = 0.22.

The best theta: theta[0] ≃ 0.57 theta[1] ≃ 0.42 theta[2] ≃ 0.41.

The necessary pictures for the outcome are included in the files, folder 'P6'.

Newton's Method

I set the same settings: # Iter = 500, $\eta(k)$ = 0.1 and normalization

The best mse and accuracy was given at iteration 497 (start: 0)

meaning the last iteration and

MSE for test set ≃ 1.07 Accuracy = 1

theta[0] ≃ 0.61 theta[1] ≃ 0.57 theta[2] ≃ 0.55

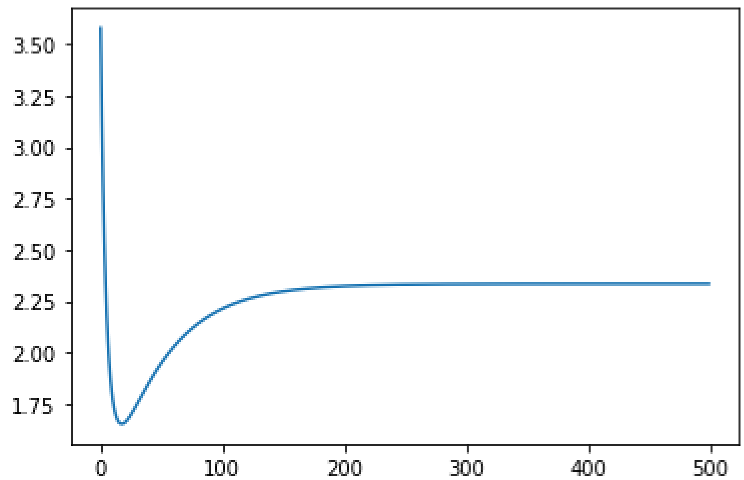
prepare-data will get the class types and return the necessary data. (here: 'Gentoo' and 'Adelie')

These are part a gradient descent:

```
Theta:  [[ 1.00007724]
 [ 0.00860324]
 [-0.00844733]]
Train Mean Squared Error:  2.33468348743862
Test Mean Squared Error:  1.1615464813050504
Test Accuracy:  0.22580645161290322
```

In [13]:
```
plt.plot(np.arange(num_iter),mselist1)
```

Out[13]: [<matplotlib.lines.Line2D at 0x1b24b782cc0>]



In [15]:
```
index1 = np.where(mselist1==np.min(mselist1))[0]
print(mselist1[np.where(mselist1==np.min(mselist1))])
print(index1[0])
best_theta1 = theta_per_iter1[index1[0]]
```

```
[1.65348051]
17
```

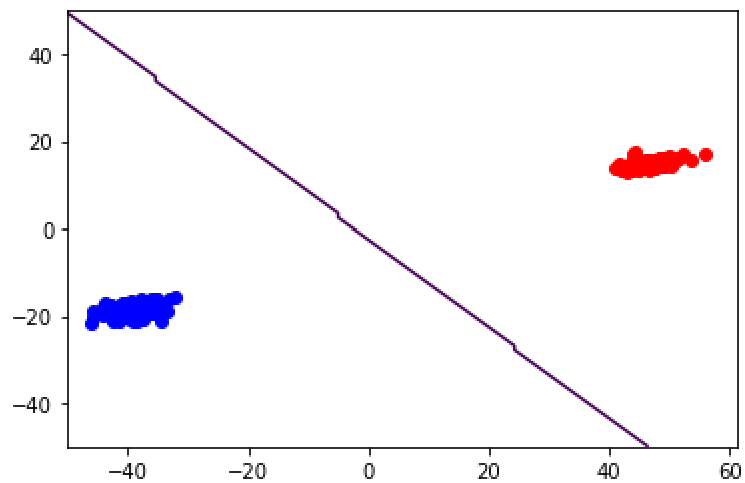
In [16]:
```
print(best_theta1)
```

```
[[0.59102189]
 [0.42504251]
 [0.41100999]]
```

In [17]:
```
h_of_x_p1, y_pred1 = Predict( x_test1, best_theta1)
mse1 = ComputeCost(y_test1, h_of_x_p1)
acc1 = accuracy(y_test1, y_pred1)
print("Test Mean Squared Error: ", mse1)
print("Test Accuracy: ", acc1)
```

```
Test Mean Squared Error:  0.8412184564255221
Test Accuracy:  1.0
```

Out[18]: <matplotlib.collections.PathCollection at 0x1b24b8b4748>

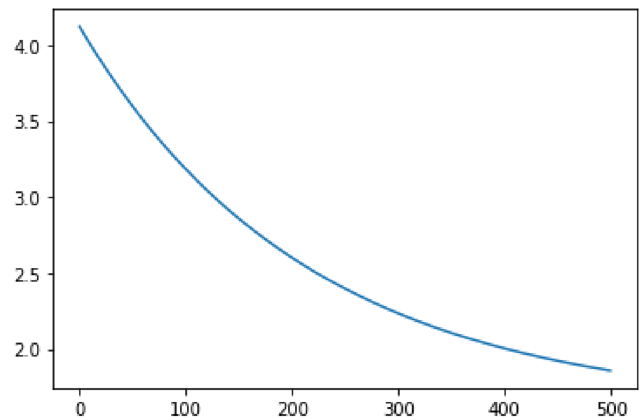


These are part a newton's method:

```
Theta:  [[0.61020646]
 [0.57047823]
 [0.55918126]]
Train Mean Squared Error:  1.8588529748781764
Test Mean Squared Error:  1.0717056217096361
Test Accuracy:  1.0
```

In [20]:
```
plt.plot(np.arange(num_iter),mselist2)
```

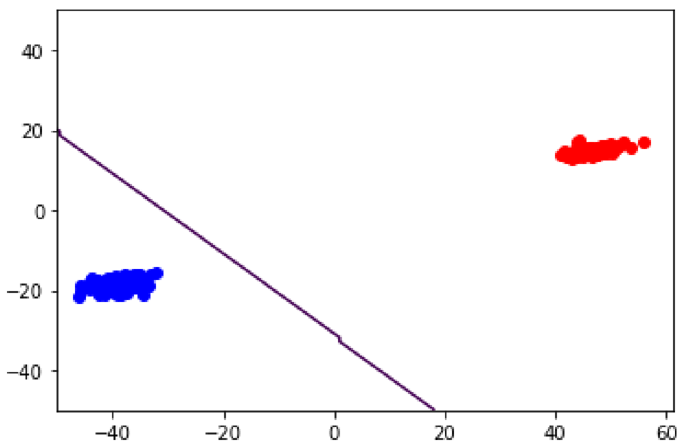
Out[20]:
```
[<matplotlib.lines.Line2D at 0x1b24d9616d8>]
```



499

In [22]:
```
xlist = np.linspace(-50.0, 50.0, 100)
ylist = np.linspace(-50.0, 50.0, 100)
X, Y = np.meshgrid(xlist, ylist)
xplot = np.append(np.expand_dims(np.ones(X.ravel().shape),
xplot = np.append(xplot,np.expand_dims(Y.ravel(), axis=1),
Z = Predict(xplot,best_theta2)
plt.contour(X,Y,Z[1].reshape(X.shape), levels=[0])
plt.scatter(gentoo.iloc[:,0], gentoo.iloc[:,1],color='red'
plt.scatter(adelie.iloc[:,0], adelie.iloc[:,1],color='blue
```

Out[22]: <matplotlib.collections.PathCollection at 0x1b24d9cd080>

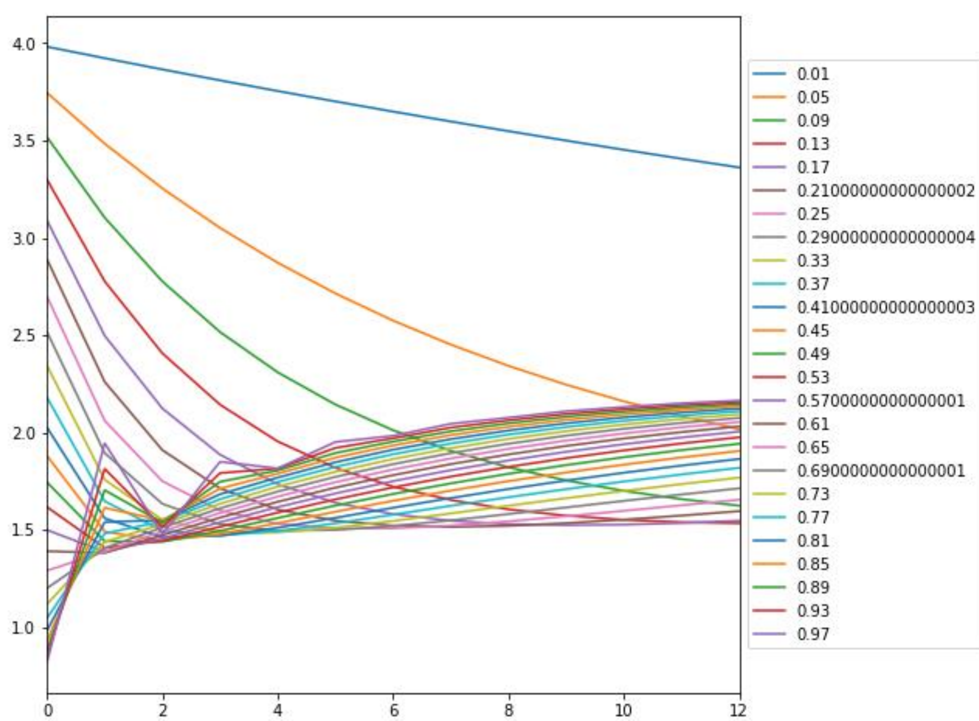


b. Display the variation of convergence time versus learning rate. What is the minimum learning rate that fails to lead convergence?

I set # Iteration = 100 but I'm displaying [0-12] for better seeing the divergence.

I also set $\eta(k)$ from 0.01 to 0.99 by step 0.04.

As you can see in the picture in Folder 'P6' somewhere around $\eta(k) = 0.61$ the algorithm starts with divergence. and the 'MSE' error gets larger and larger, there's also some spikes in the beginning for $\eta(k) > 0.61$.



d. Repeat the previous parts for the samples in Chinstrap and Gentoo categories. What are your observations?

Gradient Descent:

The initiations don't differ from part a and b.

The best MSE was at iteration 15 (start: 0).

The results:

MSE after 500 iterations for test set ≈ 1.29

Accuracy: 0.84

MSE at iteration 15: 1.16 Accuracy: 1

Theta[0] ≃ 0.47 Theta[1] ≃ 0.38 theta[2] ≃ 0.35

Newton's Method:

best MSE was for the last iteration.

The results:

MSE for test set ≃ 1.18 Accuracy = 1

theta[0] ≃ 0.48 theta[1] ≃ 0.4 theta[2] ≃ 0.37

In all of these Gentoo is the red dots.

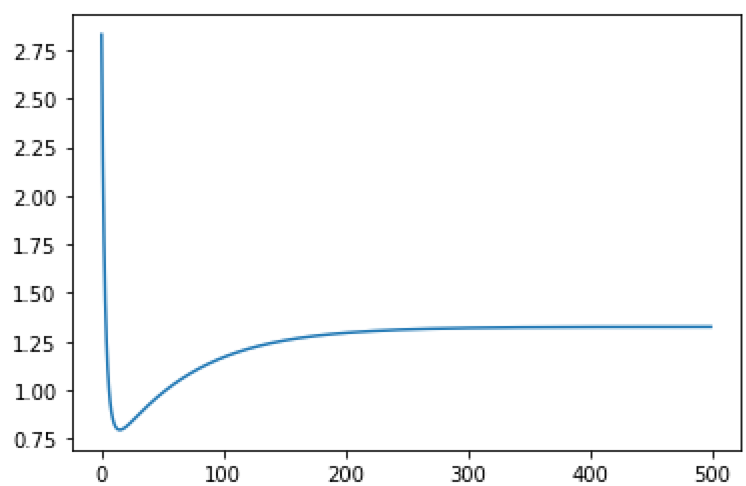
For different learning rates, I did the same thing as before. At about $\eta(K) = 0.5$ the algorithm starts with divergence and MSE gets larger and larger and there are way more sharp spikes.

Gradient descent part:

```
Theta:  [[ 0.99907684]
 [ 0.03504257]
 [-0.03247072]]
Train Mean Squared Error:  1.3250706922398336
Test Mean Squared Error:  1.999700220719057
Test Accuracy:  0.8461538461538461
```

```
In [13]: plt.plot(np.arange(num_iter),mselist1)

Out[13]: [<matplotlib.lines.Line2D at 0x14bd948c748>]
```



In [14]:
```
index1 = np.where(mselist1==np.min(mselist1))[0]
print(mselist1[np.where(mselist1==np.min(mselist1))])
print(index1[0])
best_theta1 = theta_per_iter1[index1[0]]
```

```
[0.79487784]
15
```

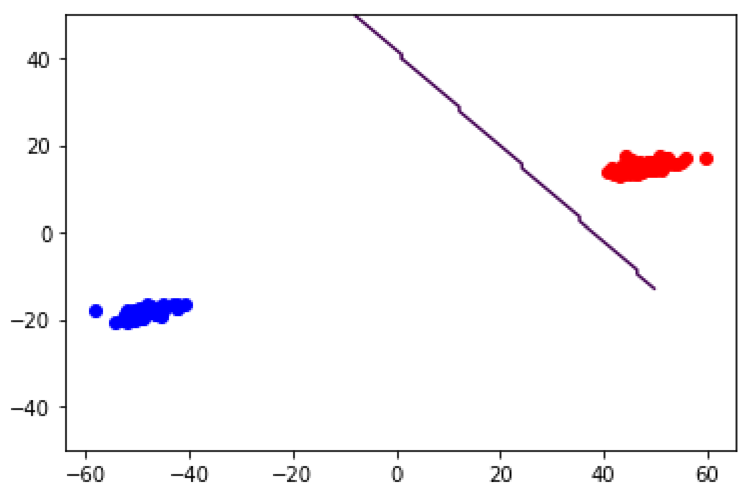
In [15]:
```
print(best_theta1)
```

```
[[0.47830022]
 [0.38917782]
 [0.35749298]]
```

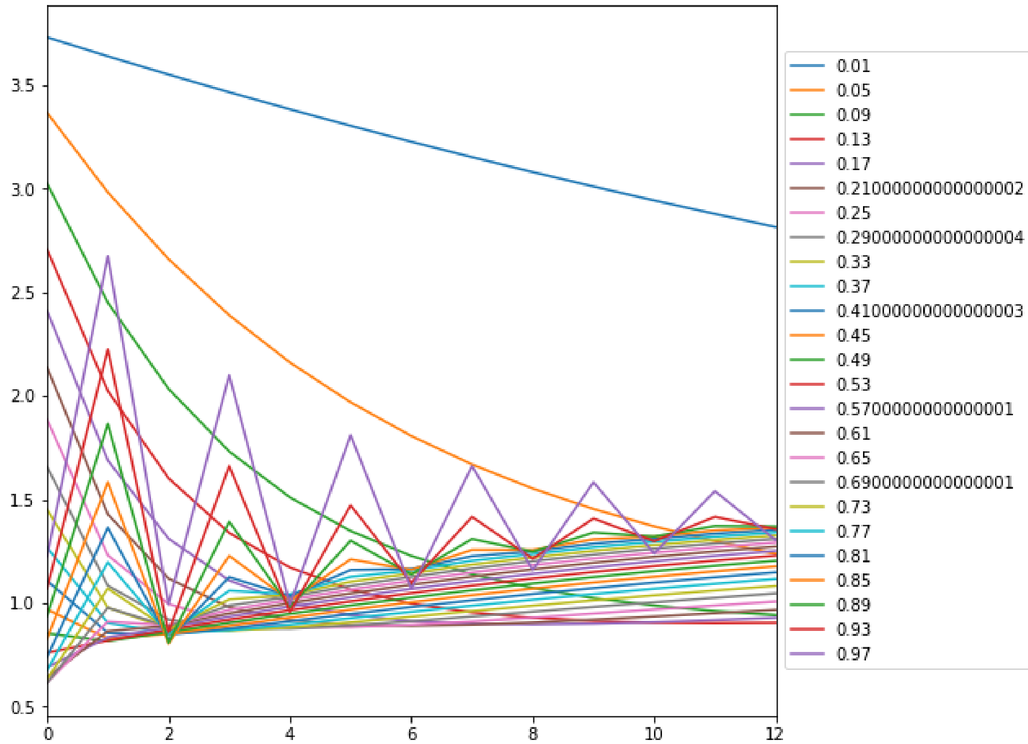
In [16]:
```
h_of_x_p1, y_pred1 = Predict( x_test1, best_theta1)
mse1 = ComputeCost(y_test1, h_of_x_p1)
acc1 = accuracy(y_test1, y_pred1)
print("Test Mean Squared Error: ", mse1)
print("Test Accuracy: ", acc1)
```

```
Test Mean Squared Error:  1.1692486768201293
Test Accuracy:  1.0
```

Out[17]:
```
<matplotlib.collections.PathCollection at 0x14bd964d860>
```



Learning Rate Change:

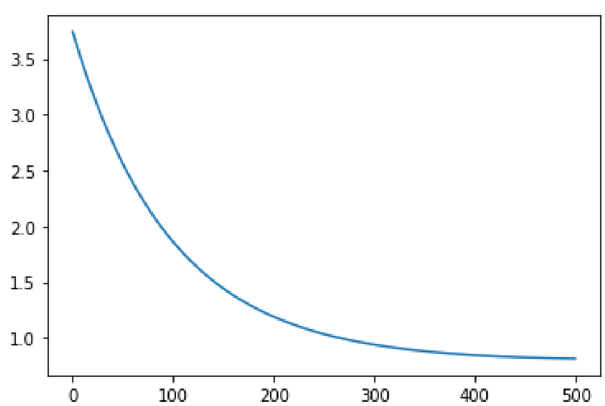


Newton's method part:

```
Theta:  [[0.48373428]
 [0.40619584]
 [0.37559562]]
Train Mean Squared Error:  0.8121038424829041
Test Mean Squared Error:  1.1885643323834536
Test Accuracy:  1.0
```

In [19]:
```
plt.plot(np.arange(num_iter),mselist2)
```

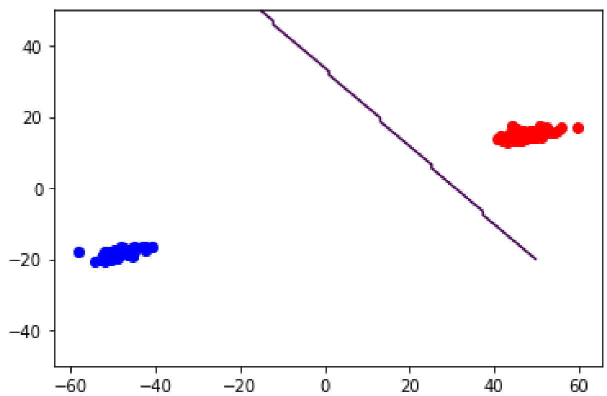
Out[19]: [<matplotlib.lines.Line2D at 0x14bdb705da0>]



```
499
```

```
In [21]: xlist = np.linspace(-50.0, 50.0, 100)
         ylist = np.linspace(-50.0, 50.0, 100)
         X, Y = np.meshgrid(xlist, ylist)
         xplot = np.append(np.expand_dims(np.ones(X.ravel().
         xplot = np.append(xplot,np.expand_dims(Y.ravel(), a
         Z = Predict(xplot,best_theta2)
         plt.contour(X,Y,Z[1].reshape(X.shape), levels=[0])
         plt.scatter(w0.iloc[:,0], w0.iloc[:,1],color='red')
         plt.scatter(w1.iloc[:,0], w1.iloc[:,1],color='blue'

Out[21]: <matplotlib.collections.PathCollection at 0x14bdc73
```



e. Starting with $a=0$, apply the Perceptron algorithm to the samples from Gentoo and Adélie. Report the number of iterations needed for convergence. Also display the decision boundary on the samples distribution and report the prediction accuracy on test samples.

Best Iteration was 24. Test MSE = 0 Accuracy = 1.

theta[0] ≈ -0.031 theta[1] ≈ 0.085 theta[2] ≈ 0.029

After 500 iterations MSE ≈ 1.25 Accuracy ≈ 0.68

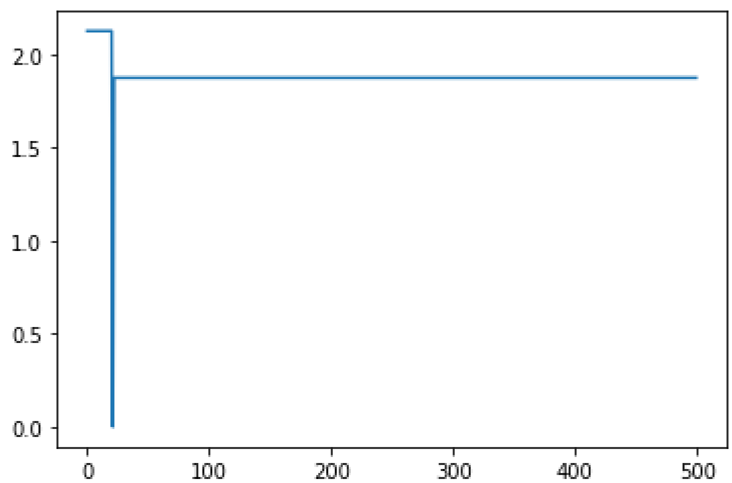
(Need to remind you: I'm shuffling data, not always the shuffled data is collected evenly.)

All iterations:

```
Theta:  [[-22.4439834 ]
 [-19.77816926]
 [-21.06469591]]
Train Mean Squared Error:  1.8755186721991701
Test Mean Squared Error:  1.2571428571428571
Test Accuracy:  0.6857142857142857
```

In [52]:
```
plt.plot(np.arange(num_iter),mselist1)
```

Out[52]: [<matplotlib.lines.Line2D at 0x20022408ac8>]



Best Model:

In [54]:
```
print(best_theta1)
```

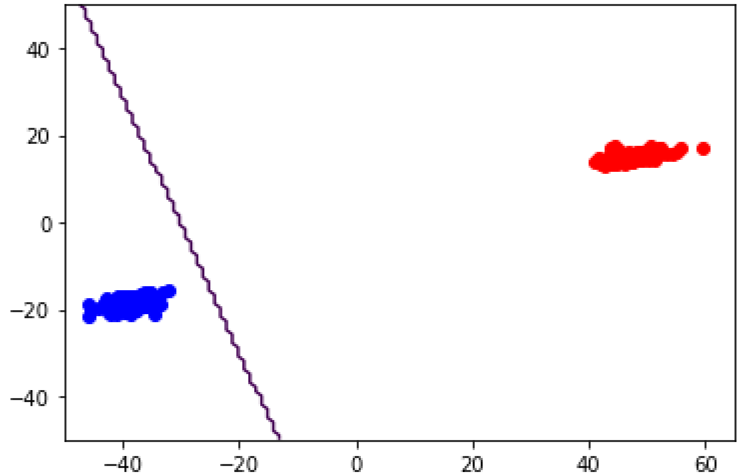
```
[[-0.03153527]
 [ 0.08576055]
 [ 0.02915338]]
```

In [55]:
```
h_of_x_p1, y_pred1 = Predict( x_test1, best_theta1)
mse1 = ComputeCost(y_test1, h_of_x_p1)
acc1 = accuracy(y_test1, y_pred1)
print("Test Mean Squared Error: ", mse1)
print("Test Accuracy: ", acc1)
```

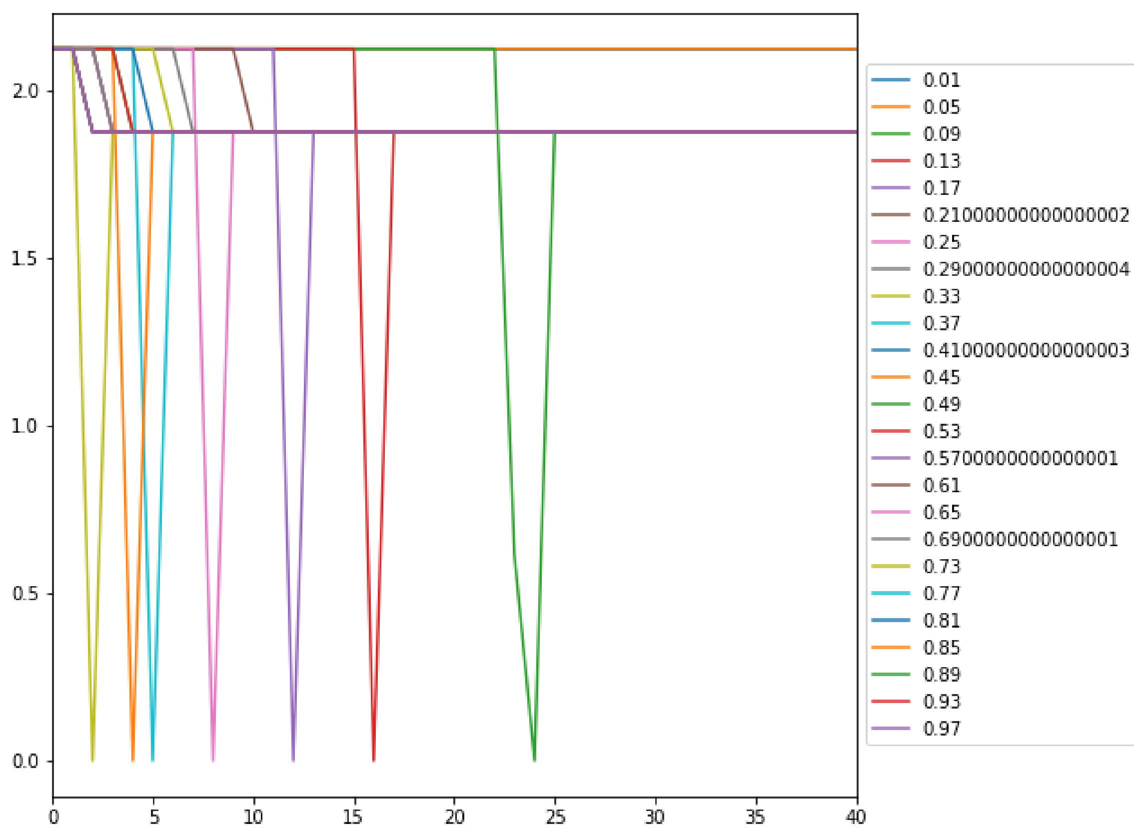
```
Test Mean Squared Error:  0.0
Test Accuracy:  1.0
```

Decision Boundary:

```
Out[56]: <matplotlib.contour.QuadContourSet at 0x20022981c18>
```



Learning Rate change:



f. Repeat the previous part for samples from Chinstrap and Gentoo.

Best Iteration was 22 Test MSE = 0 Accuracy = 1

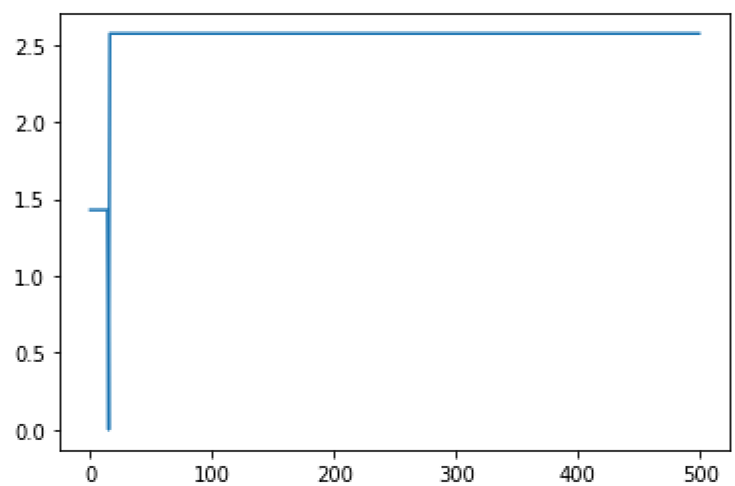
theta[0] $\simeq -0.028$ theta[1] $\simeq 0.076$ theta[2] $\simeq 0.032$

All iterations:

```
Theta:  [[-31.14285714]
 [-27.86682542]
 [-29.24505663]]
Train Mean Squared Error:  2.571428571428571
Test Mean Squared Error:  2.6666666666666665
Test Accuracy:  0.3333333333333333
```

In [12]:
```
plt.plot(np.arange(num_iter),mselist1)
```

Out[12]: [<matplotlib.lines.Line2D at 0x2002210e630>]



Best model:

In [14]:
```
print(best_theta1)
```

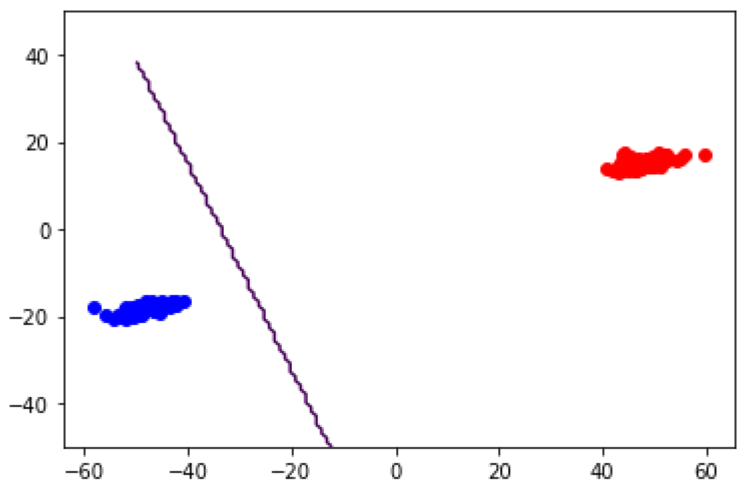
```
[[-0.02857143]
 [ 0.07626159]
 [ 0.03215819]]
```

In [15]:
```
h_of_x_p1, y_pred1 = Predict( x_test1, best_theta1)
mse1 = ComputeCost(y_test1, h_of_x_p1)
acc1 = accuracy(y_test1, y_pred1)
print("Test Mean Squared Error: ", mse1)
print("Test Accuracy: ", acc1)
```

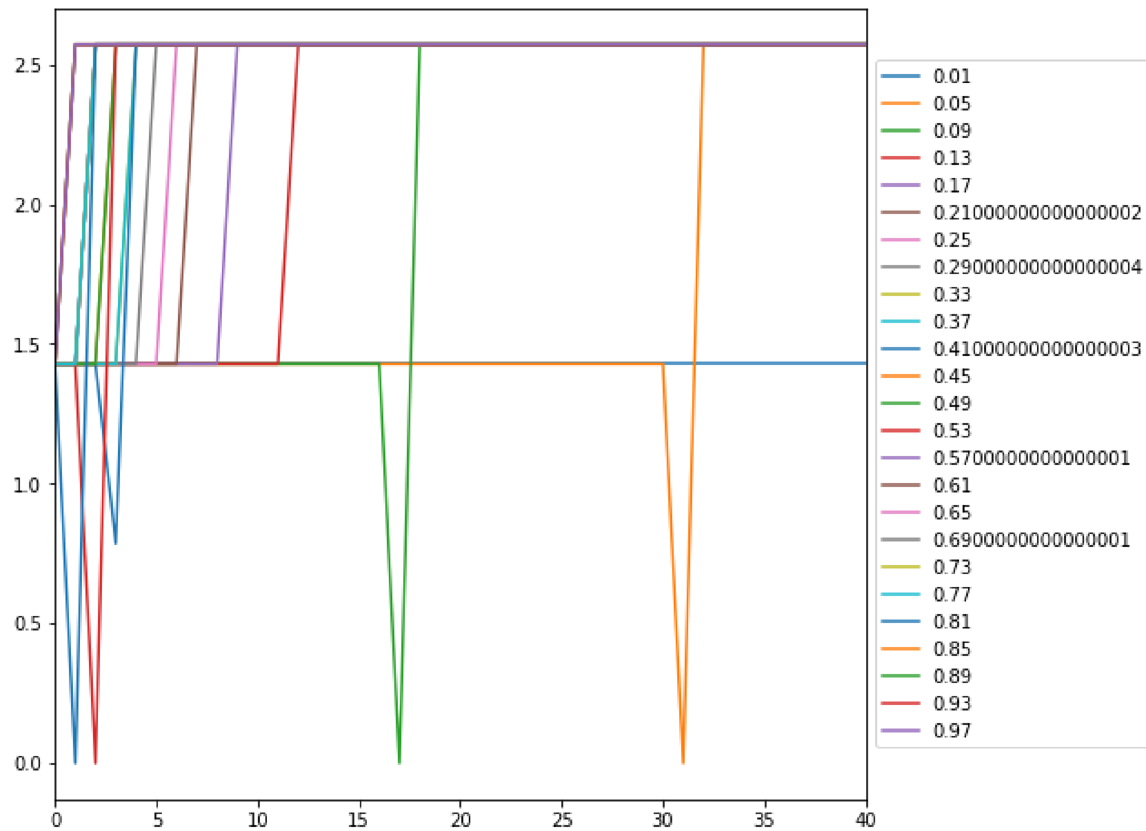
```
Test Mean Squared Error:  0.0
Test Accuracy:  1.0
```

Decision Boundary:

Out[16]:
```
<matplotlib.contour.QuadContourSet at 0x2002221dd68>
```



Learning Rate change:



After 500 iteration Test MSE ≈ 2.66 , Accuracy ≈ 0.33.

g. Comment on the difference between the iterations needed for convergence in the two cases.

In this case you have to be careful to find a specific amount of iteration. It seems MSE converges faster and also Gentoo, Chinstrip seem to get to convergence faster.

There is an order to the learning rate choice in MSE.

7. Some Explanatory Questions

a. In Parzen window density estimation, is there a method to find an appropriate value for bandwidth? Explain.

As we want the bandwidth that minimizes the error between the estimated density and the true density, we could use MSE. There is a method that minimizes the mean integrated squared error. We assume a standard density function and find the optimal bandwidth by Minimizing MISE.

$$h_{opt} = \underset{h}{\operatorname{argmin}} \left\{ E\left| \int (P_{KDE}(x) - p(x))^2 dx \right| \right\}$$

We use a standard density function since we usually have a lot features and understanding what $p(x)$ is likely to be our true distribution just by looking at the data would not be possible.

b. Are KNN and minimum distance classifiers related? If yes, how? If no, why?

Minimum distance classifiers are parametric while KNN is a non-parametric method. We're not using any class specific parameters.

Now if we consider $k=1$ it might seem as if they are doing the same thing (if we are using same distance metric. euclidean distance) but still the truth behind it is that Knn is using the samples while minimum distance classifiers use distribution related parameters.

c. Is KNN decision boundary always linear? Justify your answers with examples.

No it's not. KNN decision boundary always depends on the dataset. So in a specific case it might become linear but in general most of the times it's non-linear.

Also the distance function used to find the k-nearest neighbors is not linear.

d. What is the relationship between the values of k and the smoothness of the decision boundary of a KNN classifier?

The higher the K the smoother the boundary. You see, by increasing K, bias gets higher and variance decreases. Therefore a few samples will not make any bumps in a decision boundary.

e. Consider a two-class classification problem in which there are samples of both classes in the Training set. Is it possible for a 1-NN classifier to always assign a specific label to all test samples? If yes, give an example. If no, explain.

The only way for this to happen is for all the test samples to actually be closer to the samples from a specific label. Like I said when talking about Knn specially 1-NN the dataset performs a great role. If we are talking in general, no, because there is a possibility that we have a test data close or equal to one of the samples from the other class. (If test samples for the other class were on the boundary we might also end up with the issue in question.)

f. Explain a drawback of a k-NN classifier, and suggest a modification to the method in order to avoid it.

There are some drawbacks in KNN, including:

KNN is slow when it comes to large data, its performance depends on the quality of data. and it's so sensetive to scale and irrelevant features, also if you're gonna be predicting some samples on some other computer, you're taking the training data with you! :)

We could use normalization for the sensetivity to scale, use feature selection and dimension reduction to avoid irrelevant features and also reduce the computational complexity. There are some ways to reduce the computational complexity; One of them is using trees. The data points are used to build a tree structure that hierarchically partitions the data space. KD-Tree is one of those trees that can be used. It's a binary search tree where data in each node is a k-dimentional point in space. (Because binary search decreases the complexity here.)

g. Under what circumstances do you recommend using SVM instead of MLP? What about the reverse? Which problems do both SVM and MLP suffer to solve?

The feature scale would matter a lot in SVM. SVM would perform well on fewer data but with large data it takes a lot of time. If we had a type of dataset where one could draw a decision boundary with a continuos line then SVM will perform fine but imagine XOR problem; MLP would work way better for these types of problems and also in the case we have a lot of data MLP would be faster.

I think they would both suffer to solve the problem in which there are a lot of noise and outliers.

h. In order to apply backpropagation algorithm on a neural network, which topological condition should it have? Explain.

Our Error function must be differentiable and continuos. Since we use gradient dessent then this is the important factor.

I need to add that for part g I had considered SVM with linear kernel. With other kernels it is possible to get a good decision boundary for xor example.

Also at the end I have written about noise and outliers. I have condisered SVM with linear kernel and hard margin.

In other words I have considered the most basic forms.