HW5

Q1 a) Pseudocode:

for local linear regression:

For each x_0: $Y = \alpha(x_0) + \beta(x_0) x_i + \varepsilon$

We find α, β by doing

$$\min_{\alpha(x_0), \beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i) [Y - \alpha(x_0) - \beta(x_0) x_i]^2$$

$$= \frac{1}{\sqrt{2\pi}\lambda} e^{-\frac{1}{2\lambda}(x_i - x_0)^2}$$

1)b)

For each x_0:

$$\min_{\alpha,\beta} \sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[y - \alpha - \beta x_i\right]^2$$

$$K_\lambda(x_0, x_i) \rightarrow = \frac{1}{\sqrt{2\pi}\lambda} e^{-\frac{1}{2\lambda}(x_i - x_0)^2}$$

let $J(\alpha,\beta) = \sum_{i=1}^{N} K_\lambda(x_0, x_i)\,(y - \alpha - \beta x_i)^2$

So, $\frac{dJ}{d\beta} = \frac{d}{d\beta}\left[\sum_{i=1}^{N} K_\lambda(x_0, x_i)(y - \alpha - \beta x_i)^2\right] \overset{\text{Set to}}{=\!=} 0$

$$J(\alpha,\beta) = \sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[y - (\alpha + \beta x_i)\right]^2$$

$$= \sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[y^2 - 2y(\alpha + \beta x_i) + \alpha^2 + 2\alpha\beta x_i + \beta^2 x_i^2\right]$$

$$= \sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[y^2 - 2y\alpha - 2y\beta x_i + \alpha^2 + 2\alpha\beta x_i + \beta^2 x_i^2\right]$$

$$= \sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[y^2 - 2y\alpha + \alpha^2 + (-2yx_i + 2\alpha x_i)\beta + \beta^2 x_i^2\right]$$

So $\frac{dJ}{d\beta} = \sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[-2yx_i + 2\alpha x_i + 2x_i^2\beta\right] = 0$

So $\sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[-y_i x_i + \alpha x_i + x_i^2 \beta\right] = 0$

$$-\sum_{i=1}^{N} K_n(X_0, X_i)(X_i Y_i) + \sum_{i=1}^{N} K_n(X_0, X_i)\cdot \hat{\alpha} X_i + \sum_{i=1}^{N} K_n(X_0, X_i) X_i^2 \hat{\beta} = 0$$

set this as ①

$$\frac{dJ}{d\alpha} = \sum_{i=1}^{N} K_n(X_0, X_i)[-2Y + 2\alpha + 2\beta X_i] = 0$$

$$\sum_{i=1}^{N} K_n(X_0, X_i)[-2Y + 2\hat{\alpha} + 2\hat{\beta} X_i] = 0$$

$$-\sum_{i=1}^{N} K_n(X_0, X_i) Y_i + \sum_{i=1}^{N} K_n(X_0, X_i)\hat{\alpha} + \hat{\beta}\sum_{i=1}^{N} K_n(X_0, X_i) X_i = 0$$

$$\hat{\alpha} = \frac{-\hat{\beta}\sum_{i=1}^{N} K_n(X_0, X_i) X_i + \sum_{i=1}^{N} K_n(X_0, X_i) Y_i}{\sum_{i=1}^{N} K_n(X_0, X_i)} \quad - ②$$

input ② into ①

$$-\sum_{i=1}^{N} K_n(X_0, X_i)\cdot(X_i Y_i) + \sum_{i=1}^{N} K_n(X_0, X_i)\cdot \frac{-\hat{\beta}\sum_{i=1}^{N} K_n(X_0, X_i) X_i + \sum_{i=1}^{N} K_n(X_0, X_i)\cdot Y_i}{\sum_{i=1}^{N} K_n(X_0, X_i)} + \sum_{i=1}^{N} K_n(X_0, X_i)\cdot X_i^2 \hat{\beta} = 0$$

$$-\sum_{i=1}^{N} K_n(x_0, x_i)\cdot(x_i y_i) - \hat{\beta}\sum_{i=1}^{N} K_n(x_0, x_i)\, x_i + \sum_{i=1}^{N} K_n(x_0, x_i)\cdot y_i$$

$$+\sum_{i=1}^{N} K_n(x_0, x_i)\cdot x_i^2\,\hat{\beta} = 0$$

$$\left(\sum_{i=1}^{N} K_n(x_0, x_i)\, x_i^2 - \sum_{i=1}^{N} K_n(x_0, x_i)\, x_i\right)\hat{\beta} = -\sum_{i=1}^{N} K_n(x_0, x_i)\cdot y_i + \sum_{i=1}^{N} K_n(x_0, x_i)\cdot(x_i y_i)$$

$$\hat{\beta} = \frac{-\sum_{i=1}^{N} K_n(x_0, x_i)\cdot y_i + \sum_{i=1}^{N} K_n(x_0, x_i)\cdot(x_i y_i)}{\sum_{i=1}^{N} K_n(x_0, x_i)\, x_i^2 - \sum_{i=1}^{N} K_n(x_0, x_i)\cdot x_i}$$

In [3]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [165]:

```
iris=pd.read_csv("iris.csv")
```

In [166]:

```
diseases=pd.read_csv("us_contagious_diseases.csv")
```

Question 1 b

In [178]:

```
def kernel_function(x,x0,tuning_parameter):
    return (1/np.sqrt(2*np.pi)*tuning_parameter)*np.exp((-1/(2*tuning_parameter)
)*((x-x0)**2))

def estimating_alpha_beta(x0,x,y,tuning_parameter):

    beta=(-np.sum(kernel_function(x,x0,tuning_parameter)*y)+np.sum(kernel_functi
on(x,x0,tuning_parameter)*x*y))/((np.sum(kernel_function(x,x0,tuning_parameter)*
(x**2))-np.sum(kernel_function(x,x0,tuning_parameter)*x))) #analytical solution
of beta
    alpha=(-beta*np.sum(kernel_function(x,x0,tuning_parameter)*x)+np.sum(kernel_
function(x,x0,tuning_parameter)*y))/np.sum(kernel_function(x,x0,tuning_parameter
)) #analytical solution of alpha


    return alpha, beta


def local_linear_regression(x,y,tuning_parameter):
    Alphas=[]
    Betas=[]
    for i in range(len(x)):
        a,b = estimating_alpha_beta(x[i],x,y,tuning_parameter)
        Alphas.append(a)
        Betas.append(b)

    return Alphas, Betas
```

Question 1c

In [168]:

```
counts_of_measles=[]
for j in range(76):
    for i in range(18870):
        if diseases['disease'][i] =='Measles' and diseases['year'][i] == 1928+j:
            counts_of_measles.append(diseases['count'][i])
```

In [169]:

```
total_counts_by_year=[]


for i in range(76):
    Sum=0
    for j in range(51):
        Sum+=counts_of_measles[51*i+j]
    total_counts_by_year.append(Sum)
```

In [170]:

```
year=[]
for i in range(76):
    year.append(1928+i)
```

In [171]:

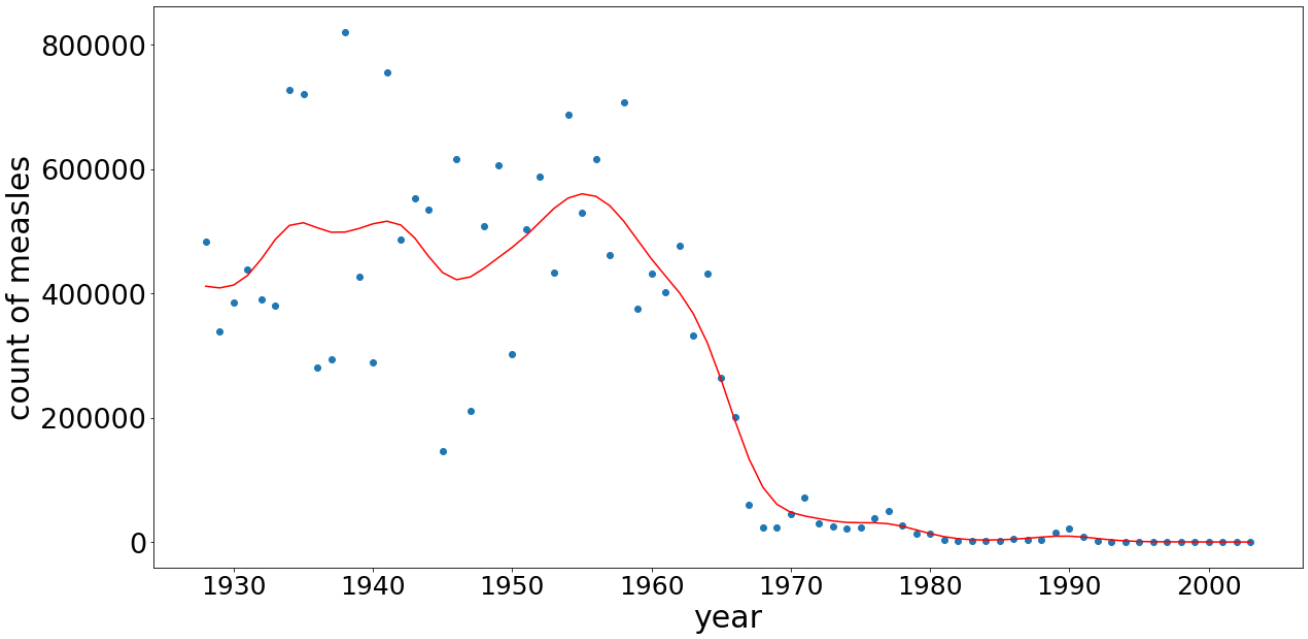
```
x = np.array(year)
y = np.array(total_counts_by_year)

alphas,betas=local_linear_regression(x,y,tuning_parameter=4)
```

In [172]:

```
Y_data = []
for i in range(len(alphas)):
    Y_data.append(alphas[i]+betas[i]*year[i])
```

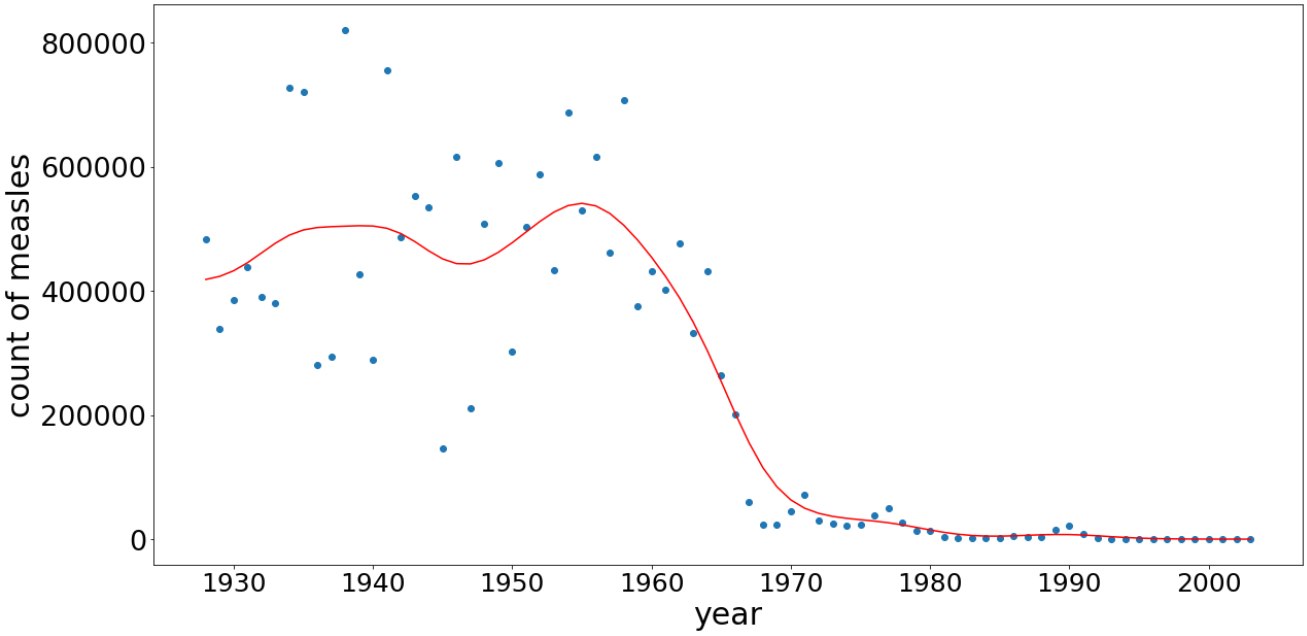
In [173]:

```
fig=plt.figure(figsize=(20,10))
plt.scatter(year,total_counts_by_year)
plt.xlabel('year', fontsize=30)
plt.ylabel('count of measles', fontsize=30)
plt.yticks(fontsize =27)
plt.xticks(fontsize =25)

plt.plot(year,Y_data,color='red')
plt.show()
```



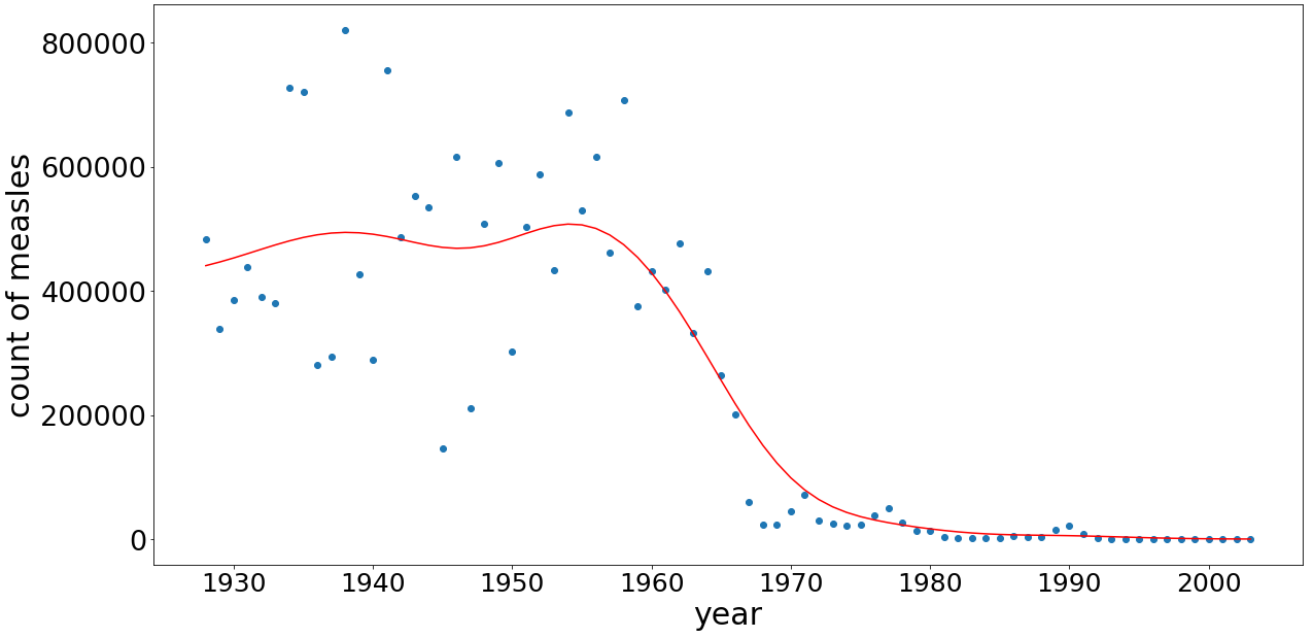
In [174]:

```
alphas,betas=local_linear_regression(x,y,tuning_parameter=8)
Y_data = []
for i in range(len(alphas)):
    Y_data.append(alphas[i]+betas[i]*year[i])
fig=plt.figure(figsize=(20,10))
plt.scatter(year,total_counts_by_year)
plt.xlabel('year', fontsize=30)
plt.ylabel('count of measles', fontsize=30)
plt.yticks(fontsize =27)
plt.xticks(fontsize =25)

plt.plot(year,Y_data,color='red')
plt.show()
```



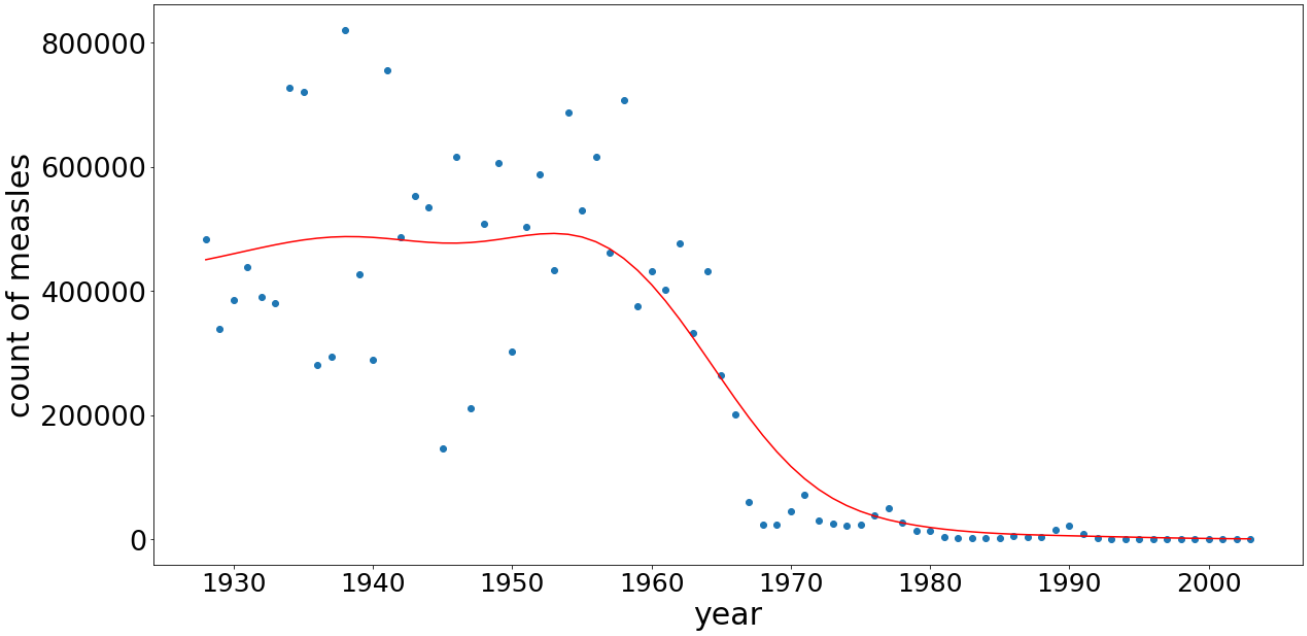
In [175]:

```
alphas,betas=local_linear_regression(x,y,tuning_parameter=20)
Y_data = []
for i in range(len(alphas)):
    Y_data.append(alphas[i]+betas[i]*year[i])
fig=plt.figure(figsize=(20,10))
plt.scatter(year,total_counts_by_year)
plt.xlabel('year', fontsize=30)
plt.ylabel('count of measles', fontsize=30)
plt.yticks(fontsize =27)
plt.xticks(fontsize =25)

plt.plot(year,Y_data,color='red')
plt.show()
```



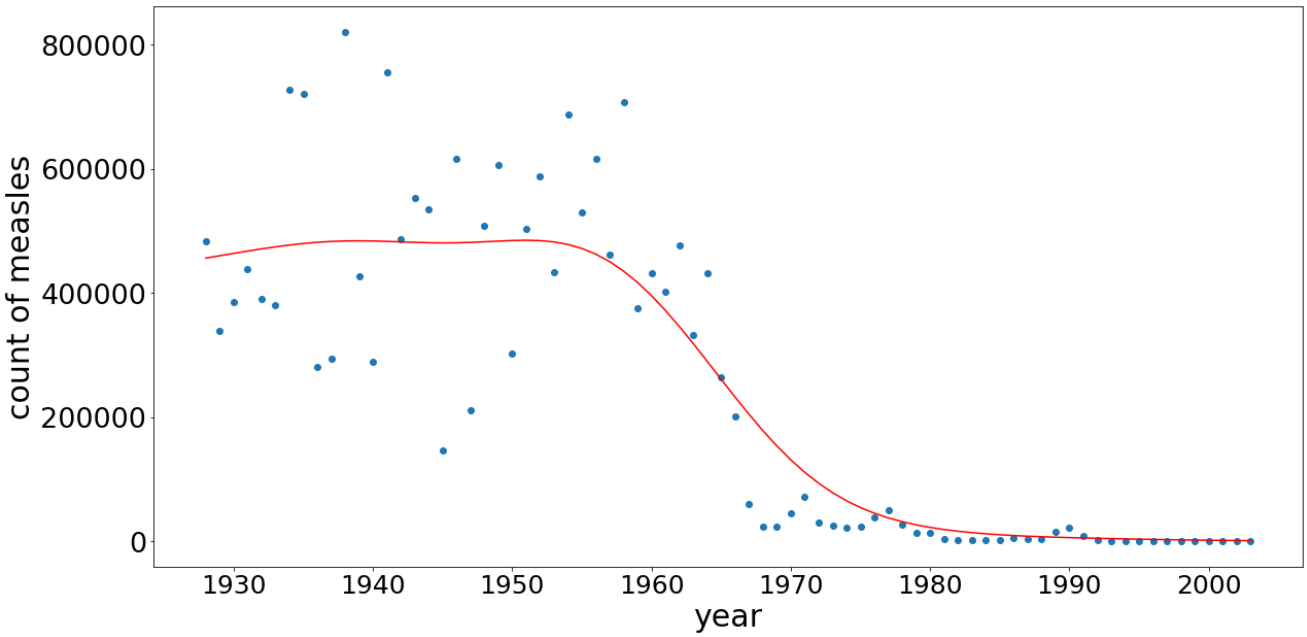
In [176]:

```
alphas,betas=local_linear_regression(x,y,tuning_parameter=30)
Y_data = []
for i in range(len(alphas)):
    Y_data.append(alphas[i]+betas[i]*year[i])
fig=plt.figure(figsize=(20,10))
plt.scatter(year,total_counts_by_year)
plt.xlabel('year', fontsize=30)
plt.ylabel('count of measles', fontsize=30)
plt.yticks(fontsize =27)
plt.xticks(fontsize =25)

plt.plot(year,Y_data,color='red')
plt.show()
```



In [179]:

```
alphas,betas=local_linear_regression(x,y,tuning_parameter=40)
Y_data = []
for i in range(len(alphas)):
    Y_data.append(alphas[i]+betas[i]*year[i])
fig=plt.figure(figsize=(20,10))
plt.scatter(year,total_counts_by_year)
plt.xlabel('year', fontsize=30)
plt.ylabel('count of measles', fontsize=30)
plt.yticks(fontsize =27)
plt.xticks(fontsize =25)

plt.plot(year,Y_data,color='red')
plt.show()
```



Note: I summed up all the counts for each year only because the numbers of observations at different years are the same; so this is a shortcut for this question. But in reality we have to be careful.

Role of tuning parameter: the larger the tuning parameter, the smoother the curve.

Question 1 d

I can conclude that effective policies were implemented around the year of 1967, because since then the count of measles has stayed low.

(Q2) a) Pseudocode:

"Batch" algorithm:

- Initialize $\theta^{(0)} \leftarrow 0$

Repeat until no changes in θ {

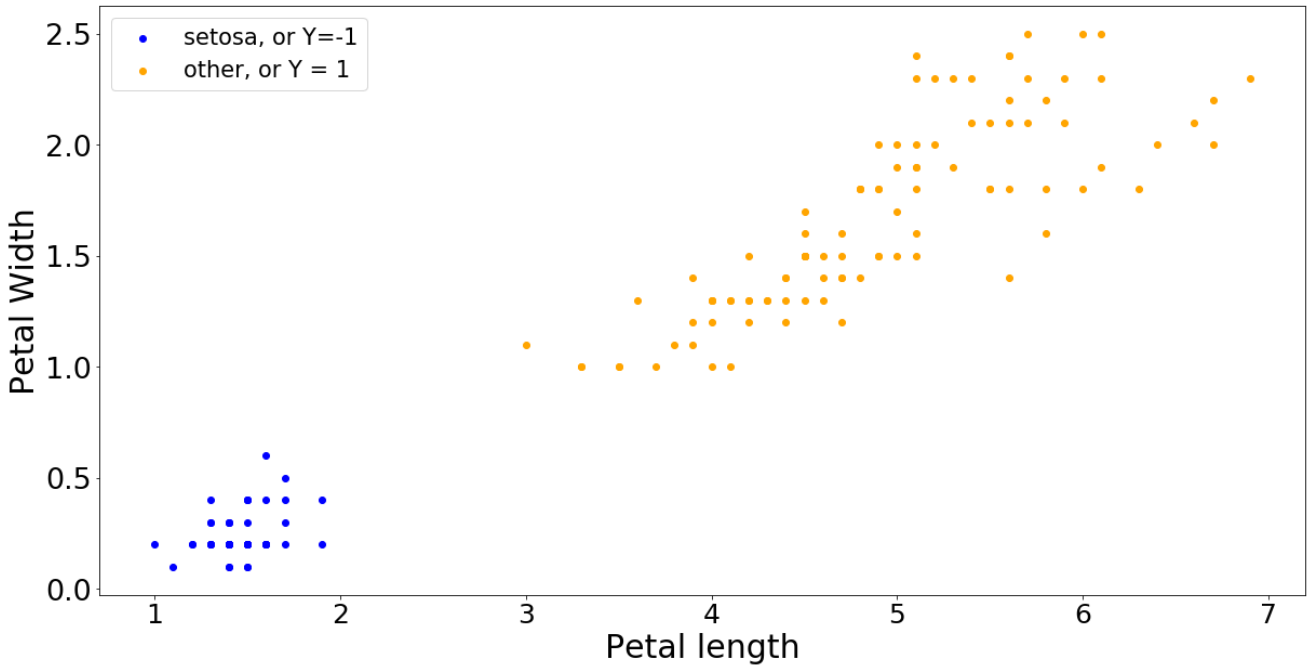
$$\theta^{(t+1)} \leftarrow \theta^{(t)} + \alpha \sum_{i=1}^{N} (y_i - \text{sign}(\theta^T x_i)) x_i$$

}

question 2b and c

In [11]:

```
fig=plt.figure(figsize=(20,10))
plt.scatter(iris[iris['Species']=='setosa']['Petal.Length'],iris[iris['Species']
=='setosa']['Petal.Width'],color='blue')
plt.scatter(iris[iris['Species']!='setosa']['Petal.Length'],iris[iris['Species']
!='setosa']['Petal.Width'],color='orange')
plt.legend(('setosa, or Y=-1', 'other, or Y = 1'), loc='upper left', fontsize=21
)
plt.xlabel('Petal length', fontsize=30)
plt.ylabel('Petal Width', fontsize=30)
plt.yticks(fontsize =27)
plt.xticks(fontsize =25)
plt.show()
```



In [181]:

```
def perceptron(x, y, initial_theta, alpha=0.01):

    x = np.array(x)
    y = np.array(y)

    old_theta = np.array([1,1,1])
    predictions = []
    all_thetas = []
    all_thetas.append(initial_theta)
    while np.array_equal(all_thetas[-1]-old_theta,np.array([0,0,0])) == False:
        old_theta = np.copy(all_thetas[-1])
        predictions = []
        theta_new = np.copy(all_thetas[-1])
        for i in range(len(x)):
            if np.dot(np.transpose(all_thetas[-1]),x[i]) >= 0:
                predictions.append(1)
            else:
                predictions.append(-1)
            theta_new = theta_new + alpha*(y[i] - predictions[i])*x[i]
        all_thetas.append(theta_new)


    return all_thetas
```

In [142]:

```
matrix_x=np.c_[np.ones(iris['Petal.Length'].shape[0]),iris['Petal.Length'],iris[
'Petal.Width']]
matrix_y=[]
for i in range(50):
    matrix_y.append(-1)
for i in range(100):
    matrix_y.append(1)
```

In [162]:

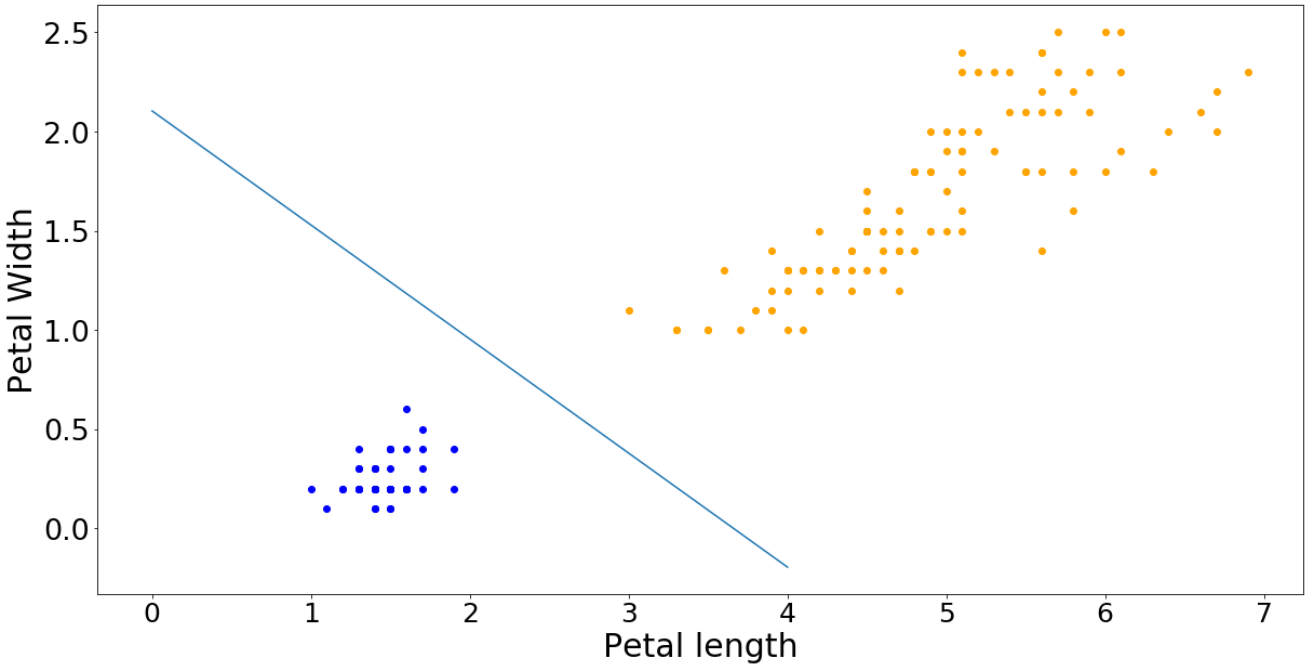
```
all_thetas1 = perceptron(matrix_x,matrix_y,[0,0,0])
print(all_thetas1[-1])
```

```
[-3.96   1.082  1.882]
```

So decision boundary: X2= -(-3.96)/1.882+(-1.082/1.882)X1 or X2 = (2.1041445)+(-0.5749)*X1

In [163]:

```
fig=plt.figure(figsize=(20,10))
plt.scatter(iris[iris['Species']=='setosa']['Petal.Length'],iris[iris['Species']
=='setosa']['Petal.Width'],color='blue')
plt.scatter(iris[iris['Species']!='setosa']['Petal.Length'],iris[iris['Species']
!='setosa']['Petal.Width'],color='orange')
plt.plot(np.arange(0,4,0.000001),(2.1041445)+(-0.5749)*np.arange(0,4,0.000001))
#decision boundary
plt.xlabel('Petal length', fontsize=30)
plt.ylabel('Petal Width', fontsize=30)
plt.yticks(fontsize =27)
plt.xticks(fontsize =25)
plt.show()
```



The perceptron has worked well in this case, because the linear decision boundary has clearly seperated the two classes.

3)a)

We can find the next split by maximizing impurity reduction

Gini index: $i(t) = P(1|t)(1-P(1|t)) + P(0|t)(1-P(0|t))$

Gini index for defaulted borrower:

$$i(t) = \frac{3}{10} \times \frac{7}{10} + \frac{7}{10} \times \frac{3}{10} = 2 \times \frac{3}{10} \times \frac{7}{10} = \frac{21}{50} = 0.42$$

Split 1

Gini index for Yes – homeowner:

$$i(t) = P(\text{Yes})\, i(\text{Yes}) + P(\text{No})\, i(\text{No})$$

$$= \frac{3}{10} \times 2 \times \frac{0}{3} \times \frac{3}{3} + \frac{7}{10} \times 2 \times \frac{4}{7} \times \frac{3}{7}$$

$$= \frac{12}{35} = 0.343$$

		Homeowner: Yes	Homeowner: No
Defaulted Borrower	Yes	0	3
	No	3	4

Gini index for married – marital status:

$$i(t) = \frac{4}{10} \times 2 \times \frac{0}{4} \times \frac{4}{4} + \frac{6}{10} \times 2 \times \frac{3}{6} \times \frac{3}{6}$$

$$= 0.3$$

		married	single, divorced
Defaulted borrower	Yes	0	3
	No	4	3

Gini index for single – marital status:

$$i(t) = \frac{4}{10} \times 2 \times \frac{2}{4} \times \frac{2}{4} + \frac{6}{10} \times 2 \times \frac{1}{6} \times \frac{5}{6}$$

$$= \frac{11}{30} = 0.3667$$

		single	married, divorced
Defaulted borrower	Yes	2	1
	No	2	5

Gini index for divorced – marital status:

$$i(t) = \frac{2}{10} \times 2 \times \frac{1}{2} \times \frac{1}{2} + \frac{8}{10} \times 2 \times \frac{2}{8} \times \frac{6}{8}$$

$$= 0.4$$

		divorced	single, married
Defaulted borrower	Yes	1	2
	No	1	6

Gini - index for income > 95k:

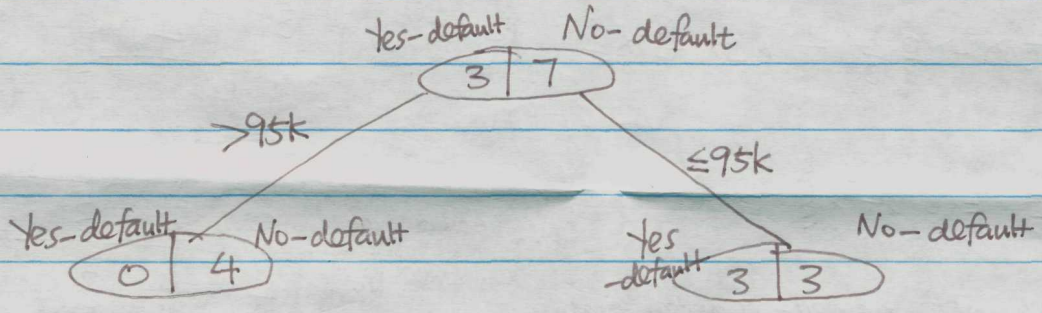
$i(t) = \frac{4}{10} \times 2 \times \frac{0}{4} \times \frac{4}{4} + \frac{6}{10} \times 2 \times \frac{3}{6} \times \frac{3}{6}$

$= 0.3$

		income >95k	income ≤95k
defaulted borrower	Yes	0	3
	No	4	3

To maximize impurity reduction, we choose the split that gives the smallest Gini index. Therefore, we choose the split to be income > 95k or ≤ 95k

So we now have:



Split 2

Gini index for single:

$i(t) = \frac{3}{6} \times 2 \times \frac{2}{3} \times \frac{1}{3} + \frac{3}{6} \times 2 \times \frac{1}{3} \times \frac{2}{3}$

$= \frac{4}{9} = 0.444$

		single	married, divorced
income ≤95k	Yes -default	2	1
	No -default	1	2

Gini index for married:

$i(t) = \frac{2}{6} \times 2 \times \frac{0}{2} \times \frac{2}{2} + \frac{4}{6} \times 2 \times \frac{1}{4} \times \frac{3}{4}$

$= \frac{1}{4} = 0.25$

		married	single, divorced
income ≤95k	Yes -default	0	3
	No -default	2	1

Gini index for divorced:

$$i(t) = \frac{1}{6} \times 2 \times \frac{0}{1} \times \frac{1}{1} + \frac{5}{6} \times 2 \times \frac{2}{5} \times \frac{3}{5}$$

$$= \frac{2}{5} = \boxed{0.4}$$

		divorced	single, married
income ≤95k	Yes default	1	2
	No default	0	3

All homeowners have income >95k so there's nothing to split.

Gini index for income <85k:

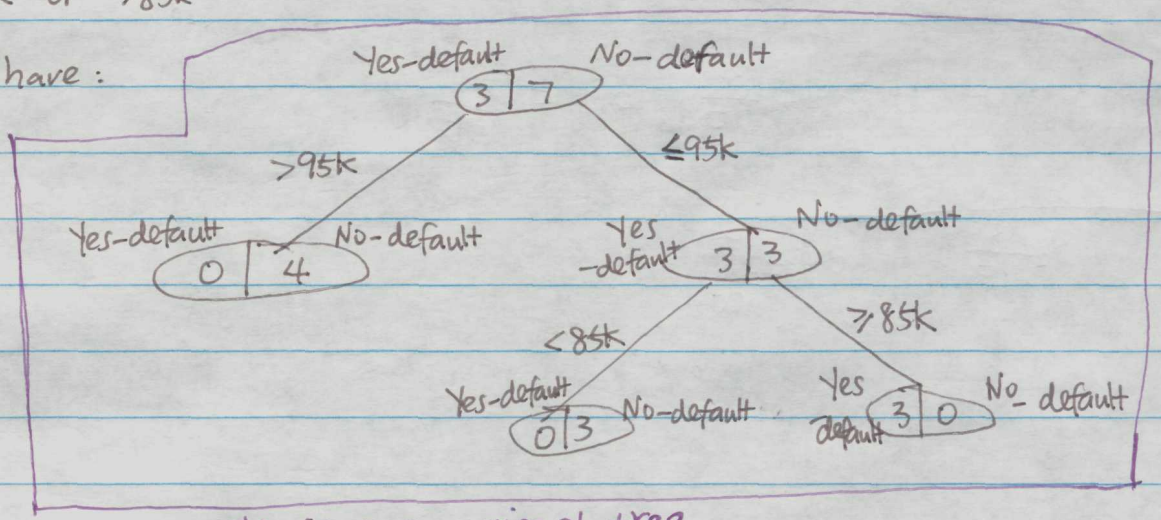
$$i(t) = \frac{3}{6} \times 2 \times \frac{0}{3} \times \frac{3}{3} + \frac{3}{6} \times 2 \times \frac{0}{3} \times \frac{3}{3}$$

$$= \boxed{0}$$

		income	
		<85k	≥85k
income ≤95k	Yes default	0	3
	No default	3	0

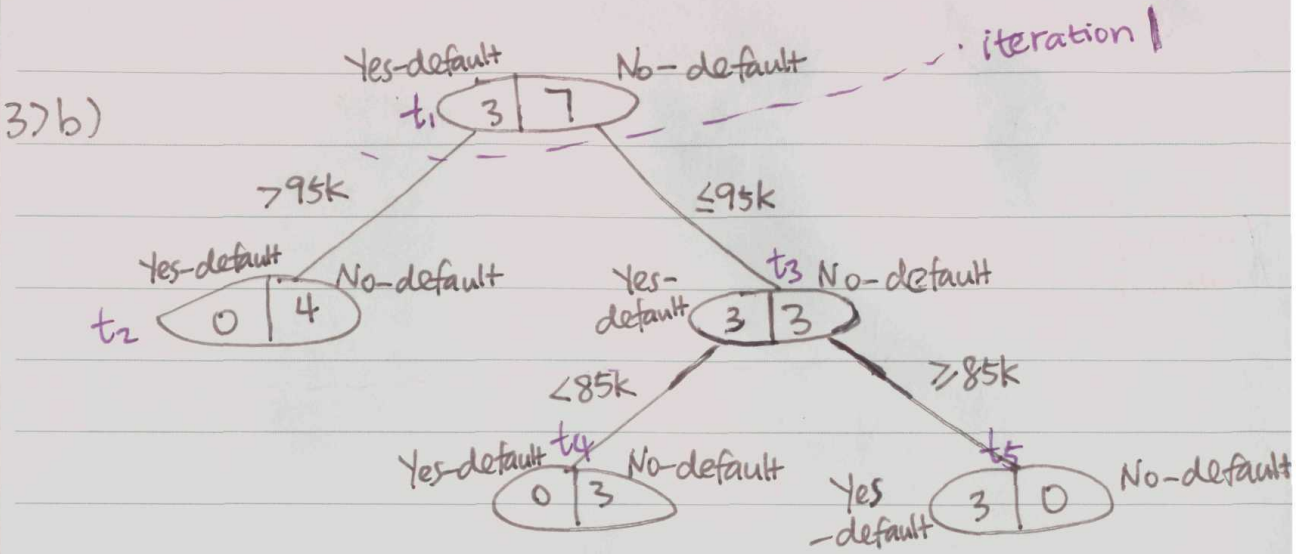
To maximize impurity reduction, we choose the split that gives the smallest Gini index. Therefore, we choose the split to be income <85k or ≥85k

So now we have:



This is our maximal tree.

3) b)



Iteration 1:

Calculate α for each non-terminal node

$$g(t_3) = \frac{\frac{6}{10}\cdot\frac{3}{6} - \left(\frac{6}{10}\cdot\frac{3}{6}\cdot\frac{0}{6} + \frac{6}{10}\cdot\frac{3}{6}\cdot\frac{0}{6}\right)}{2-1}$$

(at intermediate node t_3; at leaf t_4, t_5)

$$= \frac{3}{10} = 0.3$$

$$g(t_1) = \frac{1\times\frac{3}{10} - \left(1\times\frac{4}{10}\times\frac{0}{10} + 1\times\frac{6}{10}\times\frac{0}{6}\times\frac{3}{6} + 1\times\frac{6}{10}\times\frac{0}{6}\times\frac{3}{6}\right)}{3-1}$$

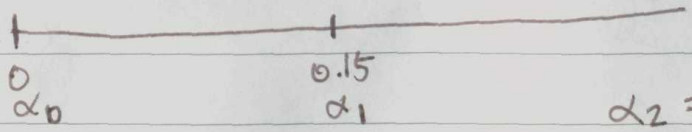
(root node)

$$= 0.15$$

So, the node with the smallest α is 0.15.

So, prune t_1.

Output: $\alpha = [0, 0.15]$



set $\beta_1 = 0.1$

$\beta_2 = 0.2$

3)c)

Splitting data in 2:

	Homeowner	Marital status	Annual income	Defaulted borrower
1	Yes	Single	125K	No
2	No	married	100K	No
3	No	single	70K	No
4	Yes	married	120K	No
5	No	divorced	95K	Yes
6	No	married	60K	No
7	Yes	divorced	220K	No
8	No	single	85K	Yes
9	No	married	75K	No
10	No	single	90K	Yes

For the first 5 rows:

Split 1 (training set)

Gini index for Yes-homeowner:

$$i(t) = \frac{2}{5} \times 2 \times \frac{0}{2} \times \frac{2}{2} + \frac{3}{5} \times 2 \times \frac{1}{3} \times \frac{2}{3}$$

$$= 0.2667$$

Defaulted borrower \ homeowner	Yes	No
Yes	0	1
No	2	2

Gini index for income > 95k

$$i(t) = \frac{3}{5} \times 2 \times \frac{0}{3} \times \frac{3}{3} + \frac{2}{5} \times 2 \times \frac{1}{2} \times \frac{1}{2}$$

$$= 0.2$$

defaulted borrower \ income	>95K	≤95K
Yes	0	1
No	3	1

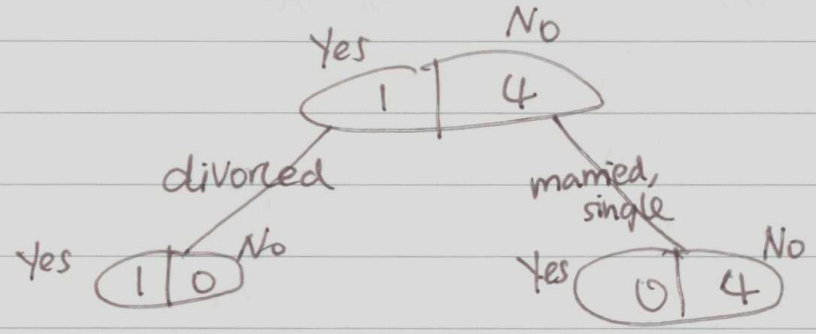
Gini index for divorced:

$i(t) = \frac{1}{5} \times 2 \times 0 + \frac{4}{5} \times 2 \times 0$

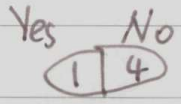
$= 0$

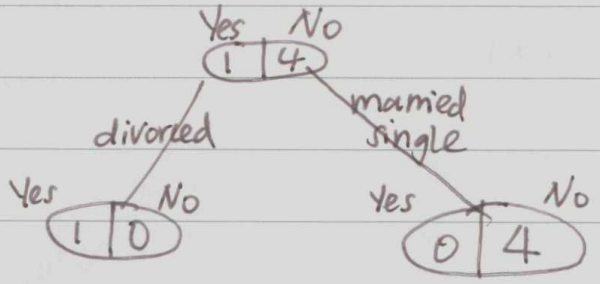
	divorced	married, single
Yes	1	0
No	0	4

So we have found our best maximal tree



In this case, there are 2 subtrees:





$R(t) + \beta_1 |T|$

$= 1 \times \frac{1}{5} + 0.1 \times 1$

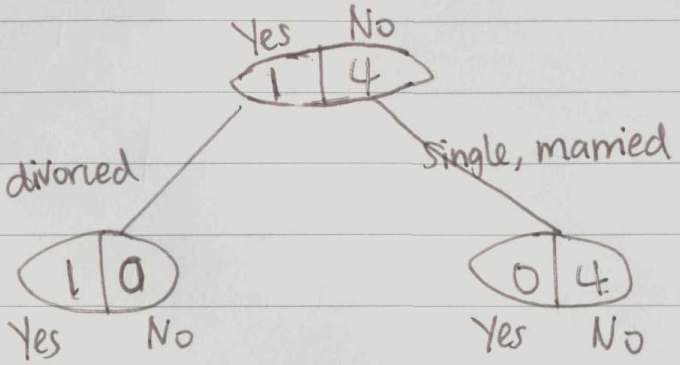
$= 0.2$

$R(t) + \beta_1 |T|$

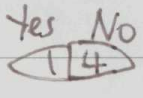
$= 0 + 0 + 0.1 \times 2 = 0.2$

so, ↓ this subtree minimizes $R(t) + \beta_1 |T|$

applying this subtree to validation subset:



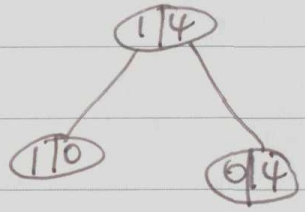
Number of misclassified observations: 2



$R(t) + \beta_2 |T|$

$= 1 \times \frac{1}{5} + 0.2 \times 1$

$= 0.4$

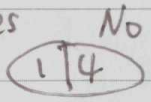


$R(t) + \beta_2 |T|$

$= 0 + 0 + 0.2 \times 2 = 0.4$

They give same results. So choose either.

Applying this to validation set:



Number of misclassified observations: 1

Setting last 5 rows as training set:

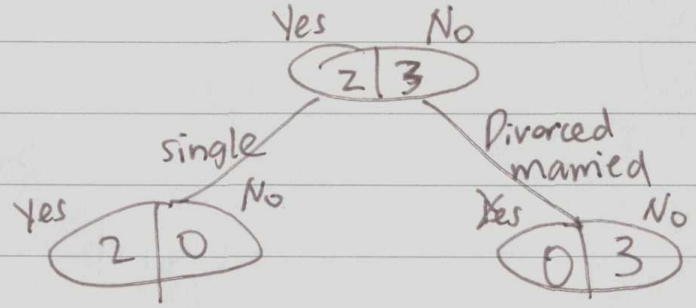
Split 1:

Gini index for single:

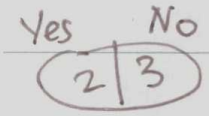
$i(t) = \frac{2}{5} \times 2 \times 0 + \frac{3}{5} \times 2 \times 0 = 0$

	Single	divorced, married
Yes	2	0
No	0	3

So we have found our best maximal tree:



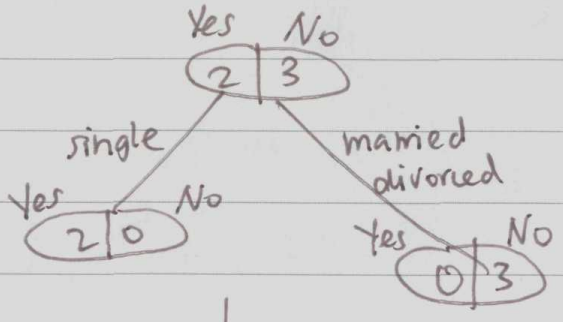
In this case, we have 2 subtrees:



$R(t) + \beta_1 |T|$

$= 1 \times \frac{2}{5} + 0.1 \times 1$

$= 0.5$

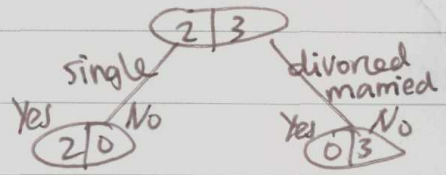


$R(t) + \beta_1 |T|$

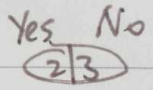
$= 0 + 0 + 0.1 \times 2 = 0.2$

So this subtree minimizes $R(t) + \beta_1 |T|$

applying this to validation set



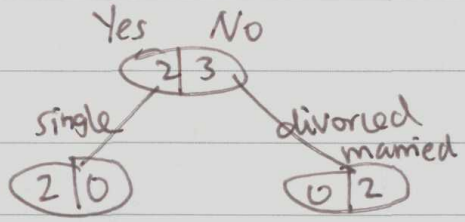
number of misclassified observations: 3



$R(t) + \beta_2|T|$

$= 1 \times \frac{2}{5} + 0.2 \times 1$

$= 0.6$



$R(t) + \beta_2|T|$

$= 0 + 0 + 0.2 \times 2$

$= 0.4$

so this subtree minimizes $R(t) + \beta_2|T|$

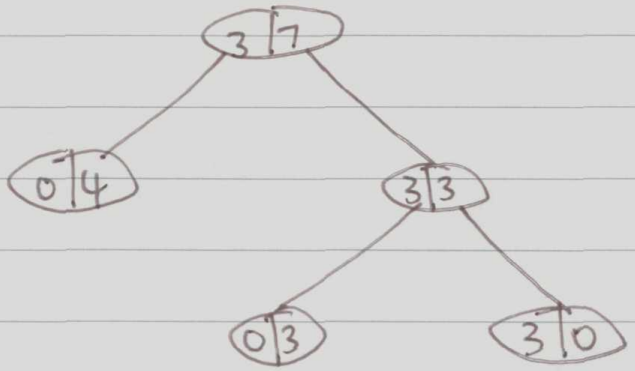
So, as shown before, number of missclassified observations = 3.

For β_1 : total number of misclassified observations
$= 2+3=5$

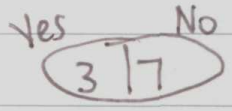
β_2 : total number of misclassified observations $= 1+3=4$

So $\beta_{optimal}$ is $\beta_2 = 0.2$.

Original full tree:



1st sub tree:

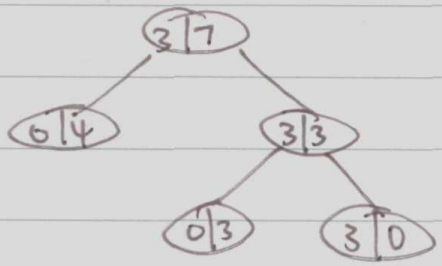


$R(T) + \beta_2|T|$

$= \frac{3}{10} + 0.2 \times 1$

$= 0.5$

2nd sub tree:

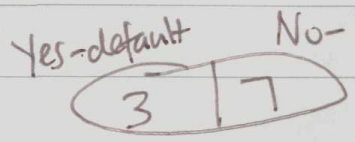


$R(T) + \beta_2|T|$

$= 0+0+0+0.2\times3 = 0.6$

So, the subtree that minimizes $R(T) + \beta_{optimal}|T|$

is just



4) a) The utility of kernel trick.

When we create classifiers with non-linear decision boundaries, the "traditional" approach would be to transform feature space to higher dimensions, aka basis expansion. This separates the transformed features by a maximum-margin hyperplane. However, as the number of basis functions increases, it could lead to overfitting. So, instead, there is a "novel" solution: in the optimization step, the feature vectors only participate in the objective function via a dot-product; So we can work with the dot product directly, without the basis expansion. This is the utility of kernel trick.

b) $k(x,z) = (xz+1)^2 = x^2z^2 + 2xz + 1$

So, there exists a $\phi(x)$ such that

$$\langle \phi(x), \phi(z) \rangle = x^2z^2 + 2xz + 1.$$

In this case,

$$\phi(x) = \begin{bmatrix} 1 \\ \sqrt{2}x \\ x^2 \end{bmatrix} \quad \text{or,} \quad \phi(z) = \begin{bmatrix} 1 \\ \sqrt{2}z \\ z^2 \end{bmatrix}$$

So, this is a valid kernel

c) $k(x,z) = (xz-1)^3$

$= (xz-1)(xz-1)(xz-1)$

$= (x^2z^2 - 2xz + 1)(xz-1)$

$= x^3z^3 - x^2z^2 - 2x^2z^2 + 2xz + xz - 1$

$= x^3z^3 - 3x^2z^2 + 3xz - 1$

there does NOT exist a $\phi(x)$ such that

$\langle \phi(x), \phi(z) \rangle = x^3z^3 - 3x^2z^2 + 3xz - 1$

So, this is NOT a valid kernel