SBU CSE-512

Name: Venkata Subba Narasa Bharath, Meadam

SBU ID: 112672986

Q1) Baye's rule

a)

i) Given :-

Event A → it is raining

Event B → carrying an umbrella

$P(A) = \frac{1}{2}$

$P(\neg A) = 1 - \frac{1}{2} = \frac{1}{2}$ (Not raining)

$P(B|A) = \frac{3}{4}$ (75%)

$P(B|\neg A) = \frac{1}{20}$ (5%)

$$P(B) = P(B|A)\,P(A) + P(B|\neg A)\,P(\neg A)$$

Verbally:- A person carries an umbrella, if it rains (or) when it does not rain.

$$P(B) \Rightarrow \frac{3}{4} \times \frac{1}{2} + \frac{1}{20} \times \frac{1}{2}$$

$$\Rightarrow \frac{3}{8} + \frac{1}{40} \Rightarrow \frac{16}{40} = \frac{4}{10}$$

To find:- $P(A|B)$

Applying Baye's Rule:-

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} = \frac{\frac{3}{4} \times \frac{1}{2}}{\frac{16}{40}} = \frac{\frac{3}{8}}{\frac{16}{40}}$$

$$= \frac{15}{16}$$

Probability that it is raining given that, I am using an umbrella is $\frac{15}{16}$.

ii) $P(B) = (0.75)^8 (0.25)^2 (0.5) + (0.05)^8 (0.95)^2 (0.5)$

(as all events are INDEPENDENT)

Verbally:- A person carries an umbrella, if it rains (or) when it does not rain.

$$P(A|B) = \frac{(0.75)^8 (0.25)^2 (0.5)}{(0.75)^8 (0.25)^2 (0.5) + (0.05)^8 (0.95)^2 (0.5)}$$

$$= 0.99$$

Probability that it is raining today, given my observation is 0.99.

Q1)

(b) Given:-

Current Positive COVID tests = 9.1 %.

$P(PPV \mid IgG/IgM\ test) = 82.5\% = 0.825$

$P(NPV \mid IgG/IgM\ test) = 99.9\% = 0.999$

i) $P(\text{Person has COVID} \mid \text{Positive Result})$

using Bayes' rule,

$$\Rightarrow \frac{P(\text{Positive} \mid \text{Person has covid}) * P(\text{covid +ve})}{\left(P(\text{Positive} \mid \text{Person has covid}) * P(\text{covid positive}) + P(\text{Positive Result} \mid \text{Person has no covid}) * P(\text{covid negative}) \right)}$$

$$= \frac{0.825 \times 0.091}{(0.825 \times 0.091) + ((1-0.999) * (1-0.091))}$$

$$= \frac{0.825 \times 0.091}{(0.825 \times 0.091) + (0.001 * 0.909)}$$

$= 0.98$

$\therefore Pr(\text{Person has covid} \mid \text{positive Result}) = 0.98$

ii) $Pr(\text{no covid} \mid \text{negative test})$

Using Bayes Rule:

$$\Rightarrow \frac{Pr(\text{Negative Result} \mid \text{no covid}) * Pr(\text{no covid})}{Pr(\text{negative Result} \mid \text{no covid}) * Pr(\text{no-covid}) + Pr(\text{negative Result} \mid \text{covid}) * Pr(\text{covid})}$$

$$\Rightarrow \frac{0.999 * 0.909}{(0.999 * 0.909) + [(1-0.825) * (0.091)}$$

$$\Rightarrow \frac{0.999 * 0.909}{(0.999 \times 0.909) * (0.175 * 0.091)} = 0.98$$

$\therefore Pr(\text{Person has no covid} \mid \text{negative Test}) = 0.98$

Q2) Given loss function:-

$$L(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \log\left(\sigma(y_i x_i^T \theta)\right)$$

$$\sigma(s) = \frac{1}{1+e^{-s}}$$

Sol:-

let, $\alpha_i = \sigma(y_i x_i^T \theta)$

$$\log \alpha = \log\left(\sigma(y_i x_i^T \theta)\right)$$

$$\frac{\partial}{\partial \theta_j}(\log \alpha) = \frac{1}{\alpha} \frac{\partial}{\partial \theta_j}(\alpha)$$

$$\frac{\partial}{\partial \theta_j}(\alpha) = \frac{\partial}{\partial \theta_j}\left[\sigma(y_i x_i^T \theta)\right]$$

$$= \frac{\partial}{\partial \theta_j}\left(\frac{1}{1+e^{-y_i x_i^T \theta}}\right)$$

$$= \frac{-e^{-y_i x_i^T \theta}(-y_i x_i^T)}{\left(1+e^{-y_i x_i^T \theta}\right)^2}$$

$$\Rightarrow \frac{e^{-y_i x_i^T \theta} \left(y_i x_i^T\right)}{\left(1 + e^{-y_i x_i^T \theta}\right)^2}$$

$$\frac{\partial}{\partial \theta_j} (\log L) = \frac{1}{\sigma(y_i x_i^T \theta)} \frac{e^{-y_i x_i^T \theta} \left(y_i x_i^T\right)}{\left(1 + e^{-y_i x_i^T \theta}\right)^2}$$

$$= \frac{\cancel{\left(1 + e^{-y_i x_i^T \theta}\right)} e^{-y_i x_i^T \theta} \left(y_i x_i^T\right)}{\left(1 + e^{-y_i x_i^T \theta}\right)^{\cancel{2}}}$$

$$= y_i x_i^T \left(1 - \frac{1}{e^{-y_i x_i^T \theta}}\right)$$

$$\frac{\partial L}{\partial \theta} = -\frac{1}{m} \sum_{i=1}^{m} \left(y_i x_i^T \left(1 - \sigma(y_i x_i^T \theta)\right)\right)$$

$$\frac{\partial^2 L}{\partial \theta^2} = +\frac{1}{m} \sum_{i=1}^{m} y_i x_i^T \frac{\partial}{\partial \theta} \sigma(y_i x_i^T \theta)$$

$$\frac{\partial}{\partial\theta}\left(\sigma(y_i x_i^T\theta)\right) = \frac{\partial}{\partial\theta}\left(\frac{1}{1+e^{-y_i x_i^T\theta}}\right)$$

$$= \frac{+\,y_i x_i^T\, e^{-y_i x_i^T\theta}}{\left(1+e^{-y_i x_i^T\theta}\right)^2} = y_i x_i^T\left(\frac{1}{1+e^{-y_i x_i^T\theta}} * \left(1-\frac{1}{1+e^{-y_i x_i^T\theta}}\right)\right)$$

$$= y_i x_i^T\left[\sigma(y_i x_i^T\theta)\,\left(1-\sigma(y_i x_i^T\theta)\right]\right.$$

$$\frac{\partial^2 L}{\partial\theta^2} = \frac{-1}{m}\sum_{i=1}^{m}\left((y_i x_i^T)^2\left[\sigma(y_i x_i^T\theta)\,\left(1-\sigma(y_i x_i^T\theta)\right]\right]\right)$$

$$\therefore \frac{dL}{d\theta} = \frac{-1}{m}\sum_{i=1}^{m}\left(y_i x_i^T\left(1-\sigma(y_i x_i^T\theta)\right)\right)$$

$$\frac{\partial^2 L}{\partial\theta^2} = \frac{1}{m}\sum_{i=1}^{m}\left((y_i x_i^T)^2\left[\sigma(y_i x_i^T\theta) * (1-\sigma(y_i x_i^T\theta)\right]\right)$$

Q3)

Conditions for norm:-

Norms are any function $R^d \to R$ that satisfy,

1) $\|x\| \geq 0 \quad \forall x$

2) $\|\beta x\| = |\beta| \cdot |x|$

3) $\|x+y\| \leq |x| + |y|$

4) $\|x\| = 0 \iff x = 0$

i) $f(x) = \sum_k x[k]$

i) $\|x\| \geq 0$

$$\|x\| = \sum_{k=1}^{d} x[k] = x[1] + x[2] \cdots x[k]$$

Assume, $x = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$

$\|x\| = 1 + -2 = -1$

$\|x\| < 0$

$\therefore \sum_k x[k]$ is not a norm

To disprove that a function is not a norm, we can give a counter-example

b) $f(x) = \left(\sum_{k=1}^{d} \sqrt{|x[k]|} \right)^2$

i) $\|x\| = \left(\sum_{k=1}^{d} \sqrt{|x[k]|} \right)^2$

$= \left(\sqrt{x_1} + \sqrt{x_2} + \sqrt{x_3} \cdots \sqrt{x_d} \right)^2$

$\forall i,\ i \in [1, d] \Rightarrow \sqrt{x_i} \geq 0$ (as $\sqrt{a} \geq 0$)

$\therefore \|x\| \geq 0$

ii) $\|x\| = 0$ iff $x_i = 0\ \forall i,\ i \in [1, d]$

$\therefore \|x\| = 0$ iff $x = 0$

iii) $\|\beta x\| = \left(\sum_{k=1}^{d} \sqrt{|\beta||x[k]|} \right)^2$

$= \left(\sqrt{|\beta x_1|} + \sqrt{|\beta x_2|} \cdots \sqrt{|\beta x_d|} \right)^2$

$= \left(\sqrt{\beta} \left(\sqrt{x_1} + \sqrt{x_2} \cdots \sqrt{x_d} \right) \right)^2$

$$= (\sqrt{\beta})^2 \left(\sqrt{x_1} + \sqrt{x_2} \cdots \sqrt{x_d}\right)^2$$

$$= \beta \left(\sum_{k=1}^{d} \sqrt{|x[k]|} \right)^2$$

$$= \|\beta\| \quad \|x\|$$

$$\therefore \|\beta x\| = |\beta| \quad |x|$$

iv) let, $x = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}$ $y = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$

$$x + y = \begin{bmatrix} 2 \\ 4 \\ 3 \end{bmatrix}$$

$$\|x\| = \left(\sqrt{1} + \sqrt{2} + \sqrt{0}\right)^2 = 5.82$$

$$\|y\| = \left(\sqrt{1} + \sqrt{2} + \sqrt{3}\right)^2 = 17.19$$

$$\|x_1 + y_1\| = \left(\sqrt{2} + \sqrt{4} + \sqrt{3}\right)^2$$

$$= 26.484$$

$$23.01 < 26.484$$

$$|\|x+y\|| > (|\|x\|| + |\|y\||)$$

↑ This is a contradiction, for the properties of norm.

$$\therefore f(x) = \left| \sum_{k=1}^{d} \sqrt{|x[k]|} \right|^2 \text{ is not a norm.}$$

c) $$f(x) = \sqrt{\sum_{k=1}^{d} \frac{|x[k]|^2}{k}}$$

$$|\|x\|| = \sqrt{\frac{x[1]^2}{1} + \frac{x[2]^2}{2} \cdots \cdots \frac{x[d]^2}{d}}$$

$\forall i \quad x[i]^2 \geq 0$

$$\therefore |\|x\|| \geq 0$$

$x[i]^2 = 0$ iff $x[i] = 0$

$$\therefore |\|x\|| = 0 \quad \text{iff} \quad x = 0$$

$$\|\beta x\| = \sqrt{\sum_{k=1}^{d} \frac{(\beta x[k])^2}{k}}$$

$$= \sqrt{\frac{(\beta x[1])^2}{1} + \frac{(\beta x[2])^2}{2} \cdots + \frac{(\beta x[d])^2}{d}}$$

$$= \sqrt{\beta^2\left(\frac{x[1]^2}{1}\right) + \beta^2\left(\frac{x[2]^2}{2}\right) \cdots + \beta^2\left(\frac{x[d]^2}{d}\right)}$$

$$= \sqrt{\beta^2 \left(\frac{x[1]^2}{1} + \frac{x[2]^2}{2} \cdots \frac{x[d]^2}{d}\right.}$$

$$= |\beta| \, \|x\|$$

$$\therefore \|\beta x\| = |\beta| \, \|x\|$$

$$\|x+y\| = \sqrt{\sum_{k=1}^{d} \frac{|(x[k] + y[k])|^2}{2}}$$

$$= \sqrt{\frac{|x_1+y_1|^2}{1} + \frac{|x_2+y_2|^2}{2} \cdots \frac{|x_d+y_d|^2}{d}}$$

$$|x| = \sqrt{\frac{|x_1|^2}{1} + \frac{|x_2|^2}{2} \cdots + \frac{|x_d|^2}{d}}$$

$$|y| = \sqrt{\frac{|y_1|^2}{1} + \frac{|y_2|^2}{2} \cdots + \frac{|y_d|^2}{d}}$$

$$|x_1|^2 + |y_1|^2 = \frac{|x_1|^2}{1} + \frac{|x_2|^2}{2} \cdots + \frac{|x_d|^2}{d}$$

$$\cdots + \frac{|y_1|^2}{1} + \frac{|y_2|^2}{2} \cdots + \frac{|y_d|^2}{d}$$

$$= \frac{|x_1|^2+|y_1|^2}{1} + \frac{|x_2|^2+|y_2|^2}{2} \cdots + \frac{|x_d|^2+|y_d|^2}{d}$$

$$\leq \frac{|x_1+y_1|^2}{1} + \frac{|x_2+y_2|^2}{2} \cdots + \frac{|x_d+y_d|^2}{d}$$

By, using $|x+y| \leq |x| + |y|$

$\therefore |x+y| \leq |x| + |y|$

As it satisfies all the properties of a norm, $f(x) = \sqrt{\sum_{k=1}^{d} \frac{x[k]^2}{k}}$ is a norm

(4) b) minimize $\frac{1}{2} \|X\theta - y\|_2^2$
$\theta \in R^p$

should be equivalent to

minimize $\frac{1}{2} \sum_{i=1}^{m} (\theta_1 + \theta_2 w_i + \theta_3 w_i^2 \ldots \theta_p w_i^{p-1})$

we know, y is a $(m \times 1)$ vector

θ is a $(p \times 1)$ vector.

$\therefore X = m \times p$ matrix

$$X = \begin{bmatrix} 1 & w_1 & w_1^2 & w_1^3 & \cdots & w_1^{p-1} \\ 1 & w_2 & w_2^2 & w_2^3 & \cdots & w_2^{p-1} \\ \vdots & & & & & \\ 1 & w_m & w_m^2 & w_m^3 & \cdots & w_m^{p-1} \end{bmatrix}_{(m \times p)}$$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}_{m \times 1}$$

(1) Minimize $\frac{1}{2}\|X\theta - y\|^2$
$\theta \in \mathbb{R}^p$

We know that, $A^2 = A^T A$ for any matrix.

$$\|X\theta - y\|^2 = (X\theta - y)^T (X\theta - y)$$

$$J = (X\theta)^T - y)^T (X\theta - y)$$

↑
(Cost-function)

$$= (\theta^T x^T - y^T)(x\theta - y)$$

$$= \theta^T x^T x\theta - y\theta^T x^T - y^T x\theta + y^T y$$

For minima,

$$\frac{\partial J}{\partial \theta} = 0$$

$$\frac{\partial}{\partial \theta}(\theta^T x^T x\theta - y\theta^T x^T - y^T x\theta + y^T y) = 0$$

$$\Rightarrow \frac{\partial J}{\partial \theta} = 2x^T x\theta - 2x^T y$$

$$\frac{\partial J}{\partial \theta} = 0$$

$$2x^T x\theta - 2x^T y = 0$$

$$2x^T x\theta = 2x^T y$$

$$X^T X \theta = X^T y$$

$$A \quad \theta^* \quad = B$$

$$A = X^T X \quad , \quad B = X^T y$$

d) $$\min_{\theta} \left(\frac{1}{2} \| X\theta - y \|^2 + \frac{\alpha}{2} \| \theta \|^2 \right)$$

$$\min_{\theta} \left[(X\theta - Y)^T (X\theta - Y) + \alpha \, \theta^T \theta \right]$$

$$\frac{\partial}{\partial \theta} \left[(X\theta - Y)^T (X\theta - Y) \right] = 2X^T X\theta - 2X^T y$$

$$\frac{\partial}{\partial \theta} (\alpha \theta^T \theta) = 2\alpha \theta$$

$$2X^TX\theta - 2X^Ty + 2\alpha\theta = 0$$

$$X^TX\theta + \alpha\theta = X^Ty$$

$$(X^TX + \alpha I)\theta = X^Ty$$

$$\therefore A = X^TX + \alpha I$$

$$B = X^Ty$$

e) condition-number = 5

Largest-eigen value = 1

smallest eigen value = $\frac{1}{5} = 0.2$

as, condition-number = $\frac{\alpha_i}{\alpha_j}$

α_i = largest eigen value

α_j = smallest eigen value

eigen values of $A + \alpha I$ would be $(\lambda + \alpha)$ where λ are the eigen values of A.

Largest eigen-value of $A+\alpha I = 1 + \alpha$

smallest eigen value of $A+\alpha I = 0.2 + \alpha$

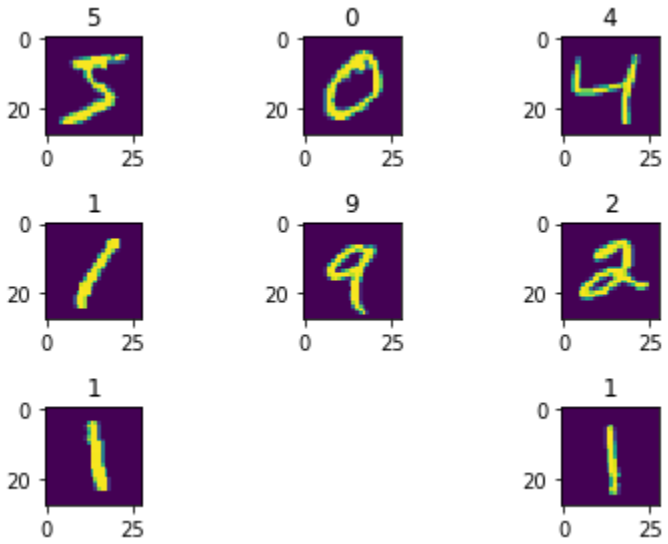
condition-number of $A + \alpha I \Rightarrow \dfrac{1+\alpha}{0.2+\alpha}$

for any $\alpha > 0$, condition-number is < 5.

$\therefore$ condition-number $= \dfrac{1+\alpha}{0.2+\alpha} < 5 \quad (\alpha > 0$

In [1]:
```
import scipy.io as sio
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:
```
data = sio.loadmat('mnist.mat')
for k in range(9):
    plt.subplot(3,3,k+1)
    plt.imshow(np.reshape(data['trainX'][k,:],(28,28)))
    plt.title(data['trainY'][0,k])
    plt.tight_layout()
```



In [3]:
```
X = data['trainX']
y = data['trainY']
xTest = data['testX']
yTest = data['testY']

idx = np.logical_or(np.equal(y,4) , np.equal(y,9))
idxTest = np.logical_or(np.equal(yTest,4) , np.equal(yTest,9))
X = X[idx[0, :],  :]
y = y[idx]
xTest = xTest[idxTest[0, :],  :]
yTest = yTest[idxTest]

y[np.equal(y,4)] = 0
y[np.equal(y,9)] = 1

yTest[np.equal(yTest,4)] = 0
yTest[np.equal(yTest,9)] = 1

print("XTrain Shape modified: " +str(np.shape(X)))
print("yTrain Shape modified: " +str(np.shape(y)))
print("XTest Shape modified: " +str(np.shape(xTest)))
print("yTest Shape modified: " +str(np.shape(yTest)))
```

```
XTrain Shape modified: (11791, 784)
yTrain Shape modified: (11791,)
XTest Shape modified: (1991, 784)
yTest Shape modified: (1991,)
```

In [16]:
```
## Helper Functions

def sigmoid(s):
    return 1/(1 + np.exp(-s))

def normalizeData(X):
    # finding mean
    mean=np.mean(X,axis=0)
    # divide with (max -min)
    # maxValue = 255, minValue = 0
    X_norm = (X - mean)/255
    return X_norm

def costFunction(weights,X, y):
    m=len(y)
    z = np.dot(X, weights)
    loss_for_1 = y * np.log(sigmoid(z))
    loss_for_0 = (1 - y) * np.log(1 - sigmoid(z))
    grad = 1/m * np.dot(X.transpose(),(sigmoid(z) - y))
    return -sum(loss_for_1 + loss_for_0) / m, grad

def gradientDescent(X,y,theta,alpha,num_iters):
    m=len(y)
    lossFunctionHistory =[]

    for i in range(num_iters):
        loss, gradient = costFunction(theta,X,y)
        theta = theta - (alpha * gradient)
        lossFunctionHistory.append(loss)
    return theta , lossFunctionHistory


def predict(theta, X):
    p = np.round(sigmoid(X.dot(theta)))
    return p
```

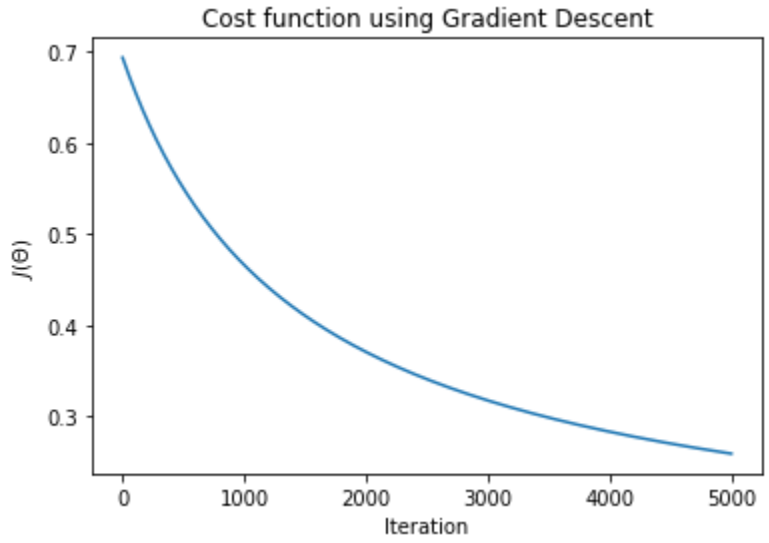
In [17]:
```
m , n = X.shape[0], X.shape[1]
X_normalized = normalizeData(X)
X_normalized = np.append(np.ones((m,1)),X_normalized,axis=1)
y = y.reshape(m,1)
initial_theta = np.zeros((n+1,1))
cost, grad= costFunction(initial_theta,X_normalized,y)
print("Initial theta cost is",cost)
```

```
Initial theta cost is [0.69314718]
```

In [18]:
```
theta , J_history = gradientDescent(X_normalized,y,initial_theta,0.001,5000)
```

In [19]:
```
plt.plot(J_history)
plt.xlabel("Iteration")
plt.ylabel("$J(\Theta)$")
plt.title("Cost function using Gradient Descent")
```

Out[19]:
```
Text(0.5, 1.0, 'Cost function using Gradient Descent')
```



In [20]:
```
print("Final Loss is ",J_history[-1])
```

```
Final Loss is  [0.25915116]
```

In [21]:
```
m , n = xTest.shape[0], xTest.shape[1]
Xtest_normalized = normalizeData(xTest)
Xtest_normalized = np.append(np.ones((m,1)),Xtest_normalized,axis=1)
p = predict(theta, Xtest_normalized)
print('Train Accuracy: {:.2f} %'.format(np.mean(p == yTest) * 100))
```

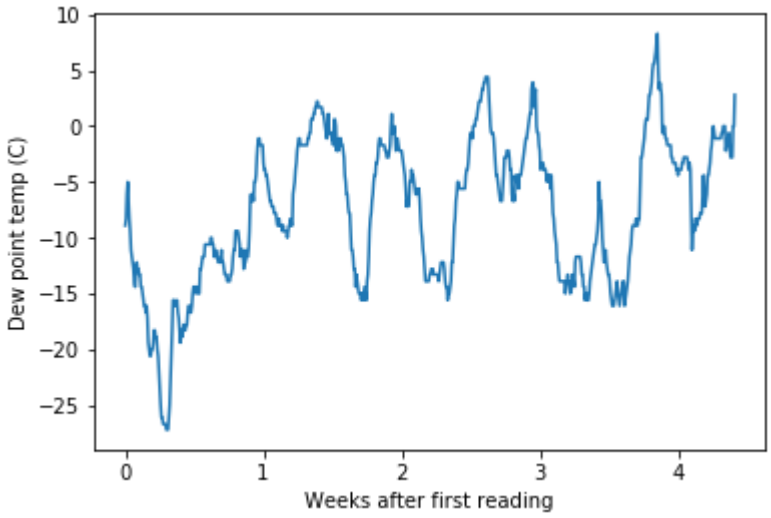
```
Train Accuracy: 50.01 %
```

In [ ]:
```
```

In [1]:
```
import scipy.io as sio
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import Ridge
from sklearn import metrics
import math
from numpy import linalg as LA
```

In [2]:
```
q4Data = sio.loadmat('weatherDewTmp.mat')
```

In [3]:
```
plt.plot(q4Data['weeks'].T, q4Data['dew'].T)
plt.xlabel('Weeks after first reading')
plt.ylabel('Dew point temp (C)')
plt.show()
```



In [4]:
```
X = q4Data['weeks'].T
y = q4Data['dew'].T
```

In [5]:
```
def conditionNumber(alpha,p,X):
    poly = PolynomialFeatures(degree= p-1)
    modifiedX = poly.fit_transform(X)
    X_reg = np.dot(modifiedX.T,modifiedX) + alpha*np.identity(p)
    return LA.cond(X_reg)
```

In [6]:
```
X = q4Data['weeks'].T
m = np.shape(X)[0]
pValues = [2,3,6,11]
alphaValues = [0,0.1*m,m,10*m,100*m]
for p in pValues:
    for alpha in alphaValues:
        A = conditionNumber(alpha, p, X)
        print("p = "+str(p), " alpha = " +str(alpha)+ " condition number for A = "+ str(A))
```

```
p = 2  alpha = 0 condition number for A =  32.47252166639873
p = 2  alpha = 74.2 condition number for A =  22.751009568791662
p = 2  alpha = 742 condition number for A =  6.754235401156347
p = 2  alpha = 7420 condition number for A =  1.6887590213669559
p = 2  alpha = 74200 condition number for A =  1.0702597386922308
p = 3  alpha = 0 condition number for A =  1476.3924507441645
p = 3  alpha = 74.2 condition number for A =  530.0467387593937
p = 3  alpha = 742 condition number for A =  79.11371158831945
p = 3  alpha = 7420 condition number for A =  9.20220564723802
p = 3  alpha = 74200 condition number for A =  1.82434509660882
p = 6  alpha = 0 condition number for A =  730851302.2885445
p = 6  alpha = 74.2 condition number for A =  2707863.7217707727
p = 6  alpha = 742 condition number for A =  271693.2496061315
p = 6  alpha = 7420 condition number for A =  27179.318088769152
p = 6  alpha = 74200 condition number for A =  2718.9227736362654
p = 11  alpha = 0 condition number for A =  7.20110883077699e+18
p = 11  alpha = 74.2 condition number for A =  3970315661934.136
p = 11  alpha = 742 condition number for A =  397031763152.4492
p = 11  alpha = 7420 condition number for A =  39703178289.79932
p = 11  alpha = 74200 condition number for A =  3970317849.5246153
```

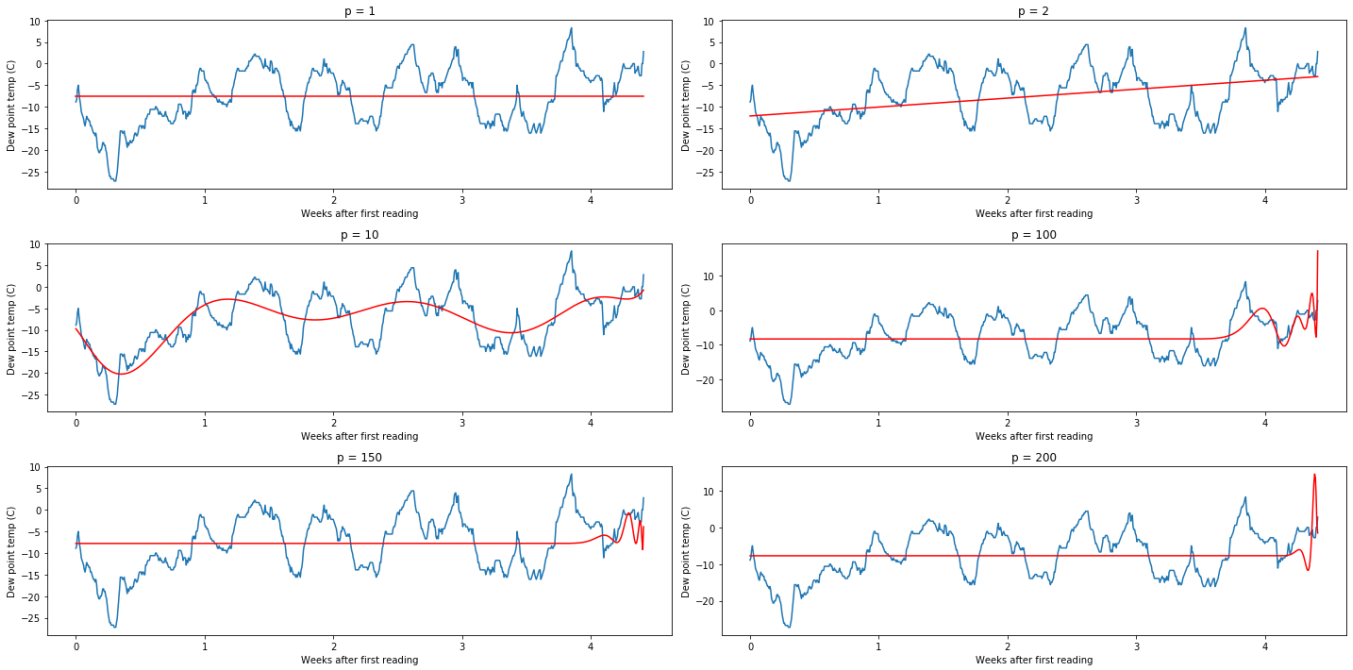
4f

p	$A, \alpha = 0$	$A_{reg}, \alpha = 0.1 * m$	$A_{reg}, \alpha = m$	$A_{reg}, \alpha = 10 * m$	$A_{reg}, \alpha = 100 * m$
2	32.47	22.75	6.75	1.68	1.07
3	1476.39	530.04	79.11	9.20	1.82
6	730851302.28	2707863.72	271693.24	27179.31	2718.92
11	7.299e+18	3970315661934.136	397031763152.4492	39703178289.79932	3970317849.5246153

In [7]:
```
def plotPolyFitLinearRegression(x,y,pValues):
    plt.figure(figsize=(20,10))
    for k in range(len(pValues)):
        plt.subplot(3,2,k+1)
        polynomial_features= PolynomialFeatures(degree=pValues[k-1]-1)
        x_poly = polynomial_features.fit_transform(x)
        model = LinearRegression()
        model.fit(x_poly, y)
        plt.title("p = "+str(pValues[k-1]))
        y_poly_pred = model.predict(x_poly)
        plt.plot(x, y)
        plt.plot(x, y_poly_pred,'r')
        plt.xlabel('Weeks after first reading')
        plt.ylabel('Dew point temp (C)')
    plt.tight_layout()
    return
```

4g

In [8]:
```
x = q4Data['weeks'].T
y = q4Data['dew'].T
pValues = [ 2,10, 100,150, 200,1]
plotPolyFitLinearRegression(x,y,pValues)
```

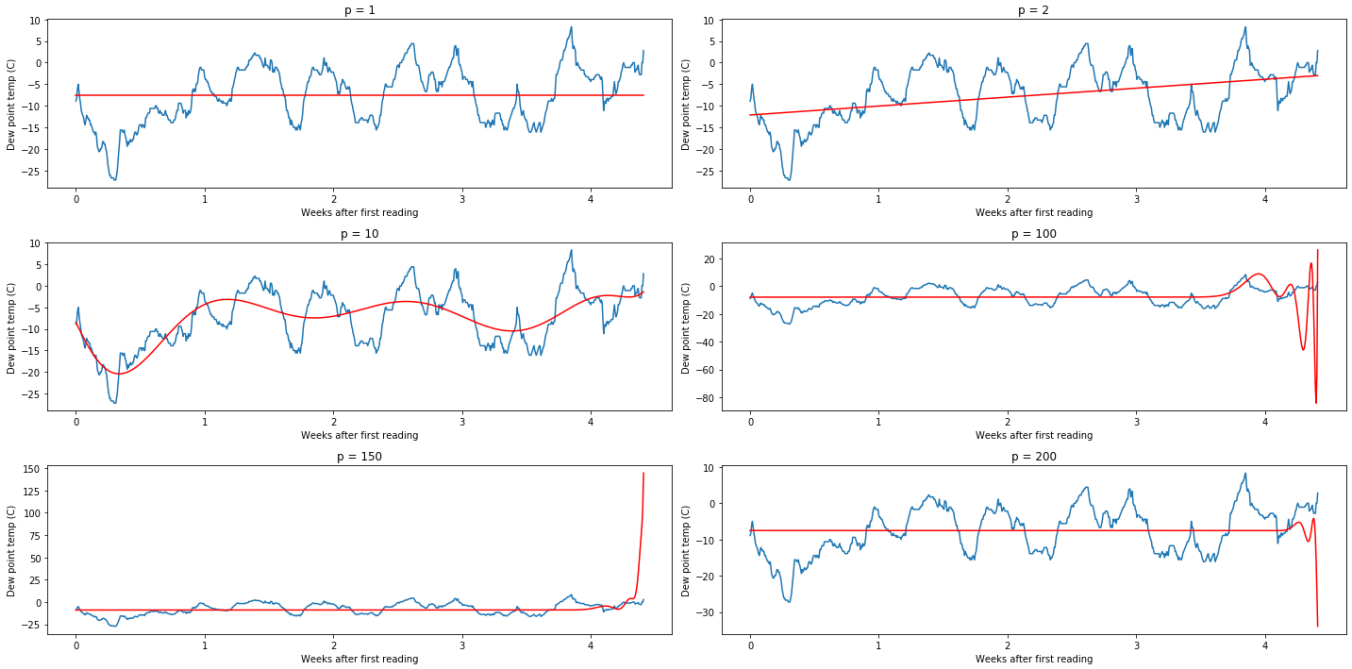


In [9]:
```
def plotPolyFitRidge(x,y,pValues,alpha=0.0001):
    plt.figure(figsize=(20,10))
    for k in range(len(pValues)):
        plt.subplot(3,2,k+1)
        plt.title("p = "+str(pValues[k-1]))
        polynomial_features= PolynomialFeatures(degree=pValues[k-1]-1)
        x_poly = polynomial_features.fit_transform(x)
        model = Ridge(alpha)
        model.fit(x_poly, y)
        y_poly_pred = model.predict(x_poly)
        plt.plot(x, y)
        plt.plot(x, y_poly_pred,'r')
        plt.xlabel('Weeks after first reading')
        plt.ylabel('Dew point temp (C)')
    plt.tight_layout()
    return
```

4h

In [10]:
```
x = q4Data['weeks'].T
y = q4Data['dew'].T
pValues = [ 2,10, 100,150, 200,1]
plotPolyFitRidge(x,y,pValues)
```

```
/Users/mvsnbharath/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/ridge.py:147: LinAlgWarning: Ill-conditioned matrix (rcond=5.97965e-18): result may not be accurate.
  overwrite_a=True).T
```



4i

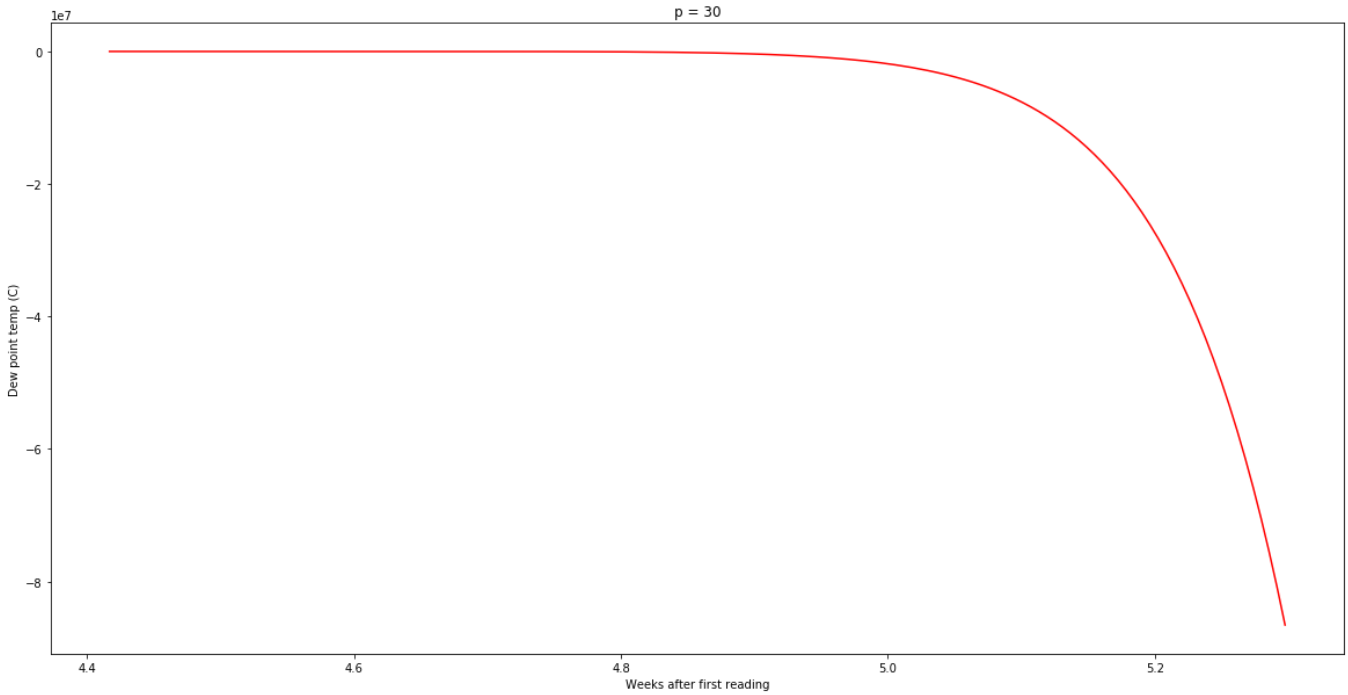
In [11]:
```
# adding one more week
xNew = []
temp = X[-1]
stepSize = 0.00595238
for i in range(1,150):
    xNew.append(temp + (i*stepSize))
```

In [12]:
```
xFinal = np.concatenate((x, xNew))
```

In [13]:
```
polynomial_features= PolynomialFeatures(degree=29)
x_poly = polynomial_features.fit_transform(x)
model = Ridge(alpha)
model.fit(x_poly, y)
x_poly_1 = polynomial_features.fit_transform(xNew)
y_poly_pred = model.predict(x_poly_1)
```

In [14]:
```
yFinal = np.concatenate((y, y_poly_pred))
```

In [15]:
```
plt.figure(figsize=(20,10))
plt.title("p = "+str(30))
# plt.plot(x, y)
plt.plot(xNew, y_poly_pred,'r')
plt.xlabel('Weeks after first reading')
plt.ylabel('Dew point temp (C)')
plt.show()
```



In [ ]: