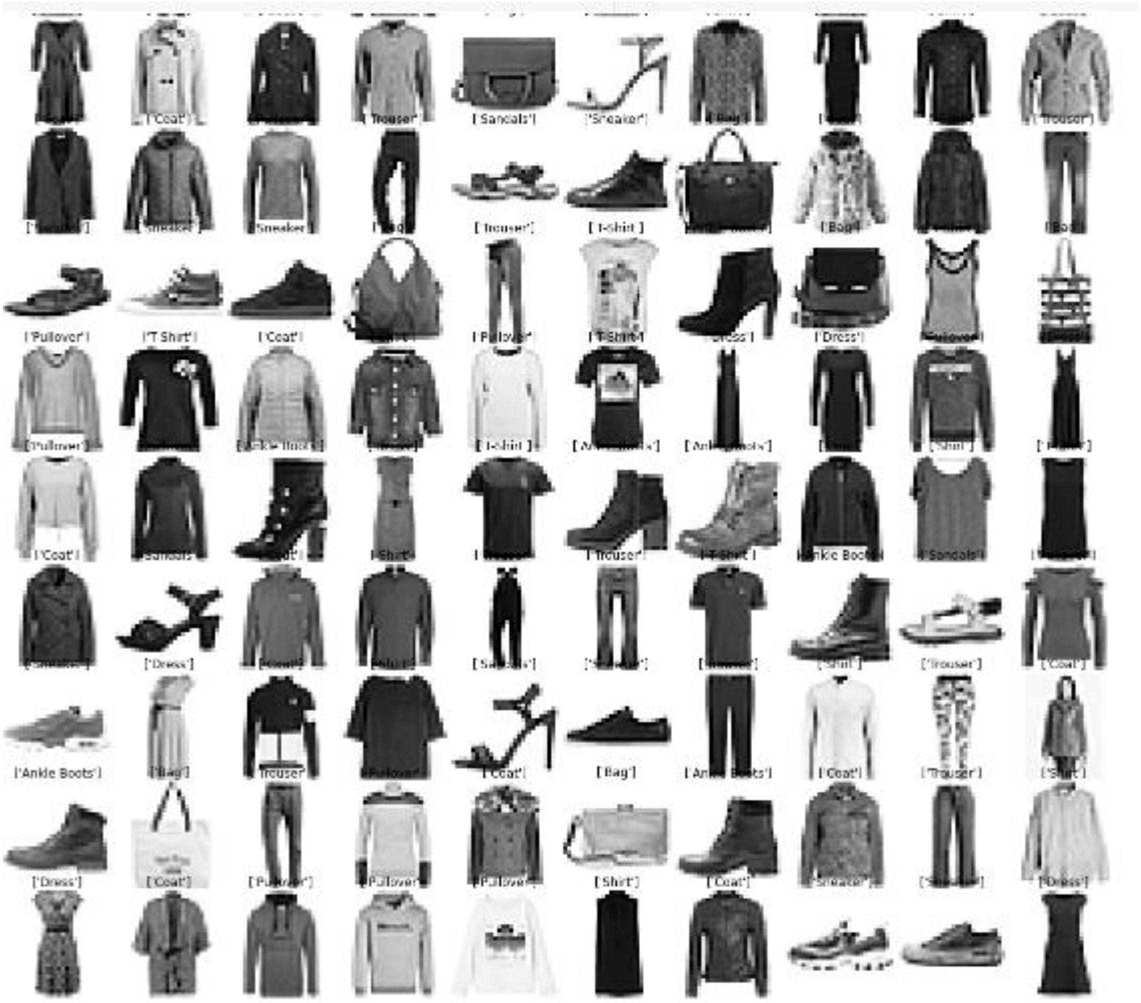
1.

```
label = np.array(["T-Shirt","Trouser","Pullover","Dress","Coat","Sandals","Shirt","Sneaker","Bag","Ankle Boots"])
data = pd.read_csv("/content/fashion-mnist_train.csv")
y = np.array(data['label'])
data = data.drop(['label'],axis=1)
x = np.array(data)
m,n = x.shape
y = y.reshape((y.size,1))
hei = int(np.round(np.sqrt(n)))
fig, ax_array = plt.subplots(10, 10, figsize=(20,20))
fig.subplots_adjust(wspace=0.025, hspace=0.025)
ax_array = ax_array.ravel()
for i, ax in enumerate(ax_array):
    ax.imshow(x[i,:].reshape(hei,hei, order='C'),cmap='Greys', extent=[0, 1, 0, 1])
    ax.set(title=label[y[i]])
    ax.axis('off')
```



2.

```
def data_clean(self,data):
        data.dropna(inplace=True)
        y = np.array(data['label'])
        data = data.drop(['label'],axis=1)
        m,n = data.shape
        i=0
        while i<n:
          if data.iloc[:,i].max()>0:
            data.iloc[:,i] = (data.iloc[:,i] - data.iloc[:,i].min())/(d
ata.iloc[:,i].max()-data.iloc[:,i].min())
          i = i+ 1
        x = np.array(data)
        y = y.reshape((y.size,1))
        return x, y

    def sigmoid(self,z):
        if np.isscalar(z):
          z = 1/(1+np.exp(-z))
        if (z.ndim==1):
          for i in range(z.size):
            z[i] = 1/(1+np.exp(-z[i]))
        if(z.ndim==2):
          m ,k = z.shape
          for i in range(m):
            for j in range(k):
              z[i,j] = 1/(1+np.exp(-z[i,j]))
        return z


    def costFunctionReg(self,w,x,y,lambda_):
        w = np.reshape(w,(int(w.size/10),10), order='C')
        m,n = x.shape
        q = x@w
        h = self.sigmoid(q)
        F = np.multiply(np.log(h),y)
        J1 = np.sum(F)
        G = np.multiply(np.log(1-h),1-y)
        J2 = np.sum(G)
        J3 = (lambda_/2)*np.sum(np.square(w[1:,:]))
        J = (-J1-J2+J3)/m
        grad = x.T @ (h - y) + (lambda_*w)
        grad[0,:] = grad[0,:]   - w[0,:]
        return J, np.ravel(grad/y.size)
```

```
def minCostFun(self, data):
        x,Y = self.data_clean(data)
        m,n=x.shape
        x = np.concatenate((np.ones((m,1)),x),axis=1)
        w = np.zeros(10*n+10)
        y = np.zeros((m,10))
        for i in range(m):
          y[i,Y[i,0]]=1
        opt={'maxiter':4000}
        res = optimize.minimize(self.costFunctionReg,w,(x,y,0.1),method
='TNC', jac=True,options=opt)
        w_ = res.x
        w_ = np.reshape(w_,(int(w_.size/10),10), order='C')
        a = self.predictOneVsAll(w_,x,10).reshape((m,1)) - Y
        q=0
        k=0
        for ele in a:
          if(ele==0):
            q+=1
          else:
            k+=1
        acr = np.around((q/m)*100,3)
        return np.around(w_,3),   acr

data = pd.read_csv("/content/fashion-mnist_train.csv")
obj = costing()
weight,acr = obj.minCostFun(data)
```

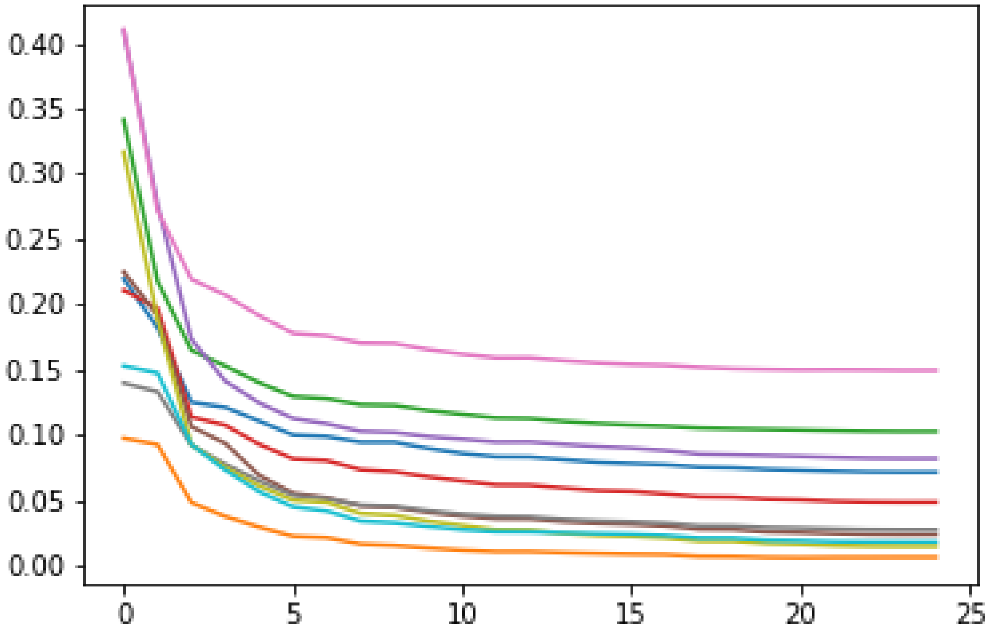
Accuray is coming 92%.

Weights have too many values was filling 12 pages, So, I haven’ added it.

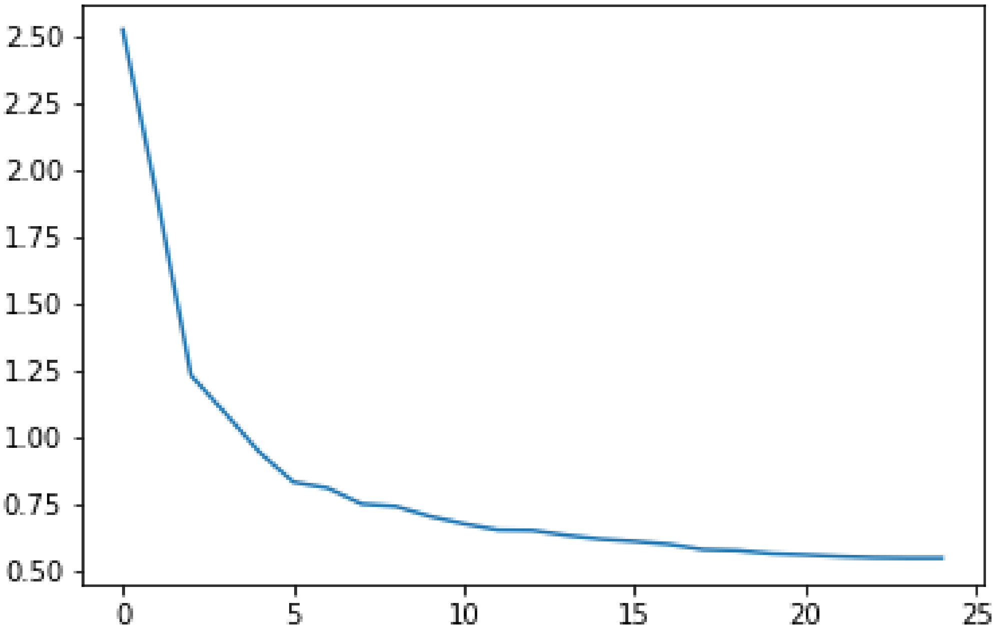
3.

I have used callbackfunction to plot J code for that is given below:

```
def Callback(self, v):
        #v is value of weights provided by minimize function at every
        iteration
        v = np.reshape(v,(int(v.size/10),10), order='C')
        m,n = X_.shape
        q = X_@ v
        h = self.sigmoid(q)
        F = np.multiply(np.log(h),Y_)
        J1 = np.sum(F, axis=0)
        G = np.multiply(np.log(1-h),1-Y_)
        J2 = np.sum(G, axis=0)
        s = (0.05)*np.square(v[1:,:])
        J3 = np.sum(s,axis=0)
        J_cost.append((-J1-J2+J3)/m)
J_cost = []
data = pd.read_csv("/content/fashion-mnist_train.csv")
obj = costing()
X_,Y1 = obj.data_clean(data)
m,n=X_.shape
X_ = np.concatenate((np.ones((m,1)),X_),axis=1)
Y_ = np.zeros((m,10))
for i in range(m):
    Y_[i,Y1[i,0]]=1
weight,acr = obj.minCostFun(data)
label = np.array(["T-
Shirt","Trouser","Pullover","Dress","Coat","Sandals","Shirt","Sneaker",
"Bag","Ankle Boots"])
k = np.array(J_cost)
for i in range(10):
  plt.plot(k[:,i])
plt.savefig("Class_Wise_Cost.png")
```



Class-Wise Cost Vs. Iterations



OverAll Cost Iterations.

4.

```
def predictOneVsAll(self,w,x,num_labels):
        m, n = x.shape
        h = self.sigmoid(x @ w)
        s = np.nanargmax(h, axis=1).reshape((m,1))
        return s

def TestingAccu(self, test_data,w_):
        x,Y = self.data_clean(data)
        m,n=x.shape
        x = np.concatenate((np.ones((m,1)),x),axis=1)
        a = self.predictOneVsAll(w_,x,10).reshape((m,1)) - Y
        q=0
        k=0
        for ele in a:
          if(ele==0):
            q+=1
          else:
            k+=1
        acr = np.around((q/m)*100,3)
        return  acr
```

Accuracy on testing data is coming 90%.

5.

```
label = np.array(["T-
Shirt","Trouser","Pullover","Dress","Coat","Sandals","Shirt","Sneaker",
"Bag","Ankle Boots"])
obj = costing()
data2 = pd.read_csv("/content/fashion-mnist_test.csv")
X2,Y2 = obj.data_clean(data2)
m,n=X2.shape
X__= np.concatenate((np.ones((m,1)),X2),axis=1)
a = obj.predictOneVsAll(weight,X__,10).reshape((m,1))
hei = int(np.round(np.sqrt(n)))
fig, ax_array = plt.subplots(3, 10, figsize=(20,20))
fig.subplots_adjust(wspace=0.025, hspace=0.025)
ax_array = ax_array.ravel()
for i, ax in enumerate(ax_array):
    ax.imshow(X2[i,:].reshape(hei,hei, order='C'),
                              cmap='Greys', extent=[0, 1, 0, 1])
    ax.set(title=label[Y2[i]])
    ax.axis('off')
```