Machine Learning ICA-01

import numpy as np

import matplotlib.pyplot as plt

theta0 = np.random.uniform(0, 1, 1000)

theta1 = np.random.uniform(0, 1, 1000)

plt.scatter(theta0, theta1)

plt.title("Θ⁰ vs Θ¹")

plt.xlabel("Θ⁰")

plt.ylabel("Θ¹")

plt.show()

def cost\_function(x, y, theta0, theta1):

  m = len(x)

  total\_error = 0.0

  for i in range(m):

    total\_error += (y[i] - (theta0 + theta1 \* x[i]))\*\*2

  return total\_error / m

x = np.linspace(0, 10, 100)

y = 2 + 0.5 \* x

theta0 = np.random.uniform(0, 1, 1000)

theta1 = np.random.uniform(0, 1, 1000)

cost = np.zeros(shape=(1000, 1000))

for i in range(1000):

  for j in range(1000):

    cost[i,j] = cost\_function(x, y, theta0[i], theta1[j])

from mpl\_toolkits.mplot3d import Axes3D

fig = plt.figure()

ax = fig.gca(projection='3d')

ax.plot\_surface(theta0, theta1, cost, cmap='viridis', linewidth=0.2)

ax.set\_xlabel('Θ⁰')

ax.set\_ylabel('Θ¹')

ax.set\_zlabel('J(Θ⁰, Θ¹ )')

plt.title("J(Θ⁰, Θ¹ ) vs Θ⁰ vs Θ¹")

plt.show()

min\_index = np.argmin(cost)

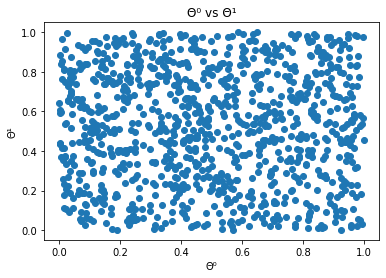
i, j = np.unravel\_index(min\_index, cost.shape)

theta\_0\_opt = theta0[i]

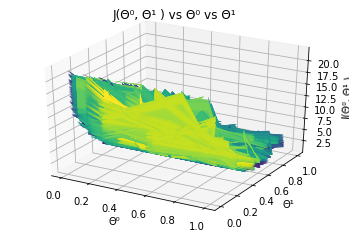
theta\_1\_opt = theta1[j]

print("The best fitting model is f(x) = {} + {}x".format(theta\_0\_opt, theta\_1\_opt))

**Teta0 vs Teta1 Graph**

****

**J(Teta0,Teta1) vs Teta0 vs Teta1 Graph**

****