Identifying fraudulent taxpayers using Variational Autoencoders

Team: Puneet Mangla (CS17BTECH11029)
Yash Khasbarge (CS17BTECH11044)
Rushikesh Tammewar (CS17BTECH11041)

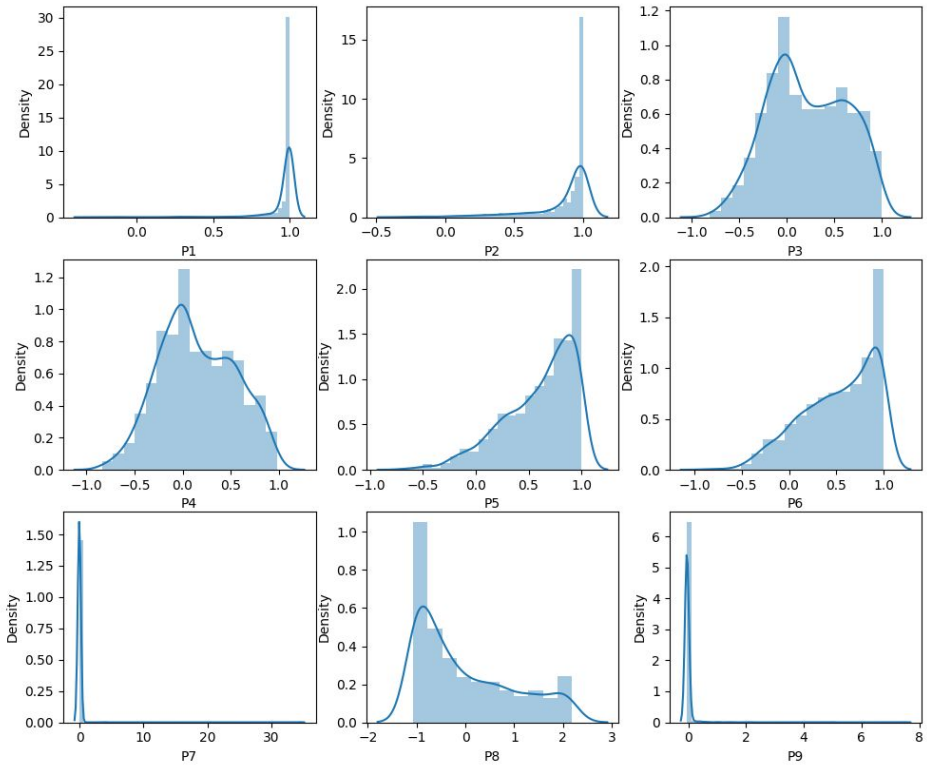
Objective: Implement a Variational Autoencoder (VAE) to identify fraudulent taxpayers. Given data points **x =** [x_1, x_2 ...x_n,] train a VAE $D_\theta \circ E_\phi$ by minimizing the following objective

$$J(\theta, \phi) = \frac{1}{N} \sum_{i=1}^{N} \left|\left| D_\theta \circ E_\phi(x_i) - x_i \right|\right|_2 + KL(E_\phi(x) \,||\, z \sim N(0, I))$$

After training the VAE, we cluster the data points using K-means clustering after projecting them in encoder representation i.e $E_\phi(x)$

Dataset Summary:

- 1163 Samples
- 9 Features, P1 to P9
- Here is the distribution of all 9 features



-
- Mean of sample is :

[0.959524 , 0.85616583 , 0.2161109 , 0.14919403 ,0.60036861, 0.52484523,00098841, -0.00269966 ,-0.02701196]

- Variance of sample is :
  [0.01632525, 0.05948979 ,0.16666798, 0.15047038 ,0.11192182 ,0.14930412, 1.03006199,0.99289213, 0.08219917]

- Examples

	A	B	C	D	E	F	G	H	I
1	P1	P2	P3	P4	P5	P6	P7	P8	P9
2	0.595378442	-0.531958013	0.679654045	-0.126799374	0.432045865	0.988091839	-0.029812601	0.768741642	-0.054166781
3	0.982236593	0.991480958	0.337646147	0.228143988	0.920032191	0.999985324	-0.032258992	2.161650801	-0.05434974
4	0.996162337	0.893987494	0.767413498	0.606839713	0.970807848	0.882601755	-0.032266729	0.369607281	-0.05415696
5	0.999928399	0.922747892	-0.444438457	-0.371287107	0.528037544	-0.221645387	-0.032692409	-1.065439464	0.381914177
6	0.985838408	0.937511528	0.699591982	0.585263408	0.838804121	0.999601694	-0.033713342	2.08972004	-0.054379021
7	0.994265688	0.917955271	0.441420963	0.588541183	0.714375568	0.997313804	-0.030094889	2.081266066	-0.054256954
8	0.999477958	0.989320554	0.65413855	0.705176765	0.350524732	0.877973508	-0.030983776	0.605392647	-0.054115822
9	0.999645753	0.275265043	-0.33356861	-0.441021828	0.576963119	0.607380222	-0.032128802	1.039586381	-0.054263518
10	0.825371384	-0.24824377	0.220238261	0.128478367	0.364442719	0.951264631	-0.030247053	1.512327358	-0.054220936

Experimentation Results:

- Implementation Details:
  - Number of epochs: 1000
  - Adam Optimizer with learning rate 1e-3 used for optimizing
  - We take the hidden dimension of VAE as 2 so that we can plot the points on a 2D plane for visualization

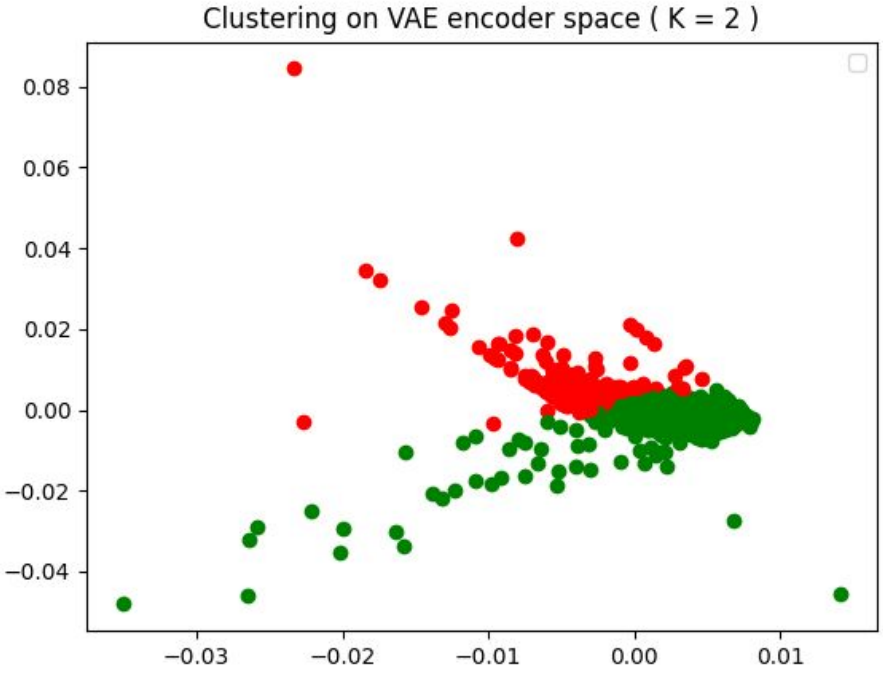
```
class VAE(nn.Module):
    def __init__(self, out_dim=9):
        super(VAE, self).__init__()
        self.encoder  = nn.Sequential( nn.Linear(out_dim, out_dim//2),
                                       nn.ReLU(),
                                       nn.Linear(out_dim//2, 2*out_dim//4),)

        self.decoder = nn.Sequential( nn.Linear(out_dim//4, out_dim//2),
                                       nn.ReLU(),
                                       nn.Linear(out_dim//2, out_dim))

    def reparameterize(self, mu, logvar):
        std = logvar.mul(0.5).exp_()
        esp = torch.randn(*mu.size())
        z = mu + std * esp
        return z

    def forward(self, x):
        h = self.encoder(x)
        mu, logvar = torch.chunk(h, 2, dim=1)
        z = self.reparameterize(mu, logvar)
        xr = self.decoder(z)
        return xr, mu, logvar
```

- Results:



From the plot, we can see that the data gets separated into red and green clusters.
#Points in the red cluster are 352
#Points in the green cluster are 811

So Red cluster probably depicts fraudulent taxpayers and Green ones as genuine taxpayers.