Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Assignment 4

Table of Contents

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Introduction

The purpose of this assignment is to create a GAN neural network that can generate samples from the distribution of two different datasets. Several preprocessing steps were executed to prepare the data for the training phase. The Generator network learns to generate samples the manage to fool the Discriminator network.

Datasets

The datasets used in this assignment are diabetes and German credit. The diabetes data set includes 768 samples with 8 features per sample, and a patient condition. The German credit data set includes 1000 samples with 20 features per sample, and a credit status (good or bad customer). All the samples were used as part of the training phase.

Preprocess

For both datasets, the same preprocessing steps were applied. For the numeric features, a min-max normalization is applied so all the features will be in the same range - [0,1]. For the categorical features, we used one-hot encoding and dummy features. Each categorical feature was expanded into several features by the number of applicable values of it. The Diabetes Dataset is of shape (768,9) and the German credit is of shape (1000,62).

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Network's architecture

The Network's architecture we used is a typical GAN. We created 2 different NNs the first it the generator network which constructed by _ fully connected (dense) layers, the first layer is in the size of randomly generated noise vector – 100. Which then followed by several dense layers while the last layer size is of the targeted dataset number of features. The second network is the discriminator. Which is also constructed by _ dense layers. The first layer (input) size is the size of the generator output. The discriminator's output size is always 1, because its only goal is to classify whether the inputs are real or synthetic. Furthermore, as mentioned above the architecture of the model and the inputs size changes according to targeted dataset. in both models the specific number of units in each layer is different and determined as a function of the number of features and noise vector length. The exact architecture of both models is presented in appendix A.

For each categorical feature we created dummy variables to assist the training. This approach required further processing to match the generator output samples to the given dataset samples. The maximum value of a specific categorical feature's values was set to 1 and all the others dummy variables of this feature set to 0.

During the training phase we discovered that the discriminator model can recognize the real data more quickly than the generator is able to learn how to generate data like the real one. While trying to overcome this issue we added Dense layers to both models, hoping that it will help the generator train better. This addition increased the number of parameters that each model needs to learn, yet the discriminator still recognized the real data faster than the generator could train. The next steps we took to try and mitigate this issue, were to only train the discriminator every few training steps, in other words, we freeze the discriminator values so it learns slower than the generator, and reducing the learning rate of the discriminator relatively to the generator's.

We conclude that the generator has difficulty competing with the discriminator because its target task is much harder. Furthermore, in the process of exploring the capabilities and behaviors of these models, we observed several instances of mode collapse. In mode collapse the generator learns to generate samples with very small variety (very similar sample or even 1 sample at all). Also, we observed several instances of non-convergence. The reported on models are the best performing models that we could train in the given time.

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

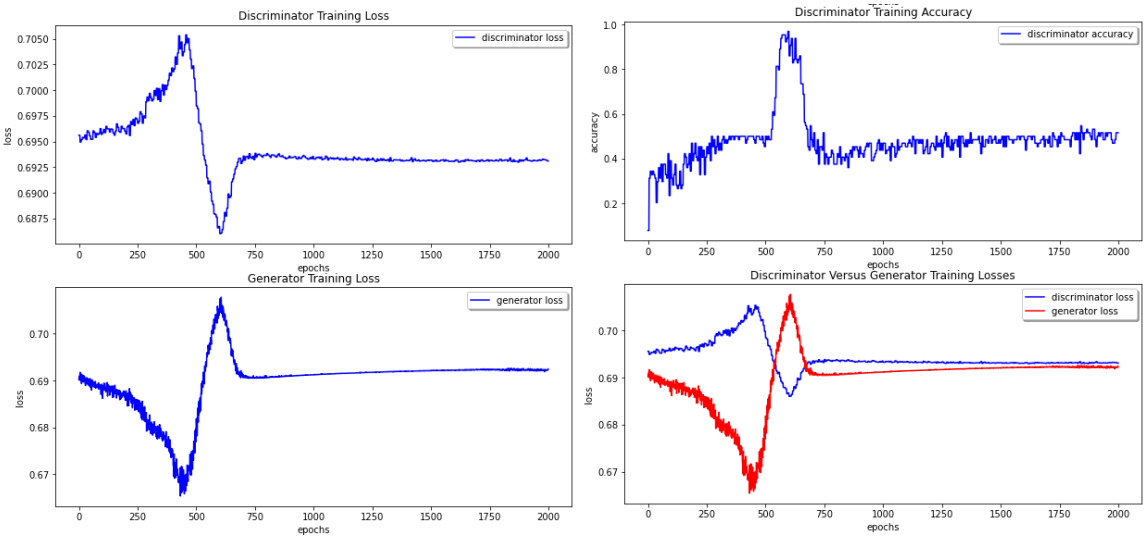
Part One

Training Phase

For the first part of this assignment, we trained our GANs over the entire datasets, in other words, we did not split the datasets into train and test set.

Performance Analysis – Diabetes dataset

The model is fitted on the training data over 2000 epochs with batch size of 32 and a low learning rate of 0.00005. As mentioned above we updated the generator model on every training step, while the discriminator model was updated every 5 steps. The following graphs show the losses of both models and the accuracy of the discriminator model.



The graphs clearly show that the models were “fighting” to overcome one another. When the discriminator loss is high the loss of the generator is low. We see that the models learn to overcome one another, as shown by the fact that the lowest loss model is changing throughout the training phase (generator->discriminator->generator).
Another insight we can draw from the graphs is that the models converged to a local minimum. The losses of both models have stabilized and the accuracy of the discriminator revolves around 50% for more then 1000 epochs.

We generated 100 samples using the trained generator. The number of synthetic samples that were able to pass as real samples is 94. A close examination of the synthetic samples may reveal more interesting insights to the decision made by the discriminator. As a crude and basic method to explore the similarity between the synthetic and real data we calculated for each class a mean vector of features from the dataset samples.

age	class	insu	mass	pedi	plas	preg	pres	skin
0.267786	1.0	0.118600	0.523734	0.201751	0.709836	0.286216	0.580530	0.223881
0.169833	0.0	0.081314	0.451627	0.150185	0.552663	0.194000	0.558885	0.198626

Samples that the discriminator determined as real:

	preg	plas	pres	skin	insu	mass	pedi	age	class
0	0.198404	0.487252	0.0	0.0	0.606112	0.184121	0.0	0.0	0.0
1	0.343596	0.424778	0.0	0.0	0.378338	0.122793	0.0	0.0	1.0

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

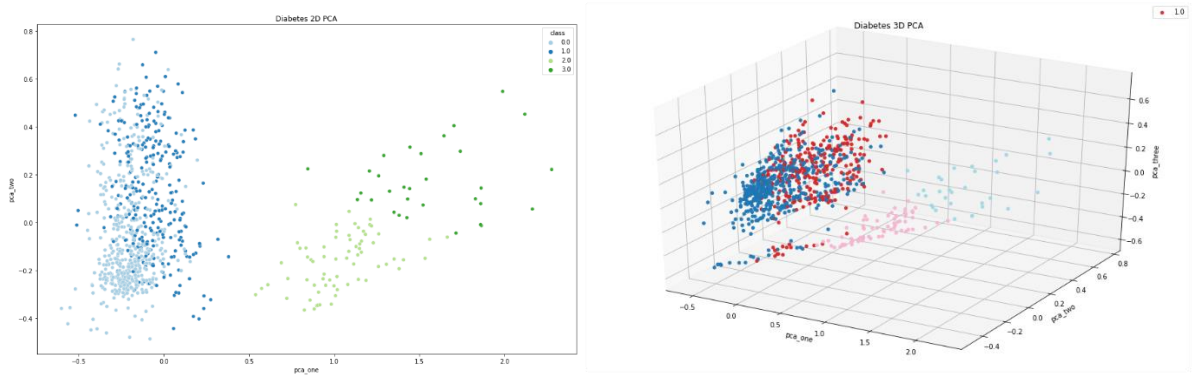
Samples that the discriminator determined as fake:

	preg	plas	pres	skin	insu	mass	pedi	age	class
8	0.131395	0.367635	0.0	0.0	0.310553	0.135454	0.0	0.0	0.0
9	0.246256	0.547841	0.0	0.0	0.472628	0.217935	0.0	0.0	1.0

We cannot clearly see that the samples that were classified as real by the discriminator are much like the original data. The usage of Cosine Similarity and Euclidean distance reveals no significant difference that would indicate that the samples that were classified as real are more similar to the original data than the samples that were classified as fake.

Metrics compared to avg/mode vectors of each class	Class 0 – judged “real”	Difference ⇔	Class 0 – judged “fake”	Class 1 – judged “real”	Difference ⇔	Class 1 – judged “fake”
Cosine Similarity	0.54652	**0.07663**	0.62315	0.81152	**0.0165**	0.82802
Euclidean distance	0.86860	**-0.09797**	0.77063	0.90128	**-0.03743**	0.86385

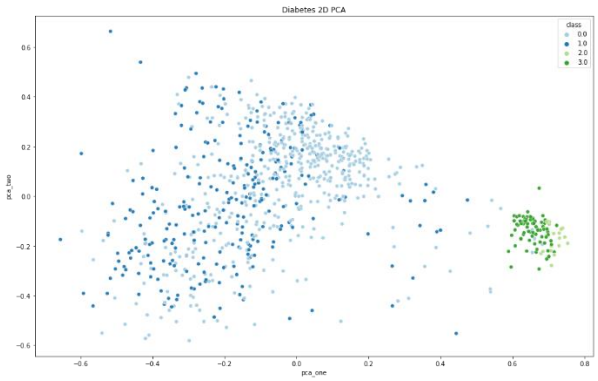
To inspect the similarity further we used PCA to reduce the features dimension from 9 to 2 and 3 dimensions. In the next experiments we used this form of qualitive analysis.

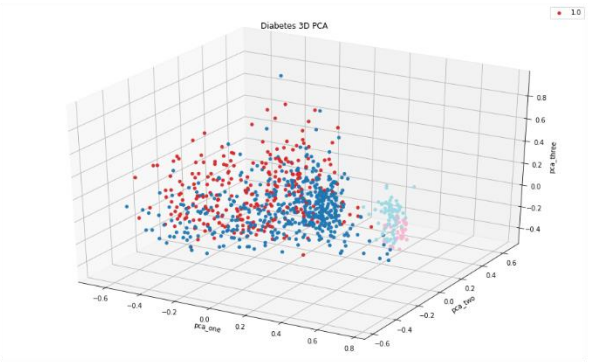


Visualizing the results, a clear separation between the synthetic and real samples can be observed. Several synthetic samples appear to resemble real samples more closely. Therefore, we can draw two different conclusions. First, the generator model is not yet capable of fully mimicking the distribution of the original data. Secondly, the discriminator is too weak or converged to a local minimum, which inhibits it to effectively separate the real and the fake samples.

As mentioned above, we also encountered several instances of mode collapse. We can observe in the following graphs such instance

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

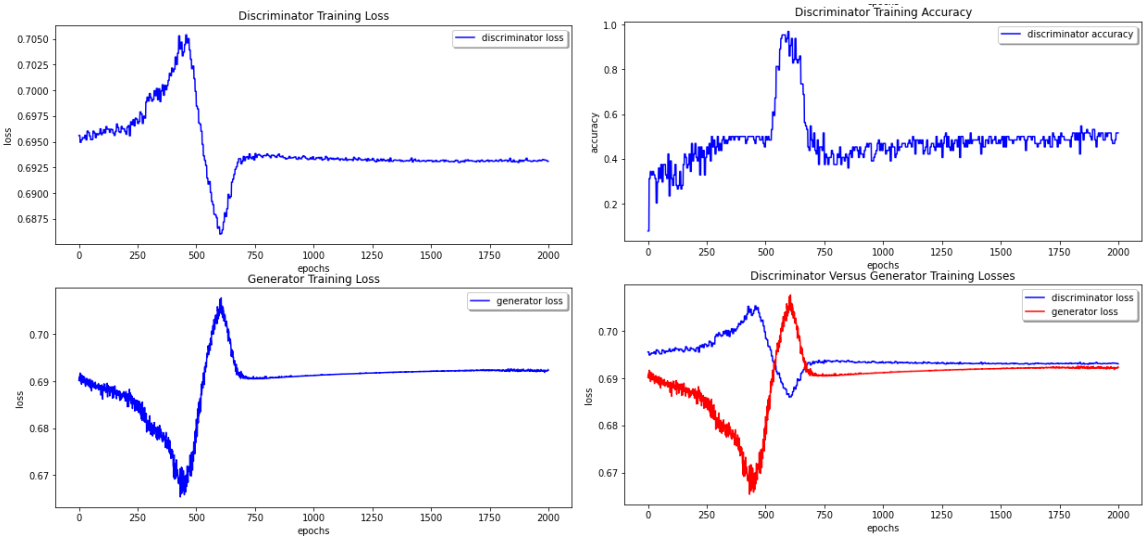




The generator model did learn to generate samples that somewhat resemble the real data but it produced samples with low variance.

Performance Analysis – German Credit dataset

The model is fitted on the training data over 2000 epochs with batch size of 32 and a low learning rate of 0.00001. As mentioned above we updated the generator model on every training step, while the discriminator model was updated every 5 steps. The following graphs show the losses of both models and the accuracy of the discriminator model.



The graphs clearly show that the models were “fighting” to overcome one another. When the discriminator loss is high the loss of the generator is low. We see that the models learn to overcome one another, as shown by the fact that the lowest loss model is changing throughout the training phase (generator->discriminator->generator).
Another insight we can draw from the graphs is that the models converged to a local minimum. The losses of both models have stabilized and the accuracy of the discriminator revolves around 50% for more then 1000 epochs.

We generated 100 samples using the trained generator. The number of synthetic samples that were able to pass as real samples is 94. A close examination of the synthetic samples may reveal more interesting insights to the decision made by the discriminator. As a crude and basic method to explore the similarity between the synthetic and real data we calculated for each class a mean vector of features from the dataset samples.

age	class	insu	mass	pedi	plas	preg	pres	skin
0.267786	1.0	0.118600	0.523734	0.201751	0.709836	0.286216	0.580530	0.223881
0.169833	0.0	0.081314	0.451627	0.150185	0.552663	0.194000	0.558885	0.198626

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Samples that the discriminator determined as real:

	preg	plas	pres	skin	insu	mass	pedi	age	class
0	0.198404	0.487252	0.0	0.0	0.606112	0.184121	0.0	0.0	0.0
1	0.343596	0.424778	0.0	0.0	0.378338	0.122793	0.0	0.0	1.0

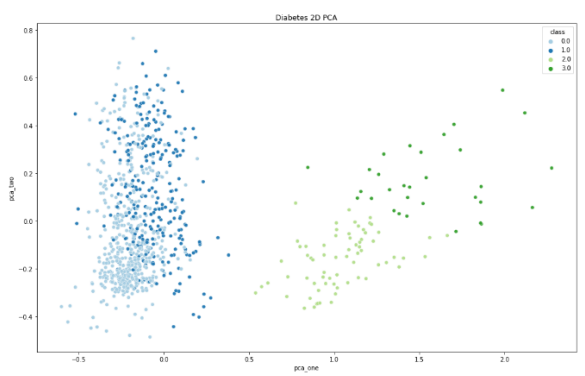
Samples that the discriminator determined as fake:

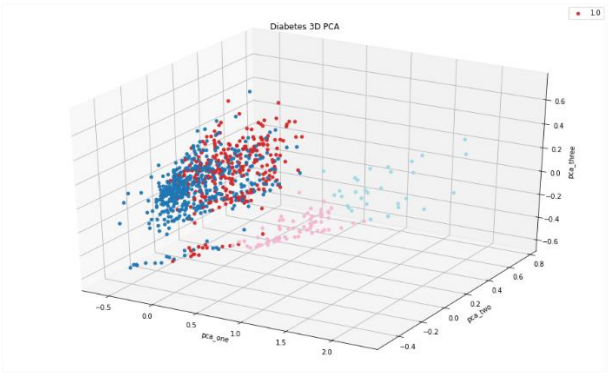
	preg	plas	pres	skin	insu	mass	pedi	age	class
8	0.131395	0.367635	0.0	0.0	0.310553	0.135454	0.0	0.0	0.0
9	0.246256	0.547841	0.0	0.0	0.472628	0.217935	0.0	0.0	1.0

We cannot clearly see that the samples that were classified as real by the discriminator are much like the original data. The usage of Cosine Similarity and Euclidean distance reveals no significant difference that would indicate that the samples that were classified as real are more similar to the original data than the samples that were classified as fake.

Metrics compared to avg/mode vectors of each class	Class 0 – judged "real"	Difference ⟺	Class 0 – judged "fake"	Class 1 – judged "real"	Difference ⟺	Class 1 – judged "fake"
Cosine Similarity	0.54652	**0.07663**	0.62315	0.81152	**0.0165**	0.82802
Euclidean distance	0.86860	**-0.09797**	0.77063	0.90128	**-0.03743**	0.86385

To inspect the similarity further we used PCA to reduce the features dimension from 9 to 2 and 3 dimensions. In the next experiments we used this form of qualitive analysis.

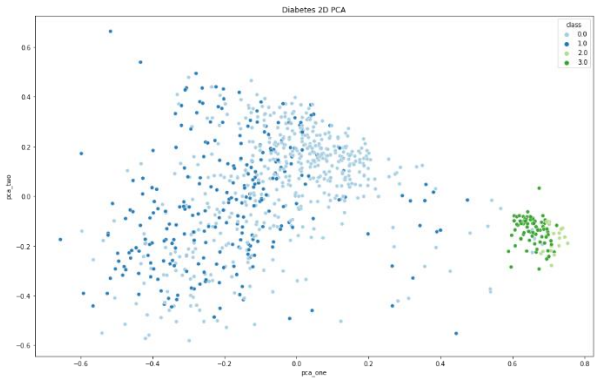


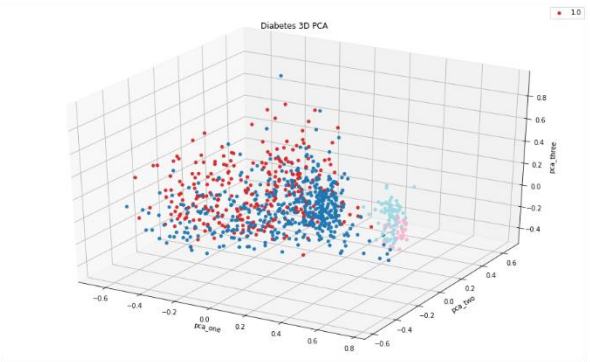


Visualizing the results, a clear separation between the synthetic and real samples can be observed. Several synthetic samples appear to resemble real samples more closely. Therefore, we can draw two different conclusions. First, the generator model is not yet capable of fully mimicking the distribution of the original data. Secondly, the discriminator is too weak or converged to a local minimum, which inhibits it to effectively separate the real and the fake samples.

As mentioned above, we also encountered several instances of mode collapse. We can observe in the following graphs such instance

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268





The generator model did learn to generate samples that somewhat resemble the real data, but it produced samples with low variance.

Part Two

Train-Test split

We used 70-30 split on the datasets, resulting in training and testing sets in the following sizes:

- Diabetes Training set: 537 samples
- Diabetes Testing set: 231 samples
- German Credit Training set: 700 samples
- German Credit Testing set: 300 samples

Random Forest Classifier – Black Box Model

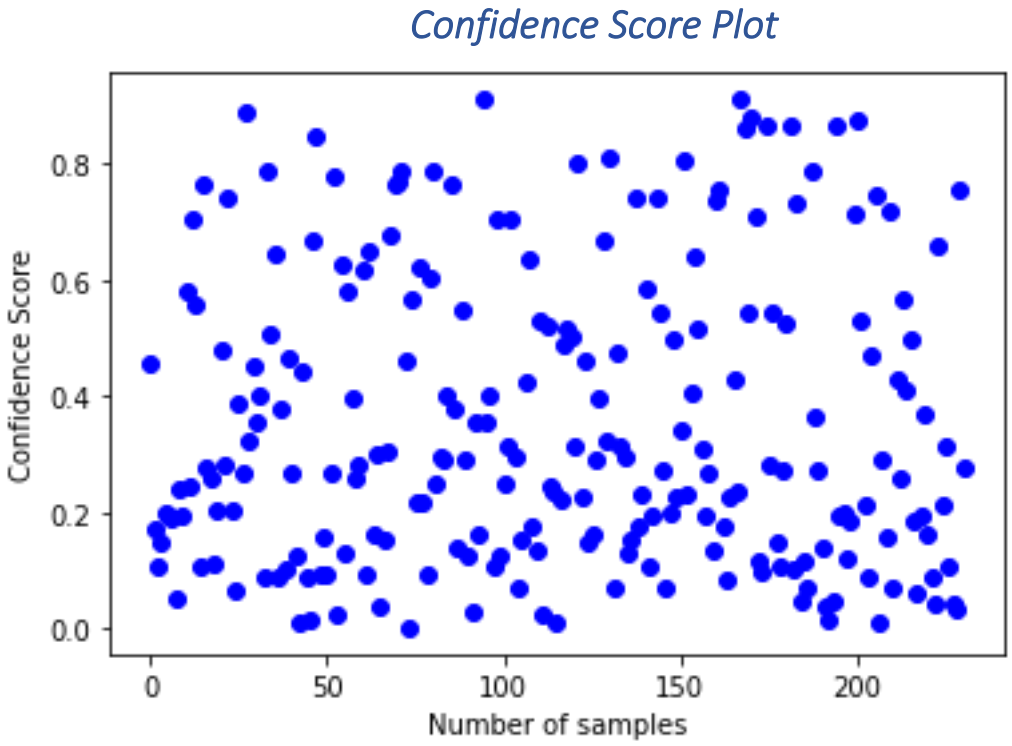
We trained a Random Forest classifier on each of the datasets using both Grid Search and Cross Validation. We searched over a range of several hyper parameters (included in the appendix) and chose the combination which produced the highest performing model. The best performing model was then trained on all the training set data and used as the Black Box model.

Performance of the Black Box model – Test set

The Black Box model performance on the diabetes test set with the hyper parameters used:

- {'criterion': 'gini', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 8, 'min_samples_split': 2, 'n_estimators': 9}
- Min confidence score: 0.0020768431983385254
- Max confidence score 0.9095931457773562
- Average confidence score: 0.35276447991484494

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

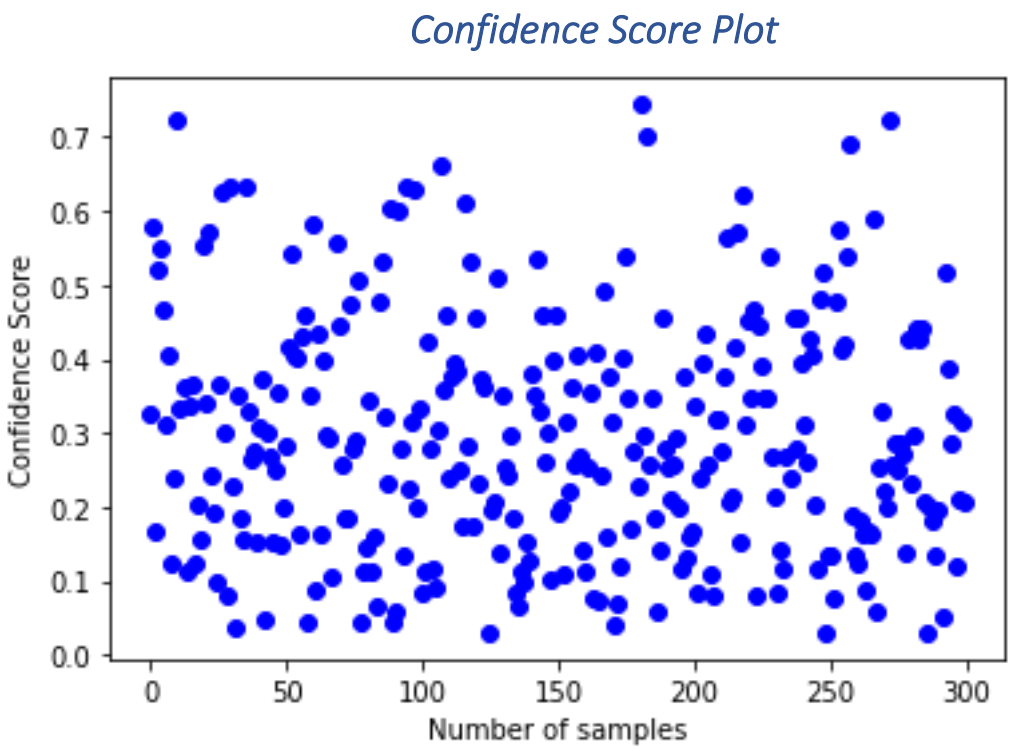


Final model train set accuracy: 0.8379888268156425
Final model test set accuracy: 0.7965367965367965

The Black Box model performance on the german credit test set with the hyper parameters used:

- {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 5, 'min_samples_split': 2, 'n_estimators': 9}
- Min cs: 0.03047194351542178
- Max cs 0.7438146648672964
- Average cs 0.2940039262577002

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268



Final model train set accuracy: 0.8342857142857143
Final model test set accuracy: 0.7433333333333333

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Performance

The model is fitted on the training data over 40 epochs, we used exponential learning rate decay with starting value of 0.001, number of decay steps 1000 and decay rate of 0.95. As mentioned above the model is overfitted to the training data as represented in the graphs below:

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Results

Possible improvements

While trying to overcome the non-convergence and the mode collapses, we suggest several possible improvements. As for the mode collapse, we suggest implementing a mini-batch GAN approach that would reward the diversity of the generator generated samples. In this approach the discriminator would discriminate based on the diversity of the entire batch of generated samples. To overcome the non-convergence, we suggest several steps including, architectural changes to the NNs and making the discriminator's task harder. A possible way to accomplish that is to add an additional task to it such as determining the label of the generated sample (supervision with labels).

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Appendix

Part 1

Diabetes dataset

Discriminator Model

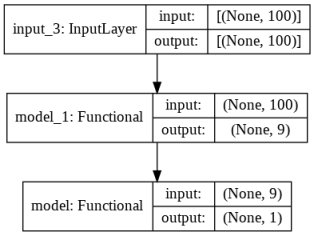
```
Model: "discriminator"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 9)                 90
_________________________________________________________________
dense_1 (Dense)              (None, 9)                 90
_________________________________________________________________
dense_2 (Dense)              (None, 10)                100
_________________________________________________________________
dense_3 (Dense)              (None, 10)                110
_________________________________________________________________
dense_4 (Dense)              (None, 10)                110
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 11
=================================================================
Total params: 511
Trainable params: 511
Non-trainable params: 0
```

Generator Model

```
Model: "generator"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_6 (Dense)              (None, 100)               10100
_________________________________________________________________
dense_7 (Dense)              (None, 100)               10100
_________________________________________________________________
dense_8 (Dense)              (None, 9)                 909
_________________________________________________________________
dense_9 (Dense)              (None, 9)                 90
_________________________________________________________________
dense_10 (Dense)             (None, 9)                 90
=================================================================
Total params: 21,289
Trainable params: 21,289
Non-trainable params: 0
```

Combined Model

```
Model: "combined"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         [(None, 100)]             0
_________________________________________________________________
model_1 (Functional)         (None, 9)                 21289
_________________________________________________________________
model (Functional)           (None, 1)                 511
=================================================================
Total params: 21,800
Trainable params: 21,289
Non-trainable params: 511
```



German Credit dataset

Discriminator Model

Generator Model

Yiftach Savransky – 312141369
Gal Rosenthal – 312585268

Part 2

Hyper Parameters – Black Box Model

'n_estimators': [4, 6, 9],
'max_features': ['log2', 'sqrt','auto'],
'criterion': ['entropy', 'gini'],
'max_depth': [2, 3, 5, 10],
'min_samples_split': [2, 3, 5],
'min_samples_leaf': [1,5,8]

Diabetes dataset

Discriminator Model

Generator Model

German Credit dataset

Discriminator Model

Generator Model