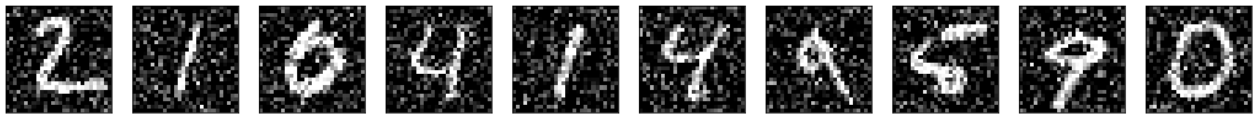
De-noising like an autoencoder

Okay, you have just built an `autoencoder` model. Let's see how it handles a more challenging task.

First, you will build a model that encodes images, and you will check how different digits are represented with `show_encodings()`. You can change the `number` parameter of this function to check other digits in the console.

Then, you will apply your `autoencoder` to noisy images from `MNIST`, it should be able to clean the noisy artifacts.

`X_test_noise` is loaded in your workspace. The digits in this data look like this:



Apply the power of the autoencoder!

- Build an `encoder` model with the first layer of your trained `autoencoder` model.
- Predict on `X_test_noise` with your `encoder` and show the results with `show_encodings()`.

```
# Build your encoder

encoder = Sequential()

encoder.add(autoencoder.layers[0])


# Encode the images and show the encodings

preds = encoder.predict(X_test_noise)

show_encodings(preds)
```

- Predict on `X_test_noise` with your `autoencoder`.
- Plot noisy vs decoded images with `compare_plot()`.

```
# Build your encoder

encoder = Sequential()

encoder.add(autoencoder.layers[0])


# Encode the images and show the encodings

preds = encoder.predict(X_test_noise)

show_encodings(preds)


# Predict on the noisy images with your autoencoder

decoded_imgs = autoencoder.predict(X_test_noise)


# Plot noisy vs decoded images

compare_plot(X_test_noise, decoded_imgs)
```