MNIST_TSNE_Demonstration

[ ]:
```
import pandas as pd
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import fetch_openml
import seaborn as sb

mnist = fetch_openml('mnist_784')
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/datasets/_openml.py:968:
FutureWarning: The default value of `parser` will change from `'liac-arff'` to
`'auto'` in 1.4. You can set `parser='auto'` to silence this warning. Therefore,
an `ImportError` will be raised from 1.4 if the dataset is dense and pandas is
not installed. Note that the pandas parser may return different data types. See
the Notes Section in fetch_openml's API doc for details.
  warn(
```

[ ]:
```
X = mnist.data / 255.0
print(X.shape)
y = mnist.target
feat_cols = [ 'pixel' + str(i) for i in range(X.shape[1]) ]
df = pd.DataFrame(X,columns=feat_cols)
df['y'] = y
df['label'] = df['y'].apply(lambda i: str(i))


df["pixel0"] = 0
df.head()
```

```
(70000, 784)
```

[ ]:

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	\
0	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

```
   pixel9  …  pixel776  pixel777  pixel778  pixel779  pixel780  pixel781  \
0     0.0  …       0.0       0.0       0.0       0.0       0.0       0.0
1     0.0  …       0.0       0.0       0.0       0.0       0.0       0.0
2     0.0  …       0.0       0.0       0.0       0.0       0.0       0.0
3     0.0  …       0.0       0.0       0.0       0.0       0.0       0.0
4     0.0  …       0.0       0.0       0.0       0.0       0.0       0.0

   pixel782  pixel783  y  label
0       0.0       0.0  5      5
1       0.0       0.0  0      0
2       0.0       0.0  4      4
3       0.0       0.0  1      1
4       0.0       0.0  9      9

[5 rows x 786 columns]
```

[ ]:
```
X, y = None, None
np.random.seed(42)
rndperm = np.random.permutation(df.shape[0])

N= 4000
df_subset = df.loc[rndperm[:N],:].copy()
data_subset = df_subset[feat_cols].values
tsne = TSNE(n_components=2, verbose=1, perplexity=5, n_iter=300)
tsne_results = tsne.fit_transform(data_subset)

df_subset['tsne-2d-one'] = tsne_results[:,0]
df_subset['tsne-2d-two'] = tsne_results[:,1]
plt.figure(figsize=(16,10))
sb.scatterplot(
     x="tsne-2d-one", y="tsne-2d-two",
     hue="y",
     palette=sb.color_palette("hls", 10),
     data=df_subset,
     legend="full",
     alpha=0.3
)

df.head()
```

```
[t-SNE] Computing 16 nearest neighbors…
[t-SNE] Indexed 4000 samples in 0.017s…
[t-SNE] Computed neighbors for 4000 samples in 1.233s…
[t-SNE] Computed conditional probabilities for sample 1000 / 4000
[t-SNE] Computed conditional probabilities for sample 2000 / 4000
[t-SNE] Computed conditional probabilities for sample 3000 / 4000
[t-SNE] Computed conditional probabilities for sample 4000 / 4000
```

```
[t-SNE] Mean sigma: 1.512472
[t-SNE] KL divergence after 250 iterations with early exaggeration: 88.642929
[t-SNE] KL divergence after 300 iterations: 3.367960
```

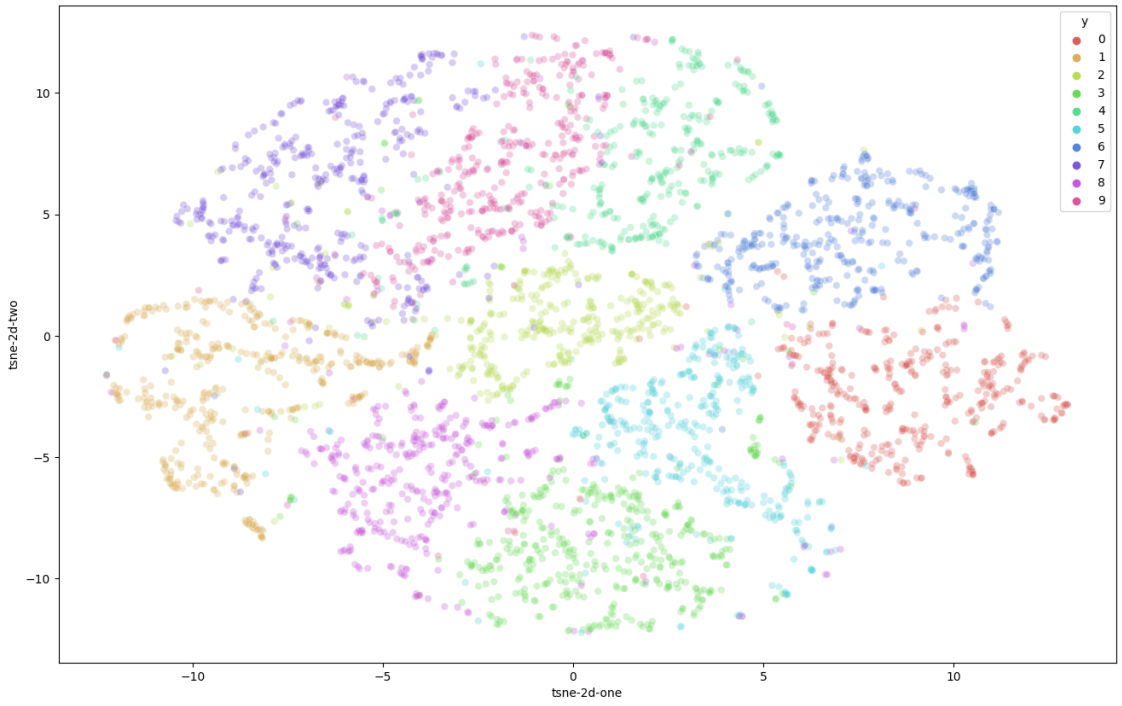
[ ]:

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	\
0	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

	pixel9	…	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781	\
0	0.0	…	0.0	0.0	0.0	0.0	0.0	0.0	
1	0.0	…	0.0	0.0	0.0	0.0	0.0	0.0	
2	0.0	…	0.0	0.0	0.0	0.0	0.0	0.0	
3	0.0	…	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.0	…	0.0	0.0	0.0	0.0	0.0	0.0	

	pixel782	pixel783	y	label
0	0.0	0.0	5	5
1	0.0	0.0	0	0
2	0.0	0.0	4	4
3	0.0	0.0	1	1
4	0.0	0.0	9	9

[5 rows x 786 columns]



[ ]:
```
from matplotlib.offsetbox import OffsetImage, AnnotationBbox

pixel_cols = df_subset.columns.str.startswith('pixel')
img_w, img_h = 28,28
zoom = 0.5

fig, ax = plt.subplots(figsize=(16,10))
for i,row in df_subset.iterrows():
    image = row[pixel_cols].values.astype(float).reshape((img_w, img_h))
    im = OffsetImage(image, zoom=zoom)
    ab = AnnotationBbox(im, (row["tsne-2d-one"], row["tsne-2d-two"]),␣
 ↪xycoords='data', frameon=False)
    ax.add_artist(ab)
    ax.update_datalim([(row["tsne-2d-one"], row["tsne-2d-two"])])
    ax.autoscale()
```