Programming for AI
(Home Work)

The goal in this assignment is to develop a machine learning model that can classify between images of T-shirts and dress-shirts. You are given the following files:

- **TrainData.csv**: It contains 12000 training examples. Each row contains 784 values. The dataset has been derived from Fashion-MNIST dataset. Each example is a flattened 28x28 pixel gray-scale image. You can reshape the examples to visualize what each image looks like.

- **TrainLabels.csv**: This file contains true labels for the examples in TrainExamples.csv (T-shirts = 1 , dress-shirts = -1)

- **TestData.csv**: This file contains test examples.

- You can load train and test data using the following code:

```
Xtr=np.loadtxt("TrainData.csv")

Ytr=np.loadtxt("TrainLabels.csv")

Xts=np.loadtxt("TestData.csv")
```

- To visualize an example (say trainining example at index 10, you can use the following code):

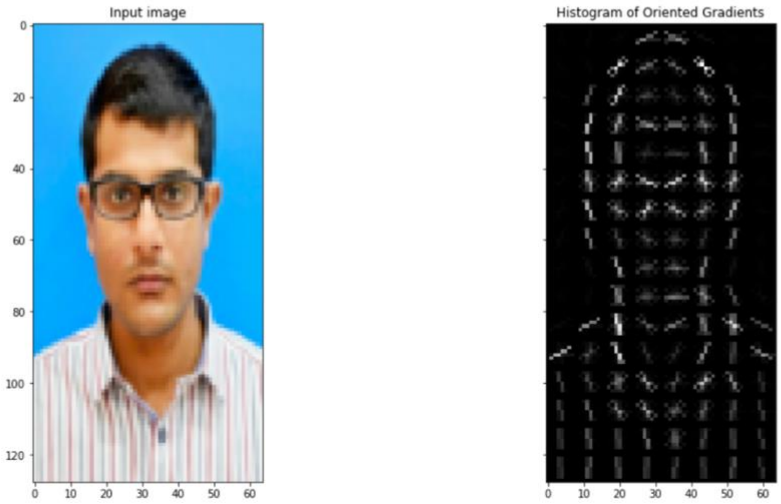
```
import matplotlib.pyplot as plt

plt.imshow(Xtr[10].reshape([28,28]))
```

Requirements:

1. You are required to extract hog features from these images for the classification help at: https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/

2. Output the extracted version of yourself in the output (If not comfortable create an image that has your name and roll number written in it under a car picture)

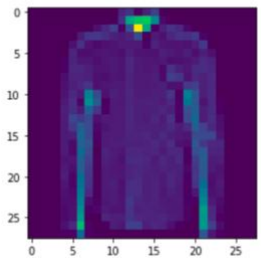
```
In [20]:
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 8), sharex=True, sharey=True)

ax1.imshow(resized_img, cmap=plt.cm.gray)
ax1.set_title('Input image')

# Rescale histogram for better display
hog_image_rescaled = exposure.rescale_intensity(hog_image, in_range=(0, 10))

ax2.imshow(hog_image_rescaled, cmap=plt.cm.gray)
ax2.set_title('Histogram of Oriented Gradients')

plt.show()
```



3. Now use the dataset provided and plot any one image from there like

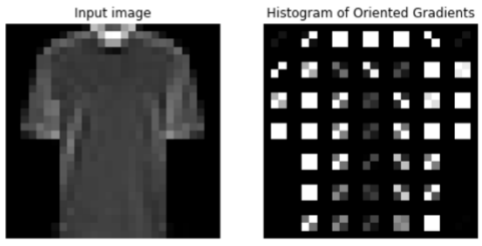
```
In [31]:
plt.imshow(Xtr[2650].reshape([28,28]))
```

```
Out[31]: <matplotlib.image.AxesImage at 0x1c2ff72710>
```



4. Next plot the original image and its extracted features

```
In [34]:
fd, hog_image = hog(new_p, orientations=8, pixels_per_cell=(4, 4),
                    cells_per_block=(1, 1), visualize=True, multichannel=None)

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(8, 4), sharex=True, sharey=True)

ax1.axis('off')
ax1.imshow(new_p, cmap=plt.cm.gray)
ax1.set_title('Input image')

# Rescale histogram for better display
hog_image_rescaled = exposure.rescale_intensity(hog_image, in_range=(0, 10))

ax2.axis('off')
ax2.imshow(hog_image_rescaled, cmap=plt.cm.gray)
ax2.set_title('Histogram of Oriented Gradients')
plt.show()
```



5. Now you have extracted features for each image and you are ready to classify them
6. Load these features and split them into 80% and 20% ratios for training and testing respectively
7. You are supposed to train a model using your extracted features. Try at least 2 different classification techniques of your choice. An example training code is given.

```
from sklearn.naive_bayes import GaussianNB model = GaussianNB()

model.fit(Xtrain, ytrain)

y_model = model.predict(Xtest)
```

8. you are required to use Decision Trees and SVM for classification.
9. Make sure you only use training data for the training purpose and the test set is used only for finding the accuracy. Never use test data while training the model that is cheating. The marks do not depend on the accuracy score.
10. Finally print the accuracy for your model using

```
In[17]: from sklearn.metrics import accuracy_score

accuracy_score(ytest, y_model)

Out[17]: 0.97368421052631582
```

Follow all the 10 points strictly in a python notebook.
Rename your notebook to “Name_RollNo.ipnyb”

Note:
DON’T CLEAR THE NOTEBOOK OUTPUTS SAVE ALL THE CELLS ALONG WITH THEIR OUTPUTS.
Failure to do so will result in Zero marks