# Data Mining FinalTerm Project

## First Name: Harini Reddy

## Last Name: Gade

## NJIT UCID: hg294

## E-Mail ID: hg294@njit.edu

### Option 1

### Data set used: https://paperswithcode.com/dataset/fashion-mnist

### Algorithms Used: Decision Tree Classifer and Random Forest Classifier

```
# import basic libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

```
# read the data
# the data consists of pixel values of 28*28 images of fashion items
data = pd.read_csv('fashion-mnist_train.csv')
data.head()
```

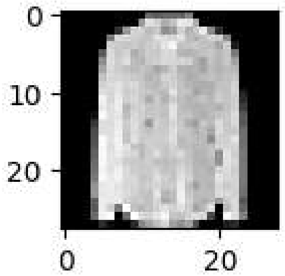
|   | label | pixel1 | pixel2 | pixel3 | pixel4 | pixel5 | pixel6 | pixel7 | pixel8 | pixel9 | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... |
| **1** | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... |
| **2** | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | ... |
| **3** | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 0 | 0 | 0 | ... |
| **4** | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... |

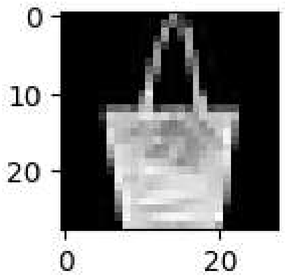
5 rows × 785 columns

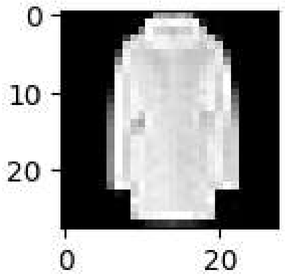
```
# different types of fashion items
data.label.unique()
# there are 10 different types of fashion items
```

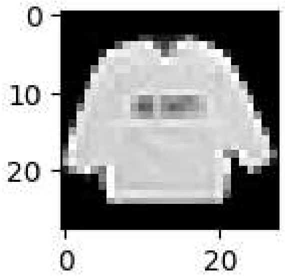
```
     array([2, 9, 6, 0, 3, 4, 5, 8, 7, 1], dtype=int64)
```

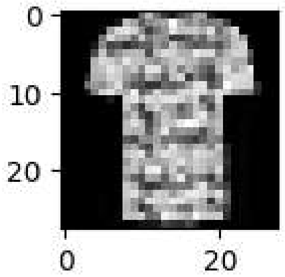
```
# print image of different fashion items
# the pixel values are in the range of 0 to 255
# 0 means black and 255 means white
k = 0
for i in data.label.unique():
    plt.subplot(3,4,k+1)
    plt.imshow(data.iloc[i,1:].values.reshape(28,28),cmap='gray')
    plt.show()
```

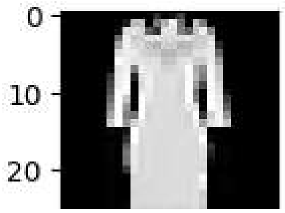












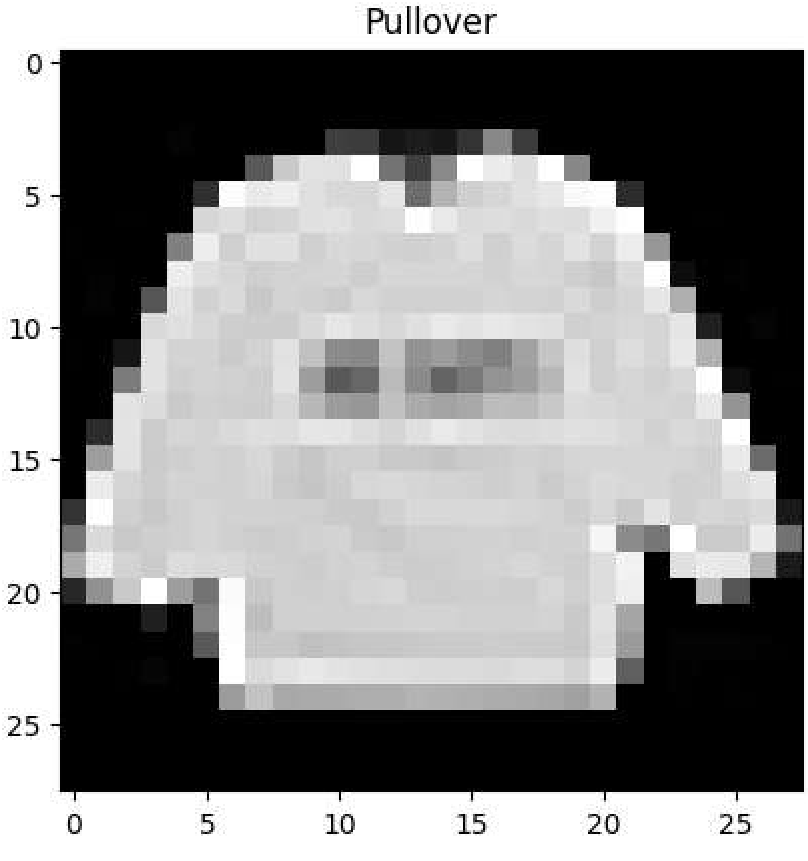
```
# let's label the fashion items
fashion_items = {0:'T-shirt/top',1:'Trouser',2:'Pullover',3:'Dress',4:'Coat',5:'Sandal',6:'Sh
```

```
# let's see the distribution of different fashion items
data.label.value_counts()
# the data is balanced
```

```
     2    6000
     9    6000
     6    6000
     0    6000
     3    6000
     4    6000
     5    6000
     8    6000
     7    6000
     1    6000
     Name: label, dtype: int64
```

```
# let's create a function to show the image of fashion item and its label
def show_image(df, index):
    item = df.iloc[index, 1:].values.reshape(28,28)
    label = df.iloc[index, 0]
    plt.title(fashion_items[label])
    plt.imshow(item, cmap='gray')
    plt.show()

show_image(data, 0)
```



```
# since the pixel values are in the range of 0 to 255, let's scale them to 0 to 1
```

```
data.iloc[:,1:] = data.iloc[:,1:]/255
data.head()
```

|   | label | pixel1 | pixel2 | pixel3 | pixel4 | pixel5 | pixel6 | pixel7 | pixel8 | pixel9 | . |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.000000 | 0.0 |  |
| **1** | 9 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.000000 | 0.0 |  |
| **2** | 6 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.019608 | 0.0 |  |
| **3** | 0 | 0.0 | 0.0 | 0.0 | 0.003922 | 0.007843 | 0.0 | 0.0 | 0.000000 | 0.0 |  |
| **4** | 3 | 0.0 | 0.0 | 0.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.000000 | 0.0 |  |

5 rows × 785 columns

```
# let's split the data into train and test
X = data.drop('label', axis=1)
y = data.label
```

```
# using 10-fold cross validation
from sklearn.model_selection import KFold
kf = KFold(n_splits=10, shuffle=True, random_state=42)

# let's create a function to calculate the accuracy
def accuracy(y_true, y_pred):
    return np.sum(y_true == y_pred)/len(y_true)

# let's create a function to plot the confusion matrix
def plot_confusion_matrix(cm):
    plt.figure(figsize=(10,10))
    sns.heatmap(cm, annot=True, fmt='.0f', cmap='Blues')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()

# let's create a function to plot the classification report
from sklearn.metrics import classification_report
def plot_classification_report(y_true, y_pred):
    print(classification_report(y_true, y_pred))
```

## Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier
dtc = DecisionTreeClassifier(max_depth=50)
```

```
# let's train the model
```

```
from sklearn.metrics import confusion_matrix
scores = []
cm = []
report = []
for train_index, test_index in kf.split(X):
    X_train, X_test, y_train, y_test = X.iloc[train_index], X.iloc[test_index], y.iloc[train_
    dtc.fit(X_train, y_train)
    y_pred = dtc.predict(X_test)
    scores.append(accuracy(y_test, y_pred))
    cm.append(confusion_matrix(y_test, y_pred))
```

```
scores
```

```
[0.8048333333333333,
 0.7936666666666666,
 0.789,
 0.7991666666666667,
 0.7928333333333333,
 0.8013333333333333,
 0.7973333333333333,
 0.7896666666666666,
 0.8015,
 0.7861666666666667]
```

## Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=20, max_depth=10)
```

```
rfcscores = []
for train_index, test_index in kf.split(X):
    X_train, X_test, y_train, y_test = X.iloc[train_index], X.iloc[test_index], y.iloc[train_
    rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    rfcscores.append(accuracy(y_test, y_pred))
```

```
rfcscores
```

```
[0.846,
 0.8475,
 0.8495,
 0.8446666666666667,
 0.8503333333333334,
 0.8455,
 0.8516666666666667,
 0.8451666666666666,
 0.8476666666666667,
 0.8521666666666666]
```