导入数据

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) w
ill list all files under the input directory

import os
for dirname, _, filenames in os.walk('D:/学习/大三上/机器学习/fashion min
st'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
import time
import xgboost
%matplotlib inline
#读取数据
fashion_mnist_test = pd.read_csv("D:/学习/大三上/机器学习/fashion minst/f
ashion-mnist_test.csv")
fashion_mnist_train = pd.read_csv("D:/学习/大三上/机器学习/fashion minst/
fashion-mnist_train.csv")
```

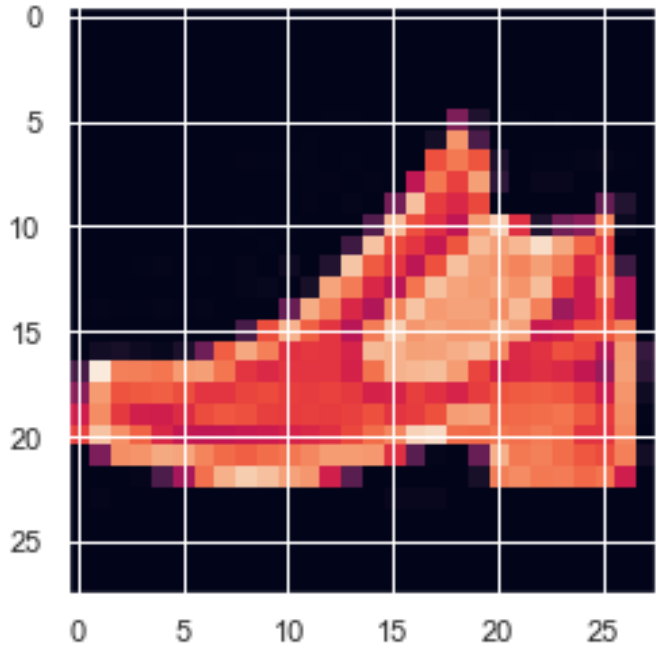
```
D:/学习/大三上/机器学习/fashion minst\fashion minst.py
D:/学习/大三上/机器学习/fashion minst\fashion-mnist_test.csv
D:/学习/大三上/机器学习/fashion minst\fashion-mnist_train.csv
D:/学习/大三上/机器学习/fashion minst\t10k-images-idx3-ubyte
D:/学习/大三上/机器学习/fashion minst\t10k-labels-idx1-ubyte
D:/学习/大三上/机器学习/fashion minst\train-images-idx3-ubyte
D:/学习/大三上/机器学习/fashion minst\train-labels-idx1-ubyte
D:/学习/大三上/机器学习/fashion minst\一个可能可以用的程序.py
D:/学习/大三上/机器学习/fashion minst\archive\fashion-mnist_test.csv
D:/学习/大三上/机器学习/fashion minst\archive\fashion-mnist_train.csv
D:/学习/大三上/机器学习/fashion minst\archive\t10k-images-idx3-ubyte
D:/学习/大三上/机器学习/fashion minst\archive\t10k-labels-idx1-ubyte
D:/学习/大三上/机器学习/fashion minst\archive\train-images-idx3-ubyte
D:/学习/大三上/机器学习/fashion minst\archive\train-labels-idx1-ubyte
```

数据分析或处理

提取数组 查看数据大小 图像化展示

```
#利用np.array 提出数组
x_train = np.array(fashion_mnist_train.iloc[:,1:])
y_train = np.array(fashion_mnist_train.iloc[:,0])
x_test = np.array(fashion_mnist_test.iloc[:,1:])
y_test = np.array(fashion_mnist_test.iloc[:,0])
print(x_train.shape, type(x_train))
print(y_train.shape, type(y_train))
print(x_test.shape, type(x_train))
print(y_test.shape, type(y_train))
plt.imshow(x_train[1].reshape((28,28)))
plt.show()
```

```
(60000, 784) <class 'numpy.ndarray'>
(60000,) <class 'numpy.ndarray'>
(10000, 784) <class 'numpy.ndarray'>
(10000,) <class 'numpy.ndarray'>
```



png

采用的算法或结构

利用 xgboost 算法

其中参数设置为 gamma=0，max_depth=6，min_child_weight=0.9，n_estimators=500，eta=0.5

```
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, f1_score,
 precision_score, recall_score
from sklearn.metrics import classification_report
```

```
start = time.time()
xgb_clf = XGBClassifier(n_estimators=500, n_jobs=-1, learning_rate=0.5,
 max_deth= 6, min_child_weight= 0.9, seed=0)
xgb_clf.fit(x_train,y_train,eval_metric='mlogloss', early_stopping_roun
ds = 50, eval_set = [(x_test, y_test)])
stop = time.time()
print(xgb_clf.get_params)
print(f"Training time: {stop - start}s")
y_pred = xgb_clf.predict(x_test)
print("test accuracy")
print(accuracy_score(y_test, y_pred))
```

```
C:\Users\Lenovo\Anaconda3\lib\site-packages\xgboost\sklearn.py:1224: Us
erWarning: The use of label encoder in XGBClassifier is deprecated and
will be removed in a future release. To remove this warning, do the fol
lowing: 1) Pass option use_label_encoder=False when constructing XGBCla
ssifier object; and 2) Encode your labels (y) as integers starting with
 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)


[13:46:13] WARNING: C:\Windows\Temp\abs_557yfx631l\croots\recipe\xgboos
t-split_1659548953302\work\src\learner.cc:576:
Parameters: { "max_deth" } might not be used.

  This could be a false alarm, with some parameters getting used by lan
guage bindings but
  then being mistakenly passed down to XGBoost core, or some parameter
actually being used
  but getting flagged wrongly here. Please open an issue if you find an
y such cases.


[0] validation_0-mlogloss:1.01037
[1] validation_0-mlogloss:0.78003
[2] validation_0-mlogloss:0.64309
[3] validation_0-mlogloss:0.55457
[4] validation_0-mlogloss:0.49499
[5] validation_0-mlogloss:0.45017
[6] validation_0-mlogloss:0.41872
[7] validation_0-mlogloss:0.39678
[8] validation_0-mlogloss:0.37983
[9] validation_0-mlogloss:0.36622
[10]    validation_0-mlogloss:0.35392
[11]    validation_0-mlogloss:0.34477
[12]    validation_0-mlogloss:0.33597
[13]    validation_0-mlogloss:0.32884
[14]    validation_0-mlogloss:0.32275
[15]    validation_0-mlogloss:0.31803
[16]    validation_0-mlogloss:0.31372
```

```
[17]    validation_0-mlogloss:0.30818
[18]    validation_0-mlogloss:0.30420
[19]    validation_0-mlogloss:0.30139
[20]    validation_0-mlogloss:0.29936
[21]    validation_0-mlogloss:0.29745
[22]    validation_0-mlogloss:0.29524
[23]    validation_0-mlogloss:0.29351
[24]    validation_0-mlogloss:0.29193
[25]    validation_0-mlogloss:0.29007
[26]    validation_0-mlogloss:0.28835
[27]    validation_0-mlogloss:0.28690
[28]    validation_0-mlogloss:0.28546
[29]    validation_0-mlogloss:0.28434
[30]    validation_0-mlogloss:0.28293
[31]    validation_0-mlogloss:0.28210
[32]    validation_0-mlogloss:0.28124
[33]    validation_0-mlogloss:0.28036
[34]    validation_0-mlogloss:0.27935
[35]    validation_0-mlogloss:0.27860
[36]    validation_0-mlogloss:0.27793
[37]    validation_0-mlogloss:0.27699
[38]    validation_0-mlogloss:0.27692
[39]    validation_0-mlogloss:0.27672
[40]    validation_0-mlogloss:0.27603
[41]    validation_0-mlogloss:0.27648
[42]    validation_0-mlogloss:0.27573
[43]    validation_0-mlogloss:0.27573
[44]    validation_0-mlogloss:0.27552
[45]    validation_0-mlogloss:0.27514
[46]    validation_0-mlogloss:0.27474
[47]    validation_0-mlogloss:0.27466
[48]    validation_0-mlogloss:0.27448
[49]    validation_0-mlogloss:0.27370
[50]    validation_0-mlogloss:0.27312
[51]    validation_0-mlogloss:0.27345
[52]    validation_0-mlogloss:0.27351
[53]    validation_0-mlogloss:0.27376
[54]    validation_0-mlogloss:0.27302
[55]    validation_0-mlogloss:0.27297
[56]    validation_0-mlogloss:0.27314
[57]    validation_0-mlogloss:0.27331
[58]    validation_0-mlogloss:0.27264
[59]    validation_0-mlogloss:0.27278
[60]    validation_0-mlogloss:0.27271
[61]    validation_0-mlogloss:0.27233
[62]    validation_0-mlogloss:0.27291
[63]    validation_0-mlogloss:0.27299
[64]    validation_0-mlogloss:0.27285
[65]    validation_0-mlogloss:0.27299
[66]    validation_0-mlogloss:0.27305
```

```
[67]    validation_0-mlogloss:0.27277
[68]    validation_0-mlogloss:0.27306
[69]    validation_0-mlogloss:0.27321
[70]    validation_0-mlogloss:0.27331
[71]    validation_0-mlogloss:0.27349
[72]    validation_0-mlogloss:0.27365
[73]    validation_0-mlogloss:0.27362
[74]    validation_0-mlogloss:0.27371
[75]    validation_0-mlogloss:0.27391
[76]    validation_0-mlogloss:0.27380
[77]    validation_0-mlogloss:0.27371
[78]    validation_0-mlogloss:0.27374
[79]    validation_0-mlogloss:0.27381
[80]    validation_0-mlogloss:0.27400
[81]    validation_0-mlogloss:0.27401
[82]    validation_0-mlogloss:0.27388
[83]    validation_0-mlogloss:0.27400
[84]    validation_0-mlogloss:0.27455
[85]    validation_0-mlogloss:0.27434
[86]    validation_0-mlogloss:0.27434
[87]    validation_0-mlogloss:0.27444
[88]    validation_0-mlogloss:0.27475
[89]    validation_0-mlogloss:0.27496
[90]    validation_0-mlogloss:0.27528
[91]    validation_0-mlogloss:0.27540
[92]    validation_0-mlogloss:0.27607
[93]    validation_0-mlogloss:0.27643
[94]    validation_0-mlogloss:0.27643
[95]    validation_0-mlogloss:0.27645
[96]    validation_0-mlogloss:0.27665
[97]    validation_0-mlogloss:0.27653
[98]    validation_0-mlogloss:0.27703
[99]    validation_0-mlogloss:0.27703
[100]   validation_0-mlogloss:0.27750
[101]   validation_0-mlogloss:0.27774
[102]   validation_0-mlogloss:0.27785
[103]   validation_0-mlogloss:0.27779
[104]   validation_0-mlogloss:0.27844
[105]   validation_0-mlogloss:0.27886
[106]   validation_0-mlogloss:0.27926
[107]   validation_0-mlogloss:0.27967
[108]   validation_0-mlogloss:0.27951
[109]   validation_0-mlogloss:0.27962
[110]   validation_0-mlogloss:0.27993
[111]   validation_0-mlogloss:0.28013
<bound method XGBModel.get_params of XGBClassifier(base_score=0.5, boos
ter='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, enable_categorica
l=False,
              gamma=0, gpu_id=-1, importance_type=None,
```

```
              interaction_constraints='', learning_rate=0.5, max_delta_
step=0,
              max_depth=6, max_deth=8, min_child_weight=0.9, missing=na
n,
              monotone_constraints='()', n_estimators=500, n_jobs=-1,
              num_parallel_tree=1, objective='multi:softprob', predicto
r='auto',
              random_state=0, reg_alpha=0, reg_lambda=1, scale_pos_weig
ht=None,
              seed=0, subsample=1, tree_method='exact', use_label_encod
er=True, ...)>
Training time: 3239.727637529373s
test accuracy
0.9061

print(os.path.abspath('.'))

C:\Users\Lenovo
```