Lab 11 - CCPS 844 Data Mining

Salman AlMaskati

Answer the following questions and submit a PDF file on the D2L.

Q-1 Select a dataset/datasets of your choice.

- Apply/Fit MLP Classification
- Evaluate the results

Q-2 Select a multi-label dataset/datasets of your choice.

- Apply/Fit MLP Classification
- Call the predict function to get a multi label value for your test data

Q-1

In [ ]:
```
import pandas as pd

df=pd.read_csv("Social_Network_Ads.csv")
df
```

Out[ ]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

In [ ]:
```
from sklearn.neural_network import MLPClassifier
#get dummies for col 'Gender'
final_df = pd.get_dummies(df, columns=['Gender'])
feature_col=['Gender_Female','Gender_Male','Age','EstimatedSalary']
```

```
X=final_df[feature_col]
y=final_df['Purchased']

X
```

Out[ ]:

	Gender_Female	Gender_Male	Age	EstimatedSalary
0	False	True	19	19000
1	False	True	35	20000
2	True	False	26	43000
3	True	False	27	57000
4	False	True	19	76000
...	...	...	...	...
395	True	False	46	41000
396	False	True	51	23000
397	True	False	50	20000
398	False	True	36	33000
399	True	False	49	36000

400 rows × 4 columns

In [ ]:
```
#split data
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=4)
```

In [ ]:
```
#MLPClassifier object

clf = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2), rand
```

In [ ]:
```
clf.fit(X_train,y_train)
```

```
/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-package
s/sklearn/neural_network/_multilayer_perceptron.py:541: ConvergenceWarning: lb
fgs failed to converge (status=2):
ABNORMAL_TERMINATION_IN_LNSRCH.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_iter)
```

Out[ ]:

▾ MLPClassifier

```
MLPClassifier(alpha=1e-05, hidden_layer_sizes=(5, 2), random_state=1,
              solver='lbfgs')
```

In [ ]:
```
#evaluate accuracy
from sklearn import metrics
y_pred=clf.predict(X_test)
print(metrics.accuracy_score(y_test, y_pred))
```

```
0.3
```

In [ ]:
```
[coef.shape for coef in clf.coefs_]
```

Out[ ]:
```
[(4, 5), (5, 2), (2, 1)]
```

In [ ]:
```
clf.coefs_
```

Out[ ]:
```
[array([[-0.1355025 ,  0.35978839, -0.81630981, -0.32278956, -0.57684521],
        [-0.66570829, -0.51233493, -0.25219828, -0.16857801,  0.06338746],
        [-0.13195481,  0.30246218, -0.48262746,  0.61746319, -0.77177283],
        [ 0.27837228, -0.13504069,  0.09584009, -0.58724567, -0.4929982 ]]),
 array([[ 0.73066956, -0.76834882],
        [-0.85350469, -0.61135527],
        [ 0.70018386, -0.74371715],
        [-0.14608029,  0.84784666],
        [ 0.06141017,  0.35528738]]),
 array([[ 0.94646433],
        [-1.3624865 ]])]
```

In [ ]:
```
import numpy as np
pred = np.array([[19, 1, 0,10000], [25, 0, 1,10000]])
clf.predict_proba(pred)
```

```
/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but MLPClassifier was fitted with feature names
  warnings.warn(
```

Out[ ]:
```
array([[0., 1.],
       [0., 1.]])
```

Q-2

In [ ]:
```
from sklearn.datasets import make_multilabel_classification
X, y = make_multilabel_classification(n_samples=5, n_features=10, n_classes=3,

# convert into a DataFrame
df_X = pd.DataFrame(X)
df_y = pd.DataFrame(y)

# combine X and y
df = pd.concat([df_X, df_y], axis=1)

df
```

Out[ ]:

	0	1	2	3	4	5	6	7	8	9	0	1	2
0	3.0	3.0	6.0	7.0	8.0	2.0	11.0	11.0	1.0	3.0	1	1	0
1	7.0	6.0	4.0	4.0	6.0	8.0	3.0	4.0	6.0	4.0	0	0	0
2	5.0	5.0	13.0	7.0	6.0	3.0	6.0	11.0	4.0	2.0	1	1	0
3	1.0	1.0	5.0	5.0	7.0	3.0	4.0	6.0	4.0	4.0	1	1	1
4	4.0	2.0	3.0	13.0	7.0	2.0	4.0	12.0	1.0	7.0	0	1	0

In [ ]:
```
clf = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(15,), randc
clf.fit(X, y)
```

Out[ ]:
```
MLPClassifier
MLPClassifier(alpha=1e-05, hidden_layer_sizes=(15,), random_state=1,
              solver='lbfgs')
```

In [ ]:
```
clf.predict([[3.0,3.0,6.0,7.0,8.0,2.0,11.0,11.0,1.0,3.0]])
```

Out[ ]:
```
array([[1, 1, 0]])
```

In [ ]:
```
clf.predict([[5,4,3,2,6,7,8,9,0,1]])
```

Out[ ]:
```
array([[0, 1, 0]])
```

In [ ]:
```

```