Tutorial 8: ID5055 Foundations of Machine Learning

Topics Covered: Linear Discriminant Analysis, Logistic Regression

Questions

1. As part of this tutorial you are supposed to use LDA and LR implementations in `sklearn` library and then carry out model fitting on datasets 5 and 6 as provided below. Both the datasets have already been split into train and test sets (can be done by running corresponding cells). On each of these datasets, fit both models on the train set and report classification statistics on the test set using `sklearn.metrics.classification_report`.

2. **(Optional)** Using the `mpl_toolkits.mplot3d` module attempt to plot a decision boundary for 3d data.

3. **(Optional)** Using the `sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis` function attempt to do QDA on the 2D datasets and visualize the decision boundary for the same.

Write your code in the cells mentioned. Please ensure your notebook runs correctly without errors when using **Kernel -> Restart & Run All**

```
import math
import random
import sklearn
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from mpl_toolkits.mplot3d import axes3d
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.inspection import DecisionBoundaryDisplay
from sklearn.datasets import make_blobs, make_circles

### DO NOT EDIT ###
# set seeds
seed = 0
random.seed = 0
```

```
np.random.seed = 0
sns.set_style('darkgrid')
```

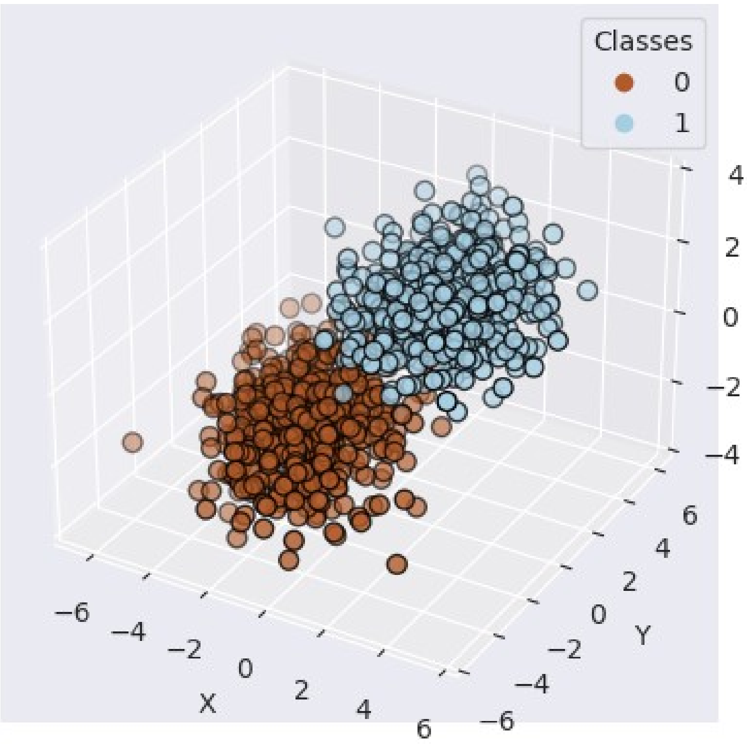
Dataset 5: 3D Data - Same Covariance

In this section, you may generate the dataset by running the cell below. Use the train and test datasets as split here. Fit both LR and LDA models on both datasets and print classification statistics on the **test set** using the `sklearn.metrics.classification_report` function.

```
means5 = [[-1.5, -1.5, -1.5], [1.5, 1.5, 1.5]]
covs5 = [
    [[2, 0, 0], [0, 1.8660254, 0.5], [0, 0.5, 0.5]],
    [[2, 0, 0], [0, 1.8660254, 0.5], [0, 0.5, 0.5]]
]
num = 500  # datapoints per class

X5 = np.concatenate([
    np.random.multivariate_normal(means5[0], covs5[0], size=(num,)),
    np.random.multivariate_normal(means5[1], covs5[1], size=(num,))
], axis=0)
y5 = np.concatenate([[0] * num, [1] * num], axis=0)
X5_train, X5_test, Y5_train, Y5_test = train_test_split(X5, y5,
test_size = 0.2,
                            random_state = seed, shuffle = True)

# visualizing the dataset
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
scatter5 = ax.scatter(X5[:, 0], X5[:, 1], X5[:, 2], s = 50, c = y5,
cmap = 'Paired_r', edgecolor='k')
legend5 = ax.legend(*scatter5.legend_elements(), title="Classes")
plt.title('Dataset 5: 3D data with same covariance')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.show()
```


Dataset 5: 3D data with same covariance

```
### WRITE CODE HERE ###
# Fit both LR and LDA on the dataset 5
# report all classification metrics on the TEST SET

lr5 = LogisticRegression(penalty = None, solver = "lbfgs", multi_class
= "multinomial", random_state = seed)
lda5 = LinearDiscriminantAnalysis()

lr5.fit(X5_train, Y5_train)
lda5.fit(X5_train, Y5_train)

Y5_lr5_pred = lr5.predict(X5_test)
Y5_lda5_pred = lda5.predict(X5_test)

print(
    f'Classification Report of Logistic Regression\n'
    f'{classification_report(Y5_test, Y5_lr5_pred)}'
)
```

```
Classification Report of Logistic Regression
              precision    recall  f1-score   support

           0       1.00      0.99      0.99        98
           1       0.99      1.00      1.00       102

    accuracy                           0.99       200
```

```
   macro avg       1.00      0.99      0.99       200
weighted avg       1.00      0.99      0.99       200
```

```
print(
    f'Classification Report of Linear discriminant analysis\n'
    f'{classification_report(Y5_test, Y5_lda5_pred)}'
)
```

```
Classification Report of Linear discriminant analysis
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        98
           1       1.00      1.00      1.00       102

    accuracy                           1.00       200
   macro avg       1.00      1.00      1.00       200
weighted avg       1.00      1.00      1.00       200
```

Dataset 6: 3D Data - Different Covariance

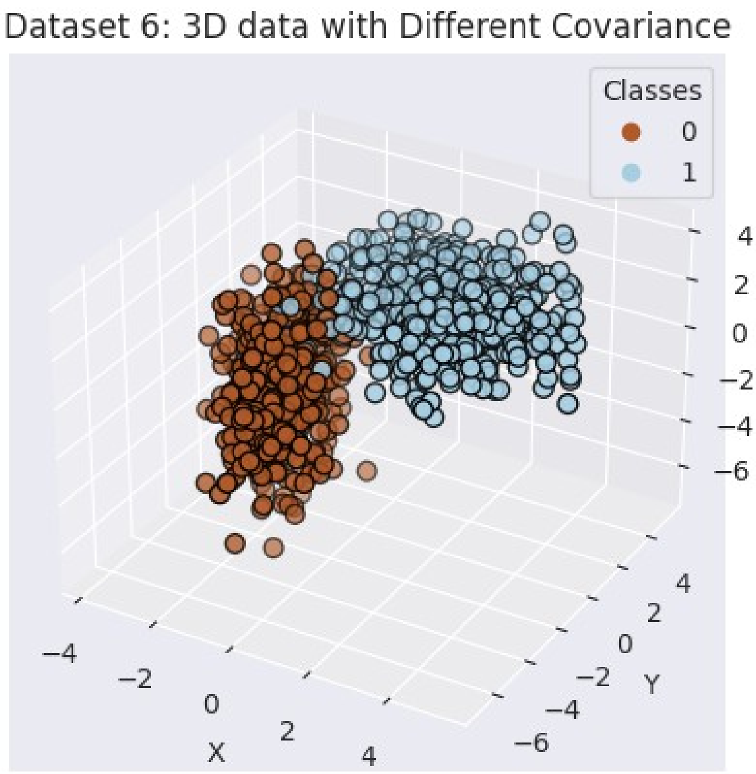
In this section, you may generate the dataset by running the cell below. Use the train and test datasets as split here. Fit both LR and LDA models on both datasets and print classification statistics on the **test set** using the `sklearn.metrics.classification_report` function.

```
means6 = [[-1.5, -1.5, -1.5], [1.5, 1.5, 1.5]]
covs6 = [
    [[0.5, -0.5, 0], [-0.5, 2.5, 0], [0, 0, 3]],
    [[2, 0, 0], [0, 1.8660254, 0.5], [0, 0.5, 0.5]]
]
num = 500  # datapoints per class

X6 = np.concatenate([
    np.random.multivariate_normal(means6[0], covs6[0], size=(num,)),
    np.random.multivariate_normal(means6[1], covs6[1], size=(num,))
], axis=0)
y6 = np.concatenate([[0] * num, [1] * num], axis=0)
X6_train, X6_test, Y6_train, Y6_test = train_test_split(X6, y6,
test_size = 0.2,
                                                        random_state = seed, shuffle = True)

# visualizing the dataset
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
scatter6 = ax.scatter(X6[:, 0], X6[:, 1], X6[:, 2], s = 50, c = y6,
cmap = 'Paired_r', edgecolor='k')
legend6 = ax.legend(*scatter5.legend_elements(), title="Classes")
plt.title('Dataset 6: 3D data with Different Covariance')
```

```
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.show()
```


Dataset 6: 3D data with Different Covariance

```
### WRITE CODE HERE ###
# Fit both LR and LDA on the dataset 6
# report all classification metrics on the TEST SET

lr6 = LogisticRegression(penalty = None, solver = "lbfgs", multi_class
= "multinomial", random_state = seed)
lda6 = LinearDiscriminantAnalysis()

lr6.fit(X6_train, Y6_train)
lda6.fit(X6_train, Y6_train)

Y6_lr6_pred = lr5.predict(X6_test)
Y6_lda6_pred = lda5.predict(X6_test)

print(
    f'Classification Report of Logistic Regression\n'
    f'{classification_report(Y6_test, Y6_lr6_pred)}'
)
```

```
Classification Report of Logistic Regression
              precision    recall  f1-score   support

           0       0.97      0.79      0.87        98
           1       0.83      0.98      0.90       102

    accuracy                           0.89       200
   macro avg       0.90      0.88      0.88       200
weighted avg       0.90      0.89      0.88       200
```

```
print(
    f'Classification Report of Linear discriminant analysis\n'
    f'{classification_report(Y6_test, Y6_lda6_pred)}'
)
```

```
Classification Report of Linear discriminant analysis
              precision    recall  f1-score   support

           0       0.97      0.79      0.87        98
           1       0.83      0.98      0.90       102

    accuracy                           0.89       200
   macro avg       0.90      0.88      0.88       200
weighted avg       0.90      0.89      0.88       200
```