Exercises for the course
Machine Learning for Data Science
Winter Semester 2022/23

G. Montavon
Institute of Computer Science
Department of Mathematics and Computer Science
Freie Universität Berlin

Exercise Sheet 7 (programming part)

Exercise 3: Implementing the Fisher Discriminant (25 + 25 P)

We would like to implement and test the Fisher Linear Discriminant on the Breast Cancer Dataset available in sklearn. The following code loads the dataset, standardizes the input features, and stores points of the two classes in the variables `XM` and `XB` (for malignant and benign).

In [1]:

```
%matplotlib inline

import matplotlib.pyplot as plt
import numpy,sklearn,sklearn.datasets
X,T = sklearn.datasets.load_breast_cancer(return_X_y=True)
X = (X - X.mean(axis=0)) / X.std(axis=0)
XM,XB = X[T==0],X[T==1]
```

(a) Create a function `w = fisher(X1,X2)` that takes as input the data for the two classes (two numpy arrays of size $N_1 \times d$ and $N_2 \times d$ respectively) and returns the Fisher linear discriminant.

In [38]:

```
def fisher(X1,X2):
    mu_1 = X1.mean(axis=0, keepdims=True)
    mu_2 = X2.mean(axis=0, keepdims=True)
    S_1 = (X1 - mu_1).T @ (X1 - mu_1)
    S_2 = (X2 - mu_2).T @ (X2 - mu_2)

    w = numpy.linalg.pinv(S_1 + S_2) @ (mu_1 - mu_2).T
    return w / numpy.linalg.norm(w)
```

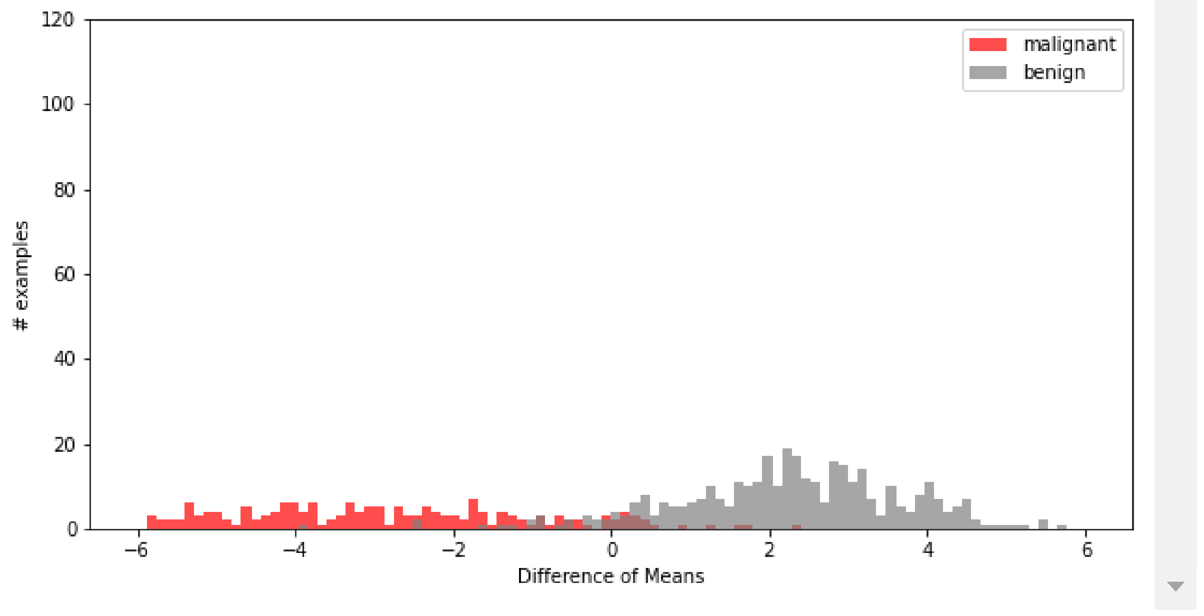
(b) Plot the histogram of projected data for several projection vectors `w`, in particular

1. `w` is the difference between the mean vectors of the two classes
2. `w` is the Fisher discriminant, and
3. `w` is the weight vector of a linear SVM (with hyperparameter C=10.0).

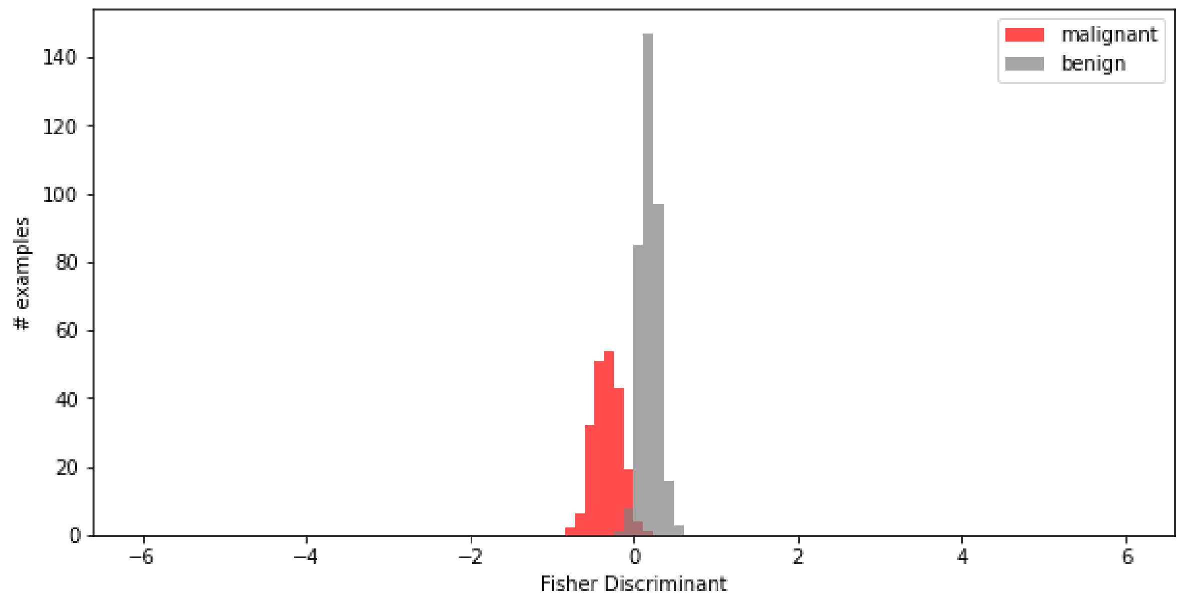
For the SVM, you can make use of the sklearn implementation (`sklearn.svm.SVC`). Before applying the projection and plotting, ensure that `w` is rescaled to have norm 1 in each case. When plotting, use fixed axis ranges and bin sizes in order to ease comparison.

In [58]:

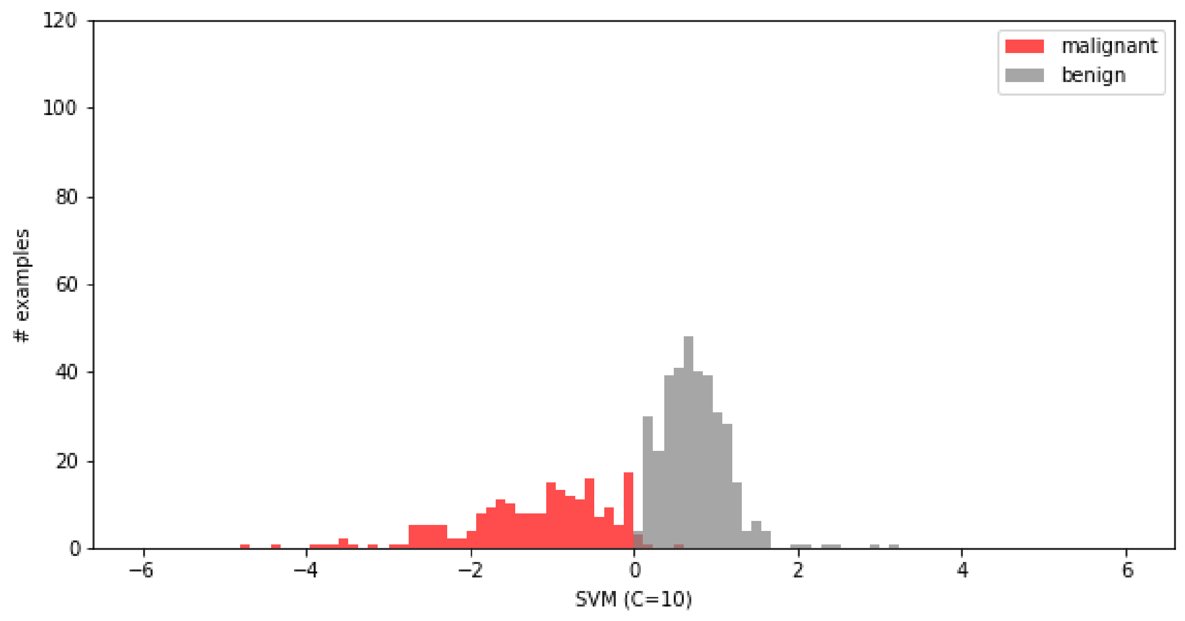
```python
def mean_difference(X1,X2):
    mu_1 = X1.mean(axis=0, keepdims=True)
    mu_2 = X2.mean(axis=0, keepdims=True)
    w = (mu_1 - mu_2).T
    return w / numpy.linalg.norm(w)

def svc_weight(X1,X2):
    my_svc = sklearn.svm.SVC(C=10.0, kernel='linear')
    X_svc = numpy.vstack((X1,X2))
    y_svc = numpy.asarray([0,] * X1.shape[0] + [1,] * X2.shape[0])
    my_svc.fit(X_svc,y_svc)
    w = -my_svc.coef_.T / numpy.linalg.norm(my_svc.coef_.T)
    return w

# Plotting the difference of means
w = mean_difference(XB, XM)
plt.hist(XM @ w, bins=100, range=(-6,6), fc=(1, 0, 0, 0.7), label='malignant')
plt.hist(XB @ w, bins=100, range=(-6,6), fc=(0.5, 0.5, 0.5, 0.7), label='benign')
plt.ylim((0,120))
plt.legend()
plt.xlabel("Difference of Means")
plt.ylabel("# examples")
plt.rcParams["figure.figsize"] = (10, 5)
plt.show()

# Plotting the Fisher discriminant
w = fisher(XB, XM)
plt.hist(XM @ w, bins=100, range=(-6,6), fc=(1, 0, 0, 0.7), label='malignant')
plt.hist(XB @ w, bins=100, range=(-6,6), fc=(0.5, 0.5, 0.5, 0.7), label='benign')
plt.legend()
plt.xlabel("Fisher Discriminant")
plt.ylabel("# examples")
plt.rcParams["figure.figsize"] = (10, 5)
plt.show()

# Plotting the weights of a linear SVM
w = svc_weight(XB,XM)
plt.hist(XM @ w, bins=100, range=(-6,6), fc=(1, 0, 0, 0.7), label='malignant')
plt.hist(XB @ w, bins=100, range=(-6,6), fc=(0.5, 0.5, 0.5, 0.7), label='benign')
plt.ylim((0,120))
plt.legend()
plt.xlabel("SVM (C=10)")
plt.ylabel("# examples")
plt.rcParams["figure.figsize"] = (10, 5)
plt.show()
```



# examples
Difference of Means
malignant
benign



# examples
Fisher Discriminant
malignant
benign



# examples
SVM (C=10)
malignant
benign

We observe that different discriminants extract different aspects of the data. For example, the difference of means algorithm produces the highest average distance between classes. The Fisher vector produces rather small distance between classes but also very low variance within classes. Lastly, the SVM focuses on achieving a low overlap between the two classes through learning a separating hyperplane.