```python
"""
Example 4.2 1D Decision boundary for the Iris dataset
@author: austin_downey
"""

import IPython as IP
IP.get_ipython().run_line_magic('reset', '-sf')

import numpy as np
import matplotlib.pyplot as plt
import sklearn as sk


cc = plt.rcParams['axes.prop_cycle'].by_key()['color']
plt.close('all')


#%% Load your data

# We will use the Iris data set.  This dataset was created by biologist Ronald
# Fisher in his 1936 paper "The use of multiple measurements in taxonomic
# problems" as an example of linear discriminant analysis

iris = sk.datasets.load_iris()

# for simplicity, extract some of the data sets
X = iris['data'] # this contains the length of the pedals and sepals
Y = iris['target'] # contains what type of flower it is
Y_names = iris['target_names'] # contains the name that aligns with the type of the flower
feature_names = iris['feature_names'] # the names of the features

# plot the Sepal data
plt.figure(figsize=(6.5,3))
plt.subplot(121)
plt.grid(True)
plt.scatter(X[Y==0,0],X[Y==0,1],marker='o')
plt.scatter(X[Y==1,0],X[Y==1,1],marker='s')
plt.scatter(X[Y==2,0],X[Y==2,1],marker='d')
plt.xlabel(feature_names[0])
plt.ylabel(feature_names[1])


plt.subplot(122)
plt.grid(True)
plt.scatter(X[Y==0,2],X[Y==0,3],marker='o',label=Y_names[0])
plt.scatter(X[Y==1,2],X[Y==1,3],marker='s',label=Y_names[1])
plt.scatter(X[Y==2,2],X[Y==2,3],marker='d',label=Y_names[2])
plt.xlabel(feature_names[2])
plt.ylabel(feature_names[3])
plt.legend(framealpha=1)
plt.tight_layout()


#%% Train a Logistic Regression model

# define the features (X) and the output (Y)
X_pedal = iris["data"][:, 3:] # consider just the petal width
y_pedal = iris["target"] == 2 # 1 if Iris-Virginica, else 0

# Build the logistic Regression model and train it.
log_reg = sk.linear_model.LogisticRegression( C=1)
log_reg.fit(X_pedal, y_pedal)
# Note: The hyper-parameter controlling the regularization strength of a
# Scikit-Learn LogisticRegression model is not alpha (as in other linear models),
# but its  inverse: C. The higher the value of C, the less the model is regularized.

```

```python
# Build a range of the feature (X) to predict over. Here we just consider pedal width.
X_new = np.linspace(0, 3, 1000)
X_new = np.expand_dims(X_new, axis=1)

# Use the Logistic Regression Model to predict the pedal type based on pedal width
y_proba = log_reg.predict_proba(X_new)

# plot the probability plots
plt.figure(figsize=(6.5,3))
plt.grid(True)

# plot the data used for training, set at 0 and 1
plt.scatter(X[Y==0,3],np.zeros(50),marker='o',label=Y_names[0],zorder=10)
plt.scatter(X[Y==1,3],np.zeros(50),marker='s',label=Y_names[1],zorder=10)
plt.scatter(X[Y==2,3],np.ones(50),marker='d',label=Y_names[2],zorder=10)

# plot the probability
plt.plot(X_new,y_proba[:,0], '--',color=cc[3],label='Not Iris - Virgincia')
plt.plot(X_new,y_proba[:,1],color=cc[2], label='Iris - Virgincia')

# find and plot the 50% decision boundary
x_at_50 = X_new[np.argmin(np.abs(y_proba[:,0] - 0.5))]
plt.vlines(x_at_50,0,1,color='k',linestyles=':',label='Decision Boundary')

plt.xlabel('pedal width (cm)')
plt.ylabel('probability')
plt.legend(framealpha=1)
plt.tight_layout()

# make predictions on the model trained on the data.
log_reg.predict([[1.7]])
```