

```

1 import csv
2
3 import matplotlib.pyplot as plt
4 import numpy as np
5 import json
6
7
8 def add_intercept_fn(x):
9     """Add intercept to matrix x.
10
11     Args:
12         x: 2D NumPy array.
13
14     Returns:
15         New matrix same as x with 1's in the 0th column.
16     """
17     new_x = np.zeros((x.shape[0], x.shape[1] + 1), dtype=x.dtype)
18     new_x[:, 0] = 1
19     new_x[:, 1:] = x
20
21     return new_x
22
23 def load_csv(csv_path, label_col='y', add_intercept=False):
24     """Load dataset from a CSV file.
25
26     Args:
27         csv_path: Path to CSV file containing dataset.
28         label_col: Name of column to use as labels (should be 'y' or 'l').
29         add_intercept: Add an intercept entry to x-values.
30
31     Returns:
32         xs: Numpy array of x-values (inputs).
33         ys: Numpy array of y-values (labels).
34     """
35
36     # Load headers
37     with open(csv_path, 'r', newline='') as csv_fh:
38         headers = csv_fh.readline().strip().split(',')
39
40     # Load features and labels
41     x_cols = [i for i in range(len(headers)) if headers[i].startswith('x')]
42     l_cols = [i for i in range(len(headers)) if headers[i] == label_col]
43     inputs = np.loadtxt(csv_path, delimiter=',', skiprows=1, usecols=x_cols)
44     labels = np.loadtxt(csv_path, delimiter=',', skiprows=1, usecols=l_cols)
45
46     if inputs.ndim == 1:
47         inputs = np.expand_dims(inputs, -1)
48
49     if add_intercept:
50         inputs = add_intercept_fn(inputs)
51
52     return inputs, labels
53
54 def load_spam_dataset(tsv_path):
55     """Load the spam dataset from a TSV file
56
57     Args:
58         csv_path: Path to TSV file containing dataset.
59
60     Returns:
61         messages: A list of string values containing the text of each message.
62         labels: The binary labels (0 or 1) for each message. A 1 indicates spam.
63     """
64
65     messages = []
66     labels = []
67
68     with open(tsv_path, 'r', newline='', encoding='utf8') as tsv_file:
69         reader = csv.reader(tsv_file, delimiter='\t')
70
71         for label, message in reader:
72             messages.append(message)

```

```

73         labels.append(1 if label == 'spam' else 0)
74
75     return messages, np.array(labels)
76
77 def plot(x, y, theta, save_path, correction=1.0):
78     """Plot dataset and fitted logistic regression parameters.
79
80     Args:
81         x: Matrix of training examples, one per row.
82         y: Vector of labels in {0, 1}.
83         theta: Vector of parameters for logistic regression model.
84         save_path: Path to save the plot.
85         correction: Correction factor to apply (Problem 2(e) only).
86     """
87     # Plot dataset
88     plt.figure()
89     plt.plot(x[y == 1, -2], x[y == 1, -1], 'bx', linewidth=2)
90     plt.plot(x[y == 0, -2], x[y == 0, -1], 'go', linewidth=2)
91
92     # Plot decision boundary (found by solving for  $\theta^T x = 0$ )
93     x1 = np.arange(min(x[:, -2]), max(x[:, -2]), 0.01)
94     x2 = -(theta[0] / theta[2] * correction + theta[1] / theta[2] * x1)
95     plt.plot(x1, x2, c='red', linewidth=2)
96
97     # Add labels and save to disk
98     plt.xlabel('x1')
99     plt.ylabel('x2')
100    plt.savefig(save_path)
101
102
103 def plot_contour(predict_fn):
104     """Plot a contour given the provided prediction function"""
105     x, y = np.meshgrid(np.linspace(-10, 10, num=20), np.linspace(-10, 10, num=20))
106     z = np.zeros(x.shape)
107
108     for i in range(x.shape[0]):
109         for j in range(y.shape[1]):
110             z[i, j] = predict_fn([x[i, j], y[i, j]])
111
112     plt.contourf(x, y, z, levels=[-float('inf'), 0, float('inf')], colors=['orange', 'cyan'])
113
114 def plot_points(x, y):
115     """Plot some points where x are the coordinates and y is the label"""
116     x_one = x[y == 0, :]
117     x_two = x[y == 1, :]
118
119     plt.scatter(x_one[:, 0], x_one[:, 1], marker='x', color='red')
120     plt.scatter(x_two[:, 0], x_two[:, 1], marker='o', color='blue')
121
122 def write_json(filename, value):
123     """Write the provided value as JSON to the given filename"""
124     with open(filename, 'w') as f:
125         json.dump(value, f)

```