Solution 1 (a)

Step 1

Given that we must predict y and have no knowledge of x, the best predictor for y is the mean of the y values.

Let $y_1 = 1$, $y_2 = 3$, $y_3 = 4$, $y_4 = 6$, $n = 4$.

Calculate mean of the y values.

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i = \frac{y_1 + y_2 + y_3 + y_4}{n} = \frac{1+3+4+6}{4} = 3.5$$

Hence, the best predictor for y is $\bar{y} = 3.5$.

Step 2

Calculate the mean squared error (MSE).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2 = \frac{(y_1-\bar{y})^2 + (y_2-\bar{y})^2 + (y_3-\bar{y})^2 + (y_4-\bar{y})^2}{n}$$

$$MSE = \frac{(1-3.5)^2 + (3-3.5)^2 + (4-3.5)^2 + (6-3.5)^2}{4} = 3.25$$

Hence, the MSE for the four points is $MSE = 3.25$.

$\therefore$ the best predictor for y is $\bar{y} = 3.5$ with $MSE = 3.25$.

Solution 1 (b)

Step 1

Calculate $y_{prediction}$ using $y = x$ for the following points:

$$(1,1),(1,3),(4,4),(4,6)$$

$y_{prediction}(1,1) = 1$

$y_{prediction}(1,3) = 1$

$y_{prediction}(4,4) = 4$

$y_{prediction}(4,6) = 4$

Step 2

Calculate the mean squared error (MSE).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_{predicted}(x_i, y_i))^2$$

$$MSE = \frac{(y_1 - y_{predicted}(1,1))^2 + (y_2 - y_{predicted}(1,3))^2 + (y_3 - y_{predicted}(4,4))^2 + (y_4 - y_{predicted}(1,1))^2}{n}$$

$$MSE = \frac{(1-1)^2 + (3-1)^2 + (4-4)^2 + (6-4)^2}{4} = 2$$

$\therefore$ the MSE of the linear function $y = x$ on the points, $(1,1),(1,3),(4,4),(4,6)$,is 2.

Solution 1 (c)

Step 1

The line that minimizes the MSE is the *line of best fit* is:

$$MSE(a,b) = \frac{1}{n}\sum_{i=1}^{n}(y_i - (ax_i + b))^2$$

or

$$MSE(a,b) = \frac{(y_1 - (ax_1 + b))^2 + (y_2 - (ax_2 + b))^2 + \ldots + (y_n - (ax_n + b))^2}{n}$$

Let $x_1 = 1$, $y_1 = 1$, $x_2 = 1$, $y_2 = 3$, $x_3 = 4$, $y_3 = 4$, $x_4 = 6$, $y_4 = 6$, $n = 4$.

Subsitute the values into the loss function($MSE(a,b)$):

$$MSE(a,b) = \frac{(y_1 - (ax_1 + b))^2 + (y_2 - (ax_2 + b))^2 + (y_3 - (ax_3 + b))^2 + (y_4 - (ax_4 + b))^2}{4}$$

$$MSE(a,b) = \frac{(1 - (a(1) + b))^2 + (3 - (a(1) + b))^2 + (4 - (a(4) + b))^2 + (6 - (a(4) + b))^2}{4}$$

$$MSE(a,b) = \frac{(1 - (a + b))^2 + (3 - (a + b))^2 + (4 - (4a + b))^2 + (6 - (4a + b))^2}{4}$$

Step 2

The line that minimizes the MSE is the *line of best fit* can be obtained from the following two normal equations:

$$\begin{cases} nb + a\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \\ \\ b\sum_{i=1}^{n} x_i + a\sum_{i=1}^{n} {x_i}^2 = \sum_{i=1}^{n} x_i y_i \end{cases}$$

From the above equations, we can compute the slope a and intercept b of the line of best fit using the following equations:

$$a = \frac{n\sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n\sum_{i=1}^{n} {x_i}^2 - (\sum_{i=1}^{n} x_i)^2} \qquad b = \bar{y} - a\bar{x}$$

Where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ are the means of x and y respectively.

Substituting the values of x and y into the equations:

$$a = \frac{4((1 \times 1) + (1 \times 3) + (4 \times 4) + (4 \times 6)) - (1 + 1 + 4 + 4)(1 + 3 + 4 + 6)}{4(1^2 + 1^2 + 4^2 + 4^2) - (1 + 1 + 4 + 4)^2}$$

$$a = \frac{4(1 + 3 + 16 + 24) - (10)(14)}{4(1 + 1 + 16 + 16) - (10)^2}$$

$$a = \frac{4(44) - 140}{4(34) - 100}$$

$$a = \frac{176 - 140}{136 - 100}$$

$$a = \frac{36}{36} = 1$$

Substituting the value of a into the equation for b:

$$b = \bar{y} - a\bar{x} = 3.5 - 1(2.5) = 1$$

The line of best fit is $y = x + 1$.

Step 3

Calculate MSE of the line of best fit using equation from **Step 1**:

$$MSE(a, b) = \frac{(1 - (a + b))^2 + (3 - (a + b))^2 + (4 - (4a + b))^2 + (6 - (4a + b))^2}{4}$$

$$MSE(a, b) = \frac{(1 - (1(1) + 1))^2 + (3 - (1(1) + 1))^2 + (4 - (1(4) + 1))^2 + (6 - (1(4) + 1))^2}{4}$$

$$MSE(a, b) = \frac{(1 - (1 + 1))^2 + (3 - (1 + 1))^2 + (4 - (4 + 1))^2 + (6 - (4 + 1))^2}{4}$$

$$MSE(a, b) = \frac{(1 - 2)^2 + (3 - 2)^2 + (4 - 5)^2 + (6 - 5)^2}{4}$$

$$MSE(a, b) = \frac{(1)^2 + (1)^2 + (1)^2 + (1)^2}{4}$$

$$MSE(a, b) = \frac{1 + 1 + 1 + 1}{4} = 1$$

$\therefore$ the line of best fit is $y = x + 1$ with $MSE = 1$.

Solution 2 (a)

Step 1

The loss function is defined as:

$$L(s) = \frac{1}{n}\sum_{i=1}^{n}(x_i - s)^2$$

Compute the derivative of $L(s)$ with respect to s ($\frac{dL}{ds}$).

$$\frac{dL}{ds} = \frac{1}{n}\sum_{i=1}^{n}(x_i - s)^2\frac{d}{ds} = \frac{1}{n}\sum_{i=1}^{n}({x_i}^2 - 2x_i s + s^2)\frac{d}{ds} = -\frac{2}{n}\sum_{i=1}^{n}(x_i - s)$$

$\therefore$ the derivative of $L(s)$ with respect to s is:

$$\frac{dL}{ds} = -\frac{2}{n}\sum_{i=1}^{n}(x_i - s)$$

Solution 2 (b)

Step 1

Set the derivative from part (a) to zero to find the value of s:

$$-\frac{2}{n}\sum_{i=1}^{n}(x_i - s) = 0 \tag{1}$$

$$-\frac{n}{2} \cdot -\frac{2}{n}\sum_{i=1}^{n}(x_i - s) = 0 \cdot -\frac{n}{2} \tag{2}$$

$$\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} s = 0 \tag{3}$$

$$\sum_{i=1}^{n} x_i - ns = 0 \tag{4}$$

$$\sum_{i=1}^{n} x_i - ns + ns = 0 + ns \tag{5}$$

$$\sum_{i=1}^{n} x_i = ns \tag{6}$$

$$\frac{1}{n}\sum_{i=1}^{n} x_i = s \tag{7}$$

$$\bar{x} = s \tag{8}$$

$\therefore$ the value of s is $\bar{x}$.

Solution 3

Proof:

We are given a dataset (DS) such that for all data points $(x^{(i)}, y^{(i)})$, where $x^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \mathbb{R}$.

If we predict $\hat{y}^{(i)}$ and the true value is $y^{(i)}$, the penalty is the absolute difference:

$$|y^{(i)} - \hat{y}^{(i)}|$$

Let $x^{(i)}$ be the vector of all $x^{(i)} \in DS$ and m be the vector of slope coefficients.

$$x^{(i)} = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(n)} \end{bmatrix} \qquad m = \begin{bmatrix} m^{(1)} \\ m^{(2)} \\ \vdots \\ m^{(n)} \end{bmatrix}$$

Assume y as a linear function of x, we can express $\hat{y}^{(i)}$ as:

$$\hat{y}^{(i)} = m^\top \cdot x^{(i)} + b$$

where $m \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are the parameters of the linear model.
The total penalty on the training set with n points is the sum of the absolute errors over all n data points:

$$\sum_{i=1}^{n} \left| y^{(i)} - \hat{y}^{(i)} \right|$$

Substituting $\hat{y}^{(i)} = m^\top \cdot x^{(i)} + b$ into the equation yields:

$$\sum_{i=1}^{n} \left| y^{(i)} - (m^\top \cdot x^{(i)} + b) \right|$$

$\therefore$The loss function corresponding to the total penalty on the training set is:

$$L(m, b) = \sum_{i=1}^{n} \left| y^{(i)} - (m^\top \cdot x^{(i)} + b) \right|$$

Solution 4 (a)

Given that x is a picture of an animal and y is the name of the animal. Typically each animal has a unique appearance, then the mapping from x to y is deterministic.

$\therefore$ there is generally little inherent uncertainty in this scenario

Solution 4 (b)

Given that x consists of the dating profiles of two people and y is whether they will be interested in each other. A person's interest in others depends on many observable and unobservable factors.

$\therefore$ there is a significant amount of inherent uncertainty in this scenario.

Solution 4 (c)

Given that x is a speech recording and y is the transcription of the speech into words. Assuming the recording is high quality then the mapping from x to y is deterministic.

$\therefore$ there is generally little inherent uncertainty in this scenario.

Solution 4 (d)

Given that x is the recording of a new song and y is whether it will be a big hit. Whether or not a sing is a hit depends on many factors.

$\therefore$ there is a significant amount of inherent uncertainty in this scenario.

Solution 5 (a)

Let, labels $y \in \{-1, 1\}$.
The decision boundary is defined as:

$$\Pr(y = 1 \mid x) = \Pr(y = -1 \mid x)$$

We know that sum of the probabilities over all labels is equal to 1:

$$\Pr(y = 1 \mid x) + \Pr(y = -1 \mid x) = 1$$

From the above two equations, we can see that $\Pr(y = 1 \mid x) = 0.5$ and $\Pr(y = -1 \mid x) = 0.5$.

$\therefore$ $c = 0.5$ is the decision boundary for this classifier.

Solution 5 (b)

The case $\Pr(y = 1 \mid x) = \frac{3}{4}$ represents a region where the model predicts $y = 1$ with higher confidence.

Solution 5 (c)

The case $\Pr(y = 1 \mid x) = \frac{1}{4}$ represents a region where the model predicts $y = -1$ with higher confidence and is symmetrically opposite the decision boundary (a) compared to (b).

Solution 6 (a)

Strategy

The *mystery.dat* dataset contains 101 features. However, only ten of these features are relevant, while the remaining 91 features are simply noise.

$\therefore$ I will use Lasso Regression because it forces coefficients to 0 for irrelevant features.

Solution 6 (b)

The ten features identified as relevant are:

$$[4, 6, 1, 22, 10, 26, 2, 12, 18, 16]$$

Solution 6 Code

Python Code

```python
## import libraries
import numpy as np
from sklearn.linear_model import LassoCV, Lasso

## read mystery.dat
data = np.loadtxt('mystery.dat', delimiter=',')
x = data[:, :-1]
y = data[:, -1]

## normalize x based on maximum value in all x data
x_norm = x / np.max(np.max(X, axis=0), axis=0)

## find best alpha using LassoCV
## test 100 alphas ranging from 1e-4 to 1e4
lasso_cv = LassoCV(alphas=np.logspace(-4, 4, 100), cv=5)
lasso_cv.fit(x_norm, y)
alpha_best = lasso_cv.alpha_
print(f"best alpha: {alpha_best}")

## fit Lasso model with best alpha
lasso = Lasso(alpha=alpha_best)
lasso.fit(x_norm, y)

## get absolute value of lasso coefficients
coefs = np.abs(lasso.coef_)

## get indices of the 10 largest coefficients
## note: that the coefficients are sorted in descending order
indices = np.argsort(coefs)[::-1][:10]

## print the indices (coordinate number) of the 10 largest coefficients
print("indices of the 10 largest coefficients:")
for i in indices:
    print(f"coordinate number: {i}, coefficient: {coefs[i]}")
```

Solution 7 (a)

I would choose the 3 features that have the largest |coefficients| for the logistic regression.

$\therefore$ I would select top 3 coefficients and the coresponding features: ca (coefficient ≈ 1.78), cp (coefficient ≈ 1.65) and thalach (coefficient ≈ 1.52)

Solution 7 (b)

$\therefore$ the test error for the logistic regression was 0.1845 or 18.45%.

Solution 7 (c)

Comparing error using 5-fold cross-validation, to the models error on the test set was the following:

$$\text{log\_reg test error} = 0.1845$$
$$\text{mean 5-fold cross-validation error} = 0.2000$$
$$\text{Difference (log\_reg - cv)} = 0.0155$$

Solution 7 Code

Python Code

```python
## import libraries
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import numpy as np
import os

## read heart.csv
## note: the first column is the index, so we skip it
## col names are:
## age,sex,cp,trestbps,chol,fbs,restecg,thalach,exang,oldpeak,slope,ca,thal,target
data = np.loadtxt('heart.csv', delimiter=',', dtype=None, skiprows=1)
column_names = os.popen('head -1 heart.csv').read().split(',')
x = data[:, :-1]
y = data[:, -1]

## normalize each x column based on maximum value in column
## note: this will scale each feature [0,1]
x_norm = x / np.max(x, axis=0)

## split data into training and test sets
## note: train_test_split shuffles the data before splitting
## note: test points = 103 and train points = 200 for test_size=0.339
x_train, x_test, y_train, y_test = train_test_split(x_norm, y, test_size=0.339,
    random_state=33)

## fit logistic regression model
log_reg = LogisticRegression(max_iter=1000,solver='liblinear')
log_reg.fit(x_train, y_train)

## predict on test set and calculate accuracy
y_pred = log_reg.predict(x_test)
test_accuracy = accuracy_score(y_test, y_pred)

## get absolute value of logistic regression coefficients
coefs = np.abs(log_reg.coef_[0])

## get indices of the 3 largest coefficients
## note: that the coefficients are sorted in descending order
indices = np.argsort(coefs)[::-1][:3]

## top 3 coefficients and features
print("top 3 coefficients and features:")
for i in indices:
    print(f"index: {i} coefficient: {coefs[i]}, feature: {column_names[i]}")

## log_reg test error
test_error = 1 - test_accuracy
print("log_reg test error:", test_error)

## 5-fold cross-validation error
cv_scores = cross_val_score(log_reg, x_train, y_train, cv=5, scoring='accuracy')
cv_error = 1 - cv_scores.mean()
print("mean 5-fold cross-validation error:", cv_error)

# compare log_reg test error and 5-fold cross-validation error
print(f"Difference (log_reg - cv): {test_error - cv_error:.4f}")
```