Problem 1. (10 pts) Write **True** or **False** next to each of the following statements.

This problem will be graded as follows: +1 points for each correct answer, -1 points for each incorrect answer, and 0 points for each blank answer. So, for instance, one correct answer and one incorrect answer will cancel each other out. You cannot get less than zero points on this problem, even if every answer is incorrect.

a) We use the validation set to compare different models, and to make our final model selection. True

b) Logistic regression is a regression algorithm (as opposed to a classification algorithm). False

c) In general, increasing the number of features used in a model will tend to increase the chances that the model will overfit. True

d) When using `MinMaxScaler()` to scale numerical features, we will create and fit separate instances of the scaler for the training, validation, and testing sets. False

e) The ridge algorithm is a regularized version of least squares regression that tries to avoid overfitting by applying a penalty to models based on the size of their coefficients. True

f) Increasing the hyper-parameter α in a LASSO regression model will typically make it less likely that the model will overfit. True

g) Increasing the hyper-parameter K in a KNN model will typically make it less likely that the model will overfit. True

h) When selecting a value of α in a ridge or LASSO model, we typically choose the value that results in the greatest r-squared score for the training set. False

i) The coefficients in a logistic regression model are trained by minimizing the sum of squared errors objective function. False

j) One downside of a KNN classification algorithm is that training a model of this type is often very computationally expensive. False

Problem 2 (8 pts). Assume that you are provided with a feature array named X and a label array named y. Consider the code below. Provide the output for the last 6 print statements on the lines provided. Provide your answers exactly as they would be displayed in Python.

Code

```
print(X.shape)

print(y.shape)

from sklearn.model_selection import train_test_split

X_train, X_tv, y_train, y_tv = train_test_split(
      X, y, test_size=0.30, random_state=1)

X_val, X_test, y_val, y_test = train_test_split(
      X_tv, y_tv, test_size=0.5, random_state=1)

print(X_train.shape)

print(X_val.shape)

print(X_test.shape)

print(y_train.shape)

print(y_val.shape)

print(y_test.shape)
```

Output

```
(360, 20)

(360,)
```

(252, 20)

(54, 20)

(54, 20)

(252,)

(54,)

(54,)

Problem 3. (12 pts) The confusion matrix for a testing set in a classification problem with three classes is provided below. Find the precision and recall for each class, as well as the overall accuracy of the model. **Round to three decimal places.**

	Class 0	Class 1	Class 2	
Class 0	9	3	4	16
Class 1	1	12	2	15
Class 2	2	10	25	37
	12	25	31	68

	Precision	Recall
Class 0	0.75	0.563
Class 1	0.48	0.8
Class 2	0.806	0.676

Accuracy: 0.676

Problem 4 (12 pts). Assume that you are provided with two 2D arrays, `X_num` and `X_cat`, whose contents are as shown below. The array `X_num` contains the values for a single numerical (quantitative) feature, while `X_cat` contains the values for two categorical (qualitative) features. The feature arrays are preprocessed and combined into a single array by running the code below. Provide the contents of the array `X_preprocessed` by completing the table on the right. You may not need all of the columns provided. If not, leave any extra columns blank.

```
from sklearn.preprocessing import PolynomialFeatures, OneHotEncoder

poly = PolynomialFeatures(3)
Xp = poly.fit_transform(X_num)

enc = OneHotEncoder(sparse=False)
Xe = enc.fit_transform(X_cat)

X_preprocessed = np.hstack((Xp, Xe))
```

X_num

3
-2
1
4
-1

X_cat

c	e
a	a
b	o
b	a
a	o

X_preprocessed

1	3	9	27	0	0	1	0	1	0
1	-2	4	-8	1	0	0	1	0	0
1	1	1	1	0	1	0	0	0	1
1	4	16	64	0	1	0	1	0	0
1	-1	1	-1	1	0	0	0	0	1

Problem 5 (10 pts). Assume that you are provided with three 2D arrays, named `X_train`, `X_val`, and `X_test`. The contents of these arrays are provided in the tables on the right below. Complete the tables below to show the contents of the arrays `Xs_train` and `Xs_val` after executing the following code:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaler.fit(X_train)
Xs_train = scaler.transform(X_train)
Xs_val = scaler.transform(X_val)
Xs_test = scaler.transform(X_test)
```

X_train

6	0.8
4	0.6
2	0
7	0.4

X_val

0	0.2
1	1
5	0.4
3	0.6

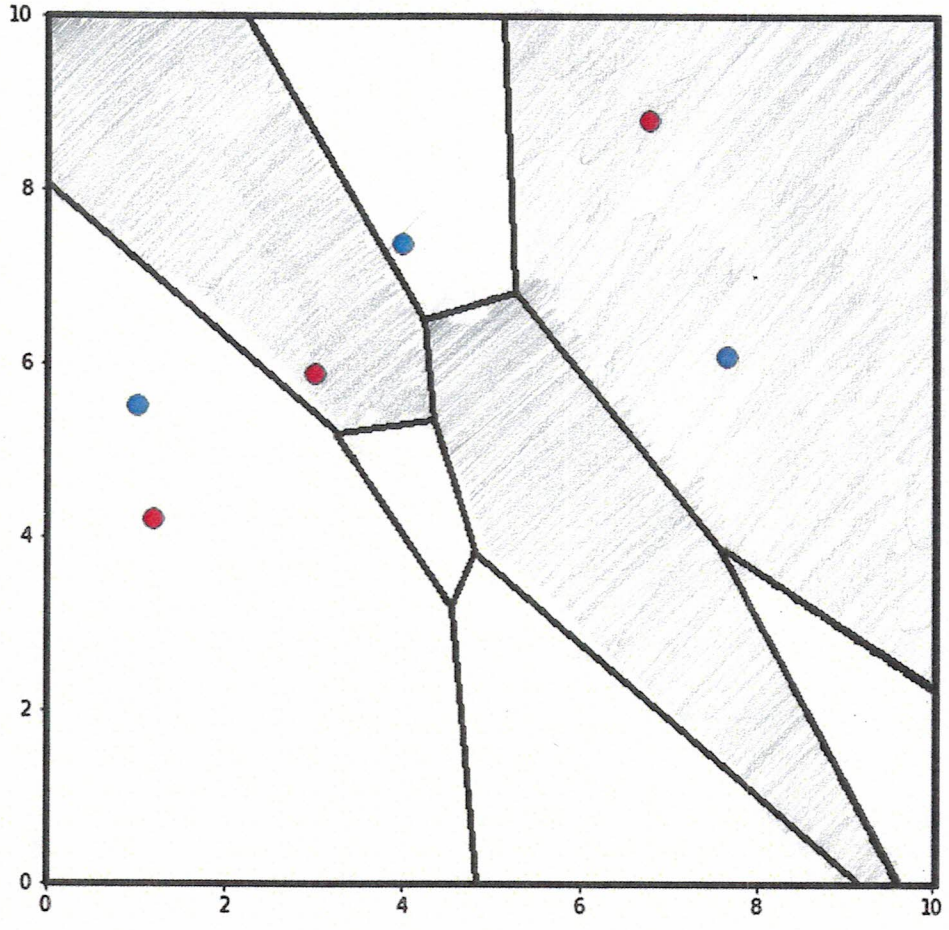
Xs_train

0.8	1
0.4	0.75
0	0
1	0.5

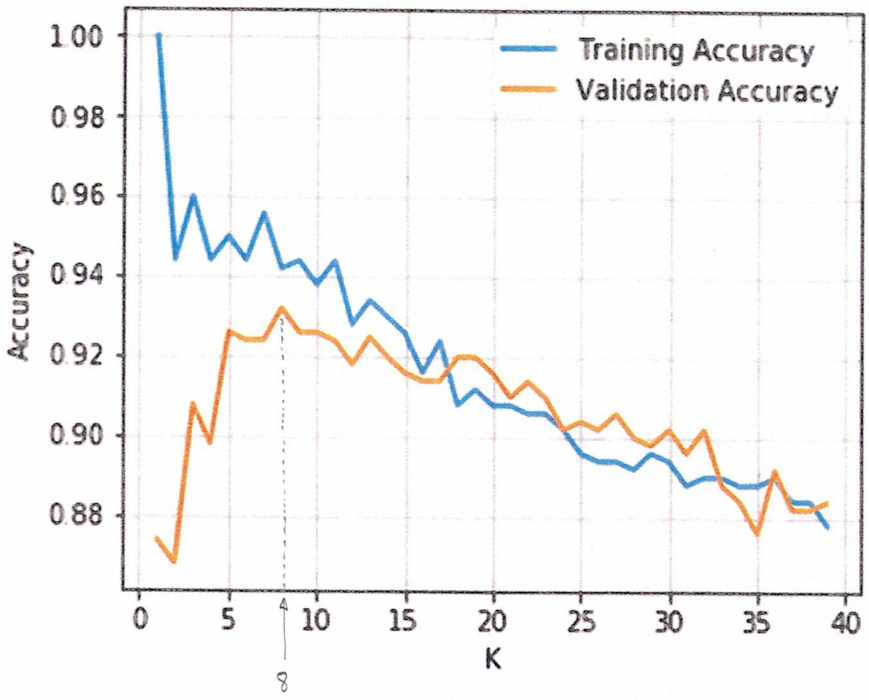
Xs_val

-0.4	0.25
-0.2	1.25
0.6	0.5
0.2	0.75

Problem 6. (10 pts) A scatter plot consisting of six points is shown below. Each point is labelled as one of two classes: "red" or "blue". Assume a KNN classification algorithm is trained on this dataset with K=3. Determine how points in each of the displayed regions would be classified using this algorithm. Shade in any region that would be classified as the "blue" class, and leave blank any region that would be classified as the "red" class. A compass might be useful for this problem.



Problem 7. (6 pts) The graph below the training and validation accuracy scores of a K-Nearest Neighbors models for varying values of K. Determine the value of K that would be the best to use for this classification task. Briefly explain your answer.



K = 8

This is where validation accuracy is at its highest.

Problem 8 (16 pts). Assume you are provided with a dataset containing 5 observations and three features to use in creating a regression model. The table below contains information for two different proposed models, with each row relating to a different model. The first four columns provide the proposed coefficient estimates for each model. Columns 5 – 9 contain the **predicted** y values resulting from each of the models. The **true** y values are provided in the bottom row of the table.

For both of the proposed models, calculate its loss using the linear regression loss function, the ridge regression loss function with $\alpha = 0.5$, and the LASSO regression loss function with $\alpha = 0.5$. In each case, use SSE rather than MSE as the "baseline" loss.

After calculating each of the loss values, circle the best loss for each of the three different loss functions. In other words, for each of the last three columns, circle the best loss value that appears in that specific column.

$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$	$\hat{y}_4$	$\hat{y}_5$	Linear Regression	Ridge Regresssion	LASSO Regression
5	4	-3	2	9	11	6	6	5	(7)	21.5	(11.5)
6	2	-2	2	6	12	5	4	5	10	(16)	13
True y Values →				8	10	6	4	6			

Problem 9. (16 pts) A training set for a classification task is provided below. The dataset has two features and one categorical label, y, which has two possible classes: "red" and "blue". You are asked to score a logistic regression model for this training data.

Round all answers to three decimal places on this problem.

$x^{(1)}$	1	5	6	2	7
$x^{(2)}$	6	3	5	2	1
y	red	blue	blue	red	red

a) In the model below, $\hat{p}$ is an estimate for the probability that an observation falls into the red class. Let $\hat{\pi} = \hat{p}$ for red observations and let $\hat{\pi} = 1 - \hat{p}$ for blue observations. For the following model, find $\hat{p}$ and $\hat{\pi}$ for each observation.

$$\hat{p} = 1/\left[1 + \exp\left(-4 + 0.6x^{(1)} + 0.5x^{(2)}\right)\right] = \frac{1}{1 + e^{-4+0.6x^{(1)}+0.5x^{(2)}}}$$

$\hat{p}$	0.599	0.378	0.109	0.858	0.332
$\hat{\pi}$	0.599	0.622	0.891	0.858	0.332

negative

b) Calculate the log-likelihood score for this model.

$$NLL = -\ln(0.599) - \ln(0.622) - \ln(0.891) - \ln(0.858) - \ln(0.332)$$

$$= \boxed{2.358}$$

c) When calculating likelihood scores, why do we work typical work with log-likelihood rather than likelihood?

① Log-likelihood is easier to work with computationally. Likelihood scores tend to be very small, and could get rounded off to zero.

② Log-likelihood is easier to work with mathematically. It tends to be easier to work with sums rather than products.