SAMPLE MIDTERM EXAM QUESTIONS

CS 178, SPRING 2023

May 2023

- These questions are intended to provide you with a general idea of the level of the type and difficulty of exam questions for the midterm.
- Each set of questions is followed by a page with solutions; so you may want to first try the questions and then check your answers against the solutions.
- The presence or absence of a particular topic in these questions does not mean that the topic will be present or absent on the exam. These are just sample questions.
- Similarly, the format and length of these questions will not necessarily match the format and length of the exam questions

True/False Questions

1. True or False: In general in machine learning the accuracy of a classifier on a test data set will be lower than the accuracy of the classifier on the training data it was trained on.

2. True or false: the kNN classifier can ignore features that are not relevant to the prediction:

3. True or false: The nearest centroid classifier does not use gradient descent in its training algorithm:

4. True or false: In a classification problem with K classes, the baseline classifier that always predicts the majority class (for any input $\mathbf{x}$) will always have a classification accuracy of $1/K$.

5. True or false: when we train a logistic model on a dataset with d features, the number of parameters will in general be $d + 1$.

6. True or false: in a kNN classifier, increasing the value of k will increase the accuracy of the classifier on test data.

7. True or false: Suppose we have a feature vector $x = (x_1, x_2)$ with dimension $d = 2$. The degree $p = 2$ polynomial feature expansion of x is given by $z = (x_1, x_2, x_1^2, x_2^2)$.

8. True or false: The space complexity of making a prediction with a linear regression model scales linearly with the number of training datapoints.

9. True or false: The learning rate λ must be fixed and cannot change during the gradient descent algorithm.

Solutions to True/False Questions

1. True or False: In general in machine learning the accuracy of a classifier on a test data set will be lower than the accuracy of the classifier on the training data it was trained on.

   True

2. True or false: the kNN classifier can ignore features that are not relevant to the prediction.

   False: kNN doesn't have any ability to directly ignore features since it uses a distance function which is a function of all the features. This is unlike a classifier like a logistic classifier which could assign a weight of 0 to an input feature to effectively ignore it.

3. True or false: The nearest centroid classifier does not use gradient descent in its training algorithm.

   True

4. True or false: In a classification problem with K classes, the baseline classifier that always predicts the majority class (for any input $\mathbf{x}$) will always have a classification accuracy of $1/K$.

   False: the accuracy will be probability of the most likely class, e.g., if $K = 2$ and $P(y = 1) = 0.9$, then the majority classifier will predict $y = 1$ all the time and be correct 90% of the time.

5. True or false: when we train a logistic model on a dataset with d features, the number of parameters will in general be $d + 1$.

   True: one parameter θ_j per input feature x_j and an additional intercept term with parameter θ_0

6. True or false: in a kNN classifier, increasing the value of k will increase the accuracy of the classifier on test data.

   False: we saw in lectures and homework that test accuracy can increase or decrease as k changes for the kNN classifier.

7. True or false: Suppose we have a feature vector $x = (x_1, x_2)$ with dimension $d = 2$. The degree $p = 2$ polynomial feature expansion of x is given by $z = (x_1, x_2, x_1^2, x_2^2)$.

   False: The correct feature expansion is $z = (x_1, x_2, x_1^2, x_2^2, x_1x_2)$.

8. True or false: The space complexity of making a prediction with a linear regression model scales linearly with the number of training datapoints.

   False: The space complexity of linear regression is $O(d)$, where d is the number of features in the data.

9. True or False: The learning rate λ must be fixed and cannot change during the gradient descent algorithm.

   False: you can change the learning rate over the course of the iterations, as is done with learning rate schedulers.

Nearest Centroids

Suppose that, when implementing nearest centroid we replaced the centroids with the following vectors, one per class:

$$\mathbf{s}_k = \sum_{i:y_i=k} \mathbf{x}_i.$$

and we then use a classifier where we assign a new datapoint $\mathbf{x}$ to class whose vector $\mathbf{s}_k$ is closest to $\mathbf{x}$ in terms of Euclidean distance. In the equation above, $\sum_{i:y_i=k}$ is interpreted to mean we are summing only over the training examples with labels $y_i = k$.

1. What is the correct definition of centroids $\boldsymbol{\mu}_k$ for the nearest centroid (NC) classifier instead of the equation for $\mathbf{s}_k$ above?

2. Would the classifier proposed above perform as well (in general) making predictions as the NC classifier which uses the correct centroids $\boldsymbol{\mu}_k$? State "yes" or "no" and explain your answer in one or two sentences.

3. In general for the nearest-centroid algorithm what is the space complexity for the classifier at prediction time?

Solutions to Nearest Centroids

1. $$\boldsymbol{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} \mathbf{x}_i.$$

   where n_k is the number of terms in the sum, i.e., the number of datapoints in the training data that have class label k.

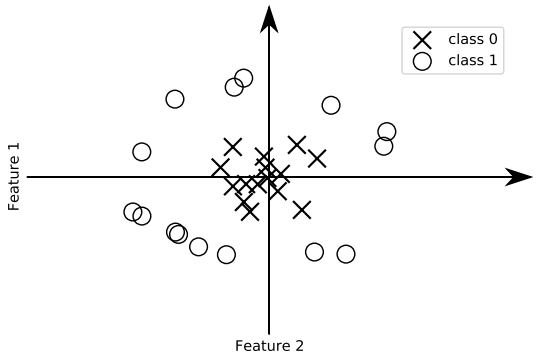
2. No. It would not work as well in general, as the new vectors per class $\mathbf{s}_k$ would be on average be much larger in magnitude than the true centroids $\boldsymbol{\mu}_k$. For large datasets this would "push" the data to be far away from the original data (in the d-dimensional feature space) and in many cases the resulting classifier would end up just predicting whatever class happens to have an $\mathbf{s}_k$ vector that is closest to the original data (e.g. a class with relatively few labels n_k compared to the other classes).

3. We need to store one centroid per class, and each centroid is a vector of length d, so if we have K classes then the space complexity is $O(dK)$

kNN Classifier

Consider a kNN classifier with training data $X = \{0, 1, 2, 3, 4, 5, 10, 20, 30, 40, 50, 60, 70\}$, and $y = \{0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1\}$.

1. What is the classification accuracy of kNN on this training dataset when $k = 1$?
2. Let $k = 13$, what is the accuracy score of the training dataset? What are the labels of the numbers $1.2, 3.6, 7, 8, 12, 15$?
3. When $k = 3$, at what point(s) do(es) the decision boundary(ies) lie at?
4. Even values of k are traditionally avoided when using kNN classifiers on binary data. Explain why.
5. Suppose we have given a dataset to perform binary classification with K-NN, but 10% of labels have randomly been flipped from their true values in the training data. Is it better to use $k = 1$ or $k = 3$ for this problem?
6. Suppose we are trying to classify points with class 0 from class 1 in the following plot with two features x_1 and x_2.



Is it better to choose K-NN with $k = 1$ or a nearest-centroid (NC) classifier as a classification model for this task? Answer "kNN" or "NC" (as your preference) and provide a 1 sentence explanation for your choice.

Solutions to kNN Classifier

1. The accuracy score would be 1, as there are no ties.
2. The accuracy score would be 7/13, as all predictions would be the majority class. The labels are all 1's.
3. The decision boundary is located at $x = 12$.
4. With an even value of k, ties are possible, which requires having a method to break ties when predicting a class label; so it is simpler to avoid ties by making k be odd.
5. $k = 3$ should perform better since it can help in averaging over the noise from the randomly flipped labels.
6. kNN will be better. The NC classifier will have its centroids for each class near 0 and will be unable to make accurate predictions: will have a classification accuracy around 0.5 if both classes occur in equal proportions in the data.

Logistic Classifier

The logistic classifier is defined by the following equations, as discussed in class:

$$z(\mathbf{x};\boldsymbol{\theta}) = \theta_0 + \sum_{j=1}^{d} \theta_j x_j$$

and

$$P(y=1|\mathbf{x}) = f(\mathbf{x};\boldsymbol{\theta}) = \frac{1}{1+e^{-z(\mathbf{x};\boldsymbol{\theta})}}$$

1. Suppose that you have trained a logistic regression classifier, and it produces a value of $z(\mathbf{x}_1;\boldsymbol{\theta}) = 0$ for $\mathbf{x}_1$, and produces a value of $z(\mathbf{x}_2;\boldsymbol{\theta}) = \infty$ for $\mathbf{x}_2$. What are the corresponding values of $P(y = 1|\mathbf{x}_1)$ and $P(y = 1|\mathbf{x}_2)$?

2. In the logistic classifier, when training a model via Gradient Descent, provide one sentence describing why using a large learning rate is not always ideal

3. What is the length of the gradient vector used in the gradient descent algorithm for the logistic classifier?

4. Imagine with the logistic classifier that instead of minimizing log-loss function, we try to directly minimize *the classification error rate* on the training data. Could we use Gradient Descent to achieve this? Answer Yes or No, and explain your answer in one sentence.

5. How can you extend the logistic classifier from 2 classes to K classes? (Explain in 1 or 2 sentences).

Solution for Logistic Classifier

1. 0.5 and 1
2. A large learning rate can cause gradient descent to overshoot the solution in parameter space and not converge.
3. It will be $d + 1$ where d is the number of features. The length of the gradient vector for any classification model will correspond to the number of parameters in the model, which for the logistic model is $d + 1$ in general.
4. No, because the new objective function is not differentiable.
5. Define K logistic functions, each with their own parameters, with a softmax function to ensure that K values sum to 1.

Evaluating Classifiers

Consider a classifier trained on a training dataset with 3 classes, which then gives a prediction on 10 test samples as $\hat{\mathbf{y}} = \{1, 2, 0, 2, 1, 0, 1, 1, 0, 2\}$. The true labels of these test samples are $\mathbf{y} = \{1, 0, 1, 2, 1, 2, 1, 1, 0, 1\}$.

1. What is the classification accuracy and classification error of this classifier on the test samples?
2. If in the training data, the class label "1" occurs most frequently (compared to the frequency of the other 2 classes), what would be the error rate of the majority classifier on the text data above?
3. Draw a table showing the confusion matrix based on the true label $\mathbf{y}$ (as rows) and the predicted label $\hat{\mathbf{y}}$ (as columns)
4. In one sentence, clearly and briefly one possible reason for using a separate test dataset to evaluate a classifier.

Solutions to Evaluating Classifiers

1. The accuracy would be $\frac{1}{10}(1+0+0+1+1+0+1+1+1+0) = 0.6$. The error rate would be $1 - 0.6 = 0.4$.

2. The majority class corresponds to always predicting the same class (for every $\mathbf{x}$), where the class being predicted is the one that occurs most commonly in the training data. In this case that class is "1" and if we predict $\hat{\mathbf{y}}$ to be all 1's then the test accuracy will be 6/10 = 0.6. (So the classifier in part 1 is doing no better than the majority classifier).

3. The confusion matrix would be

	$\hat{y}=0$	$\hat{y}=1$	$\hat{y}=2$
$y=0$	1	0	1
$y=1$	1	4	1
$y=2$	1	0	1

4. We want to know performance on new data it has not seen before and accuracy on the training data will typically be "optimistic" (better) than accuracy on new test data.