CS 530 Spring 2021—Final Exam

You have 3 and a half hours (210 minutes) to complete the final examination below. Each question is worth the same number of points. We left you a marker for each of your answers. Though you can of course use more space any answer whenever needed. You should complete all 3 questions and all the parts of all the questions. Make sure to also sign and date the attestation directly below.

Sign the below with your full name:

I hereby certify that I worked on this exam by myself and everything written in this exam was conceived and developed by me alone. I did not discuss or share any part of this exam with anyone else and certainly did not use information from anyone else in any part of the exam.

**Your full name and date: Toby Chappell - 05/20/21**

Question 1

The Fashion-MNIST dataset is a dataset of Zalando's article images, with 28x28 grayscale images of 70,000 fashion products from 10 categories, and 7,000 images per category. The training set has 60,000 images, and the test set has 10,000 images.

1. You decided to create a 10-dimensional representation of the images using an autoencoder. So, you created an autoencoder that went from 784 to 100 to 30 to 10 to 30 to 100 to 784 neurons, in consecutive layers, all with linear transfer functions. You trained it in the usual manner and then kept only the part of the network from the input layer to the 10-neuron code layer, as is common. Your friend instead used PCA and took the top 10 principal components. Which of the 2 methods is better? Or is one not superior to the other? Explain your answer.

**Your answer: For this particular case, an autoencoder would be better suited for this task. Autoencoders allow you to learn far more complex relationships compared to PCA, which in turn helps attain a higher accuracy. The only drawback here is that autoencoders require far more computations and tuning compared to PCA.**

1. You try classifying on the 10-dimensional representation of the Fashion-MNIST images from (a) and get unimpressive accuracy. So, instead, you decide to use a ConvNet to directly classify the images into the 10 categories. You run your network and plot the training error over epochs and get the plot below. Do you think that it is problematic that the network converges so slowly? If so, suggest 3 different methods that are likely to speed up the convergence. If you think the convergence speed is fine, explain why.

**Your answer: Given that the size of the dataset is very large, a slow convergence can be problematic (the computational cost is going to be very large). To help with this issue there are a few methods to try. For one, adding Dropout layers will decrease the number of parameters that need to be learned making the runtime quicker (it also makes the model more generalizable which can help performance). Secondly, adding pooling layers condenses the feature maps which will again reduce the number of parameters to be learned (pooling also “blurs” the image which will also make it more generalizable to unseen samples as well). Lastly, you can increase the learning rate so that it converges quicker (you would just need to make sure that the learning rate is not to large so that you do not overshoot the optimal parameters in your model).**

![Chart, line chart, histogram

Description automatically generated]()

1. You get your network to work well and get good results on your classification. You show the results to your friend, but they are unimpressed. They say that if you were to take all the 28x28 grayscale images in the Fashion-MNIST dataset as is and run them into the K-means algorithm with k=10, you would get a clear separation into the 10 categories of Fashion-MNIST. Do you agree that this would be the output of K-means on the Fashion-MNIST dataset? Explain your answer.

**Your answer: It would be very surprising if K-means was able to come to this result. For one, K-means does not learn using labels which is fairly useful information when attempting to classify something. Furthermore, images in general are extremely noisy (especially ones that are unfiltered as in this case) and as such can run into issues in finding clear separations. Lastly, the ConvNet is going to learn far more informative and complex features in making classifications that K-means will simply just not be able to do.**

Question 2

Below we describe 4 datasets. For each dataset, you can use any of the classification algorithms that we learned in class—logistic regression, LDA, K-NN, SVM, Random Forest, and Deep Neural Networks (with any architecture). Explain which algorithm you would be especially eager to try and which one you would specifically not use. Your answers could be based on the goal of the study, the different underlying assumptions for each algorithm, the characteristics of the preprocessing steps you must take, and so on.

1. You work at a self-driving car company and want to build a program to find the position of all the vehicles in a video stream captured by the self-driving car camera. You plan to break each frame of the video stream into a 12x12 grid and build a classifier to predict whether a vehicle is present in each square of the grid. You can train on 10,000 images with the labels “vehicle present” and “vehicle absent” in each square of the grid.

**Your answer: For this particular problem, the algorithm most suited for this would probably be a Deep Neural Network, specifically a ConvNet. The reason for this is because the data is highly nonlinear and a ConvNet would use information about related pixels in order to extract the most important features. In addition, high accuracy is a must so that the self-driving car does not end up crashing into other cars on the road, and a ConvNet would be able to achieve this type of accuracy. An algorithm that I would definitely not use would be LDA since LDA’s can only make linear separations. Since the data is nonlinear, LDA would obtain a fairly low accuracy.**

1. You are a neuroscientist running an experiment to predict, in real time, whether a participant is about to move their left or right hand or their left or right foot from their brain activity alone, before the subject moves. You will rely on EEG (high-dimensional time-series data that reflects brain activity from a collection of electrodes on the scalp) as your input. You collect a dataset of 50 movements each of the left or right hand and left or right foot.

**Your answer: Since the data is highly dimensional, an SVM with an RBF kernel might be able to perform fairly well. This is because SVM is able to project data into even higher dimension in which the data would become linearly or nearly linearly separable. For this case, I would not use Logistic Regression since we are attempting to predict more than 2 classes (even though it may be possible, it would be extremely cumbersome to do so without a guarantee in a higher accuracy).**

1. You have a health dataset containing 12068 records of patients from a hospital. Each record contains various demographic information about the patients (age, gender, weight, …) as well as their medical history (hospitalizations, surgeries, major past illnesses, …), vital signals, bloodwork, current and past medication, and so on. All in all, you have 268 fields (or variables) for each record (which is associated with a single patient), though not all of them are filled for each patient (unavailable information is denoted by NaNs). Your algorithm should predict the likelihood that the patient would recover well from open-heart surgery.

**Your answer: The optimal algorithm for this problem would be Logistic Regression. The primary reason for this is because we want the likelihood that a patient would recover and Logistic Regression is the only algorithm that returns the probability of an event happening. In addition, it should perform fairly well since we have a large number of samples and can make use of most of these features. I would probably not use K-NN for this case because K-NN does not scale very well to large datasets such as this. In addition, given the large number of features, K-NN would suffer from the curse of dimensionality (the data would be sparser in high-dimensional space so a data points “neighbors” would likely be very far away).**

1. You collected survey data with 50 psychological measures (e.g., stress, IQ, and personality) and 25 demographic measures (e.g., age, income, and political affiliation) for 2000 subjects from an online survey related to the Covid-19 pandemic. You want to build a diagnosis system to predict the depression level (high, medium, or low) of each participant. You further want the system to output which of the measures are the most saliant in their relation to depression.

**Your answer: For this case, I would use Random Forest since it would determine on its own which features are the most optimal in making predictions. In addition, it would decorrelate the data which would help improve accuracy. I would not use LDA since it would result in dimensionality reduction. Since we want to determine which features are the most influential, it would be very difficult to interpret what the resulting, lower-dimensional feature-space corresponds to.**

Question 3

1. A young data-scientist friend enthusiastically tells you about a classification model that she has developed for a data science competition. She is instructed to predict the price of houses based on various information about the houses (size, lot size, number of rooms, zip code, etc.). The competition gives a training set of 1134 houses, with their various attributes (variables), and their prices. She came up with a specific model, and her model has performed very well on the available training data. She is therefore eagerly waiting to test her model on the test data, which has not been released yet. A week later you meet her again, and unfortunately, her model performed terribly on the final test data.
   1. What is the most likely explanation for what happened to your friend?

**Your answer: The most likely reason that she performed badly on the test set was because she overfitted the training set. As such, her model suffers from overfitting since it does not generalize well.**

* 1. Suggest two methods that she could have used to test the results of her model and potentially realize that there was a problem before she tried it on the test set.

**Your answer: One way to have realized the fault in the model beforehand is if she had randomly split her training set into a training and testing set. If the training accuracy was much better than the testing accuracy, it would have implied that the model was overfitting. Another method she could use is to use k-fold cross validation and find the average accuracy over all the folds. This would subset her training set into k training/testing sets and give a more accurate accuracy of her model on unseen data.**

* 1. If she would have found the problem earlier, suggest two things that she could have done to improve the accuracy of her model on the test set.

**Your answer: In order to improve her accuracy, she needs to make her model more generalizable. To do this, she can add regularization into her model which would make her model less complex and as such, should generalize well to the test set. Another option is to remove features from the model which again would reduce the complexity of the model and generalize better to the test set.**

* 1. You propose that she use data augmentation to increase the size of the dataset 10-fold, which would likely improve the accuracy. However, this is a regression problem (house prices are continuous). Can you nevertheless think of a method to carry out some form of data augmentation on this dataset? Explain how it would work.

**Your answer: One way to add more samples to the dataset is by using bootstrapping. To do this, you would just need to randomly sample the dataset with replacement for as large you needed your dataset to be (in this case you would take 11340 samples to make the dataset 10 time larger). This would increase the size of the data set while preserving the mean and variance.**

1. You work for a startup company that is trying to automate the diagnosis of Alzheimer’s Disease (AD) from structural 3D MRI brain scans. The company has a dataset of 230 such scans, each containing healthy brains (80% of the data) and AD brains (the remaining 20% of the data). Each brain scan is 192 x 192 x 100 voxels.
   1. Your dataset is imbalanced. But you would like to train your model on a balanced data set. Suggest 2 methods that you could use to balance the dataset out. What are the advantages and disadvantages of each method in this case?

**Your answer: The probably most simple way to balance the dataset is to randomly remove 138 healthy brains so that you have 46 brains of each class. Quite obviously, the disadvantage of doing this would be that you lose a lot of information that would have been useful. However, the dataset would be balanced and as such would not automatically favor classifying brains as healthy. The next option is to use data augmentation and add 138 AD brains. This is probably the better option since there would be no information loss and the models should be able to learn more complex relationships with more samples. The downside to augmenting the dataset is that it can create bias and not generalize very well to unseen AD brain scans. In addition, it does require more work since the optimal augmentation method would have to be determined.**

* 1. Would it be better to use a deep ConvNet (say, 8-10 layers) or a shallow one (say, 2-3 layers) for this problem? Explain your answer.

**Your answer: For this problem, a deeper ConvNet would most likely be better. Given that each sample is very large and we do not have a lot of samples, it is going to very difficult to find which features are going to be the most important. As such, adding more layers should help the model improve its accuracy and be able to determine what to look for when making classification decisions. The obvious downside is that performance is going to suffer greatly, so a lot of layers would need to be devoted into condensing the data in addition to some preprocessing on the data.**

* 1. Your clients insist that you use a deep CNN to classify the scans. Suggest 2 methods that you could implement to improve the chances that a deep CNN would work well on your dataset. Explain why you think each method would work.

**Your answer: For one, you would need a multiple Dropout layers and Pooling layers. This would make the model more generalizable (which is required since there are only 230 samples) and help with runtime (which is also going to be a big issue here). Secondly, by condensing the data beforehand using an autoencoder or PCA, the model will be able to run faster and the condensed data should highlight features of interest that the model can use to better classify.**