**Homework 1:**

***Supervised Learning***

**Datasets and why are they important?**

**Nursery**

The first dataset I have chosen comes from 1980’s Ljubljana, Slovenia. During that period, there used to be an excessive number of applications to nursery schools but not enough school capacity to admit all of the applicants. Therefore, the government discretized some certain aspects of the applicants by assigning points and discrete values to them. In the end, there were a total of three main categories that the applications were evaluated: parents’ occupations, family structure and financial standing, and social and health picture of the family. In this dataset, these three subcategories are described with 8 different features (attributes) and these attributes have around 3-5 options each. The target value (label) is also categorized in five different categories, ranging from ‘*not recommended*’ to ‘*special priority*’. In this dataset, there are 12960 instances, meaning 12960 children who are classified. I separated my training and test data as 80% and 20% respectively of the total number of instances.

The reason I have chosen this dataset is because it comes from real life and the end result (labels) was something that had a great impact on thousands of families and their children. In reality, government formulized a way to classify the children and in this project, I will be trying to capture their classification methods and models. Throughout this paper, I will be referring this dataset as the ‘nursery dataset.’

**Letter Recognition (Letters)**

This dataset contains 20,000 instances of digitally typed and distorted in different ways letters from 20 different fonts of the English alphabet, meaning the result (label) has 26 different available options. Each letter appears between 734-813 times in the dataset, suggesting a somewhat balanced distribution. Each instance is defined by 16 different features including height, width of the char box, the number of pixels available and other features that describe edges of the letters. Each attribute is discredited from a continuous scale to an integer range between 1-15.

The reason I have chosen this dataset was to be able to compare and contrast with the first dataset. Since this dataset contains attributes that are derived from a continuous range, intuitively, the classification process seems to be different compared to the first one, a dataset with very discreet and polarized features. Another reason I have chosen this dataset was to be able to work with one of the most popular concepts in the industry: OCR (optical character recognition). I wanted to see how different machine learning algorithms would be able to capture the basics of this method and be successful when it comes to recognizing shapes and classifying them as the correct letters. Throughout this paper, I will be referring this dataset as the ‘letter dataset.’

**Project Overview**

This homework will have analyses of 5 different supervised learning algorithms. I will be exploring how different variables and kernels affect the results of these algorithms and how do they perform when ran on the two datasets I have. I separated each dataset as 80% training data, and 20% testing data to be able to test the trained classifiers on the test data. In the analyses part, there will be learning curve graphs and varying characteristic variables versus the accuracy graphs. Most of the classifier models are cross-validated 7 times and this will be later explored.

**Decision Trees**

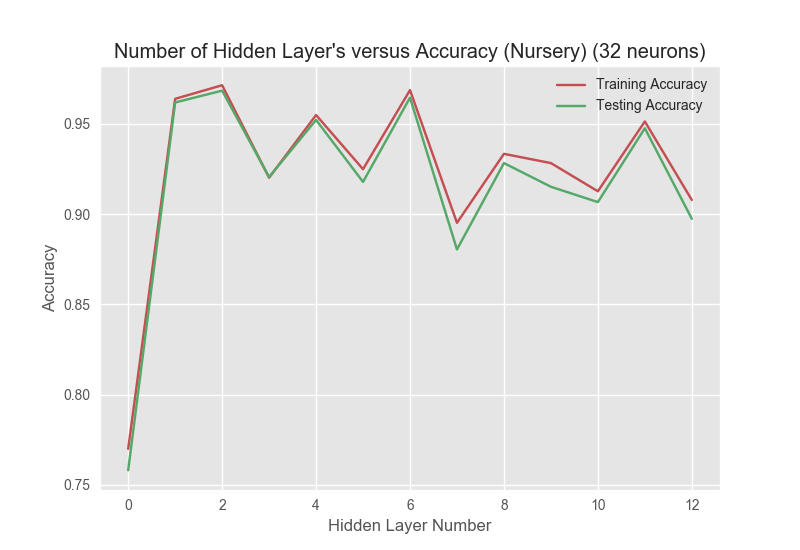
**Nursery**

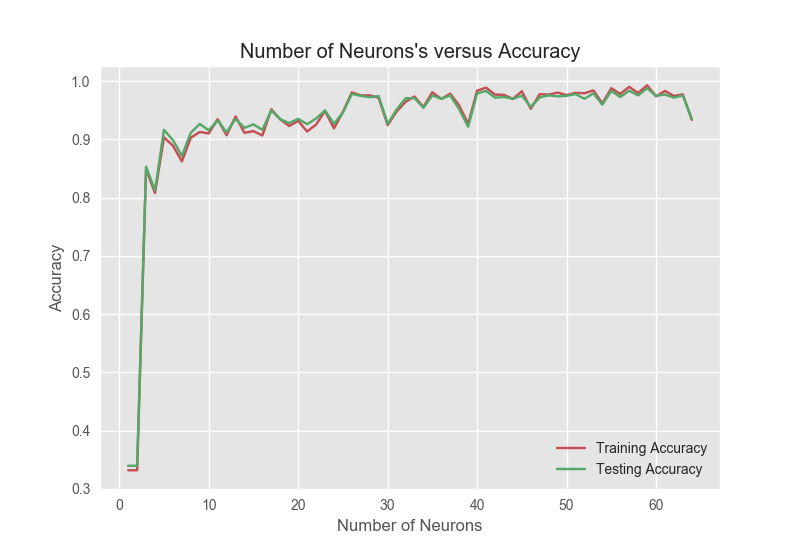
**Letter Recognition**

**Neural Nets**

Neural nets are one of the most popular and effective ways of learning data. In order to explore different features of neural nets, I ran experiments on the number of hidden layers and the number of neurons in each hidden layer, as well as changing the training set size to see if neurons need a lot of data and examples as it seems intuitively.

**Nursery**

**Figure:** Model complexity graph for neural nets on nursery dataset, exploring hidden layer number

**Figure:** The model complexity graph for neural nets on nursery dataset, exploring neuron number

Figures 123 and 123 show how changing the hidden layer number and neuron number on each hidden layer affects the learning. From the figure , it can be inferred that 2 is the optimal hidden layer number for the dataset even though 4 layers produce the same result, because if accuracies are the same, the simplest one should always be preferred. In figure , it can be seen that as the neuron number on each layer increases, the accuracies increase. Around 60 neurons, the classifier predicts almost all of the test data and the training data right. I have not applied cross validation in my neural net algorithm since building a model was taking already too long. Cross validating in the training set would building the classifier model each time from scratch, making this algorithm implausible for this homework’s sake.

**Letter Recognition**

**Boosting**

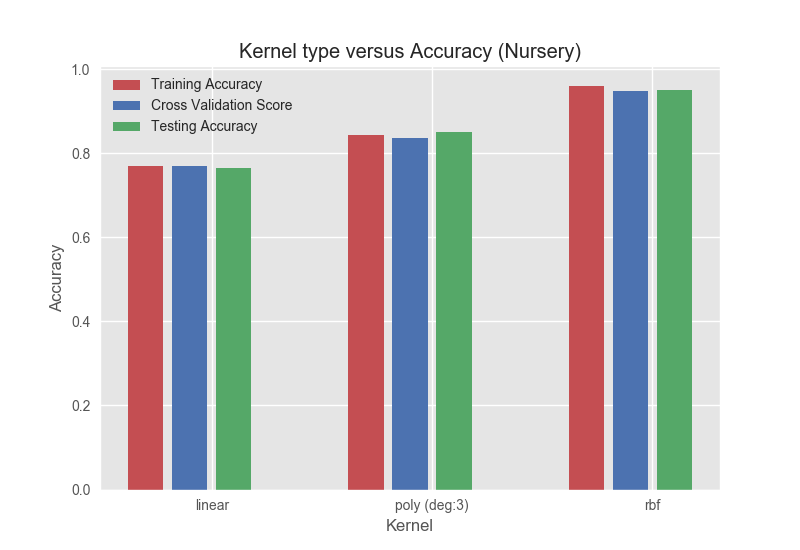
**Nursery**

**Letter Recognition**

**Support Vector Machine**

Support vector machine algorithms use different kernels to separate the instances into clusters. In my experiments with SVM’s, I explored different kernels and how each one performs on my datasets, as well as experimenting on the effect of increasing the training set size.

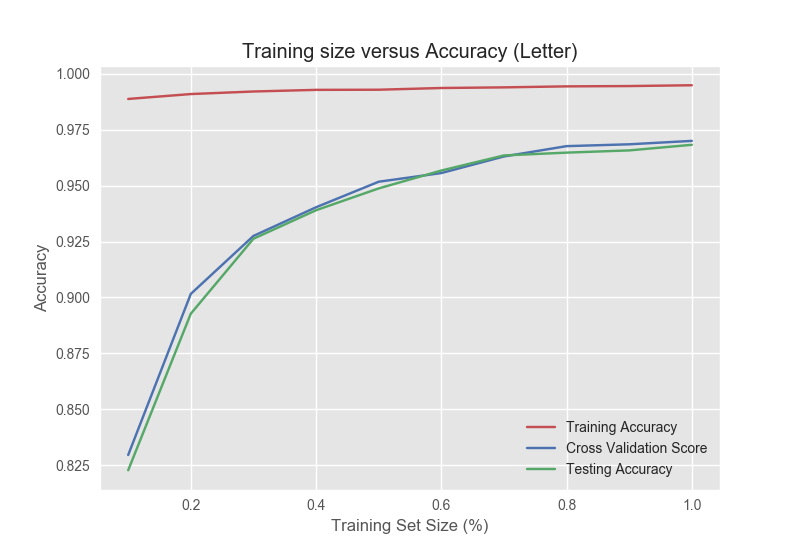
**Nursery**

**Figure:** The model complexity chart for SVM on the nursery dataset that explores different kernels.

The bar chart given in figure shows how different kernels of SVM work on the nursery dataset. As it can be seen, as the complexity of the kernel is increasing, the accuracy also increases for this dataset. While experimenting, the linear kernel took around half the time of the radial basis function kernel, suggesting that the complexity has a direct effect on the running time. The RBF kernel performs with 96.5% accuracy on this dataset while the linear kernel performs only with 78% percent.

**Figure:** Learning curve for SVM on the nursery dataset, using RBF kernel

The figure shows the learning curve for SVM. The red line (training accuracy) is consistently 2-3% higher than the green line, which is the testing accuracy. This suggests that overfitting is an issue here but not very significantly. Also, even though the 10% of the entire training set given, the algorithm starts to perform with a little more than 91% accuracy, revealing that unlike most of the algorithms, SVM does not highly depend on the training data. That’s because the SVM creates and estimator to classify and usually uses only relevant data to separate clusters.

**Figure:** Learning curve for SVM on the letter dataset, using the RBF kernel

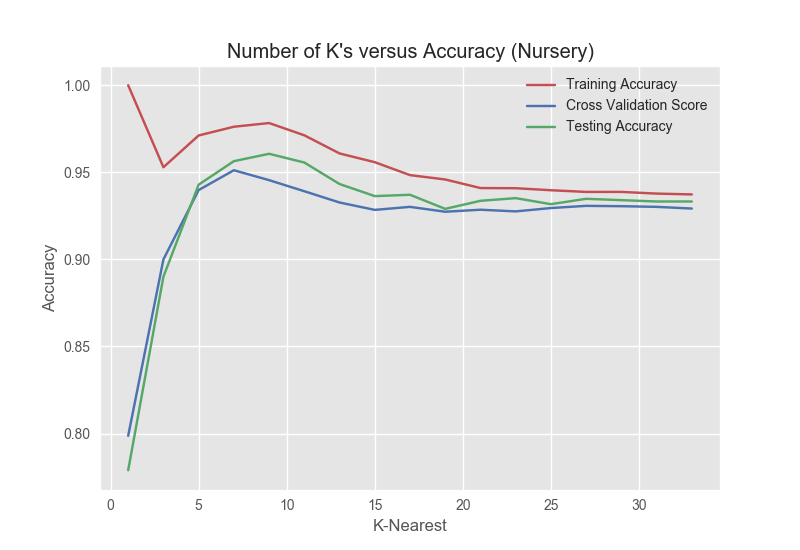
After testing different kernels on the letter dataset, checking the cross validation score and seeing how close it is to the testing score on RBF kernel as well as it’s high accuracy value, I decided that RBF was the optimal kernel for this dataset as well. The figure shows the learning curve for SVM. The most striking feature of this graph is the very high and consistent training accuracy curve. Compared to that, cross validation and testing accuracies start with a lower accuracy and gets closer to the training curve, however, cannot catch it. This suggests that the fitted SVM model overfits the testing data. Since the increasing training set size suggests that the more the data, the less overfitting, we can come to the conclusion that the complexity of the kernel might be overcomplicating the function. Another kernel with lesser complexity could have performed better on this dataset.

**Letter Recognition**

**K-Nearest Neighbors**

K-nearest neighbors is a classification algorithm that tries to classify the instance by finding the closest k-number of neighbors and identifying that the instance belongs to that cluster. The first step I have taken was to find the optimal k value for each dataset and then I plotted learning curves to see if changing the training set size has a significant effect on the accuracy.

**Nursery**

****

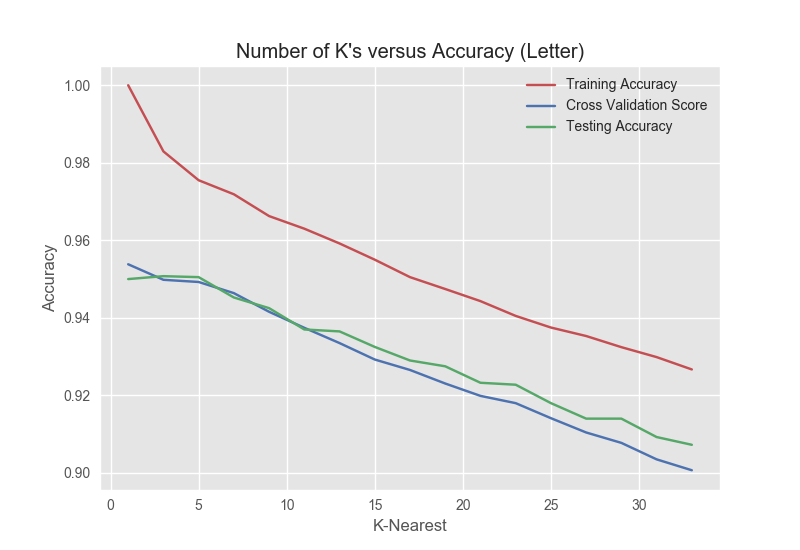
**Figure:** The model complexity graph for KNN algorithm on nursery dataset for k values.

Running the KNN algorithm on the nursery data set using a range of odd k values between 1 and 35, I obtained the model complexity graph given in figure. Looking at the graph, it can be seen that on lower k values, the trained model performs really well on the trained data, scoring close to 100% and over 95%. On the other hand, at these k values, the testing accuracy is low, suggesting the model performs much better on the seen data compared to unseen data. This is a typical example of overfitting since the model tailors its classifier very specifically for the trained data. The plot also reveals that the optimum k value for the nursery dataset is 9, and after this point, overfitting rate decreases and all three accuracy values start to converge but decrease, suggesting that increasing the k value all the way doesn’t increase the accuracy. In the end, looking at a limited amount of neighbors only make sense and as k increases, the algorithm starts to merge different clusters into each other, decreasing the accuracy and causing false positives.

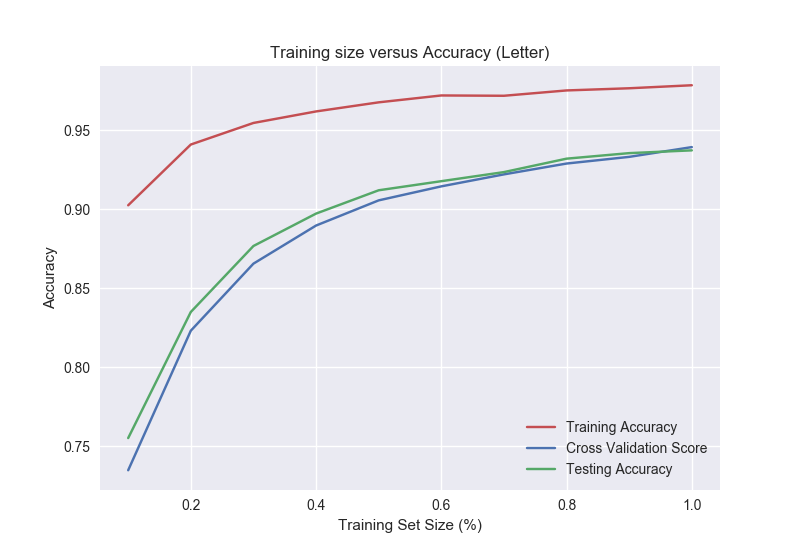
**Figure:** Learning curve of KNN on the nursery dataset

Looking at the figure, it can be observed that as the training set size increases, the accuracy of the model increases as well. However, it is possible to observe that the training accuracy is always higher than the testing accuracy, showing that the algorithm performs better on the seen data. A viable reason for this occasion is the KNN algorithm uses all the given training data rather than using the data to make an estimator, therefore, the entire training data is stored to run this algorithm, which leads to overfitting. Also, having very similar testing and cross validation curves show that there is not much bias in the testing set. In the final analysis, we see that, given the entire training set, the model performs with around 93.5% accuracy, suggesting that with 9-nearest neighbors, the labels can be classified and clustered with a significant accuracy.

**Letter Recognition**

**Figure:** The model complexity graph for KNN algorithm on the letter dataset for k’s

The figure shows the accuracy for KNN algorithm on the letter dataset, iterating over different k values ranging from 1 to 35. Unlike the previous dataset, the k value starts decreasing immediately, suggesting the optimum k value is around 3.

**Figure:** Learning curve for KNN algorithm on the letter dataset.

The learning curve in figure suggests that KNN has an increasing accuracy with the increasing training sample set size. This makes sense since KNN stores all the input instances, meaning the more possible neighbors, the more accurate the model gets. Other than that, overfitting is an issue like the previous dataset, since training accuracy exceeds the testing and cross validation data in a significant way.

**Conclusion**