Supervised Learning Assignment

1.1 Requirements

Jupyter Notebook 4.4.0, Python 3.6, Scikit-learn 0.21.3, Pandas 0.23.0, Numpy 1.14.3, Keras 2.2.4 with Tensorflow backend, Matplotlib 2.2.2

1.2 Dataset

Dataset 1

The first dataset is a modification of the "Otto Group Product Classification" from Kaggle (https://www.kaggle.com/c/otto-group-product-classification-challenge). Features are retail item features and classes are item categories. To allow model training to finish within a reasonable timeframe, this dataset is modified to balance to number of each label (see Data Pre-processing).

Dataset 2

The second dataset is the EEG eye state dataset from UCI machine learning repository (https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State), where features are facial features and class is eye state (open or closed). The dataset is used as it is without any modifications.

Problem Description

These two datasets are to address binary classification problem (EEG) and multiclass classification problem (Otto), separately. The contributors for datasets did not specify the meaning of each feature, which precludes feature engineering using domain knowledge. I select these datasets for the following reasons:

1. Both datasets are non-trivial. The smaller one (EEG) contains 15,000 data points with 15 features, and the larger one (Otto) has 47,000 data points with 93 features (pre-processing) and 40 (post-processing)
2. Both problems have practical application. The Otto dataset is from the retail industry. The inputs are 90 features and the output are item labels, which can be used for recommender system. The EEG dataset can be used for facial recognition, where eye blinking is important to differentiate real face vs. fake ones such as photos

Dataset Pre-processing

The original Otto dataset has the following counts:

```
Class_2    16122
Class_6    14135
Class_8     8464
Class_3     8004
Class_9     4955
Class_7     2839
Class_5     2739
Class_4     2691
Class_1     1929
```

The class with highest counts (Class_2) is about 8 times as much as the class with lowest counts (Class_1). If the original dataset is applied for model training, classes with lower counts are prone to be treated as noise, as machine learning models are better to identify similarity rather than abnormity. To address this issue, I reduce the dataset to include only the 4 classes with highest counts (Class_2, _6, _8, _3).

Otto dataset also contains a large number of features (93), which leads to a long model training time. To finish this assignment in a reasonable time frame, a new Otto dataset is created based on 40 randomly selected features out of the original 93 features.

Some key attributes are tabulated below.

	Otto Dataset	EEG dataset
Size	46725 x 40	14980 x 15
Features	All numeric	All numeric
Classification Type	Multiclass (4 classes)	Binary (2 classes)

Results

1. No cross validation is applied for neural network (time constraint), boosting (time constraint) and kNN (no training stage). 5-fold cross-validation is applied for decision tree and SVM
2. Training and testing accuracy, training and testing loss function are presented when available.

2.1 Decision Trees

Item	Description
Package	Sklearn.tree.DecisionTreeClassifier
Cross validation?	Yes, 5 folds
Base hyperparameter	Criterion: GINI Min_samples_split: 3 Min_samples_leaf: 2 Max_features: ‘auto’ Scoring: ‘accuracy’ The rest are the default params

Effect of Pruning

To test the effect of pruning on model results, an array of different maximal depth is selected as [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75]. Results for Otto dataset are shown in Figure 2.1.1, and for EEG dataset is shown in Figure 2.1.2.

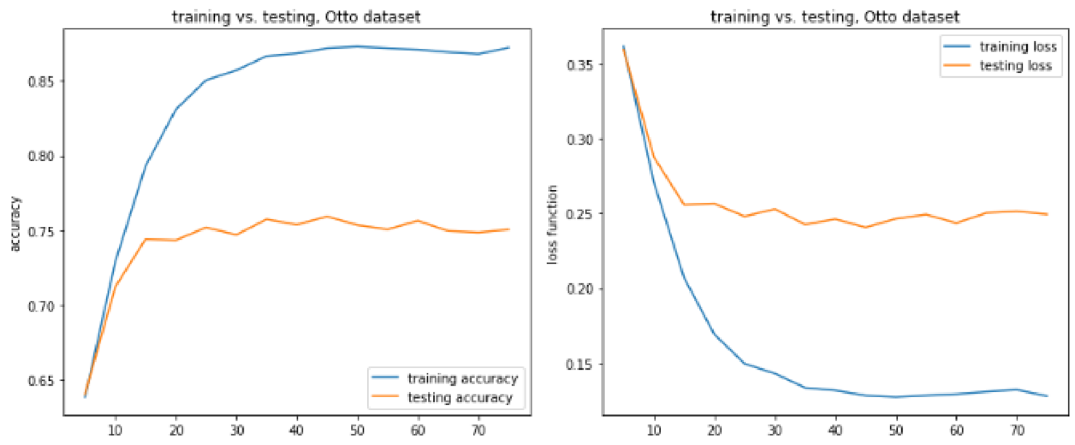

Figure 2.1.1 accuracy (left) and loss (right) for Otto dataset

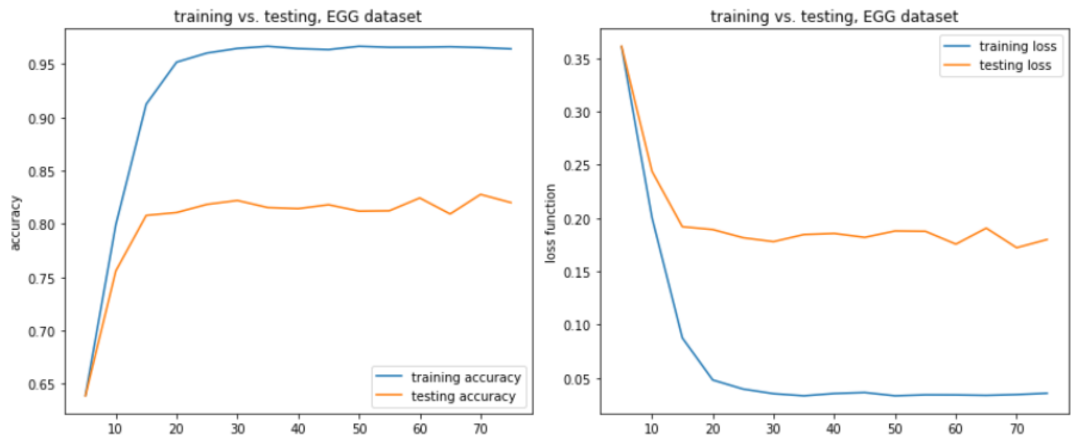

Figure 2.1.2 accuracy (left) and loss (right) for EEG dataset

X axis is the maximal depth. And the score is the average score from cross validation.

Results from both datasets show similar pattern: initially as maximal depth grows, both training and testing accuracy improve. When maximal depth grows beyond 20, testing accuracy reaches its plateau even though training accuracy continues to improve. In other words, testing score reaches the maximum before training score. This is a clear indication of overfitting. Even new information can still be learned from the training dataset (more depth growth for information gain), the new knowledge cannot be generalized when validated against the testing set, where deeper depth does not improve the testing accuracy score at all. For this case, maximal depth should be set at around 20 for the best testing accuracy.

Effect of Training and Testing Split

Maximal depth is set to be 20 for this test. Accuracy scores are computed for both training and testing sets, Training set portion is set as [0.2, 0.4, 0.6, 0.8, 0.9], and the remaining data points becomes testing set.

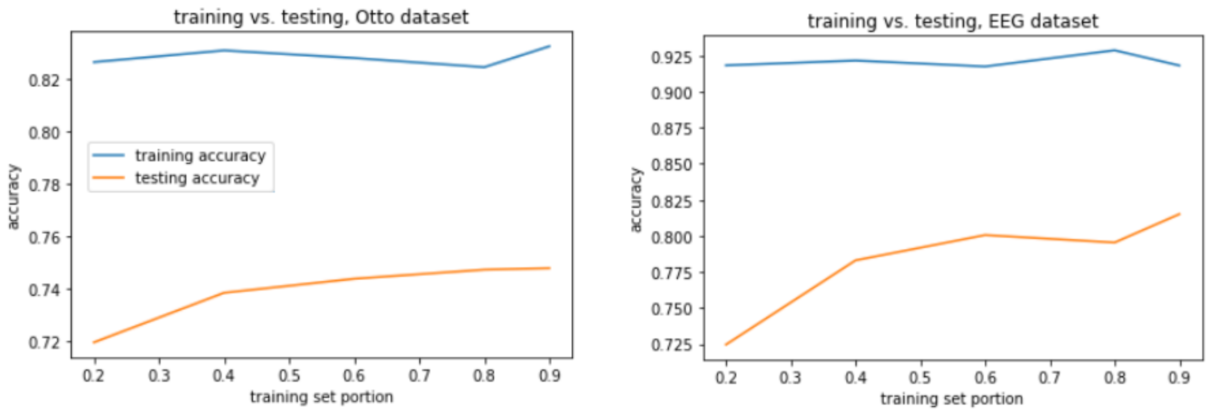

Figure 2.1.3 Accuracy score for Otto (left) and EEG (right)

It is expected that model is able to learn more knowledge as data points grow, and therefore improves testing score. Figure 2.1.3 meets the expectation. In both cases, testing scores improve as more data is used for training, while training score exhibits little variation.

2.2 Neural Networks

Item	Description
Package	Keras with Tensorflow backend
Cross validation?	No, train/test split = 0.8/0.2
Base hyperparameter	Number of hidden layers: 4 Number of neurons for each hidden layer: 1024, 512, 128, 32 Dropout between each layer: 0.2 Activation function: 'relu' for hidden layer, 'sigmoid' for output layer Loss: 'sparse_categorical_crossentropy' for Otto; 'binary_crossentropy' for EEG Epoch: 300 for Otto dataset, and 1200 for EEG dataset Early stopping: when loss does not improve for 40 epochs (EEG dataset)

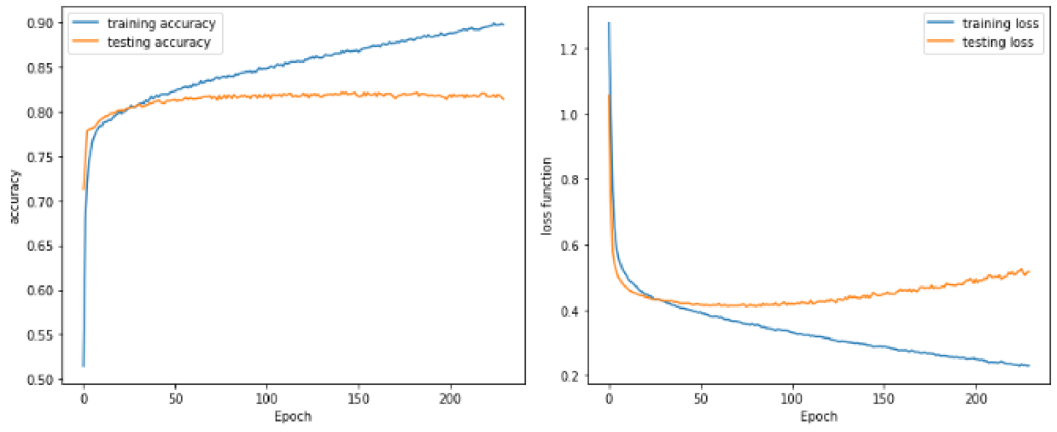

Figure 2.2.1 accuracy (left) and loss (right) for training and testing, Otto dataset

Training and testing accuracy scores (Figure 2.2.1, left) keep improving until epoch 50, after which further iterations only improve training scores. This observation echoes with the results seen in the previous section (Decision Trees), where new knowledge learned from the training dataset does not improve the model predictive capability.

Loss function (Figure 2.2.1, right) shows minimal loss for testing is reached at around epoch 50, which agrees with the observation in accuracy score plot. After epoch 50, the decreasing training loss does not reduce testing loss. Quite to the contrary, testing loss starts to increase after epoch 50. Here loss is calculated from probability which varies between 0 and 1. When loss increases but accuracy does not change, it means the model is getting less confident in predicting the labels, even though the labels are predicted correctly. For instance, a label is predicted as 1 when the probability of being 1 is 0.8. When the probability of being 1 decreases (for example, 0.7), loss increases, but the predicted label does not change given 0.7 is still greater than 0.5.

Figure 2.2.2 shows the results for EEG dataset. Model training stops at epoch 898 when it does not see any improvement in training loss.

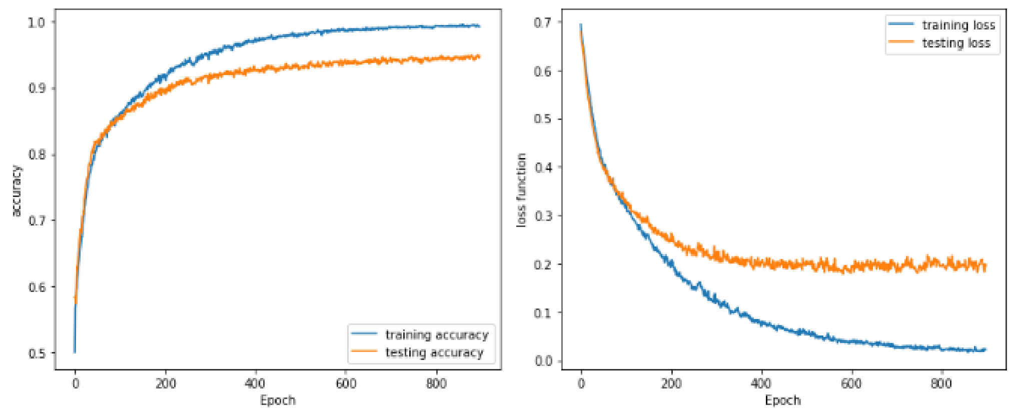

Figure 2.2.2 accuracy (left) and loss (right) for training and testing, EEG dataset

Unlike Figure 2.2.1, loss function for testing does not deteriorate after it reaches the minimum in Figure 2.2.2. Instead it remains at the minimal value beyond epoch 400, meanwhile training loss keeps decreasing. After epoch 400, training accuracy improve by 3% while during the same period testing accuracy only improves by 1%.

Effect of Training and Testing Split

Epoch is set to be 100 for Otto dataset and 600 for EEG dataset. Accuracy scores are computed for both training and testing sets, where training set portion is set to be [0.2, 0.4, 0.6, 0.8, 0.9], and the remaining data points becomes testing set.

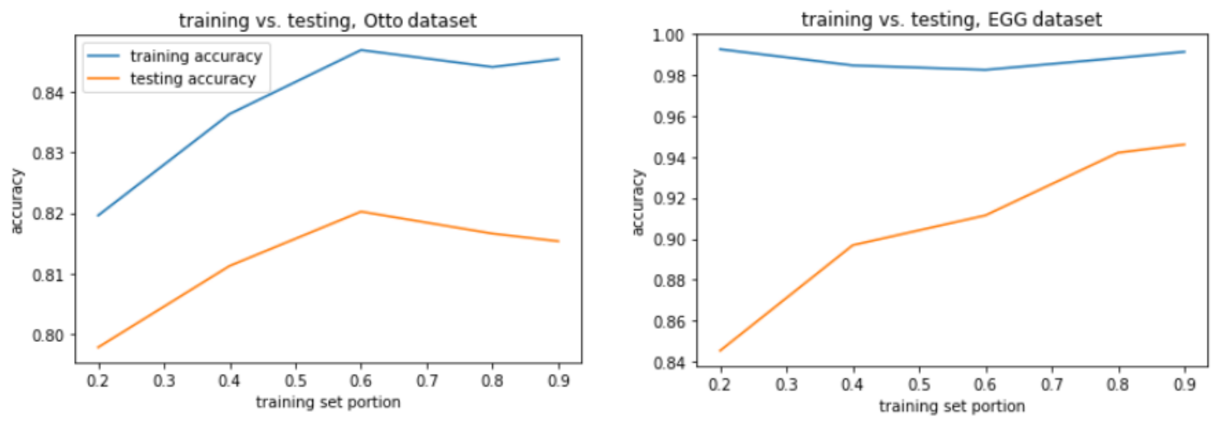

Figure 2.2.3 Accuracy score for Otto (left) and EEG (right)

For EEG dataset, training score almost does not change with training set portion, while testing score keeps improving. With more training data and fewer testing data, neural network is able to improve its predictive power, resulting in better testing accuracy.

What is quite interesting is the result of Otto dataset. Both training and testing scores increase with training set portion until it gets 0.6, then training score plateaus while testing score starts to decrease. This behavior is not logically reasonable and one possible explanation is that the number of epoch (100) used for training smaller datasets does not learn as much knowledge as it does for larger datasets.

2.3 Boosting

There are many boosting algorithms and packages available. In this assignment, I only use the Gradient Boost package from Scikit-learn.

Item	Description
Package	Sklearn.ensemble.GradientBoostingClassifier
Cross validation?	No. train/test split: 0.8/0.2
Base hyperparameter	N_ estimators: 300 (Otto), 1200 (EEG) max_depth: 3 The rest are the default params

One interesting, and also attractive, behavior of boosting algorithm is that it usually does not overfit, if the weak learner does not overfit. In this case, the weak learner is a tree model with depth (max_depth in model settings) = 3. Recall the results obtained in 2.1 Decision Trees where overfitting happens at max_depth=20, a tree model with maximal depth 3 underfits the data. Therefore overfitting for both datasets using boosting is unlikely here.

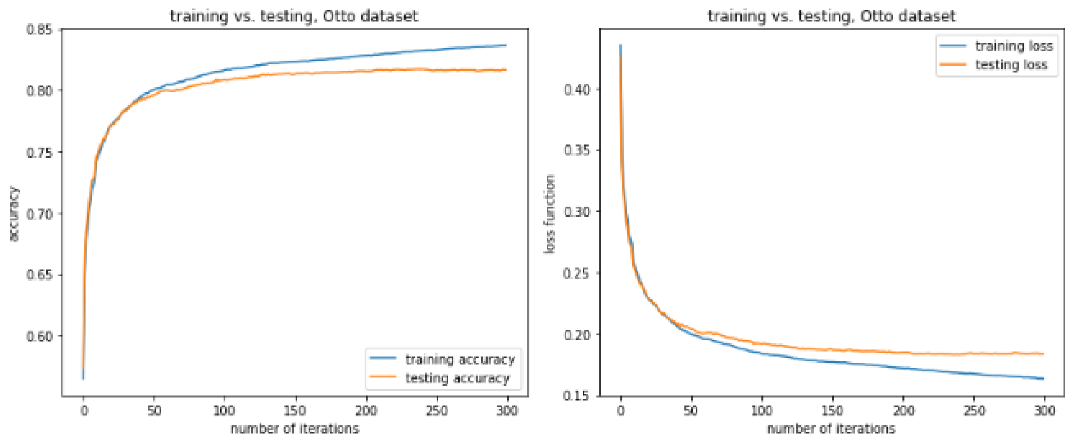

Figure 2.3.1 accuracy (left) and loss (right) for training and testing, Otto dataset

Results in both Figures 2.3.1 (Otto dataset) and 2.3.2 (EEG dataset) agree with the argument. Both training and testing accuracies are improving as number of iterations increases, while training and testing losses are decreasing. Especially for the EEG dataset, after 1200 iterations there are still noticeable improvement in accuracy and loss for both training and testing.

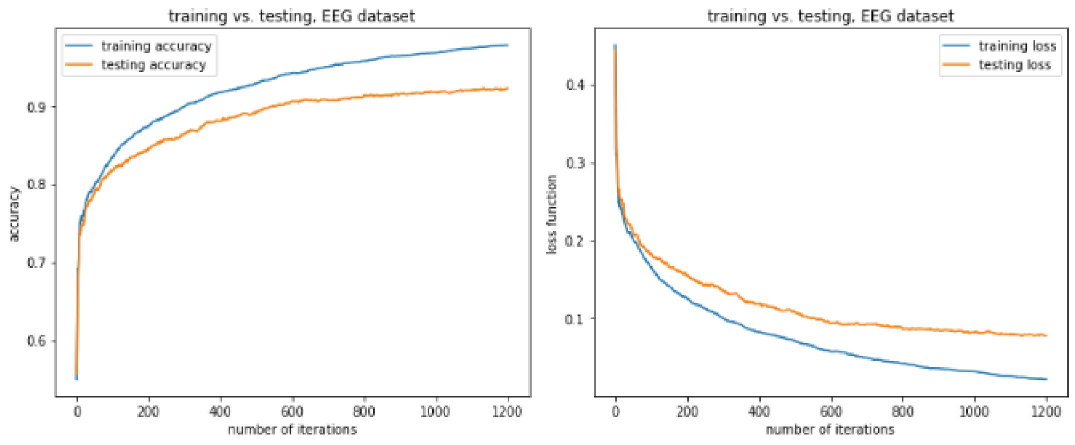

Figure 2.3.2 accuracy (left) and loss (right) for training and testing, EEG dataset

Effect of Pruning

The above results are all based on a base learner with maximal depth = 3. To understand how depth variation would change model results, a list of different maximal depth is used ([3, 10, 20, 30]). To finish training within a reasonable time frame, number of iterations is reduced to 50 for Otto dataset, and 800 for EEG dataset. It should be noted that these numbers (50 and 800) are far below the iterations in Figures 2.3.1 and 2.3.2, therefore each boosting model is likely to underfit.

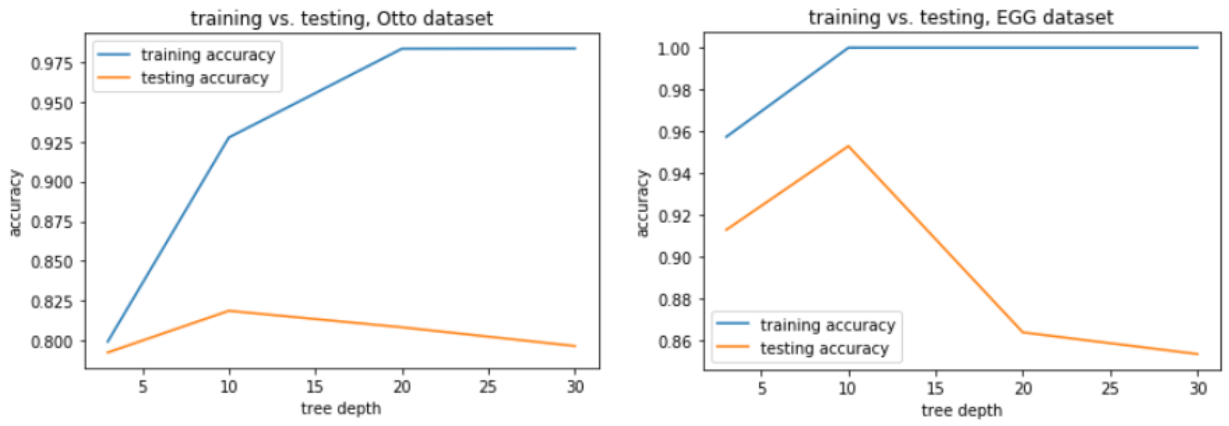

Figure 2.3.3 Accuracy score for both Otto (left) and EEG (right)

When maximal depth increases, training score increases monotonically for both cases. Testing accuracy increases until maximal depth gets 10, and decreases afterwards. When maximal depth <= 10, increment in maximal depth counterbalances the effect of short iterations and improves both training and testing scores. After maximal depth reaches 10, the base learner starts to overfit the data and therefore results in a decrease in testing score (overfitting is much worse in EEG than Otto).

Effect of Training and Testing Split

Max depth is set to be 3 for both datasets, and max number of iterations is kept the same as previous section (300 for Otto and 1200 for EEG). Accuracy scores are computed for both training and testing sets, where training set portion is set as [0.2, 0.4, 0.6, 0.8, 0.9], and the remaining data points are used for testing.

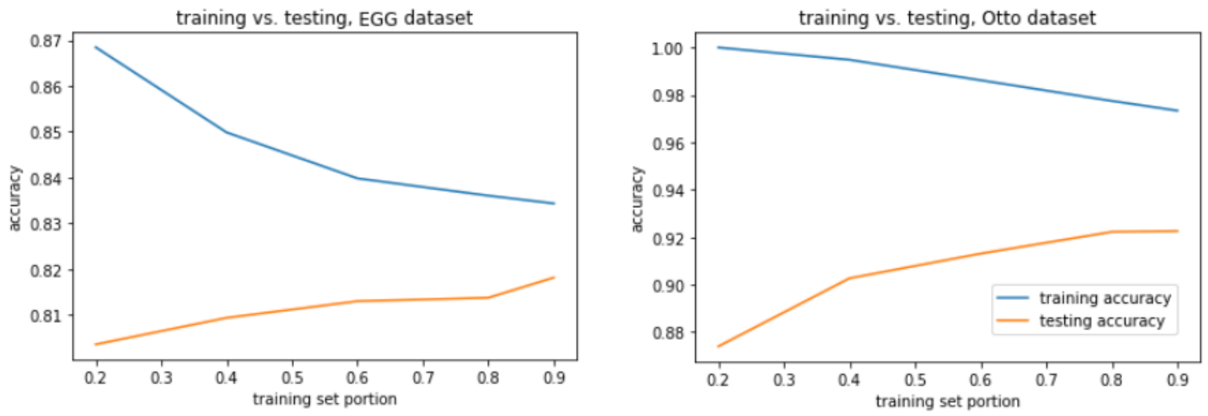

Figure 2.3.4 Accuracy score for both Otto (left) and EEG (right)

Training accuracy score decreases monotonically as training set portion increases. This could be attributed to the fixed iteration number imposed for both datasets. As can be seen in Figures 2.3.1 and 2.3.2, training accuracy is still increasing even after the max number of iterations (300 for Otto and 1200 for EEG) is reached. It is expected that given enough iterations, training accuracy scores for higher training set portion will increase to the level of lower training set portion, just as Figure 2.1.3.

2.4 Support Vector Machines

For Support Vector Machines, two kernel functions are tested: RBF and Linear. For each kernel function, an array of penalty C is selected.

Item	Description
Package	Sklearn.svm.SVC
Cross validation?	Yes, 5 folds
Base hyperparameter	Use default params

Note: RBF and Linear kernels on Otto dataset is extremely slow and is not able to finish in time for this assignment. As a result, the following plots are only for EEG dataset.

RBF Kernel

Support Vector Machine with RBF kernel results are shown below. X axis is the penalty value which ranges from 0 to 15000.

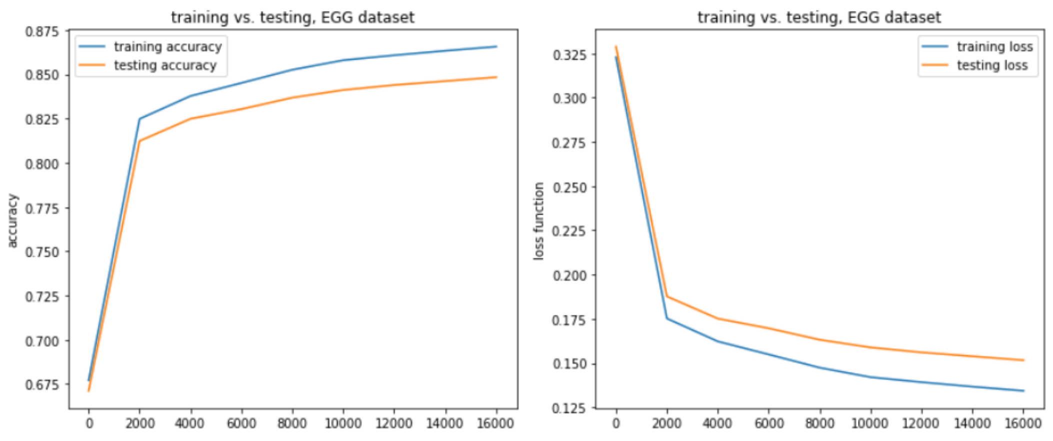

Figure 2.4.1 accuracy (left) and loss (right) for training and testing, EEG dataset

Even with C value up to 15000, both training and testing scores are still increasing. I did not attempt to test even larger C value because long time is required for model to converge with a large C value.

I have never encountered such large C value in my work. It is interesting to dive deeper into the dataset and try to understand why it happens, but it is beyond the scope of this assignment.

Linear Kernel

Support Vector Machine with linear kernel results are shown below. X axis is the penalty value which ranges from 0 to 1000.

Similar to the RBF kernel, both training and testing scores improve with penalty. One significant difference is the accuracy value: accuracy value with linear kernel is about 20% lower than the model with RBF kernel. However, it requires significantly longer time for RBF kernel than linear kernel, when it comes to model training.

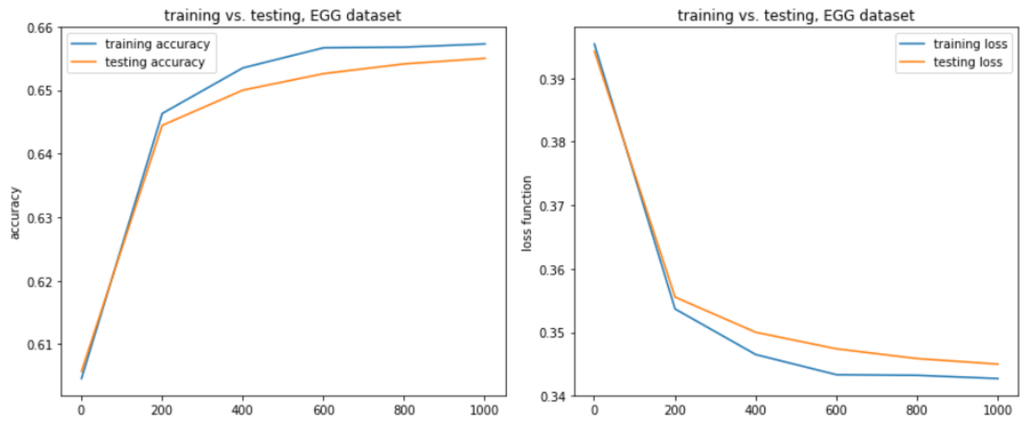

Figure 2.4.2 accuracy (left) and loss (right) for training and testing, EEG dataset

Kernel Comparison

Linear kernel generates hyperplanes for data classification, while RBF kernel generates transition zones instead. This fundamental difference leads to a longer training time for RBF kernel but also better training score.

Effect of Training and Testing Split

With EEG dataset, both RBF and linear kernels are tested. Accuracy scores are computed for both training and testing sets, where training set portion is set as [0.2, 0.4, 0.6, 0.8, 0.9], and the remaining data points are used for testing.

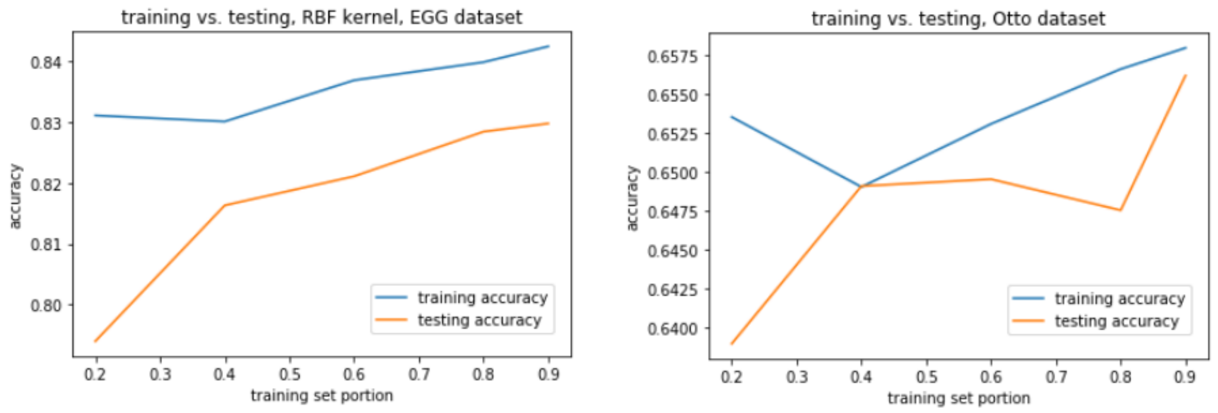

Figure 2.4.3 training and testing accuracy scores. RBF kernel (left), linear kernel (right)

It is relatively small changes in training scores for both cases (less than 1%). Testing scores exhibits improvement as more data points are used for training. This observation largely aligns with observations made for other models.

2.5 k-Nearest Neighbors

KNN is an instance-based model. There is no training accuracy or loss. Only testing accuracy and loss is computed.

Item	Description
Package	Sklearn.neighbors.KNeighborClassifier
Cross validation	In model settings, yes, but in fact kNN does not meet the definition of cross validation because there is no training stage
Base hyperparameter	Use default params

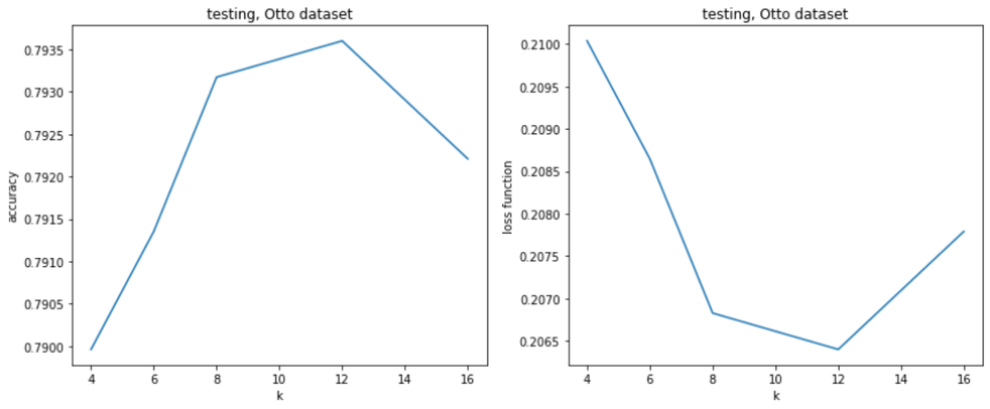

Figure 2.5.1 accuracy (left) and loss (right) for testing, Otto dataset

For the Otto dataset, variation in k value does not change testing accuracy or loss significantly. For the k range tested (4~16), there is less than 1% accuracy difference observed. The best accuracy occurs at k=12.

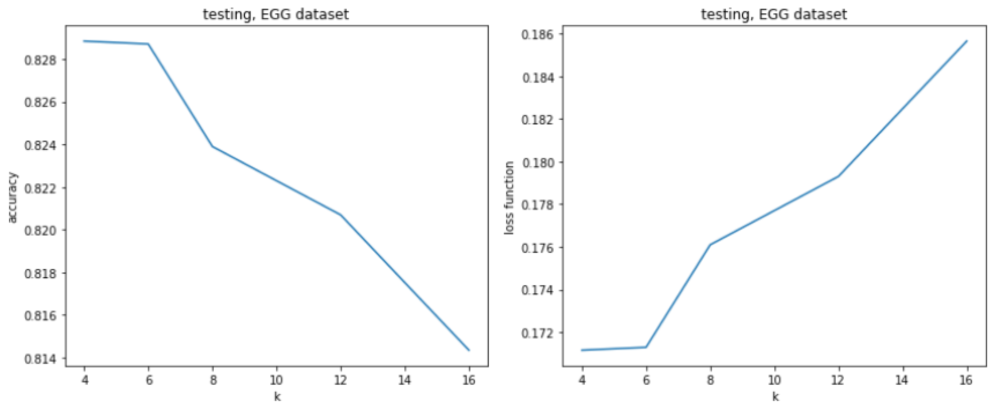

Figure 2.5.2 accuracy (left) and loss (right) for testing, EEG dataset

Similar to the Otto dataset, accuracy and loss variation is also very small in the same range of k for the EEG dataset. Unlike the Otto dataset, fewer neighbor numbers seem to help improve accuracy. The best accuracy occurs where k=4 and 6.

The fact that testing scores are in general insensitive to the number of neighbors indicate that data points with the same label in both datasets tend to cluster together in hyper dimensions. Although it cannot be visualized directly, techniques such as PCA or tSNE could be used for dimension reduction in the first place.

Effect of Training and Testing Split

There is no accuracy score difference among different training and testing split since k-Nearest Neighbors is an instance-based learning algorithm that learning stage does not generate any information.

Analysis

3.1 Model Comparison

The following table summarizes models in this assignment.

Model Name	Training Time*	Accuracy **	Comments
Decision Trees	Low	6	Easy to use, but it is prone to overfitting. Model training is fast
Neural Networks	Mid	1	1. Setting up neuron layers is a science and an art. The whole process can be time-consuming 2. It is quite surprising to see for both datasets, Neural Networks can obtain really good accuracy score. I always believed neural networks can outperform other ML algorithms only when many (millions of) data points are available, but it is clear not the case from this assignment
Gradient Boosting	Mid	1	1. It is relatively fast, and can usually outperform other ML algorithm in lots of cases 2. If weak learner is chosen carefully, this algorithm almost never overfits
Support Vector Machine (RBF kernel)	High	3	1. Training can be very slow in high dimensional space, probably because it takes a long time to converge at high penalty value 2. Kernel function is very important for model accuracy. In this assignment, SVM with RBF kernel obtains accuracy 20% higher than Linear kernel
Support Vector Machine (linear kernel)	High (but lower than RBF kernel, everything else being equal)	5	
k-Nearest Neighbors	High	3	Very cheap to train, but very expensive to predict, compared to other algorithms. It is probably not suitable for situation where real time prediction is required

* kNN does not require any time for training, however the prediction stage could take a long time depends on the size of datasets

* Accuracy is ranked from 1 to 6. 1 is the most accurate and 6 is the least accurate

Under the current settings for this assignment, gradient boosting and neural network obtain the highest training and testing accuracy for both datasets. However, it takes smaller effort to configure gradient boost model and shorter time for training, as compared to neural network. When it comes to larger datasets (millions of data points), the advantage of neural network might be more prominent: numerous applications show that neural network can outperform gradient boost in terms of accuracy with lots of training data.

The performance of support vector machine relies greatly on the kernel function. This assignment shows the accuracy score difference can be as much as 20% because of kernel choice. It should also be noted that support vector machine is a relatively low efficient algorithm and might not be suitable for large datasets.

Decision tree is a fast algorithm but is prone to overfitting. Compared to gradient boosting, which uses a ‘short’ tree as the base learner, decision tree usually does not produce scores as high as gradient boosting. An ensemble of trees can often improve model performance significantly.

k-nearest neighbors differs from the aforementioned algorithms because it does not have a model training stage but requires real-time distance computation once prediction is needed. Therefore it could be quite resource-demanding for prediction.

3.2 One-dimensional Grid Search

To finish modeling training within a reasonable timeframe, only one hyperparameter is allowed to change while the remaining are kept constant (for example, only penalty C is allowed to change in Support Vector Machine). In real world, hyperparameters form a hyperspace with much higher dimensions and to search for the best combination of hyperparameters usually requires significant computational resources. A practical approach for grid search (either exhaustive or randomized) is always a balanced effort between model accuracy (or any other metrics that is appropriate) and available resources.

3.3 Dataset Comparison

Even though EEG dataset has less than half data points (~20,000 vs. ~ 47,000) than Otto dataset, testing accuracy for all models in the assignment all shows a better accuracy of EEG than Otto. Among the many factors that could contribute, Curse-of-Dimensionality is the one that deserves attention.

Curse-of-Dimensionality suggests when dataset dimensionality increases, the number of data points needs to increase exponentially in order to keep the predictive power of a machine learning model. EEG dataset has 14 features, and is substantially fewer than Otto dataset which has 40 features. From the point of feature numbers, it requires Otto dataset far more data points (in terms of magnitudes) to be statistical representative for each class than EEG dataset, not to mention Otto has 4 classes while EEG only has 2 classes.