SCOTT BAKER

SCOTT.BAKER@COLORADO.EDU

PROJECT FOR STAT 5610 - STATISTICAL LEARNING

APR. 29, 2020

CLASSIFICATION OF THE HUMAN ACTIVITY RECOGNITION DATA

CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXPLORATORY DATA ANALYSIS

1. INITIAL DATA EXPLORATION (I)

To begin discussion of the Human Activity Recognition (HAR) dataset [3], Figure 1 is a set of density histograms of the mean linear body acceleration from the training data for each axis during the labeled activity `LAYING`.

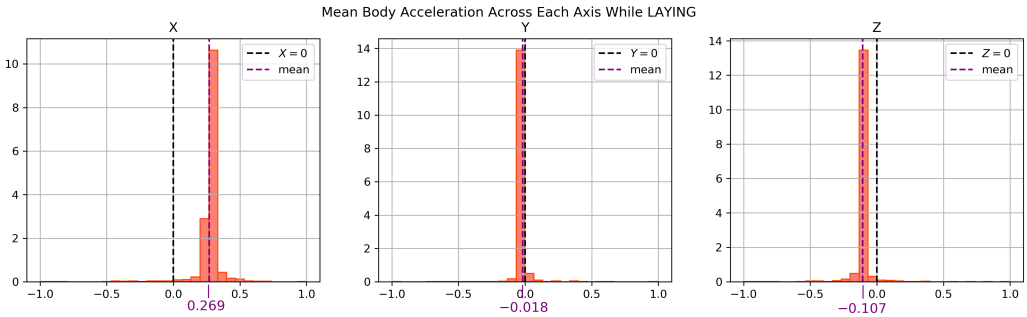


Figure 1: Density plots of the mean linear body acceleration while `LAYING` for each axis (x, y, z)

Figure 1 also includes a vertical marker of the aggregate mean for each axis. These aggregate mean values are included in Table 1.

AXIS	AGG. MEAN
x	0.269
y	-0.018
z	-0.107

Table 1: Aggregate mean values of the mean linear body acceleration while `LAYING` for each axis (x, y, z)

It is important to note that while a person is lying down, their phone still experiences acceleration due to gravity. Therefore, it makes sense that these aggregate mean values are not necessarily zero even though the person is not moving during the labeled activity `LAYING`.

2. INITIAL DATA EXPLORATION (II)

Figure 2 is a set of density histograms showing mean body angular velocity from the training data for each axis (x, y, z) during the labeled activity `LAYING`.

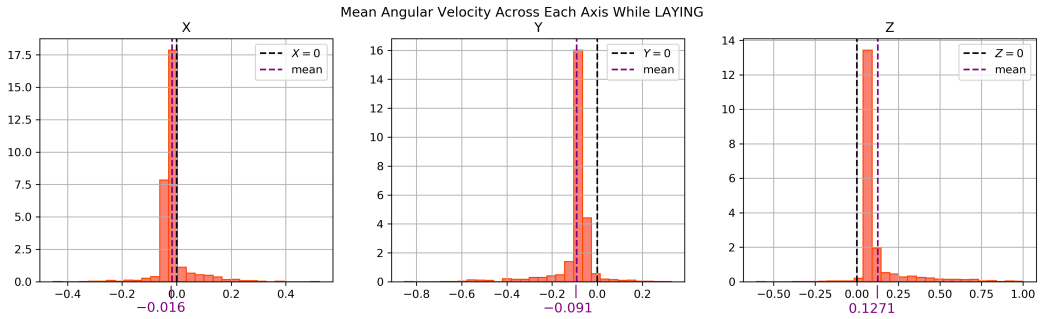


Figure 2: Density plots of the mean body angular velocity while LAYING for each axis (x, y, z)

Similarly, Table 2 includes the aggregate mean marked in Figure 2.

AXIS	AGG. MEAN
x	-0.016
y	-0.091
z	0.127

Table 2: Aggregate mean values of the mean body angular velocity while LAYING for each axis (x, y, z)

Much like the aggregate mean body linear acceleration, the aggregate mean body angular velocity is not necessarily zero during the activity LAYING due to the presence of gravity.

3. Linear Coupling Between Sensor Measurements

Another exploratory piece of information about the training data is computing a pairwise correlation matrix. Figure 3 shows a heatmap and colorbar describing the nature of the pairwise correlations between the 561 measurements for each instance in the training data.

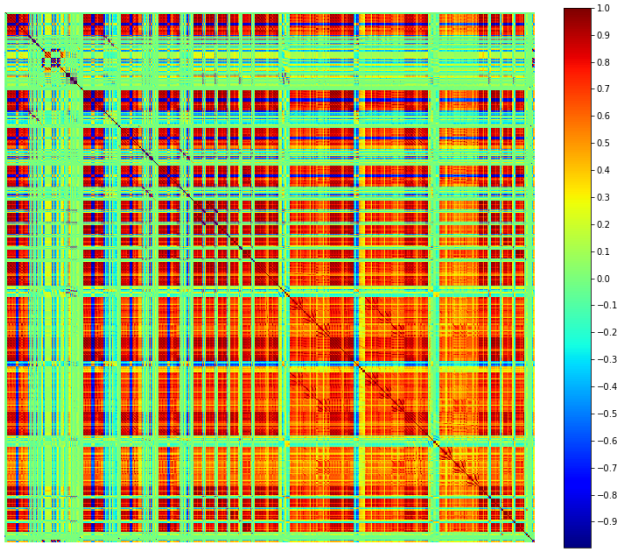


Figure 3: Heatmap of pairwise correlations between all 561 features in the training data

Features are correlated (positively or negatively) with other features, seen either in blue or red-brown. When two features are correlated, this could mean that when values vary across one dimension, they vary in a similar way across another dimension. With the complexity of the kernel Support Vector Machine (SVM), the hope is that a balance can be found between complexity (to handle the correlated features) and the curse of dimensionality [8].

CLASSIFICATION OF ACTIVITIES USING SUPPORT VECTOR MACHINE

4. HYPERPARAMETER TUNING

To begin the discussion of finding the right value of the slack penalty λ, Figure 4 shows the classification accuracy on the testing set for $n = 100$ values of λ between 1 and 4000. For this classification, the radial basis function kernel will be used and defined as follows:

$$\kappa(x_i, x_j) = exp\{-\gamma||x_i - x_j||^2\} \tag{1}$$

In the first radial-kernel SVM, the value of γ is $\frac{1}{N}$, where N is the number of instances in our training set ($N = 7352$).

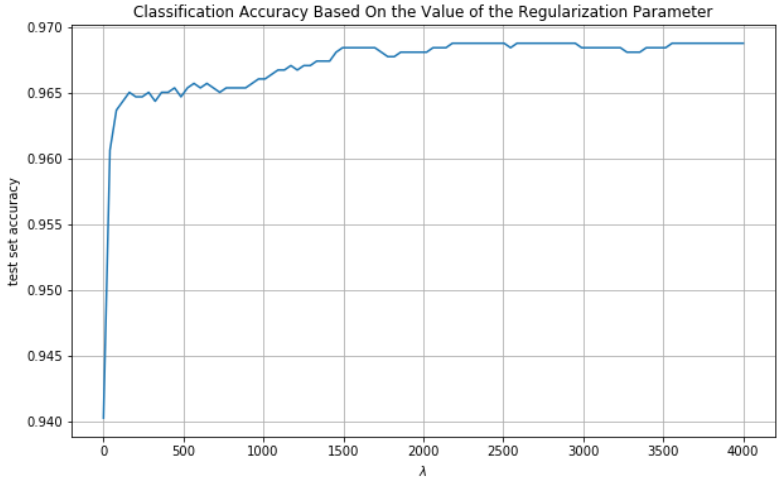


Figure 4: Classification accuracy on the testing set for $n = 100$ values of λ such that $1 \leq \lambda \leq 4000$

From Figure 4, the testing set accuracy of the radial-kernel SVM increases as the slack penalty λ increases. However, because of the nature of the radial kernel, this means that there is potential over-fitting and a value of $\lambda = 4000$ should not be used. For better generalization [4], accuracy from a 5-Fold Cross-validation using the training set for $n = 100$ values of λ between 1 and 4000 is plotted in Figure 5. Again, the value of γ is $\frac{1}{N}$, where N is the number of instances in the training set ($N = 7352$).

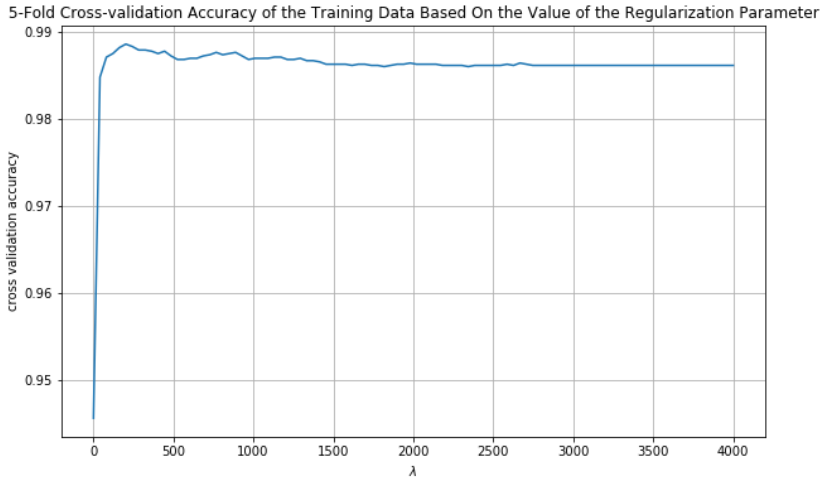


Figure 5: 5-Fold Cross-validation accuracy for $n = 100$ values of λ such that $1 \leq \lambda \leq 4000$

From Figure 5, it is concluded that a value of λ between 1 and 500 should generalize better. Specifically in this case, $\lambda = 202.969$ is optimal. However, to be more precise, a grid search [1] with λ and γ is performed. The first grid search focuses on $n_1 = 20$ values of λ between 1 and 500 and $n_2 = 20$ values of γ between $\frac{1}{12*7352} \approx 0.00001133$ and $\frac{32}{7352} \approx 0.00435$. Figure 6 shows a heatmap of the testing set accuracy for this 400-point grid search.

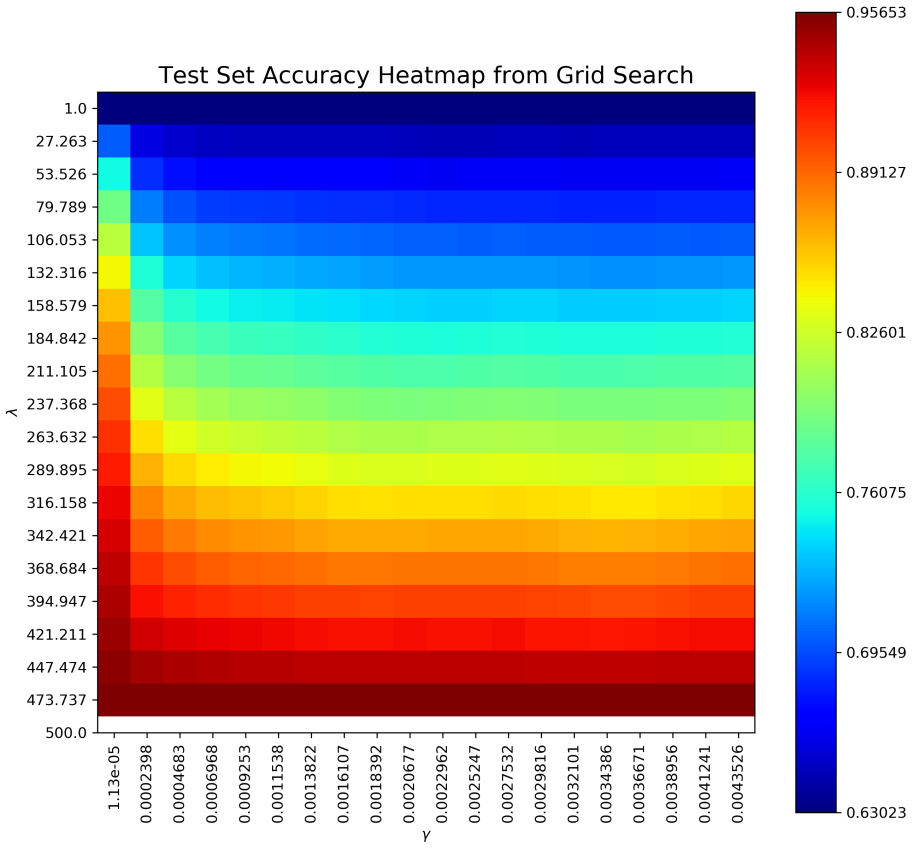


Figure 6: Grid search testing set accuracy for $n_1 = 20$ values of λ such that $1 \leq \lambda \leq 4000$ and $n_2 = 20$ values of γ such that $0.00001133 \leq \gamma \leq 0.00435$

The heatmap in Figure 6 suggests that the optimal values are $\lambda = 473.373$ and $\gamma = 0.0001133$. In addition to evaluating the radial-kernel SVM grid search on the testing set, 5-Fold Cross-validation grid search accuracy is plotted in Figure 7.

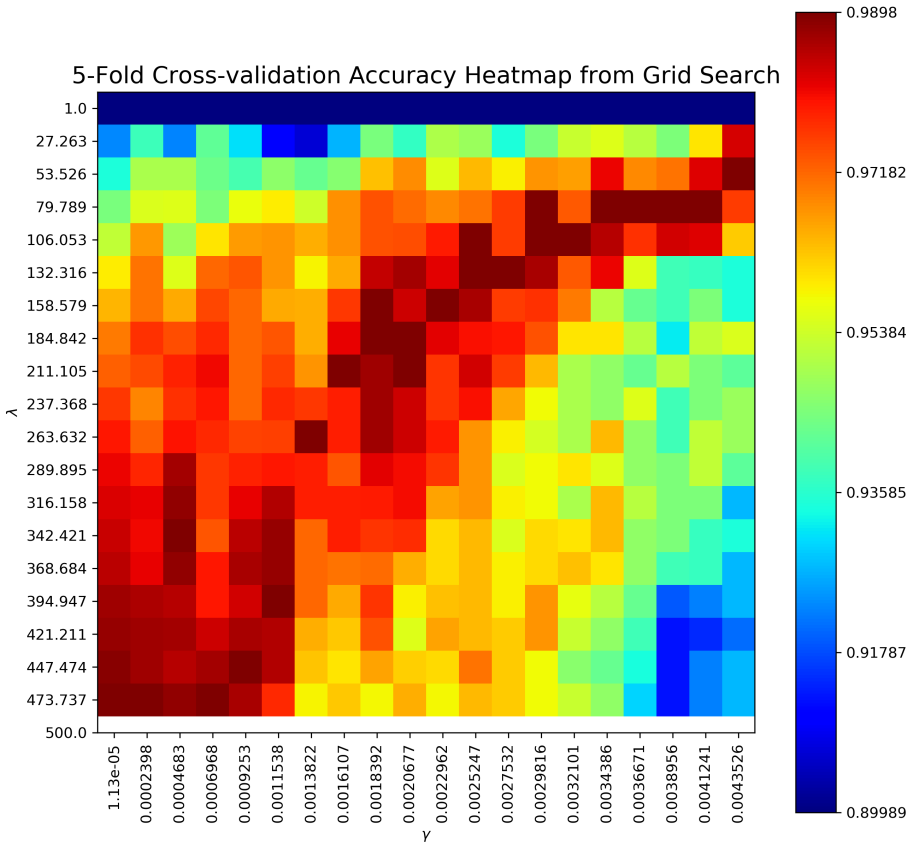


Figure 7: Grid search 5-fold Cross-validation accuracy for $n_1 = 20$ values of λ such that $1 \leq \lambda \leq 4000$ and $n_2 = 20$ values of γ such that $0.00001133 \leq \gamma \leq 0.00435$

From the heatmap in Figure 7, the optimal values for the hyperparameters are $\lambda = 79.789$ and $\gamma = 0.0034386$.

In summary, Table 3 is a comparison between different values of λ and γ. In addition to values, this table includes the evaluation method used (test set or k-fold), accuracy, and pros/cons of using each set of values for the optimal choice λ^* and γ^*.

ID	λ	γ	EVAL	EVAL ACC.	PROS	CONS
1	4000	0.000136	test set	96.87%	more accurate than 3	overfitting
2	202.969	0.000136	5-fold cv	98.85%	accurate	large λ
3	473.737	0.000136	grid search test set	95.65%	none	overfitting
4	79.789	0.0034386	grid search 5-fold cv	98.98%	accurate	test set accuracy

Table 3: Summarizing different choices of hyperparameters in radial-kernel SVM

From Table 3, the choice of optimal hyperparameters comes from ID 4. This set, $\lambda = 79.789$ and $\gamma = 0.0034386$, has the smallest value of λ, which should help the classifier avoid overfitting the data. It has the largest value of γ, which should help distinguish points that might be close to each other (see Eqn. (1)). Using 5-Fold Cross-validation within the grid search allows this classifier to generalize well to new data.

5. Evaluation of Optimal λ^* and γ^*

The choice of $\lambda^* = 79.789$ and $\gamma^* = 0.0034386$ leads to 96.64% accuracy on the testing data. Further, Table 4 shows the confusion matrix from the testing set evaluation.

	LAYING	SITTING	STANDING	WALKING	WALK DOWNST.	WALK UPST.
LAYING	537	0	0	0	0	0
SITTING	0	442	47	0	0	2
STANDING	0	12	520	0	0	0
WALKING	0	0	0	492	3	1
WALK DOWNST.	0	0	0	3	403	14
WALK UPST.	0	0	0	16	1	454

Table 4: Confusion matrix from evaluating optimal radial-kernel SVM classifier

As a refresher, this confusion matrix is structured [14] in a way that puts true class labels in the rows and predicted class labels in the columns. The value in each element is the number of examples that fall into each category. For example, the entry 537 in the top left is the number of instances of the true label LAYING being classified as LAYING by our SVM; these are correct classifications. Further, the element at $[2, 6]$ (row: SITTING, column: WALK UPST.) means that there are 2 instances of the true label SITTING being classified as WALKING UPSTAIRS by our SVM; these are misclassifications.

6. Discussion of Confusion Matrix

In general, there is one major pattern that emerges from the confusion matrix in Table 4. This pattern is that static activities (LAYING, SITTING, STANDING) are generally classified as static activities and that dynamic activities (WALKING, WALKING UPSTAIRS, WALKING DOWNSTAIRS) are classified as dynamic activities. There are only 2 cases in which a static activity (SITTING) is classified as dynamic (WALKING UPSTAIRS) and no cases in which a dynamic activity is classified as static. Within static activities, there seems to be issues distinguishing between SITTING and STANDING. There are 47 instances of SITTING that are classified as STANDING and 12 instances of STANDING that are classified as SITTING. The intuition behind these misclassifications is that in the real world, the orientation of a phone while sitting and while standing could be exactly the same. Regarding the dynamic activities, the optimal SVM misclassifies 14 instances of WALKING DOWNSTAIRS as WALKING UPSTAIRS and 16 instances of WALKING UPSTAIRS as WALKING. These dynamic misclassifications are less intuitive. However, there could be similar movement of a person's phone when the person is walking upstairs, walking downstairs, and walking regularly.

IS THE DATA LINEARLY SEPARABLE?

7. 3D Data Visualization (I)

Figure 8 contains two three-dimensional plots showing different perspectives of the three coordinates of mean body linear acceleration in the training data. Static activities are denoted with dots while dynamic activities are denoted with triangles.

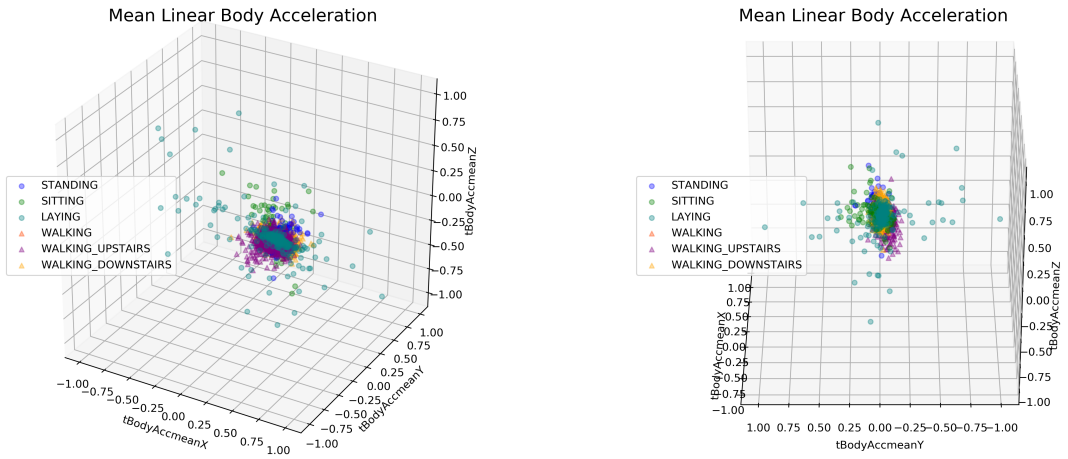


Figure 8: Two perspectives of a three-dimensional plot of (x, y, z) coordinates for mean body linear acceleration in the training set

8. Analysis of 3D Scatterplot (I)

From Figure 8, it appears that there is a difference between static activities and dynamic activities. Although they do not seem linearly separable, the static activities seem to have higher variability on the x axis, lower values of z, and lower values of y. For more through analysis, Table 5 shows the comparison between aggregate mean values of (x, y, z) in static versus dynamic activities.

AXIS	STATIC	DYNAMIC
x	0.273	0.275
y	-0.016	-0.020
z	-0.107	-0.112

Table 5: Comparison between static and dynamic activity aggregate mean values of mean body linear acceleration for each axis (x, y, z)

9. 3D Data Visualization (II)

Much like Figure 8, Figure 9 is two three-dimensional plots showing different perspectives of the three-coordinates mean body angular velocity in the training data. Again, static activities are denoted with dots while dynamic activities are denoted with triangles.

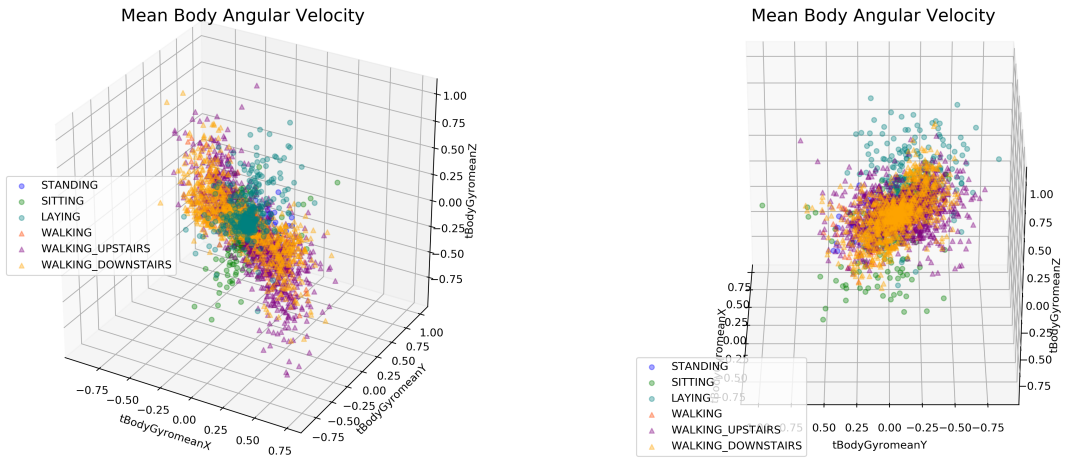


Figure 9: Two perspectives of a three-dimensional plot of the (x, y, z) coordinates for mean body angular velocity in the training set

10. Analysis of 3D Scatterplot (II)

Table 6 compares the aggregate mean values of mean body angular velocity between static and dynamic activities.

AXIS	STATIC	DYNAMIC
X	-0.026	-0.027
Y	-0.077	-0.075
Z	0.095	0.076

Table 6: Comparison between static and dynamic activity aggregate mean values of mean body angular velocity for each axis (x, y, z).

From this, it appears that the dynamic activities have a lower average value of the mean body angular velocity across the z axis. However, to answer this question more accurately, further analysis of a new classification problem, `STATIC` versus `DYNAMIC` activities, is provided in the following section.

SUPPLEMENTAL ANALYSIS: REDUCED KERNEL SVM CLASSIFICATION

Instead of training a SVM to distinguish between six classes of activities (`SITTING`, `STANDING`, `LAYING`, `WALKING`, `WALKING_UPSTAIRS`, `WALKING_DOWNSTAIRS`), the problem in this section reduces to distinguishing between the type of activity: `STATIC` or `DYNAMIC`.

Similar to previous analysis, a radial-kernel SVM is fit to classify the type of activity. Again, a grid search is performed to find an optimal value of λ and γ. For this grid search, there are $n_1 = 20$ values of λ between 1 and 500 and $n_2 = 20$ values of γ between 0.00001133 and 0.00435. Figure 10 shows two accuracy heatmaps from the grid search.

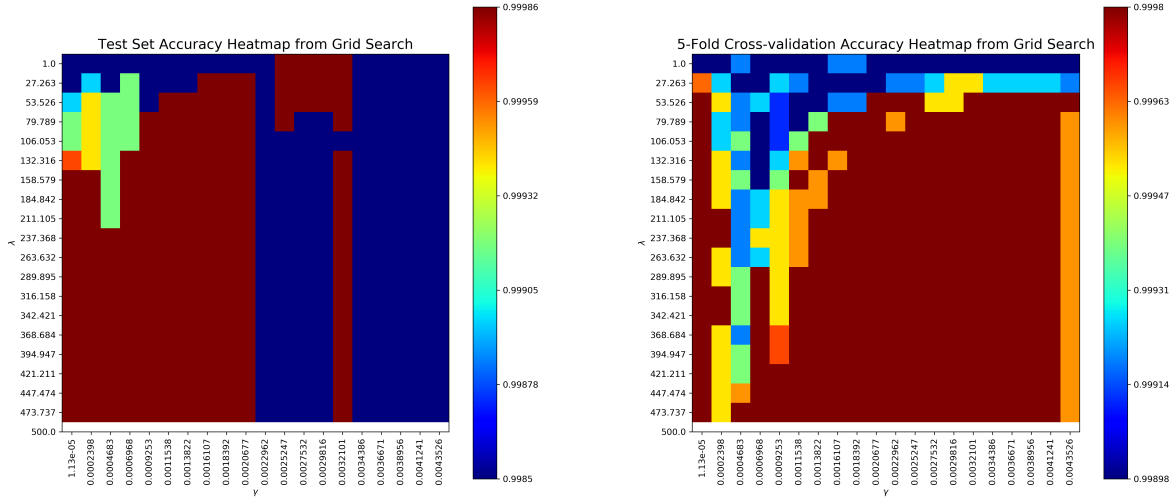


Figure 10: Two accuracy heatmaps from hyperparameter tuning for optimal λ between 1 and 500 and optimal γ between 0.00001133 and 0.00435 for a binary classifier: `STATIC` vs. `DYNAMIC`

From Figure 10, the choice of λ^* is 53.526 and the choice of γ^* is 0.0032101. This choice comes from picking λ^* that is small enough to avoid overfitting and picking γ^* that does well on the testing set.

Next, these hyperparameters are evaluated on the testing set and the accuracy is 100.00%. Table 7 shows the resulting confusion matrix.

	STATIC	DYNAMIC
STATIC	1387	0
DYNAMIC	0	1560

Table 7: Confusion matrix from testing set evaluation of radial-kernel SVM with $\lambda^* = 53.526$ and $\gamma^* = 0.0032101$.

From Table 7, the evaluation of this binary radial-kernel SVM results in perfect classification of the type of activity STATIC versus DYNAMIC.

In conclusion, using a radial-kernel SVM for multi-class classification of the HAR dataset results in 96.64% accuracy on testing data when trying to distinguish between the labels SITTING, STANDING, LAYING, WALKING, WALKING DOWNSTAIRS, and WALKING UPSTAIRS. Using a radial-kernel SVM for binary classification between STATIC and DYNAMIC activities results in 100.00% accuracy on testing data. This suggests that a SVM classifier perfectly separating the HAR data into the six activity labels is difficult. However, reducing this problem into a binary classification, a SVM classifier demonstrates perfect separation of the data into the two types of activities. Although this binary classification performed well on the given testing data, it does not necessarily guarantee separation of any other HAR data. Instead of reducing the problem to binary classification, exploration of dimensionality reduction is contained in what follows.

DIMENSION REDUCTION

11/12. PCA FOR WALKING UPSTAIRS

Before testing other classifiers, the discussion of Principal Component Analysis (PCA) begins with Figure 11. This graph shows the amount of residual variance as a function of the first k Principal Components (PCs) for the data labeled WALKING UPSTAIRS. Recall that in PCA [8], the first Principal Component (PC) corresponds to the largest eigenpair of the scatter matrix and the direction of the largest amount of explained variance in the data.

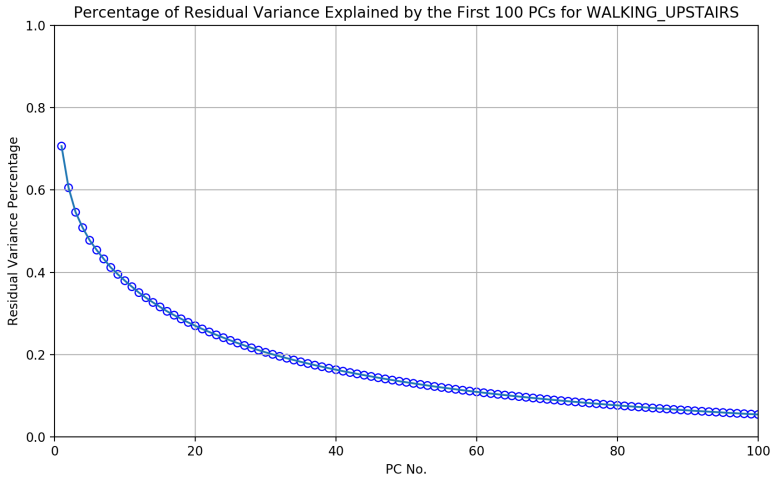


Figure 11: Percentage of residual variance plotted as a function of the first k PCs for WALKING UPSTAIRS

From Figure 11, it seems reasonable that roughly 90% of the variation in the data can be expressed by $k = 60$ PCs. As an aside, an important step in using PCA for classification is to center and sphere the data [9]. Not only is this good practice in general, it prevents unnecessary covariance and skew between features in the data when projecting on to the reduced set of PCs. For example, for the data corresponding to WALKING UPSTAIRS, the amount of variation explained by the first PC is 77.78% when the data is standardized and 70.66% when the data is not standardized.

After computing the PCA for data labeled WALKING UPSTAIRS, the projection of this data onto the first three PCs is shown in Figure 12.

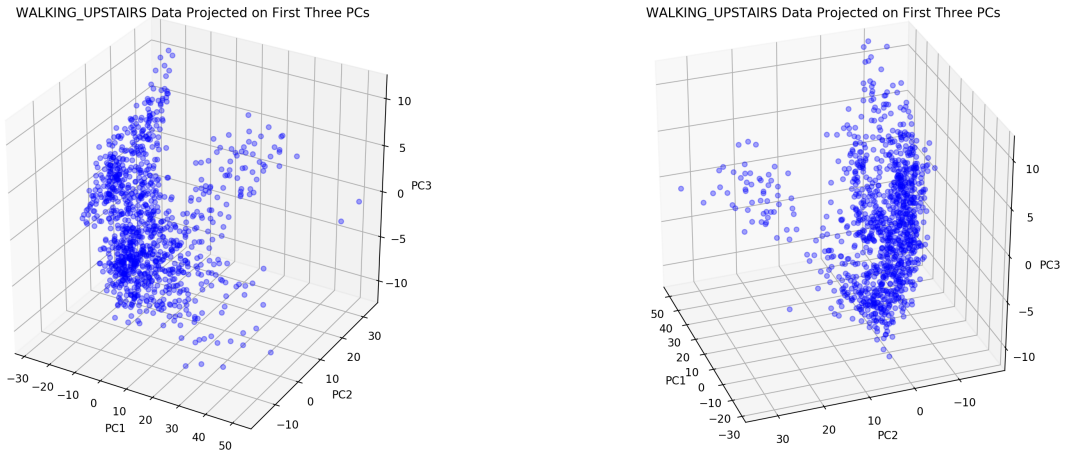


Figure 12: Two perspectives of projecting data labeled `WALKING UPSTAIRS` onto the first three PCs

13. PCA FOR `LAYING`

A similar analysis to section **11/12.** is done for the data corresponding to `LAYING`. Figure 13 plots the residual variance as a function of the first k PCs.

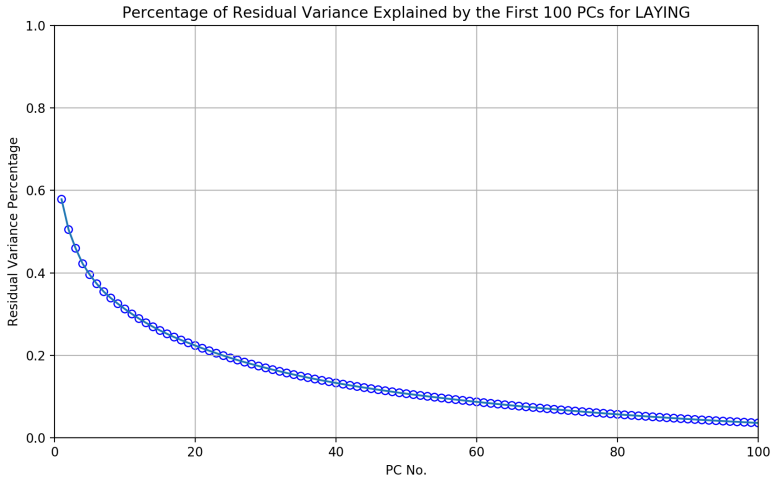


Figure 13: Percentage of residual variance plotted as a function of the first k PCs for `LAYING`

Again, the first $k = 60$ PCs capture about 90% of the variance in the data labeled `LAYING`. One difference between Figure 13 and Figure 11 is that the first couple of PCs for data marked `LAYING` explain less variation than in the first couple of PCs for data marked `WALKING UPSTAIRS`. An explanation for this is that when a person is walking, they usually have a direction and therefore sensor data picks up that direction. However, if a person is laying, it can be harder to interpret the differences in sensor data based on a person's orientation because there is less movement. Figure 14 shows that the first three PCs are clustered closer than in Figure 12. There appears to be several data points far away from the cluster, which might potentially lead to misclassification later on.

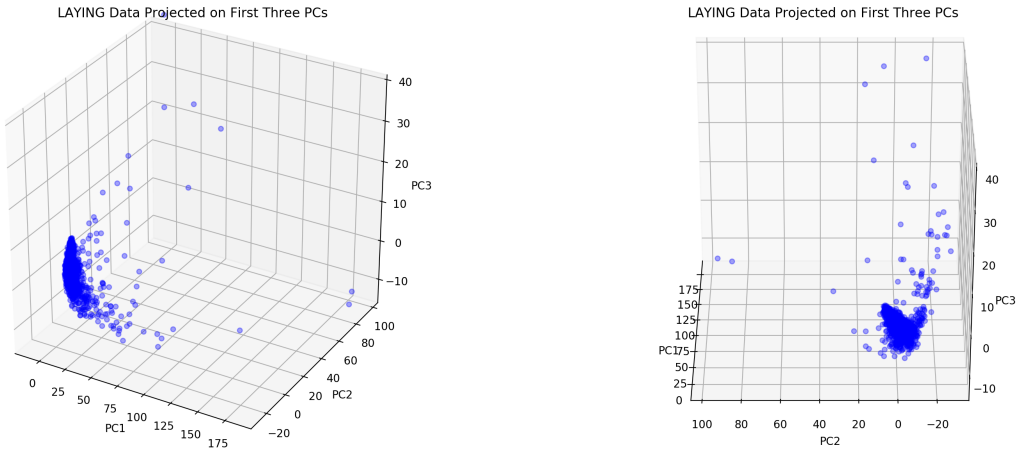


Figure 14: Two perspectives of projecting `LAYING` data onto the first three PCs

14/15/16. PCA FOR HAR DATA

After analyzing the first three PCs for `WALKING UPSTAIRS` and `LAYING`, it appears that the shape of PCs should differ across activities. To explore this further, PCA is performed on the entire training dataset. Figure 15 shows the percentage of residual variance for this data as a function of the first k PCs.

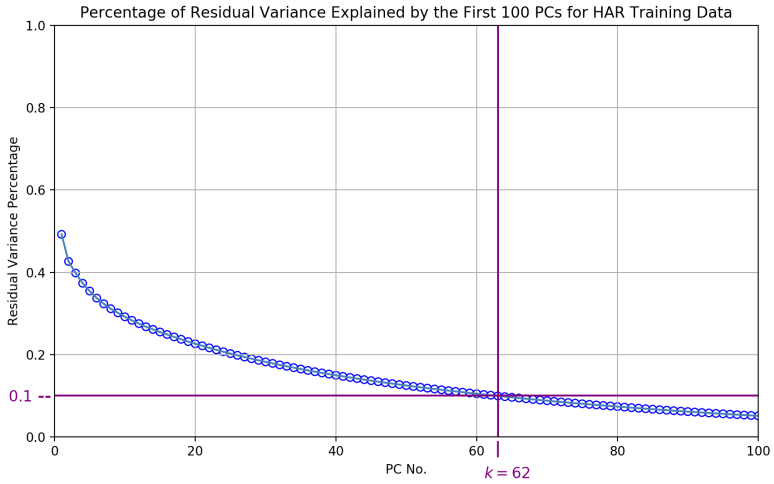


Figure 15: Percentage of residual variance plotted as a function of the first k PCs for the HAR training data

A cross is marked in purple on Figure 15 indicating that $k = 62$ PCs from the HAR training data explain at least 90% of the variation in the data. Specifically, $k = 62$ PCs explain 90.05% of this variation.

Additionally, the HAR training data projected onto the first three PCs is shown in Figure 16.

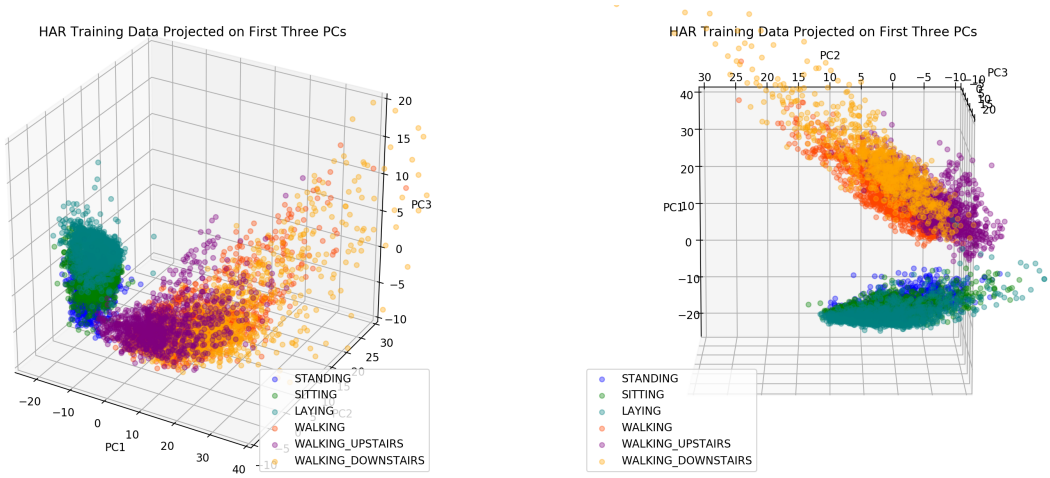


Figure 16: Two perspectives of projecting the HAR training data onto the first three PCs

From Figure 16, it appears that in the first three PCs, the data is distinctly different between static and dynamic activities. Although not perfect, the separation of the six class is much more distinguished than in Figure 17 (created from Figure 8 and Figure 9). Plotting individual features, such as mean linear body acceleration and mean body angular velocity, make it much more difficult to distinguish between classes than plotting the first three PCs.

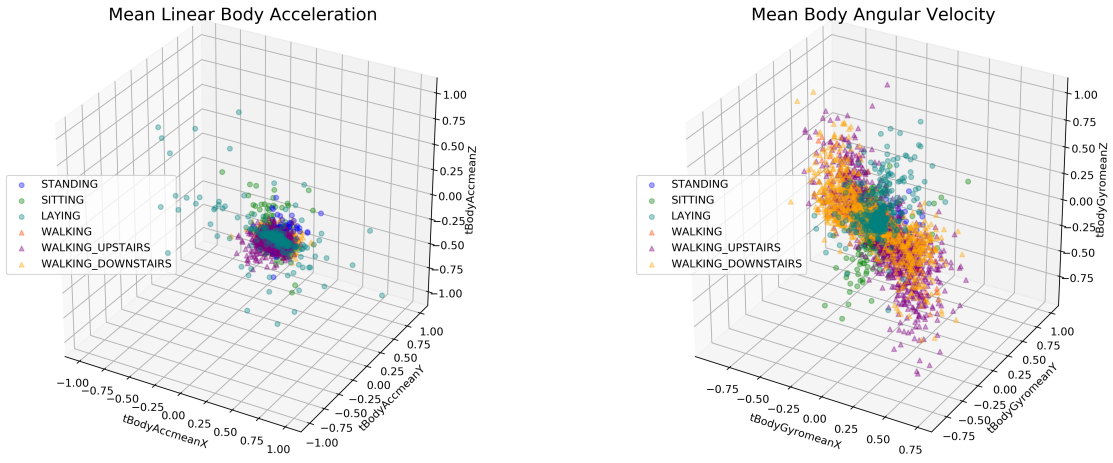


Figure 17: Mean linear body acceleration (left) and mean body angular velocity (right) for each axis (x, y, z) from the HAR training data

A downside of PCA is that the interpretation of the data becomes more muddled because each PC corresponds to a direction and strength, not necessarily a direct measurement from the real world (or wherever data is measured and quantified). However, the advantage is that at the cost of losing some explained variance, PCA allows use of a reduced set of features that are orthogonal. In this case, training classifiers that have ≈ 60 features is much less computationally expensive than using the full feature space of 561. The question now becomes: *how accurate are classification tasks when using a reduced set of PCs as features?*

CLASSIFICATION USING THE REDUCED COORDINATES

The following five classifiers will be analyzed and tested using a reduced feature space produced by PCA:

- Logistic Regression

- Quadratic Discriminant Analysis
- Single-Layer feed forward Neural Network
- Radial-kernel Support Vector Machine
- Three-Layer feed forward Neural Network

For each section that follows, each of these classifiers are first analyzed with a range of values for the number of PCs used. Second, the hyperparameters are tuned to find an optimal classifier for the first $k = 60$ PCs. Then, a confusion matrix for each classifier is discussed. Finally, these optimal classifiers are compared.

17/18. LOGISTIC REGRESSION

To begin, a Logistic Regression (LR) classifier with $\lambda = 1.0$ is fit to a series of eleven values of the number of PCs between $k = 10$ to $k = 60$. For each value of k, the accuracy on testing data is plotted in Figure 18. This LR classifier uses a Stochastic Average Gradient (SAG) solver [12], which should help with the large training dataset size.

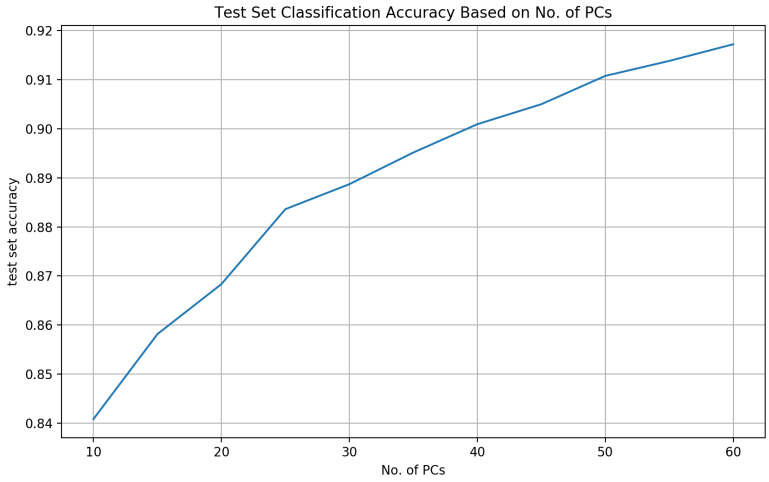


Figure 18: LR testing data accuracy for values of k, the number of PCs

It makes sense that the LR classifier is more accurate as we add more PCs. However, only using $k = 10$ PCs results in an impressive testing data classification accuracy of $\approx 84\%$. Further, for $k = 60$ PCs, the hyperparameter tuning of λ (L2 regularization [13]), is plotted in Figure 19.

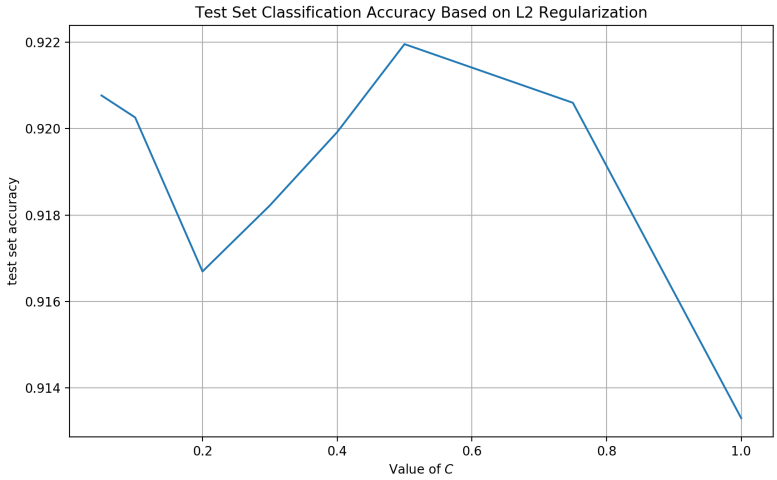


Figure 19: Hyperparameter tuning for LR classifier based on training set accuracy for $k = 60$ PCs

The optimal choice of the regularization is $\lambda = 0.5$ and the testing set accuracy of this classifier is 91.55%. The confusion matrix is shown in Table 8.

	LAYING	SITTING	STANDING	WALKING	WALK DOWNST.	WALK UPST.
LAYING	531	6	0	0	0	0
SITTING	0	425	64	0	0	2
STANDING	0	48	484	0	0	0
WALKING	0	0	0	473	7	16
WALK DOWNST.	0	0	1	7	383	29
WALK UPST.	0	0	0	32	15	424

Table 8: Confusion matrix from evaluating optimal LR classifier on first $k = 60$ PCs

The optimal LR classifier with $k = 60$ PCs does well distinguishing between static and dynamic activities, as there are only 3 errors distinguishing between these types of activities. However, it struggles with characterizing SITTING versus STANDING. There are 48 examples of the true label STANDING classified as SITTING and 64 examples of the true label SITTING classified as STANDING.

19/20. Quadratic Discriminant Analysis

Next, a Quadratic Discriminant Analysis (QDA) is preformed to classify activities. First, QDA testing set accuracy is plotted as a function of number of PCs in Figure 20. Results are similar to

the LR classifier, with accuracy increasing as the number of PCs increases.

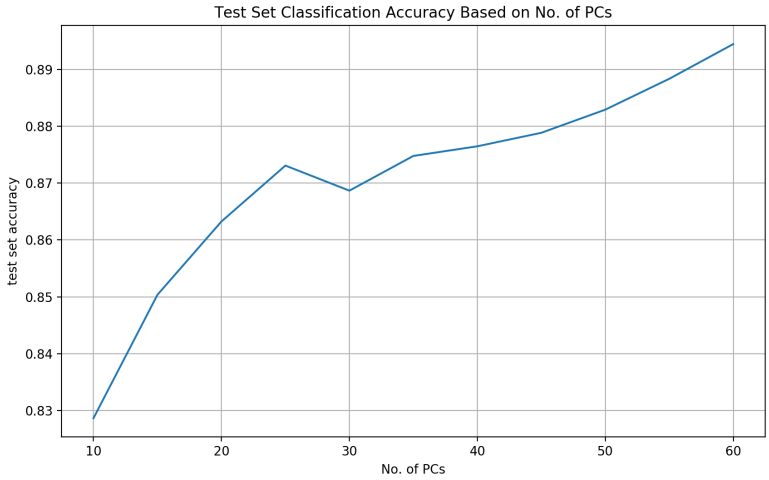


Figure 20: QDA testing data accuracy for values of k, the number of PCs

Further, to attempt a more optimal solution using QDA on $k = 60$ PCs, the value of the regularization parameter [7] is tuned.

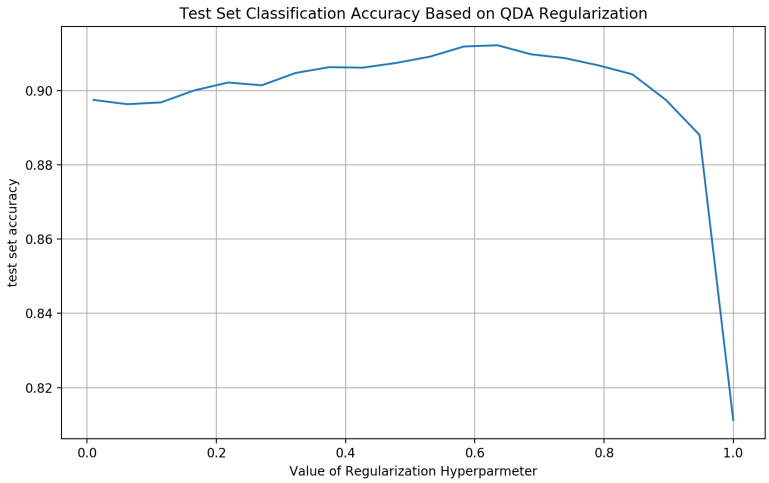


Figure 21: Hyperparameter tuning for QDA classifier based on training set accuracy for $k = 60$ PCs

From Figure 21, the optimal value of the regularization parameter in QDA [16] is 0.63526. Evaluating the optimal QDA classifier results in 91.22% accuracy on the testing set. The confusion matrix associated with this optimal classifier is in Table 9.

	LAYING	SITTING	STANDING	WALKING	WALK DOWNST.	WALK UPST.
LAYING	517	20	0	0	0	0
SITTING	2	385	103	0	0	1
STANDING	0	34	498	0	0	0
WALKING	0	0	0	473	19	4
WALK DOWNST.	0	0	0	11	380	29
WALK UPST.	0	0	0	22	8	441

Table 9: Confusion matrix from evaluating optimal QDA classifier on first $k = 60$ PCs

The optimal QDA classifier for $k = 60$ PCs only makes one mistake distinguishing between static and dynamic activities. However, it struggles even more than the LR in distinguishing between SITTING and STANDING, with 103 instances of the true label SITTING being classified as STANDING. Further, there are 46 instances of the true label WALKING DOWNSTAIRS being classified as WALING UPSTAIRS, which is the same number of misclassifications in this category as the LR classifier.

21/22. SINGLE-LAYER FEED FORWARD NEURAL NETWORKS

Similarly, a single-layer feed forward Neural Network (NN) with ten hidden units in the hidden layer and Rectified Linear Unit (ReLU), $f(x) = max\{0, x\}$, activation is evaluated on eleven values of k, the number of PCs used. As an aside, this Multi-Layer Perceptron [15] classifier uses the ADAM [11] algorithm for weight/bias estimation. Testing set accuracy is plotted in Figure 22, which shows a similar trend to LR and QDA. It also shows the most accurate classifier in the range being the one with $k = 60$.

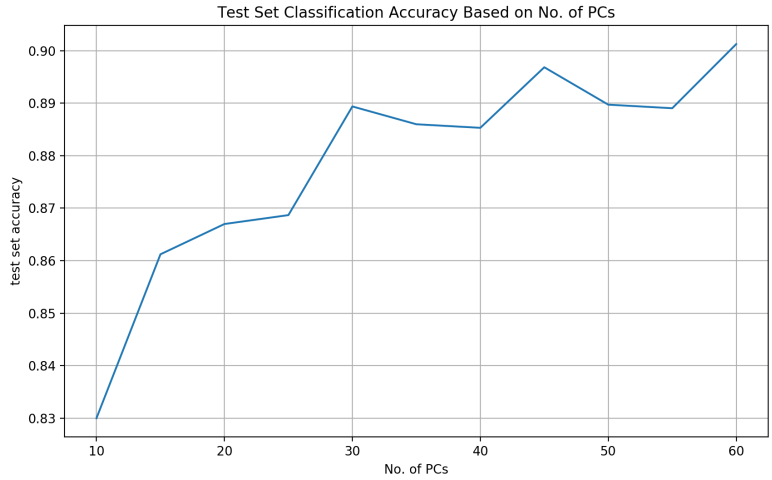


Figure 22: Single-layer NN testing data accuracy for values of k, the number of PCs

Next, the single-layer NN classifier is tuned based on the value of α, the L2 regularization term [15]. Accuracy on the testing data is seen in Figure 23.

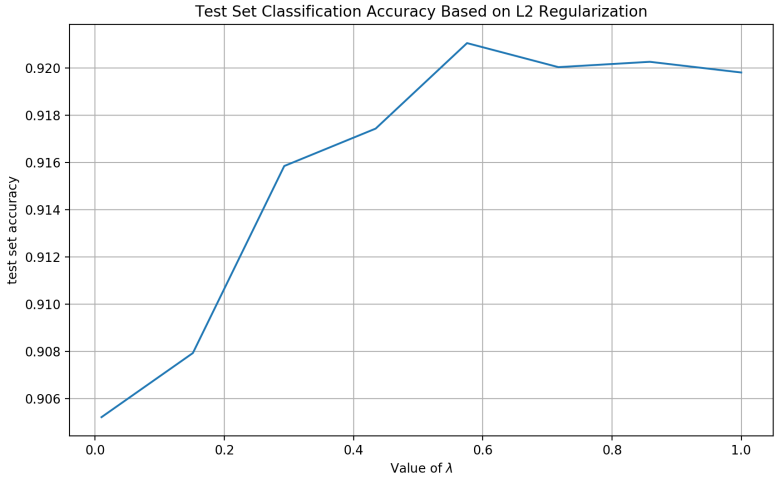


Figure 23: Hyperparameter tuning for single-layer NN classifier based on training set accuracy for $k = 60$ PCs

With the optimal value of $\alpha = 0.57571$, the optimal single-layer NN has a testing accuracy of 92.19%. Table 10 is a confusion matrix for this classifier.

	LAYING	SITTING	STANDING	WALKING	WALK DOWNST.	WALK UPST.
LAYING	532	5	0	0	0	0
SITTING	1	417	71	0	0	2
STANDING	0	42	490	0	0	0
WALKING	0	0	0	472	5	19
WALK DOWNST.	0	0	0	8	366	46
WALK UPST.	0	0	0	23	8	440

Table 10: Confusion matrix from evaluating optimal single-layer NN on first $k = 60$ PCs

Similar to the first two, the single-layer NN struggles with distinguishing between SITTING and STANDING. However, it does worse than QDA and LR with 46 cases of the true label WALKING DOWNSTAIRS classified as WALKING UPSTAIRS. So far, the confusion matrices seen in Table 8, Table 9, and Table 10 show that for a reduced feature space based on $k = 60$ PCs, classification between static and dynamic activities is done well but classification within each type of activity is generally worse than using the full feature space.

SUPPLEMENTAL ANALYSIS: REDUCED COORDINATES KERNEL SVM

To contrast previous analysis with a radial-kernel Support Vector Machine (SVM), a parallel analysis is performed on the reduced feature space. First, testing set accuracy based on the number of PCs used is plotted in Figure 24. To note, this classifier is using $\lambda = 1.0$ and $\gamma = \frac{1}{N}$, where N is the number of instances in the training set ($N = 7352$).

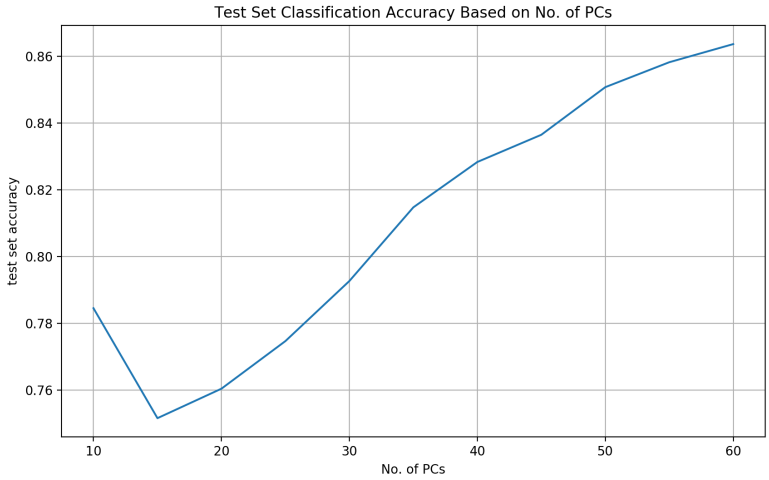


Figure 24: Radial-kernel SVM testing data accuracy for values of k, the number of PCs

In general, the trend in Figure 24 is the same as with LR, QDA and single-layer NN. However, using the radial-kernel SVM seems to overall be less accurate when using this subset of the number of PCs in the range of 10 to 60. One example of this is that the testing set accuracy of the radial-kernel SVM using $k = 15$ PCs is only 75.16% accurate whereas using the LR classifier with the same number of PCs is 85.82% accurate.

To find the optimal radial-kernel SVM for $k = 60$ principal components, a grid search is performed based on the testing set accuracy for $n_1 = 10$ values of λ between 0.25 and 2.0 and $n_2 = 10$ values of γ between 0.00136 and 0.00435. Figure 25 shows a heatmap of the result.

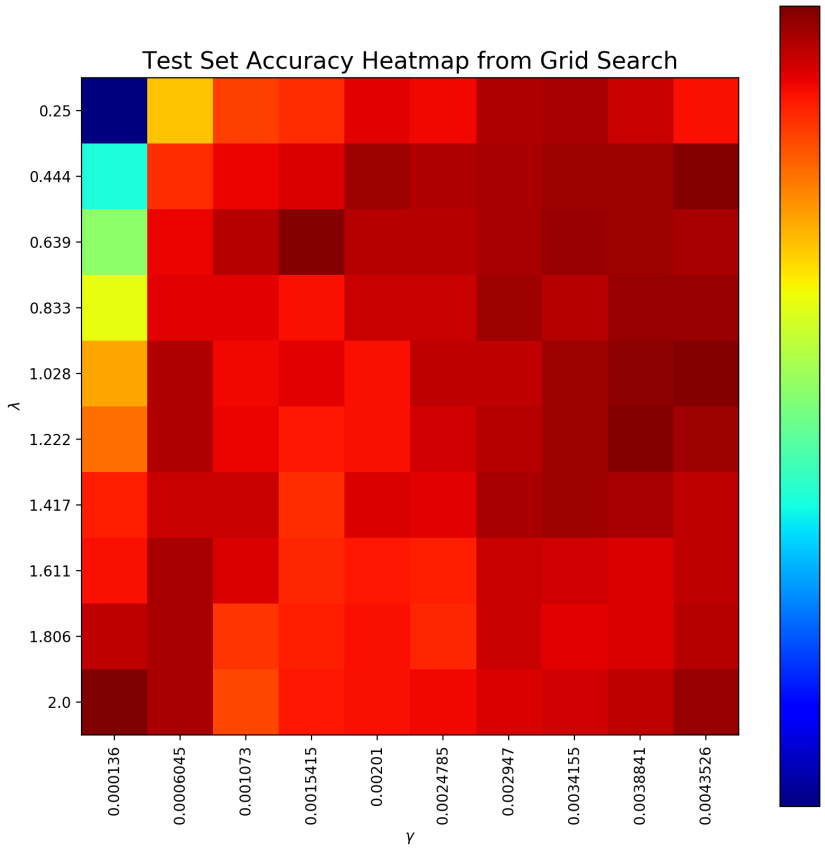


Figure 25: Grid search testing set accuracy for $n_1 = 20$ values of λ such that $1 \leq \lambda \leq 4000$ and $n_2 = 20$ values of γ such that $0.00136 \leq \gamma \leq 0.00435$

The optimal values from Figure 25 are $\lambda = 0.639$ and $\gamma = 0.001542$. Using these values with a radial-kernel SVM and $k = 60$ PCs results in a 91.35% accuracy on the testing data. The resulting confusion matrix is in Table 11.

	LAYING	SITTING	STANDING	WALKING	WALK DOWNST.	WALK UPST.
LAYING	519	18	0	0	0	0
SITTING	2	411	77	0	0	1
STANDING	0	39	493	0	0	0
WALKING	0	0	0	471	7	18
WALK DOWNST.	0	0	0	10	360	50
WALK UPST.	0	0	0	24	9	438

Table 11: Confusion matrix from evaluating optimal radial-kernel SVM on first $k = 60$ PCs

This optimal radial-kernel SVM is similar to the single-layer NN. It beats the QDA classifier with less misclassifications of the true label SITTING being classified as STANDING (77 versus 103).

SUPPLEMENTAL ANALYSIS: REDUCED COORDINATES NEURAL NETWORKS

The final classifier is a three-layer Neural Network (NN) using TensorFlow [2]. The reason for using this as a comparison is two-fold. First, including this classifier is a look into a more "deep learning" approach to the problem, focusing on tuning hyperparameters and network architecture [6]. It is important to note that this is only one approach and there are many ways to solve this classification problem with a "deep learning" approach. Second, using the three-layer NN will allow a comparison of accuracy between this and the classifiers above, in hopes to spark discussion of the use of PCA in complex classification tasks.

In a similar way as above, a three-layer NN is trained using different values of the number of PCs used. In this first network architecture, seen in Figure 26, the number of units in the first, second, and third hidden layers are 300, 150, and 60, respectively. In each layer, the activation function is ReLU and the kernel [10] has a uniform initialization. The sigmoid activation, $f(x) = \frac{1}{(1+e^{-x})}$ is used as the output activation. Additionally, the ADAM [11] optimizer is used with 50 epochs and a batch size of 256. Just to note, batch size is the number of times the data is randomly looked at during the training. Batch size is the number of instances of the data that is looked at during each step of updating the weights and biases in the network.

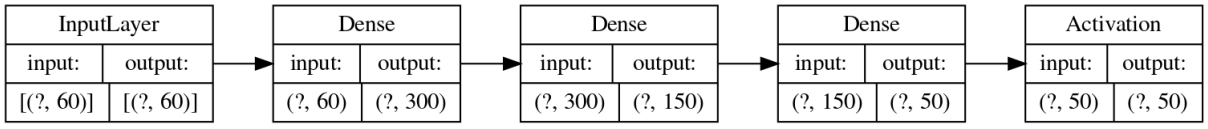


Figure 26: First three-layer NN architecture

Accuracy of this first three-layer NN based on the value of k is plotted in Figure 27.

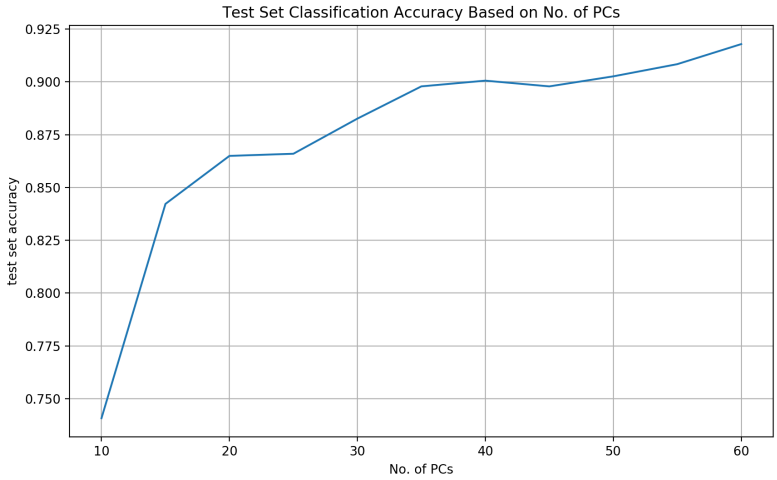


Figure 27: Three-layer NN testing data accuracy for values of k, the number of PCs

This three-layer NN follows the same trend as the other classifiers, improving in accuracy as the number of PCs increases. However, it seems as though for $k = 10$ PCs, it does much worse than the others and for $k = 60$, it performs better than the others. A potential reason for this is the network architecture. Therefore, the next step is performing a grid search on many different three-layer architectures in hopes of finding an optimal structure. This grid search includes 5 different sizes of the first layer between 50 and 250, 5 different sizes of the second layer between 75 and 175, and 5 different sizes of the third layer between 10 and 50. In this grid search, the number of epochs and the batch size mimicked the first set of evaluations above (i.e. 50 and 256, respectively).

The optimal structure found during the grid search is one with hidden layer sizes of 100, 100 and 20 for the first, second, and third layer, respectively. This architecture is seen in Figure 28.

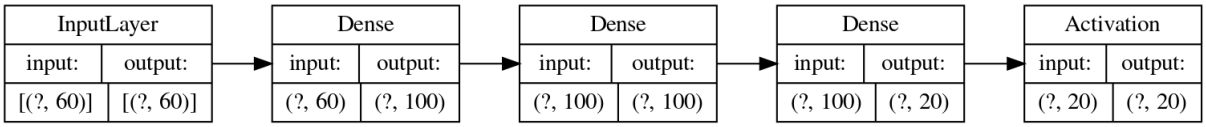
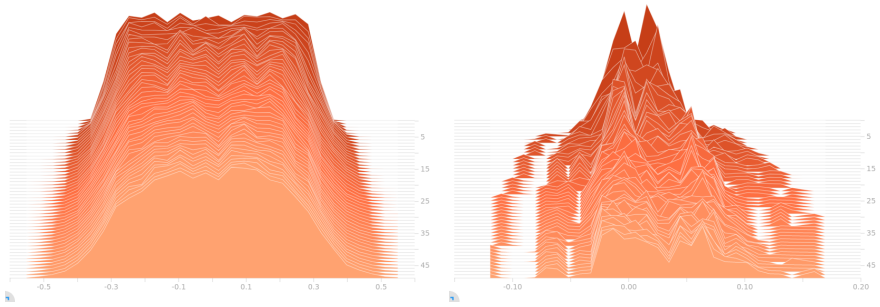


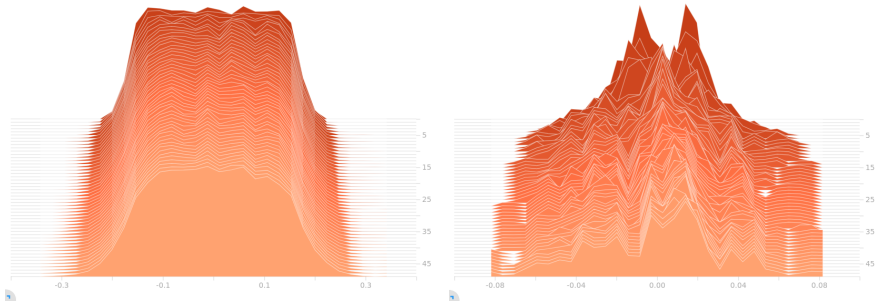
Figure 28: Optimal three-layer NN architecture

The optimal three-layer NN classifier has 92.23% accuracy on the testing data when using $k = 60$ PCs. Further, using TensorBoard [5], Figure 29, Figure 30, and Figure 31, show the evolution of weights and biases at each epoch for each layer.



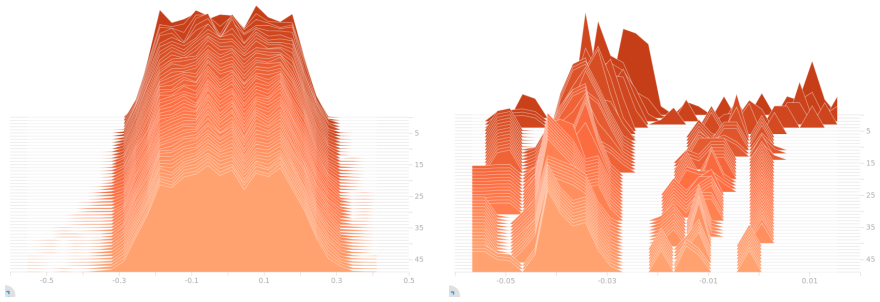
L1 WEIGHT L1 BIAS

Figure 29: Evolution of weight and bias in first hidden layer of optimal three-layer NN using $k = 60$ PCs



L2 WEIGHT L2 BIAS

Figure 30: Evolution of weight and bias in second hidden layer of optimal three-layer NN using $k = 60$ PCs



L3 WEIGHT L3 BIAS

Figure 31: Evolution of weight and bias in third hidden layer of optimal three-layer NN using $k = 60$ PCs

The change in bias seen on the right side of Figure 31 is a direct result of updating the difference

between predictions at the end of a batch and the truth in the training value. Interestingly enough, there seems to be roughly six spikes in the bias by the end of the evolution. These peaks correspond to the network attempting to distinguish between the six class labels. Further, the large gap in the middle must represents divide between static and dynamic activities.

Finally, Table 12 shows the confusion matrix for the optimal three-layer NN classifier.

	LAYING	SITTING	STANDING	WALKING	WALK DOWNST.	WALK UPST.
LAYING	528	9	0	0	0	0
SITTING	2	416	74	0	0	0
STANDING	1	41	489	1	0	0
WALKING	0	0	0	475	6	15
WALK DOWNST.	0	0	0	7	367	46
WALK UPST.	1	0	0	38	14	418

Table 12: Confusion matrix from evaluating optimal 3-hidden layer NN on first $k = 60$ PCs

There are two main differences between the result of the three-layer NN and all other classifiers using $k = 60$ PCs. First, all other classifiers misclassify at least one instance of of the true label SITTING as WALKING UPSTAIRS, whereas the three-layer NN does not. However, the three-layer NN has one misclassification of the true label STANDING as WALKING and one misclassification of the true label WALKING UPSTAIRS as LAYING. Although these are minor details, the difference means that instead of particular instances in the testing set contributing to the misclassifications, the methods in which these classifiers train can lead to different results. Therefore it is important to look at a testing set accuracy [4] for many different methods.

SUPPLEMENTAL ANALYSIS: COMPARING APPROACHES

To begin the comparison between methods, Figure 32 shows the initial testing set accuracy for each method when evaluating for values of k, the number of PCs used.

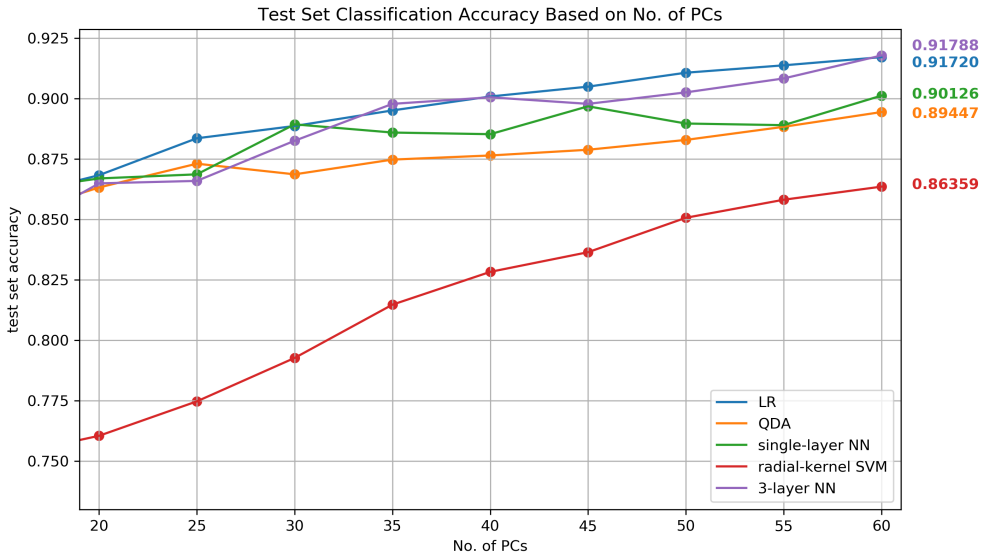


Figure 32: Comparing testing data accuracy between classifiers based on values of k, the number of PCs

From 32, for $k = 60$, the initial accuracy for the three-layer NN is the highest and the radial-kernel SVM is the lowest. Importantly, the LR classifier initially performs similarly to the three-layer NN. Further, it makes sense that the general trend is that accuracy improves when using more PCs.

To recap the process, after each initial look at accuracy, the values of the hyperparameters in each model are tuned to find an optimal classifier. Resulting accuracy for each optimal model is shown in Table 13.

MODEL	OPTIMAL ACCURACY
LOGISTIC REGRESSION	92.1550%
QUADRATIC DISCRIMINANT ANALYSIS	91.2250%
SINGLE-LAYER NEURAL NETWORK	92.1955%
RADIAL-KERNEL SUPPORT VECTOR MACHINE	91.3471%
THREE-LAYER NEURAL NETWORK	92.2294%

Table 13: Comparing testing set accuracy between optimal classification models using $k = 60$ PCs

All of the values in Table 13 are roughly within 1% of each other. This means that in conclusion, none of these classifiers stand out to be the "best". The methods in which these algorithms train

their classifiers deal with some randomness in the order of seeing each instance of the training data. Therefore, based on the randomness, there can be different testing data evaluation accuracy. For example, in the three-layer NN grid search there was not a single accuracy below 89%, which means that there are many valid network architectures that are sufficiently accurate. Similar results are seen in the grid searches of the other classifiers as well. More importantly, there seems to be a threshold around $\approx \ 92\%$ of testing accuracy because of the use of PCA. In principal, PCA reduces the dimension and therefore reduces the information given to the classifier. Having less information from the data means that naturally, the classifier is at a disadvantage to a classifier with the complete data. For example, the radial-kernel SVM with the full data of 561 features has an accuracy of 96.64% whereas the radial-kernel SVM with $k = 60$ PCs has an accuracy of 91.35%. However, it is still impressive that with only $k = 60$ PCs, which explain $\approx 90\%$ of the variation in the data, all the classifiers discussed achieved such accurate results.

REFERENCES

[1] 3.2. Tuning the hyper-parameters of an estimator. Retrieved from https://scikit-learn.org/stable/modules/grid_search.html

[2] Abadi, M., Agarwal, A., Barham, P., Et al. *TensorFlow: Large-scale machine learning on heterogeneous systems*. 2015. Software available from tensorflow.org

[3] Anguita, D., Ghio, A., Oneto, L., Parra, X., & Reyes-Ortiz, J.L. A public domain dataset for human activity recognition using smartphones. In *European Symposium on Artificial Neural Networks, Computational Intelligence And Machine Learning (ESA NN)* (2013), pp. 437-442.

[4] Daume, H. (2017). *A Course in Machine Learning* (2nd ed.). Retrieved from https://github.com/hal3/ciml/

[5] Get started with Tensorboard. Retrieved from https://www.tensorflow.org/tensorboard/get_started

[6] Gron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* (1st ed.). O'Reilly Media, Inc.

[7] Guo, Y. Hastie, T., & Tibshirani, R. *Regularized Discriminant Analysis and Its Applications in Microarrays*. Biostatics (2005). Retrieved from https://web.stanford.edu/~hastie/Papers/RDA-6.pdf

[8] Hastie, T., Friedman, J., & Tisbshirani, R. (2017). *The Elements of statistical learning: data mining, inference, and prediction* (2nd ed.). New York: Springer.

[9] Importance of Feature Scaling. Retrieved from https://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html

[10] Keras: Usage of initializers. Retrieved from https://keras.io/initializers/

[11] Kingma, D., & Ba, J. *Adam: A Method for Stochastic Optimization*. (2014). https://arXiv.org/abs/1412.6980

[12] Schmidt, M., Le Roux, N. & Bach, F. *Minimizing finite sums with the stochastic average gradient*. Math. Program. 162, 83–112 (2017). https://doi.org/10.1007/s10107-016-1030-6

[13] sklearn.linear_model.LogisticRegression. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[14] sklearn.metrics.confusion_matrix. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

[15] sklearn.neural_network.MLPClassifier. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

[16] sklearn.qda.QDA. Retrieved from https://scikit-learn.org/0.16/modules/generated/sklearn.qda.QDA.html

CODE APPENDIX

For code, inquire via email: `scott.alexander.baker@gmail.com`