X033524: Statistical Learning and Inference
Final Project

Simona Emilova Doneva (J11803099007)

January 19, 2019

Contents

1 Introduction

The project aims at applying model selection techniques to a given Machine Learning (ML) task. The objective is classification and the considered algorithms are K-Nearest Neighbors (KNN), Logistic Regression (LR), Linear Discriminant Analysis (LDA), and Support Vector Machine (SVM). The library used for the work is *sklearn* [1].

2 Data Description

The data set has been generated using Deep Neural Networks for feature extractions from images. It can be summarized as follows:

- target classes: 12
- train data size: 7800 (650 images for each class)
- number of features: 4097

3 Models Description

Four models were considered for the task. Firstly, the KNN for classification is a classic algorithm based on majority voting. Secondly, two techniques which are based on a linear log-odds, i.e. Linear LR and LDA. Finally, the maximum-margin based SVM. In their original formulation, all of them yield a linear classifier, but are following different objectives in the construction of the classifier and therefore generally should lead to different results.
For a new observation, KNN algorithm first chooses the k closest training examples in the feature space. The 'closeness' is based on a similarity metric, i.e. distance, such as the Euclidean distance. Then the sample is assigned to the

[1] Please refer to `https://scikit-learn.org/stable/` for details.

most frequent class among its k nearest neighbors.
LDA classifies to the class that has a the maximum discriminant function value. This function is obtained using the Bayes theorem that models the probability of a class with the post probability (for a class given the observation). One important assumption is that that all classes share the same co-variance matrix and using the linear log-odds function implies a linear decision boundary.
LR has the same form as LDA. However the difference lies in the fact that LR makes less assumptions about the marginal density of the inputs. The model parameters are fit by maximizing the conditional likelihood, while LDA maximizes the full log-likelihood, based on the joint density. In theory this should make LR more robust, as it relies on less assumptions. [1](p.127)
SVM is based on the theory of separating hyperplanes. A hyperplane is optimal if it is the unique hyperplane which correctly classifies the data and has maximal distance (margin) from the training data. Finding the weights to define the hyperplane is modeled as a quadratic programming problem, which can be approached by deriving and solving a dual problem based on the Lagrangian definition.
Note further that since the number of classes is larger than three, Linear Regression was not considered as a solution due to the danger of 'masking', i.e. lining up of classes such that some of the classes are never dominant [1](p.105).

4 Baseline Generation

4.1 Approach and Results

For this and subsequent parts sklearn's Pipeline has been used [2]. It allows to "assemble several steps that can be cross-validated together while setting different parameters".
In the baseline selection part, the out-of-the box models are applied directly to the whole data set with no feature selections or fine-tuning of the hyper-parameters. The only pre-processing of the data is Standardization. The motivation is that models like SVM and LR with regularization expect that all features are centered around 0 and have variance in the same order. Otherwise, the objective function might be dominated by a feature. For this purpose the StandardScaler has been selected. It removes the mean and scales each feature to unit variance. Note that all reported scores are from the private leaderboard.
The obtained results can be seen in the table below.

Technique	Datails	Baseline
KNN	Nr. neighbors = 5 Euclidean distance	0.87179
Logistic Regression	L2 Norm	0.92289
LDA	Solver: sva	0.32371
SVM	Linear Kernel	0.91172

Table 1: Baseline Results

It is evident that the best performing model at this point is Linear Regression. Important detail to note is that for this algorithm, the selected solver is *saga* [2], a recent fast incremental gradient method. It was selected due to its support both for L1 and L2 regularization, as well as expected faster convergence. Also, the default value for the normalization term is 1.0.
In this setting KNN achieves an accuracy below 90%. This can be explained with the so called 'curse of dimensionality' phenomenon. In fact as the number of dimensions grows, it becomes more difficult to gather k observations close to the target point. This is due to the fact that the neighborhoods are spatially large and most of the points are at the boundary [1] (p. 22).
The worst performance is observed for LDA, where the default solver is *sva*. Again, the main explanation for the bad results is that the algorithm suffers in high-dimensional context. Since it is trying to estimate dependence between covariates in the construction of the classifier, the presence of collinear variables can greatly influence the performance [3].

[2]Please refer to `https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html` for details.

5 Dimensionality Reduction

5.1 Background

The number of features for the task at hand is very large, which impacts the performance of the models. For example during training of LDA, the warning for 'Collinear features' was observed. Therefore it was decided to focus on dimensionality reduction for the improvement of the classification performance.
Principal Component Analysis (PCA) is a technique for deriving a new representation of the original data points. PCA creates new features (dimensions) defined as linear combinations (principal components) over all original features. In particular, standard linear principal components (PCA) are obtained from the eigenvectors of the covariance matrix, and give directions in which the data have maximal variance. The technique can be used as dimensionality reduction by choosing a smaller number of components to be used as new inputs for the training of the models.
LDA also allows for projection of the data into a reduced dimensional space. In contrast to PCA however the process considers not only the input data, but also the class labels. We can visualise the differences between the two techniques by keeping only the first two components after the projection and plotting the samples.

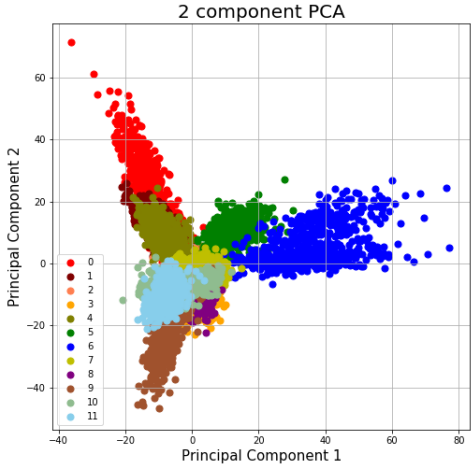

Figure 1: PCA Projection.

We can observe that the projected data has a significant overlap when using the PCA approach.

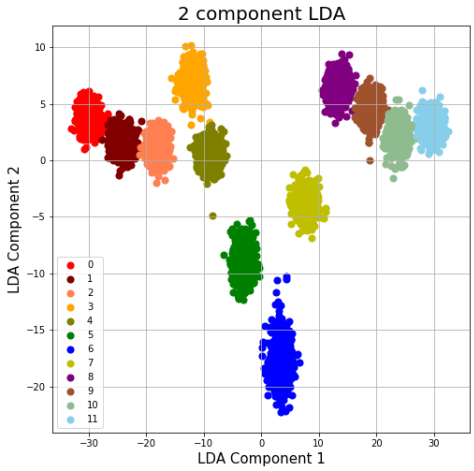

Figure 2: LDA Projection.

For LDA however the discriminant direction minimizes this overlap. The classes are much more clearly separated.

5.2 Approach

The method used to estimate the prediction error is cross-validation (fivefold CV as suggested by [4]). By using the sklearn Pipeline mentioned earlier, it guarantees that the validation approach will mimic correctly evaluating the model against an independent test set. That is, the samples will be divided first, then for each fold the parameters tuning/ variables selection will be performed on all samples outside the fold. Finally, the performance is evaluated for the samples in the fold.

Two steps were followed to decide on an appropriate number of principal components to keep. First, the performance in terms of accuracy is plotted for different number of components. Second, CV was applied for a targeted range of values, for which the initial estimate showed a better score. Based on the CV results, the final number of PC to keep was fixed.
The results from the first step are presented in the following graphs.

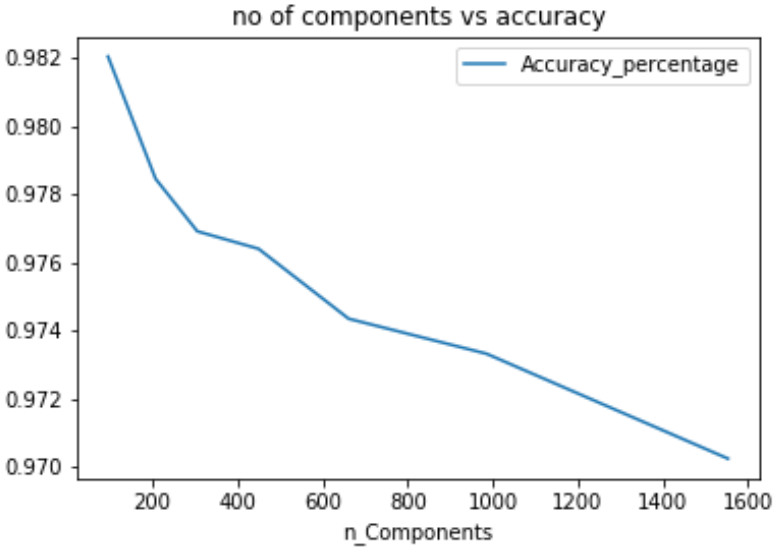

Figure 3: Number of PC estimation for KNN.

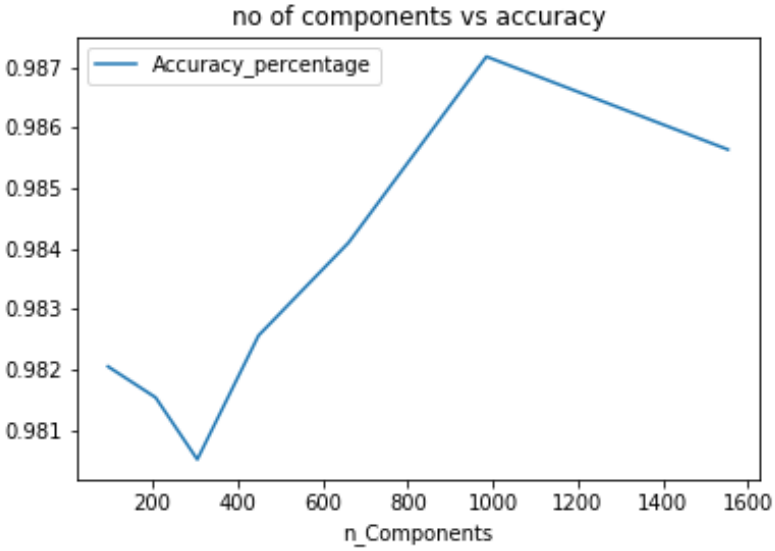

Figure 4: Number of PC estimation for Logistic Regression.

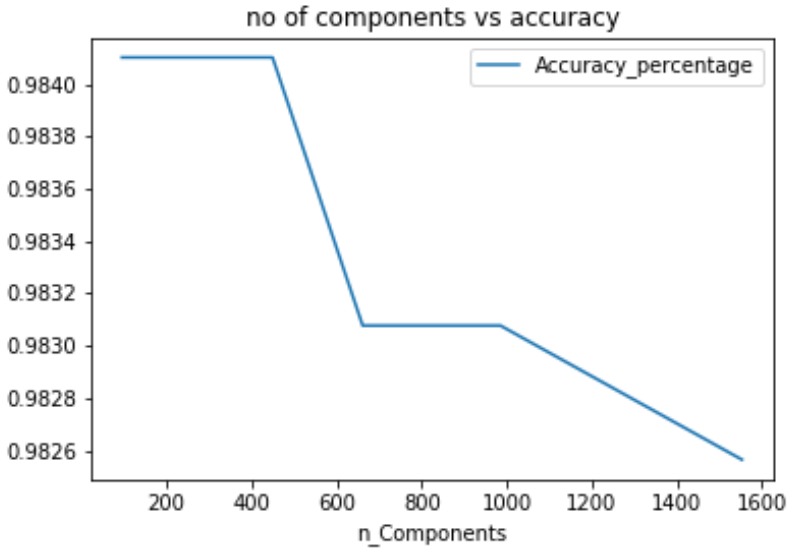

Figure 5: Number of PC estimation for LDA.

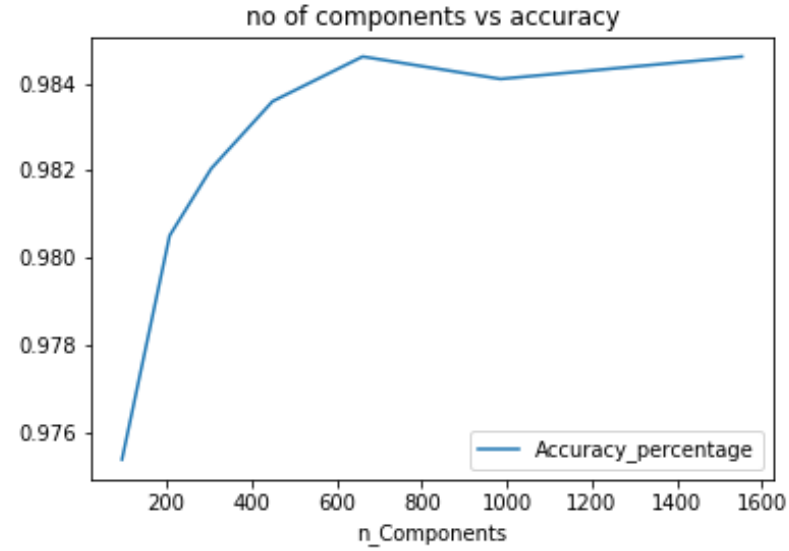

Figure 6: Number of PC estimation for SVM.

We can observe that different ranges of number of features seem to be suitable for the different models. Above all the contrast between KNN, LDA and the other two models is evident. That is, KNN shows to performs best with dimensions below 100, while the considered range for LDA is below 600 components. On the other hand, SVM and LR seem to perform better with higher number of features (around 1000).

Having established appropriate ranges for the models, CV was used to pick a final value and provide a more robust estimate on the expected performance on an independent data set. KNN chose 20 components as optimal with CV accuracy of 0.981. LDA selected 280 components with CV accuracy of 0.982. SVM selected 2500 components with CV accuracy 0.986. LR selected 1100 components with accuracy CV 0.984.

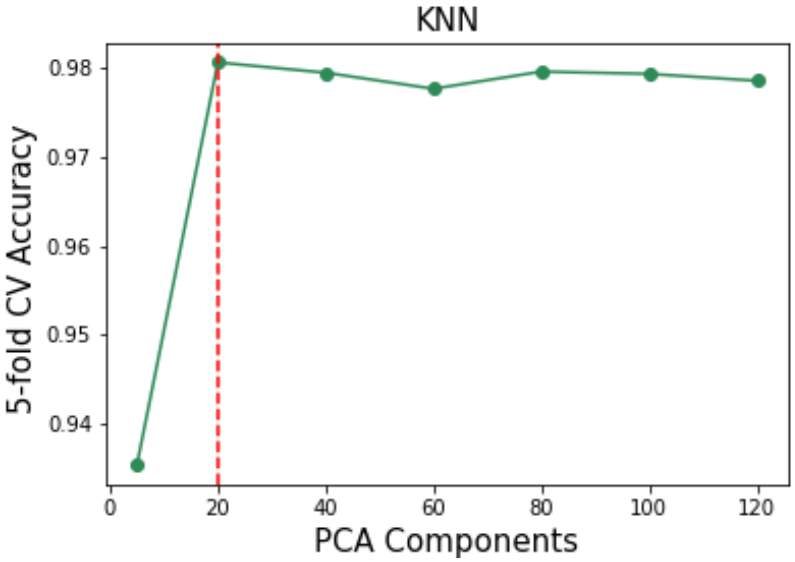

Figure 7: Number of PC estimation for KNN.

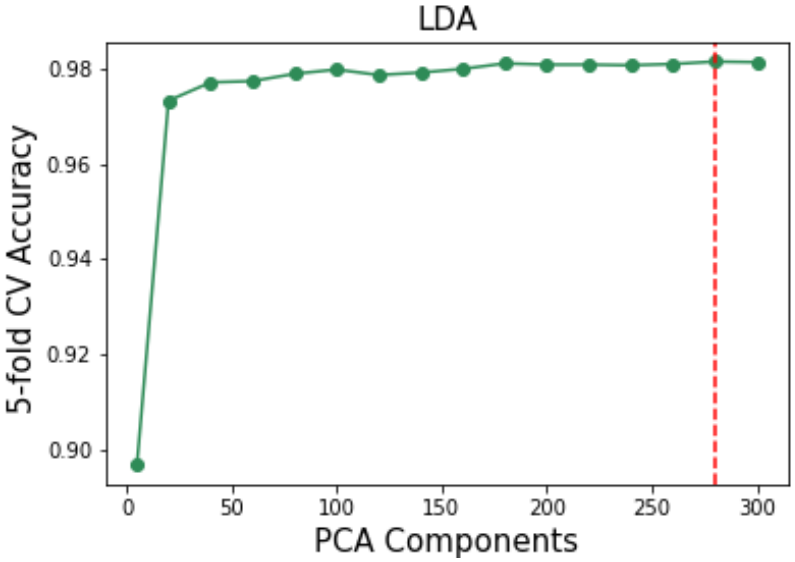

Figure 8: Optimal number of PCA Components for LDA.

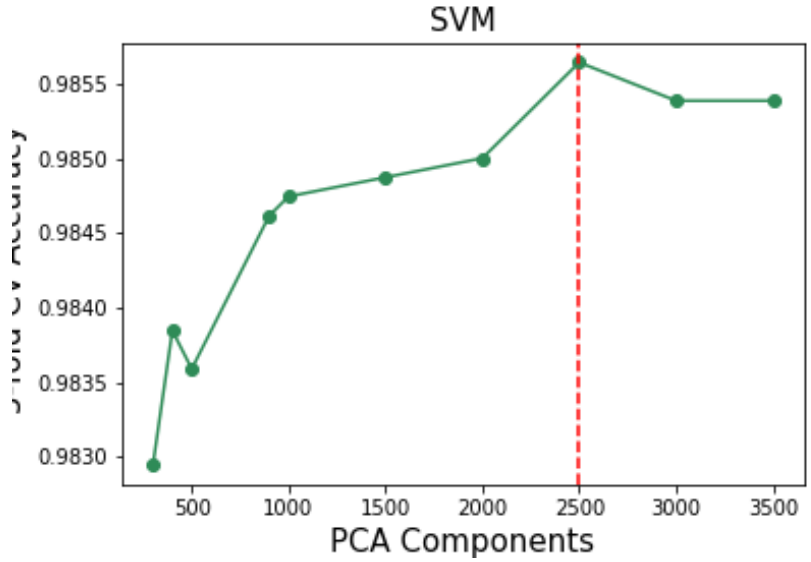

Figure 9: Optimal number of PCA Components for SVM.

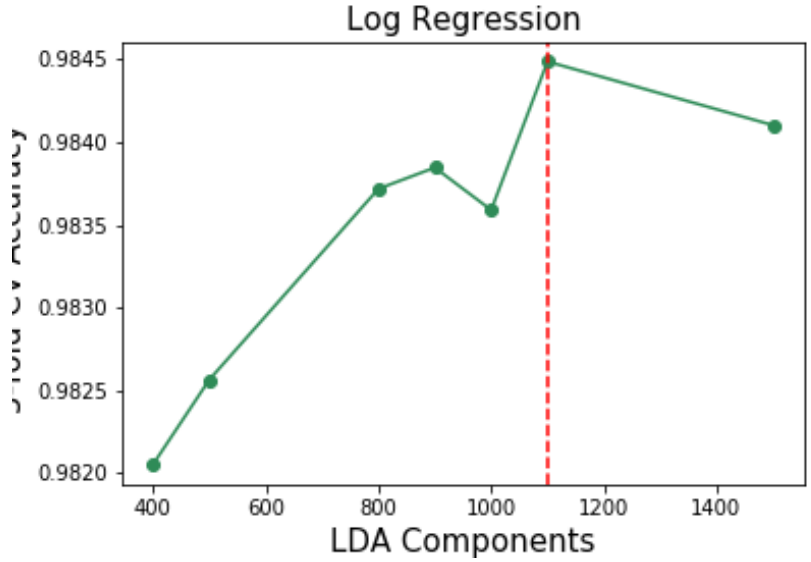

Figure 10: Optimal number of PCA Components for Logistic Regression.

5.3 Performance with reduced dimensions

Finally, the smaller number of components was fixed and the performance of the models with this setting was evaluated against the official test set.

Technique	Nr Components	Datails	CV Results	Test Results	Baseline
KNN	20	Nr. neighbors = 5	0.981	0.91199	0.87179
Logistic Regression	1100	L2 Norm	0.984	0.92216	0.92289
LDA	280	SVD Solver	0.982	0.8984	0.32371
SVM	2500	Linear Kernel	0.986	0.9117	0.9117

Table 2: Models Performance after dimensionality reduction.

As we can observe from the results the impact on LDA is extremely positive, improving the performance by more

than 50%. The other two models do not show significant improvement.
It was also experimented with using the dimensionality reduction capabilities of LDA, but this strategy did not yield good results.

6 Hyperparameters Tuning

6.1 Background

There are several hyperparameters for each model that could play a role in the classification quality.
The most important parameter for the KNN algorithm is k, i.e. the number of neighbours, considered for the majority vote of the classification. Generally, larger values of k reduces effect of the noise on the classification.
For LR there are two possible so called 'shrinkage techniques' that constrain the values of the learned weights and thus should reduce variance. They are the L2 and L1 norm and differ by the type of posed constraint. The first one reduces the weights of features with smaller eigenvalues, while the latter one can set some of the coefficients to zero. The strength of the regularization is determined by the parameter C. In sklearn the larger C, the more freedom is given to the model, while reducing it constraint the model more. The figure below visualises the impact of different regularization norms and strengths for a LR model. To aid visualisation, a total number of 10 dimensions was preserved. One can observe that the sparsity with L1 penalty is much higher than with L2 regularization.

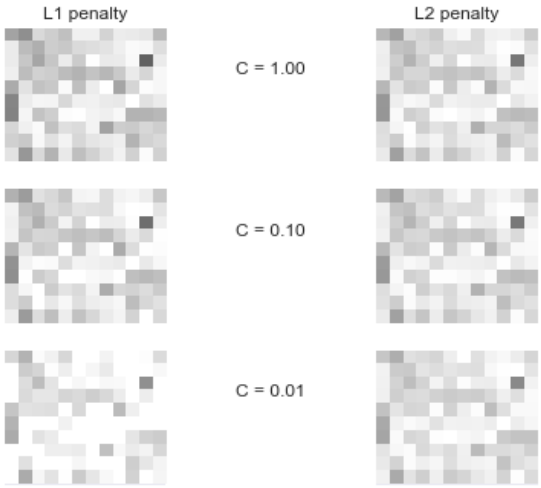

Figure 11: LR model weights.

In the context of SVM the regularization parameter tells the optimization how much misclassifying each training example should be penalized. Further important concept for SVM is the one of *kernel*. By choosing a non-linear kernel, one allows the model to work in a higher-dimension where a linear function can be learned for potentially non-linear separable classes in the original space.

6.2 Approach

Since the dimensionality reduction showed positive impact on the performance of the models and furthermore has posititve impact on training speed, this technique was kept as pre-processing step in the pipeline. For each model there were different hyper-parameters that could be fine-tuned. The CV function of sklearn was utilized by defining a grid of possible values for the different hyper-parameters. The technique then performs the CV accuracy estimation on each possible parameter values combination.

6.3 Final Performance

The chosen hyperparameter values by CV are given in 'Details' of Table 3. One can observe that LR and LDA agree on a regularizatoin value of 0.01 with the L2 Norm. Furthermore the preferred solver for LDA is *eigen*. In contrast to the one used until now (*svd*), this solver "is based on the optimization of the between class scatter to within class scatter ratio".

Technique	Datails	Test Results	Baseline
KNN	PCA: 20 Nr. neighbors = 25	0.91501	0.87179
Logistic Regression	PCA: 1100 L2 Norm, C = 0.01	0.92408	0.92289
LDA	PCA: 280 Solver: eigen Shrinkage: 0.01	0.9027	0.32371
SVM	PCA: 2500 RBF Kernel gamma = 0.001, C = 10	0.91712	0.91172

Table 3: Hyperparameters values and final performance

7 Further Experiments

Combining PCA and LDA dimensionality reduction seemed like an interesting experiment to perform. Therefore first the previously established 280 components of PCA were kept and then LDA was performed with eleven components. This resulted in a performance of 0.91068, which is indeed an improvement of the already established best score.
A closely related model to LDA is the Quadratic Discriminant Analysis (QDA). This model does not make any linearity assumptions. With fine-tuning of the regularization parameter to 0.5 and keeping the dimensions to 280 as for LDA, the obtained test score is 0.91318. This is a better accuracy both compared to plain LDA and to the PCA+LDA approach described previously.
Sklearn provides an alternative implementation of the SVM classifier with linear kernel, i.e. LinearSVM. It differs from the original one by two aspects. It minimizes the squared hinge loss while SVC minimizes the regular hinge loss. Furthermore it relies on the One-vs-All (also known as One-vs-Rest) multiclass reduction while SVC uses the One-vs-One multiclass reduction. [3] Using this method (without dim reduction, with L2 and C set to 0.001) resulted in a score of 0.92298. This number is a significant improvement over the final result shown in the previous chapter.

8 Conclusion and Outlook

The model with best generalization performance that could be obtained is Logistic Regression with L2 Norm and regularization strength 0.01.
Hyperparameters tuning showed to be the most important step to improve the models performance. SVM and LR showed to be robust to higher dimensions of the features. However the given classification problem seems to be of non-linearly separable classes, as indicated by the optimal choice of RBF kernel, as well as the indication that QDA achieves a better accuracy compared to LDA.
Further approaches for dimensionality reduction can be considered for LDA/QDA and KNN, e.g. Recursive Features Elimination. Furthermore, different data scaling techniques could be experimented with, e.g. Normalization. Finally, combining different models showes to be a good idea and hybrid models such as Bagging and Stacking of models should be considered for future work.

References

[1] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:, 2001.

[3] `https://scikit-learn.org/stable/auto_examples/svm/plot_iris.html`

[2] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654, 2014.

[3] Peter J Bickel, Elizaveta Levina, et al. Some theory for fisher's linear discriminant function,naive bayes', and some alternatives when there are many more variables than observations. *Bernoulli*, 10(6):989–1010, 2004.

[4] Leo Breiman and Philip Spector. Submodel selection and evaluation in regression. the x-random case. *International statistical review/revue internationale de Statistique*, pages 291–319, 1992.