report

50098, 39375, 38682, 50450

2024-12-09

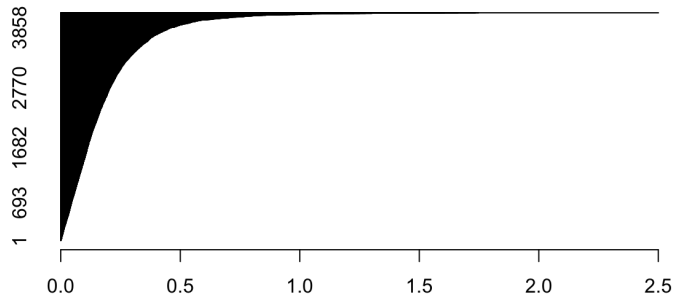
Task 1: Binary Classification

Abstract Task 1

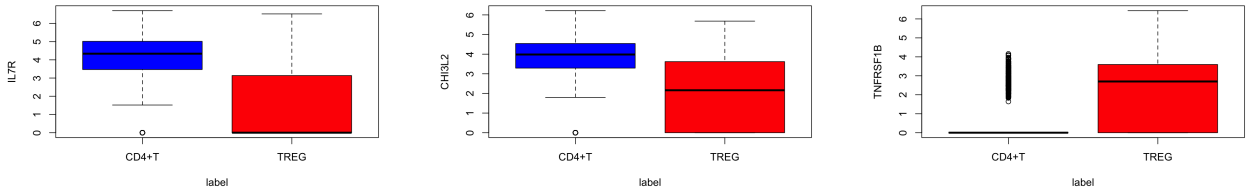
In the following task the binary classification problem of CD4+T cells and TREG cells is going to be elaborated. We aim in this classification task to identify via various classifiers to predict whether a cell, based on a variety of genes (being features/predictors), is part of the cell type/class/label "CD4+T"-cell or "TREG"-cell. Here we are using a training and test split of 80%/20% to train our classifiers on the given data of n=5471 (n observations, number of cells) and p=4123 (number of genes, predictors). After training and testing the classifiers, we are evaluating them based on given performance metrics which are accuracy, balanced accuracy, AUC and the F1 score. Ultimately, we are trying to improve the given classifiers using PCA, and finally focusing on improving the F1 score by either boosting, hyper-parameter tuning or regularisation of the classifiers. Thus, we are yielding in the end a classifier as our mypredict() function which is going to be used for future binary classification problems of "CD4+T" and "TREG" cells.

Task 1.1 Data Description

The task_one_df gives us n=5471 cells (classified either as "CD4+T" or as "TREG") and respectively p=4123 genes (features variables) which is expressing the logarithmically normalized RNA expression level for a specific gene. In total we can find 3356 CD4+T cells and 2115 TREG cells in our total dataset. With regards to our problem, we aim to be able to predict whether a cell is a "CD4+T" cell or a "TREG" cell based on the provided features. Therefore it is of interest to find the genes with the most predictive power and information to help us classify the cells. One initial approach with the EDA would be to identify the genes which have the highest absolute difference between the mean of CD4+T cells' expression level and the mean of TREG cells' expression level. This gives intiuitvely the genes which can seperate both cell types the best due to their difference in RNA values. First we can observe how many cells have which absolute mean difference in relation.



We can see that not all genes have a high amount of information for differentiating between the two cell types CD4+T and TREG and being thus a useful predictor for classification purposes. Rather it is a lower amount of top cells which are able to influence the classification models the most. We will discuss more on this when we talk about PCA and its principal components. In the following we can observe the summary of top 3 cells with the highest absolute difference in mean values, and in addition their boxplots.



The last plot for the third largest difference in absolute mean values differs from the other since the CD4+T 3rd quarter for TNFRSF1B is still 0 and the mean is located in the 4th quarter and therefore the box plot visually cannot be generated here.

Task 1.2 Summary Table without PCA

In the following we are evaluating seven different classifiers regarding their classification performance on the given RNA expression data set. First of all, here is a brief overview of each classifier and its classification algorithm:

- **LDA (Linear Discriminant Analysis)**: The LDA is a dimensionality reduction and uses discriminant functions (linear combinations) within the data and tries to maximize the distance of each class mean and minimize the spread of each class in order to linearly separate each classes from each other in a lower dimensional space.
- **Log Reg (Logistic Regression)**: The logistic regression uses a sigmoid function to classify the cells where the genes represent the coefficients of the function. The function outputs a value between 0 and 1 and a threshold (typcially 0.5) is used to either classify the cell as a CD4+T or TREG cell.
- **QDA (Quadratic Discriminant Analysis**): The QDA assumes that each class follows a Gaussian distribution with its own covariance matrix. It works by finding a decision boundary that is quadratic (curved) rather than linear, unlike Linear Discriminant Analysis (LDA). QDA models the class-specific

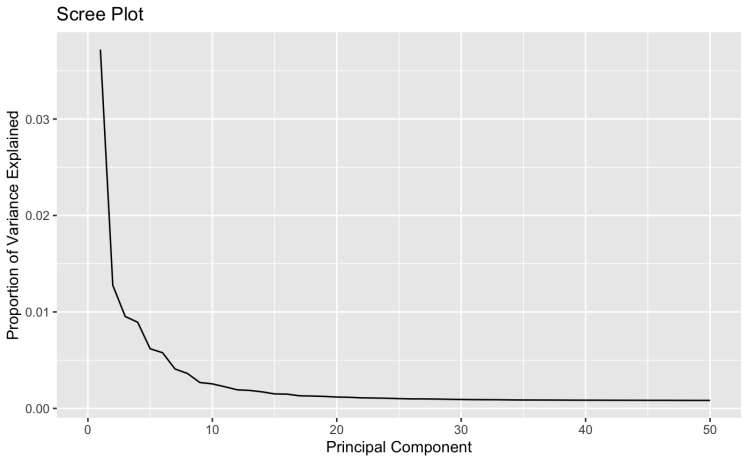
distributions and classifies data by comparing the likelihoods of the data points belonging to each class. The current dimensionality of the data is too high for a QDA classifier. During training, the covariance matrix estimation for each class, k, fails. This occurs because, when the data is partitioned into k classes, the number of observations, n, is much smaller than the number of features, p (i.e n « p). As a result, the covariance matrix estimate for class k becomes singular or nearly singular. Therefore, without dimension reduction or regularisation, training this classifier is not feasible.

- **k-NN (k-nearest Neighbor)**: k-NN is a nonoparametric classficiation alogrithm which classifies a data point based on the majority class of its 'k' closest neighbors in the feature space, using a distance metric like Euclidean distance. K-NN makes predictions by examining the local structure of the data, with no explicit training phase, making it highly sensitive to the choice of 'k' and the scale of the features.
- **GBDT (Gradient Boosted Decision Trees)**: Gradient Boosted Decision Trees (GBDT) are an ensemble learning method that builds a series of decision trees sequentially, where each tree corrects the errors of the previous one. It minimizes a loss function by fitting new trees to the residual errors of the combined model, improving prediction accuracy.
- **RF (Random Forest)**: The RF uses bootstrapped data sets (random sampling with replacement) and uses this data for decision trees where each node is representing a subset of genes to find the best split. Moreover, multiple decision trees of different structures are made and ultimately aggregated where the majority vote classifies the cell. This makes RF already a bagged (bootstrap aggregated) classifier, allowing to reduce overfitting and improving robustness.
- **SVM (Support Vector Machine)**: The SVM works by finding a hyperplane that best separates data into different classes, maximizing the margin between the classes. SVM can also handle non-linear data by using kernel functions to transform the data into higher-dimensional spaces.

w/o PCA	Accuracy	Bal. Accuracy	AUC	F1
LDA	0.9416	0.9407	0.9407	0.9238
Logistic	0.6393	0.6413	0.6597	0.5766
QDA	N/A	N/A	N/A	N/A
k-NN	0.7425	0.6964	0.6964	0.5983
GBDT	0.9087	0.8887	**0.9731**	0.8698
RF	0.9352	0.9171	0.9171	0.9077
SVM	**0.9553**	**0.9479**	0.9479	**0.9394**

Task 1.2 Summary Table with PCA

In the following we are applying the dimensionality reduction tool PCA (Principal Component Analyis). It transforms data into a new coordinate system, where the axes (principal components) capture the maximum variance in the data. It finds a set of orthogonal directions (principal components) that explain the most variability, with the first component explaining the most, the second the second most, and so on. PCA is often used to reduce the number of features while preserving as much information as possible, simplifying data for analysis or visualization. In the following scree plot we can additionally see that approximately ten principal components form the elbow of the scree plot, meaning that ten principal components account for the most Variance within the data set which makes them powerful to use for prediction models.



with PCA	Accuracy	Bal. Accuracy	AUC	F1
LDA	0.9352	0.9176	0.9892	0.9079
Logistic	0.9489	0.9414	**0.9913**	0.9309
QDA	0.9416	0.9298	0.9876	0.9194
k-NN	0.9187	0.8982	0.8982	0.8834
GDBT	0.8703	0.8394	0.9483	0.8060
RF	0.9379	0.9259	0.9259	0.9144
SVM (Linear Kernel)	0.9489	**0.9414**	0.9414	0.9309
SVM (Radial Kernel)	**0.9498**	0.9397	0.9397	**0.9312**

Task 1.3 Improved Classifiers

Now we want to focus on improving a selection of classifiers with regards to the F1 score. For this exercise we choose to improve two linear classifiers and one more complex one. Hence, we are improving the LDA, Logistic Regression and AdaBoost with given improvement techniques.

	F1 Score before	F1 Score after
LDA (with CV and Hyperparameter tuning)	0.9079	0.9487
Logistic (with CV and Regularization)	0.9309	0.9510
AdaBoost	0.9289	0.9362

PCA is performed on the data as a means of dimension reduction and scaling. For each of the classifiers, 5-fold cross validations are run on the training data to estimate the F1-Score under different hyperparameters. The main hyperparameters tuned for all classifiers is the number of principal components used in training and the classification threshold. The threshold, in particular, is a key hyperparameter due to its direct impact on the precision and recall of a model. There are other hyperparameters considered and tuned for specific to some of the chosen classifiers.

For a classifier, the set of hyperparameters selected is a set that produces a high average F1-Score from the five folds with a low variance. The classifiers are then trained on the selected number of principal components produced by the training data with the selected set of hyperparameters. Predictions are then made on the principle components of the test set, which are obtained through the same transformation according to the training decomposition. The highest scoring classifier on the test set according to the F1-Score is then the selected classifier for the `mypredict()` function.

The following table shows the hyperparameters considered for each classifier, the range of values cross-validated for and the optimal set chosen.

Classifier	Hyperparameters	Range Cross-Validated	Chosen Set
LDA	No. Principal Components, Decision Threshold	$\{10, 11, \ldots, 30\}$, $\{0.1, 0.2, \ldots, 0.9\}$	26, 0.2
Logistic Ridge Regression	No. Principal Components, Decision Threshold, Regularisation Term (λ)	$\{10, 11, \ldots, 30\}$, $\{0.1, 0.2, \ldots, 0.9\}$, $\{0, 0.00001, 0.0001, 0.001, 0.01\}$	48, 0.4, 0.01
AdaBoost	No. Principal Components, Decision Threshold, Number of Learners (M), Depth of Trees	$\{15, 16, \ldots, 30\}$, $\{0.1, 0.2, \ldots, 0.9\}$, $\{50, 100, 150, 200\}$, $\{3, 5, 7\}$	26, 0.5, 200, 3

The optimal decision thresholds based on the F1-Score varies between classifiers. For the two linear models, the optimal decision boundary is < 0.5, which could be attributed to the lower proportion of positive labels in the training data (~3000 CD4+T labels and ~2000 TREG labels). Additionally for LDA, the optimal threshold is much lower (0.2), which can stem from a violation of the multivariate normal assumption in the PCA space. The larger F1-Score results from lowering the threshold to below 0.5 indicating that the increase in recall is larger than the decrease in precision. AdaBoost's optimal threshold remains close to 0.5 because the boosting algorithm's focus on "harder-to-classify" samples make it less sensitive to a class imbalance.

It is worth noting that both the Ridge and Lasso regularisation techniques for the Logistic Regression were considered as part of cross-validation. The Lasso showed to be too aggressive even for different values of lambda on the principal components. After 35 components, additional coefficients were sent to 0. The Ridge regularisation provided a less aggressive approach and incorporated an additional increase in the F1-Score after 45 components are trained upon suggesting relationships introduced by such components. The optimal decision thresholds based on the F1-Score varies between classifiers. For the two linear models, the optimal decision boundary is < 0.5, which could be attributed to the lower proportion of positive labels in the training data (~3000 CD4+T labels and ~2000 TREG labels). Additionally for LDA, the optimal threshold is much lower (0.2), which can stem from a violation of the multivariate normal assumption in the PCA space. The larger F1-Score results from lowering the threshold to below 0.5 indicating that the increase in recall is larger than the decrease in precision. AdaBoost's optimal threshold remains close to 0.5 because the boosting algorithm's focus on "harder-to-classify" samples make it less sensitive to a class imbalance.

Due to the computational expense of the AdaBoost, the search for optimal number of principal components was limited. Given a more efficient coding of the cross valudation and time a wider search for the optimal number of principal components according to the F1-Score could reveal hidden relationships that would benefit classification.

Ultimately, after comparing the F1 Scores, we are choosing the **logistic regression with CV and ridge regularisation** to implement into our mypredict() function for Task 1.

If we could extend the analysis and the tuning of the classifiers at hand, we would focus more on the following aspects:

- **More searches for PCs**: Another improvement step could be to increase the amount of tested principal components to find a better number of principal components. This would help us to find hidden relationships between covariates, especially for the AdaBoost. In the case of AdaBoost we could also even train the model on the whole train data instead of only PCA data.
- **Tune other hyper parameters**: When looking at the AdaBoost model for decision trees, there are still other hyper parameters to be tuned. In the next step tuning the pruning parameter would be an option to optimize the model even more.

- **10 fold CV**: To improve the models even more, stepping up from a 5-fold CV to a 10-fold CV could make the choice of data sets more specific and accurate to increase the performance of all improved models.
- **Explore LDA's possibilities**: In addition to PCA as a dimension reduction tool, we would use LDA mainly as a dimension reduction tool to reduce dimensions to a single dimension which would allow us then to build a logistic classifier on top of it.