# Lab Course Machine Learning

**Exercise Sheet 4**

Prof. Dr. Dr. Lars Schmidt-Thieme
Kiran Madhusudhanan

HiWi: Basharat Mubashir Ahmed
Sophia Damilola Lawal

Submission deadline : **December 04, 2022 (Extended due to SRP presentations)**

### General Instructions

1. Perform a data analysis, deal with missing values and any outliers.
2. Unless explicitly noted, you are not allowed to use scikit, sklearn or any other library for solving any part.
3. Data should be normalized.
4. Train to Test split should be 80-20.
5. Convert any non-numeric values to numeric values. For example you can replace a country name with an integer value or more appropriately use one-hot encoding.

## 1. Regularization and Hyper Parameter Tuning (15 points)

### A. Extending the Optimization Routines and Loss Functions

In the previous assignment we implemented a class for ***Optimization***. Extend the class with an additional functionality for adapting the step-length.

1. Bold-Driver Step Length (slide 23/37)

Extend the ***Loss*** class implemented in the previous exercise with the following capabilities.

1. Case Weights (slide 33/37): Implement an additional ***balanced-cross-entropy*** loss function that takes care of the class imbalance within the dataset, by applying proportionally higher case weights to the minority class.
2. Regularization (slide 10/29): Implement ***L1-reg-cross-entropy*** and ***L2-reg-cross-entropy*** for L1 regularized and L2 regularized versions of the basic *cross-entropy* loss function.

For the optimization process, you would also need functions that calculate gradients for the same.

### B. Regularization and Model Selection

In this task, you work with the same data set as before, named *'logistic.csv'*. Use the previous implementation of class **LogisticRegression** for fitting a logistic regression model and predicting the results. Fit a logistic regression model with the following pair:

1. Cross Entropy Loss and Stochastic Gradient Descent.
2. Balanced Cross Entropy Loss and Stochastic Gradient Descent.
3. L1-Regularized Cross Entropy Loss and Stochastic Gradient Descent.
4. L2-Regularized Cross Entropy Loss and Stochastic Gradient Descent.

Tune the step length using the **bold driver** step length method as discussed in lecture slides.
Secondly, **do backward selection iteratively using AIC Metric (slide 6/29) and select the important features.** (Only do this for the combination of Cross-Entropy Loss and Stochastic Gradient Descent).
In the end, generate the loss trajectory for both training and testing for all cases mentioned above. Also report the final train and test accuracy. You are allowed to use sklearn for reporting the accuracy and plotting a confusion matrix.

## 2. K-Fold Cross Validation (5 points)

In this part of the assignment you will implement ***Grid Search*** with ***K-Fold Cross-Validation*** for model selection i.e., for choosing the best hyperparameters.

1. For this exercise we will use $k = 3$ fold validation and find optimal L2-regularization parameter $\lambda$ and step length $\alpha$. (Use L2-Regularized Cross Entropy Loss with Stochastic Gradient Descent and fixed step-length control).
2. Keep track of the mean performance across the $k$-folds for each of the hyper parameters.
3. Plot on the grid $\alpha$ vs $\lambda$ the Accuracy score for all combinations. You can use a 3D plot for the same with axis as $\alpha, \lambda$ and Accuracy.
4. Finally, for the optimal value of $\alpha$ and $\lambda$ train your model on complete training data and evaluate on test data.

## 3. **BONUS : Extending the Framework even further (5 points)

1. Extend the ***Optimization*** class by implementing another step length controller named RMSProp (slide 21/27 DL). Research about this Adaptive Step Length controller and explain how RMSProp helps to avoid exploding gradients. Implement the RMSProp algorith and compare its performance with Bold-Driver Step Length controller for the logistic regression problem.
2. Extend the ***Loss*** class by implementing functions for ElasticNet (slide 10/29) regularization. Compare the performance with L1 and L2 regularization implemented in the first part.