GLM Investigation

GLMs are commonly used in statistical modeling. Various frameworks utilize different optimization approaches to efficiently estimate parameters. Here's a comparison of optimization approaches among frameworks.

Module/framework/package	**Name and a brief description of the algorithm**	**An example of a situation where using the provided GLM implementation provides superior performance compared to that of base R or its equivalent in Python (identify the equivalent in Python)**
Base R (in the stats library)	It uses iteratively reweighted least squares (IRLS) to estimate GLM parameters. IRLS is an extension of weighted least squares in which weights are iteratively updated based on current estimates.	Suitable for small to medium-sized datasets where processing efficiency is not a primary priority.
Big data version of R	GLM fitting on large datasets is optimized using parallel computation and distributed frameworks such as Rcpp or data. table. For LASSO and Ridge regression, libraries such as bigglm (from biglm) or glmnet could be used.	Superior for large-scale healthcare datasets, where memory-efficient algorithms enable the study of millions of patient records.
Dask ML	Using gradient descent optimization techniques, GLMs can be computed in parallel and out of core. Handles huge datasets through incremental learning.	When working with extremely big datasets, such as national-level epidemiological research, it outperforms base R due to efficient computation distribution across several workers.
Spark R	Spark's distributed computing infrastructure is used in conjunction with	Suitable for handling enormous datasets, such as insurance claims data, that

	iterative optimization techniques such as L-BFGS (Limited-memory BFGS) to estimate parameters efficiently.	cannot fit in memory and require distributed processing.
Spark optimization	Uses advanced optimization techniques such as stochastic gradient descent (SGD) and L-BFGS to solve GLM problems in a distributed manner.	Superior in large-scale financial fraud detection models that demand rapid iteration across billions of transactions.
Scikit-learn	Lasso and Ridge regression are carried out using stochastic gradient descent (SGD) and coordinate descent algorithms. Implements efficient solvers such as 'lbfgs', 'liblinear', and ‘saga' based on the model type.	Superior for dealing with high-dimensional data (e.g., genetic risk prediction), where computing efficiency is critical, and Python's ecosystem facilitates integration with other ML frameworks.

Each GLM implementation is tailored to specific use cases. Base R is suitable for small datasets, whereas Big Data R, Dask ML, and Spark offer scalability for large datasets using parallel and distributed computation. Spark Optimization improves the efficiency of huge data processing, whereas Scikit-learn is appropriate for Python's high-dimensional challenges. The appropriate framework is determined by the amount of the data, computational resources available, and performance requirements.

CITATIONS: -

- R Core Team. *glm: Fitting Generalized Linear Models*. R Documentation. Published 2019. Accessed March 30, 2025. Available from: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/glm
- The Comprehensive R Archive Network. *High-Performance Computing View*. CRAN. Accessed March 30, 2025. Available from: https://cran.r-project.org/web/views/HighPerformanceComputing.html
- Dask Development Team. *Generalized Linear Models in Dask-ML*. Dask-ML Documentation. Accessed March 30, 2025. Available from: https://ml.dask.org/glm.html
- Apache Spark. *spark.glm: Fitting Generalized Linear Models in SparkR*. Apache Spark Documentation. Published 2023. Accessed March 30, 2025. Available from: https://spark.apache.org/docs/3.5.0/api/R/reference/spark.glm.html
- Apache Spark. *MLlib Optimization: Algorithms and Implementations*. GitHub. Accessed March 30, 2025. Available from: https://github.com/apache/spark/blob/master/docs/mllib-optimization.md
- Pedregosa F, Varoquaux G, Gramfort A, et al. *Scikit-learn: Machine Learning in Python – Linear Models*. Scikit-learn Documentation. Published 2024. Accessed March 30, 2025. Available from: https://scikit-learn.org/stable/modules/linear_model.html