GLM Optimization Investigation

Introduction:

This report provides an in-depth look at several statistical and machine learning frameworks, including Base R, Big Data R packages, Dask ML, Spark R, Spark MLlib, and Scikit-learn, with a focus on the optimization techniques used to fit Generalized Linear Models (GLMs).These frameworks cater to different computational needs, from handling small in-memory datasets to managing large-scale distributed data. Each implementation applies unique optimization strategies tailored to its performance goals. The aim is to assess their optimization strategies and effectiveness, highlighting the situations where each framework performs better than its Base R or Python counterparts.

GLM Optimization Comparison Table:

Module/framework/package	Name and a brief description of the algorithm	Example (compared to Base R or its equivalent in Python)
Base R (stats::glm)	A common technique for fitting GLMs in R's statistics package is Iteratively Reweighted Least Squares (IRLS). Weighted least squares are applied iteratively until convergence is achieved in order to solve GLMs.	The GLM in Base R works well for academic and statistical modeling as well as small to medium-sized datasets. Performs better in terms of formula syntax and simplicity of use than Python's stats models.

Big Data R (bigstatsr)	IRLS designed for large-scale matrix operations on disk-stored datasets is used by bigstatsr to handle data larger than RAM by memory mapping and parallel processing.	Handles large datasets (such genomic data with 10M+ observations) that are larger than RAM and uses disk-backed matrix operations to outperform base R.
Dask ML	For regularized GLM fitting, coordinate L-BFGS and Descent (Lasso and Ridge). Designed for parallel processing and out-of-core computing on distributed datasets in mind.	Works with big tabular datasets that are too big to fit in memory, such online click data. Uses Dask's parallel cluster backend to scale, outperforming scikit-learn.
Spark R (sparklyr)	IRLS optimization with the MLlib GLM module wrapped in sparklyr, utilizing Spark's distributed computing backend. Scalable GLM fitting on big clusters is made possible by it.	Modeling the behavior of millions of users is done in distributed environments like Hadoop or Kubernetes. More effective than Base R when used alone.

Spark Optimization (MLlib)	Optimizes GLMs with L1/L2 regularization using OWL-QN (Orthant-Wise Limited-memory Quasi-Newton) or L-BFGS methods, which are appropriate for sparse, huge data.	R or Python local models are outperformed by Spark MLlib's L-BFGS or OWL-QN optimization for sparse high-dimensional text data (such as spam detection).
Scikit-learn (Python)	For regularized GLMs, coordinate SAG (Stochastic Average Gradient) and descent. Cross-validation is supported and Scikit-learn's implementation is effective for medium-to-large data.	Uses elastic net regularization to handle high-dimensional datasets (bioinformatics, for example). More effective than StatsModels or Base R because of the compiled Cython backend.

Conclusion:

In summary, Base R's glm() function is well-suited for traditional statistical modeling on small datasets but struggles with large-scale data and high-performance parallelism. Tools like Bigstatsr and Dask ML enhance GLM capabilities by leveraging memory mapping and parallel/distributed computing. Spark-based implementations, such as Spark R and MLlib, offer scalable solutions ideal for cluster and cloud environments. For regularized GLMs in Python, Scikit-learn strikes a balance between speed and flexibility. Hence, the choice of GLM implementation depends on factors such as dataset size, available computing resources, and performance requirements.