Active Learning System Diagram

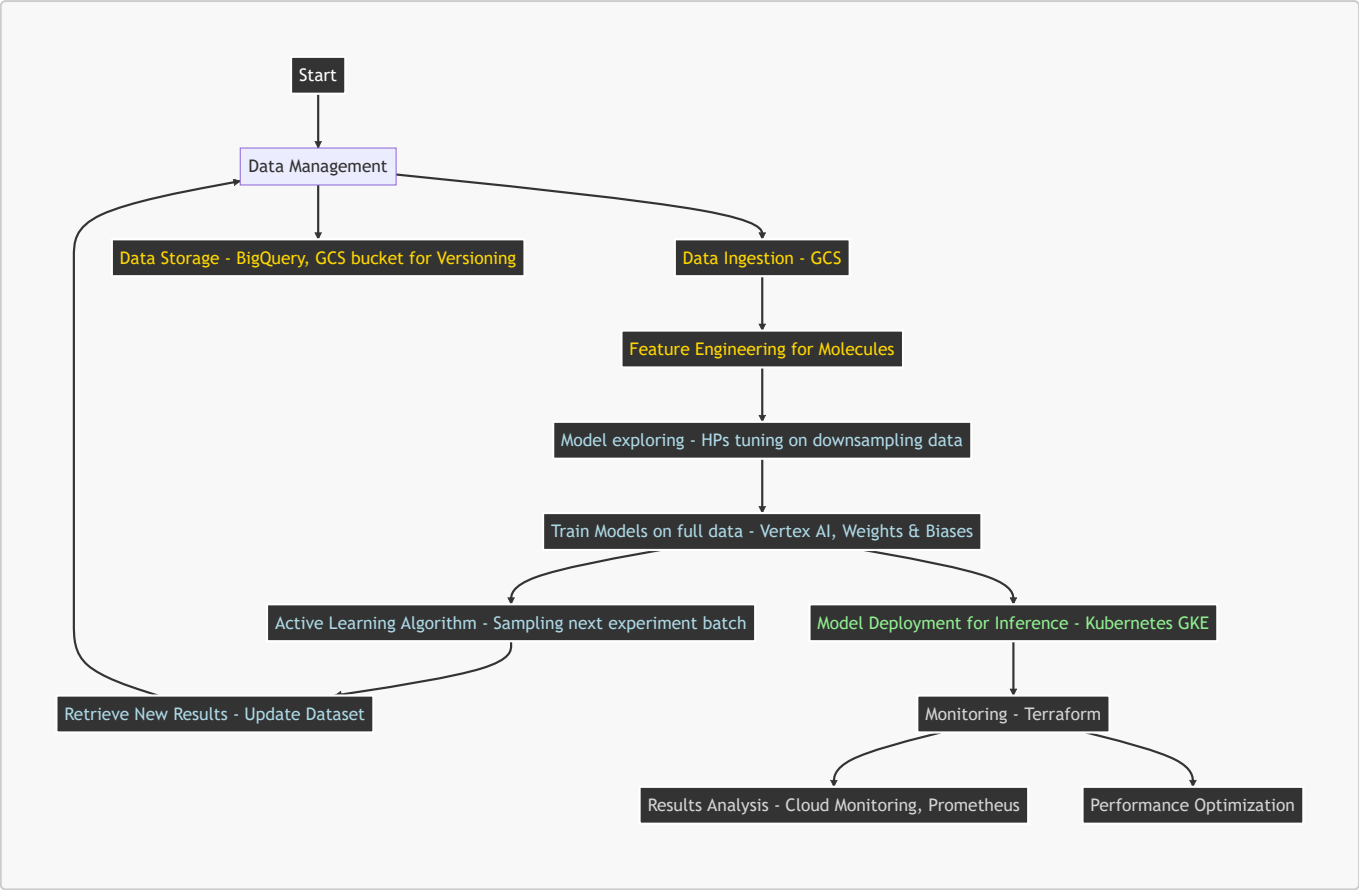

Diagram Description:

1. **Start**: The initiation of the machine learning workflow.
2. **Data Management**:
   - **Data Ingestion**: Data from various sources is collected, processed and stored temporarily in Google Cloud Storage (GCS).
   - **Data Storage and Processing**: Data is permanently stored and managed in BigQuery, where Dataflow assists in processing and GCS Bucket is utilized for dataset versioning.
3. **Feature Engineering for Molecules**:
   - Process and transform molecular data using various of featurizers to extract relevant features necessary for training the model, cache them in GCP.
4. **Model Exploring**:
   - Perform hyperparameter tuning on a downscaled version of the dataset to determine optimal model settings.
5. **Train Models on Full Data**:
   - Once optimal hyperparameters are identified, train the models on the complete dataset using Vertex AI for deep learning tasks and Weights & Biases for tracking experiments and model versions.

6. **Active Learning Algorithm**:
    - Apply an active learning algorithm to determine the most informative batch of experiments to conduct next, which helps in efficiently using experimental resources and improving model performance iteratively.
7. **Retrieve New Results - Update Dataset**:
    - Incorporate the results from the latest experiments back into the dataset, completing the active learning feedback loop.
8. **Model Deployment for Inference**:
    - Deploy the trained model into production using Kubernetes on Google Kubernetes Engine (GKE) to handle scalable, containerized application deployment.
9. **Monitoring**:
    - Utilize Terraform for infrastructure management, ensuring that resources are appropriately scaled and managed according to demand.
10. **Results Analysis and Performance Optimization**:

- Monitor system and model performance using Google Cloud Monitoring and Prometheus, analyzing results to continuously optimize the system's effectiveness and efficiency.

Key Design Considerations:

- **Model/Data Drift:** This design addresses these challenges by incorporating an Active Learning Loop, which iteratively updates the dataset and retrains the model with new results, ensuring the system adapts to changes dynamically. Additionally, continuous Monitoring and Optimization stages help detect and respond to drift by analyzing performance metrics and adjusting model parameters or retraining as needed.
- **CI/CD:** Our flowchart incorporates CI/CD to streamline and automate the deployment of updated machine learning models into production. By using Kubernetes (GKE) for scalable deployment and Terraform for infrastructure management, the system ensures that new versions of the model are tested, built, and deployed efficiently and reliably, maintaining system integrity and responsiveness.
- **Version Control for Data and Models**: GCS bucket track the data version and Weights & Biases ensures that both data and models can be rolled back to previous versions if needed, which is crucial for maintaining the integrity and reproducibility of experiments.

Scalability Considerations:

- **Handling Large Dataset Sizes**: BigQuery's ability to handle petabytes of data and Dataflow's scalable data processing capabilities ensure that the system can scale to meet the demands of large datasets seamlessly.
- **High Throughput Inference Requests**: Kubernetes GKE provides robust, scalable infrastructure management, allowing for efficient scaling of inference services. This setup can handle sudden surges in inference requests by dynamically adjusting resource allocations.
- **Performance Monitoring and Optimization**: Continuous monitoring with Google Cloud Monitoring and Prometheus ensures that any performance bottlenecks are quickly identified and addressed.

Terraform's infrastructure as code approach allows for rapid scaling adjustments and deployment of resources based on real-time demands.