Accelerating Genomic Discovery with Apache Spark

Databricks Unified Analytics Platform for Life Sciences

Agenda

11:00AM	Opening Remarks
11:45AM	Lunch
12:30PM	Workshop #1: Accelerating Variant Calls with Apache Spark
1:30PM	Workshop #2: Characterizing Genetic Variants with Spark SQL
2:30PM	Workshop #3: Disease Risk Scoring with Machine Learning

databricks

databricks®

Unified data analytics platform for accelerating innovation across data science, data engineering, and business analytics

Global company with 5,000 customers and 450+ partners

Original creators of popular data and machine learning open source projects

APACHE Spark™ | DELTA LAKE™ | mlflow™

Genomic Data Powers a Precision Revolution

Genomics married to EHR data gives direct insight to molecular phenotype

Accelerate Target Discovery

Reduce Costs via Precision Prevention

Improve Survival with Optimized Treatment

Big Data, Bigger Problems

"Hidden Technical Debt in Machine Learning Systems," Google NIPS 2015

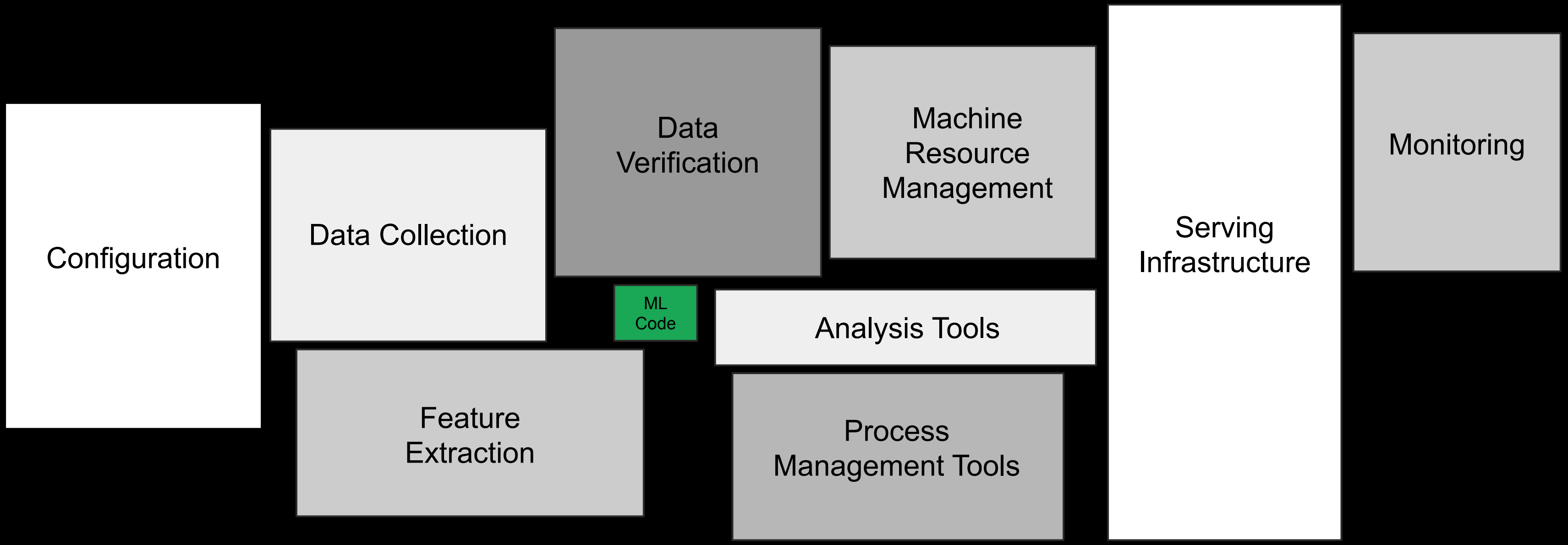
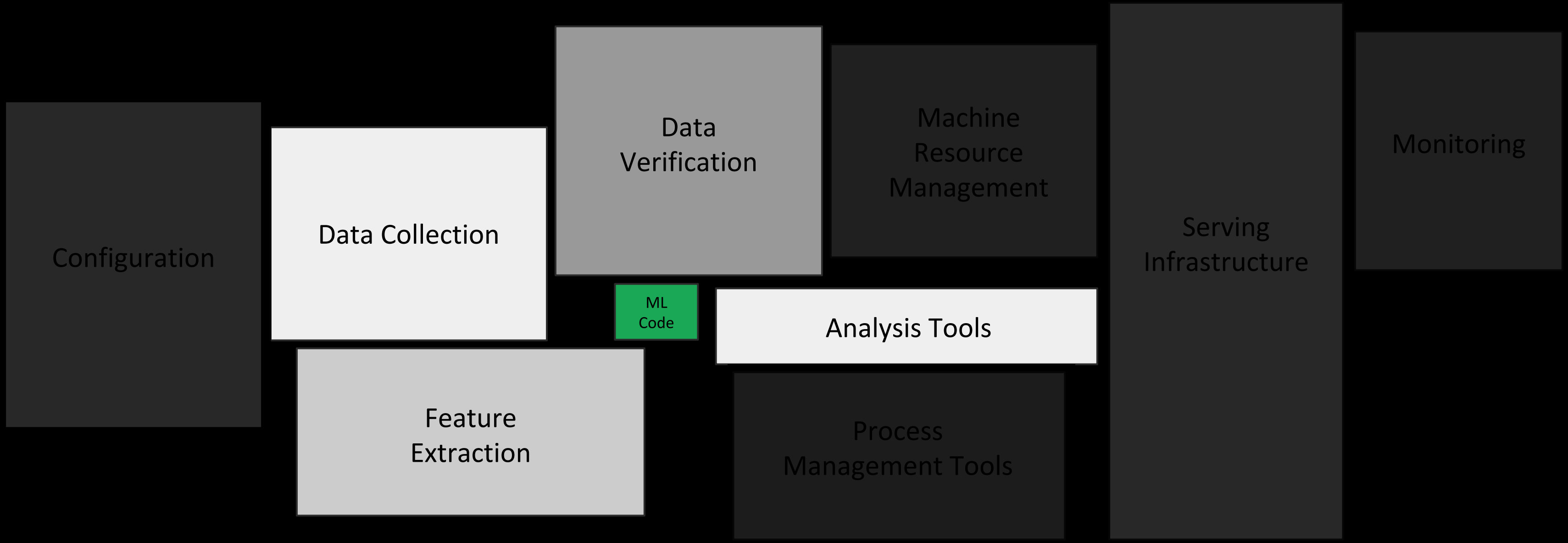


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small green box in the middle. The required surrounding infrastructure is vast and complex.

databricks

Supporting genomic-scale data

Supporting genomics at the scale of millions of patients

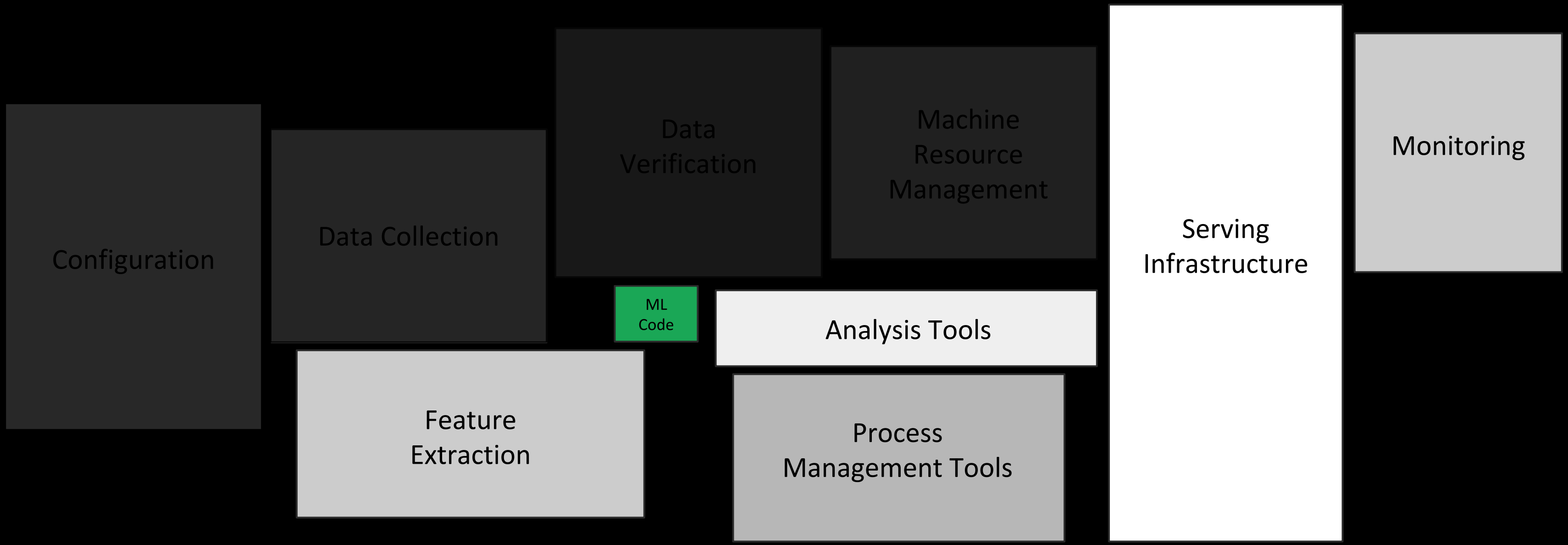


We can build easy onramps that allow medical data scientists, bioinformaticians, and biostaticians to ask and answer population health questions

databricks

Solve for “production” in the life sciences

Integrating reproducible and interpretable ML in the life sciences

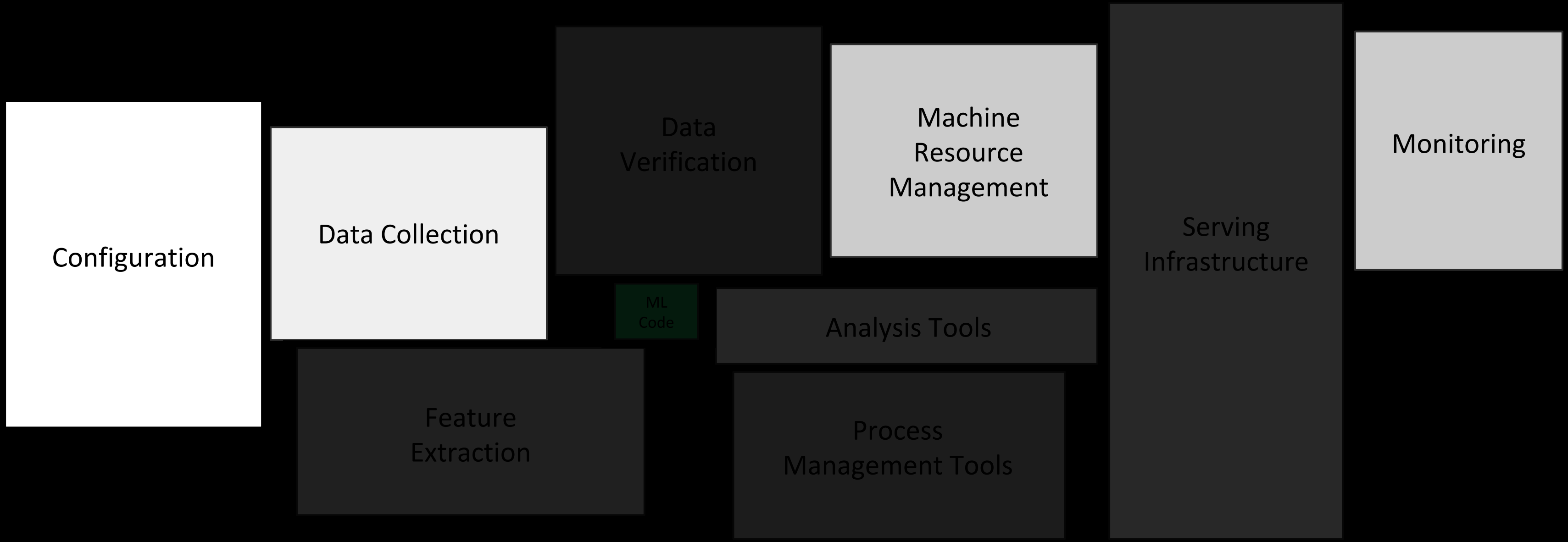


We can provide a ML ecosystem that ensures that ML models are reproducible and interpretable, while maximizing access to ML

databricks

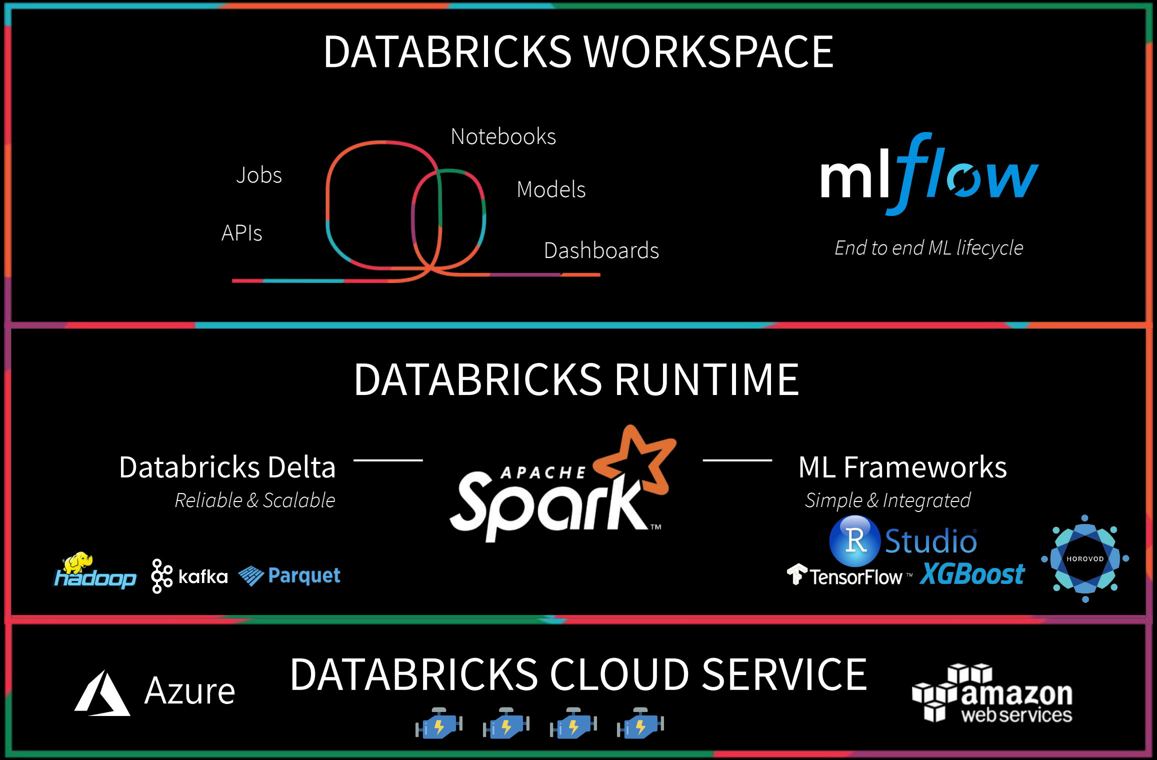
Agility with security

Provide elastic compute with fine grained security



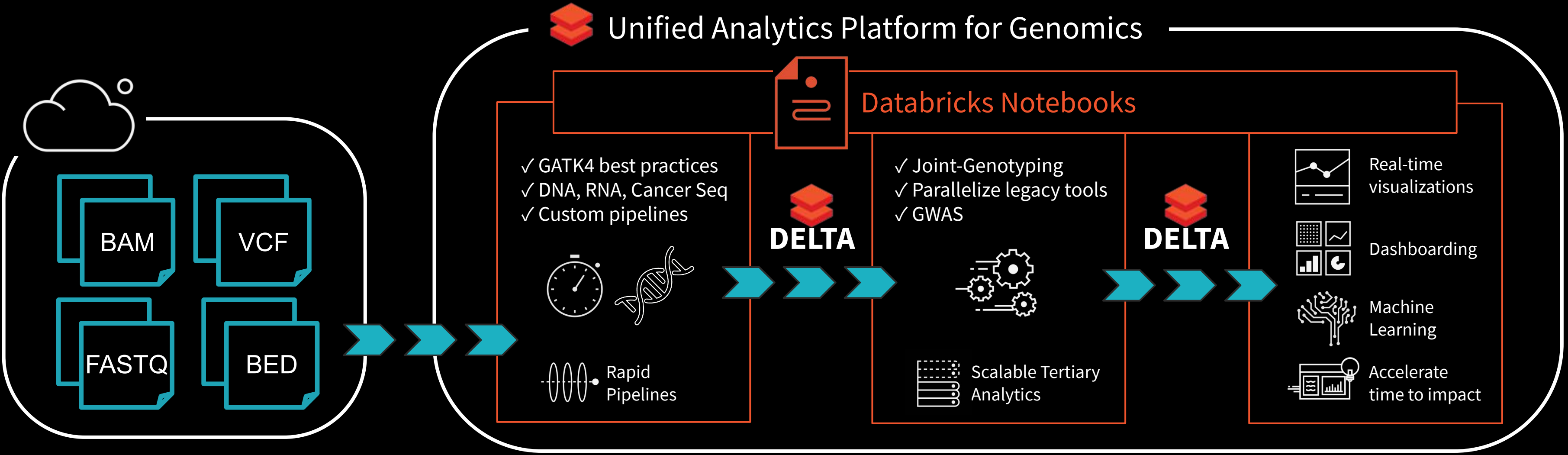
We can build a platform where each component provides fine-grained security and auditibility, while minimizing the impact of security on the end user

Databricks Unified Analytics Platform



Introducing Unified Analytics for Genomics

Collaborative platform for interactive genomic data processing and analytics at massive scale



databricks

REGENERON® GENETICS CENTER & databricks®

GLOW

projectglow.io

The power of big genomic data

Accelerate Target Discovery

Reduce Costs via Precision Prevention

Improve Survival with Optimized Treatment

databricks

The power of big genomic data

Accelerate Target Discovery

Reduce Costs via Precision Prevention

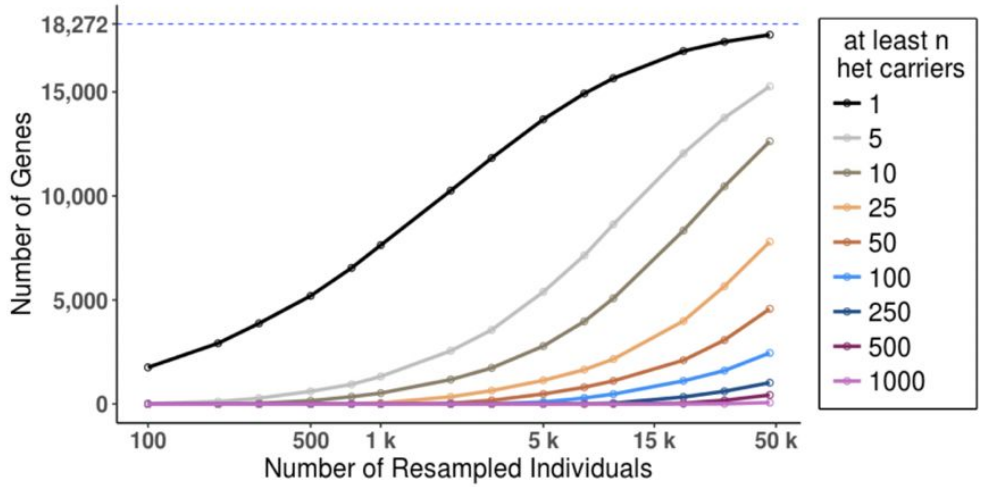
Improve Survival with Optimized Treatment

databricks

The power of big genomic data

- Identifying carriers of rare, putative loss-of-function (pLOF) variants across all genes requires large sample sizes
- Homozygous pLOF carriers ("human knockouts") are even more rare (~1k genes have >= 1 carrier in 50k samples)
- Detecting protective pLOF disease associations requires many carriers per gene

Het pLOF carrier counts by gene with increasing sample size

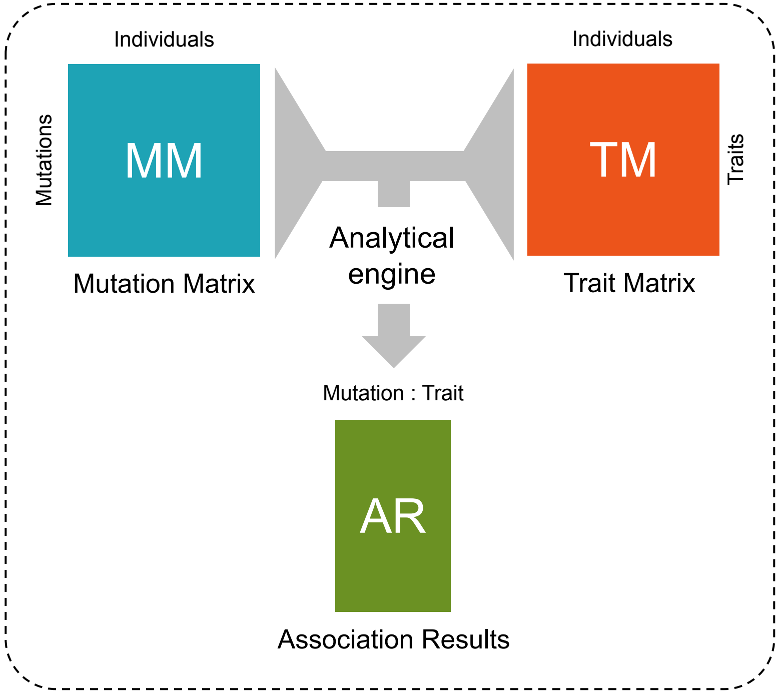


C Van Hout, *et al*. (2019) Whole exome sequencing and characterization of coding variation in 49,960 individuals in the UK Biobank. *bioRxiv*.

How do we analyze our data to gain novel insights?

- **Approach**:
    1. Sequence a large number of individuals from many cohorts (>70 to date)
    2. Obtain paired phenotypic data (e.g. de-identified electronic medical records)
    3. Run all-vs-all association tests between all mutations and traits
    4. Mine association results to extract actionable insights
    5. Design for scalability & automation

Desired goal

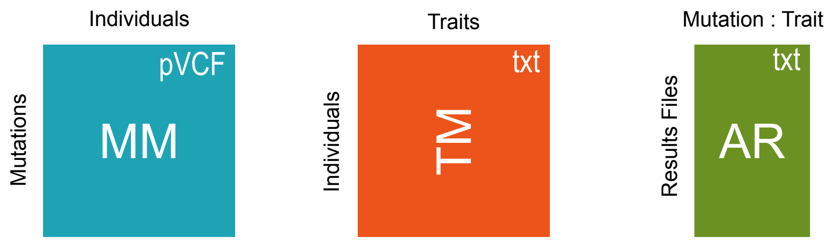


databricks

#EntSAIS14

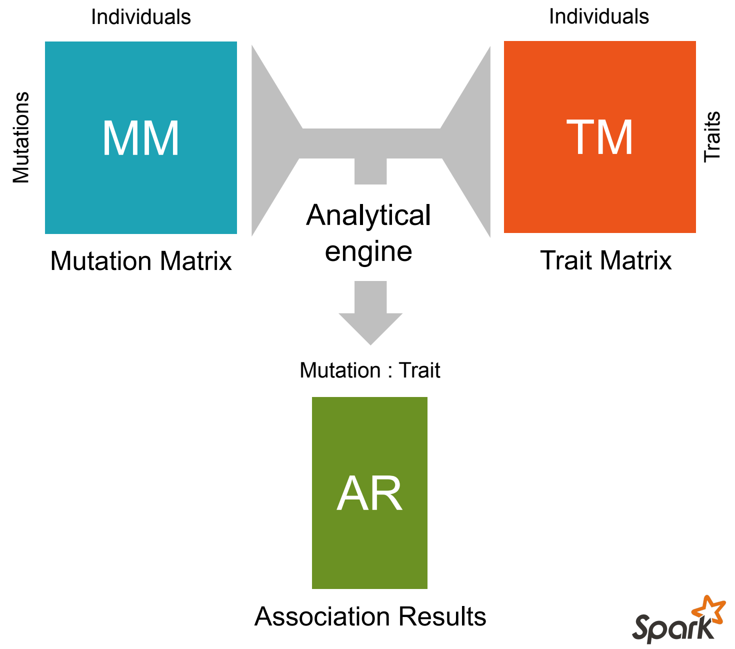
How do we analyze our data to gain novel insights? It’s complicated.

Reality



- **Hard to scale:** Data is decentralized, assumes single node architecture
- **Rigid tools:** Data is organized in different ways (e.g., not squared off, transposed, custom representations and indexing)
- **Hard to move to tertiary analyses:** Asking simple questions at scale requires time-consuming data wrangling steps

Desired goal



databricks

#EntSAIS14

GLOW

- **Open-source toolkit for large-scale genomic analysis**
- Built on Spark for biobank scale
- Query and use built-in commands with familiar languages using Spark SQL
- Compatible with existing genomic tools and formats, as well as big data and ML tools

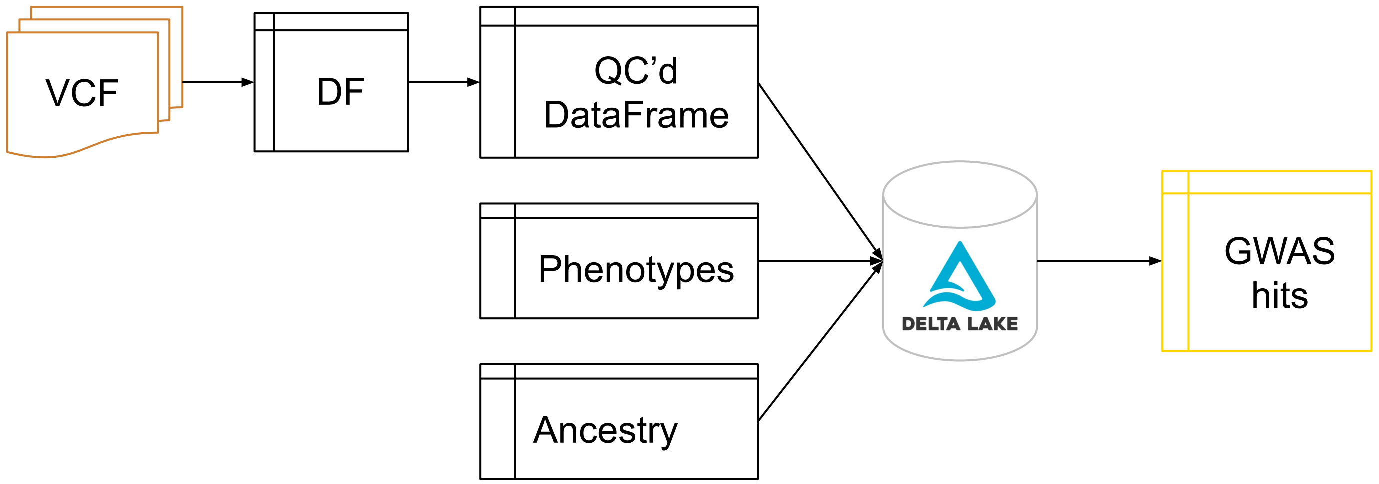
Built-in functions

- Convert genotype probabilities to hard calls
- Normalize variants
- Liftover between reference assemblies
- Annotate variants
- Genome-wide association studies
- ...

Built-in functions

- Convert genotype probabilities to hard calls
- Normalize variants
- Liftover between reference assemblies
- Annotate variants
- Genome-wide association studies
- ...

GWAS pipeline



GWAS

- Load variants
- Perform quality control
- Control for ancestry
- Run regression against trait
- Log Manhattan plot

GLOW

```
spark.read.format("vcf") \
  .load("genotypes.vcf")
```

databricks

GWAS

- Load variants
- Perform quality control
- Control for ancestry
- Run regression against trait
- Log Manhattan plot

```
variant_df.selectExpr("*", \
     "expand_struct(call_summary_stats(genotypes))", \
     "expand_struct(hardy_weinberg(genotypes))") \
  .where((col("alleleFrequencies").getItem(0) >= \
           allele_freq_cutoff) & \
         (col("alleleFrequencies").getItem(0) <= \
           (1.0 - allele_freq_cutoff)) & \
         (col("pValueHwe") >= hwe_cutoff))
```

GWAS

- Load variants
- Perform quality control
- Control for ancestry
- Run regression against trait
- Log Manhattan plot

DELTA LAKE

```
qc_df.write \
  .format("delta") \
  .save(delta_path)
```

databricks

GWAS

- Load variants
- Perform quality control
- Control for ancestry
- Run regression against trait
- Log Manhattan plot

Spark MLlib

```
matrix.computeSVD(num_pcs)
```

databricks

GWAS

- Load variants
- Perform quality control
- Control for ancestry
- **Run regression against trait**
- Log Manhattan plot

GLOW

```
genotypes.crossJoin( \
  phenotypeAndCovariates) \
  .selectExpr(
    “expand_struct( ” \
    “linear_regression_gwas( ” \
    “genotype_states(genotypes), ” \
    “phenotype_values, covariates))”)
```

GWAS

- Load variants
- Perform quality control
- Control for ancestry
- Run regression against trait
- Log Manhattan plot

```
gwas_results_rdf <- as.data.frame(gwas_results)
install.packages("qqman",
  `repos="http://cran.us.r-project.org") library(qqman)
png('/databricks/driver/manhattan.png')
manhattan(gwas_results_rdf)
```

GWAS

- Load variants
- Perform quality control
- Control for ancestry
- Run regression against trait
- Log Manhattan plot

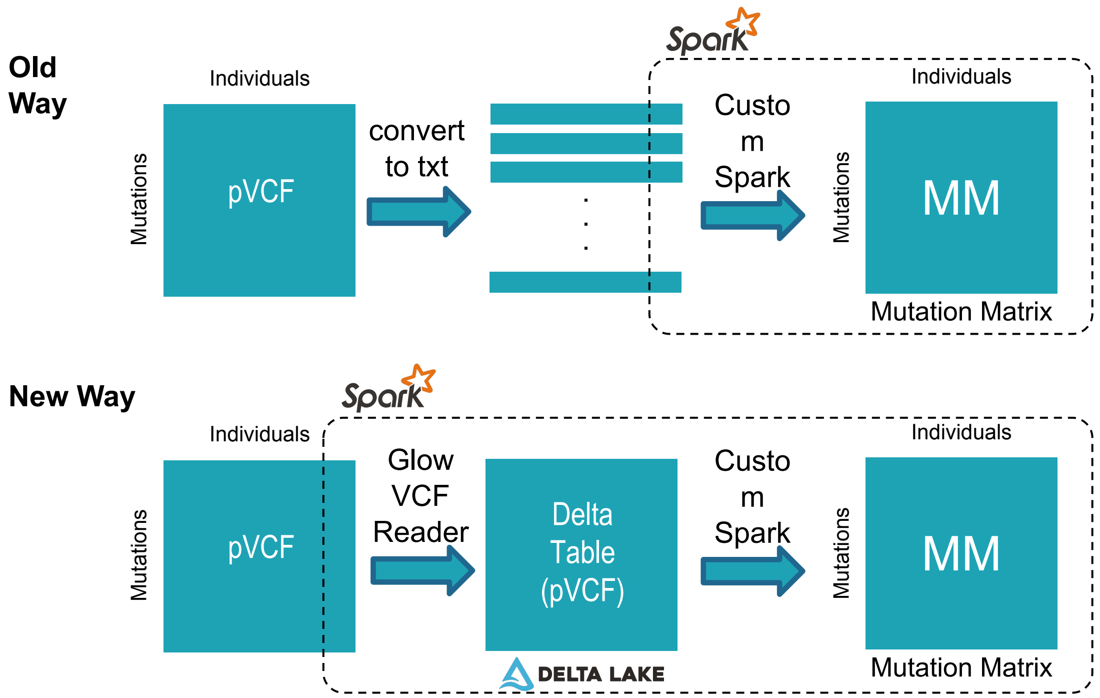
```
mlflow.log_artifact( \
  '/databricks/driver/manhattan.png')
```

Migrating VCF ingestion to Glow

- With Glow, we no longer need a custom VCF derivative for Spark ingestion
- Greatly reduces ETL code complexity/scalability:

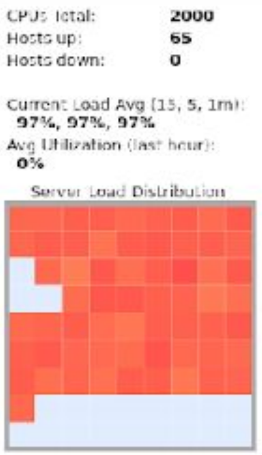
```
val pvcfDF = spark.read
  .format("vcf")
  .load(s"${vcfPath}/*.vcf.gz")

pvcfDF
  .write
  .partitionBy("contigName")
  .format("delta")
  .save(outputPath)
```

- pVCF now available as Delta table
- Similar process for BGENs



Glow VCF Reader: Processing a 6Tb pVCF with 2000 cores in 5 hours

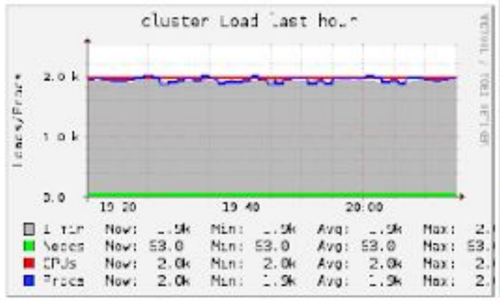
- Parallelization for “free” with Spark
- ~100% CPU utilization
- ~12Tb RAM usage
- Splittable VCF read: scales linearly with cluster size
- Output has a schema!
  - Columnar
  - Can use Spark SQL, Python, Scala, R, piping

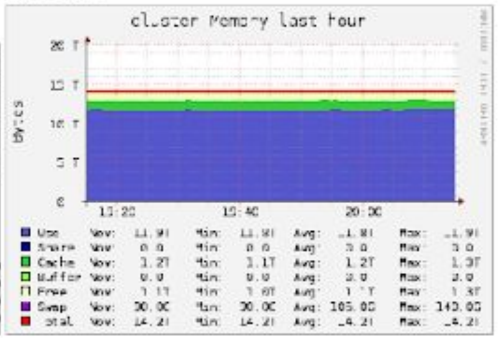


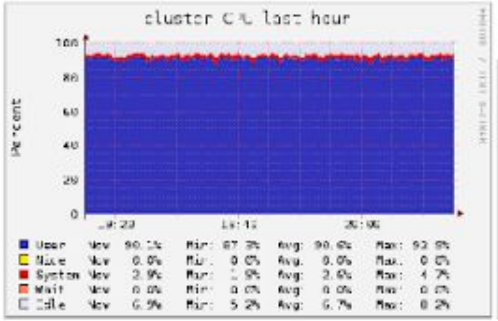
CPUs Total: 2000
Hosts up: 65
Hosts down: 0

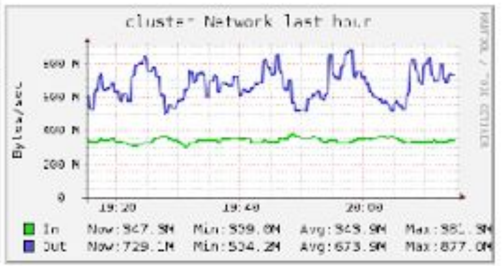
Current Load Avg (15, 5, 1m):
97%, 97%, 97%
Avg Utilization (last hour):
0%

Server Load Distribution

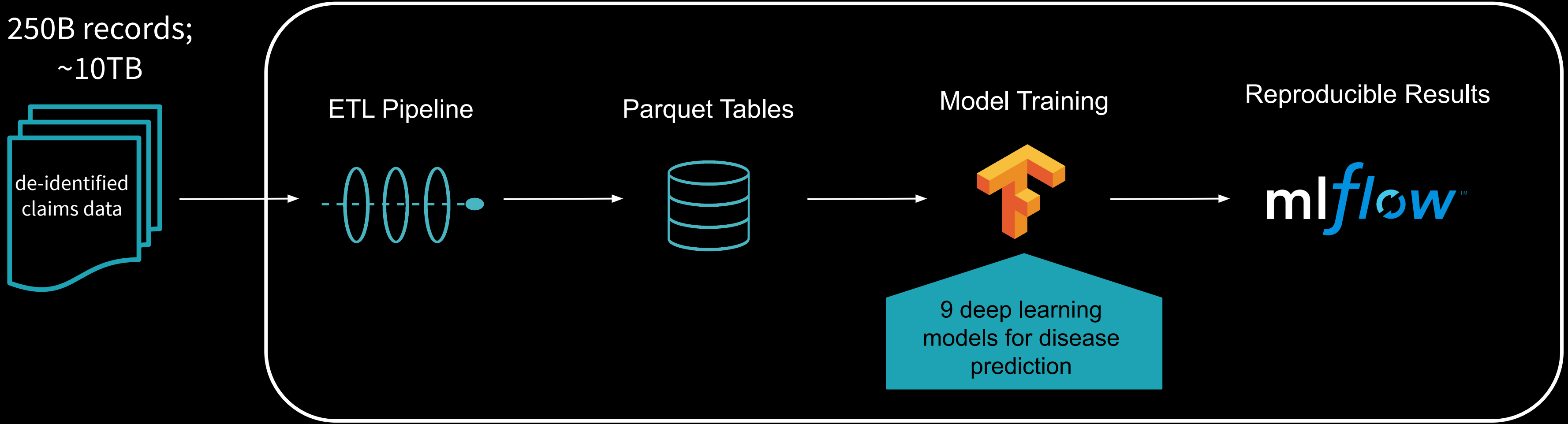








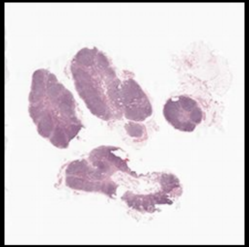
Stroke Prediction with Real World Evidence

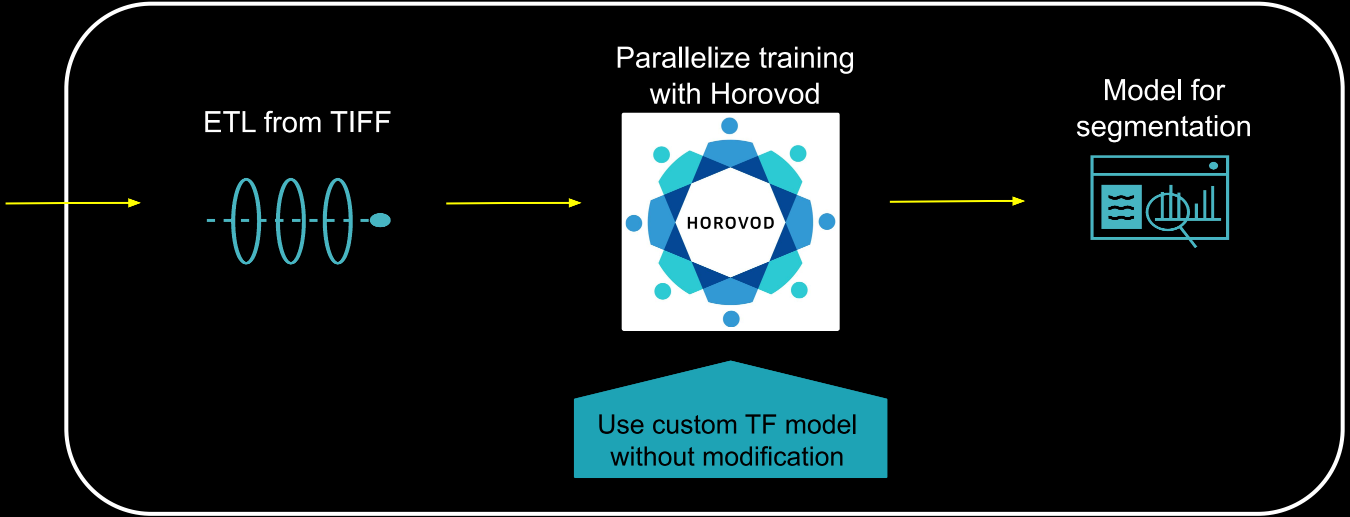


RESULTS

- Prior to Azure Databricks: Static on-prem spark cluster shared with 80 people (MapR); hard to manage; frequent job failures
- On Azure Databricks: **It just works**!

Deep learning on cellular imaging




ETL from TIFF
Parallelize training with Horovod
HOROVOD
Model for segmentation
Use custom TF model without modification

RESULTS

- Prior to Databricks: takes 1 week to process 700GB of whole slide images, cannot scale to full internal dataset
- On Databricks: leverage Horovod runner to accelerate 1 week training time down to 15 minutes

Agenda

11:00AM	Opening Remarks
11:45AM	Lunch
12:30PM	Workshop #1: Accelerating Variant Calls with Apache Spark
1:30PM	Workshop #2: Characterizing Genetic Variants with Spark SQL
2:30PM	Workshop #3: Disease Risk Scoring with Machine Learning

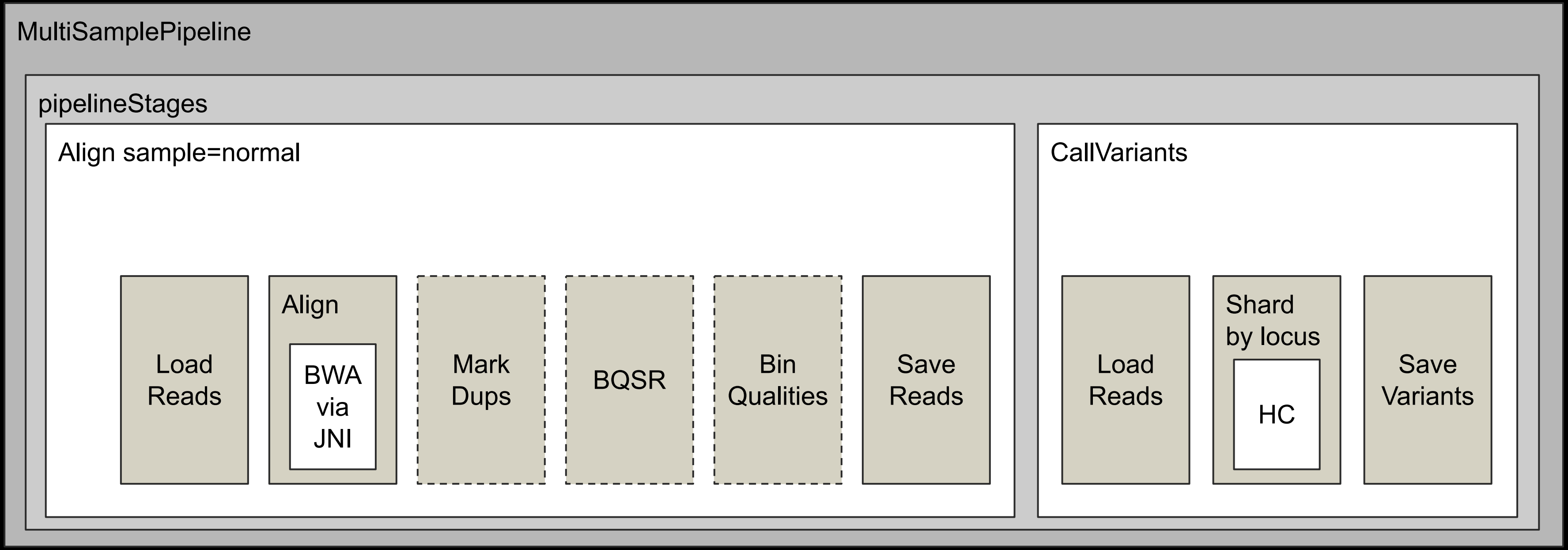
databricks

Running Genomics Pipelines on Databricks

UAP4G DNA-seq pipeline

- Pipeline is a “functionally equivalent” pipeline
  - Supports common preprocessing steps (MarkDups, Qual Binning, BQSR), with full read-level concordance
  - Runs HaplotypeCaller for genotyping, can emit both VCF- and gVCF-style output
- Can optionally run annotation (via SnpEff) on all called sites
- Accepts FASTQ, SAM/BAM/CRAM as input, can support multi-flow cell library designs
- Defaults to emit data in Parquet/Delta, but can save back to VCF
- Is a “zero-setup” pipeline

Pipeline Architecture

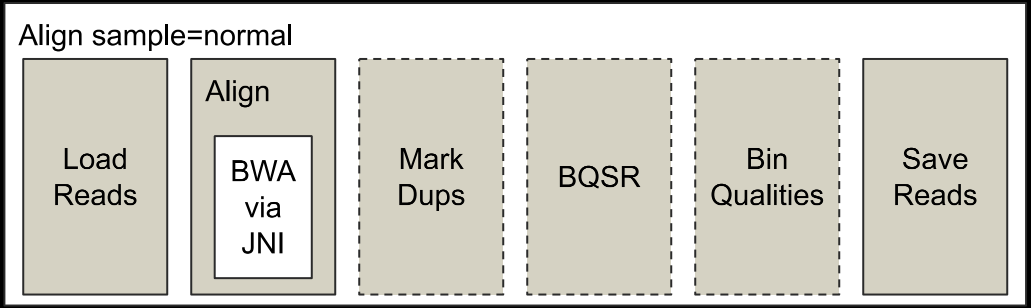


databricks

Compare/contrast vs. GATK4

- OSS GATK4 Spark-based variant calling pipeline is in beta:
    - Significant concordance issues in Spark HaplotypeCaller
    - Significant performance issues in Spark BQSR
- Differences relative to GATK4:
    - Use ADAM's BQSR and duplicate marking implementations
    - Use highly optimized custom SQL transformer for quality score binning
    - Use custom parallelization of HaplotypeCaller
- Custom sharding of HaplotypeCaller regions achieves full concordance with GATK4 single-node
- Additionally, use custom memory management strategy to allow use of compute-optimized instances

Alignment pipeline



- Can load reads from SAM/BAM/CRAM/FASTQ
- Executes GATK BWA JNI bindings from within Spark to parallelize alignment
- Custom preprocessing stages are >3x faster than GATK4 stages
- Reads are saved to Parquet and can be saved to BAM as well

Preprocessing stages pipeline

Mark Dups

- Custom implementation, based on ADAM MarkDups (which is based on Picard MarkDups), ~6x faster than GATK
- 100% concordant with Picard, with support for chimeras

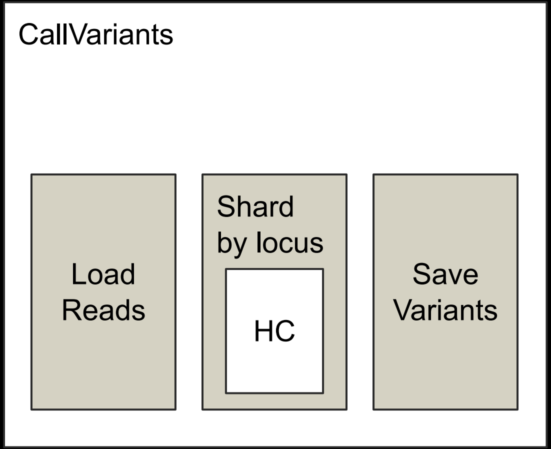
BQSR

- Leverages ADAM's BQSR implementation
- >99% concordant with GATK3, >2x faster

Bin Qualities

- Custom Spark SQL implementation, effectively free

Variant calling pipeline



- Complete rewrite of parallelization infrastructure in GATK4 OSS:
  - Achieves full concordance on a locus-by-locus basis for HaplotypeCaller/M2
  - Achieves a 2x performance improvement with scalability to 1000's of cores
- Leverages direct reuse of core HaplotypeCaller/M2 algorithms
- Saves to both Parquet and VCF

Benchmarks

Platform	Coverage	Reference Confidence Mode	Runtime
Databricks	30x	VCF	24m29s
Databricks	30x	GVCF	39m23s
Edico	30x	VCF	1h27m
Edico	30x	GVCF	2h29m

- Scale out to 300x coverage WGS = 2.6hrs at a compute cost of $65
- Compare to GATK4 Spark pipeline at >4hrs, >$15
- Compare to GATK4 single node at >30hrs, ~$5 for VCF

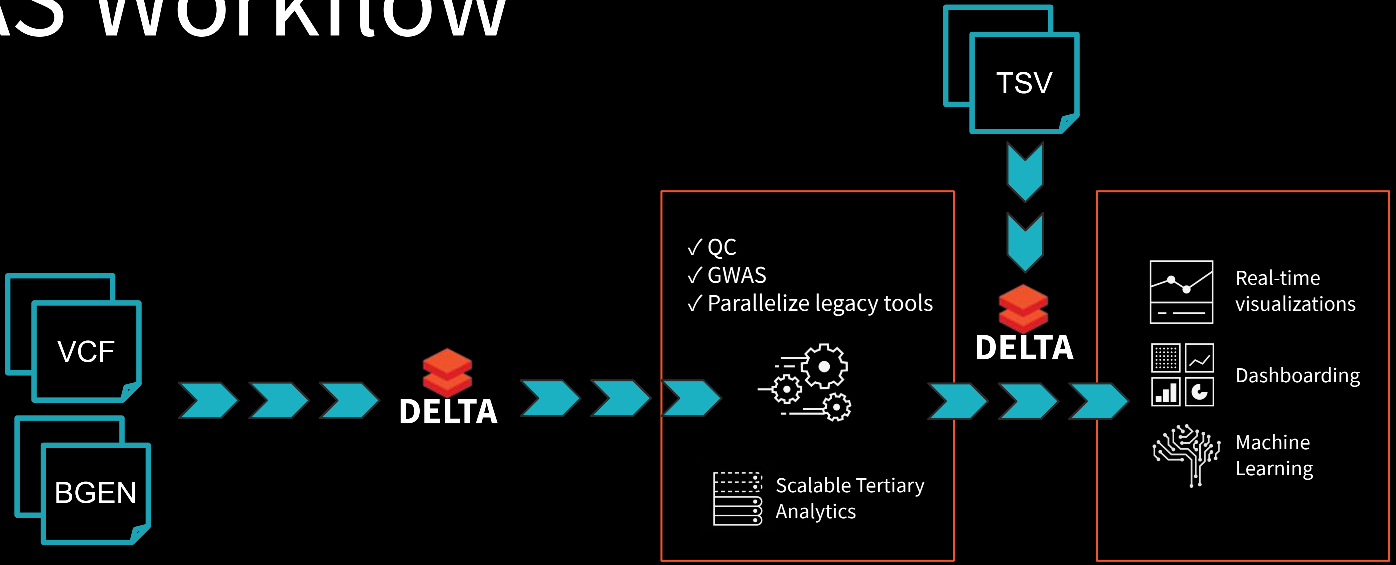
Agenda

11:00AM	Opening Remarks
11:45AM	Lunch
12:30PM	Workshop #1: Accelerating Variant Calls with Apache Spark
1:30PM	Workshop #2: Characterizing Genetic Variants with Spark SQL
2:30PM	Workshop #3: Disease Risk Scoring with Machine Learning

databricks

GWAS on Spark

GWAS Workflow



- Ingest VCF/BGEN and GWAS summary statistics into Delta
- Run QC and GWAS on Delta tables through either R or Python
- GWAS summary statistics in Delta support interactive query for exploration/dashboarding

databricks

Agenda

11:00AM	Opening Remarks
11:45AM	Lunch
12:30PM	Workshop #1: Accelerating Variant Calls with Apache Spark
1:30PM	Workshop #2: Characterizing Genetic Variants with Spark SQL
2:30PM	Workshop #3: Disease Risk Scoring with Machine Learning

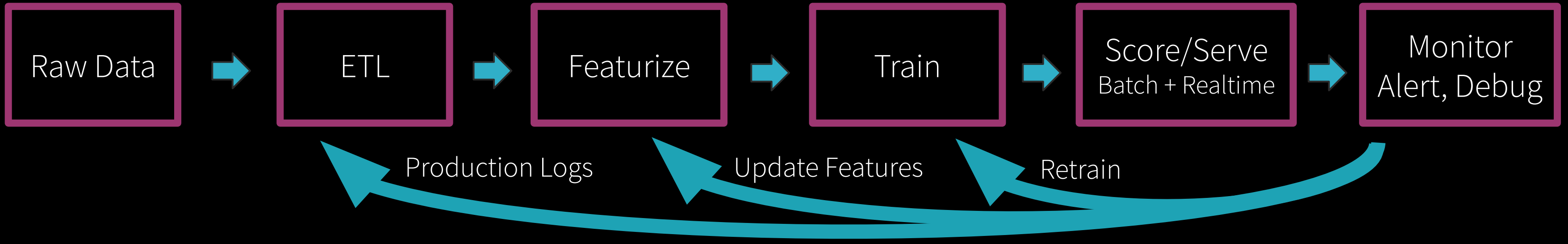
databricks

ML on Gene Expression

ML Lifecycle and Challenges

mlflow

An open source platform for the machine learning lifecycle



kafka S3 Delta Spark SQL PyTorch learn TensorFlow Azure Machine Learning Amazon SageMaker docker Spark
hadoop Microsoft Azure Blob Storage mongoDB pandas learn R Spark dmlc XGBoost

Zoo of Ecosystem Frameworks

Tuning | Deploy | Model Mgmt

Collaboration | Scale | Governance

- Feature Repository
- Experiment Tracking
- AutoML, Hyper-p. search
- Remote Cloud Execution
- Project Mgmt (scale teams)
- Model Exchange
- A/B Testing
- CI/CD/Jenkins push to prod
- Orchestration (Airflow, Jobs)
- Lifecycle mgmt.
- Data Drift
- Model Drift

databricks

MLflow Components

mlflow
Tracking

Record and query experiments: code, data, config, results

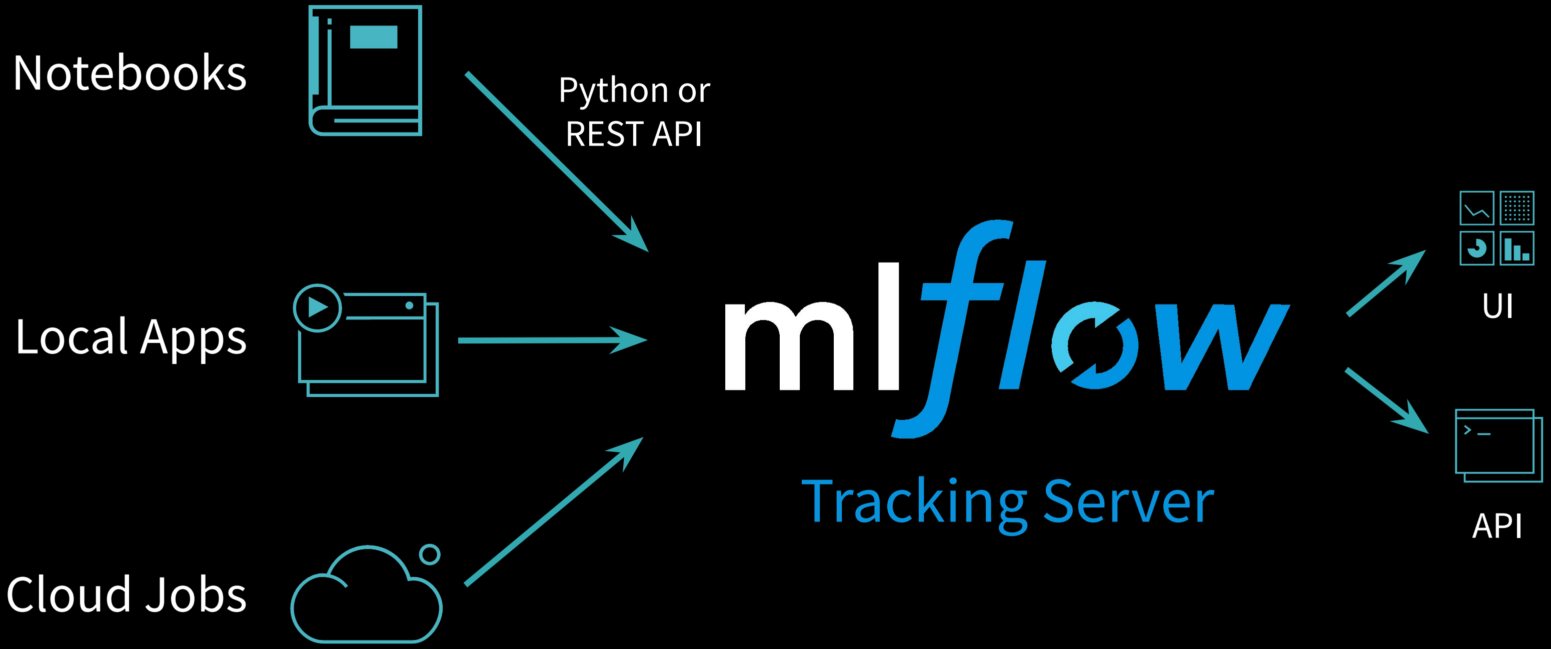
mlflow
Projects

Packaging format for reproducible runs on any platform

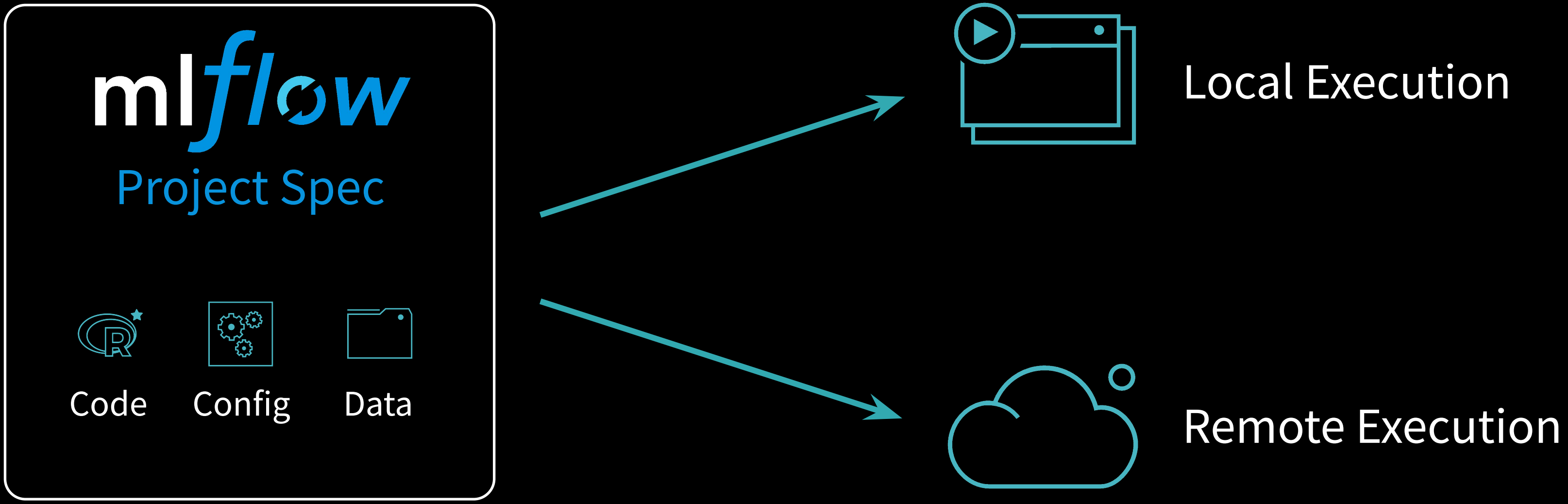
mlflow
Models

General model format that supports diverse deployment tools

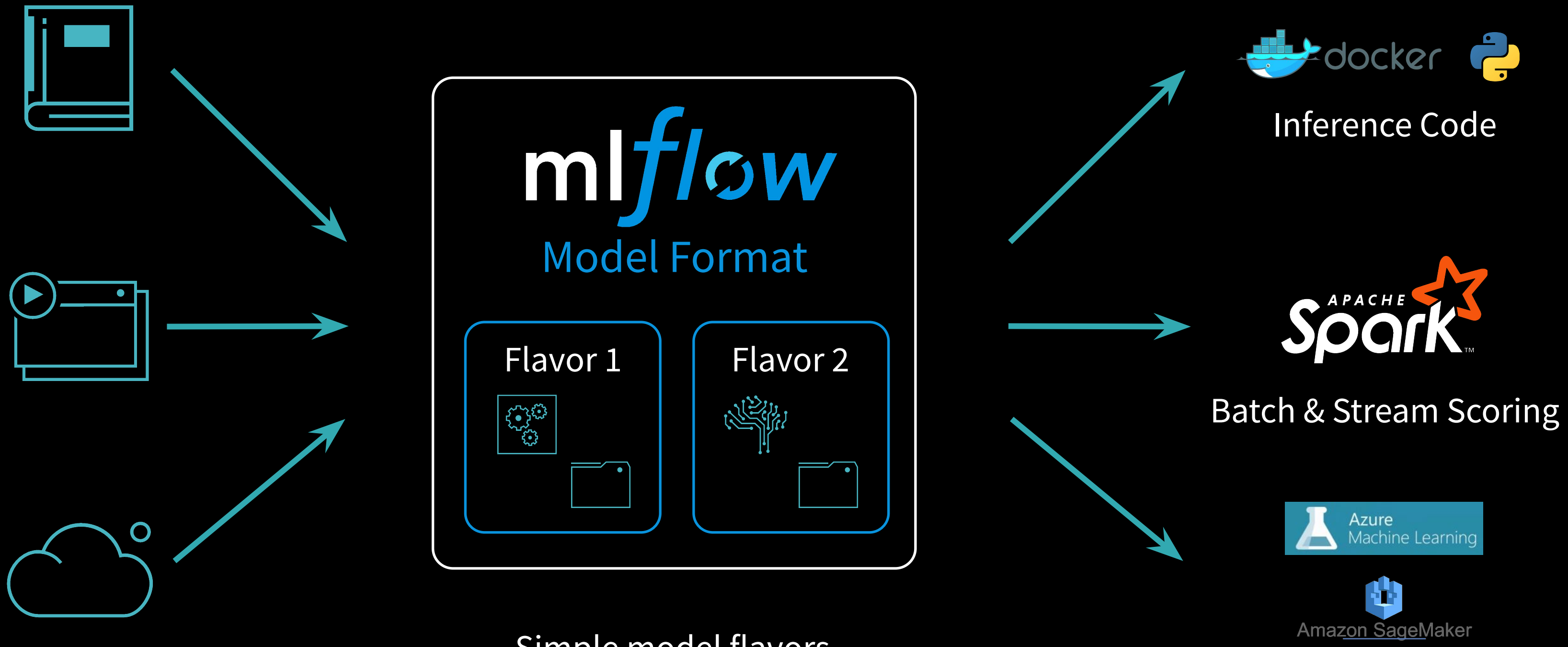
MLflow Tracking



MLflow Projects



MLflow Models



Run Sources

Simple model flavors
usable by many tools

Questions?