Disease Prediction Using Big Data Analytics with PySpark

Intro to Big Data Analytics, Fall 2025

TEAM 3

Laman Panakhova, Shahla Azizova, Khadija Ahmadova, Mahbuba Jafarzada, Emil Hajiyev

The Need: Healthcare Is Becoming a Data-Heavy Industry

Hospitals and medical systems now generate enormous amounts of data:

- Electronic health records
- Symptom logs
- Lab reports
- Wearable device streams

This data ***can*** help identify disease risk early.
But only if we can process it efficiently and consistently.

Traditional tools like Pandas or Excel cannot handle this growth.
Healthcare analytics needs a **scalable, distributed system**.

The Problem: High-Dimensional Data That Doesn’t Fit Traditional Tools

We worked with a healthcare dataset that already shows this challenge:

- 4920 patient samples + 1230 test samples
- 41 symptoms per patient → *wide* dataset
- Multi-disease classification
- Strict need for consistency and reliability

Even though this dataset is small, real medical datasets are much larger and cannot be processed on a single machine.

So the real question becomes:
How do we design a solution that can scale to real clinical workloads?

The Data

```
Train rows: 4920 columns: 134
Test rows:  42 columns: 133
```

	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	...	scurring	skin_peeling
0	1	1	1	0	0	0	0	0	0	0	...	0	0
1	0	1	1	0	0	0	0	0	0	0	...	0	0
2	1	0	1	0	0	0	0	0	0	0	...	0	0
3	1	1	0	0	0	0	0	0	0	0	...	0	0
4	1	1	1	0	0	0	0	0	0	0	...	0	0

5 rows × 134 columns

silver_like_dusting	small_dents_in_nails	inflammatory_nails	blister	red_sore_around_nose	yellow_crust_ooze	prognosis	_c133
0	0	0	0	0	0	Fungal infection	None
0	0	0	0	0	0	Fungal infection	None
0	0	0	0	0	0	Fungal infection	None
0	0	0	0	0	0	Fungal infection	None
0	0	0	0	0	0	Fungal infection	None

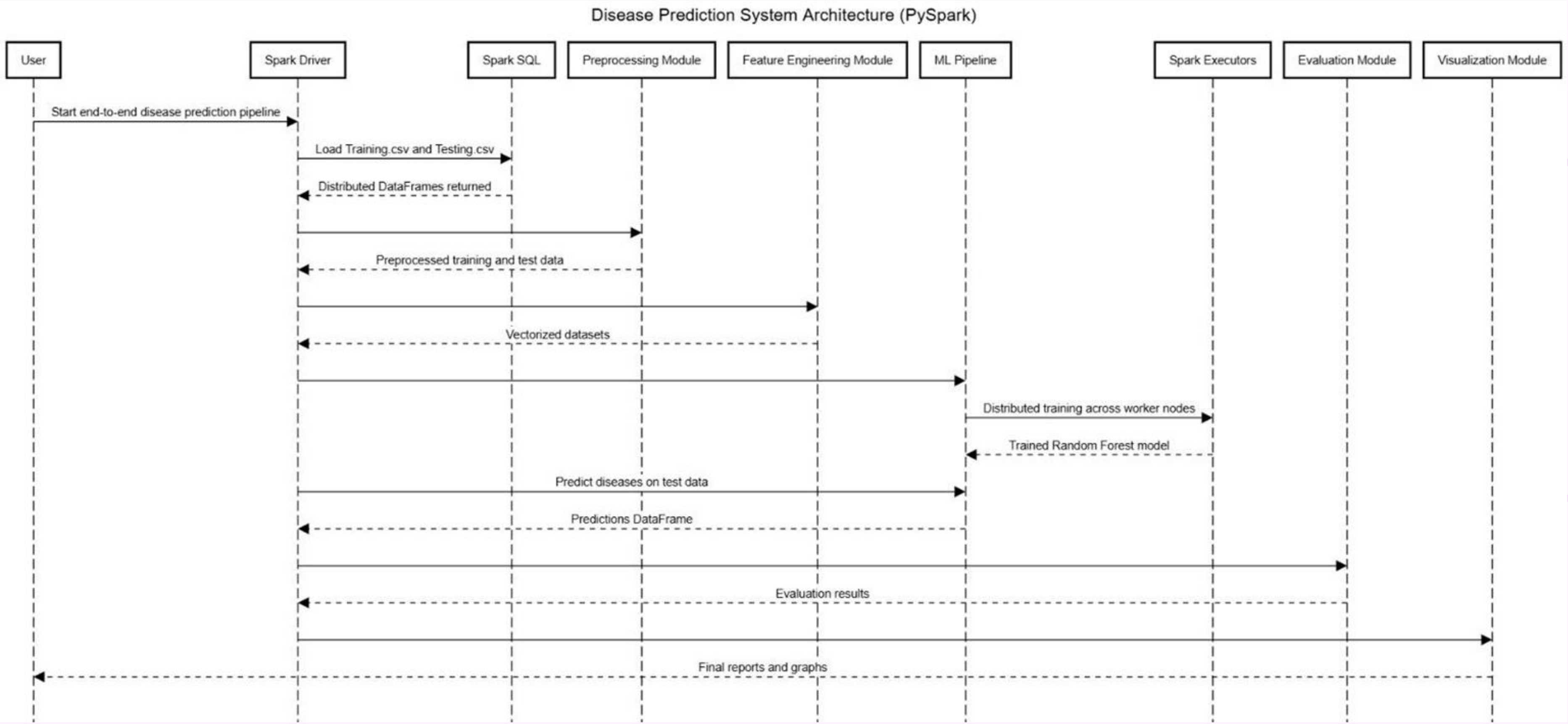
Why Spark?

Spark addresses the bottlenecks that traditional tools face:

- Distributed processing across multiple nodes
- In-memory computation for speed
- Fault tolerance through RDD lineage
- Optimized SQL engine (Catalyst)
- MLlib for distributed machine learning

Instead of struggling with RAM limits, Spark **divides work across a cluster**.
Perfect for healthcare-scale datasets.

Our Approach: Build a Big-Data Pipeline for Disease Prediction



How We Prepared the Data (Distributed Preprocessing)

We cleaned and prepared the dataset using distributed DataFrame transformations:

- Normalized inconsistent column names
- Converted 41 symptom features into numeric format
- Applied label encoding (StringIndexer)
- Created vectorized feature columns

Each transformation ran **in parallel**, not sequentially.
This demonstrates how Spark handles even wide, complex datasets.

Distributed EDA: Understanding the Data at Scale

Instead of analyzing symptoms on a single machine, Spark allowed us to compute:

- Symptom frequency
- Disease distribution
- Co-occurrence patterns

These operations used Spark SQL and DataFrames, meaning they could scale to millions of records with very little change.

The key insight: **EDA becomes scalable, repeatable, and efficient.**

The Solution in Action: Spark MLlib Pipeline & DEMO

VectorAssembler packs 41 features in parallel

RandomForestClassifier: trains multiple trees simultaneously

CrossValidator: distributes evaluation across executors

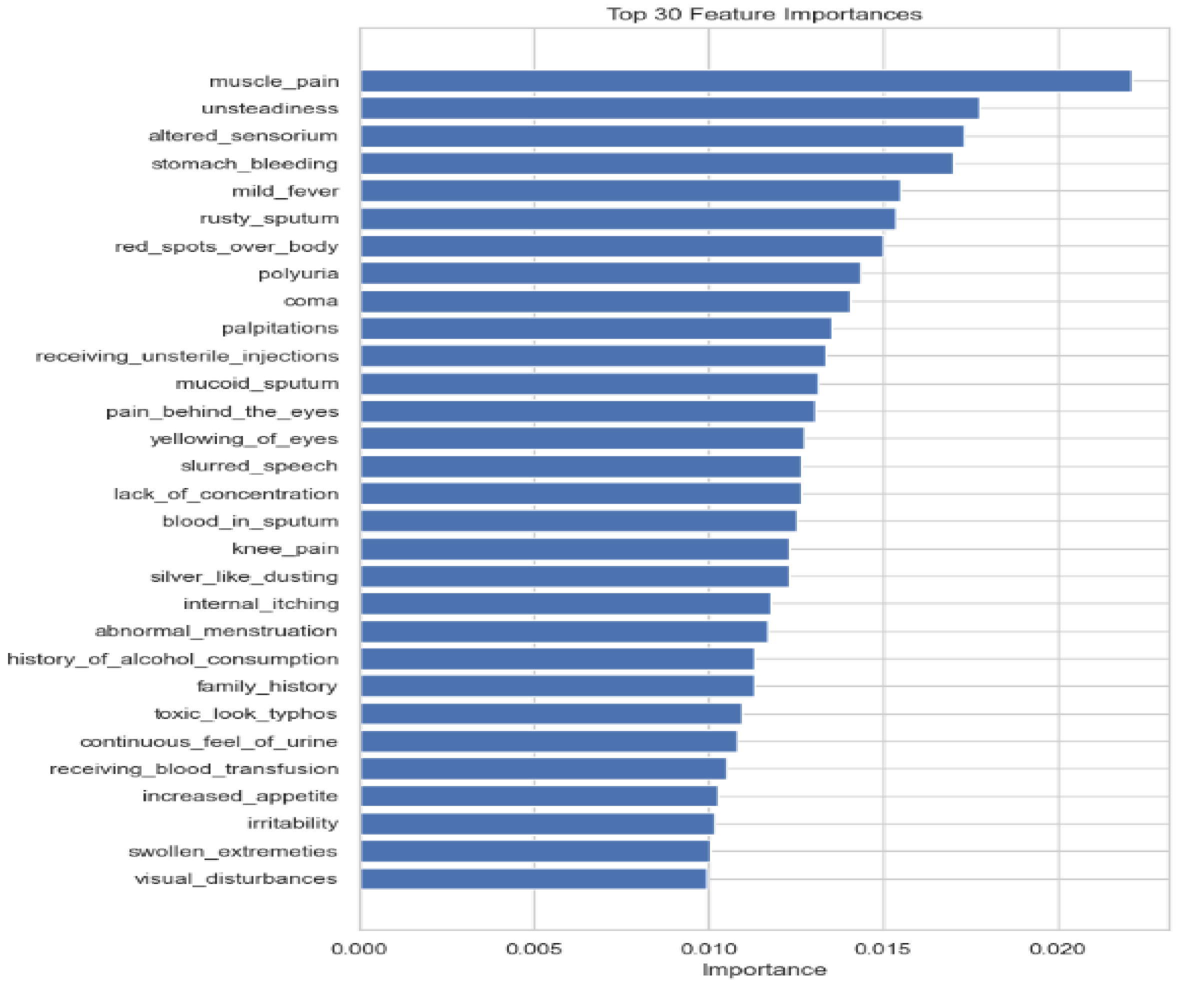
The pipeline is not only for prediction—
it demonstrates **how Spark enables large-scale analytical workflows.**

Results and What They Tell Us

Even on this dataset, the distributed workflow was effective:

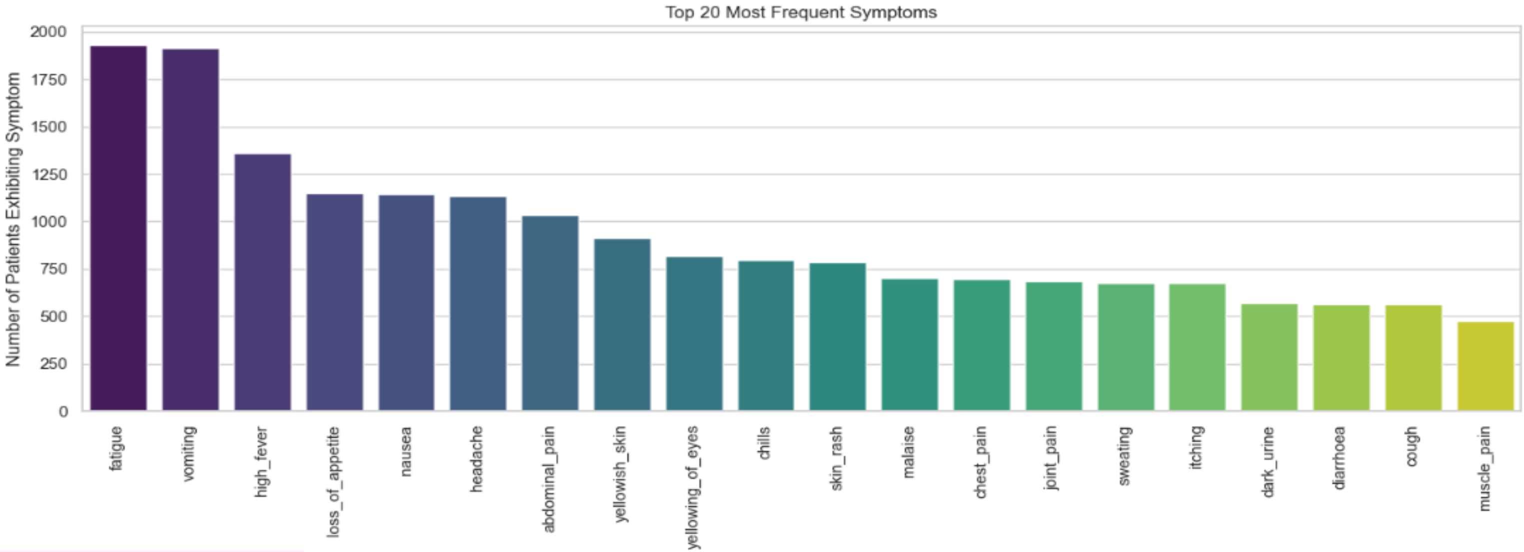
- Accuracy: **98%**
- Precision, recall, and F1 all above **0.98**

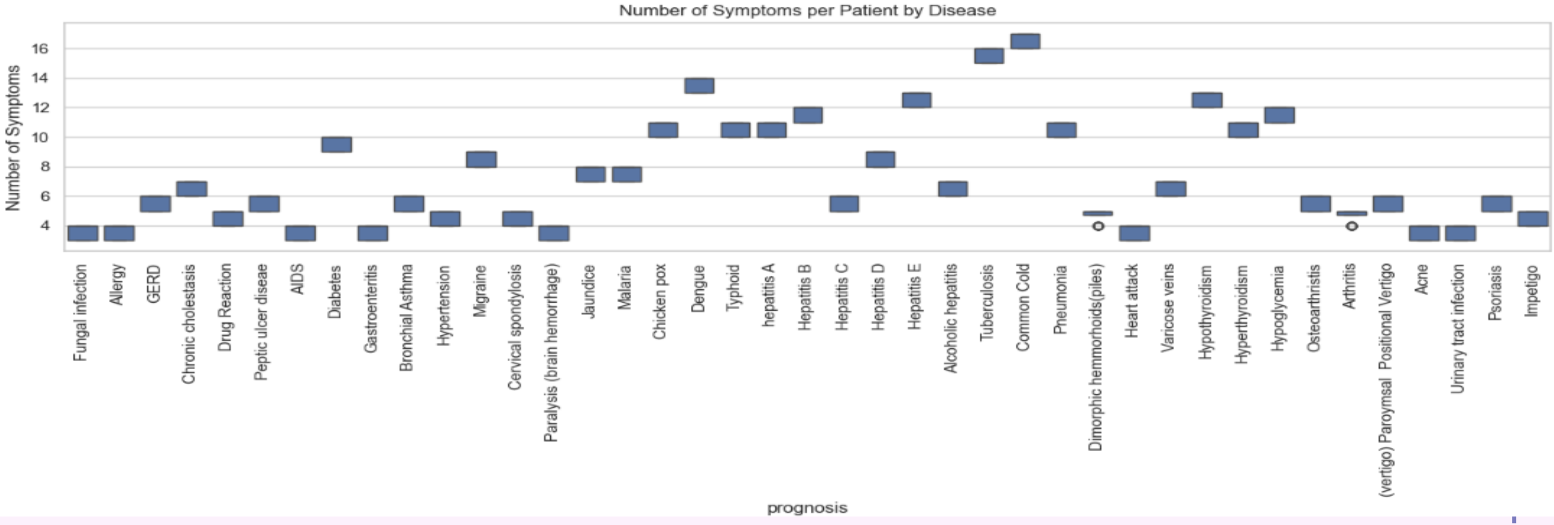
But the real point is not the number.
It is that **the pipeline works, and it scales.**
If tomorrow we receive 10 million patient records, Spark can run the same workflow with minimal changes.



[19]:

	feature	importance
0	muscle_pain	0.022095
1	unsteadiness	0.017738
2	altered_sensorium	0.017337
3	stomach_bleeding	0.017010
4	mild_fever	0.015493
5	rusty_sputum	0.015344
6	red_spots_over_body	0.015019
7	polyuria	0.014352
8	coma	0.014048
9	palpitations	0.013516
10	receiving_unsterile_injections	0.013335
11	mucoid_sputum	0.013122
12	pain_behind_the_eyes	0.013044
13	yellowing_of_eyes	0.012763
14	slurred_speech	0.012657
15	lack_of_concentration	0.012636
16	blood_in_sputum	0.012531
17	knee_pain	0.012307
18	silver_like_dusting	0.012295
19	internal_itching	0.011768
20	abnormal_menstruation	0.011710
21	history_of_alcohol_consumption	0.011312
22	family_history	0.011285
23	toxic_look_typhos	0.010966
24	continuous_feel_of_urine	0.010821
25	receiving_blood_transfusion	0.010509
26	increased_appetite	0.010251
27	irritability	0.010179
28	swollen_extremeties	0.010043
29	visual_disturbances	0.009971


Top 20 Most Frequent Symptoms
Number of Patients Exhibiting Symptom
2000
1750
1500
1250
1000
750
500
250
0
fatigue
vomiting
high_fever
loss_of_appetite
nausea
headache
abdominal_pain
yellowish_skin
yellowing_of_eyes
chills
skin_rash
malaise
chest_pain
joint_pain
sweating
itching
dark_urine
diarrhoea
cough
muscle_pain


Number of Symptoms per Patient by Disease
Number of Symptoms
16
14
12
10
8
6
4
Fungal infection
Allergy
GERD
Chronic cholestasis
Drug Reaction
Peptic ulcer diseae
AIDS
Diabetes
Gastroenteritis
Bronchial Asthma
Hypertension
Migraine
Cervical spondylosis
Paralysis (brain hemorrhage)
Jaundice
Malaria
Chicken pox
Dengue
Typhoid
hepatitis A
Hepatitis B
Hepatitis C
Hepatitis D
Hepatitis E
Alcoholic hepatitis
Tuberculosis
Common Cold
Pneumonia
Dimorphic hemmorhoids(piles)
Heart attack
Varicose veins
Hypothyroidism
Hyperthyroidism
Hypoglycemia
Osteoarthristis
Arthritis
(vertigo) Paroymsal Positional Vertigo
Acne
Urinary tract infection
Psoriasis
Impetigo
prognosis

Challenges, Lessons, and Big-Data Impact

Challenges we faced

- Inconsistent column names across CSVs
- Wide, 41-feature dataset
- Label mismatches between training/testing
- Spark's limited visualization environment

Big-Data Impact

- Pipeline demonstrates readiness for large, real clinical datasets
- Spark can support early disease prediction at scale
- Provides foundation for hospital-level analytics systems

Conclusion & Future Directions

Conclusion

Our project shows that:

- Healthcare needs scalable analytical pipelines
- Spark provides distributed, fault-tolerant infrastructure
- A well-designed data pipeline is essential for real-world disease prediction
- The focus is not only on prediction—but on **scalable, reliable data processing**

Future Work

- Move to HDFS or cloud object storage
- Integrate streaming data (Spark Structured Streaming)
- Scale to millions of patient records
- Add more advanced distributed models (GBTs, neural networks)
- Deploy as a real-time decision-support system for hospitals

Thank you!