Big Data Analytics on NYC Taxi

A four-part pipeline: 3.4B rows → Dask ETL → PCA → GAM → XGBoost

---

Emily Chen

April 17, 2026

M.S. Data Science, UC San Diego
DSC 291 Big Data Analytics

Outline

Project Overview

What was built

A four-part end-to-end pipeline on the public NYC TLC corpus (≈**3.4 billion** trip rows, 2009–2024).

- **Part 1 — ETL:** Dask-on-EC2, S3-native, 3.4B raw → 1.51M-row wide table.
- **Part 2 — PCA & stability:** 24-D hourly fingerprint, bootstrap stability, anomaly extension.
- **Part 3 — GAM:** Fare prediction with partial dependence + bootstrap CI + location features.
- **Part 4 — XGBoost:** Yellow-vs-Green classifier with PCA features and date/location extensions.

Headline outcomes
Acc 0.62 → **0.95**, **F1 0.62** → **0.94** on classification; **RMSE \$2.97** → **\$2.71** on fare prediction; **99.99 %** PCA subspace stability under 100-replicate bootstrap.

Part 1 — Distributed ETL

Part 1 — Distributed ETL Pipeline

Challenge. 16 years of heterogeneous schemas: `tpep_*`, `lpep_*`, raw lat/lon (pre-2017), 4 taxi types, ∼3.4B rows.

Solution.

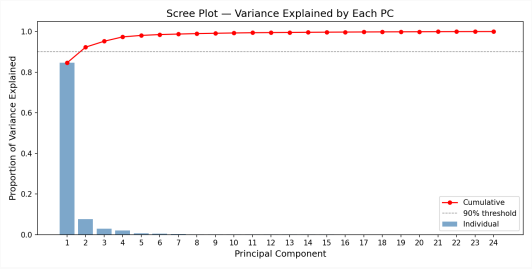
- Column-priority schema detector unifies datetime + location.
- Spatial join on `taxi_zones.shp` for legacy lat/lon (geopandas + STRtree).
- Dask LocalCluster on `r8i.4xlarge`: 6 workers × 16 GB, `processes=False` for s3fs picklability.
- Quality filter: drop < 50 rides per (date, location, taxi_type).

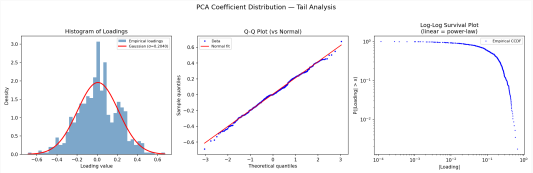
Metric	Value
Input rows	3,410,052,578
Output rows	1,513,742
Discarded	5.38 %
Wall-clock	9 min 37 s
Peak RSS	10.85 GB

A single-instance run that production-grade pipelines in industry would typically take a small Spark cluster to do.

Part 2 — PCA & Stability

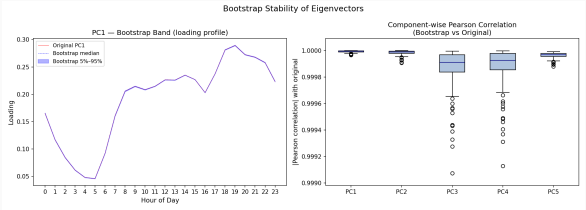
Part 2 — PCA on the 24-D Hourly Fingerprint





- Computed entirely with Dask + NumPy (`numpy.linalg.eigh`); no scikit-learn.
- Tail of 576 loadings is *light* (Gaussian-like): power-law-vs-lognormal $R = -32.8$, $p = 5.4 \times 10^{-7}$.
- PC1 = total daily volume; PC2 = AM-vs-PM peak shape.

Part 2 — Bootstrap Stability of Top-5 PCs

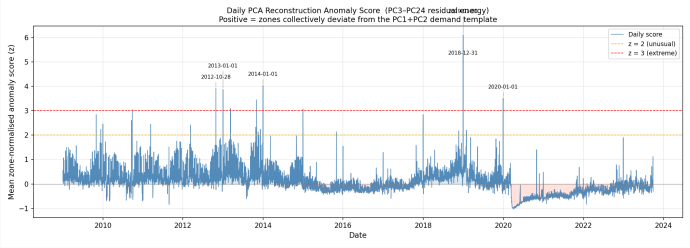


$B = 100$ **bootstrap resamples,** $K = 5$ **tracked PCs.**

- Subspace affinity: $4.99992 \pm 4.07 \times 10^{-5}$ (max 5).
- Procrustes distance: 0.00840 ± 0.00219.
- PC-wise correlation: ≥ 0.9998 for PC1–PC5.

Why it matters. Justifies using PC1–PC5 as input features in Part 4, since the basis itself is statistically indistinguishable across resamples.

Part 2 Extension — Anomaly Detection



Method. Residual energy in PC3–PC24, z-scored within each (*taxi_type*, *pickup_place*) group.

Result. 1.51M zone-day rows scored across 5,386 dates and 898 zones; 21,184 zone-days exceed $|z| > 3$.

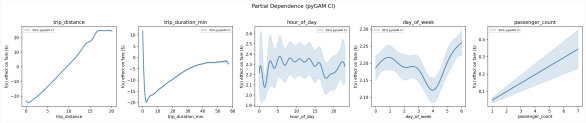
Top anomalous date: 2019-01-01 (mean $z = 6.11$). Second: 2018-12-31 — New Year holidays surface as the dominant outliers.

Part 3 — GAM Fare Prediction

Part 3 — GAM Fare Prediction

Model. $\hat{\text{fare}} = \alpha + s(\text{distance}) + s(\text{duration}) + s(\text{hour}) + s(\text{dow}) + l(\text{passenger})$

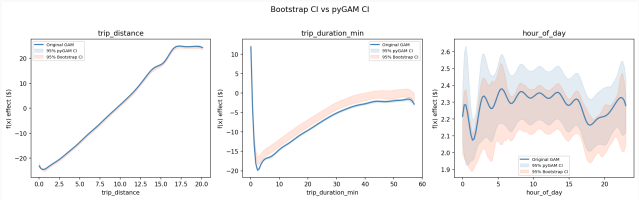
(`pyGAM LinearGAM, Gaussian, identity`)



Metric	Basic	Location	Δ
RMSE	\$2.97	**\$2.71**	−8.6%
MAE	\$0.91	**\$0.88**	−2.9%
R^2	0.9635	**0.9682**	+0.0047

Location-augmented model adds borough (pickup, dropoff) + haversine straight-line distance. **EWR** (Newark Airport) is the strongest single signal.

Part 3 Extra Credit — Bootstrap CI vs. pyGAM CI

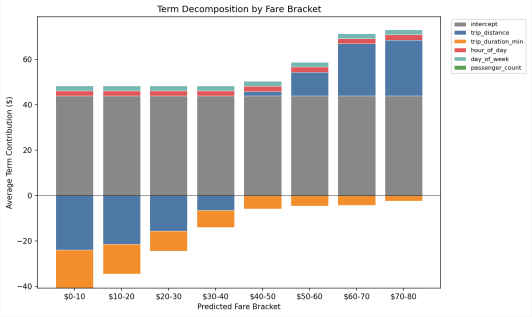


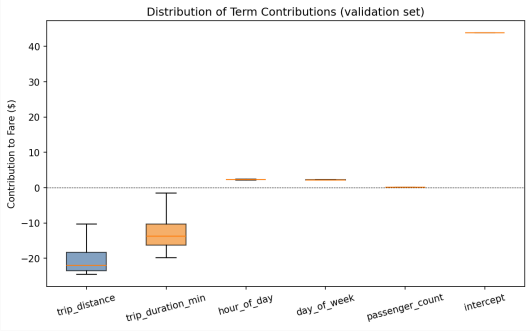
$B = 100$ parallel bootstrap resamples (7-core `joblib`).

Feature	Agreement
trip_distance	Full agreement
trip_duration	Mild divergence at extremes
hour_of_day	**pyGAM underestimates** at 0–5 am, 10–11 pm

Lesson: pyGAM's analytical CI is honest only where data is dense. Bootstrap is the safer measure for sparse regions.

Part 3 EC — Fare Decomposition

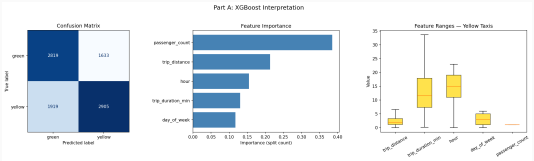




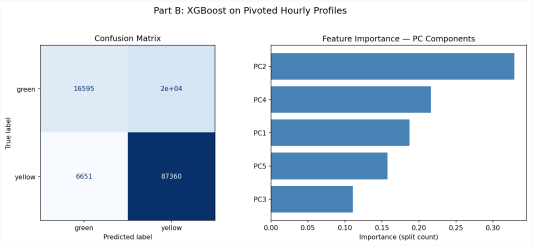
- **distance and duration** are the only terms with meaningful per-trip variance.
- **hour, day-of-week, passenger_count** are near-constant ($2–$2.5).
- Intercept of $43.87 is a parameterisation artefact of pyGAM's un-centred B-splines — predictions remain correct.

Part 4 — XGBoost Classification

Part 4 — XGBoost Yellow-vs-Green Classification



Part A: raw trip features

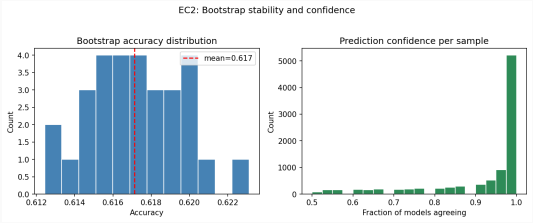


Part B: PC1–PC5 from hourly profile

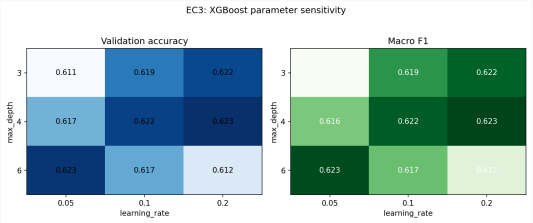
	Part A (raw)	Part B (PCA)	+ date/loc features
Accuracy	0.6171	0.7929	**0.9491**
Macro F1	0.62	0.71	**0.94**
Green recall	—	0.45	**0.88**

Cheap structured features (year, month, day-of-week, label-encoded zone) on top of PCA close the class-imbalance gap.

Part 4 Extra Credit — Stability & Sensitivity



EC2: 30 bootstrap replicates; accuracy 0.6171 ± 0.0024; 95 % CI $[0.6129, 0.6213]$; mean prediction confidence 0.9187.



EC3: 3×3 grid over `max_depth` $\times$ `lr`; best 0.6230 at (d=4, lr=0.2); deeper + faster overfits.

Closing

Quantified Impact

- 3.4B trip rows $\rightarrow$ 1.51M-row analytical wide table in 9 min 37 s on a single `r8i.4xlarge`.
- Top-5 PCA basis stable at ≥ 0.9999 correlation across 100 bootstraps; light tail ($R = -32.8$ vs. lognormal).
- XGBoost yellow-vs-green: **accuracy 0.62** $\rightarrow$ **0.95**, **F1 0.62** $\rightarrow$ **0.94**, green recall **0.45** $\rightarrow$ **0.88**, 95 % CI ± 0.004.
- GAM fare model: RMSE **\$2.97** $\rightarrow$ **\$2.71** (8.6 % better); exposed where pyGAM's analytical CI underestimates true uncertainty.

Infrastructure-and-statistics, end to end.

S3 parquet $\rightarrow$ Dask ETL $\rightarrow$ PCA features $\rightarrow$ XGBoost / GAM $\rightarrow$ bootstrap CI $\rightarrow$ shipped predictions.

Thank you.

Questions?