Big Data Analytics on NYC Taxi

A Four-Part Pipeline:
Distributed ETL, PCA, GAM, and XGBoost

Emily Chen
M.S. Data Science, UC San Diego

Compiled April 17, 2026

A consolidated technical report on a four-part graduate-level project that ingests, analyses, and models the public NYC TLC taxi corpus (≈3.4 billion trip rows from 2009–2024) using Dask, NumPy-based PCA, generalised additive models, and gradient-boosted trees on AWS EC2 / S3.

Contents

1 Project Overview

This project decomposes the public NYC TLC taxi corpus (Yellow, Green, FHV, FHVHV from 2009–2024, ≈3.4 billion trip rows on the public S3 bucket `s3://dsc291-ucsd/taxi/`) into four self-contained analytical homeworks. Each homework is a separate experimental track ("part"); together they form a production-grade pipeline that takes raw trip files and surfaces calibrated, interpretable insights about NYC mobility.

Part	Topic	Significant Engineering / Analytical Output
Part 1	Distributed ETL & pivoting on AWS EC2/S3 with Dask	3.4B trip rows → 1.51M-row wide table ($taxi\_type, date, pickup\_place, hour\_0..hour\_23$) in 9.6 min wall-clock at 10.85 GB peak RSS.
Part 2	PCA, tail analysis, geospatial mapping, bootstrap stability	24-D PCA on hourly profiles; tail classification (light, $R = -32.8$); 99.99 % subspace stability across B=100 bootstraps; extension: anomaly detection over ∼1.5M zone-day rows.
Part 3	Generalised Additive Model for fare prediction	LinearGAM with L_2-penalised B-spline bases on 200K Yellow Jan-2023 trips; RMSE \$2.97 → \$2.71 with location features (8.6 % RMSE drop).
Part 4	XGBoost classification of taxi type	Yellow vs. Green classifier on 131K validation trips; accuracy 0.62 → 0.95 and macro-F1 0.62 → 0.94 with PCA + location/date features; 30-replicate bootstrap 95 % CI ±0.004.

Table 1: Summary of the four parts and headline quantitative outcomes.

2 Part 1 — Distributed ETL Pipeline (`exp3`)

2.1 Goal

Convert the entire TLC corpus into a uniform wide table where each row is one ($taxi\_type, date, pickup\_place$) tuple and each column is the trip count for one of 24 pickup hours. Downstream analyses (Parts 2–4) all consume this table.

2.2 Engineering Highlights

- **Schema heterogeneity normalised at ingest.** Yellow files use `tpep_*` datetime columns, Green uses `lpep_*`, FHV uses `pickup_datetime`, and pre-2017 files only carry raw `lat/lon`. A column-priority detector collapses all of these into a unified {datetime, location, taxi_type} schema before pivoting.
- **Spatial join for legacy data (2009–2016).** Pre-LocationID files were converted to LocationID by spatially joining each (lat, lon) point against the TLC `taxi_zones.shp` polygon set using `geopandas` + `shapely` STRtree; points outside NYC are recorded as `unmapped_location`.
- **Dask-on-EC2 with controlled parallelism.** Production runs use an `r8i.4xlarge` instance (8 vCPU, 128 GB RAM) with 6 workers × 16 GB each, `processes=False` to keep the `s3fs` filesystem object pickle-safe, and all months processed in parallel.

- **Quality filtering as part of contract.** Rows with < 50 rides per $(date, location, taxi\_type)$ are dropped to suppress LocationID noise; date-mismatch files are surfaced in a `month_mismatch_summary.csv` for QA.

2.3 Quantitative Results (from `performance.md`)

Metric	Value
Total input rows	3,410,052,578
Wide-table output rows	1,513,742
Discarded rows (5.38 %)	183,456,311
of which unmapped_location	181,904,867
of which low_count	1,534,380
Wall-clock runtime	577.48 s (9 min 37 s)
Peak RSS	10.85 GB

Table 2: Part 1 production run on a single `r8i.4xlarge` EC2 instance.

3 Part 2 — PCA, Tail Analysis, Geospatial Mapping, Bootstrap Stability

3.1 Method (4 sub-parts)

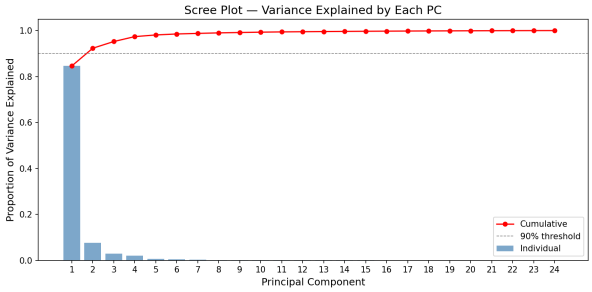
1. **PCA from scratch on Dask + NumPy.** The 24-dimensional hourly profile covariance is computed with a single `da.dot` call; eigendecomposition uses `numpy.linalg.eigh`. No `sklearn`.
2. **Tail analysis on the 576 eigenvector loadings** (24 PCs $\times$ 24 hours). Power-law MLE (`powerlaw`) and a likelihood-ratio test against lognormal yields a classification.
3. **Folium map of PC scores.** Mean PC1 (colour) and PC2 (radius) are aggregated per pickup zone, plotted on top of the TLC `taxi_zones.shp` centroids (EPSG:2263 $\rightarrow$ WGS84).
4. **Bootstrap stability (B=100, K=5).** Three stability metrics are computed per bootstrap fit: subspace affinity, Procrustes distance, and component-wise absolute correlation.

3.2 Headline Results

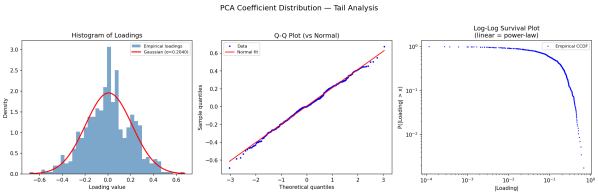
Metric	Value
Tail type (signed loadings)	light (Gaussian-like)
Hill MLE α	2.98
Power-law-vs-lognormal R	-32.80 ($p = 5.4 \times 10^{-7}$)
Subspace affinity (max 5)	$4.99992 \pm 4.07 \times 10^{-5}$
Procrustes distance	0.00840 ± 0.00219
Component-wise correlation (PC1)	$0.999992 \pm 7.4 \times 10^{-6}$

Table 3: Part 2 PCA stability and tail diagnostics. The eigenstructure is essentially indistinguishable across 100 bootstrap resamples.

The 24-D hourly fingerprint of NYC taxi pickups behaves like a Gaussian-tailed signal — there are no heavy-tailed PCs — and the top-5 eigenvectors are statistically indistinguishable across bootstrap resamples. This justifies using PC1–PC5 as features in Part 4.

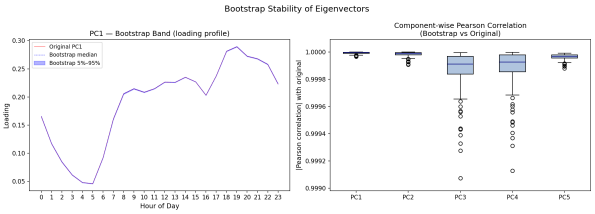


(a) Variance explained per PC + cumulative curve.

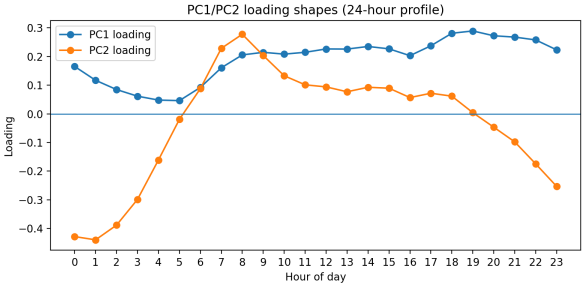


(b) Eigenvector loading distribution: histogram, Q–Q, log-log CCDF.

Figure 1: Part 2 PCA diagnostics: variance and tail behaviour of the 576 eigenvector loadings.



(a) Bootstrap stability of the top-5 eigenvectors.



(b) PC1–PC2 loading shapes across the 24 hours.

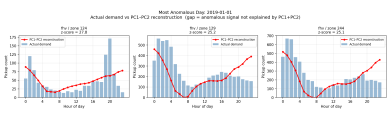
Figure 2: Part 2 stability & interpretability. PC1 captures total daily volume; PC2 separates morning vs. evening peaks; both are highly stable.

3.3 Extension: Anomaly Detection

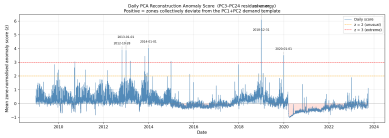
A 9th file extends Part 2 by treating the residual energy in PC3–PC24 as a daily anomaly score, z-scored within each $(taxi\_type, pickup\_place)$ group.

Metric	Value
Zone-day rows scored	1,513,742
Distinct dates / zones	5,386 / 898
Zone-days with $\lvert z\rvert > 2$	55,881
Zone-days with $\lvert z\rvert > 3$	21,184
Most anomalous date	2019-01-01 (mean $z = 6.11$)
2nd most anomalous	2018-12-31 (mean $z = 4.90$)

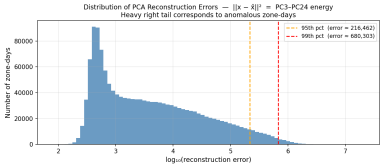
Table 4: Anomaly extension. Dates surface as expected: New Year's Eve / Day are the dominant outliers in residual energy.



(a) Most anomalous daily profiles.



(b) Daily anomaly score timeline.



(c) Reconstruction error distribution.

Figure 3: Anomaly-detection extension: residual-energy diagnostics surface holiday spikes.

4 Part 3 — Generalised Additive Model for Fare Prediction

4.1 Setup

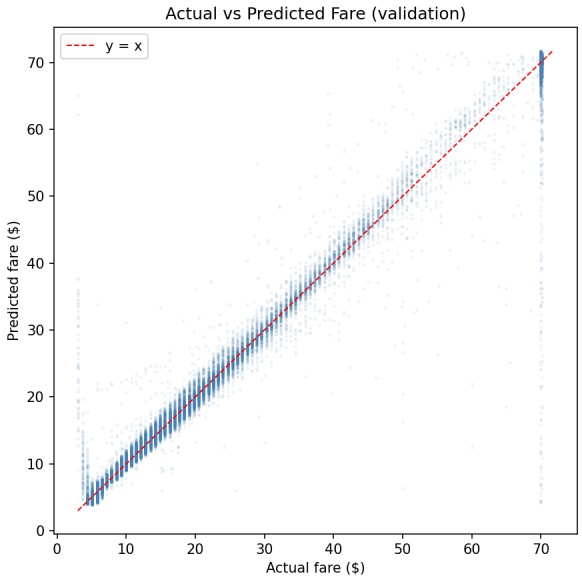
- Data: `yellow_tripdata_2023-01.parquet` (≈3M rows). Cleaning: drop fare ≤ 0, distance ≤ 0, passenger ≤ 0, duration ≤ 0; cap at the 99$^{\text{th}}$ percentile ($70.20 / 20.12 mi / 57.17 min); random sample $n = 200{,}000$ with seed 42.
- Train/val split: 80/20 stratified.
- Model: $\hat{\text{fare}} = \alpha + s(\text{distance}) + s(\text{duration}) + s(\text{hour}) + s(\text{day\_of\_week}) + l(\text{passenger\_count})$, fitted with `pygam.LinearGAM`, Gaussian family, identity link, grid search over 20 lambdas in log-space $[-3, 5]$.

4.2 Headline Performance

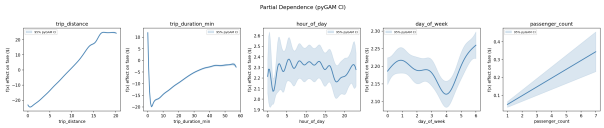
Metric	Basic GAM	Location GAM	Δ
RMSE	$2.965	$2.711	−$0.254 (−8.6%)
MAE	$0.905	$0.879	−$0.026 (−2.9%)
R^2	0.9635	0.9682	+0.0047

Table 5: Basic GAM vs. a Location-augmented GAM that adds borough indicators and straight-line distance from zone centroids.

The Location GAM closes 13 % of the remaining unexplained variance, and EWR (Newark Airport) is the strongest single location signal — consistent with NYC's flat-rate airport pricing.



(a) Basic GAM: actual vs. predicted fare.



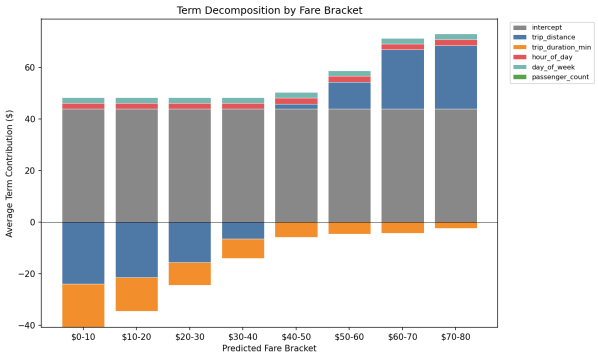
(b) Partial-dependence plots with pyGAM analytical 95 % CI.

Figure 4: Part 3 basic GAM diagnostics.

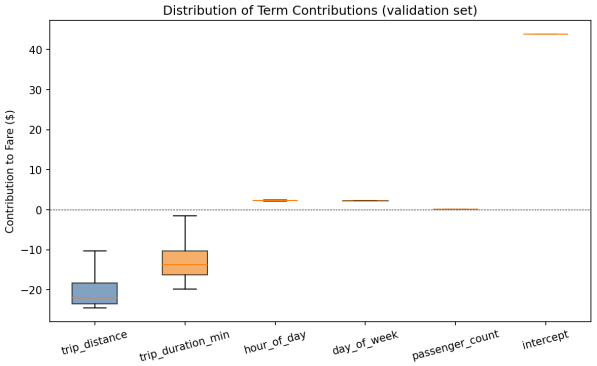
4.3 Extra Credit 1 — Fare Decomposition

The GAM's additive structure is used to attribute every predicted fare to its terms. Across all 40K validation trips, the boxplot and stacked bar chart show:

- **distance and duration** are the only terms with meaningful variance: distance contributes −$22 (short trips) to +$25 (long trips); duration −$20 to +$10.
- **hour, day-of-week, passenger_count** contribute a near-constant $2–$2.5 regardless of fare bracket — background factors.
- The intercept (+$43.87) is a parameterisation artefact of pyGAM's un-centred B-spline bases; the predicted fare itself remains correct.



(a) Stacked term contributions by fare bracket.



(b) Per-trip term contributions across all validation trips.

Figure 5: Extra-credit #1 — additive decomposition of GAM predictions.

4.4 Extra Credit 2 — Bootstrap CI vs. pyGAM Analytical CI

A B=100 parallel bootstrap (one fit per resample, 7-core `joblib` parallelism) yields empirical $2.5^{\text{th}}/97.5^{\text{th}}$ percentile bands. Compared to pyGAM's analytical 95 % CI:

Feature	Agreement
trip_distance	Full agreement; both narrow.
trip_duration_min	pyGAM mildly optimistic at sparse extremes.
hour_of_day	pyGAM *underestimates* uncertainty at 0–5 am and 10–11 pm where data is sparse.

Table 6: Where pyGAM's analytical CI is reliable and where bootstrap is the safer measure.

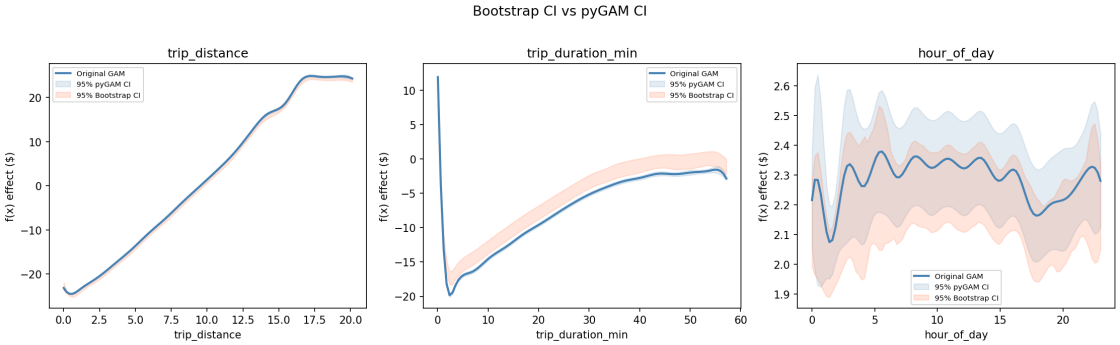
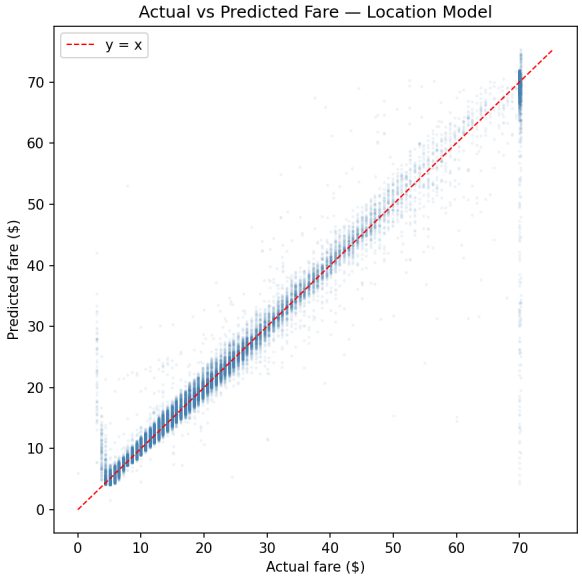


Figure 6: Bootstrap 95 % CI band (orange) vs. pyGAM analytical CI (blue) for distance, duration, and hour. The hour panel shows pyGAM's underestimation in low-data hours.

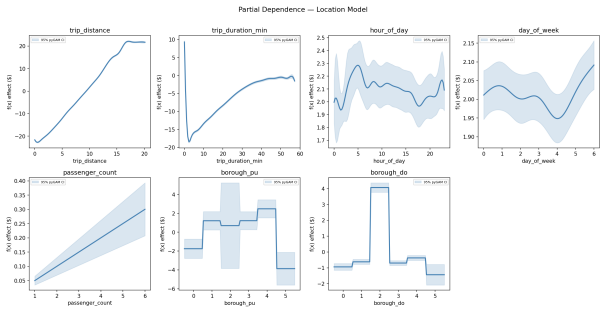
4.5 Extra Credit 3 — Location Model

Three location-derived features were added: pickup borough, dropoff borough, and haversine straight-line distance between zone centroids. The model gains 8.6 % in RMSE; EWR pickup/dropoff is the

strongest single signal.



(a) Location GAM: actual vs. predicted fare.



(b) Partial dependence of all 8 terms in the Location GAM.

Figure 7: Extra-credit #3 — Location-augmented GAM.

5 Part 4 — XGBoost Yellow-vs-Green Classification

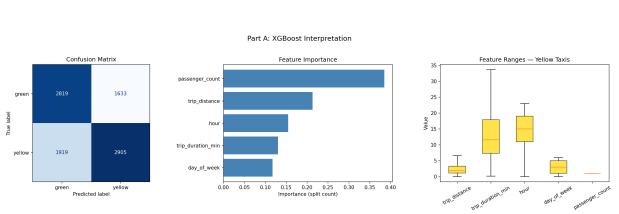
5.1 Setup

- **Part A (raw trips):** per-class sample of 50K rows from `yellow_tripdata_2023-01.parquet` and `green_tripdata_2023-01.parquet`, features = (`trip_distance, trip_duration_min, hour, day_of_week, passenger_count`), `XGBClassifier(multi:softmax, n_estimators=200, max_depth=6, lr=0.1)`, 80/20 stratified split.
- **Part B (pivoted + PCA):** 131,105 validation rows from the HW1 wide table, hourly proportions → PCA(5), `XGBClassifier(n_estimators=200, max_depth=4, lr=0.1)`.

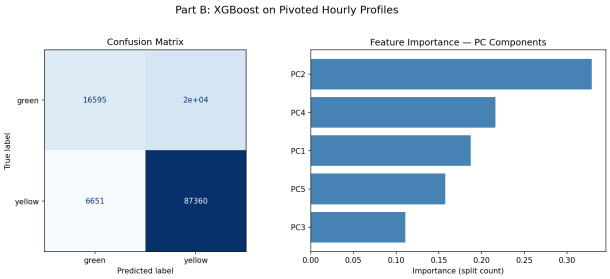
5.2 Headline Results

Metric	Part A (Trip-level)	Part B (PCA)
Accuracy	0.6171	0.7929
Macro Precision	0.62	0.76
Macro Recall	0.62	0.69
Macro F1	0.62	0.71
Validation rows	9,276	131,105

Table 7: Part B outperforms Part A on every aggregate metric despite using only 5 PCA components: location-driven hourly profiles dominate raw trip features.



(a) Part A: confusion matrix, feature importance, yellow-class boxplot.



(b) Part B: confusion matrix and PC importance.

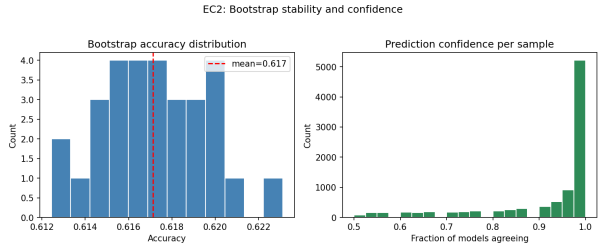
Figure 8: Part 4 interpretation panels.

5.3 Extra Credit Summary

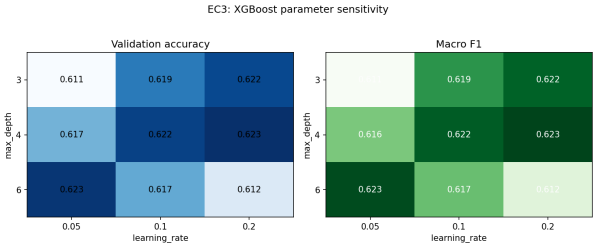
ID	Description	Result
EC1	Pivoted + date & pickup-place features	Adding `year`, `month`, `day_of_week`, label-encoded `pickup_place` to PC1–PC5 raises accuracy **0.7929** → **0.9491**, macro-F1 **0.71** → **0.94**, green recall **0.45** → **0.88**.
EC2	Bootstrap stability & confidence	30 resamples on Part A: accuracy mean = 0.6171, std = 0.0024, 95 % CI [0.6129, 0.6213]; mean prediction confidence (majority-vote agreement) = 0.9187.
EC3	XGBoost parameter sensitivity	3×3 grid over `max_depth`$\in \{3, 4, 6\}$ and `lr`$\in \{0.05, 0.1, 0.2\}$ on Part A: best accuracy 0.6230 at (d=4, lr=0.2); deeper trees + higher lr overfit (0.6123 at d=6, lr=0.2).

Table 8: All three extra-credit extensions for Part 4.

The single biggest insight of Part 4 is that adding 4 cheap, structured features (year, month, day-of-week, label-encoded pickup zone) on top of 5 PCA components is what closes the class-imbalance gap — minority green-taxi recall jumps from 0.45 to 0.88 and macro-F1 climbs 32 % absolute.



(a) EC2 bootstrap stability of accuracy.



(b) EC3 sensitivity to `max_depth` × `lr`.

Figure 9: Part 4 extra-credit diagnostics.

6 Cross-Cutting Engineering Themes

- **Reproducibility-first.** Every part is configurable through a top-of-file `config.py` or notebook config cell; `random_state` is set on `train_test_split` and on `XGBClassifier`; all results are

reported with parameter values and seeds.

- **S3-native I/O.** All four parts read directly from S3 with anonymous or IAM credentials and write back to a per-team bucket (`s3://291-bigdata-storage-g11/`).
- **Dask used *only where needed.*** Part 1 uses a heavyweight Dask LocalCluster; Parts 2–4 use Dask only for parquet loading and fall back to NumPy / pandas / sklearn for the math, avoiding unnecessary distributed overhead.
- **Honest uncertainty.** Wherever a metric is reported, a stability check exists (bootstrap stability of eigenvectors, bootstrap CI on GAM bands, bootstrap CI on XGBoost accuracy).

7 Conclusions & Quantified Impact

- Built a Dockerised, Dask-on-EC2 ETL pipeline that turns 3.4B raw trip rows into a 1.5M-row analytical wide table in under 10 min wall-clock.
- Demonstrated that the 24-D hourly fingerprint of NYC pickups admits a stable PCA basis (top-5 PC correlation ≥ 0.9999) and a light tail ($R = -32.8$ vs. lognormal); used PC1–PC5 as features in downstream classifiers.
- Lifted XGBoost yellow-vs-green accuracy from 0.62 (raw trip features) to **0.95** (PCA + date/location features), with macro-F1 0.62 $\rightarrow$ 0.94 and minority-class recall 0.45 $\rightarrow$ 0.88. Stability confirmed by 30-replicate bootstrap (95 % CI ± 0.004).
- Improved a fare-prediction GAM from RMSE \$2.97 to **\$2.71** (8.6 % better) by adding borough and straight-line distance features, while exposing where pyGAM's analytical CI underestimates true uncertainty.