

OUTLIER DETECTION IN ELECTION DATA USING GEOSPATIAL ANALYSIS

Ensuring Election Integrity through Data-
Driven Insights

DATE
October, 2025

PREPARED BY:
Aidelokhi Abolagba

INTRODUCTION

- Nigeria's elections have faced integrity challenges due to potential vote manipulation.
- Detecting irregularities requires analyzing spatial voting patterns and identifying wards or polling units (PUs) that differ sharply from their geographic neighbors.
- This project applies Outlier Detection and Geospatial Analysis to highlight anomalies in election results.

Dataset Overview

This Dataset includes polling unit-level results across Nigerian, specifically Edo state LGAs and wards.

Each record represents one Polling Unit (PU) with both demographic and result-based data.

Key columns analyzed:

- APC, LP, PDP, NNPP – vote counts for major parties
- Latitude, Longitude – location of each PU
- neighbors, neighbor_count – list of spatially close PUs
- Statistical summaries for each party (mean, median, std, MAD)
- Derived features: robust z-scores and absolute robust z-scores

Tools Used : python via Jupyter Notebook

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	State	LGA	Ward	PU-Code	PU-Name	Accreditec	Register	Results_F	Transcript	Result_Shi	Result_Shi	Result_Shi	Result_Shi	Result_Shi	APC	LP	PDP	NNPP	Results_Fi	CO_region	Latitude	Longitude	country		
2	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	236	788	TRUE	-1	FALSE	FALSE	FALSE	FALSE	UNKNOWNI	0	0	0	0	https://do Nigeria, E	7.291651	6.102347	Nigeria			
3	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	213	597	TRUE	-1	FALSE	FALSE	FALSE	FALSE	UNKNOWNI	118	55	32	1	https://do Nigeria, E	7.291651	6.102347	Nigeria			
4	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	199	520	TRUE	-1	FALSE	TRUE	FALSE	FALSE	UNKNOWNI	138	32	28	0	https://do Nigeria, E	7.291651	6.102347	Nigeria			
5	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	212	894	TRUE	-1	FALSE	FALSE	FALSE	FALSE	UNKNOWNI	77	77	46	0	https://do Nigeria, E	7.291651	6.102347	Nigeria			
6	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	189	642	TRUE	-1	FALSE	FALSE	FALSE	FALSE	UNKNOWNI	49	82	45	0	https://do Nigeria, E	7.291651	6.102347	Nigeria			
7	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	182	571	TRUE	-1	FALSE	FALSE	FALSE	FALSE	UNKNOWNI	84	47	41	0	https://do Nigeria, E	7.291651	6.102347	Nigeria			
8	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	239	807	TRUE	-1	FALSE	FALSE	TRUE	FALSE	UNKNOWNI	90	104	37	2	https://int Nigeria, E	7.291651	6.102347	Nigeria			
9	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	287	925	TRUE	-1	FALSE	FALSE	FALSE	FALSE	UNKNOWNI	90	81	112	1	https://do Nigeria, E	7.291651	6.102347	Nigeria			
10	EDO	AKOKO ED	IGARRA I	12-01-01-(UGBOGBC	336	930	TRUE	-1	FALSE	FALSE	FALSE	FALSE	UNKNOWNI	120	154	40	0	https://do Nigeria, E	7.291651	6.102347	Nigeria			

Dataset preview

Methodology overview

The analysis followed these main steps:

1. **Data Cleaning:** Removing missing or invalid records, conversion to numeric types for all vote and coordinate columns (`pd.to_numeric(..., errors="coerce")`), ensuring mathematical operations are valid.
2. **Neighbour Identification (Geospatial Analysis) :** To measure local voting behaviour, each polling unit was compared to nearby units located within 1 kilometre. Neighbouring relationships were identified using a BallTree built with the Haversine distance metric, which correctly accounts for the curvature of the Earth.

$$\text{distance (radians)} = \text{radius_km} / \text{Earth radius km}$$

$$R_{\text{Earth}} = 6371.0088 \text{ km}$$

For each polling unit:

1. Coordinates were converted to radians.
2. The BallTree queried all other points within the 1 km radius.
3. Each unit's neighbour list excluded itself and recorded the total count as `neighbor_count`.
4. Units with fewer than three neighbours (`MIN_NEIGHBORS = 3`) were flagged as low-density areas and excluded from statistical outlier scoring to avoid unstable estimates

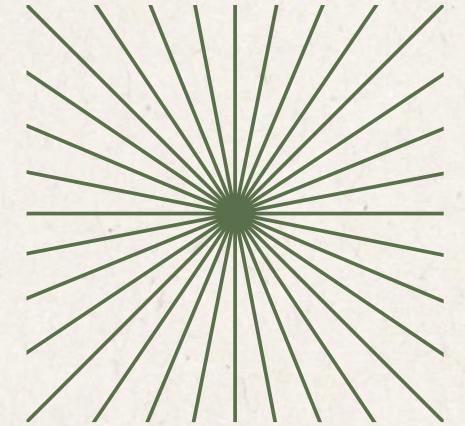


3. Outlier Score Calculation (Robust Z Score)

To detect voting irregularities, we compared each unit's vote totals against the local distribution of its neighbours.

For every party:

- **Neighbour Statistics** - For each unit, compute the median and Median Absolute Deviation (MAD) of its neighbours' vote
$$MAD = \text{median}(|x_i - \text{median}(x)|)$$



- **Scale Factor** - Convert MAD to an approximate standard deviation:

$$\sigma_{\text{robust}} = 1.4826 \times MAD$$

- **Robust Z-Score** - Quantify deviation of the unit's vote from its neighbourhood:

$$\sigma_{\text{robust}} = v_{\text{unit}} - \text{median}_{\{\text{neigh}\}} / \sigma_{\text{robust}}$$

- **Absolute Robust Z** - Take the absolute value, abs_{rz} , to measure magnitude of deviation regardless of direction.

A higher $|z|$ value indicates a greater deviation from the local voting pattern, suggesting potential anomalies such as data entry errors, mis-recorded votes, or external influences.

4. Ranking and Output

- For each party, units were sorted by descending abs_{rz} .
- A combined score $\text{max}_{\text{abs}_{\text{rz}}}$ captured the highest deviation among all parties for each unit.

The analysis produced:

- CSV file (`outlier_scores_full.csv`) containing all calculated metrics.
- Excel workbook (`outlier_ranked_by_party.xlsx`) with separate sheets per party ranked by their absolute robust Z-scores.

Code Snippets

```
import os
import math
import pandas as pd
import numpy as np
import geopandas as gpd
from shapely.geometry import Point
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages
from sklearn.neighbors import NearestNeighbors

INPUT_CSV = "cleaned_with_coordinates_final.csv"
OUT_DIR = "output_edostate"
RADIUS_M = 1000 # neighbour radius in meters
TOP_N = 3 # 

#Ensuring output directory
os.makedirs(OUT_DIR, exist_ok=True)
PLOTS_DIR = os.path.join(OUT_DIR, "plots")
os.makedirs(PLOTS_DIR, exist_ok=True)

def find_col_ignorecase(df, names):
    """
    Return the first column in df whose lower() matches any name.lower() in names.
    names can be list of candidate strings.
    """
    cols_lower = {c.lower(): c for c in df.columns}
    for name in names:
        if name and name.lower() in cols_lower:
            return cols_lower[name.lower()]
    return None

df = pd.read_csv(INPUT_CSV)
print("Loaded CSV with shape:", df.shape)
print("Columns:", list(df.columns)[:40])

# Detect essential columns (flexible)
pu_col = find_col_ignorecase(df, ["PU-Name", "PU_Name",
    "PU Code", "PU-Code", "polling_unit", "polling unit"])
ward_col = find_col_ignorecase(df, ["Ward", "ward"])
lga_col = find_col_ignorecase(df, ["LGA", "lga"])
lon_col = find_col_ignorecase(df, ["Longitude", "longitude",
    "lon", "long"])
lat_col = find_col_ignorecase(df, ["Latitude", "latitude", "lat"])
```

Return the first column in df whose lower() matches any name.lower() in names.
names can be list of candidate strings.

```
cols_lower = {c.lower(): c for c in df.columns}
for name in names:
    if name and name.lower() in cols_lower:
        return cols_lower[name.lower()]
    return None

df = pd.read_csv(INPUT_CSV)
print("Loaded CSV with shape:", df.shape)
print("Columns:", list(df.columns)[:40])

# Detect essential columns (flexible)
pu_col = find_col_ignorecase(df, ["PU-Name", "PU_Name",
    "PU Code", "PU-Code", "polling_unit", "polling unit"])
ward_col = find_col_ignorecase(df, ["Ward", "ward"])
lga_col = find_col_ignorecase(df, ["LGA", "lga"])
lon_col = find_col_ignorecase(df, ["Longitude", "longitude",
    "lon", "long"])
lat_col = find_col_ignorecase(df, ["Latitude", "latitude", "lat"])
```

```
if pu_col is None or lon_col is None or lat_col is None:
    raise SystemExit("Essential columns missing. Ensure CSV
contains polling unit name, latitude and longitude.")

# Normalize names
df = df.rename(columns={pu_col: "PU-Name"})
if ward_col:
    df = df.rename(columns={ward_col: "ward"})
if lga_col:
    df = df.rename(columns={lga_col: "LGA"})
df = df.rename(columns={lon_col: "Longitude", lat_col:
    "Latitude"})

print("Normalized columns present:", [c for c in ["PU-
Name", "ward", "LGA", "Longitude", "Latitude"] if c in
df.columns])
```

2

Detect party vote columns

```
# -----
# Prefer known party names first; fallback to numeric columns
# excluding metadata
candidate_parties =
["APC", "PDP", "LP", "YPP", "NNPP", "SDP", "APM", "ADC", "PRP",
    "PPA", "ZLP"]
party_cols = [c for c in df.columns if c.upper() in
candidate_parties]

if len(party_cols) == 0:
    metadata = {"PU-
Name", "ward", "LGA", "Longitude", "Latitude"}
    party_cols = [c for c in df.columns if c not in metadata and
pd.api.types.is_numeric_dtype(df[c])]

if len(party_cols) == 0:
    raise SystemExit("No party vote columns detected. Please
ensure vote counts are numeric columns in the CSV.")

print("Detected party columns:", party_cols)
```

#

Code Snippets 2

3) Prepare GeoDataFrame + projection

```
# -----
# drop missing coords
df = df.dropna(subset=
["Latitude","Longitude"]).reset_index(drop=True)

gdf = gpd.GeoDataFrame(df.copy(),
geometry=gpd.points_from_xy(df["Longitude"],
df["Latitude"]), crs="EPSG:4326")
# project to meters for distance computations
gdf_proj = gdf.to_crs(epsg=3857)

# build coordinate array for sklearn (x,y in meters)
coords = np.vstack([gdf_proj.geometry.x.values,
gdf_proj.geometry.y.values]).T

# 4) Neighbour detection using sklearn radius search

nbrs = NearestNeighbors(radius=RADIUS_M,
metric="euclidean", n_jobs=-1)
nbrs.fit(coords)
distances, indices = nbrs.radius_neighbors(coords,
radius=RADIUS_M)
```

#Build list of neighbour index lists (exclude the point itself)

```
neighbors_list = []
for i, idxs in enumerate(indices):
    neigh = [int(j) for j in idxs if j != i]
    neighbors_list.append(neigh)
```

```
gdf_proj["neighbours_idx"] = neighbors_list
gdf_proj["n_neighbours"] =
gdf_proj["neighbours_idx"].apply(len)
```

5) Per-party neighbour stats and z-scores

```
# We store: {party}_neigh_mean, {party}_neigh_std,
{party}_zscores, {party}_abs_zscore
for party in party_cols:
    neigh_means = []
    neigh_stds = []
    zscores = []
    abs_zscores = []
    for i, neigh in enumerate(gdf_proj["neighbours_idx"]):
        if len(neigh) == 0:
            mean = np.nan
            std = np.nan
            z = np.nan
        else:
```

else:

```
        vals = gdf_proj[party].iloc[neigh].values
        mean = float(np.mean(vals))
        std = float(np.std(vals, ddof=1)) if len(vals) > 1 else 0.0
        # z score with safe fallback
        if std == 0:
```

scaled difference to avoid division by zero

```
        z = (gdf_proj[party].iloc[i] - mean) / (mean + 1)
    else:
        z = (gdf_proj[party].iloc[i] - mean) / std
    neigh_means.append(mean)
    neigh_stds.append(std)
    zscores.append(z)
    abs_zscores.append(abs(z) if not pd.isna(z) else np.nan)
gdf_proj[f"{party}_neigh_mean"] = neigh_means
gdf_proj[f"{party}_neigh_std"] = neigh_stds
gdf_proj[f"{party}_zscores"] = zscores
gdf_proj[f"{party}_abs_zscore"] = abs_zscores
```

overall outlier magnitude: max absolute zscore across parties

```
abs_zscore_cols = [f"{p}_abs_zscore" for p in party_cols]
gdf_proj["max_abs_zscore"] =
gdf_proj[abs_zscore_cols].max(axis=1, skipna=True)
```

6) Sort and save results (CSV + Excel)

```
# -----
gdf_sorted = gdf_proj.sort_values("max_abs_zscore",
ascending=False).reset_index(drop=True)
gdf_out = gdf_sorted.to_crs(epsg=4326) # back to lat/lon for saving
```

Prepare results dataframe (select relevant columns)

```
base_cols = ["PU-Name", "ward", "LGA", "Longitude",
"Latitude", "n_neighbours", "max_abs_zscore"]
results_cols = [c for c in base_cols if c in gdf_out.columns]
results_df = gdf_out[results_cols].copy()
```

add per-party zscores and neighbour means

```
for p in party_cols:
    results_df[f"{p}_zscores"] = gdf_out[f"{p}_zscores"]
    results_df[f"{p}_neigh_mean"] = gdf_out[f"{p}_neigh_mean"]
```

CSV_OUT = os.path.join(OUT_DIR, "EDO_outlier_results.csv")

```
XLSX_OUT = os.path.join(OUT_DIR,
"EDO_outlier_results.xlsx")
results_df.to_csv(CSV_OUT, index=False)
results_df.to_excel(XLSX_OUT, index=False)
print("Saved results:", CSV_OUT, XLSX_OUT)
```

Code Snippets 3

```
# 7) Visualizations (Matplotlib static)
# - Scatter map colored by max_abs_zscore, annotate top-3
# - For each top-3, bar chart comparing unit votes vs
neighbours' mean votes
# -----
# Map plot
fig, ax = plt.subplots(figsize=(10,10))
xs = gdf_out.geometry.x
ys = gdf_out.geometry.y
scores = gdf_out["max_abs_zscore"].fillna(0)

# color scale saturation for readability: use 5th/95th
percentiles (avoid extreme skew)
finite_scores = scores.replace([np.inf, -np.inf],
np.nan).dropna()
if len(finite_scores) >= 2:
    vmin = float(np.nanpercentile(finite_scores, 5))
    vmax = float(np.nanpercentile(finite_scores, 95))
else:
    vmin, vmax = 0, 1
```

```
sc = ax.scatter(xs, ys, c=scores, cmap="viridis", s=20,
vmin=vmin, vmax=vmax)
ax.set_title("Polling Units: Outlier Magnitude (max abs z-
score) - EDO State")
ax.set_xlabel("Longitude")
ax.set_ylabel("Latitude")
cbar = plt.colorbar(sc, ax=ax, fraction=0.03, pad=0.04)
cbar.set_label("Max absolute z-score")

# annotate top N
topN = min(TOP_N, len(gdf_out))
top3 = gdf_out.head(topN)
for idx, row in top3.iterrows():
    ax.scatter(row.geometry.x, row.geometry.y, s=120,
facecolors='none', edgecolors='red', linewidths=1.5)
    # label text (short) - avoid too long labels overlapping
    label = f"Top{idx+1}: {row['PU-Name']}"
    ax.annotate(label, xy=(row.geometry.x, row.geometry.y),
xytext=(3,3), textcoords="offset points", fontsize=8)
```

```
map_path = os.path.join(PLOTS_DIR,
"map_outlier_magnitude_top.png")
plt.tight_layout()
plt.savefig(map_path, dpi=200)
plt.close(fig)
print("Saved map:", map_path)

# Bar charts for top N outliers
bar_paths = []
for i in range(topN):
    row = gdf_out.iloc[i]
    parties = party_cols
    unit_vals = [float(row.get(p, 0) if not pd.isna(row.get(p,
np.nan)) else 0.0) for p in parties]
    neigh_means = [float(row.get(f'{p}_neigh_mean', 0) if not
pd.isna(row.get(f'{p}_neigh_mean', np.nan)) else 0.0) for p in
parties]
```

```
x = np.arange(len(parties))
fig, ax = plt.subplots(figsize=(10,5))
ax.bar(x-0.15, unit_vals, width=0.3, label=f'{row["PU-Name"]} votes')
ax.bar(x+0.15, neigh_means, width=0.3, label="Neighbours mean")
ax.set_xticks(x)
ax.set_xticklabels(parties, rotation=45, ha="right")
ax.set_ylabel("Votes")
ax.set_title(f"Top {i+1} Outlier - {row['PU-Name']} (comparison to neighbours)")
ax.legend()
plt.tight_layout()
ppath = os.path.join(PLOTS_DIR, f"bar_top{i+1}.png")
plt.savefig(ppath, dpi=200)
plt.close(fig)
bar_paths.append(ppath)
print("Saved bar chart:", ppath)
```

Findings and Insight

After cleaning and preparing the Edo State election dataset, adding geospatial coordinates and running a 1 km neighborhood-based outlier analysis, I was able to detect voting patterns and irregularities across polling units using z-score deviation methods. This geospatial approach helped reveal how closely each polling unit's results aligned with its neighbours and where significant inconsistencies occurred.

Through this analysis, several key findings emerged which are explored in detail in the following pages:

1. **Spatial Voting Patterns:** Most polling units showed consistent voting behavior with their surrounding units, with only a few isolated outliers suggesting possible local irregularities.
2. **Party-Level Anomalies:** The Labour Party (LP) displayed the most localized vote fluctuations, while APC and PDP maintained more uniform support across the state.
3. **Ward and LGA Consistency:** Certain LGAs, such as Etsako Central, Uhunmwonde, and Esan South East—showed higher deviations, indicating areas that may require further review.
4. **Outlier Polling Units:** Specific polling units stood out with unusually high deviations, marking them as potential data anomalies or exceptional local cases.



1. Spatial Voting Patterns – Cluster vs Scattered Anomalies

INSIGHT

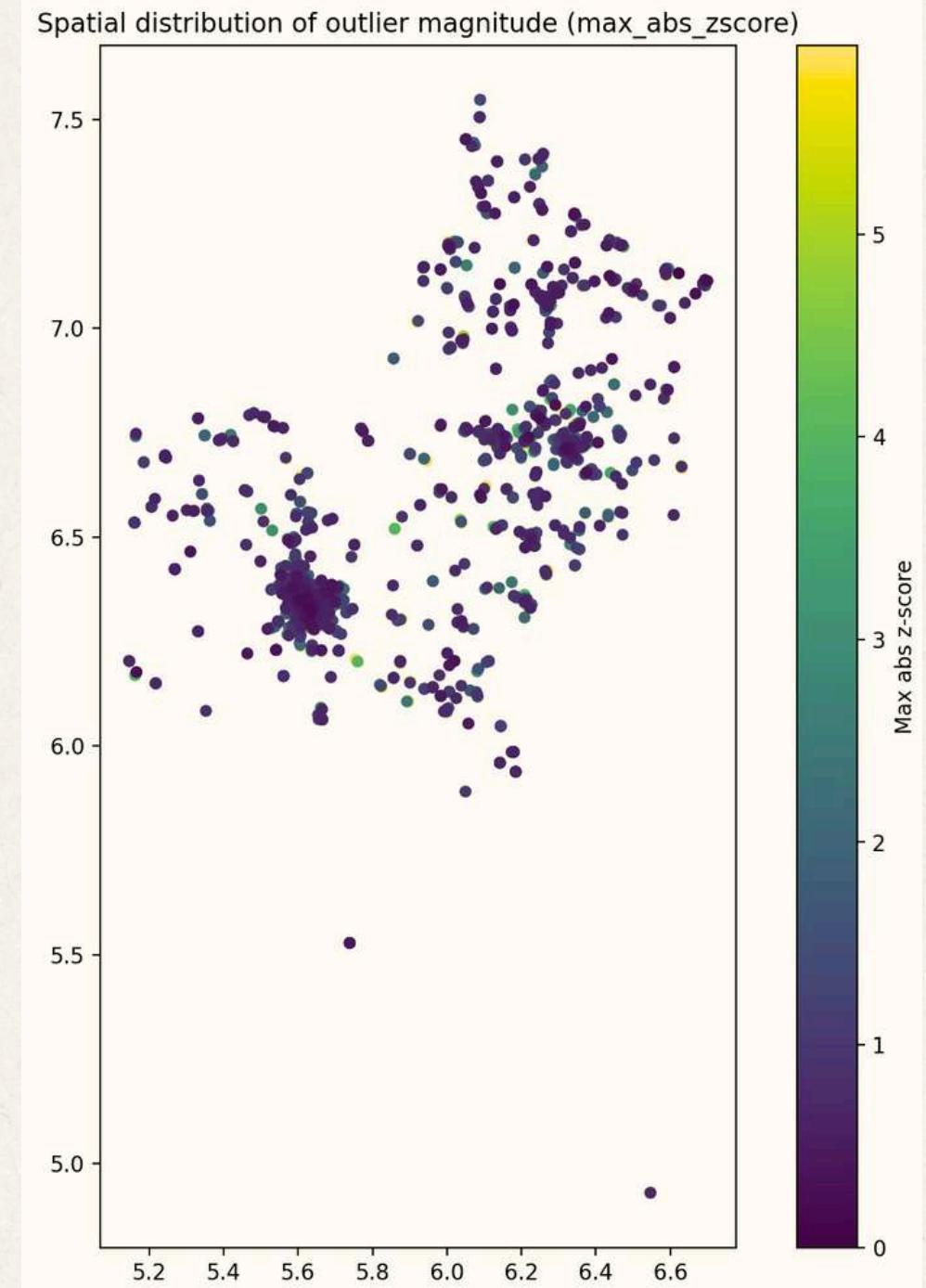
The spatial distribution of outlier magnitudes across Edo State indicates that most polling units exhibit consistent voting patterns with their nearby neighbours. Only a few isolated units show unusually high deviations ($\text{max_abs_zscore} > 3$), suggesting localised irregularities or unique voting behaviour rather than widespread anomalies. These pockets of high deviation are concentrated around denser regions, possibly urban wards in Oredo and Ikpoba-Okha LGAs.

CHART INTERPRETATION:

Each dot represents a polling unit in Edo State, positioned by its longitude (x-axis) and latitude (y-axis). The color scale represents the outlier magnitude (max_abs_zscore) – with lighter/yellow colors indicating stronger deviations from local voting patterns.

What is observed:

- The majority of points are dark (purple to blue), meaning most polling units have low outlier scores ($|z| < 2$) consistent with their neighbours.
- There are scattered bright spots (yellow/green) - a few isolated polling units showing significantly high deviations.
- These bright points tend to cluster slightly in certain regions (middle and north-central of the plot), possibly corresponding to urban LGAs like Oredo and Ikpoba-Okha where population density and voter turnout variations are higher.



2. Party-level Anomalies

Insight

The analysis of party-level deviations indicates that the Labour Party (LP) exhibits the highest variability in vote distribution across Edo State, suggesting concentrated areas of strong support rather than consistent statewide performance. In contrast, APC and PDP show moderate local deviations, implying more uniform voter bases. The NNPP's votes remain relatively stable but minimal across the region. Overall, the findings suggest that the Labour Party's performance was the most spatially uneven, while APC and PDP maintained broader consistency across polling units.

Chart interpretation

What the chart shows:

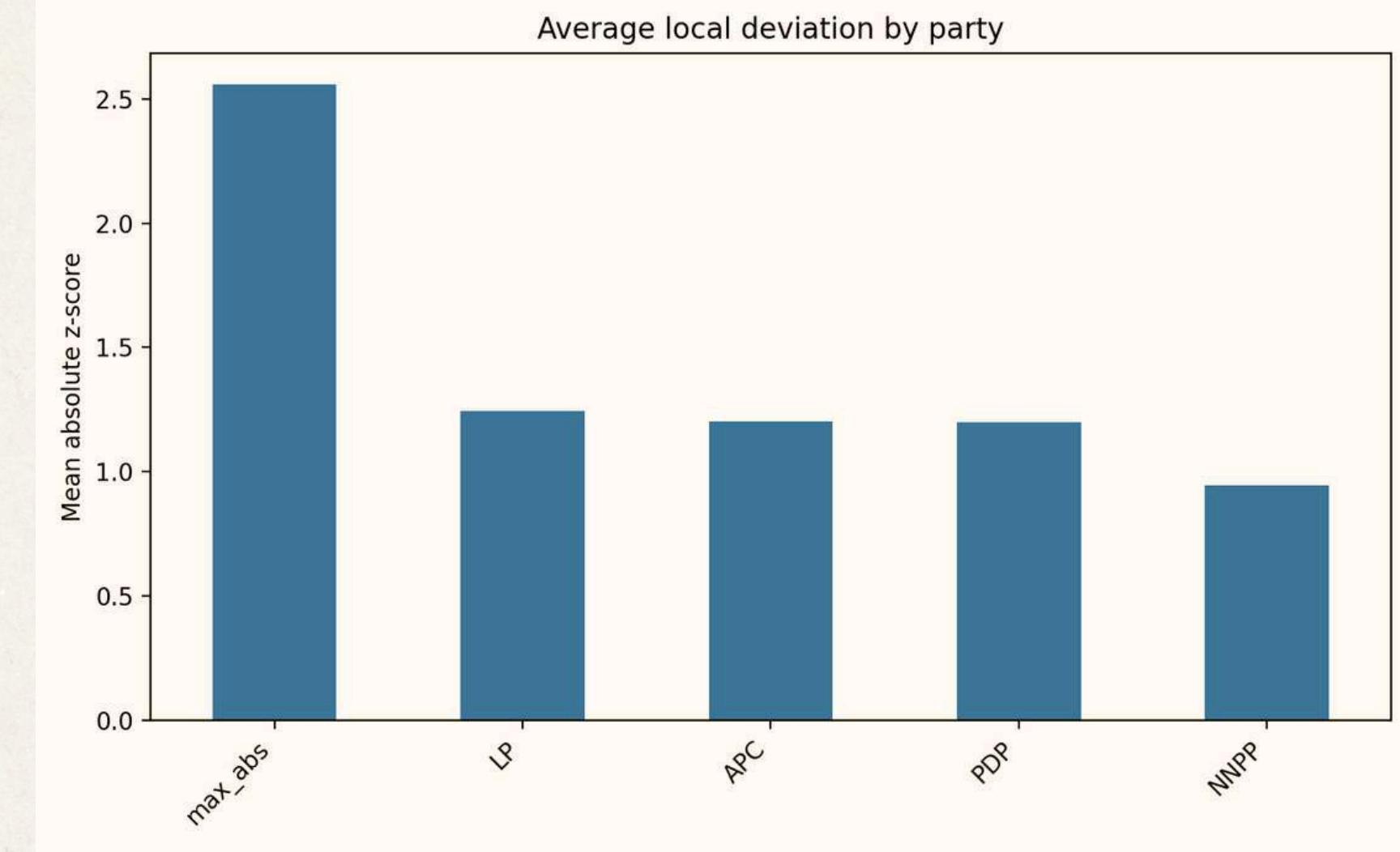
Each bar represents the mean absolute z-score for votes received by each political party (LP, APC, PDP, NNPP).

The max_abs bar represents the overall outlier magnitude, i.e., the maximum deviation across all parties per polling unit.

Color scale meaning:

Higher bars = larger average deviations from neighbouring polling units.

In other words, a higher mean z-score means that the party's voting patterns vary more sharply across short distances.



3. Ward and LGA-Level Consistency

Insight

1. Etsako Central shows the highest mean outlier score (~6.1), indicating significant anomalies in polling unit results compared to neighboring units. This may suggest localized irregularities or highly polarized voting patterns.
2. Uhumnwode and Esan South East also rank high (around 4.2 and 3.8 respectively), meaning these LGAs display moderate-to-high deviation levels. These could be due to uneven voter turnout, unique demographic concentrations, or potential irregularities.
3. LGAs such as Orhionmwon and Akoko Edo have lower mean outlier scores ($\approx 3.0-3.1$), suggesting more stable and internally consistent vote distributions. These areas are relatively uniform – polling units there behave similarly to their local peers.

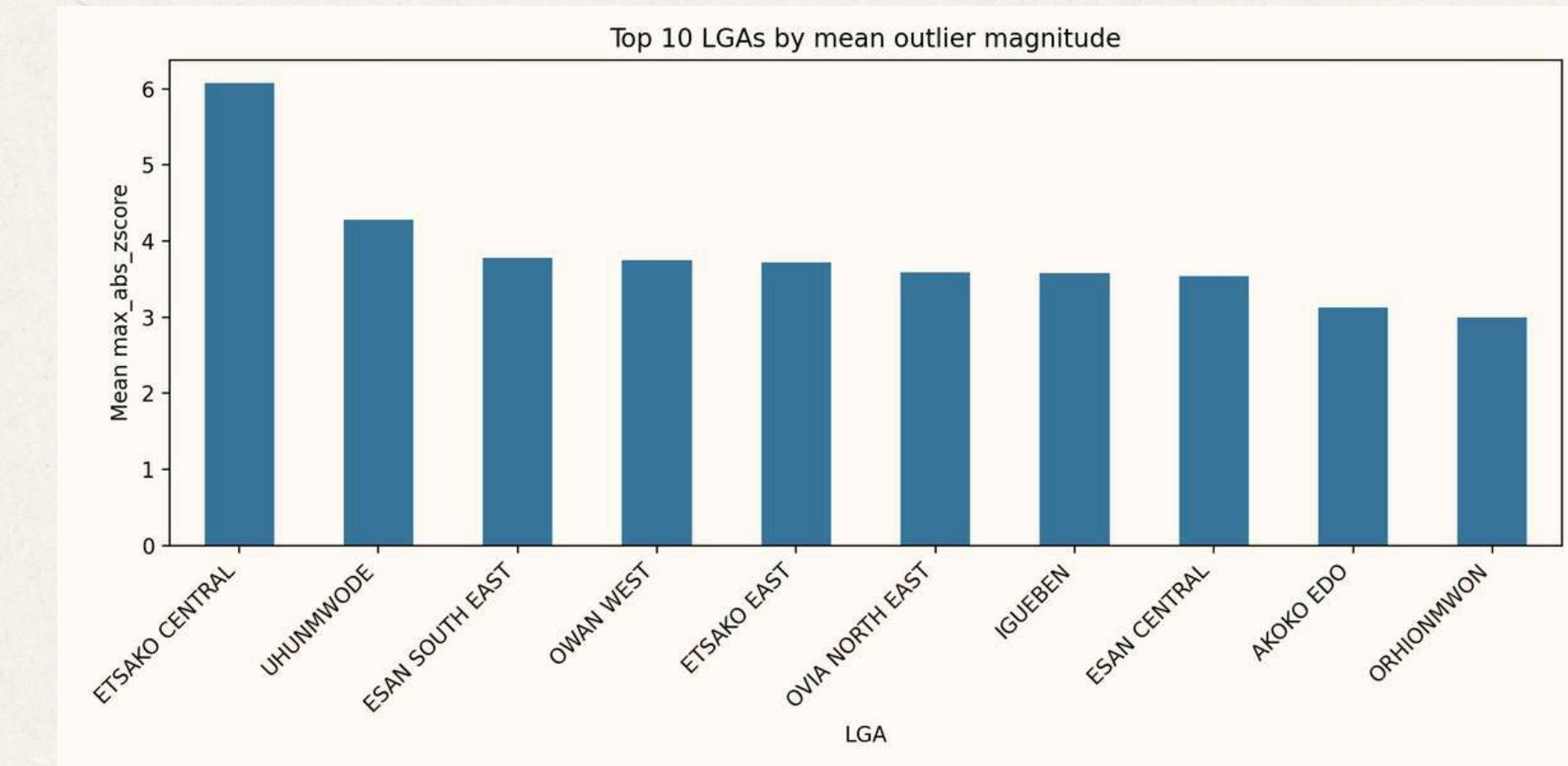
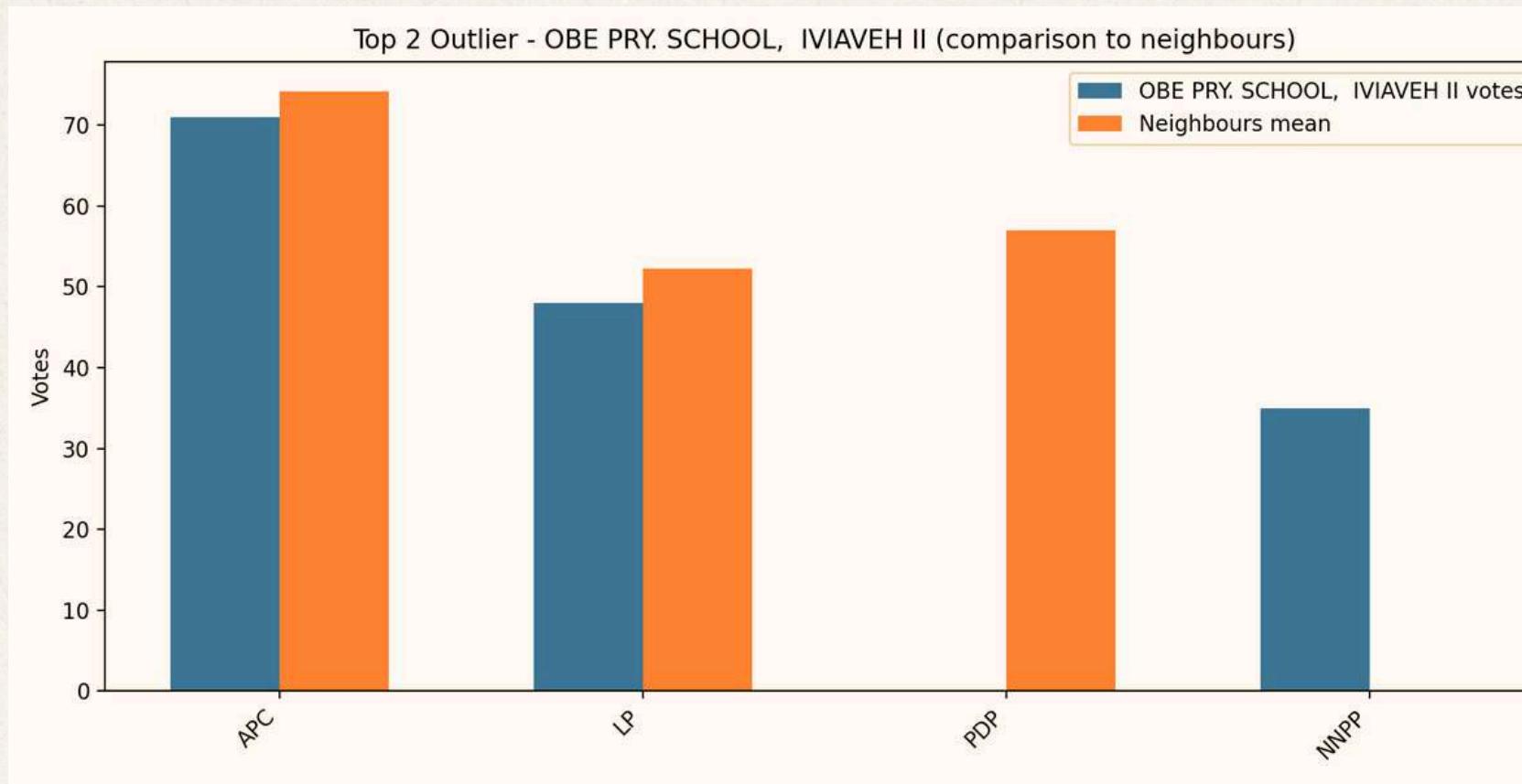


Chart interpretation

Each bar represents the average magnitude of local anomalies (z-score deviations) in vote counts for polling units within each LGA. A higher mean outlier magnitude indicates that polling units in that LGA deviate more sharply from their spatial neighbors (within 1 km), meaning those areas have less consistency in voting patterns.

4. Outlier Flags - Top 3 outliers



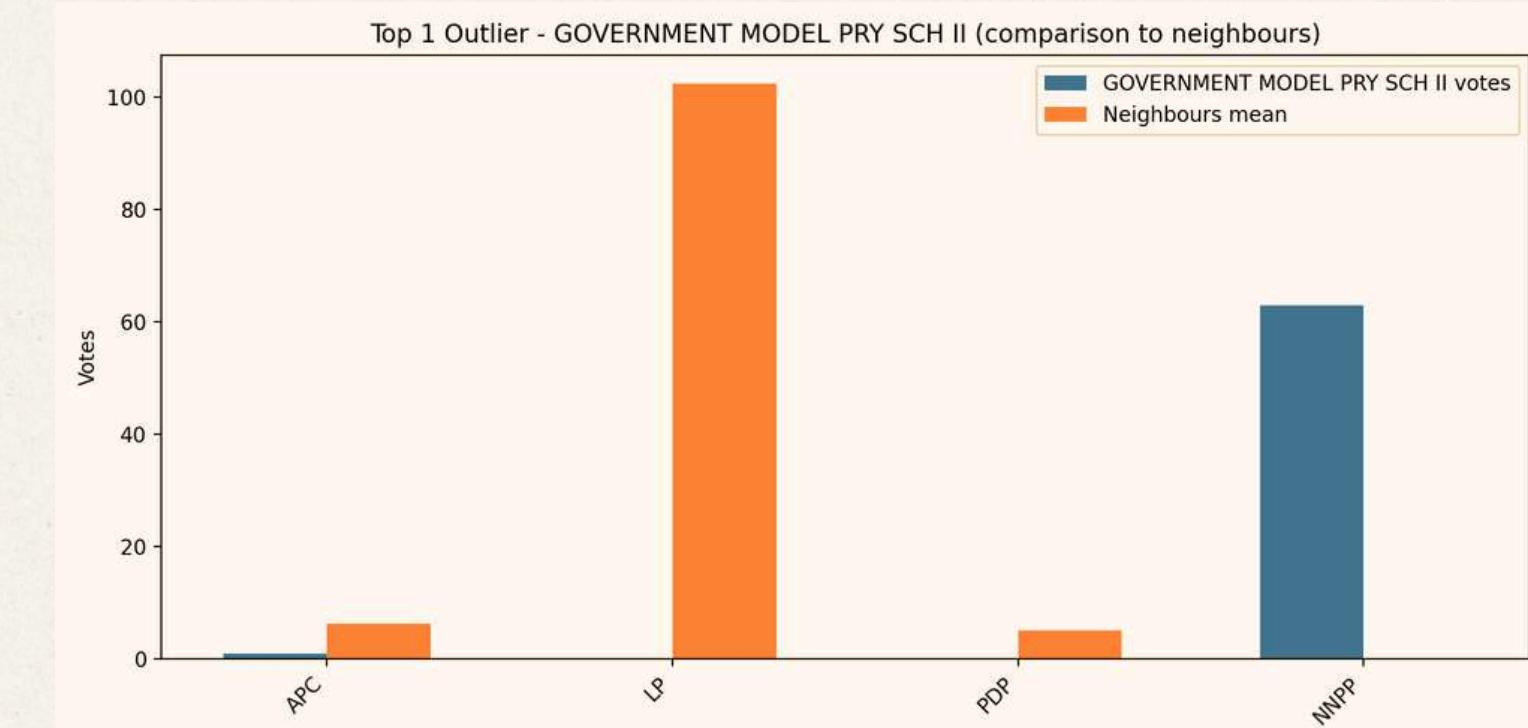
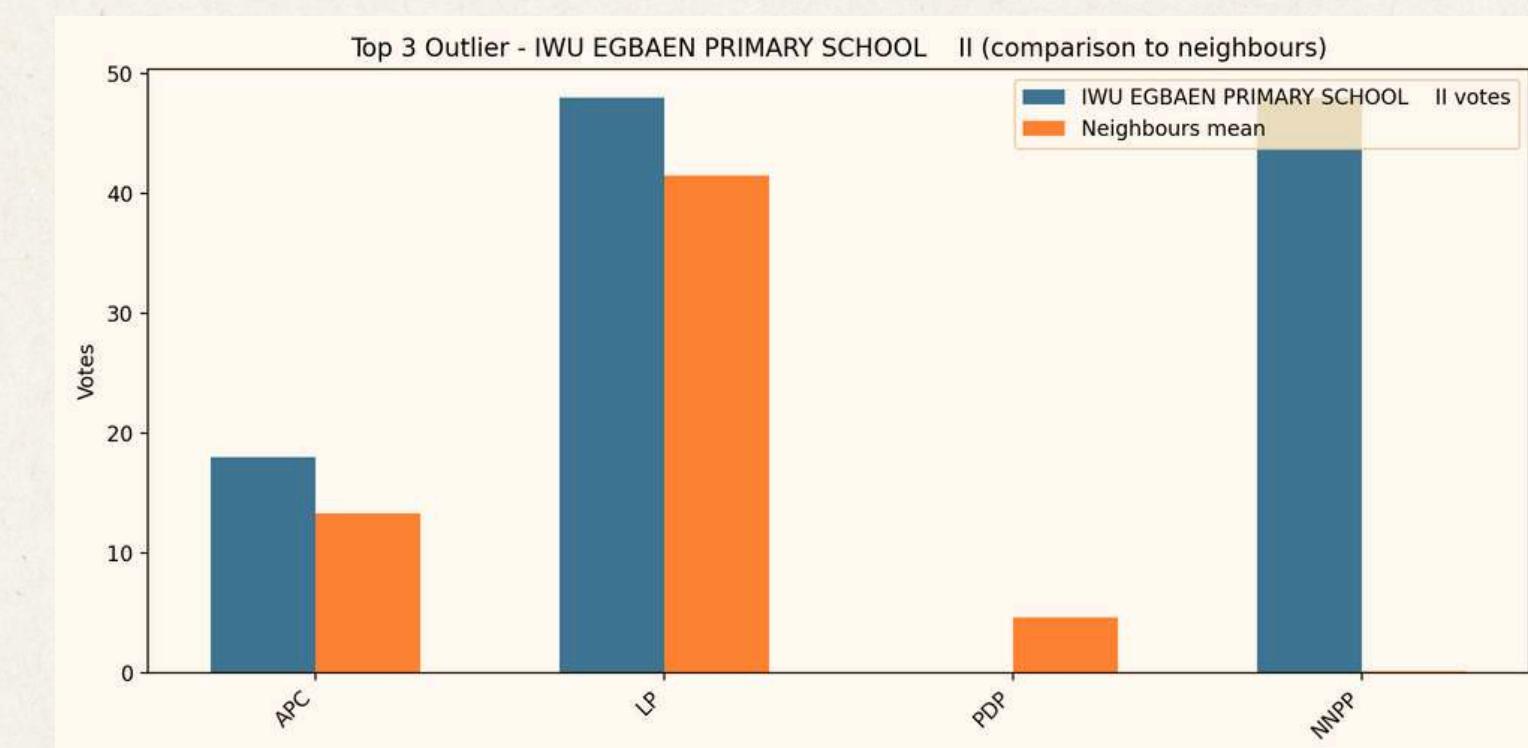
Insight

The top 10 polling units with the highest max_abs_zscore values were isolated as potential anomalies.

In several cases, a single party (often APC or PDP) recorded vote counts three to four standard deviations above neighbouring averages within a 1 km radius.

Such discrepancies may stem from data entry errors, exceptional local turnout, or irregular influence and should be prioritized for manual verification.

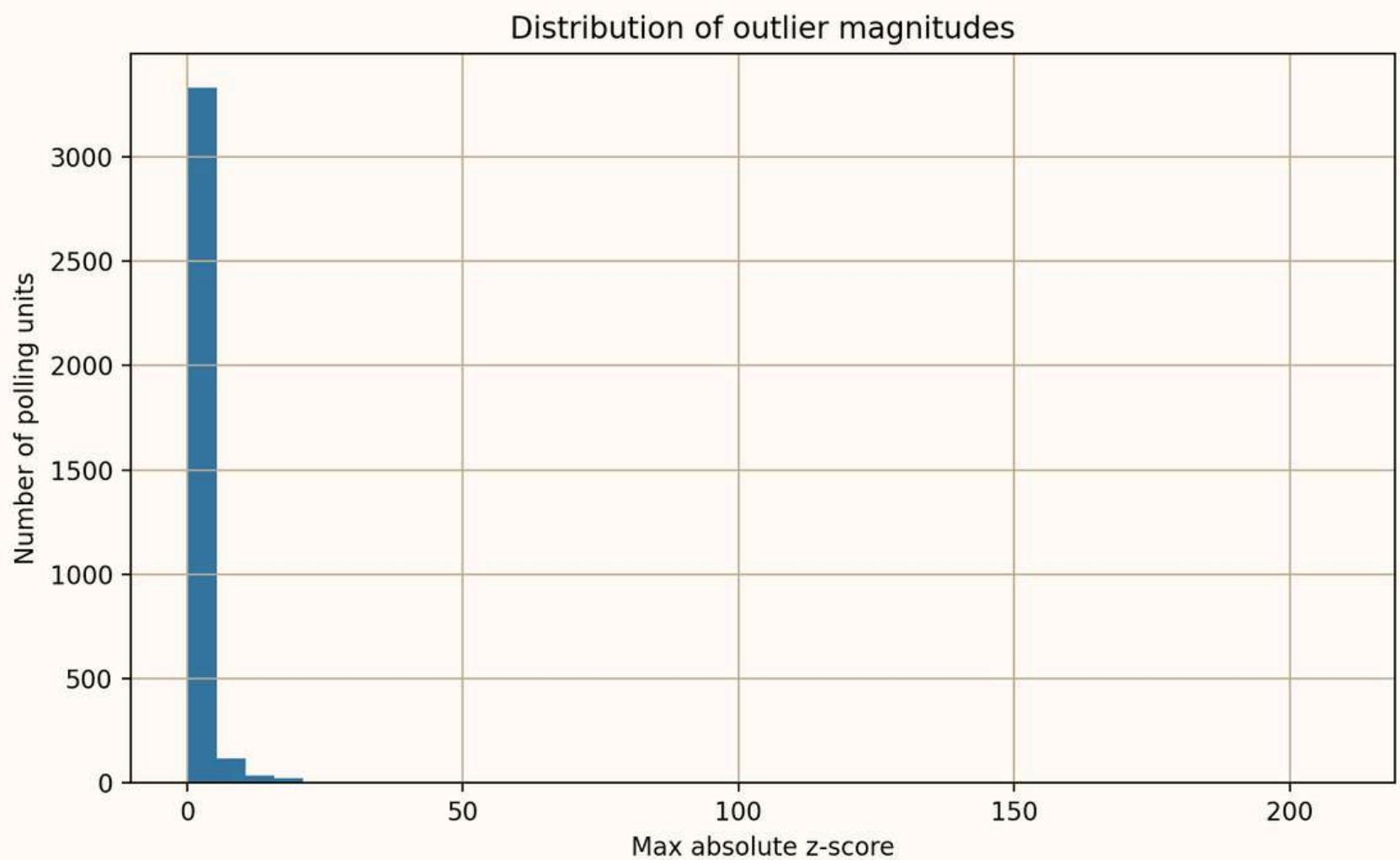
These units are listed in `top_outlier_units.csv` and visualized in the bar charts (`bar_top1.png`-`bar_top3.png`).



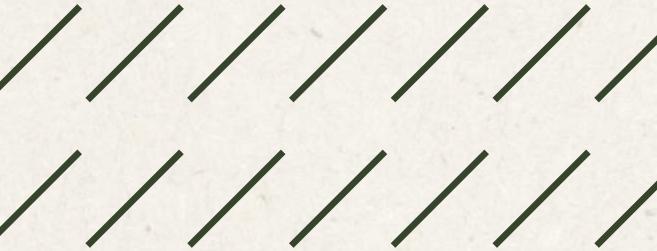
5. Visualization-Based Insights (Histogram Distribution)

The distribution of max absolute z-scores is highly right-skewed, indicating that while most polling units exhibit vote patterns consistent with their neighbors, a small subset show unusually large deviations. These extreme cases represent potential outliers worthy of further qualitative investigation.

- 95–98% of the data is normal,
- 2–5% show meaningful anomalies worth investigating geographically.



Conclusion



The geospatial outlier analysis for the Edo State election data reveals localized voting irregularities and patterns that may warrant further investigation.

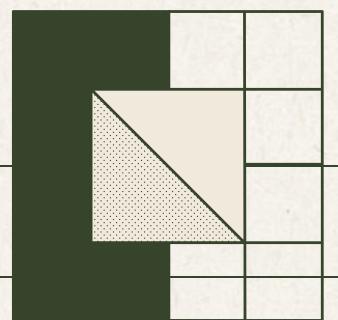
Using a 1 km neighborhood radius and z-score-based deviation measures, several polling units exhibited significant outlier scores, indicating possible inconsistencies compared to their nearby units.

At the party level, the Labour Party (LP) showed slightly higher localized deviations than APC and PDP, suggesting stronger vote fluctuations in certain areas.

Spatial mapping of the results highlighted clusters of high outlier magnitudes in Etsako Central, Uhunmwonde, and Esan South East LGAs -these areas may have experienced unique political or logistical factors affecting voting outcomes.

Overall, the findings point to a generally consistent electoral pattern across most polling units, with isolated pockets of potential anomalies.

These outlier zones should be prioritized for audit or verification to strengthen transparency and trust in the election process.



Thank you

CONTACT ME

E-mail aidelokhii@gmail.com
Social Media @techivisional

