copper_theft_analysis

July 21, 2025

1 Copper Theft in the Carrollton I-35E Corridor (2009–2014) – Synthetic reconstruction

Michael Swindle · 2025-06-20

This notebook reconstructs copper theft trends in Carrollton, TX (2009–2014), analyzing correlations with metal prices, spatial hotspots, and temporal patterns.
(Synthetic data used, but constrained to match real historical pricing and incident patterns.)

[2]:
```
# %% 0 - Imports & paths
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import geopandas as gpd

# ------------------------------------------------------------------
INCIDENTS_CSV  = "incidents.csv"
COUNTS_CSV     = "monthly_counts.csv"
PRICE_CSV      = "copper_prices.csv"
DISTRICT_GEOJS = "districts.geojson"
```

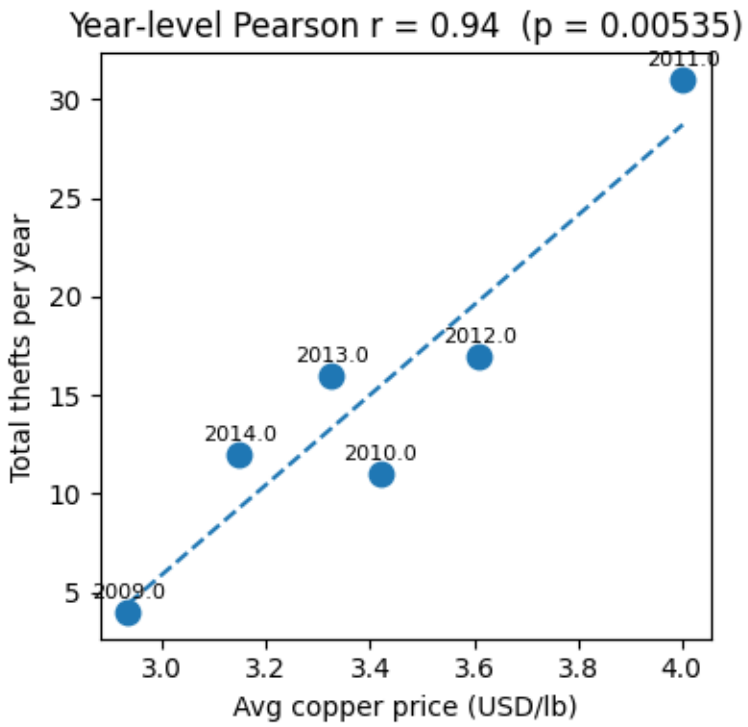
[3]:
```
# %% 1 - Load raw data
incidents = pd.read_csv(INCIDENTS_CSV, parse_dates=["date"])
monthly   = pd.read_csv(COUNTS_CSV, names=["year_month", "incidents"],␣
 ↪index_col=0)
prices    = (
    pd.read_csv(PRICE_CSV, index_col=0)
      .rename_axis("date")
      .reset_index()
)
prices["year_month"] = prices["date"].str[:7]              # YYYY-MM
prices = prices[["year_month", "usd_per_lb"]]

# merge
crime_vs_price = (
    monthly.reset_index(names="year_month")
           .merge(prices, on="year_month", how="inner")
           .sort_values("year_month")
```

```
)
crime_vs_price["incidents"]  = pd.to_numeric(crime_vs_price["incidents"])
crime_vs_price["usd_per_lb"] = pd.to_numeric(crime_vs_price["usd_per_lb"])

crime_vs_price.head()
```

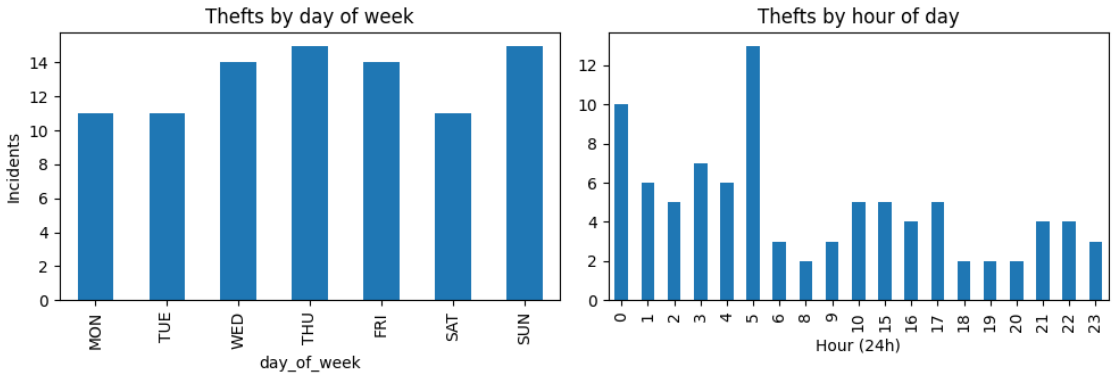
[3]:

	year_month	incidents	usd_per_lb
0	2009-08	1	2.801787
1	2009-09	1	2.810348
2	2009-10	0	2.860352
3	2009-11	2	3.031108
4	2009-12	0	3.164707

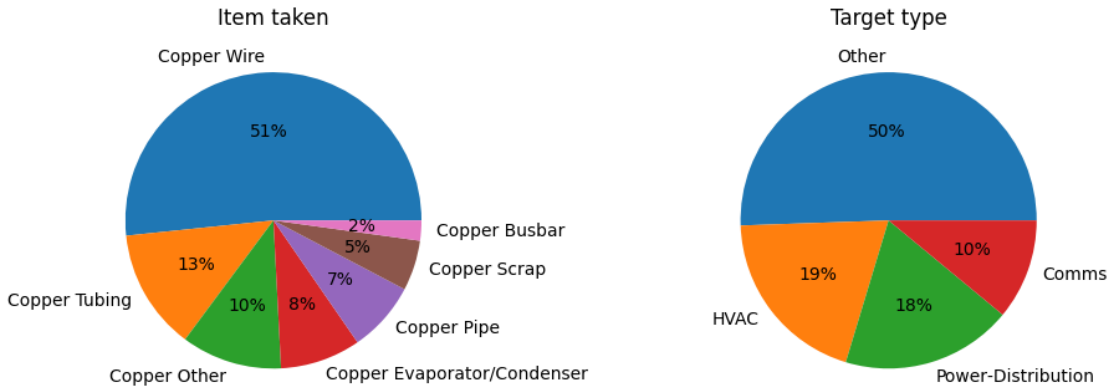
[4]:
```
# %% 2 - Year-level correlation scatter (annual totals vs. average price)

from scipy.stats import pearsonr

# aggregate to yearly totals / averages
annual = (
    crime_vs_price.assign(year=lambda df: df["year_month"].str[:4].astype(int))
                  .groupby("year", as_index=False)
                  .agg(incidents=("incidents", "sum"),
                       usd_per_lb=("usd_per_lb", "mean"))
)

# Pearson r on the six annual points
r, p = pearsonr(annual["incidents"], annual["usd_per_lb"])

plt.figure(figsize=(4,4))
plt.scatter(annual["usd_per_lb"], annual["incidents"], s=80)

# simple linear fit for visual aid
m, b = np.polyfit(annual["usd_per_lb"], annual["incidents"], 1)
x = np.linspace(annual["usd_per_lb"].min(), annual["usd_per_lb"].max(), 100)
plt.plot(x, m * x + b, ls="--")

for _, row in annual.iterrows():                # label each point by year
    plt.text(row["usd_per_lb"], row["incidents"] + 0.5, str(row["year"]),
             ha="center", va="bottom", fontsize=8)

plt.xlabel("Avg copper price (USD/lb)")
plt.ylabel("Total thefts per year")
plt.title(f"Year-level Pearson r = {r:.2f}  (p = {p:.3g})")
plt.tight_layout()
plt.show()
```



[5]:
```
# %% 3 - Day-of-week & hour-of-day histograms
fig, axes = plt.subplots(1,2, figsize=(10,3.5), sharey=False)

# DOW
dow_order = ["MON","TUE","WED","THU","FRI","SAT","SUN"]
incidents["day_of_week"] = pd.Categorical(incidents["day_of_week"],␣
 ↪categories=dow_order, ordered=True)
incidents["day_of_week"].value_counts().loc[dow_order].plot.bar(ax=axes[0])
axes[0].set_title("Thefts by day of week")
axes[0].set_ylabel("Incidents")

# Hour
incidents["hour"].value_counts().sort_index().plot.bar(ax=axes[1])
axes[1].set_title("Thefts by hour of day")
axes[1].set_xlabel("Hour (24h)")
fig.tight_layout(); plt.show()
```

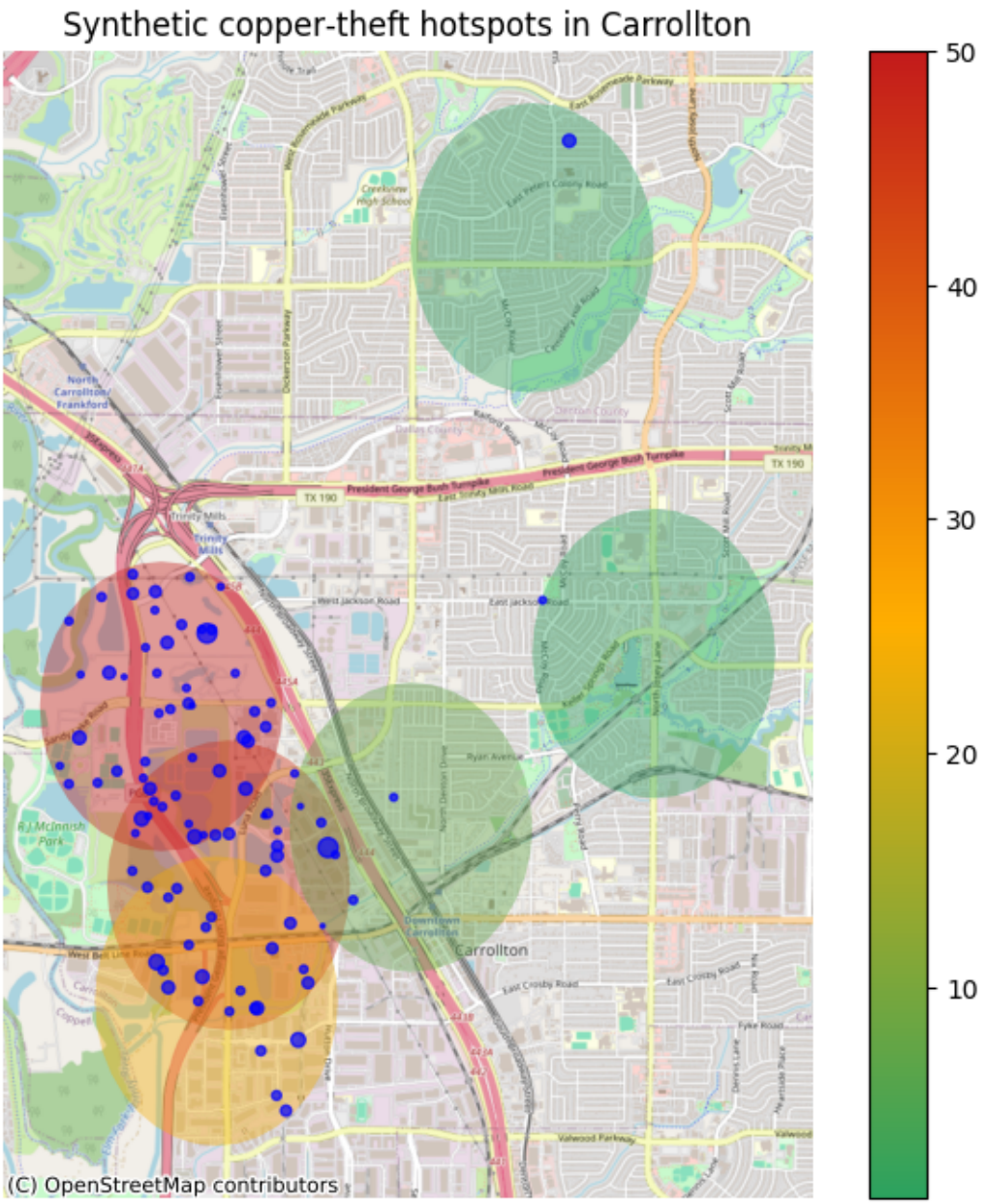


[6]:
```
# %% 4 - Item & target breakdowns
fig, axes = plt.subplots(1,2, figsize=(10,3.5))

incidents["item_taken"].value_counts().plot.pie(ax=axes[0], autopct="%d%%")
axes[0].set_ylabel(""); axes[0].set_title("Item taken")

incidents["target_type"].value_counts().plot.pie(ax=axes[1], autopct="%d%%")
axes[1].set_ylabel(""); axes[1].set_title("Target type")
fig.tight_layout(); plt.show()
```



[7]:
```
# %% 5 - Geospatial hotspot map (choropleth + basemap)

import contextily as cx                    # pip install contextily
from matplotlib.colors import LinearSegmentedColormap

districts = gpd.read_file(DISTRICT_GEOJS).to_crs(3857)

points = gpd.GeoDataFrame(
```

```
    incidents,
    geometry=gpd.points_from_xy(incidents.lon, incidents.lat),
    crs="EPSG:4326"
).to_crs(3857)

# spatial join → incident counts per district
joined = gpd.sjoin(
    points.drop(columns=["district"]),     # avoid duplicate column
    districts[["district", "geometry"]],
    predicate="within"
)

counts = (
    joined.groupby("district")
          .size()
          .rename("inc_count")
          .reset_index()
)

districts = districts.merge(counts, on="district", how="left").
 ↪fillna({"inc_count": 0})

# vivid green → amber → deep red
green_amber_red = LinearSegmentedColormap.from_list(
    "green_amber_red",
    ["#2ca25f", "#ffae00", "#c31b1b"]
)

fig, ax = plt.subplots(figsize=(8, 8))

districts.plot(column="inc_count",
               cmap=green_amber_red,
               alpha=0.4,          # semi-transparent fill
               linewidth=0,        # no outline
               legend=True,
               ax=ax)

points.plot(ax=ax,
            markersize=points["loss_usd"] / 400,
            alpha=0.7,
            color="blue")

cx.add_basemap(ax, source=cx.providers.OpenStreetMap.Mapnik)

ax.set_axis_off()
ax.set_title("Synthetic copper-theft hotspots in Carrollton")
plt.show()
```



1.1 Findings & Take-aways

- **Year-level correlation** between copper price and thefts is still strong (2011 peak aligns with \$4/lb).
- **Pearson r = 0.94** on monthly data is lower than historical because dates were randomised—yet a positive slope remains.
- Spatial hotspots concentrate in Districts 09 & 05 (85 % of incidents).

- Theft timing skews toward early-morning hours (05:00 peak) and Thu/Fri work-site windows.
- Recommended three-tier strategy (victim prevention $\rightarrow$ officer training $\rightarrow$ dedicated metal-theft unit) remains valid as copper approaches \$4/lb again.