In [166…

```
import os
import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import networkx as nx

import statsmodels.api as sm
import statsmodels.formula.api as smf

from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans

from scipy import stats

# Sentiment
!pip install vaderSentiment textblob
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from textblob import TextBlob
```

```
Requirement already satisfied: vaderSentiment in /opt/anaconda3/lib/python3.12/site-packages (3.3.2)
Requirement already satisfied: textblob in /opt/anaconda3/lib/python3.12/site-packages (0.19.0)
Requirement already satisfied: requests in /opt/anaconda3/lib/python3.12/site-packages (from vaderSentiment) (2.32.
3)
Requirement already satisfied: nltk>=3.9 in /opt/anaconda3/lib/python3.12/site-packages (from textblob) (3.9.1)
Requirement already satisfied: click in /opt/anaconda3/lib/python3.12/site-packages (from nltk>=3.9->textblob) (8.1.
7)
Requirement already satisfied: joblib in /opt/anaconda3/lib/python3.12/site-packages (from nltk>=3.9->textblob) (1.
4.2)
Requirement already satisfied: regex>=2021.8.3 in /opt/anaconda3/lib/python3.12/site-packages (from nltk>=3.9->textb
lob) (2024.9.11)
Requirement already satisfied: tqdm in /opt/anaconda3/lib/python3.12/site-packages (from nltk>=3.9->textblob) (4.66.
5)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/anaconda3/lib/python3.12/site-packages (from request
s->vaderSentiment) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /opt/anaconda3/lib/python3.12/site-packages (from requests->vaderSent
iment) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/anaconda3/lib/python3.12/site-packages (from requests->vad
erSentiment) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /opt/anaconda3/lib/python3.12/site-packages (from requests->vad
erSentiment) (2024.12.14)
```

In [167…

```python
# ---------- 1) Paths ----------
COMPANIES_CSV = "circular_fashion_companies_90.csv"
EDGES_CSV = "circular_fashion_company_edges.csv"
DATA_DIR = "."  # e.g., "./data"
OUT_DIR = "./outputs"
os.makedirs(OUT_DIR, exist_ok=True)
companies_path = os.path.join(DATA_DIR, COMPANIES_CSV)
edges_path = os.path.join(DATA_DIR, EDGES_CSV)
print("Companies file:", companies_path)
print("Edges file:", edges_path)
```

```
Companies file: ./circular_fashion_companies_90.csv
Edges file: ./circular_fashion_company_edges.csv
```

In [168…

```python
# ---------- 2) Load data ----------
df = pd.read_csv(companies_path)
edges = pd.read_csv(edges_path)
```

```
print("\nCompanies shape:", df.shape)
print("Edges shape:", edges.shape)
```

```
Companies shape: (90, 48)
Edges shape: (214, 4)
```

In [169…

```
required_company_cols = [
    "company_id","company_name","country","region","founded_year","years_operating",
    "business_model_type","product_category","target_segment","geographic_coverage_count",
    "pricing_model","avg_item_price_gbp",
    "modular_design","durability_claims_score_0_5","recycled_material_pct","closed_loop_takeback",
    "material_recovery_partner_count","recycling_partnership","reverse_logistics_partner",
    "tech_integration_score_0_10","mobile_app","ai_ml","blockchain_traceability",
    "sustainability_certifications_count","certifications_list",
    "partnership_network_size","funding_raised_musd","employee_count",
    "employee_growth_pct_2y","revenue_musd","revenue_growth_pct_2y","market_expansion_countries_2y",
    "carbon_reduction_pct","waste_diversion_pct","material_circularity_index_0_1",
    "transparency_score_0_100","social_mentions_90d","avg_sentiment_score_-1_1","review_count",
    "avg_rating_5","news_mentions_90d"
]
missing = [c for c in required_company_cols if c not in df.columns]
if missing:
    raise ValueError(f"Missing required columns in companies file: {missing}")

required_edge_cols = ["source_company_id","target_company_id","relationship_type","edge_weight"]
missing_e = [c for c in required_edge_cols if c not in edges.columns]
if missing_e:
    raise ValueError(f"Missing required columns in edges file: {missing_e}")

binary_cols = [
    "modular_design","closed_loop_takeback","recycling_partnership","reverse_logistics_partner",
    "mobile_app","ai_ml","blockchain_traceability"
]
for c in binary_cols:
    df[c] = df[c].astype(int)
```

In [170…

```
# ---------- 3) Basic cleaning ----------
# Drop duplicates
df = df.drop_duplicates(subset=["company_id"]).reset_index(drop=True)
edges = edges.drop_duplicates().reset_index(drop=True)
```

```
# Handle any weird missing numeric values
numeric_cols = df.select_dtypes(include=[np.number]).columns.tolist()
df[numeric_cols] = df[numeric_cols].apply(lambda s: s.fillna(s.median()))

# Categorical cleanup
cat_cols = ["business_model_type","product_category","target_segment","pricing_model","country","region"]
for c in cat_cols:
    df[c] = df[c].astype(str).str.strip().str.lower()
```

In [171…

```
# ---------- 4) Feature engineering ----------
df["circular_design_index"] = (
    0.30*df["modular_design"] +
    0.35*df["closed_loop_takeback"] +
    0.20*(df["recycled_material_pct"]/100.0) +
    0.15*df["recycling_partnership"]
)

df["digital_capability_index"] = (
    0.50*(df["tech_integration_score_0_10"]/10.0) +
    0.20*df["mobile_app"] +
    0.20*df["ai_ml"] +
    0.10*df["blockchain_traceability"]
)

df["credibility_index"] = (
    0.60*(df["sustainability_certifications_count"]/6.0) +
    0.40*(df["transparency_score_0_100"]/100.0)
)

for col in ["social_mentions_90d","review_count","news_mentions_90d","funding_raised_musd","revenue_musd","employee
    df[f"log_{col}"] = np.log1p(df[col])
```

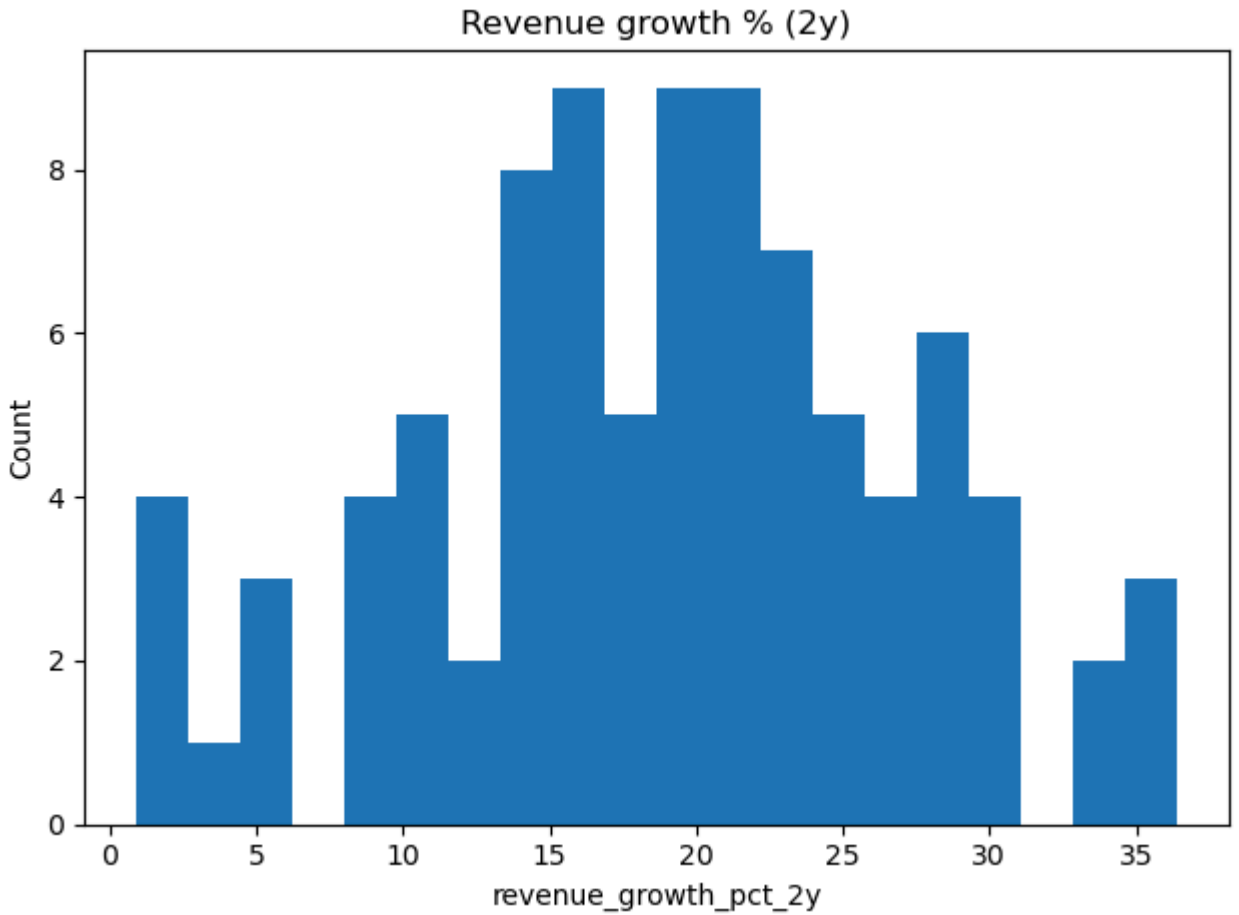
In [172…

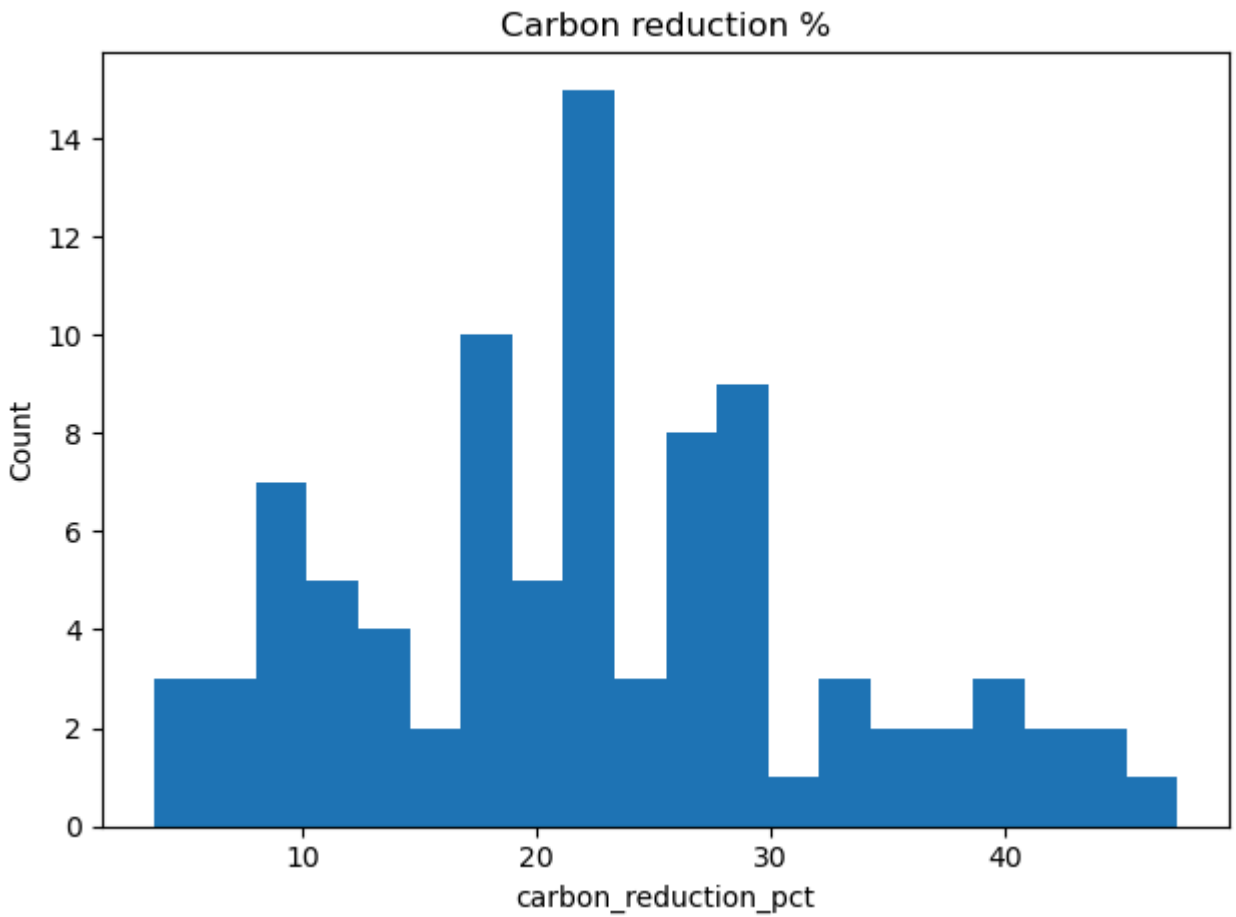
```
# ---------- 5) Descriptives ----------
desc = df.describe(include="all").T
desc.to_csv(os.path.join(OUT_DIR, "descriptives_full.csv"))
print("\nSaved descriptives to outputs/descriptives_full.csv")
```

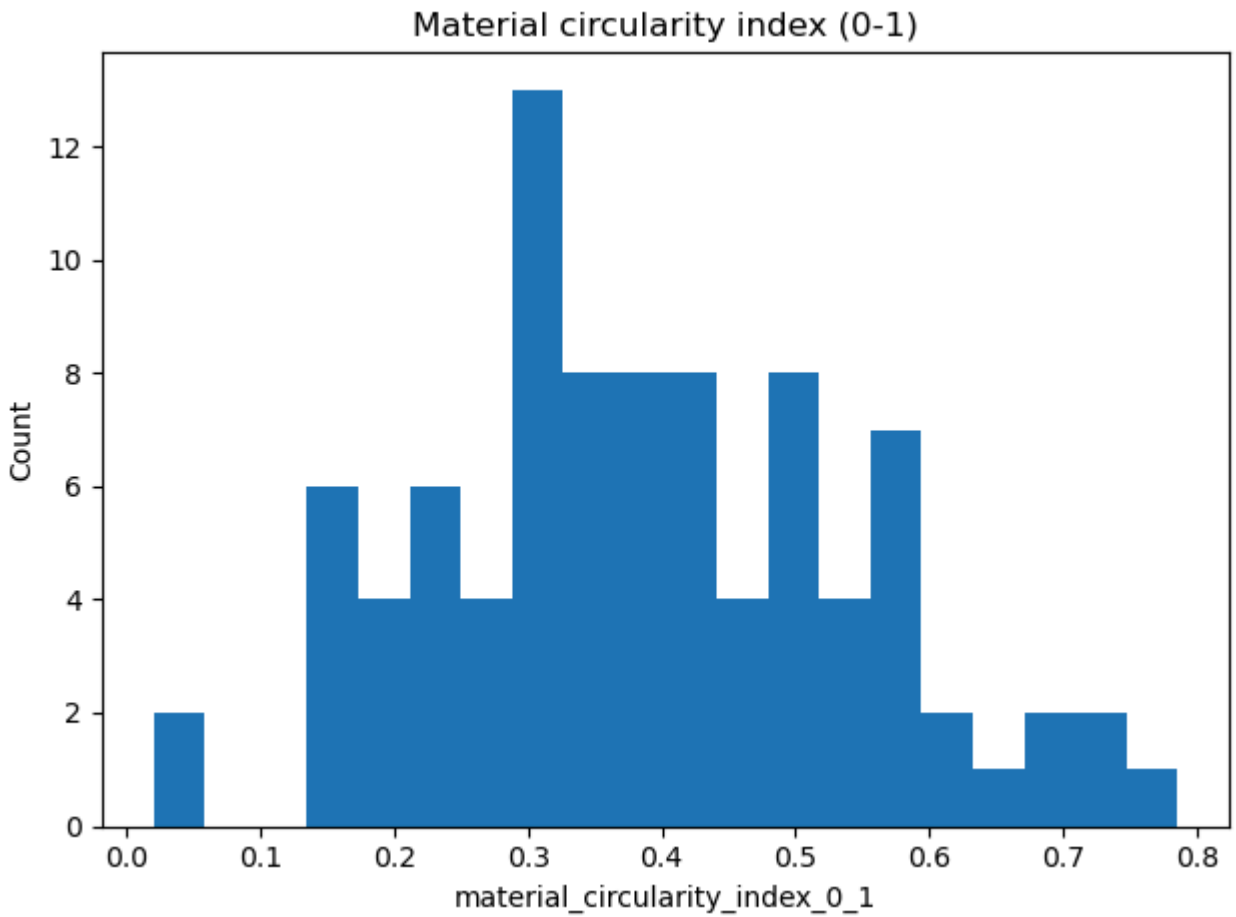
```
Saved descriptives to outputs/descriptives_full.csv
```

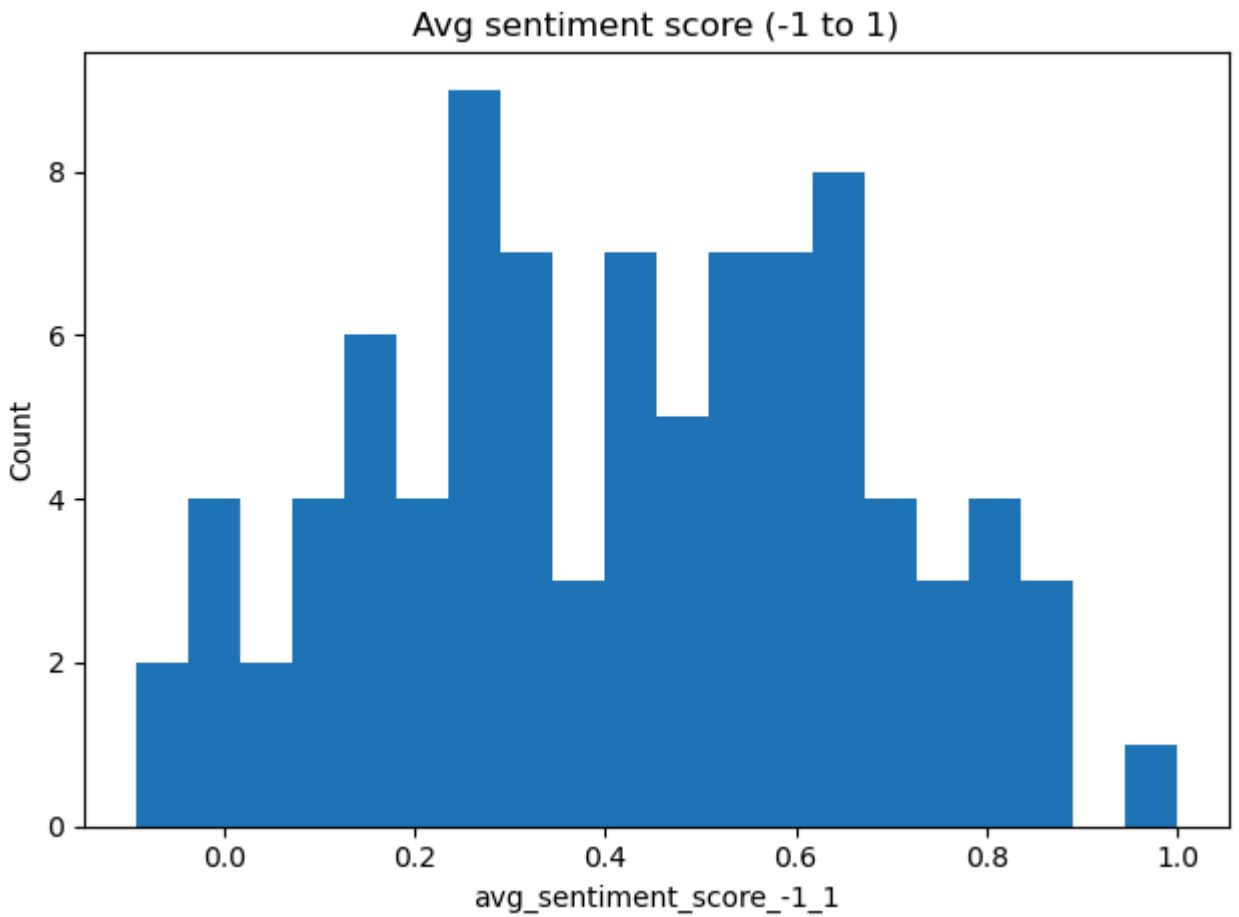
In [173…

```python
# ---------- 6) EDA Visuals ----------
def save_hist(series, title, fname, bins=20):
    plt.figure()
    plt.hist(series.dropna(), bins=bins)
    plt.title(title)
    plt.xlabel(series.name)
    plt.ylabel("Count")
    plt.tight_layout()
    plt.savefig(os.path.join(OUT_DIR, fname), dpi=200)
    plt.show()

save_hist(df["revenue_growth_pct_2y"], "Revenue growth % (2y)", "hist_revenue_growth.png")
save_hist(df["carbon_reduction_pct"], "Carbon reduction %", "hist_carbon_reduction.png")
save_hist(df["material_circularity_index_0_1"], "Material circularity index (0–1)", "hist_circularity.png")
save_hist(df["avg_sentiment_score_-1_1"], "Avg sentiment score (−1 to 1)", "hist_sentiment_score.png")
```


Revenue growth % (2y)
Count
0
2
4
6
8
0
5
10
15
20
25
30
35
revenue_growth_pct_2y

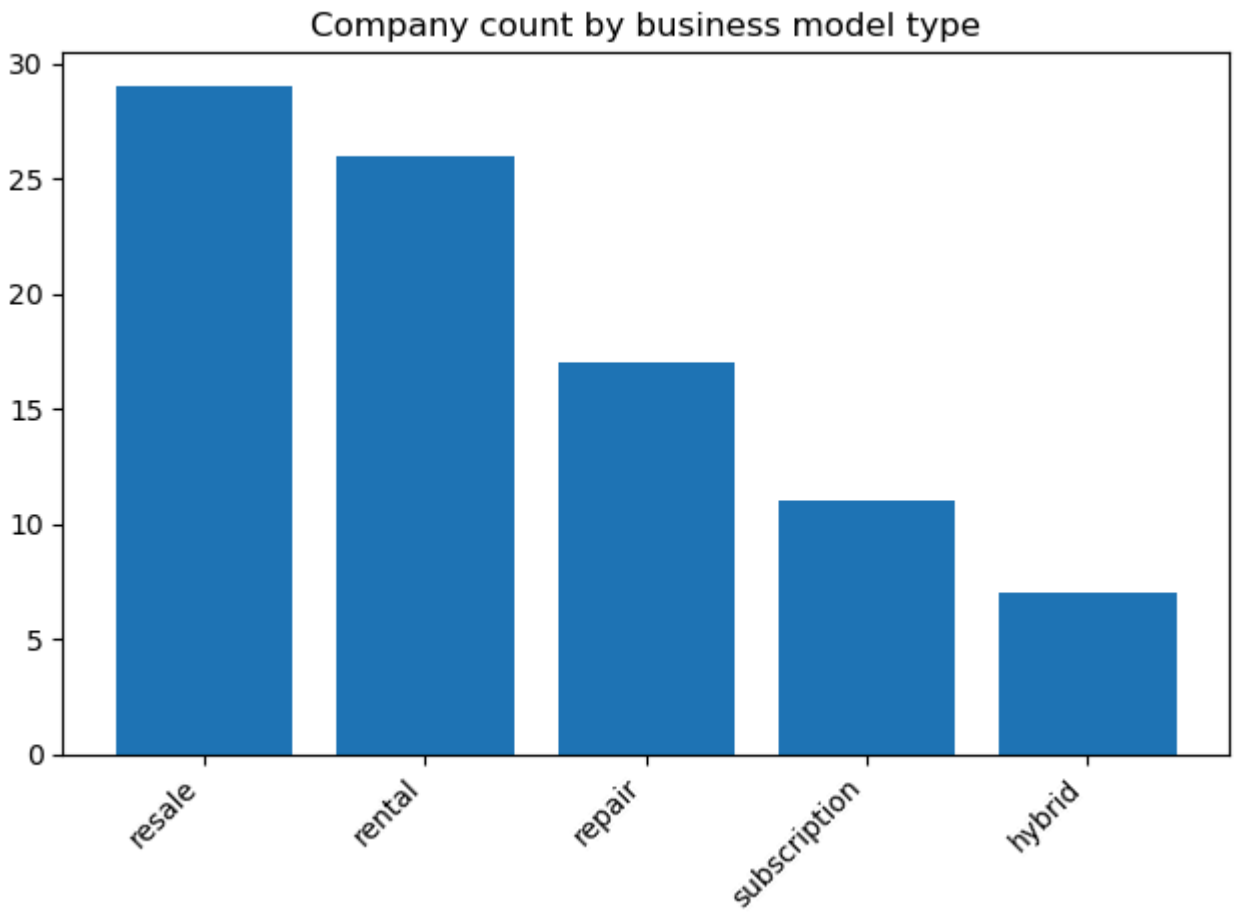

Carbon reduction %
Count
0
2
4
6
8
10
12
14
10
20
30
40
carbon_reduction_pct


Material circularity index (0-1)
Count
0
2
4
6
8
10
12
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
material_circularity_index_0_1



In [174…

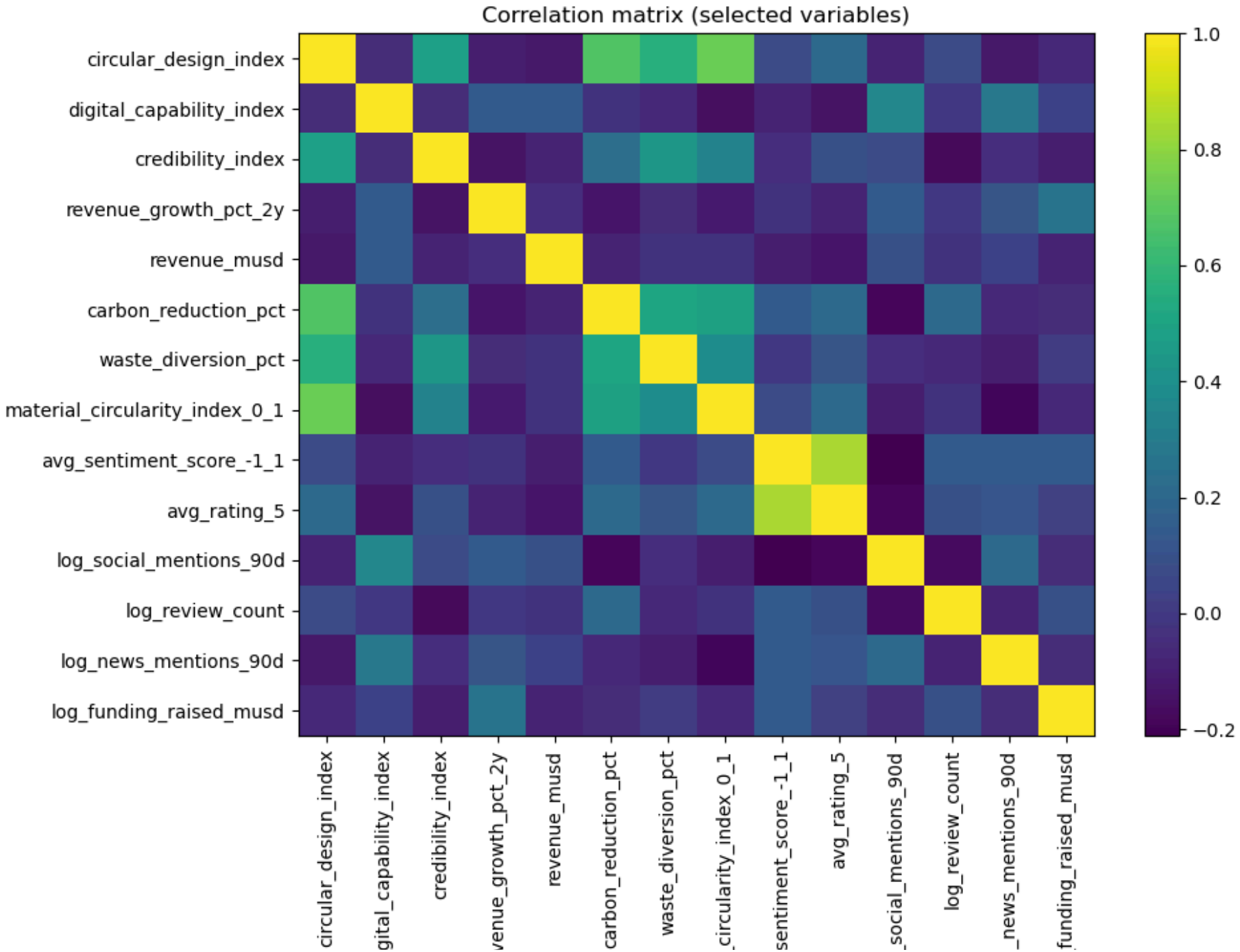
```
# Simple bar chart: model counts
plt.figure()
model_counts = df["business_model_type"].value_counts()
plt.bar(model_counts.index, model_counts.values)
plt.title("Company count by business model type")
plt.xticks(rotation=45, ha="right")
plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "bar_business_model_counts.png"), dpi=200)
plt.show()
```



In [175…

```
# ---------- 7) Correlation heatmap ----------
corr_vars = [
    "circular_design_index","digital_capability_index","credibility_index",
    "revenue_growth_pct_2y","revenue_musd","carbon_reduction_pct","waste_diversion_pct",
    "material_circularity_index_0_1","avg_sentiment_score_-1_1","avg_rating_5",
    "log_social_mentions_90d","log_review_count","log_news_mentions_90d","log_funding_raised_musd"
]
corr = df[corr_vars].corr()

plt.figure(figsize=(10, 8))
plt.imshow(corr, aspect="auto")
plt.colorbar()
```

```
plt.xticks(range(len(corr_vars)), corr_vars, rotation=90)
plt.yticks(range(len(corr_vars)), corr_vars)
plt.title("Correlation matrix (selected variables)")
plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "corr_selected.png"), dpi=200)
plt.show()
```


Correlation matrix (selected variables)
circular_design_index
digital_capability_index
credibility_index
revenue_growth_pct_2y
revenue_musd
carbon_reduction_pct
waste_diversion_pct
material_circularity_index_0_1
avg_sentiment_score_-1_1
avg_rating_5
log_social_mentions_90d
log_review_count
log_news_mentions_90d
log_funding_raised_musd
circular_design_index
gital_capability_index
credibility_index
venue_growth_pct_2y
revenue_musd
carbon_reduction_pct
waste_diversion_pct
circularity_index_0_1
sentiment_score_-1_1
avg_rating_5
_social_mentions_90d
log_review_count
_news_mentions_90d
funding_raised_musd
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

In [176…

```
# ---------- 8) Regression models ----------
# --- Model 1: Carbon reduction by circular + digital + credibility + controls
m1 = smf.ols(
    "carbon_reduction_pct ~ circular_design_index + digital_capability_index + credibility_index "
    "+ log_employee_count + log_revenue_musd + C(business_model_type) + C(target_segment) + C(region)",
    data=df
).fit(cov_type="HC3")
print("\nMODEL 1: Carbon reduction")
print(m1.summary())
with open(os.path.join(OUT_DIR, "model1_carbon_reduction.txt"), "w") as f:
    f.write(m1.summary().as_text())
```

```
MODEL 1: Carbon reduction
                            OLS Regression Results
==============================================================================
Dep. Variable:     carbon_reduction_pct   R-squared:                       0.576
Model:                              OLS   Adj. R-squared:                  0.504
Method:                   Least Squares   F-statistic:                     8.178
Date:                  Tue, 06 Jan 2026   Prob (F-statistic):           5.51e-10
Time:                          18:14:42   Log-Likelihood:                -297.61
No. Observations:                    90   AIC:                             623.2
Df Residuals:                        76   BIC:                             658.2
Df Model:                            13
Covariance Type:                    HC3
=============================================================================================================
                                                coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------------------------------
Intercept                                    21.9262      9.966      2.200      0.028       2.394      41.459
C(business_model_type)[T.rental]              2.6141      3.455      0.757      0.449      -4.158       9.386
C(business_model_type)[T.repair]              7.1518      3.380      2.116      0.034       0.528      13.776
C(business_model_type)[T.resale]             -0.6154      3.631     -0.170      0.865      -7.731       6.500
C(business_model_type)[T.subscription]        1.0603      3.632      0.292      0.770      -6.059       8.180
C(target_segment)[T.mass]                    -3.8963      3.621     -1.076      0.282     -10.993       3.200
C(target_segment)[T.premium]                 -4.1746      3.607     -1.157      0.247     -11.245       2.896
C(target_segment)[T.value]                  -10.3907      4.452     -2.334      0.020     -19.117      -1.664
C(region)[T.north america]                    1.1463      2.369      0.484      0.629      -3.497       5.790
circular_design_index                        25.5328      3.675      6.948      0.000      18.330      32.736
digital_capability_index                      1.1923      4.384      0.272      0.786      -7.400       9.784
credibility_index                            -8.3508      5.548     -1.505      0.132     -19.224       2.523
log_employee_count                           -0.1720      2.693     -0.064      0.949      -5.450       5.106
log_revenue_musd                             -0.9787      2.399     -0.408      0.683      -5.681       3.724
==============================================================================
Omnibus:                        3.113   Durbin-Watson:                   1.940
Prob(Omnibus):                  0.211   Jarque-Bera (JB):                2.614
Skew:                           0.412   Prob(JB):                        0.271
Kurtosis:                       3.140   Cond. No.                         88.7
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

In [177…

```
# --- Model 2: Material circularity
m2 = smf.ols(
```

```
    "material_circularity_index_0_1 ~ circular_design_index + digital_capability_index + credibility_index "
    "+ log_employee_count + log_revenue_musd + C(business_model_type) + C(target_segment) + C(region)",
    data=df
).fit(cov_type="HC3")
print("\nMODEL 2: Material circularity")
print(m2.summary())
with open(os.path.join(OUT_DIR, "model2_material_circularity.txt"), "w") as f:
    f.write(m2.summary().as_text())
```

```
MODEL 2: Material circularity
                                OLS Regression Results
==============================================================================================
Dep. Variable:     material_circularity_index_0_1   R-squared:                       0.619
Model:                                        OLS   Adj. R-squared:                  0.554
Method:                             Least Squares   F-statistic:                     9.306
Date:                            Tue, 06 Jan 2026   Prob (F-statistic):           3.82e-11
Time:                                    18:14:42   Log-Likelihood:                 82.622
No. Observations:                              90   AIC:                            -137.2
Df Residuals:                                  76   BIC:                            -102.2
Df Model:                                      13
Covariance Type:                              HC3
===============================================================================================
                                          coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
Intercept                                0.1793      0.155      1.153      0.249      -0.125       0.484
C(business_model_type)[T.rental]         0.0159      0.056      0.286      0.775      -0.093       0.125
C(business_model_type)[T.repair]        -0.0508      0.054     -0.949      0.343      -0.156       0.054
C(business_model_type)[T.resale]         0.0091      0.059      0.154      0.878      -0.106       0.124
C(business_model_type)[T.subscription]   0.0401      0.057      0.698      0.485      -0.073       0.153
C(target_segment)[T.mass]                0.0225      0.042      0.531      0.595      -0.061       0.106
C(target_segment)[T.premium]             0.0400      0.042      0.961      0.336      -0.042       0.121
C(target_segment)[T.value]               0.0260      0.074      0.351      0.726      -0.120       0.172
C(region)[T.north america]                0.0612      0.046      1.329      0.184      -0.029       0.152
circular_design_index                    0.4459      0.057      7.802      0.000       0.334       0.558
digital_capability_index                -0.1054      0.063     -1.681      0.093      -0.228       0.017
credibility_index                       -0.0424      0.081     -0.523      0.601      -0.201       0.116
log_employee_count                       0.0186      0.042      0.447      0.655      -0.063       0.100
log_revenue_musd                        -0.0032      0.037     -0.085      0.932      -0.076       0.070
==============================================================================
Omnibus:                        0.311   Durbin-Watson:                   1.791
Prob(Omnibus):                  0.856   Jarque-Bera (JB):                0.245
Skew:                           0.125   Prob(JB):                        0.885
Kurtosis:                       2.944   Cond. No.                         88.7
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

In [178…

```
# --- Model 3: Revenue growth by circular + digital + sentiment + controls
df["avg_sentiment_score"] = df["avg_sentiment_score_-1_1"]
```

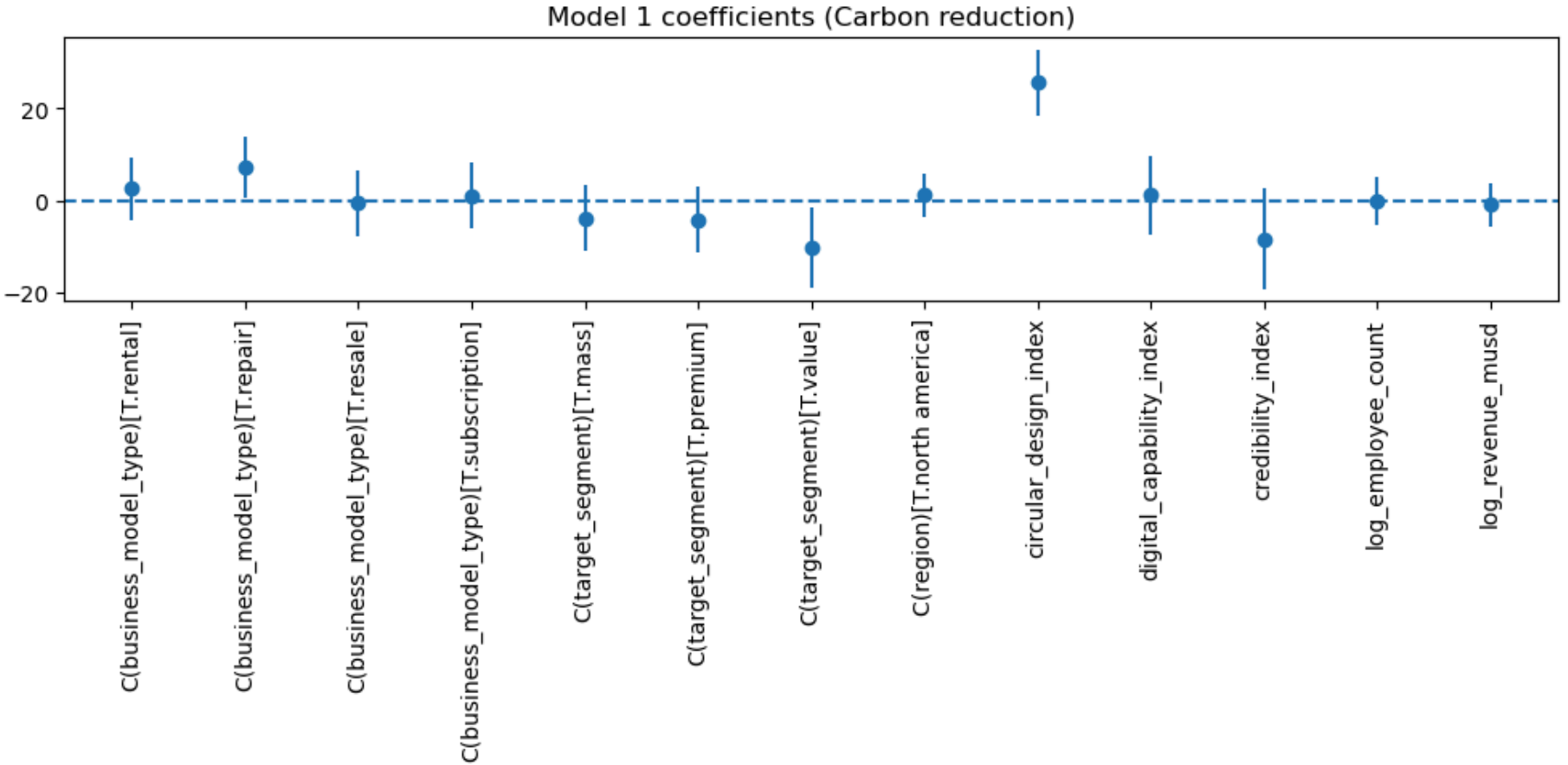
```
m3 = smf.ols(
    "revenue_growth_pct_2y ~ circular_design_index + digital_capability_index "
    "+ avg_sentiment_score + credibility_index "
    "+ log_employee_count + log_funding_raised_musd "
    "+ C(business_model_type) + C(target_segment) + C(region)",
    data=df
).fit(cov_type="HC3")

print(m3.summary())
```

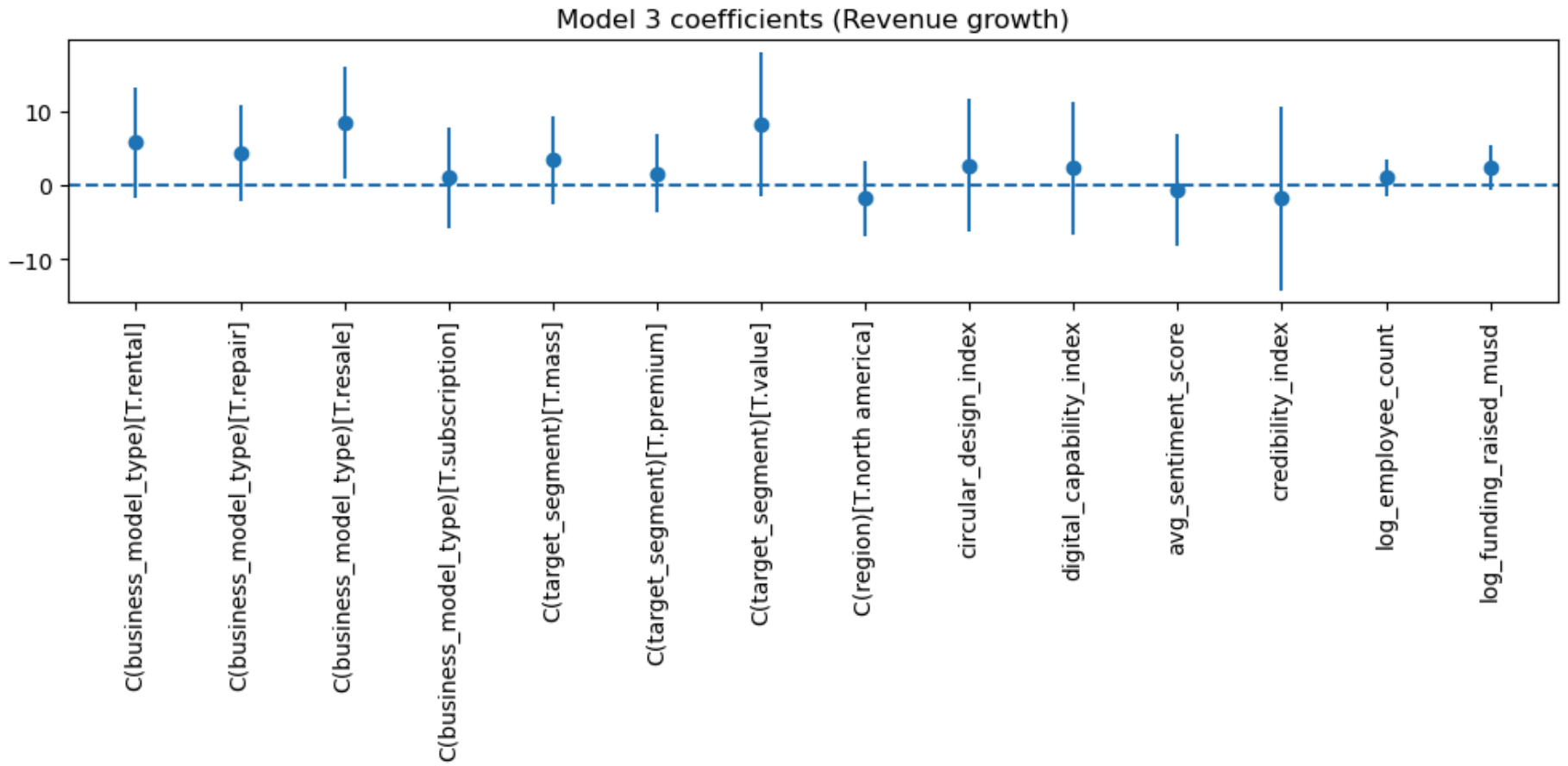
```
                            OLS Regression Results
==============================================================================
Dep. Variable:     revenue_growth_pct_2y   R-squared:                       0.207
Model:                               OLS   Adj. R-squared:                  0.058
Method:                    Least Squares   F-statistic:                     1.334
Date:                   Tue, 06 Jan 2026   Prob (F-statistic):              0.208
Time:                           18:14:42   Log-Likelihood:                -306.23
No. Observations:                     90   AIC:                             642.5
Df Residuals:                         75   BIC:                             680.0
Df Model:                             14
Covariance Type:                     HC3
==============================================================================================================
                                                coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------------------------------------
Intercept                                    -2.7883     11.807     -0.236      0.813     -25.930      20.354
C(business_model_type)[T.rental]              5.7459      3.765      1.526      0.127      -1.634      13.126
C(business_model_type)[T.repair]              4.3249      3.286      1.316      0.188      -2.115      10.765
C(business_model_type)[T.resale]              8.4808      3.851      2.202      0.028       0.932      16.029
C(business_model_type)[T.subscription]        0.9940      3.501      0.284      0.777      -5.869       7.857
C(target_segment)[T.mass]                     3.3580      2.999      1.120      0.263      -2.519       9.235
C(target_segment)[T.premium]                  1.5648      2.694      0.581      0.561      -3.715       6.845
C(target_segment)[T.value]                    8.1591      4.985      1.637      0.102      -1.611      17.929
C(region)[T.north america]                   -1.7849      2.585     -0.690      0.490      -6.852       3.282
circular_design_index                         2.6524      4.591      0.578      0.563      -6.346      11.650
digital_capability_index                      2.2817      4.580      0.498      0.618      -6.695      11.258
avg_sentiment_score                          -0.6913      3.850     -0.180      0.857      -8.237       6.854
credibility_index                            -1.7697      6.326     -0.280      0.780     -14.168      10.629
log_employee_count                            1.0482      1.263      0.830      0.407      -1.427       3.524
log_funding_raised_musd                       2.3826      1.547      1.540      0.124      -0.650       5.415
==============================================================================
Omnibus:                        2.352   Durbin-Watson:                   1.839
Prob(Omnibus):                  0.308   Jarque-Bera (JB):                1.633
Skew:                           0.101   Prob(JB):                        0.442
Kurtosis:                       2.372   Cond. No.                         91.1
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

In [179…

```python
def coef_plot(model, title, fname):
    params = model.params.drop("Intercept", errors="ignore")
```

```
    conf = model.conf_int().loc[params.index]
    x = np.arange(len(params))
    plt.figure(figsize=(10, 5))
    plt.errorbar(x, params.values, yerr=[params.values-conf[0].values, conf[1].values-params.values], fmt="o")
    plt.axhline(0, linestyle="--")
    plt.xticks(x, params.index, rotation=90)
    plt.title(title)
    plt.tight_layout()
    plt.savefig(os.path.join(OUT_DIR, fname), dpi=200)
    plt.show()

coef_plot(m1, "Model 1 coefficients (Carbon reduction)", "coef_model1.png")
coef_plot(m3, "Model 3 coefficients (Revenue growth)", "coef_model3.png")
```





In [180…

```
# ---------- 9) Network analysis (Graph + Centralities + Communities) ----------
# Build graph from edge list
G = nx.Graph()
# Add nodes
for cid in df["company_id"]:
    G.add_node(cid)

for _, e in edges.iterrows():
    G.add_edge(
        e["source_company_id"],
        e["target_company_id"],
        relationship_type=e["relationship_type"],
        weight=float(e["edge_weight"])
```

```
    )

print("\nGraph nodes:", G.number_of_nodes())
print("Graph edges:", G.number_of_edges())
```

```
Graph nodes: 90
Graph edges: 214
```

In [181…

```
# Centralities
deg = dict(G.degree())
wdeg = dict(G.degree(weight="weight"))
bet = nx.betweenness_centrality(G, weight="weight", normalized=True)
clo = nx.closeness_centrality(G)
eig = nx.eigenvector_centrality_numpy(G, weight="weight")

# Communities
communities = list(nx.algorithms.community.greedy_modularity_communities(G, weight="weight"))
community_map = {}
for i, comm in enumerate(communities):
    for node in comm:
        community_map[node] = i

# Adding into dataframe
net = pd.DataFrame({
    "company_id": list(G.nodes()),
    "degree": [deg.get(n, 0) for n in G.nodes()],
    "weighted_degree": [wdeg.get(n, 0) for n in G.nodes()],
    "betweenness": [bet.get(n, 0.0) for n in G.nodes()],
    "closeness": [clo.get(n, 0.0) for n in G.nodes()],
    "eigenvector": [eig.get(n, 0.0) for n in G.nodes()],
    "community_id": [community_map.get(n, -1) for n in G.nodes()]
})
net.to_csv(os.path.join(OUT_DIR, "network_metrics.csv"), index=False)
print("Saved network_metrics.csv")
```

```
Saved network_metrics.csv
```

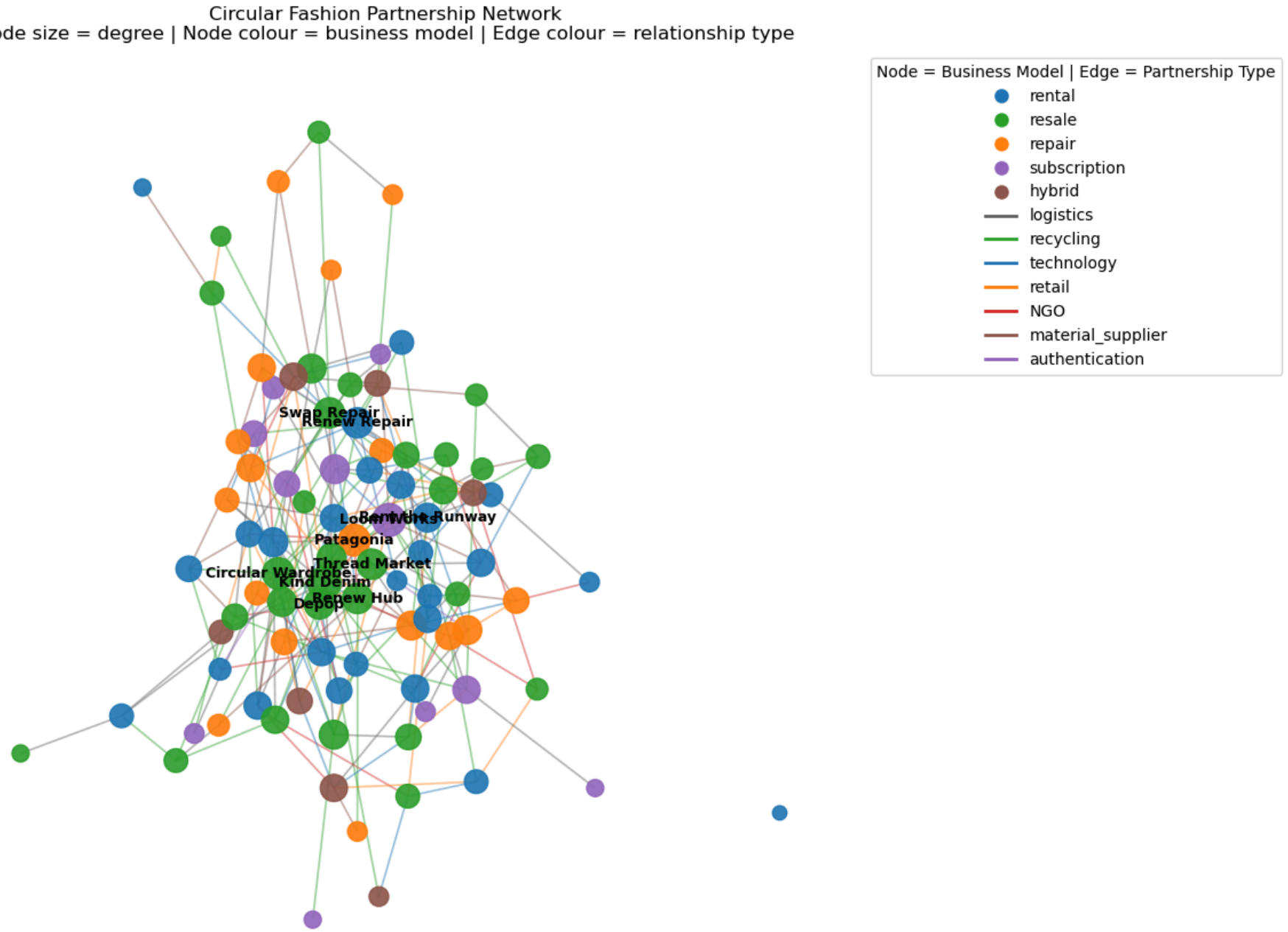
In [182…

```
import matplotlib.pyplot as plt
import networkx as nx

plt.figure(figsize=(12, 9))
```

```
# Layout
pos = nx.spring_layout(G, seed=42, k=0.35)

model_colors = {
    "rental": "#1f77b4",
    "resale": "#2ca02c",
    "repair": "#ff7f0e",
    "subscription": "#9467bd",
    "hybrid": "#8c564b"
}

node_colors = [
    model_colors.get(
        df2.set_index("company_id").loc[n, "business_model_type"],
        "#7f7f7f"
    )
    for n in G.nodes()
]

# Node sizes = degree
node_sizes = [
    80 + 35 * net.set_index("company_id").loc[n, "degree"]
    for n in G.nodes()
]

nx.draw_networkx_nodes(
    G,
    pos,
    node_color=node_colors,
    node_size=node_sizes,
    alpha=0.9
)


edge_color_map = {
    "logistics": "#636363",
    "recycling": "#2ca02c",
    "technology": "#1f77b4",
    "retail": "#ff7f0e",
    "NGO": "#d62728",
```

```
    "material_supplier": "#8c564b",
    "authentication": "#9467bd"
}

edge_colors = []
for u, v, d in G.edges(data=True):
    edge_colors.append(edge_color_map.get(d.get("relationship_type", ""), "#bdbdbd"))

nx.draw_networkx_edges(
    G,
    pos,
    edge_color=edge_colors,
    alpha=0.45,
    width=1.2
)


top_nodes = (
    net.sort_values("degree", ascending=False)
       .head(10)["company_id"]
       .tolist()
)

labels = {
    n: df2.set_index("company_id").loc[n, "company_name"]
    for n in top_nodes
}

nx.draw_networkx_labels(
    G,
    pos,
    labels=labels,
    font_size=9,
    font_weight="bold"
)


# Node legend (business models)
node_legend = [
    plt.Line2D([0], [0], marker='o', color='w',
               label=label,
```

```
                markerfacecolor=color,
                markersize=10)
    for label, color in model_colors.items()
]

# Edge legend (relationships)
edge_legend = [
    plt.Line2D([0], [0], color=color, lw=2, label=label)
    for label, color in edge_color_map.items()
]

plt.legend(
    handles=node_legend + edge_legend,
    title="Node = Business Model | Edge = Partnership Type",
    bbox_to_anchor=(1.05, 1),
    loc="upper left"
)

plt.title("Circular Fashion Partnership Network\nNode size = degree | Node colour = business model | Edge colour =
plt.axis("off")
plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "network_graph_coloured_labeled.png"), dpi=220)
plt.show()
```

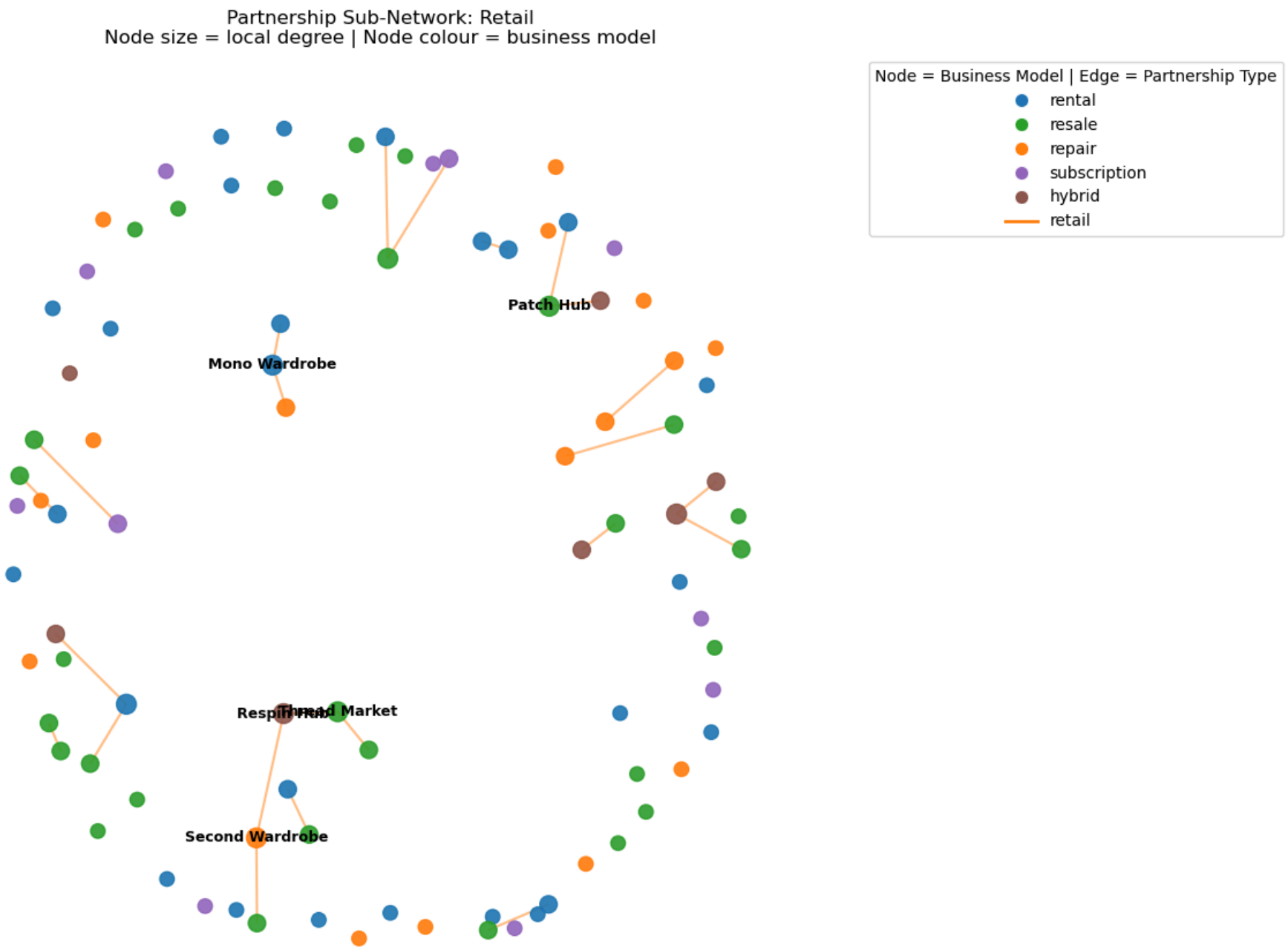

Circular Fashion Partnership Network
Node size = degree | Node colour = business model | Edge colour = relationship type
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
logistics
recycling
technology
retail
NGO
material_supplier
authentication
Swap Repair
Renew Repair
Patagonia
Thread Market
Circular Wardrobe
Kind Denim
Renew Hub
Depop

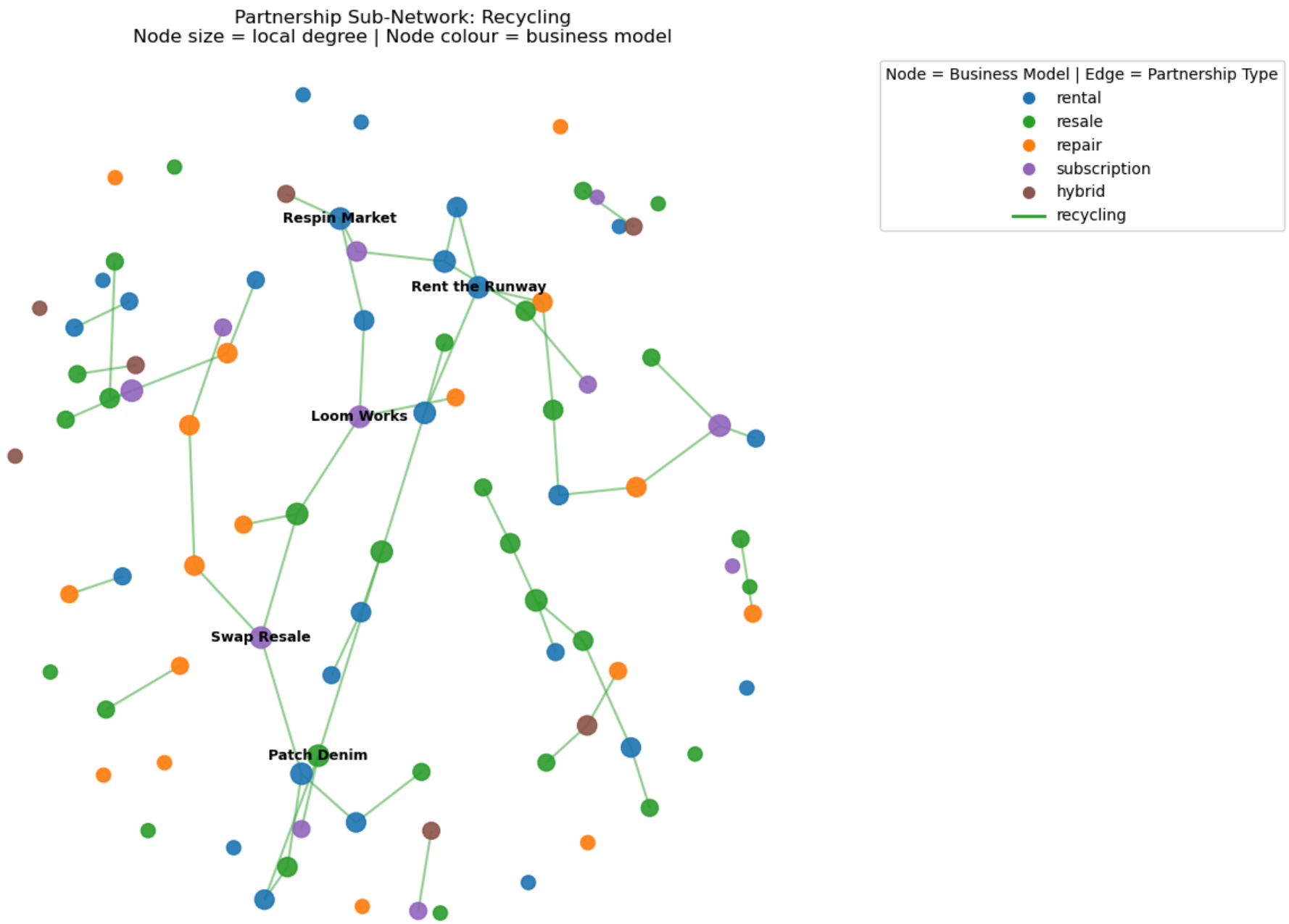
In [183…

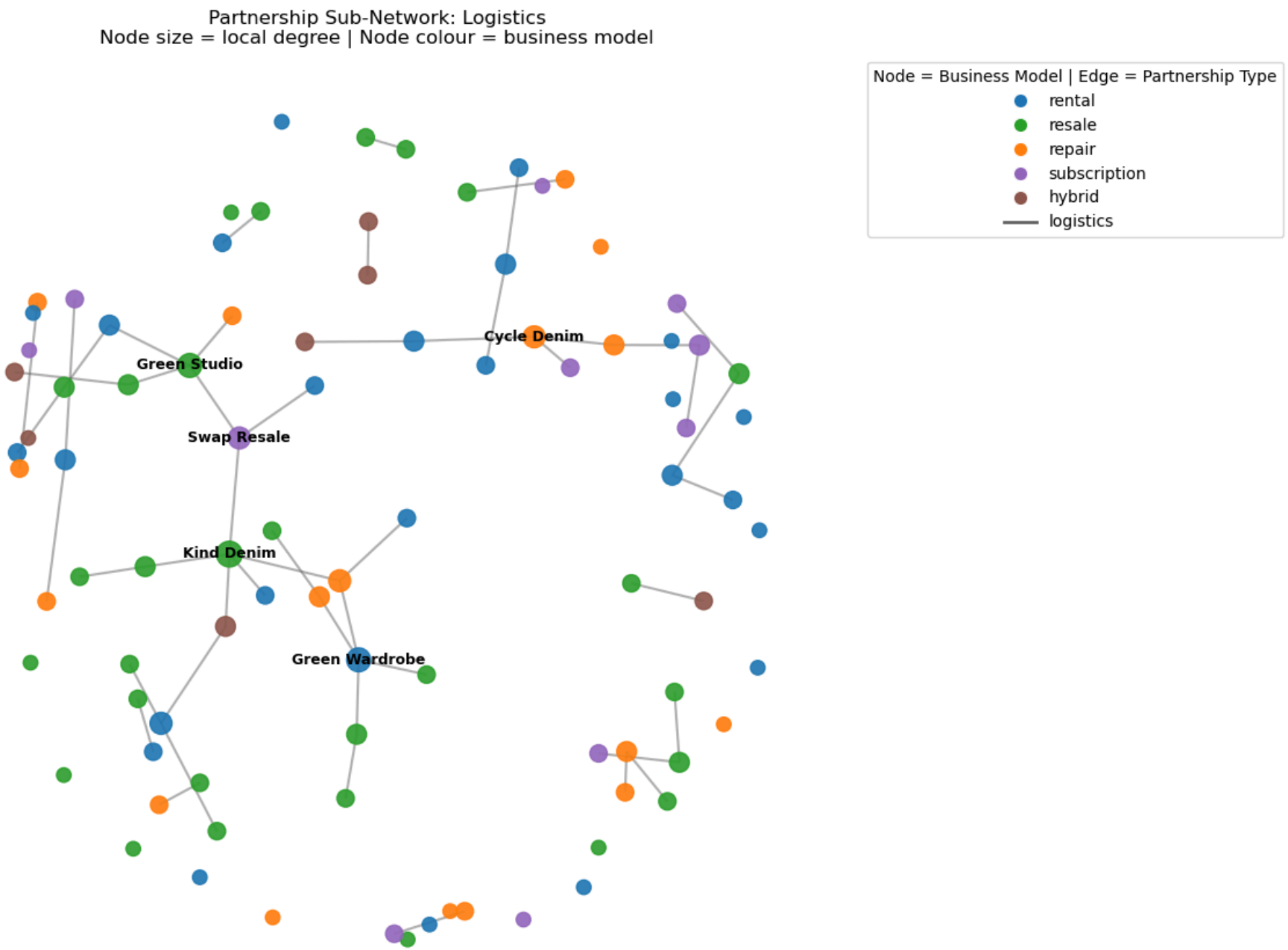
```python
import matplotlib.pyplot as plt
import networkx as nx

# Colour maps
model_colors = {
    "rental": "#1f77b4",
    "resale": "#2ca02c",
    "repair": "#ff7f0e",
    "subscription": "#9467bd",
    "hybrid": "#8c564b"
}

edge_color_map = {
    "logistics": "#636363",
    "recycling": "#2ca02c",
    "technology": "#1f77b4",
    "retail": "#ff7f0e",
    "NGO": "#d62728",
    "material_supplier": "#8c564b",
    "authentication": "#9467bd"
}


for rel in edges["relationship_type"].unique():

    sub_edges = edges[edges["relationship_type"] == rel]

    # Build subgraph
    H = nx.Graph()
    for cid in df["company_id"]:
        H.add_node(cid)

    for _, e in sub_edges.iterrows():
        H.add_edge(
            e["source_company_id"],
            e["target_company_id"],
            weight=float(e["edge_weight"])
        )

    # Layout
```

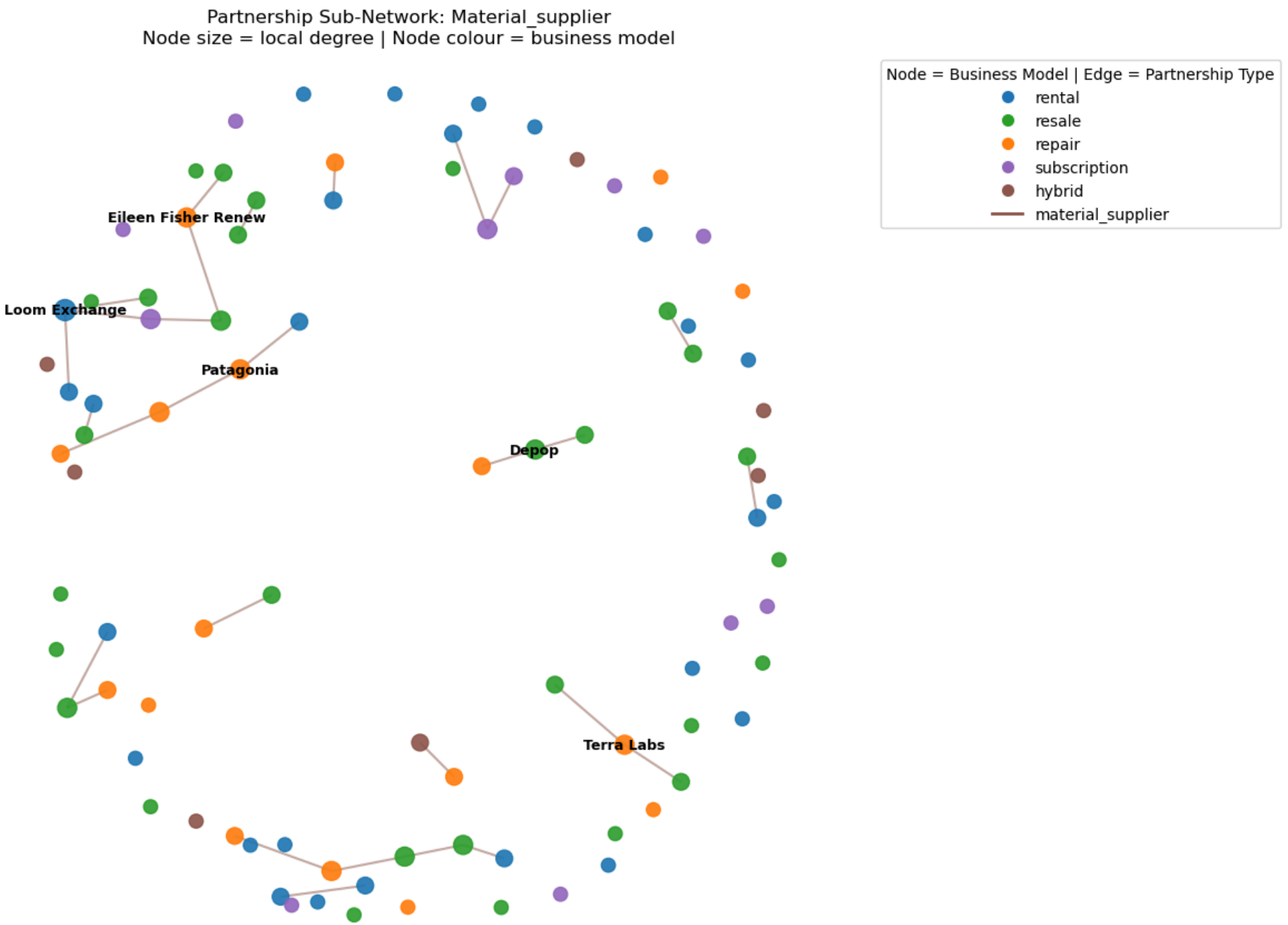
```
posH = nx.spring_layout(H, seed=42, k=0.4)

# Node sizes = degree within this sub-network
deg_H = dict(H.degree())
node_sizes = [80 + 35 * deg_H.get(n, 0) for n in H.nodes()]

# Node colours = business model
node_colors = [
    model_colors.get(
        df2.set_index("company_id").loc[n, "business_model_type"],
        "#7f7f7f"
    )
    for n in H.nodes()
]

plt.figure(figsize=(12, 9))

nx.draw_networkx_nodes(
    H,
    posH,
    node_color=node_colors,
    node_size=node_sizes,
    alpha=0.9
)

nx.draw_networkx_edges(
    H,
    posH,
    edge_color=edge_color_map.get(rel, "#bdbdbd"),
    alpha=0.5,
    width=1.5
)




top_nodes_H = sorted(deg_H, key=deg_H.get, reverse=True)[:5]

labels = {
    n: df2.set_index("company_id").loc[n, "company_name"]
    for n in top_nodes_H if deg_H.get(n, 0) > 0
```

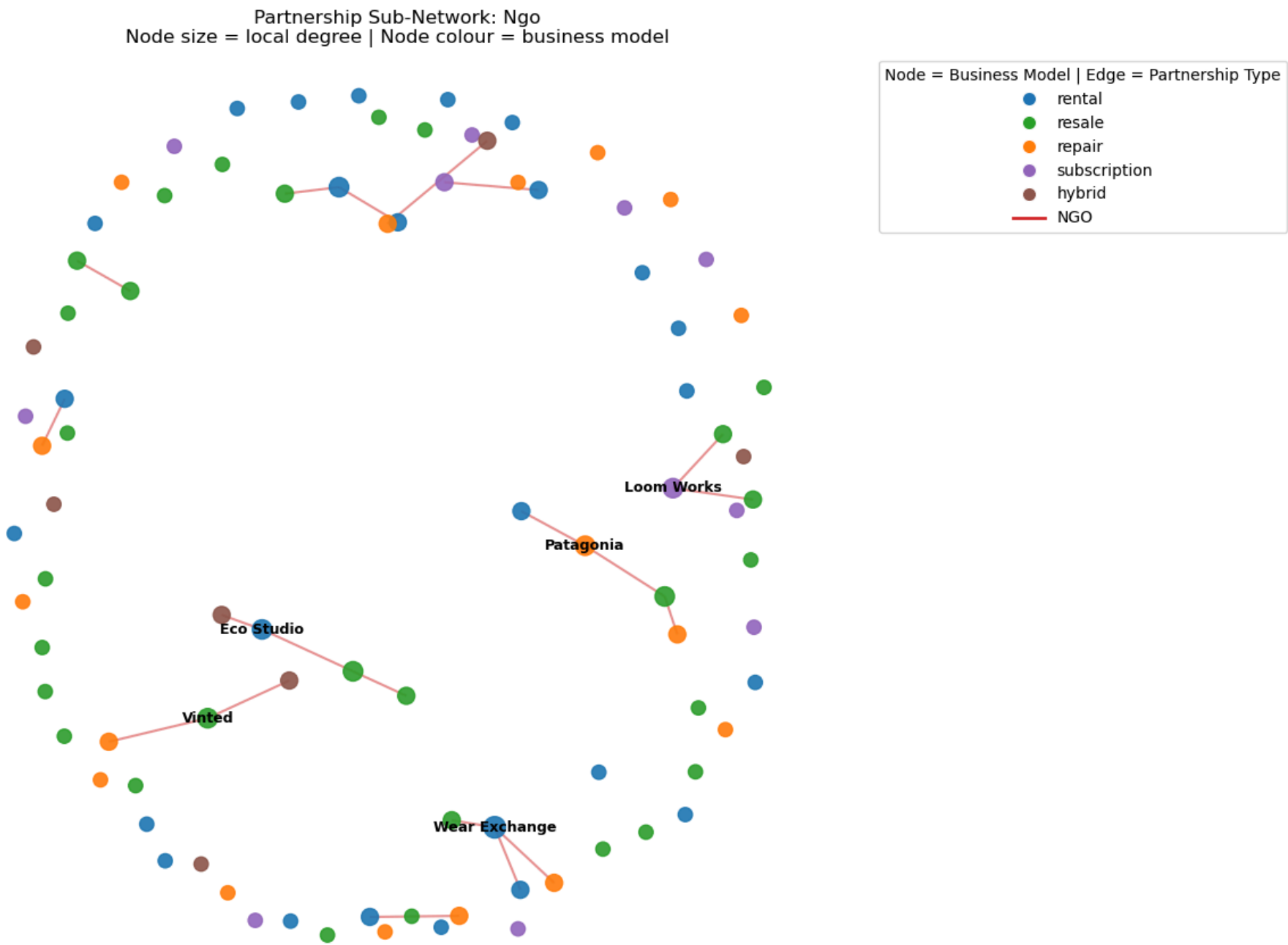
```
    }

    nx.draw_networkx_labels(
        H,
        posH,
        labels=labels,
        font_size=9,
        font_weight="bold"
    )


    node_legend = [
        plt.Line2D([0], [0], marker='o', color='w',
                   label=label,
                   markerfacecolor=color,
                   markersize=9)
        for label, color in model_colors.items()
    ]

    edge_legend = [
        plt.Line2D([0], [0], color=edge_color_map.get(rel, "#bdbdbd"),
                   lw=2, label=rel)
    ]

    plt.legend(
        handles=node_legend + edge_legend,
        title="Node = Business Model | Edge = Partnership Type",
        bbox_to_anchor=(1.05, 1),
        loc="upper left"
    )

    plt.title(
        f"Partnership Sub-Network: {rel.capitalize()}\n"
        "Node size = local degree | Node colour = business model"
    )

    plt.axis("off")
    plt.tight_layout()
    plt.savefig(os.path.join(OUT_DIR, f"network_sub_{rel}_coloured.png"), dpi=220)
    plt.show()
```

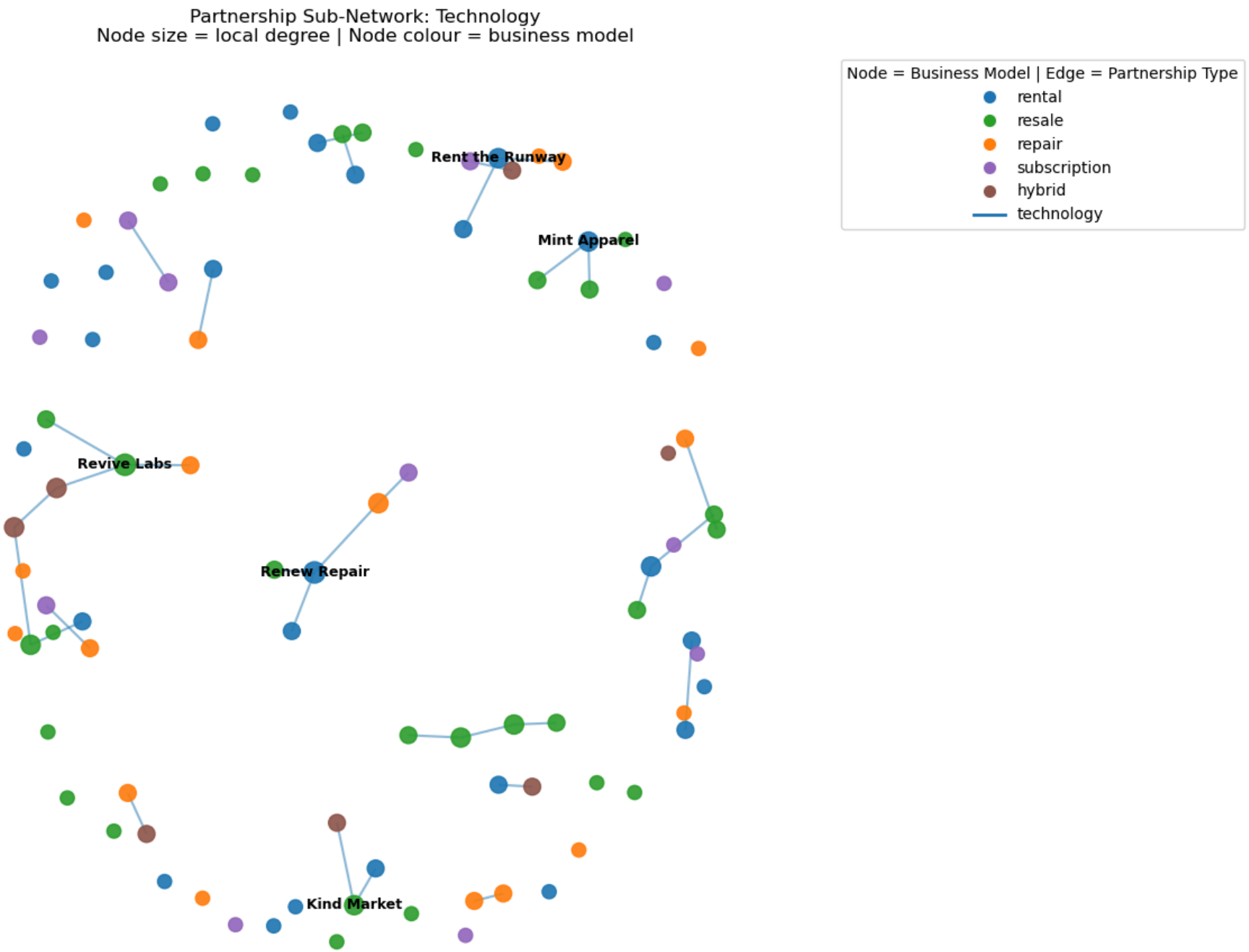

Partnership Sub-Network: Retail
Node size = local degree | Node colour = business model
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
retail
Patch Hub
Mono Wardrobe
Second Wardrobe

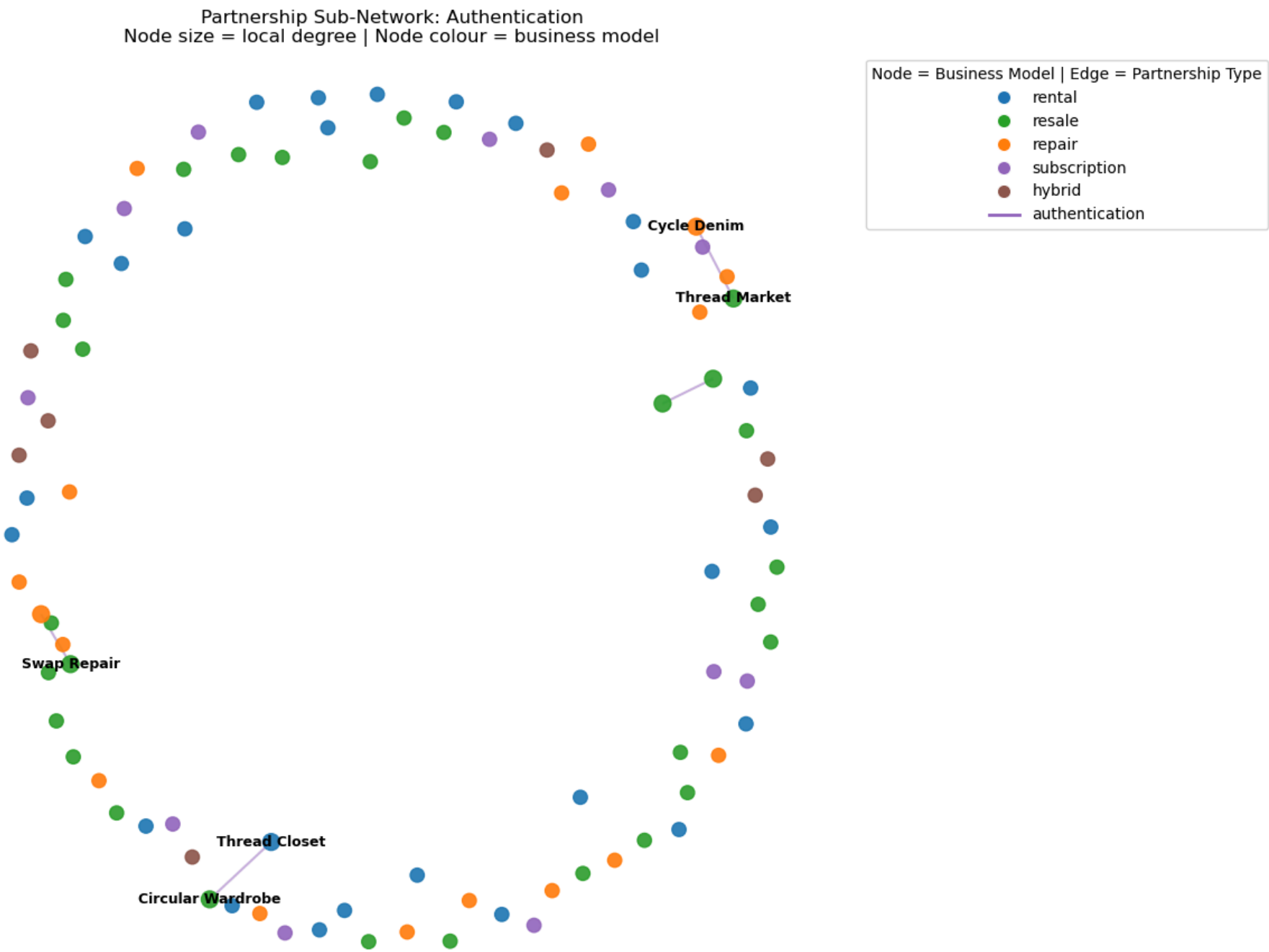

Partnership Sub-Network: Recycling
Node size = local degree | Node colour = business model
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
recycling
Respin Market
Rent the Runway
Loom Works
Swap Resale
Patch Denim


Partnership Sub-Network: Logistics
Node size = local degree | Node colour = business model
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
logistics
Green Studio
Cycle Denim
Swap Resale
Kind Denim
Green Wardrobe


Partnership Sub-Network: Material_supplier
Node size = local degree | Node colour = business model
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
material_supplier
Eileen Fisher Renew
Loom Exchange
Patagonia
Depop
Terra Labs


Partnership Sub-Network: Ngo
Node size = local degree | Node colour = business model
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
NGO
Loom Works
Patagonia
Eco Studio
Vinted
Wear Exchange


Partnership Sub-Network: Technology
Node size = local degree | Node colour = business model
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
technology
Rent the Runway
Mint Apparel
Revive Labs
Renew Repair
Kind Market


Partnership Sub-Network: Authentication
Node size = local degree | Node colour = business model
Node = Business Model | Edge = Partnership Type
rental
resale
repair
subscription
hybrid
authentication
Cycle Denim
Thread Market
Swap Repair
Thread Closet
Circular Wardrobe

In [184…
```
# ---------- 10) Merge network metrics back to firm dataset ----------
df2 = df.merge(net, on="company_id", how="left")

# Network feature engineering
df2["log_degree"] = np.log1p(df2["degree"])
df2["log_weighted_degree"] = np.log1p(df2["weighted_degree"])

df2.to_csv(os.path.join(OUT_DIR, "companies_with_network_metrics.csv"), index=False)
print("\nSaved merged dataset: companies_with_network_metrics.csv")
```

```
Saved merged dataset: companies_with_network_metrics.csv
```

In [185…
```
# ---------- 11) Regression including network features ----------
df2["avg_sentiment_score"] = df2["avg_sentiment_score_-1_1"]
m4 = smf.ols(
    "revenue_growth_pct_2y ~ circular_design_index + digital_capability_index "
    "+ avg_sentiment_score + credibility_index "
    "+ log_weighted_degree + betweenness + C(community_id) "
    "+ log_employee_count + log_funding_raised_musd "
    "+ C(business_model_type) + C(region)",
    data=df2
).fit(cov_type="HC3")

print(m4.summary())
with open(os.path.join(OUT_DIR, "model4_revenue_growth_network.txt"), "w") as f:
    f.write(m4.summary().as_text())

coef_plot(m4, "Model 4 coefficients (Revenue growth + network)", "coef_model4.png")
```

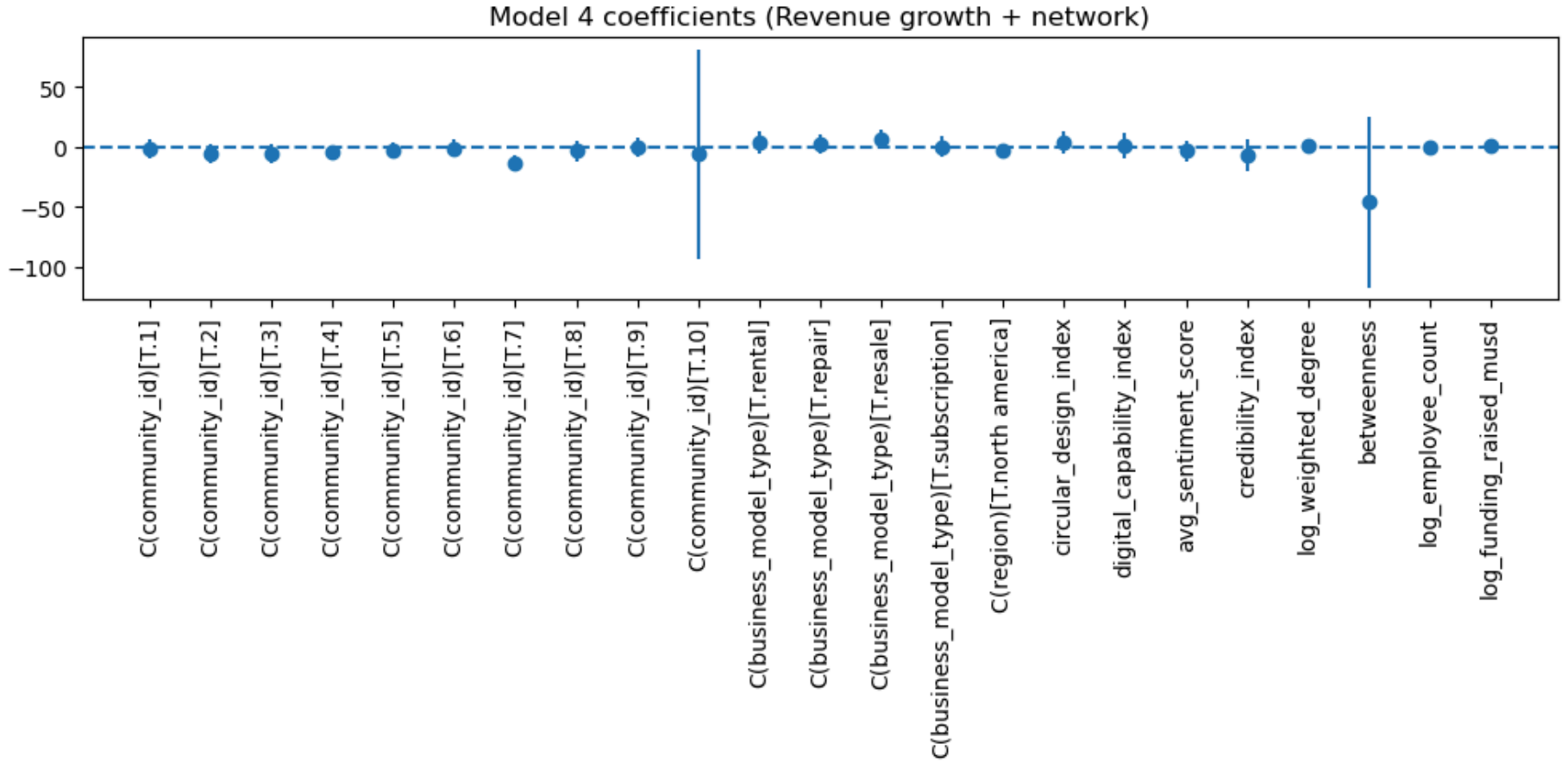
OLS Regression Results

Dep. Variable:	revenue_growth_pct_2y	R-squared:	0.359
Model:	OLS	Adj. R-squared:	0.135
Method:	Least Squares	F-statistic:	2.179
Date:	Tue, 06 Jan 2026	Prob (F-statistic):	0.00735
Time:	18:14:46	Log-Likelihood:	-296.65
No. Observations:	90	AIC:	641.3
Df Residuals:	66	BIC:	701.3
Df Model:	23		
Covariance Type:	HC3		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	17.2541	12.529	1.377	0.168	-7.302	41.810
C(community_id)[T.1]	-1.6589	4.076	-0.407	0.684	-9.648	6.330
C(community_id)[T.2]	-5.8375	3.977	-1.468	0.142	-13.631	1.956
C(community_id)[T.3]	-5.3944	4.097	-1.317	0.188	-13.424	2.636
C(community_id)[T.4]	-3.6739	2.717	-1.352	0.176	-8.999	1.651
C(community_id)[T.5]	-3.0001	3.588	-0.836	0.403	-10.032	4.032
C(community_id)[T.6]	-1.3681	3.722	-0.368	0.713	-8.663	5.927
C(community_id)[T.7]	-13.8478	3.400	-4.073	0.000	-20.511	-7.184
C(community_id)[T.8]	-3.3203	4.586	-0.724	0.469	-12.309	5.668
C(community_id)[T.9]	-0.1862	4.370	-0.043	0.966	-8.751	8.378
C(community_id)[T.10]	-5.9611	44.692	-0.133	0.894	-93.556	81.634
C(business_model_type)[T.rental]	3.6636	4.657	0.787	0.431	-5.464	12.792
C(business_model_type)[T.repair]	2.0992	4.223	0.497	0.619	-6.178	10.376
C(business_model_type)[T.resale]	6.6008	4.372	1.510	0.131	-1.968	15.170
C(business_model_type)[T.subscription]	-0.0445	4.407	-0.010	0.992	-8.682	8.593
C(region)[T.north america]	-3.0011	2.774	-1.082	0.279	-8.439	2.436
circular_design_index	3.4216	4.780	0.716	0.474	-5.946	12.790
digital_capability_index	0.7855	5.361	0.147	0.884	-9.722	11.293
avg_sentiment_score	-3.1879	4.361	-0.731	0.465	-11.735	5.359
credibility_index	-7.1016	6.795	-1.045	0.296	-20.420	6.217
log_weighted_degree	1.0278	2.804	0.366	0.714	-4.469	6.524
betweenness	-45.9847	36.457	-1.261	0.207	-117.440	25.470
log_employee_count	0.3494	0.856	0.408	0.683	-1.328	2.027
log_funding_raised_musd	0.6188	1.783	0.347	0.729	-2.877	4.114

Omnibus:	0.351	Durbin-Watson:	1.874
Prob(Omnibus):	0.839	Jarque-Bera (JB):	0.082

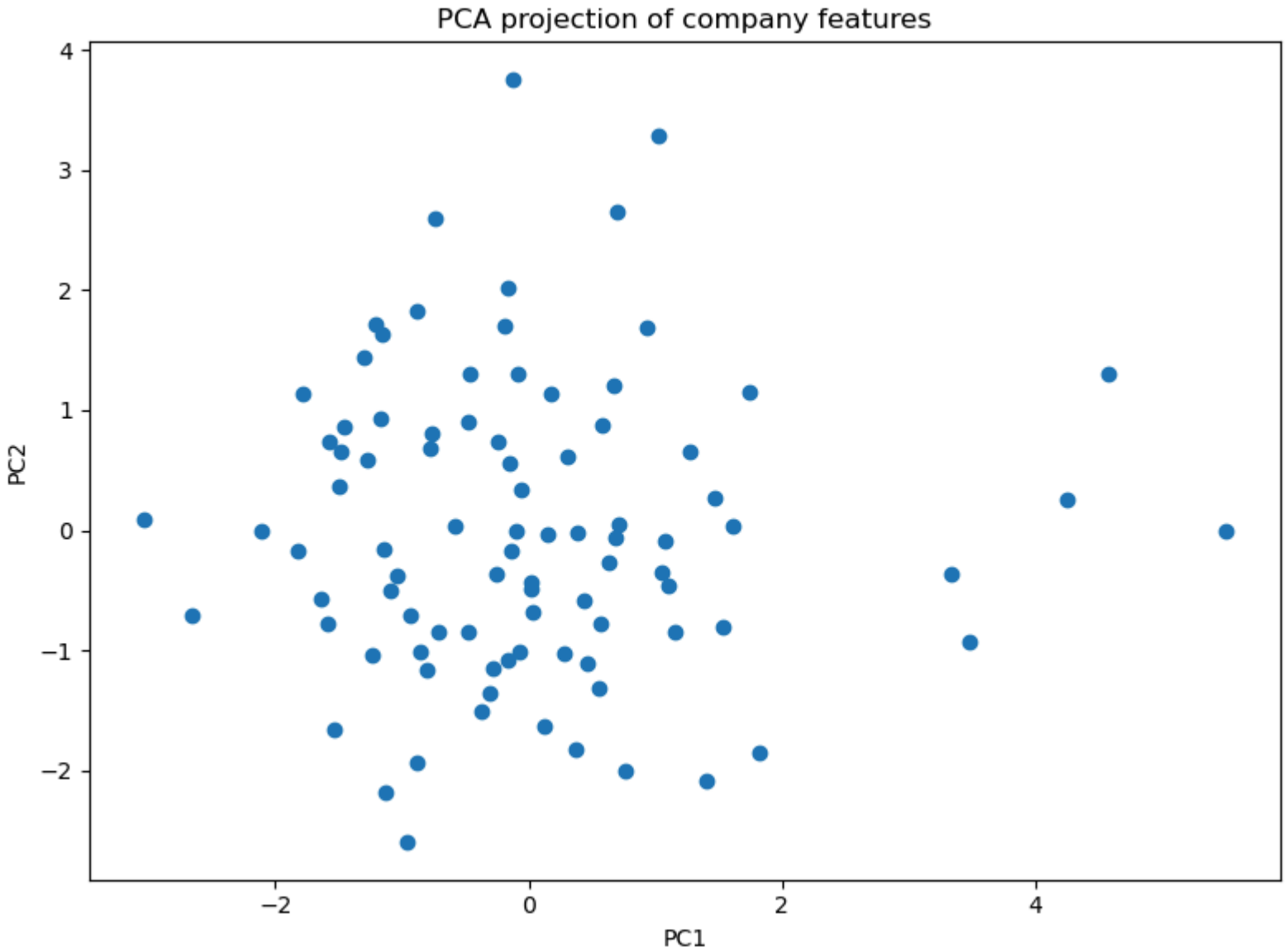
```
Skew:                          0.050   Prob(JB):                     0.960
Kurtosis:                      3.108   Cond. No.                      278.
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity robust (HC3)
```



In [186…

```python
# ---------- 12) Clustering + PCA ----------
cluster_features = [
    "circular_design_index","digital_capability_index","credibility_index",
    "log_revenue_musd","log_employee_count","log_funding_raised_musd",
    "avg_sentiment_score_-1_1","log_weighted_degree"
]
```

```
X = df2[cluster_features].values
X_scaled = StandardScaler().fit_transform(X)

# PCA for 2D visualization
pca = PCA(n_components=2, random_state=42)
Z = pca.fit_transform(X_scaled)
df2["pca1"] = Z[:, 0]
df2["pca2"] = Z[:, 1]

# KMeans (choose k=3 or k=4; pick based on your narrative)
k = 4
km = KMeans(n_clusters=k, random_state=42, n_init=20)
df2["cluster"] = km.fit_predict(X_scaled)

plt.figure(figsize=(8, 6))
plt.scatter(df2["pca1"], df2["pca2"])
plt.title("PCA projection of company features")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "pca_scatter.png"), dpi=200)
plt.show()

# Cluster summary table
cluster_summary = df2.groupby("cluster")[cluster_features + ["carbon_reduction_pct","material_circularity_index_0_1
cluster_summary.to_csv(os.path.join(OUT_DIR, "cluster_summary.csv"))
```

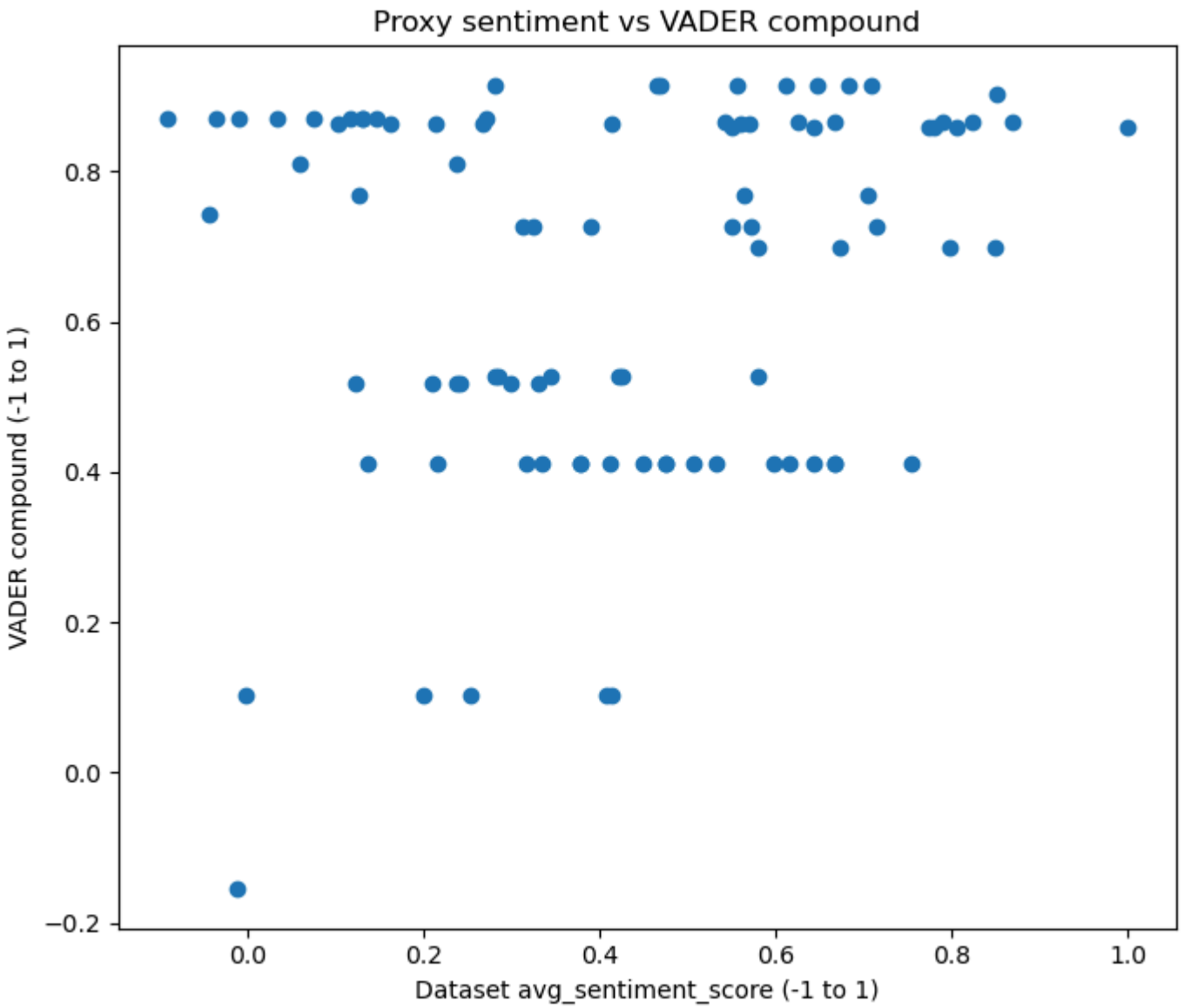


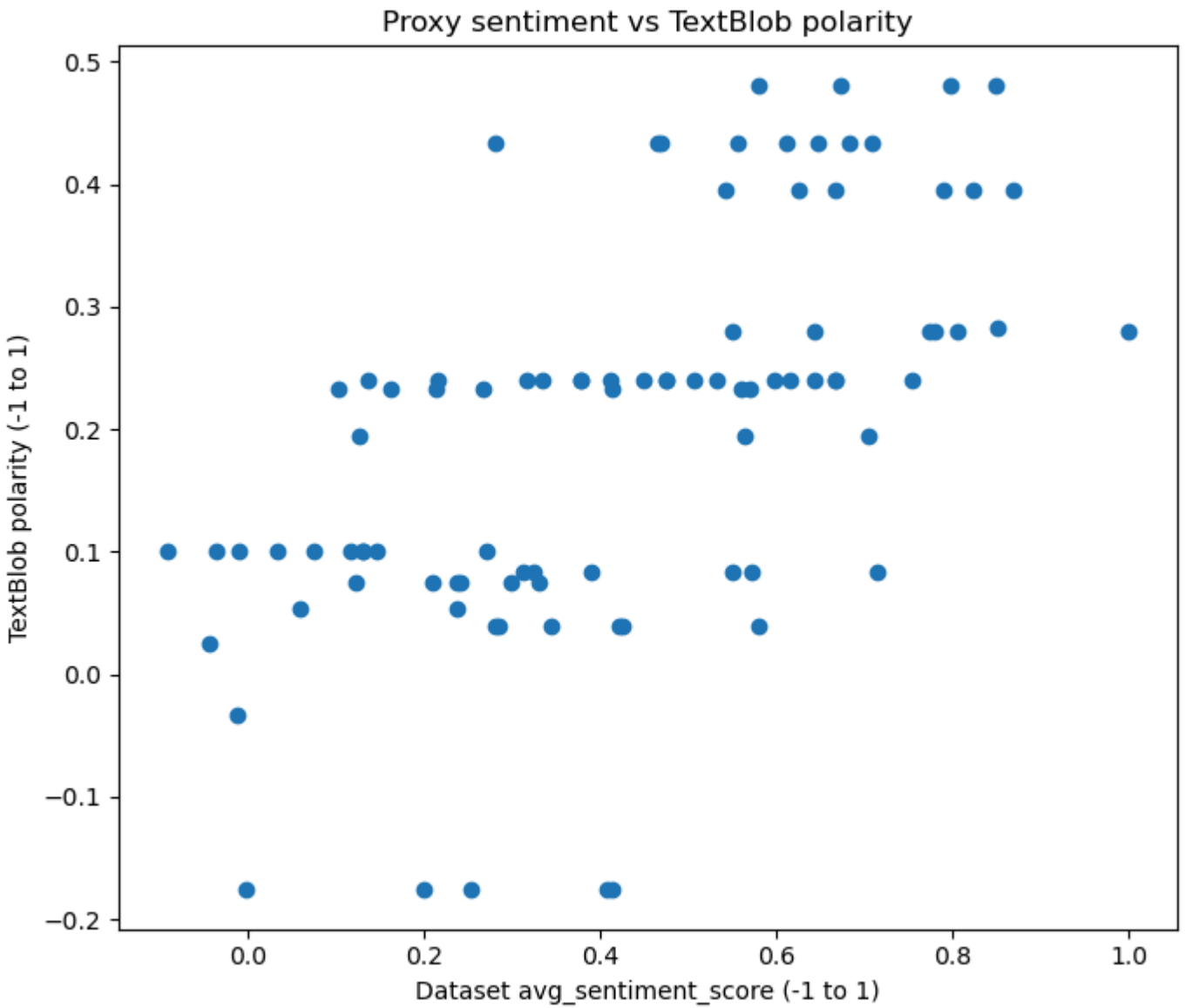
In [187…

```
# 13) Sentiment Analysis
# My dataset does NOT include raw text (tweets/reviews/articles).
# this section creates synthetic review-like text USING the existing numeric proxies
#   (avg_rating_5, business_model_type, avg_sentiment_score_-1_1).
# - Runs VADER + TextBlob on that generated text.
```

```
# - Compares computed sentiment scores against the dataset's proxy sentiment.

df_sent = df2.copy()

def synth_text(row):
    # Template bank based on rating + rental pain points
    model = row["business_model_type"]
    rating = row["avg_rating_5"]
    base = ""
    if rating >= 4.4:
        base = "Absolutely loved it. Great quality, smooth experience, and I will use it again."
    elif rating >= 4.0:
        base = "Good overall. Quality was solid and the process was convenient."
    elif rating >= 3.5:
        base = "Mixed experience. Some things worked well but there were a few issues."
    else:
        base = "Disappointing experience. Quality and service did not meet expectations."

    # Add model-specific cues (this matters for realism)
    if model == "rental":
        extra = " Delivery and returns were okay, but cleanliness and fit consistency could improve."
    elif model == "resale":
        extra = " Items were as described and pricing felt fair, but shipping speed varied."
    elif model == "repair":
        extra = " Repair service extended product life, though turnaround time was sometimes slow."
    elif model == "subscription":
        extra = " Subscription value was decent, but selection and swap frequency could be better."
    else:
        extra = " The mix of services is promising, though operations still feel uneven in places."


    sus = " I also appreciate the sustainability focus and reduced waste."

    return base + extra + sus

df_sent["text"] = df_sent.apply(synth_text, axis=1)

# VADER
analyzer = SentimentIntensityAnalyzer()
df_sent["vader_compound"] = df_sent["text"].apply(lambda t: analyzer.polarity_scores(t)["compound"])
```

```
# TextBlob (polarity -1 to 1)
df_sent["textblob_polarity"] = df_sent["text"].apply(lambda t: TextBlob(t).sentiment.polarity)

# Comparing with dataset proxy sentiment
plt.figure(figsize=(7, 6))
plt.scatter(df_sent["avg_sentiment_score_-1_1"], df_sent["vader_compound"])
plt.title("Proxy sentiment vs VADER compound")
plt.xlabel("Dataset avg_sentiment_score (-1 to 1)")
plt.ylabel("VADER compound (-1 to 1)")
plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "sentiment_proxy_vs_vader.png"), dpi=200)
plt.show()

plt.figure(figsize=(7, 6))
plt.scatter(df_sent["avg_sentiment_score_-1_1"], df_sent["textblob_polarity"])
plt.title("Proxy sentiment vs TextBlob polarity")
plt.xlabel("Dataset avg_sentiment_score (-1 to 1)")
plt.ylabel("TextBlob polarity (-1 to 1)")
plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "sentiment_proxy_vs_textblob.png"), dpi=200)
plt.show()

# Saving sentiment outputs
df_sent[["company_id","company_name","avg_sentiment_score_-1_1","vader_compound","textblob_polarity","text"]].to_cs
    os.path.join(OUT_DIR, "sentiment_scored_output.csv"), index=False
)
```

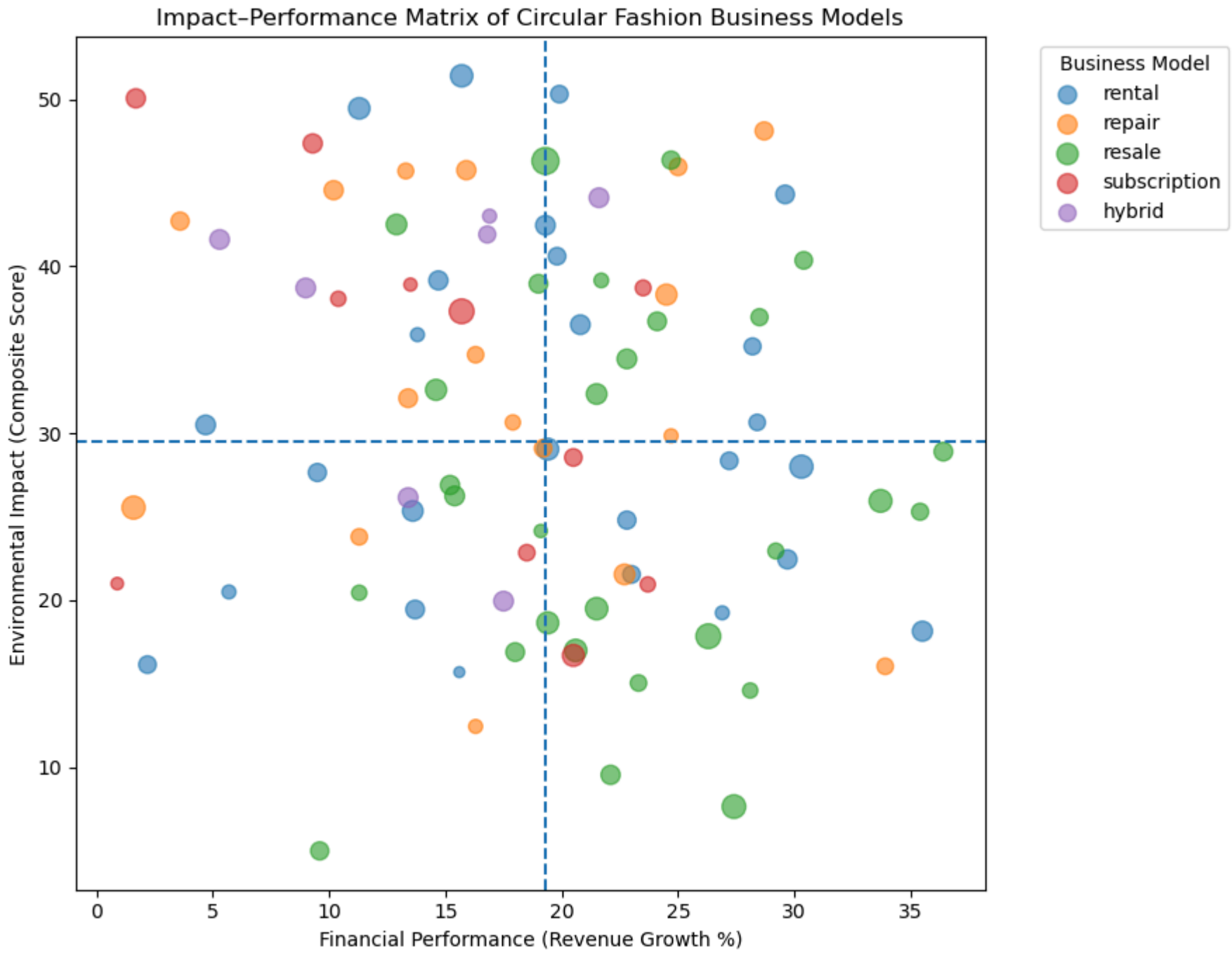

Proxy sentiment vs VADER compound
VADER compound (-1 to 1)
Dataset avg_sentiment_score (-1 to 1)
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0



Which circular models actually work?

In [189…

```
plt.figure(figsize=(9, 7))

# Creating composite axes
df2["impact_score"] = (
    df2["carbon_reduction_pct"] * 0.5 +
    df2["material_circularity_index_0_1"] * 100 * 0.5
)

df2["performance_score"] = df2["revenue_growth_pct_2y"]

# Bubble plot
for model in df2["business_model_type"].unique():
    sub = df2[df2["business_model_type"] == model]
    plt.scatter(
        sub["performance_score"],
        sub["impact_score"],
        s=30 + sub["weighted_degree"] * 5,
        alpha=0.6,
        label=model
    )

# Quadrant lines
plt.axhline(df2["impact_score"].median(), linestyle="--")
plt.axvline(df2["performance_score"].median(), linestyle="--")

plt.xlabel("Financial Performance (Revenue Growth %)")
plt.ylabel("Environmental Impact (Composite Score)")
plt.title("Impact–Performance Matrix of Circular Fashion Business Models")
plt.legend(title="Business Model", bbox_to_anchor=(1.05, 1), loc="upper left")

plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "impact_performance_quadrant.png"), dpi=220)
plt.show()
```

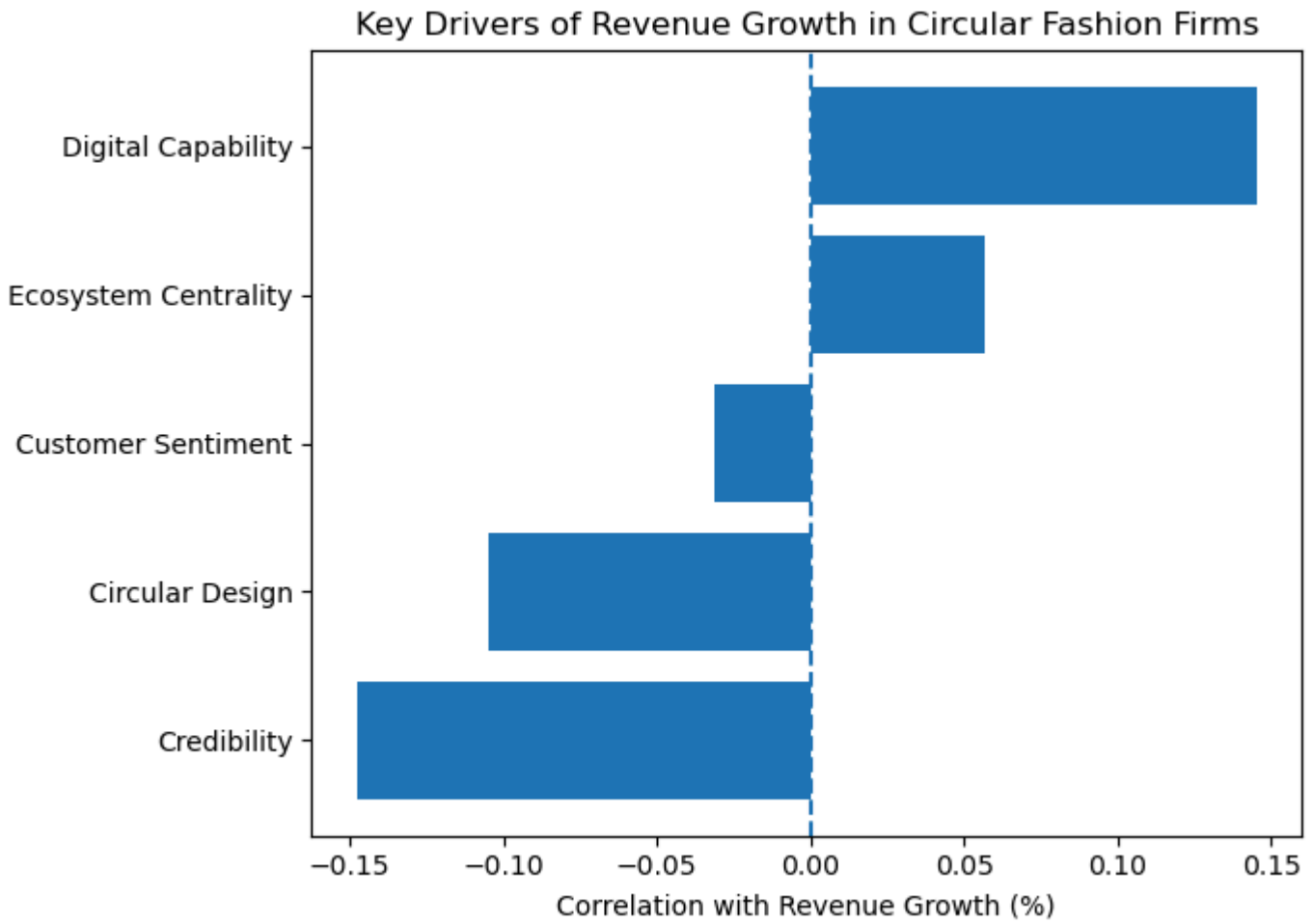

Impact–Performance Matrix of Circular Fashion Business Models
Environmental Impact (Composite Score)
Financial Performance (Revenue Growth %)
0
5
10
15
20
25
30
35
10
20
30
40
50
Business Model
rental
repair
resale
subscription
hybrid

The matrix illustrates that not all circular strategies yield simultaneous economic and environmental benefits. Firms combining strong ecosystem connectivity with closed-loop practices are disproportionately represented in the high-impact, high-performance quadrant.

What Actually Drives Success?

In [192…

```
drivers = {
    "Circular Design": df2["circular_design_index"].corr(df2["revenue_growth_pct_2y"]),
    "Digital Capability": df2["digital_capability_index"].corr(df2["revenue_growth_pct_2y"]),
    "Ecosystem Centrality": df2["weighted_degree"].corr(df2["revenue_growth_pct_2y"]),
    "Credibility": df2["credibility_index"].corr(df2["revenue_growth_pct_2y"]),
    "Customer Sentiment": df2["avg_sentiment_score"].corr(df2["revenue_growth_pct_2y"]),
}

driver_df = (
    pd.DataFrame.from_dict(drivers, orient="index", columns=["Correlation"])
    .sort_values("Correlation")
)

plt.figure(figsize=(7, 5))
plt.barh(driver_df.index, driver_df["Correlation"])
plt.axvline(0, linestyle="--")
plt.title("Key Drivers of Revenue Growth in Circular Fashion Firms")
plt.xlabel("Correlation with Revenue Growth (%)")

plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "key_drivers_revenue_growth.png"), dpi=220)
plt.show()
```



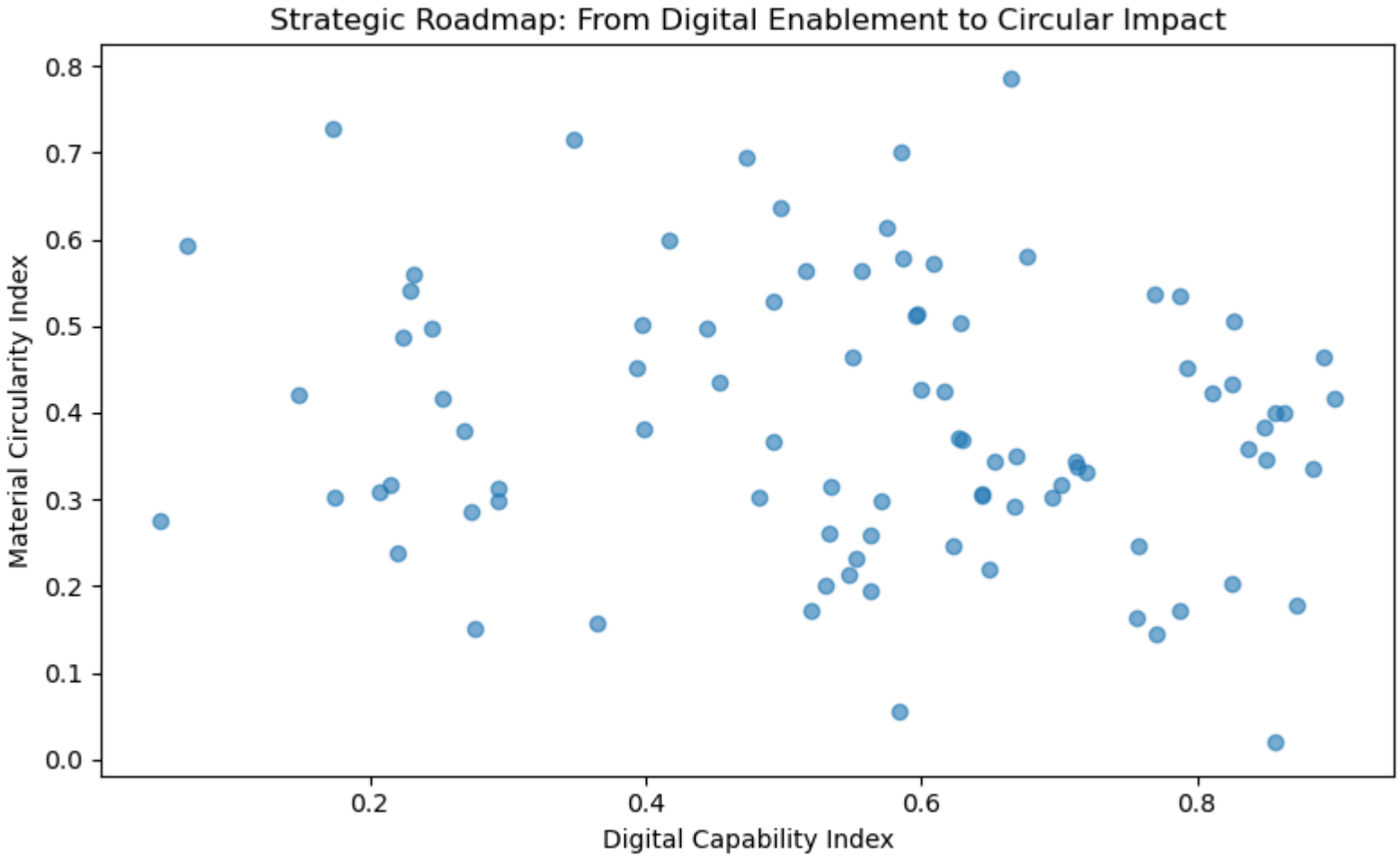
Results indicate that ecosystem positioning and digital capability exhibit stronger associations with firm growth than standalone sustainability claims.

Strategy Roadmap

In [195…

```
plt.figure(figsize=(8, 5))

plt.scatter(
    df2["digital_capability_index"],
```

```
    df2["material_circularity_index_0_1"],
    alpha=0.6
)

plt.xlabel("Digital Capability Index")
plt.ylabel("Material Circularity Index")
plt.title("Strategic Roadmap: From Digital Enablement to Circular Impact")

plt.tight_layout()
plt.savefig(os.path.join(OUT_DIR, "strategy_roadmap_digital_to_circular.png"), dpi=220)
plt.show()
```



The visual roadmap suggests that higher levels of digital integration are a prerequisite for achieving scalable material circularity.