```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
file_path='CommMktArrivals2012.csv'
# --- 1. Load and Clean Data ---

# Specify 'latin1' encoding to resolve UnicodeDecodeError
df = pd.read_csv(file_path, encoding='latin1')
# Strip whitespace from string columns to prevent matching errors
df = df.apply(lambda x: x.str.strip() if x.dtype == "object" else x)

# Standardize Month column for correct chronological sorting
month_order = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct',
df['Month'] = pd.Categorical(df['Month'], categories=month_order, ordered=True)

# Clean special characters from 'District Name' that cause font warnings
df['District Name'] = df['District Name'].str.replace('', '', regex=False)

# Subset for volume-based analysis (Quintals)
df_q = df[df['Unit'] == 'Quintal'].copy()
```

```
# --- 2. Descriptive Statistics ---
print("--- Dataset Info ---")
info_data = []
for col in df.columns:
    info_data.append({
        'Column Name': col,
        'Data Type': df[col].dtype,
        'Non-Null Count': df[col].count()
    })
df_info_table = pd.DataFrame(info_data)
display(df_info_table)
```

```
--- Dataset Info ---
```

	Column Name	Data Type	Non-Null Count
0	District Name	object	21422
1	Taluk Name	object	21422
2	Market Name	object	21422
3	Address	object	21422
4	Telephone	object	21422
5	Commodity	object	21422
6	Year	int64	21422
7	Month	category	21422
8	Arrival	int64	21422
9	Unit	object	21422

Next steps: Generate code with `df_info_table` New interactive sheet

```
print("\n--- Summary Statistics ---")
desc_stats = df.describe(include='all').T
rows_to_drop = ['mean', 'min', '25%', '50%', '75%', 'max', 'std']
desc_stats = desc_stats.drop(rows_to_drop, errors='ignore') # Dropping numerical stat
display(desc_stats)

# Demonstrate numpy usage: calculate mean arrival
mean_arrival = np.mean(df_q['Arrival'])
print(f"\n--- Mean Arrival Volume (Quintals) using NumPy: {mean_arrival:.2f} ---")
```

```
--- Summary Statistics ---
```

	count	unique	top	freq	mean	std	min	25%
District Name	21422	29	Hassan	1504	NaN	NaN	NaN	NaN
Taluk Name	21422	147	Mysore	669	NaN	NaN	NaN	NaN
Market Name	21422	148	MYSORE	669	NaN	NaN	NaN	NaN
Address	21422	148	NANJANGUD ROAD , BANDIPALYA MYSORE	669	NaN	NaN	NaN	NaN
Telephone	21422	148	08212482725	669	NaN	NaN	NaN	NaN
Commodity	21422	156	Maize	1021	NaN	NaN	NaN	NaN
Year	21422.0	NaN	NaN	NaN	2012.0	0.0	2012.0	2012.0
Month	21422	12	Dec	1863	NaN	NaN	NaN	NaN

Next steps: Generate code with `desc_stats` New interactive sheet

```
1 # --- 3. Print Data Summaries ---
2 print("\n--- Top 10 Commodities List ---")
3 display(top_10)
```

```
--- Top 10 Commodities List ---
```

	Commodity	Arrival
0	Paddy	18554513
1	Maize	15460281
2	Onion	9826018
3	Rice	9159723
4	Potato	4467185
5	Green Ginger	2952505
6	Cotton	2861896
7	Tomato	2858265
8	Arecanut	2696704
9	Tur	2562682

Next steps: Generate code with `top_10` New interactive sheet

```
# 4. SUPPLY CHAIN RISK ANALYSIS (Volatility)
# ----------------------------------------------------------
# We calculate the Coefficient of Variation (CV).
# Higher CV = More unpredictable supply = High risk for the market.
top_10_names = df_q.groupby('Commodity')['Arrival'].sum().nlargest(10).index
volatility = df_q[df_q['Commodity'].isin(top_10_names)].groupby('Commodity')['Arriva
volatility['Risk_Score(CV)'] = (volatility['std'] / volatility['mean']) * 100

print("--- Supply Chain Risk Assessment ---")
display(volatility[['Risk_Score(CV)']].sort_values(by='Risk_Score(CV)', ascending=Fa
```

```
--- Supply Chain Risk Assessment ---
```

Commodity	Risk_Score(CV)
Arecanut	693.487346
Rice	535.230129
Onion	430.868343
Maize	415.997238
Potato	343.194815
Tur	330.170121
Paddy	280.496492
Green Ginger	268.315330
Tomato	255.913079
Cotton	206.524517

```
print("\n--- Raw Data Head (Top 10 Commodities) ---")
display(df_q[df_q['Commodity'].isin(top_10['Commodity'])].head(10))
```

```
--- Raw Data Head (Top 10 Commodities) ---
```

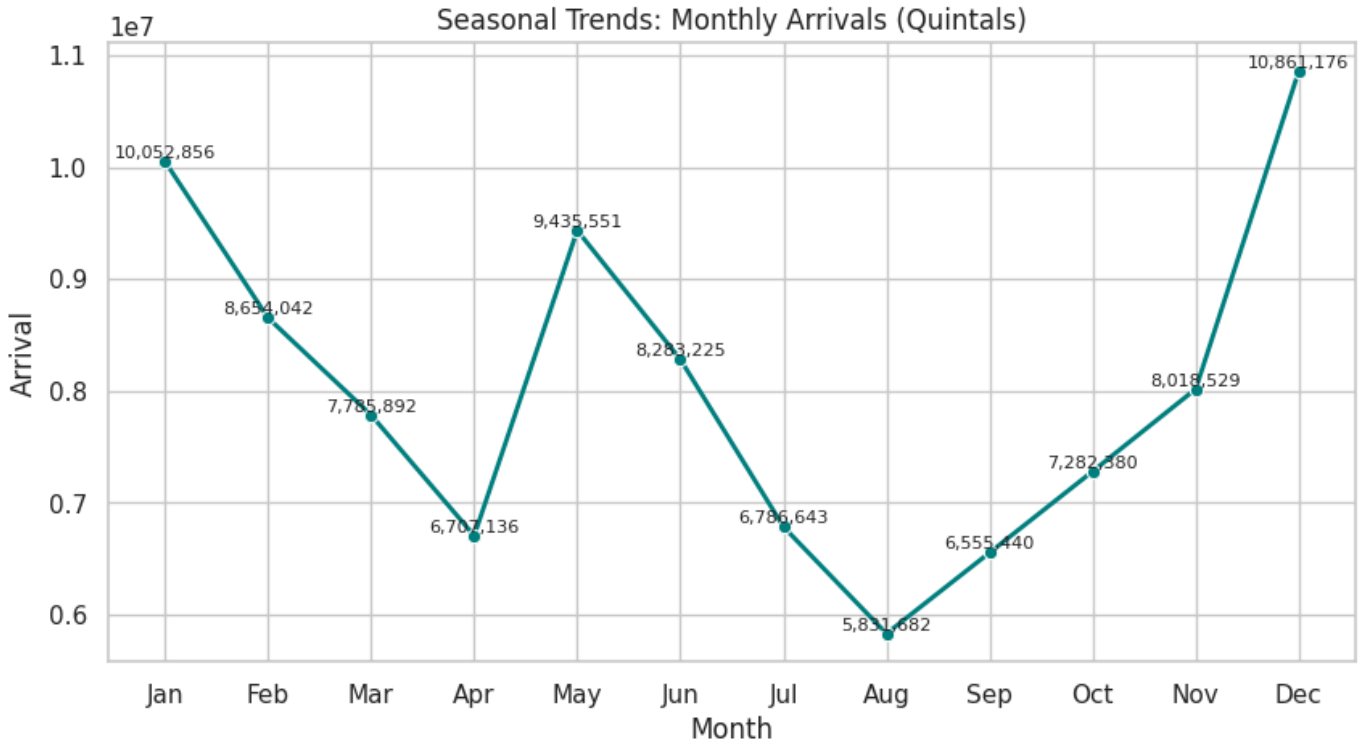
	District Name	Taluk Name	Market Name	Address	Telephone	Commodity	Year	Month	Arrival
7	Bagalakot	Badami	BADAMI	SECRATRY A.P.M.C.BADAMI BADAMI	220042	Maize	2012	Jan	21669
24	Bagalakot	Badami	BADAMI	SECRATRY A.P.M.C.BADAMI BADAMI	220042	Maize	2012	Feb	5184
40	Bagalakot	Badami	BADAMI	SECRATRY A.P.M.C.BADAMI BADAMI	220042	Maize	2012	Mar	10346
50	Bagalakot	Badami	BADAMI	SECRATRY A.P.M.C.BADAMI BADAMI	220042	Cotton	2012	Apr	106
57	Bagalakot	Badami	BADAMI	SECRATRY A.P.M.C.BADAMI	220042	Maize	2012	Apr	107

```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

# --- ADVANCED STEP: K-MEANS CLUSTERING ---
# Let's cluster districts by their top 3 crops (Paddy, Maize, Onion)
top_3_crops = ['Paddy', 'Maize', 'Onion']
cluster_prep = df_q[df_q['Commodity'].isin(top_3_crops)].pivot_table(
    index='District Name', columns='Commodity', values='Arrival', aggfunc='sum', fil
)

# Standardize for ML
scaler = StandardScaler()
scaled_data = scaler.fit_transform(cluster_prep)

# Identify 3 clusters of Districts
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
cluster_prep['Zone_Cluster'] = kmeans.fit_predict(scaled_data)

print("District Zones Identified:")
display(cluster_prep.groupby('Zone_Cluster').mean())
```

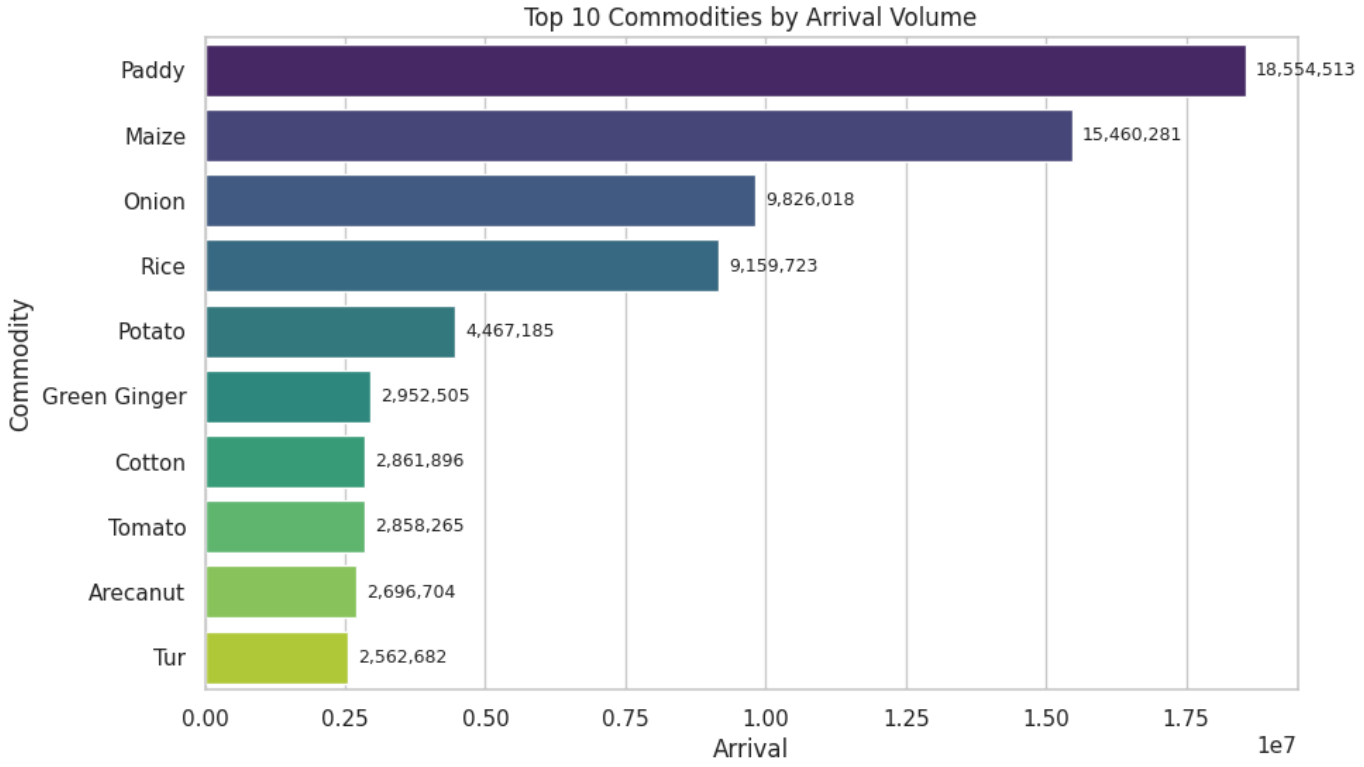
```
District Zones Identified:
```

Commodity	Maize	Onion	Paddy
Zone_Cluster			
0	7.525500e+04	6623416.0	1.300000e+02
1	2.114878e+05	128589.0	2.058246e+05
2	1.788716e+06	62274.0	2.337707e+06

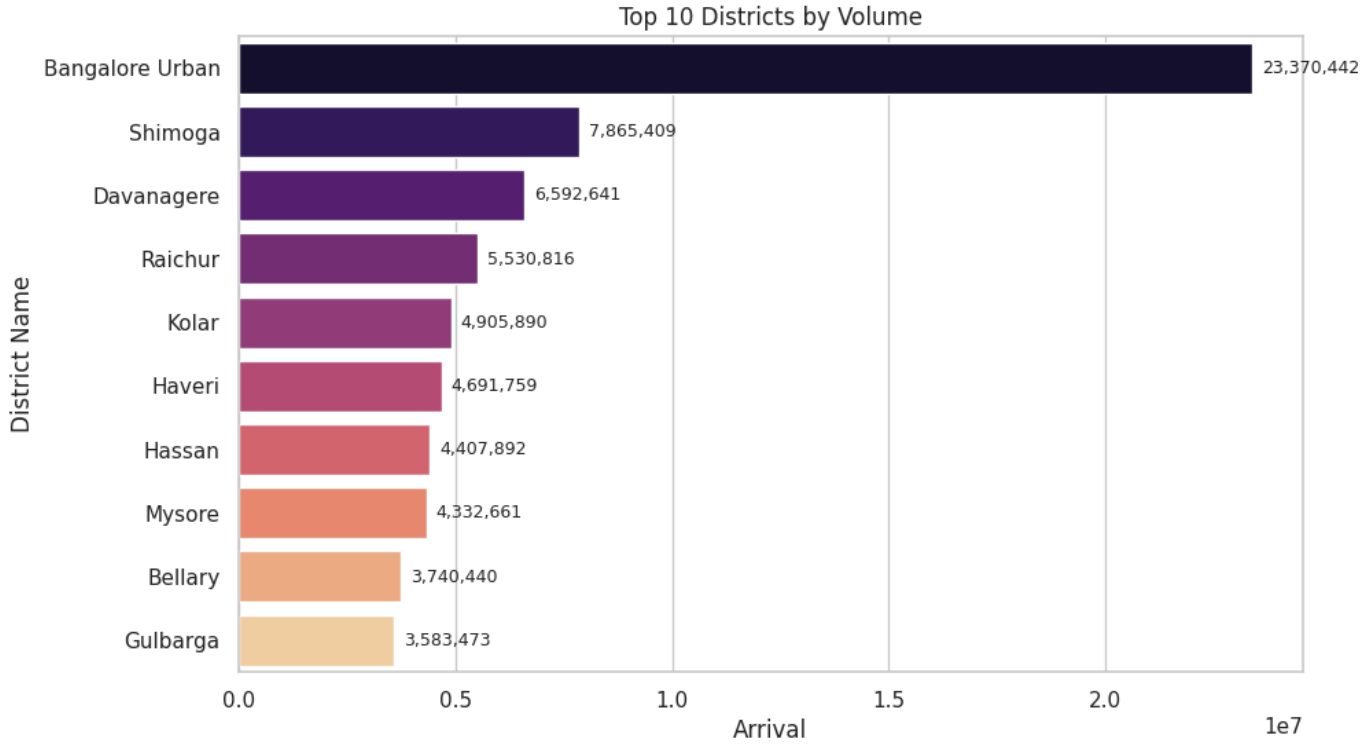
```
# --- 5. Visualization Suite ---
sns.set(style="whitegrid")

# A. Seasonal Trends (Total Volume by Month)
monthly = df_q.groupby('Month', observed=False)['Arrival'].sum().reset_index()
plt.figure(figsize=(10, 5))
ax = sns.lineplot(data=monthly, x='Month', y='Arrival', marker='o', color='teal', li
plt.title('Seasonal Trends: Monthly Arrivals (Quintals)')
for i, point in monthly.iterrows():
    ax.text(point['Month'], point['Arrival'], f"{point['Arrival']:,}", ha='center', v
```




```
# B. Top 10 Commodities
top_10 = df_q.groupby('Commodity')['Arrival'].sum().sort_values(ascending=False).hea
plt.figure(figsize=(10, 6))
ax = sns.barplot(data=top_10, x='Arrival', y='Commodity', palette='viridis', hue='Co
plt.title('Top 10 Commodities by Arrival Volume')
for p in ax.patches:
    ax.annotate(f'{p.get_width():,.0f}', (p.get_width(), p.get_y() + p.get_height() /
                ha='left', va='center', xytext=(5, 0), textcoords='offset points', fo
```

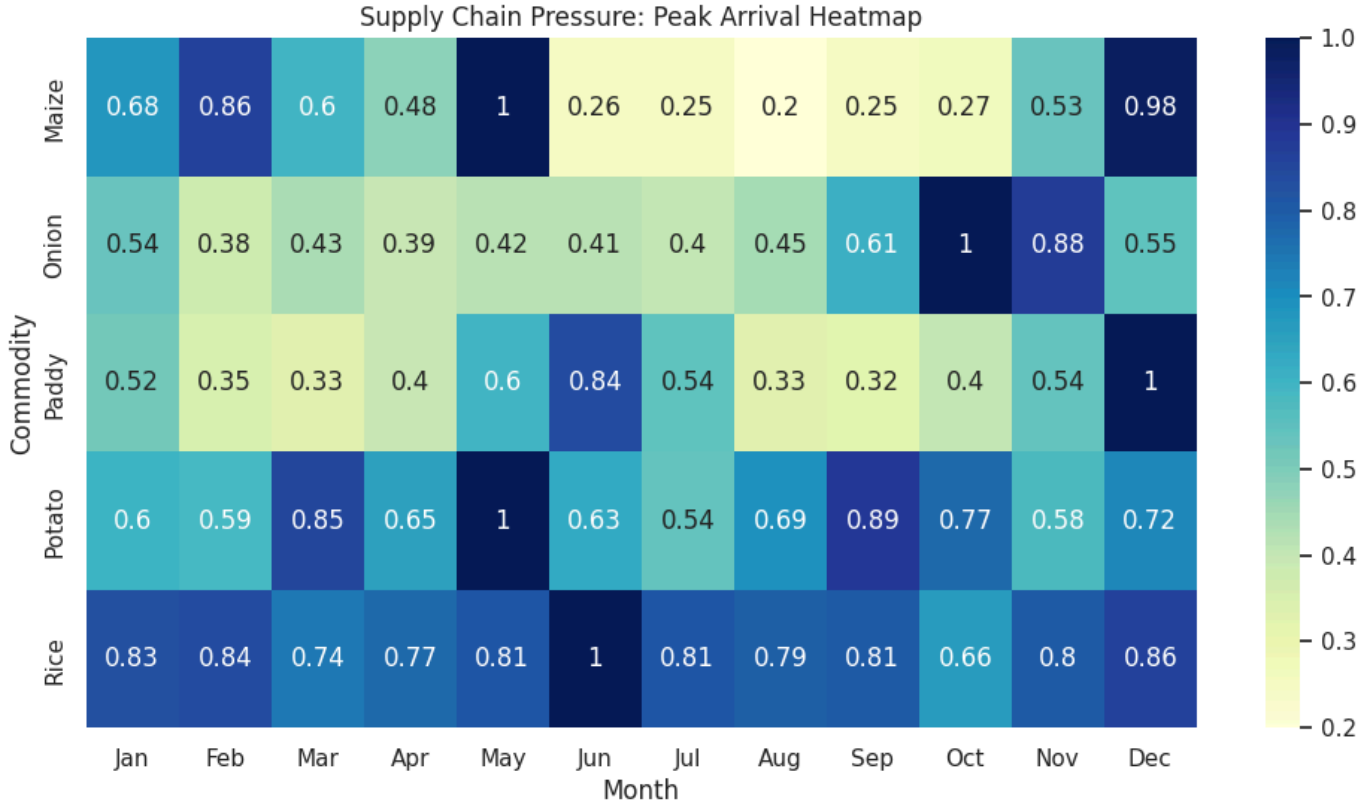



```
# C. Top 10 Districts
districts = df_q.groupby('District Name')['Arrival'].sum().sort_values(ascending=Fals
plt.figure(figsize=(10, 6))
ax = sns.barplot(data=districts, x='Arrival', y='District Name', palette='magma', hue
plt.title('Top 10 Districts by Volume')
for p in ax.patches:
    ax.annotate(f'{p.get_width():,.0f}', (p.get_width(), p.get_y() + p.get_height() /
                ha='left', va='center', xytext=(5, 0), textcoords='offset points', fo
```

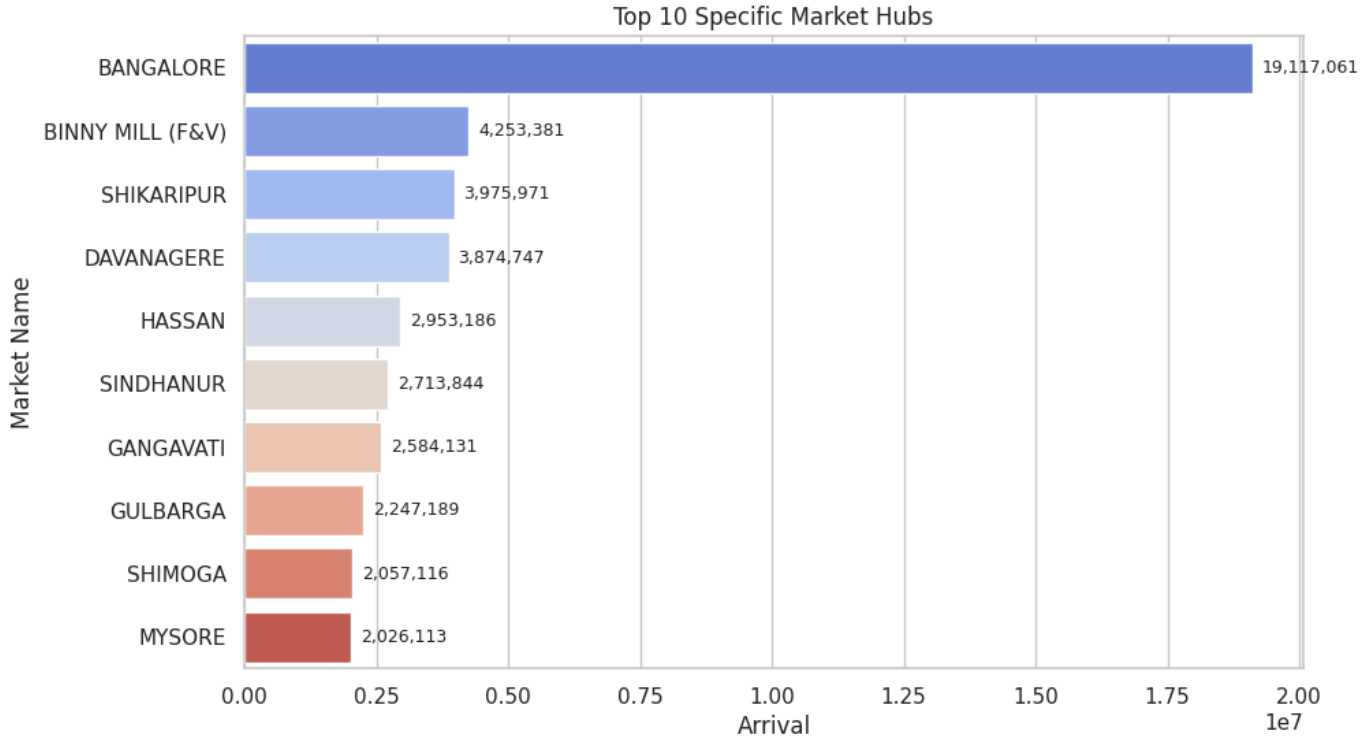



```
# D. Professional Seasonality Heatmap
plt.figure(figsize=(12, 6))
top_5 = df_q.groupby('Commodity')['Arrival'].sum().nlargest(5).index
pivot = df_q[df_q['Commodity'].isin(top_5)].pivot_table(index='Commodity', columns='M
sns.heatmap(pivot.div(pivot.max(axis=1), axis=0), cmap="YlGnBu", annot=True)
plt.title("Supply Chain Pressure: Peak Arrival Heatmap")
```

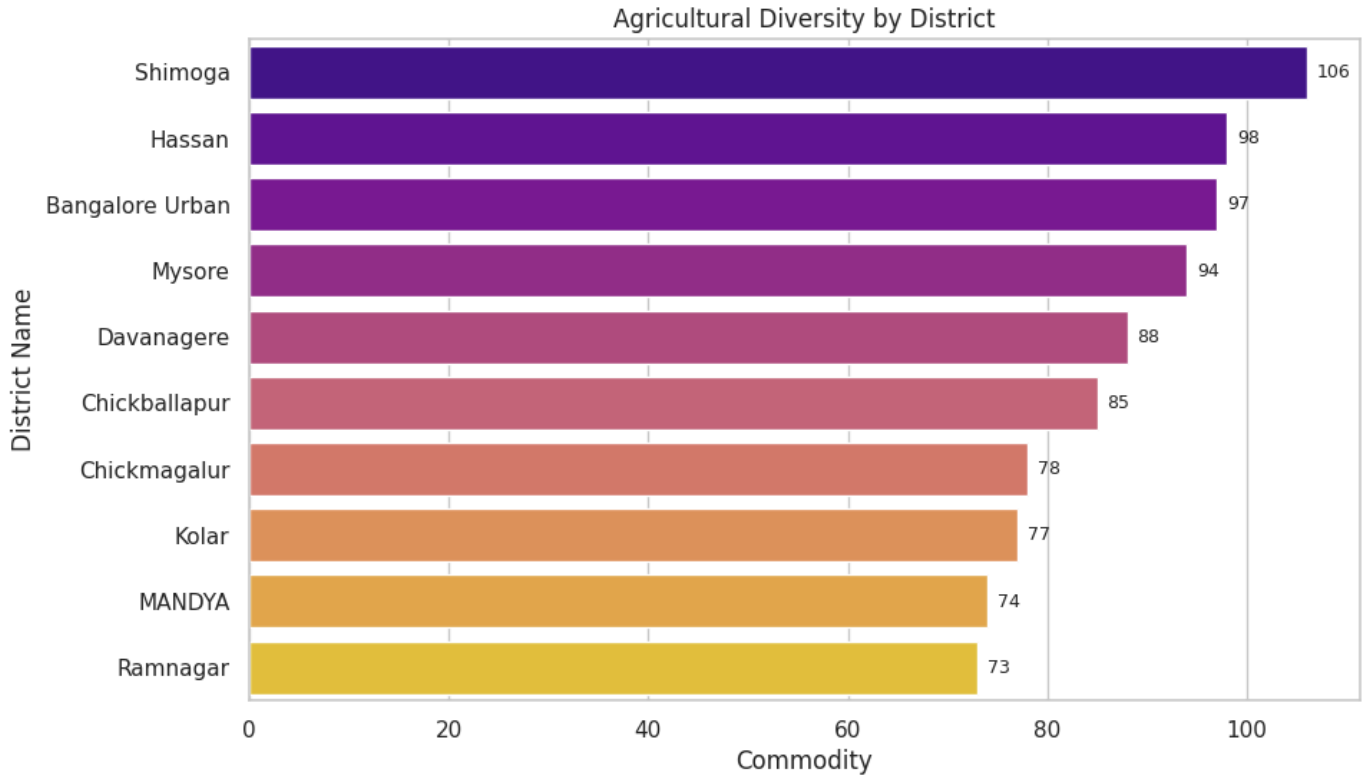
```
/tmp/ipython-input-3960951349.py:4: FutureWarning:

The default value of observed=False is deprecated and will change to observed=True in a fu

Text(0.5, 1.0, 'Supply Chain Pressure: Peak Arrival Heatmap')
```




```
# E. Top Market Hubs
markets = df_q.groupby('Market Name')['Arrival'].sum().sort_values(ascending=False).h
plt.figure(figsize=(10, 6))
ax = sns.barplot(data=markets, x='Arrival', y='Market Name', palette='coolwarm', hue=
plt.title('Top 10 Specific Market Hubs')
for p in ax.patches:
    ax.annotate(f'{p.get_width():,.0f}', (p.get_width(), p.get_y() + p.get_height() /
                ha='left', va='center', xytext=(5, 0), textcoords='offset points', fo

```

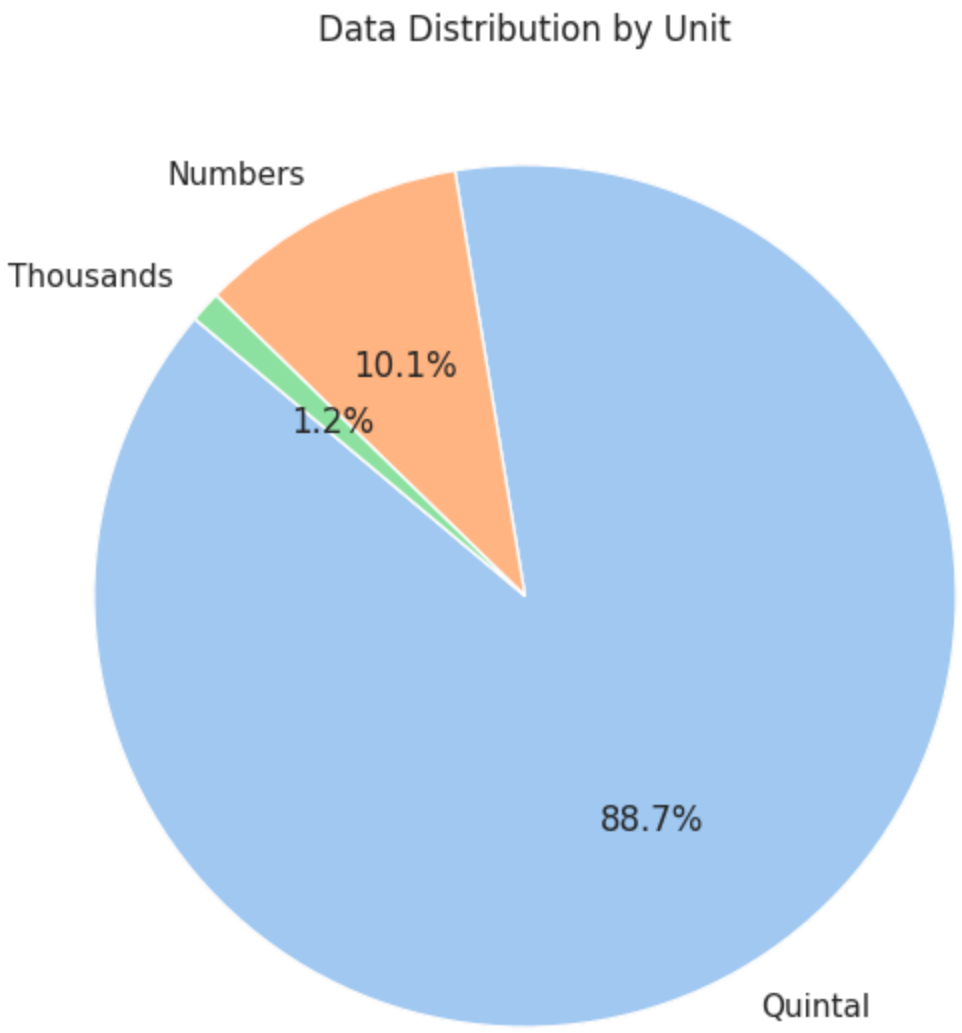



```
# F. Regional Diversity (Unique Commodities per District)
diversity = df.groupby('District Name')['Commodity'].nunique().sort_values(ascending=
plt.figure(figsize=(10, 6))
ax = sns.barplot(data=diversity, x='Commodity', y='District Name', palette='plasma',
plt.title('Agricultural Diversity by District')
for p in ax.patches:
    ax.annotate(f'{p.get_width():.0f}', (p.get_width(), p.get_y() + p.get_height() /
                ha='left', va='center', xytext=(5, 0), textcoords='offset points', fo

```

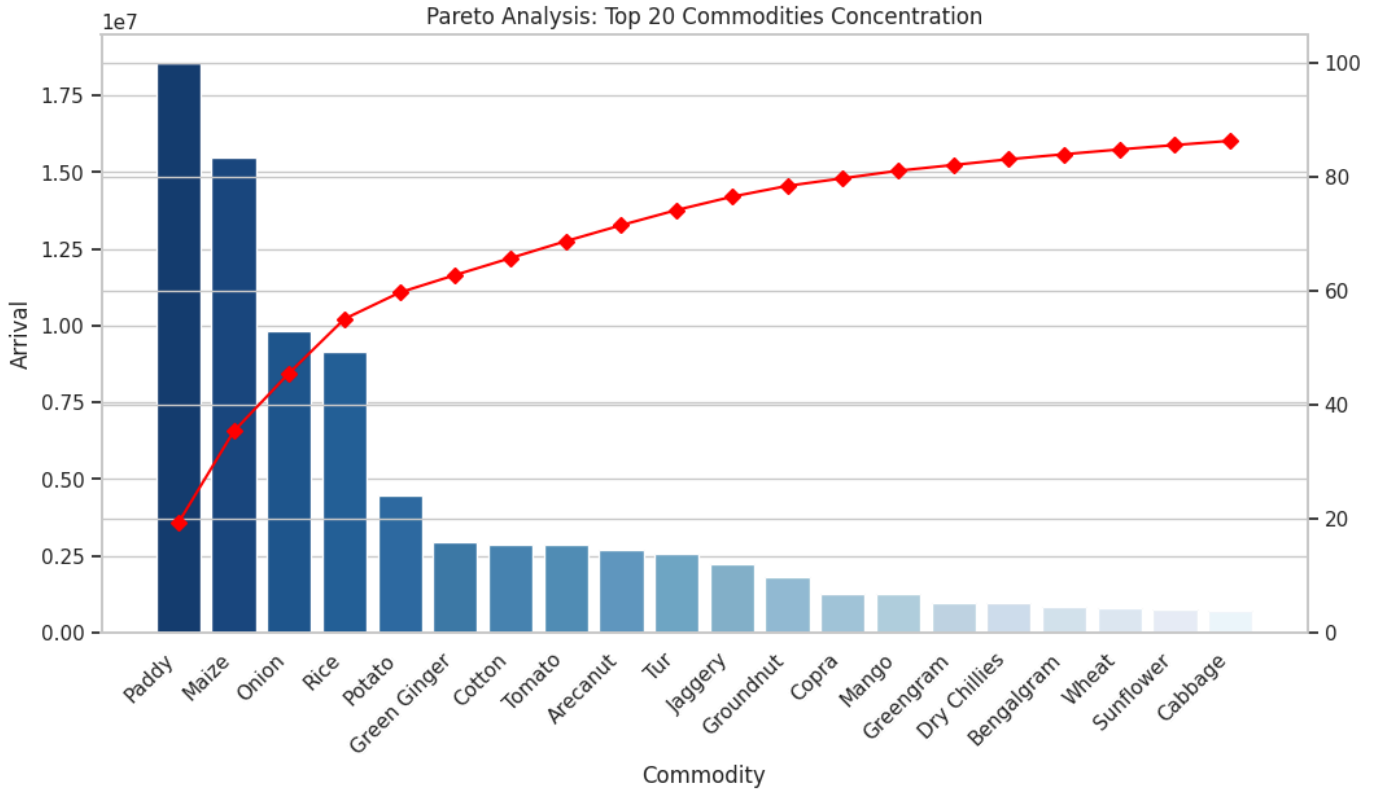



```
 # G. Unit Composition
plt.figure(figsize=(7, 7))
df['Unit'].value_counts().plot.pie(autopct='%1.1f%%', startangle=140, colors=sns.colo
plt.title('Data Distribution by Unit')
plt.ylabel('')


```

```
Text(0, 0.5, '')
```

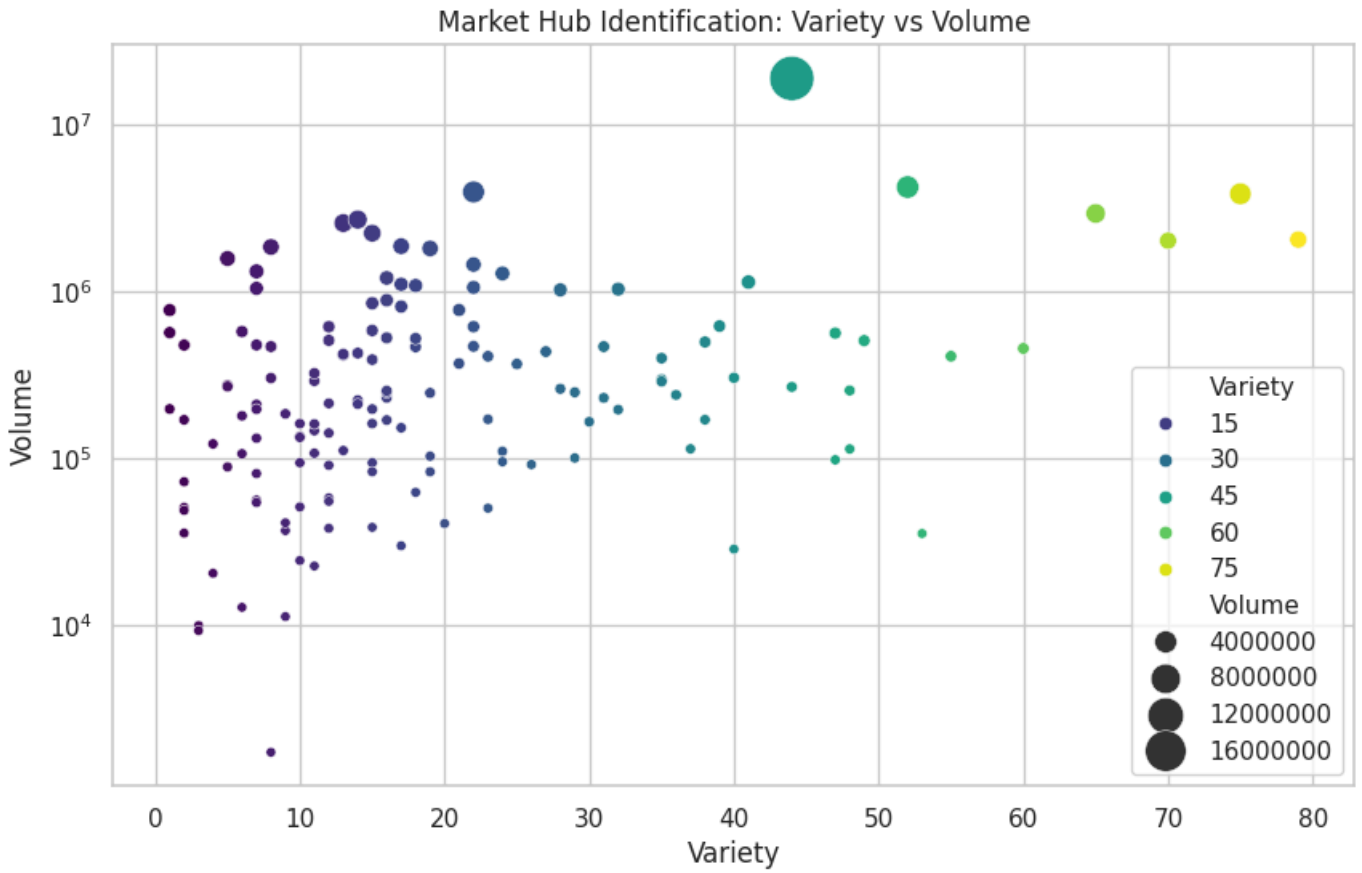



```
# H. Pareto Analysis (Volume Concentration)
comm_totals = df_q.groupby('Commodity')['Arrival'].sum().sort_values(ascending=False
comm_totals['Cumulative %'] = 100 * comm_totals['Arrival'].cumsum() / comm_totals['A
fig, ax1 = plt.subplots(figsize=(12, 6))
sns.barplot(data=comm_totals.head(20), x='Commodity', y='Arrival', ax=ax1, palette='
plt.xticks(rotation=45, ha='right')
ax2 = ax1.twinx()
ax2.plot(comm_totals.head(20)['Commodity'], comm_totals.head(20)['Cumulative %'], co
ax2.set_ylim(0, 105)
plt.title('Pareto Analysis: Top 20 Commodities Concentration')
plt.savefig('8_pareto_analysis.png')

```




```
# I. Market Diversity Plot
market_stats = df_q.groupby(['District Name', 'Market Name']).agg(Variety=('Commodity
plt.figure(figsize=(10, 6))
sns.scatterplot(data=market_stats, x='Variety', y='Volume', hue='Variety', palette='
plt.yscale('log')
plt.title("Market Hub Identification: Variety vs Volume")
```

```
Text(0.5, 1.0, 'Market Hub Identification: Variety vs Volume')
```




```python
# J MARKET DIVERSITY INDEX ---
# This identifies which markets are the backbone of food security
market_diversity = df_q.groupby(['District Name', 'Market Name']).agg(
    Commodity_Count=('Commodity', 'nunique'),
    Total_Volume=('Arrival', 'sum')
).reset_index()

# Categorize Markets
def categorize_market(row):
    if row['Commodity_Count'] > 50: return 'Mega Hub'
    if row['Commodity_Count'] > 20: return 'Regional Center'
    return 'Specialized/Local'

market_diversity['Market_Type'] = market_diversity.apply(categorize_market, axis=1)

plt.figure(figsize=(12, 6))
sns.scatterplot(data=market_diversity, x='Commodity_Count', y='Total_Volume',
                hue='Market_Type', size='Total_Volume', sizes=(50, 500), alpha=0.7)
plt.yscale('log')
plt.title('Market Classification: Diversity vs. Volume', fontsize=15)
plt.xlabel('Number of Unique Commodities', fontsize=12)
```

```
plt.ylabel('Total volume (Log Scale)', fontsize=12)
plt.show()
```