| **Concept** | **Definition** | **NFL Example** |
| --- | --- | --- |
| **Population** | Entire group of interest | QBS Caleb Williams, Joe Burrow, Josh Allen, Bo Nix, Deshaun Watson, Baker Mayfield, Kyler Murray, Justin Herbert, Patrick Mahomes, Jayden Daniels, Anthony Richardson, Dak Prescott, Tua Tagovailoa, Jalen Hurts, Kirk Cousins, Brock Purdy, Daniel Jones, Trevor Lawrence, Aaron Rodgers, Jared Goff, Jordan Love, Andy Dalton, Jacoby Brissett, Aidan O’Connell, Matthew Stafford, Lamar Jackson, Derek Carr, Geno Smith, Justin Fields, CJ Stroud, Will Levis, Sam Darnold |
| **Sample** | A subset of the population | (random selection)  Trevor Lawrence 56.3, Andy Dalton 64.4, Kyler Murray 90.7, Anthony Richardson 94.0, Jared Goff 100.6, CJ Stroud 102.6, Patrick Mahomes 113.7, Deshaun Watson 89.5, Jalen Hurts 106.9, Kirk Cousins 94.2. |
| **Parameter** | A measure describing the entire population | Patrick Mahomes (KC) - 113.7, Joe Burrow (CIN) - 112.4, Josh Allen (BUF) - 109.1, Lamar Jackson (BAL) - 108.3, Dak Prescott (DAL) - 107.2, Jalen Hurts (PHI) - 106.9, Brock Purdy (SF) - 105.7, Jordan Love (GB) - 104.2, Aaron Rodgers (NYJ) - 103.8, Kirk Cousins (ATL) - 103.5, C.J. Stroud (HOU) - 102.6, Tua Tagovailoa (MIA) - 101.9, Matthew Stafford (LAR) - 101.2, Jared Goff (DET) - 100.6  Caleb Williams (CHI) - 100.3, Jayden Daniels (WAS) - 99.8, Sam Darnold (MIN) - 98.5, Derek Carr (NO) - 97.8, Geno Smith (SEA) - 97.5, Justin Fields (CHI) - 95.3, Kirk Cousins - 94.2, Anthony Richardson (IND) - 94.0, Baker Mayfield (TB) - 92.4, Kyler Murray (ARI) - 90.7, Deshaun Watson (CLE) - 89.5, Justin Herbert (LAC) - 88.3, Jacoby Brissett (NE) - 87.6, Daniel Jones (NYG) - 86.4  Joe Flacco (NYJ) - 85.9, Aidan O'Connell (LV) - 84.1, Will Levis (TEN) - 82.7, Jimmy Garoppolo (LV) - 81.3, Mac Jones (NE) - 79.2, Andy Dalton - 64.4, Trevor Lawrence - 56.3, |
| **Statistic** | A measure describing the sample | Mean 101.29, **Variance:** Approximately **430.05**  **Standard Deviation:** Approximately **20.74** |

SD: 56.3−101.2964.4−101.2990.7−101.2994.0−101.29100.6−101.29102.6−101.29113.7−101.2989.5−101.29106.9−101.2994.2−101.29​=−44.99=−36.89=−10.59=−7.29=−0.69=1.31=12.41=−11.79=5.61=−7.09​

### **Step 3: Square Each Deviation**

(−44.99)2=2022.00(−36.89)2=1369.52(−10.59)2=112.12(−7.29)2=53.24(−0.69)2=0.48(1.31)2=1.72(12.41)2=153.78(−11.79)2=138.50(5.61)2=31.52(−7.09)2=50.10\begin{align\*} (-44.99)^2 &= 2022.00 \\ (-36.89)^2 &= 1369.52 \\ (-10.59)^2 &= 112.12 \\ (-7.29)^2 &= 53.24 \\ (-0.69)^2 &= 0.48 \\ (1.31)^2 &= 1.72 \\ (12.41)^2 &= 153.78 \\ (-11.79)^2 &= 138.50 \\ (5.61)^2 &= 31.52 \\ (-7.09)^2 &= 50.10 \\ \end{align\*}(−44.99)2(−36.89)2(−10.59)2(−7.29)2(−0.69)2(1.31)2(12.41)2(−11.79)2(5.61)2(−7.09)2​=2022.00=1369.52=112.12=53.24=0.48=1.72=153.78=138.50=31.52=50.10​

### **Step 4: Sum the Squared Deviations**

2022.00+1369.52+112.12+53.24+0.48+1.72+153.78+138.50+31.52+50.10=3880.482022.00 + 1369.52 + 112.12 + 53.24 + 0.48 + 1.72 + 153.78 + 138.50 + 31.52 + 50.10 = 3880.482022.00+1369.52+112.12+53.24+0.48+1.72+153.78+138.50+31.52+50.10=3880.48

### **Step 5: Calculate Variance**

* **For a sample**:

s2=3880.4810−1=3880.489≈430.05s^2 = \frac{3880.48}{10 - 1} = \frac{3880.48}{9} \approx 430.05s2=10−13880.48​=93880.48​≈430.05

s2=10−13880.48​=93880.48​≈430.05

### **Step 6: Calculate Standard Deviation**

s=430.05≈20.74s = \sqrt{430.05} \approx 20.74s=430.05​≈20.74

**Basis Code for Google\_Docs Data Extraction and Visualization**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from docx import Document

# Path to your document

doc\_path = r"C:\Users\jamar\Downloads\GCU\_DSC\_510\_Discussion\_Post\_NFL\_Statistics\_Example.docx"

# Load the document

doc = Document(doc\_path)

# Initialize a list to store table data

table\_data = []

# Extract data from the document

for table in doc.tables:

for row in table.rows:

row\_data = [cell.text.strip() for cell in row.cells] # Strip whitespace

table\_data.append(row\_data)

# Print the extracted table data

print("Extracted Table Data:")

for row in table\_data:

print(row)

# Convert to DataFrame (assuming first row is the header)

if table\_data: # Check if there is any table data

df = pd.DataFrame(table\_data[1:], columns=table\_data[0])

else:

df = pd.DataFrame() # Create an empty DataFrame

# Display the DataFrame

print("\nDataFrame:")

print(df)

# Check if the DataFrame is empty

if df.empty:

print("No data available to plot.")

else:

# Convert columns to numeric where possible

df = df.apply(pd.to\_numeric, errors='coerce')

df = df.dropna(axis=1, how='all') # Drop columns with all NaN values

# Create a figure with subplots

num\_features = len(df.columns)

if num\_features > 0:

fig, axs = plt.subplots(2, (num\_features + 1) // 2, figsize=(20, 10))

axs = axs.flatten() # Flatten the array of axes for easy iteration

# Boxplots

for idx, column in enumerate(df.columns):

if pd.api.types.is\_numeric\_dtype(df[column]):

sns.boxplot(data=df, y=column, ax=axs[idx], palette="Set2")

axs[idx].set\_title(f'Boxplot of {column}', fontsize=16)

axs[idx].set\_ylabel('Values')

axs[idx].set\_xlabel(column)

else:

axs[idx].remove() # Remove the subplot if the data is not numeric

plt.tight\_layout()

plt.show()

# Now, let's create histograms in a new figure

fig, axs = plt.subplots(2, (num\_features + 1) // 2, figsize=(20, 10))

axs = axs.flatten()

for idx, column in enumerate(df.columns):

if pd.api.types.is\_numeric\_dtype(df[column]):

sns.histplot(df[column], ax=axs[idx], bins=15, kde=True, color='blue')

axs[idx].set\_title(f'Histogram of {column}', fontsize=16)

axs[idx].set\_ylabel('Frequency')

axs[idx].set\_xlabel(column)

else:

axs[idx].remove() # Remove the subplot if the data is not numeric

plt.tight\_layout()

plt.show()

**Visualization Code.**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from docx import Document

# Path to your document

doc\_path = r"C:\Users\jamar\Downloads\GCU\_DSC\_510\_Discussion\_Post\_NFL\_Statistics\_Example.docx"

# Load the document

doc = Document(doc\_path)

# Initialize a list to store table data

table\_data = []

# Extract data from the document

for table in doc.tables:

for row in table.rows:

row\_data = [cell.text.strip() for cell in row.cells] # Strip whitespace

table\_data.append(row\_data)

# Convert to DataFrame (assuming first row is the header)

if table\_data: # Check if there is any table data

df = pd.DataFrame(table\_data[1:], columns=table\_data[0])

else:

df = pd.DataFrame() # Create an empty DataFrame

# Display the DataFrame

print("\nDataFrame:")

print(df)

# Check if the DataFrame is empty

if df.empty:

print("No data available to plot.")

else:

# Extracting relevant statistics from the 'Statistic' column

# This assumes you have mean, variance, and standard deviation listed in your DataFrame

stats\_data = {

'Statistic': ['Mean', 'Variance', 'Standard Deviation'],

'Values': [101.29, 430.05, 20.74] # Sample values; replace with actual data extraction if necessary

}

stats\_df = pd.DataFrame(stats\_data)

# Create a bar plot for mean, variance, and standard deviation

plt.figure(figsize=(10, 6))

sns.barplot(x='Values', y='Statistic', data=stats\_df, palette="Set2")

plt.title('Mean, Variance, and Standard Deviation of NFL Statistics', fontsize=16)

plt.xlabel('Value', fontsize=12)

plt.ylabel('Statistic', fontsize=12)

plt.show()