BSDS-207 - SAS for Data Science

Course Instructor - Ms. Neema Jha

Gauri Sharan-BSc Data Science (Sem 3)

January 6, 2024

Outline

- Overview
- Basic Syntax
- Basics - fillna, char function, datatypes, date function, num function, if else conditional statements, tranward function, formats
- Structuring data - stacking, interleaving, merging, proc sql, transposing data, retain statement
- Visualizing data - charts like hbar, vbar, scatterplot
- Analyzing data - grouping and subgrouping, Means, Frequency analysis, Univariate analysis, ODS, Macros

SAS Programming Overview

- SAS (Statistical Analysis System) is a powerful software suite for data analysis.
- It includes tools for data management, statistical analysis, and reporting.
- SAS programs consist of data steps, proc steps, and other procedures.

Basic SAS Syntax

```
data example;
   input name $ age height weight;
   datalines;
John 25 180 75
Mary 30 165 60
Alice 22 175 68
;
```

- The `data` step creates a dataset.
- `input` statement defines variables and their types.
- `datalines` indicate inline data.

Handling Missing Values - FILLNA

- The `FILLNA` function in SAS is used to replace missing values with specified values.

```
data example;
   set example;
   age = fillna(age, 0);
   /*Replace missing age with 0*/
run;
```

Character Functions - CHAR

- The CHAR function converts numeric values to character values.

```
data example;
   set example;
   char_name = char(age);
   /* Convert age to character */
run;
```

Data Types in SAS

- SAS supports various data types, including numeric and character.

```
data example;
   length name $20;
   age = 25;
   /* Define character variable 'name' */
run;
```

Date Functions in SAS

- SAS has a variety of date functions for manipulating date values.

```
data example;
   set example;
   new_date = intnx('month', date_column, 3);
   /* Add 3 months to date_column */
run;
```

Numeric Functions in SAS

- Numeric functions in SAS perform operations on numeric values.

```
data example;
   set example;
   new_weight = round(weight, 0.1);
   /* Round weight to one decimal place */
run;
```

Conditional Processing - IF-ELSE

- Use the `IF-ELSE` statement for conditional processing in SAS.

```
data example;
   set example;
   if age ge 18 then status = 'Adult';
   else status = 'Minor';
run;
```

Transposing Data - TRANWRD

- The `TRANWRD` function in SAS is used to replace specified substrings in a character variable.

```
data example;
   set example;
   new_name = tranwrd(name, 'John', 'Jonathan');
run;
```

Formats in SAS

- Formats in SAS control the appearance of data values.

```
data example;
   set example;
   format salary 8.2;
   /* Display salary with 8 total columns
   and 2 decimal places */
run;
```

Comma Format in SAS

- The `COMMA` format adds commas to large numeric values.

```
data example;
   set example;
   format revenue 10.2 comma.;
   /* Display revenue with commas
   and 2 decimal places */
run;
```

Stacking Datasets in SAS

- Stacking involves combining datasets vertically.

```
data stacked_data;
   set dataset1 dataset2;
run;
```

Merging Datasets in SAS

- Merging involves combining datasets horizontally based on common variables.

```
data merged_data;
   merge dataset1 dataset2;
   by common_variable;
run;
```

Transposing Data in SAS

- Transposing involves converting data from a wide to a long format or vice versa.

```
data transposed_data;
  set original_data;
  /* Transpose variables as needed */
run;
```

Using RETAIN Statement

- The RETAIN statement in SAS is used to carry forward values from one iteration of the data step to the next.

```
data retained_data;
  set input_data;
  retain previous_value;
  /* Process and retain values as needed */
run;
```

Data Manipulation in SAS

- SAS provides powerful tools for data manipulation.
- The `proc sql` procedure is commonly used for querying and transforming data.

```
proc sql;
   create table new_data as
   select name, age, height
   from example
   where weight > 70;
quit;
```

Using PROC SQL for Merging

- PROC SQL provides a SQL-like approach for merging datasets.

```
proc sql;
   create table merged_data as
   select *
   from dataset1
   left join dataset2
   on dataset1.common_variable
   = dataset2.common_variable;
quit;
```

Sorting Datasets in SAS using PROC

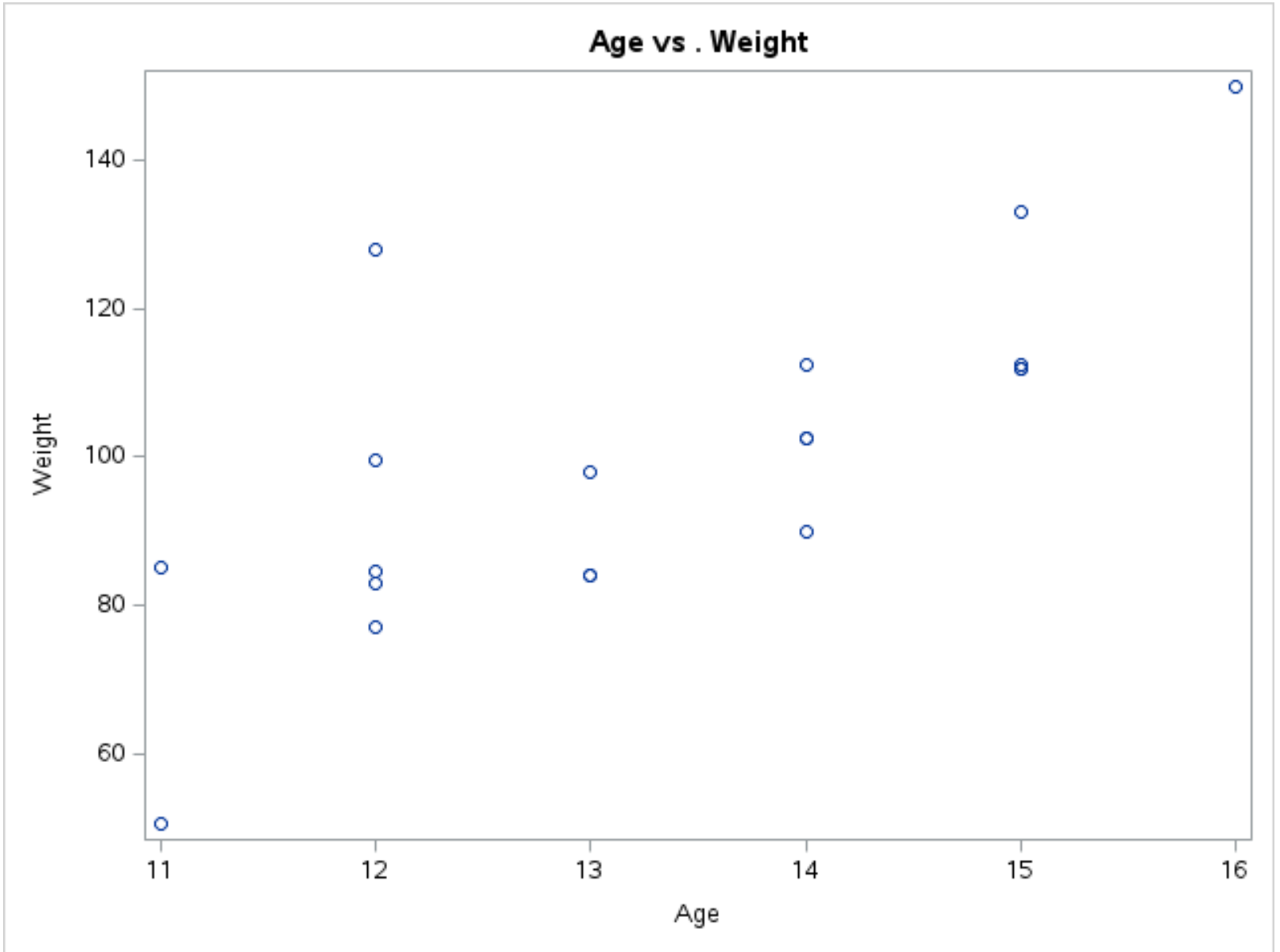
- Sorting arranges observations in a dataset based on the values of one or more variables.

```
data sorted_data;
  set unsorted_data;
  /* Sort data by variable(s) */
  proc sort data=unsorted_data;
    by variable1 variable2;
  run;
run;
```

Data Visualization in SAS

- SAS supports various procedures for creating visualizations.
- The `proc sgplot` procedure is commonly used for creating simple plots.

```
proc sgplot data=sashelp.class;
   scatter x=age y=weight;
   title 'Age-vs.-Weight';
run;
```


Age vs . Weight
Weight
140
120
100
80
60
11
12
13
14
15
16
Age

Statistical Analysis in SAS

- SAS provides a wide range of statistical procedures for analysis.
- The `proc ttest` procedure can be used for t-tests.

```
proc ttest data=sashelp.class;
   class sex;
   var weight;
run;
```

The TTEST Procedure

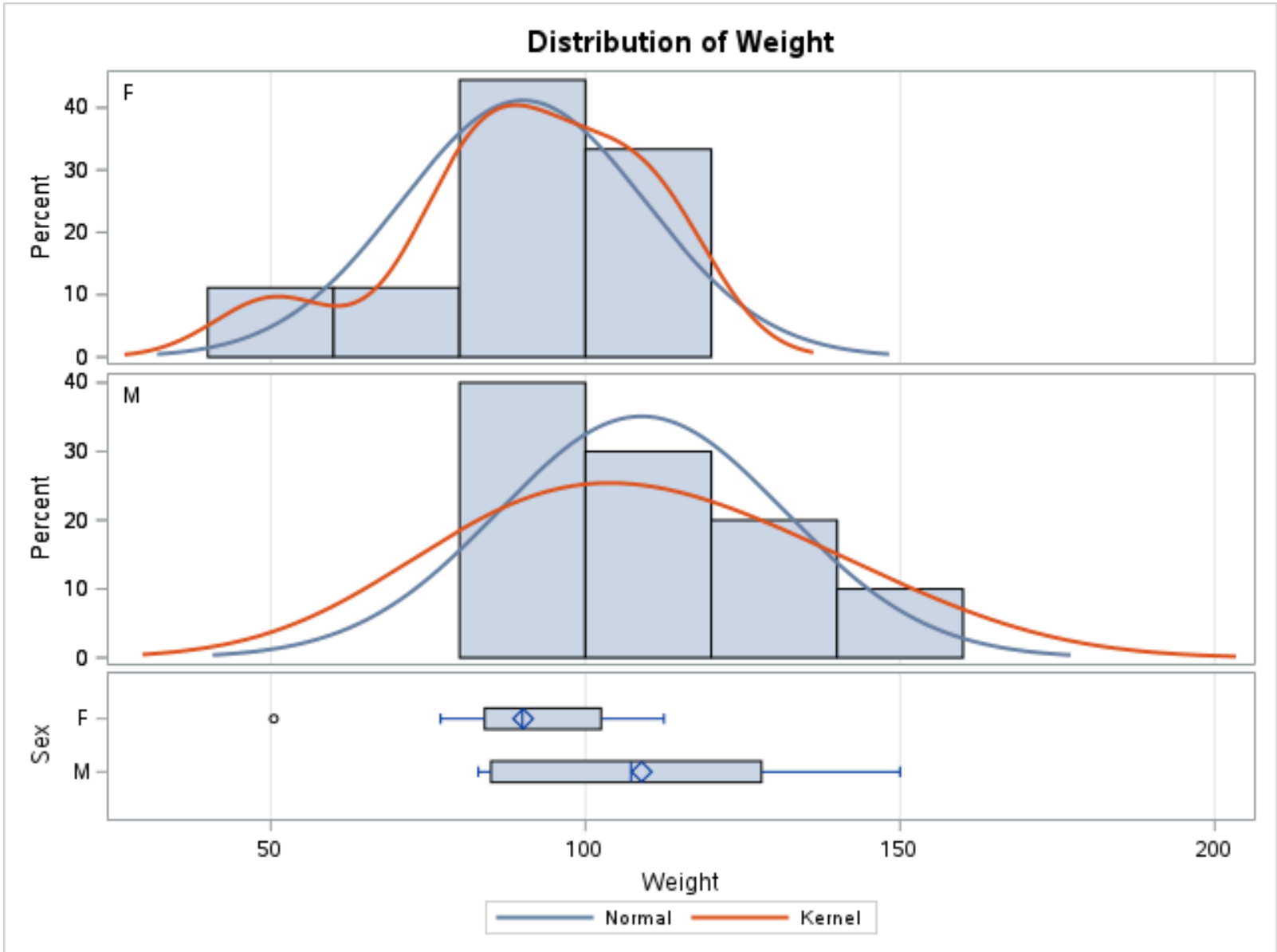
Variable: Weight

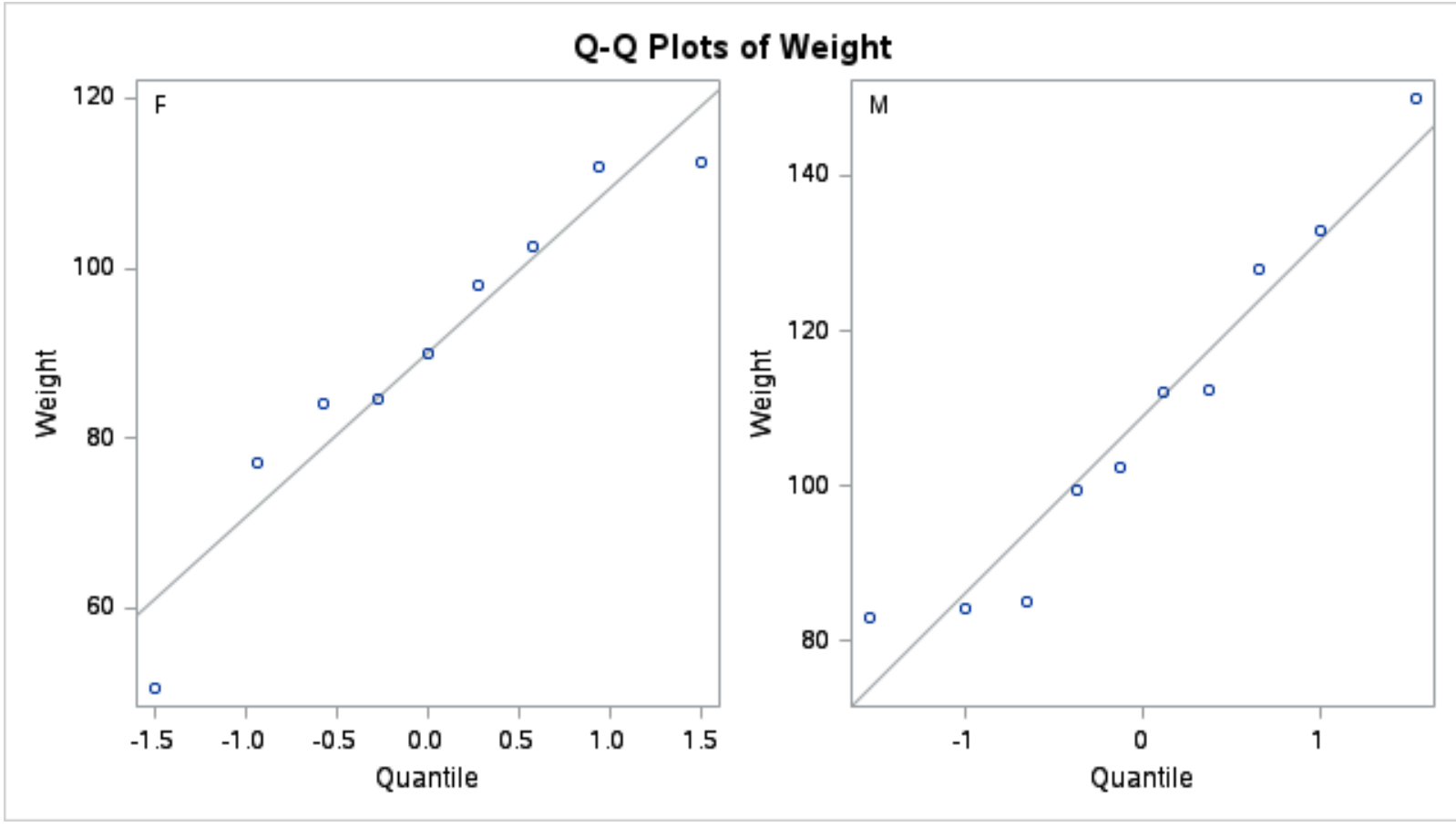
Sex	Method	N	Mean	Std Dev	Std Err	Minimum	Maximum
F		9	90.1111	19.3839	6.4613	50.5000	112.5
M		10	109.0	22.7272	7.1870	83.0000	150.0
Diff (1-2)	Pooled		-18.8389	21.2196	9.7497		
Diff (1-2)	Satterthwaite		-18.8389		9.6644		

Sex	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
F		90.1111	75.2113	105.0	19.3839	13.0930	37.1351
M		109.0	92.6920	125.2	22.7272	15.6326	41.4910
Diff (1-2)	Pooled	-18.8389	-39.4090	1.7313	21.2196	15.9229	31.8112
Diff (1-2)	Satterthwaite	-18.8389	-39.2325	1.5547			

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	17	-1.93	0.0702
Satterthwaite	Unequal	16.962	-1.95	0.0680

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	9	8	1.37	0.6645

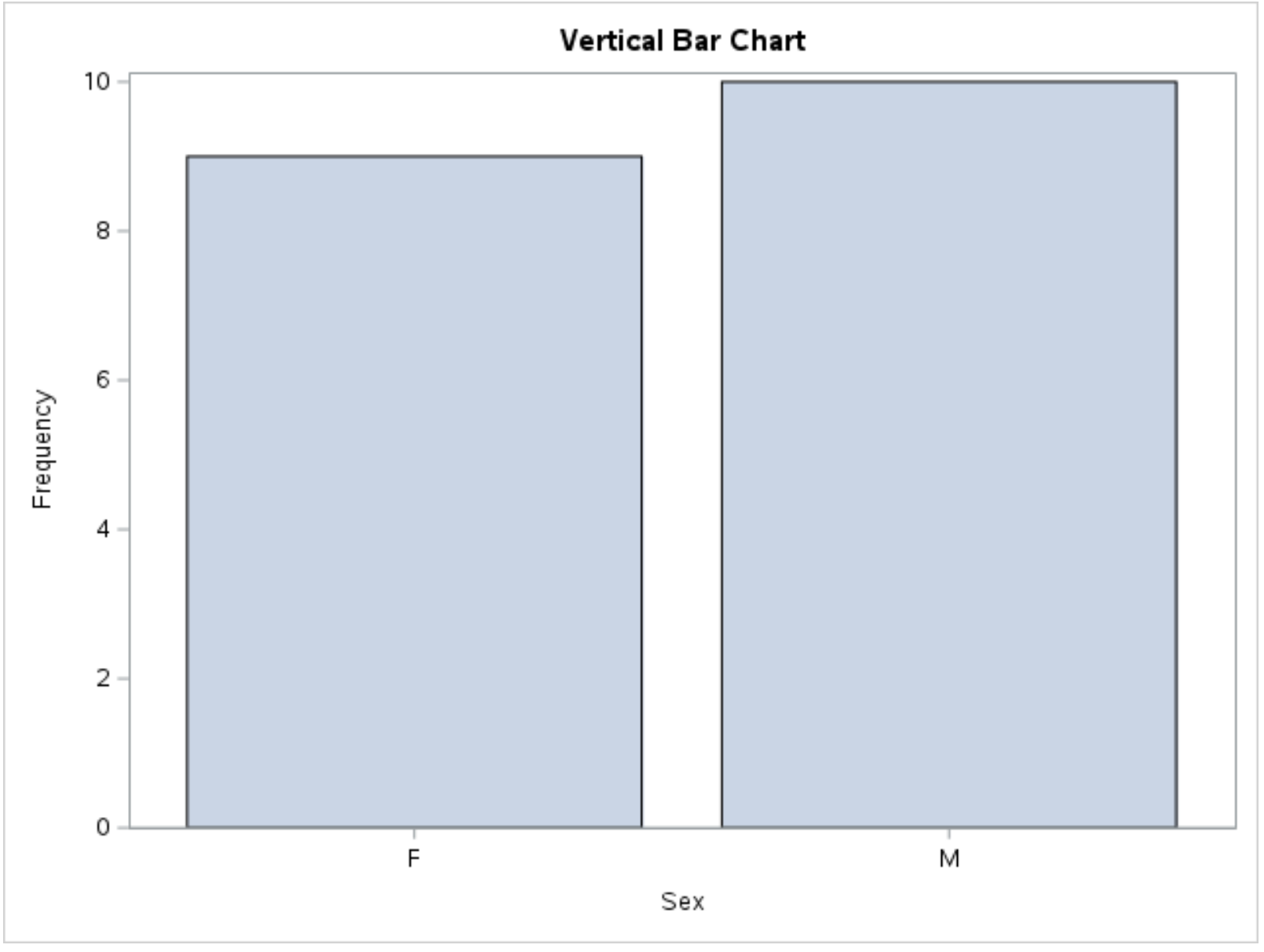

Distribution of Weight
F
M
Percent
40
30
20
10
0
Sex
F
M
o
50
100
150
200
Weight
Normal
Kernel


Q-Q Plots of Weight
F
M
Weight
Quantile
120
100
80
60
140
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
-1
0
1

Creating Charts - HBAR and VBAR

- SAS provides procedures like `proc sgplot` for creating various charts.

```
proc sgplot data=sashelp.cars;
  vbar sex; /*categorical variable*/
  title 'Vertical-Bar-Chart';
run;
```


Vertical Bar Chart
10
8
6
4
2
0
Frequency
F
M
Sex

Grouping and Subgrouping Data

- Use `group` and `subgroup` options in SAS procedures for grouping and subgrouping data in charts.

```
proc sgplot data=example;
   vbar category_variable /
   group=group_variable
   subgroup=subgroup_variable;
   title 'Grouped-and-Subgrouped-Bar-Chart';
run;
```

Calculating Means in SAS

- SAS provides procedures like `proc means` for calculating means and other summary statistics.

```
proc means data=sashelp.class;
   var height;
run;
```

The MEANS Procedure

Analysis Variable : Height				
N	Mean	Std Dev	Minimum	Maximum
19	62.3368421	5.1270752	51.3000000	72.0000000

Frequency Analysis in SAS

- Use `proc freq` for frequency analysis in SAS.

```
proc freq data=example;
  tables categorical_variable;
run;
```

Univariate Analysis in SAS

- `proc univariate` provides comprehensive univariate analysis.

```
proc univariate data=example;
   var numeric_variable;
   histogram;
run;
```

Output Delivery System (ODS) in SAS

- ODS allows you to capture and format SAS output for various destinations.

```
ods html file='output.html';
proc means data=example mean;
   var numeric_variable;
run;
ods html close;
```

Program Data Vector (PDV) in SAS

- The Program Data Vector (PDV) is a crucial concept in SAS DATA steps.
- It serves as an internal memory area for reading, processing, and temporarily storing data during a DATA step execution.
- The PDV holds values of variables being processed, allowing for data manipulation.
- Only variables referenced in the DATA step are stored in the PDV, optimizing memory usage.
- Automatic variables like _N_ and _ERROR_ are created in the PDV for additional functionality.

Example: Using the Program Data Vector (PDV)

```
data example;
   set input_data;

   /* Data manipulation using PDV */
   new_variable = old_variable * 2;

   /* Additional PDV manipulation */
   if condition then new_variable = new_variable + 10;

   output;
run;
```

- In this example, the PDV is utilized to create a new variable (`new_variable`) based on the existing variable (`old_variable`).

Macro Programming in SAS

- Macros in SAS allow for the creation of reusable code snippets.
- Use the %macro and %mend statements to define a macro.

```
%macro print_summary(data);
   proc means data=&data;
      var age weight;
   run;
%mend;

%print_summary(example);
```

Conclusion

- SAS programming is a versatile tool for data analysis.
- It includes data manipulation, visualization, and statistical analysis capabilities.
- The SAS documentation can be explored for more advanced features and techniques.

References

- SAS Documentation: `https://documentation.sas.com/doc/en/pgmsascdc/v_037/`
- SAS Communities: `https://communities.sas.com/`