SADSA

Software Application for Data Science and Analytics

User Manual

Version: Full Version (V-02.25.0.0.1)

Developer: Dr. M. Kamakshaiah / AMCHIK SOLUTIONS

Website: http://codingfigs.com

Contact: contact@codingfigs.com

Generated: December 02, 2025

SADSA - Software Application for Data Science and Analytics

Version: Full Version
Developer: Dr. M. Kamakshaiah / AMCHIK SOLUTIONS
Website: http://codingfigs.com
Contact: contact@codingfigs.com

Overview

SADSA (Software Application for Data Science and Analytics) is a comprehensive desktop application for statistical analysis, data visualization, and machine learning. Built with Python, SADSA provides an intuitive graphical interface for performing advanced analytics without writing code.

🔐 License & Trial System

30-Day Free Trial with Full Features

SADSA offers a **30-day free trial** with **ALL FEATURES ENABLED** from day one:

Installation Instructions

The open source alternative to this is available at https://github.com/codingfigs/sadsa-os

Prerequisites

- Windows 10 or later (64-bit)

- Administrator privileges required for installation

Step-by-Step Installation

1. **Download the Installer**
2. Download `sadsa.exe` from the releases section of this repository
3. Save the file to your Downloads folder or preferred location
4. **Run as Administrator**
5. Right-click on `sadsa.exe`
6. Select "Run as administrator" from the context menu
7. Click "Yes" when prompted by User Account Control (UAC)
8. **Follow Installation Wizard**
9. Click "Next" on the welcome screen
10. Read and accept the License Agreement
11. Choose installation directory (default: `C:\Program Files\SADSA`)
12. Select Start Menu folder (default: SADSA)
13. Click "Install" to begin installation
14. **Complete Installation**
15. Wait for files to be copied (approximately 1-2 minutes)
16. Click "Finish" to complete installation
17. A desktop shortcut will be created automatically
18. **First Launch**
19. Double-click the SADSA icon on your desktop
20. The application will initialize (first launch may take 10-15 seconds)
21. You're ready to start analyzing data!

Troubleshooting Installation

- **Permission Error:** Ensure you run the installer as administrator
- **Antivirus Warning:** Add SADSA to your antivirus exceptions list
- **Installation Fails:** Check that you have at least 500MB free disk space
- **Cannot Launch:** Install Microsoft Visual C++ Redistributable (included with installer)

Features & Menu Structure

File Menu

✏️ Manual Data Entry

- **Description:** Opens a spreadsheet-like window for direct manual data entry
- **Features:**
- Customizable grid dimensions (rows and columns)
- Navigation using Tab, Arrow keys
- Multi-cell selection with Alt+Arrow or Ctrl+Click
- Shift/Ctrl+Shift for text selection within cells
- **Usage:** Ideal for creating small datasets from scratch

Open Data File

- **Description:** Import data from various file formats
- **Supported Formats:** CSV, Excel (.xlsx, .xls), JSON, TSV, Parquet, LibreOffice Calc (.ods)
- **Usage:** Select file → Data loads into grid view
- **Note:** Auto-detects file format based on extension

Advanced File Import

- **Description:** Comprehensive file import dialog with advanced options and live preview
- **Features:**
- Multiple format tabs (Standard Formats)
- Live preview table before importing
- CRUD functionality for data manipulation
- Resizable preview panels
- Custom delimiter options
- **Supported Formats:** CSV, Excel, TSV, Parquet, JSON, LibreOffice Calc

Open CSV File (Legacy)

- **Description:** Legacy CSV import method for backward compatibility
- **Note:** Redirects to the standard Open Data File function

Text Input

- **Description:** Modern text input window for manual text entry or file loading
- **Features:**
- Direct text entry with line-by-line processing
- Load text from external files
- Multi-line paste support
- Placeholder guidance for new users
- **Usage:** Enter text manually or load from file, each line treated as separate entry

Save Data

- **Description:** Save current data from the grid into memory
- **Usage:** Captures all edits made in the data grid view
- **Note:** Use this to persist changes made directly in the grid

Save Data File

- **Description:** Export current dataset to various file formats with row/column selection
- **Supported Formats:**
- CSV Files (*.csv)
- Excel Files (*.xlsx)
- Tab Separated Values (*.tsv)
- Pipe Delimited (*.txt)
- Semicolon Delimited (*.csv)
- JSON Files (*.json)
- Parquet Files (*.parquet)
- LibreOffice Calc (*.ods)
- SADSA Encrypted (*.sadsa)
- **Features:**
- Row/column selection dialog before saving
- Save all data or specific subsets
- Custom delimiter options for text formats

Clear Data

- **Description:** Remove all loaded data from the application
- **Usage:** Clears the data grid and resets the dataset
- **Note:** This action cannot be undone

Python Console

- **Description:** Interactive Python shell integrated into the application
- **Features:**
- Direct access to loaded DataFrame via `df` variable
- Execute custom Python code without leaving the application
- Full access to all data analysis libraries
- Multi-line code support with Ctrl+Return execution
- **See:** Python Console section below for detailed documentation

Settings (Submenu)

- **Description:** Application settings including theme customization
- **Available Themes:**
- **Light Themes:** Modern Light, Ocean Blue, Forest Green, Purple Dream, Sunset Orange, Rose Pink, Classic
- **Dark Themes:** Dark Mode, Midnight Blue
- **Usage:** Select a theme to instantly change the application's appearance

Exit

- **Description:** Close the SADSA application
- **Features:** Confirmation dialog before exiting
- **Note:** Prompts user to confirm before closing

💻 Python Console

- Interactive Python shell integrated into the application.
- Direct access to loaded DataFrame via `df` variable.
- Execute custom Python code without leaving the application.
- Full access to all data analysis libraries (pandas, numpy, scipy, scikit-learn, etc.).
- Multi-line code support with Ctrl+Return execution.
- Call any app function directly from the console.
- Perfect for exploratory data analysis and custom operations.

Console Methods & Features

Core Methods

- `__init__(parent, app_instance, title="Python Console")` - Initializes the Python console window with parent window reference and application instance.
- `setup_ui()` - Creates the console user interface with output area, input area, buttons, and information panel.
- `setup_console_environment()` - Sets up the Python environment with pre-imported libraries and displays welcome message with available objects and examples.

Code Execution Methods

- `execute_code(event=None)` - Executes Python code from input area with error handling and output capture via Ctrl+Return or Execute button.
- `_import_module(module_name)` - Imports a module by name and handles import errors gracefully.

Output & Display Methods

- `print_output(text, tag="info")` - Prints text to the output area with color coding (info=blue, error=red, success=green).
- `clear_output()` - Clears all text from the output display area.
- `clear_input()` - Clears all text from the code input area.
- `show_help()` - Displays a comprehensive help window with available objects, commands, and usage examples.

Pre-imported Libraries & Objects

- `df` - Current DataFrame from your loaded data, accessible for analysis and manipulation.
- `app` - Reference to the SADSA application instance for accessing app methods and data.
- `pd` - Pandas library for data manipulation and analysis.
- `np` - NumPy library for numerical computations and array operations.
- `plt` - Matplotlib.pyplot library for creating plots and visualizations.
- `sns` - Seaborn library for statistical data visualization.
- `stats` - SciPy.stats library for statistical functions and distributions.

Python Console

SADSA includes an **interactive Python console** for advanced users who want to perform custom data analysis, write custom code, or explore data interactively.

How to Access

- Navigate to **Help** → **Python Console** from the menu bar
- A new window will open with the Python shell

Key Features

- **Full Python Environment**: Execute any Python code directly
- **DataFrame Access**: Access your loaded data via the `df` variable
- **App Integration**: Reference the application via `app` to call any app function
- **Pre-imported Libraries**:
- `pandas` (pd)
- `numpy` (np)
- `scipy` , `matplotlib` , `scikit-learn`
- `statsmodels` , `networkx` and more

Common Operations

```
# View data
df.head()
df.describe()
df.info()

# Data manipulation
df['new_col'] = df['col1'] + df['col2']
df_filtered = df[df['col'] > 100]
df_grouped = df.groupby('category').sum()

# Statistical analysis
correlation = df.corr()
mean_val = df['col'].mean()
std_val = df['col'].std()

# Call app functions from console
app.perform_correlation_analysis()  # Performs correlation analysis on the datas
app.data  # Access current data from app

# ==============================================================================
#                          FILE OPERATIONS
# ==============================================================================
app.open_file()                     # Opens file dialog to load data (CSV, Exce
app.open_csv()                      # Opens a file dialog to load CSV data (leg
app.save_data()                     # Saves current data in memory (grid edits)
app.save_data_file()                # Saves data to CSV, Excel, TSV, JSON, Parq
app.clear_data()                    # Clears all loaded data
app.display_data()                  # Refreshes the data display in the grid

# ==============================================================================
#                          DATA EDITING
# ==============================================================================
app.rename_columns()                # Renames one or more columns in the datase
app.compute_variable()              # Creates new computed variables from exist
app.recode_variable()               # Recodes/transforms existing variables
app.missing_data_treatment()        # Handles missing values using various meth
app.set_values()                    # Sets fixed values in the dataset
app.show_features()                 # Displays feature information

# ==============================================================================
#                          TRANSFORMATIONS
# ==============================================================================
app.data_simulations()              # Generates simulated/synthetic data
app.simulate_time_series_data()     # Generates simulated time series data
app.generate_multivariate_normal()  # Generates multivariate normal distributio
app.show_decomposition_window()     # Opens matrix decomposition window
app.show_standardization_options()  # Opens standardization options dialog
app.perform_standardization("Min-Max")     # Applies Min-Max scaling to data
app.perform_standardization("Z-Score")      # Applies Z-Score standardization to
app.perform_standardization("Decimal Scaling") # Applies Decimal Scaling transfo
app.perform_standardization("Log")          # Applies Log transformation to data
app.perform_standardization("Log-Normal")   # Applies Log-Normal transformation

# Matrix Decomposition methods (via show_decomposition_window)
# Cholesky, QR, SVD, Eigenvalue decomposition available

# ══════════════════════════════════════════════════════════════════════════════
#                          DESCRIPTIVE STATISTICS
# ══════════════════════════════════════════════════════════════════════════════
app.show_frequencies_and_summaries() # Displays frequency tables and summary sta

# ══════════════════════════════════════════════════════════════════════════════
#                          INFERENTIAL STATISTICS
# ══════════════════════════════════════════════════════════════════════════════
app.show_ttest_options()             # Opens T-Test options dialog
app.perform_ttest()                  # Performs t-test (one-sample, two-sample, 
app.show_chisquare_options()         # Opens Chi-Square options dialog
app.perform_chisquare()              # Performs Chi-Square test for independence
app.perform_normality_tests()        # Tests for data normality (Shapiro-Wilk, K
app.perform_anova()                  # Performs ANOVA (Analysis of Variance) tes
app.perform_manova()                 # Performs MANOVA (Multivariate ANOVA) test

# ══════════════════════════════════════════════════════════════════════════════
#                          EXPLORATORY ANALYSIS
# ══════════════════════════════════════════════════════════════════════════════
app.perform_correspondence_analysis() # Performs Correspondence Analysis (Simple
app.perform_mds()                    # Performs Multidimensional Scaling
app.perform_pca()                    # Performs Principal Component Analysis

# ══════════════════════════════════════════════════════════════════════════════
#                          CORRELATION & REGRESSION
# ══════════════════════════════════════════════════════════════════════════════
app.perform_correlation_analysis()   # Performs correlation analysis (Pearson, S
app.perform_regression_analysis()    # Performs linear/multiple regression analy

# ══════════════════════════════════════════════════════════════════════════════
#                          FACTOR ANALYSIS
# ══════════════════════════════════════════════════════════════════════════════
app.perform_efa()                    # Performs Exploratory Factor Analysis
app.perform_cfa()                    # Performs Confirmatory Factor Analysis
app.perform_mediation_analysis()     # Performs Mediation Analysis
app.perform_plssem()                 # Performs PLS-SEM (Partial Least Squares S

# ══════════════════════════════════════════════════════════════════════════════
#                          CLUSTER ANALYSIS
# ══════════════════════════════════════════════════════════════════════════════
app.perform_kmeans()                 # Performs K-Means clustering on the data
app.perform_hierarchical()           # Performs Hierarchical clustering on the d

# ══════════════════════════════════════════════════════════════════════════════
#                          TIME SERIES ANALYSIS
# ════════════════════════════════════════════════════════════════════════════════
app.perform_stationarity_tests()     # Tests for stationarity (ADF, KPSS)
app.perform_seasonal_decomposition() # Decomposes time series into trend, seasona
app.perform_holt_winters()           # Applies Holt-Winters exponential smoothin
app.perform_moving_averages()        # Calculates moving averages for time serie
app.show_arma_arima_menu()           # Opens ARMA/ARIMA model menu
app.show_multivariate_ts_menu()      # Opens Multivariate Time Series menu (VAR,

# ════════════════════════════════════════════════════════════════════════════════
#                          MACHINE LEARNING - SUPERVISED
# ════════════════════════════════════════════════════════════════════════════════
app.perform_logistic_regression()    # Builds logistic regression classification
app.perform_decision_tree()          # Builds decision tree classification model
app.perform_random_forest()          # Builds random forest classification/regre
app.perform_naive_bayes()            # Builds Naive Bayes classification model
app.perform_knn()                    # Builds K-Nearest Neighbors classification
app.perform_svm()                    # Builds Support Vector Machine classificat
app.perform_neural_network()         # Builds neural network classification/regr
app.perform_recommender_supervised() # Builds supervised recommender system

# Ensemble Methods
app.perform_xgboost()                # Builds XGBoost model
app.perform_lightgbm()               # Builds LightGBM model
app.perform_adaboost()               # Builds AdaBoost model
app.perform_gradient_boosting()      # Builds Gradient Boosting model
app.perform_voting_ensemble()        # Builds Voting Ensemble model
app.perform_stacking_ensemble()      # Builds Stacking Ensemble model
app.perform_bagging_ensemble()       # Builds Bagging Ensemble model

# ════════════════════════════════════════════════════════════════════════════════
#                          MACHINE LEARNING - UNSUPERVISED
# ════════════════════════════════════════════════════════════════════════════════
app.perform_kmeans_ml()              # K-Means clustering (ML interface)
app.perform_hierarchical_ml()        # Hierarchical clustering (ML interface)
app.perform_dbscan()                 # DBSCAN density-based clustering
app.perform_pca_ml()                 # PCA dimensionality reduction (ML interfac
app.perform_factor_analysis()        # Factor Analysis (ML interface)
app.perform_gmm()                    # Gaussian Mixture Model clustering
app.perform_association_rules()      # Association Rules mining (Apriori)
app.perform_recommender_unsupervised() # Unsupervised recommender system

# ════════════════════════════════════════════════════════════════════════════════
#                          NLP (Natural Language Processing)
# ════════════════════════════════════════════════════════════════════════════════
app.generate_dtm()                   # Generates Document-Term Matrix
app.extract_features()               # Extracts text features (TF-IDF, etc.)
app.generate_term_frequencies()      # Generates term frequency analysis
app.load_text_file()                 # Loads text file for NLP processing

# ════════════════════════════════════════════════════════════════════════════════
#                          META ANALYSIS
# ══════════════════════════════════════════════════════════════════════════════
app.perform_fixed_effects()          # Performs Fixed Effects meta-analysis
app.perform_random_effects()         # Performs Random Effects meta-analysis
app.perform_heterogeneity_tests()    # Performs heterogeneity tests (Q, I², H²)
app.perform_publication_bias_tests() # Performs publication bias tests (Egger's,
app.perform_masem()                  # Performs Meta-Analytic Structural Equatio

# ══════════════════════════════════════════════════════════════════════════════
#                          BIBLIOMETRICS
# ══════════════════════════════════════════════════════════════════════════════
app.analyze_basic_metrics()          # Analyzes basic bibliometric metrics
app.keyword_analysis()               # Performs keyword analysis
app.citation_analysis()              # Performs citation analysis
app.co_authorship_analysis()         # Performs co-authorship network analysis
app.bibliometric_coupling()          # Performs bibliometric coupling analysis
app.word_count_analysis()            # Performs word count analysis

# ══════════════════════════════════════════════════════════════════════════════
#                          VISUALIZATION & DISPLAY
# ══════════════════════════════════════════════════════════════════════════════
app.generate_plot()                  # Opens plot generation interface with vari
app.show_report_window(data, title)  # Displays data/results in a report window
app.show_modern_message(title, msg)  # Shows a message dialog with custom title 

# ══════════════════════════════════════════════════════════════════════════════
#                          CONSOLE & HELP
# ══════════════════════════════════════════════════════════════════════════════
app.open_python_console()            # Opens the interactive Python console wind
help()                               # Shows help guide in console
methods()                            # Shows compact list of all methods
```

Multi-line Code

- Write multiple lines of code
- Press **Ctrl+Return** to execute
- Use proper Python indentation for blocks

Edit Menu

Copy

- **Description:** Copy selected cells to clipboard
- **Shortcut:** Ctrl+C

Paste

- **Description:** Paste data from clipboard into grid
- **Shortcut:** Ctrl+V

Delete Row

- **Description:** Remove selected row(s) from dataset
- **Usage:** Select row → Delete → Confirm

Add Column

- **Description:** Insert new column with specified name and default value
- **Options:** Column name, data type, position

Transformations Menu

Standardization

- **Z-Score Standardization**
- Transforms data to mean=0, std=1
- Use for: Comparing variables on different scales
- **Min-Max Normalization**
- Scales data to range [0, 1]
- Use for: Neural networks, distance-based algorithms
- **Robust Scaling**
- Uses median and IQR (resistant to outliers)
- Use for: Data with extreme values

Data Generation

- **Multivariate Normal Distribution**
- Generate synthetic data from normal distribution
- Options: Variable names, means, covariance matrix, sample size

Matrix Decomposition

- **Cholesky Decomposition**
- Decomposes positive-definite matrix
- Use for: Linear algebra operations, optimization
- **QR Decomposition**
- Orthogonal-triangular factorization
- Use for: Solving linear systems, eigenvalue problems
- **SVD (Singular Value Decomposition)**
- Decomposes matrix into U, Σ, V components
- Use for: Dimensionality reduction, recommender systems
- **Eigenvalue Decomposition**
- Computes eigenvalues and eigenvectors
- Use for: PCA, spectral analysis

Data Analytics Menu

Descriptive Statistics

Univariate Statistics

- **Description:** Summary statistics for single variables
- **Outputs:** Mean, median, mode, std dev, variance, skewness, kurtosis, min, max, quartiles
- **Plots:** Histogram, box plot, Q-Q plot

Bivariate Statistics

- **Description:** Relationships between two variables
- **Outputs:** Cross-tabulation, correlation, contingency tables
- **Plots:** Scatter plots, grouped bar charts

Multivariate Statistics

- **Description:** Analysis of multiple variables simultaneously
- **Outputs:** Correlation matrix, covariance matrix, partial correlations

- **Plots:** Correlation heatmap, pair plot, 3D scatter

Inferential Statistics

T-Test

- **Independent Samples T-Test**
- Compares means of two independent groups
- Outputs: T-statistic, p-value, confidence interval
- Plot: Box plots comparing groups
- **Paired Samples T-Test**
- Compares means of related groups
- Use for: Before/after studies, matched pairs
- **One-Sample T-Test**
- Tests if sample mean differs from population mean
- Outputs: T-statistic, p-value

Chi-Square Test

- **Pearson Chi-Square**
- Tests independence in contingency tables
- Outputs: χ^2 statistic, p-value, degrees of freedom
- **Fisher's Exact Test**
- Exact test for 2×2 tables (small sample sizes)
- Use for: Small cell counts (<5)
- **McNemar's Test**
- Tests marginal homogeneity (paired data)
- Use for: Before/after categorical data
- **Yates' Correction**
- Continuity correction for chi-square
- Use for: 2×2 tables with small samples
- **Likelihood Ratio Chi-Square**
- Alternative to Pearson chi-square

- Better for small expected frequencies
- **Mantel-Haenszel Chi-Square**
- Tests association controlling for confounders
- Use for: Stratified 2×2 tables
- **G-Test (Alternative)**
- Likelihood ratio test for independence
- More accurate for small samples

Normality Tests

- **Shapiro-Wilk Test**
- Most powerful normality test
- Best for: Sample sizes < 2000
- **Kolmogorov-Smirnov Test**
- Tests fit to any distribution
- Use for: Large samples, known distribution
- **Anderson-Darling Test**
- Gives more weight to tails
- Use for: Detecting outliers
- **D'Agostino K^2 Test**
- Based on skewness and kurtosis
- Use for: Larger samples (n > 20)
- **Jarque-Bera Test**
- Tests skewness and kurtosis
- Use for: Financial data, regression residuals
- **Lilliefors Test**
- Modified K-S test (parameters estimated)
- Use for: When parameters unknown

ANOVA (Analysis of Variance)

- **Description:** Compares means across multiple groups
- **Outputs:** F-statistic, p-value, group means, post-hoc tests

- **Assumptions:** Normality, homogeneity of variance
- **Plot:** Box plots, means plot

MANOVA (Multivariate ANOVA)

- **Description:** Tests differences on multiple dependent variables
- **Outputs:** Wilks' Lambda, Pillai's trace, Roy's largest root
- **Use for:** Multiple outcome variables

Exploratory Analysis

Correspondence Analysis

- **Simple Correspondence Analysis**
- Analyzes relationships in contingency tables
- Outputs: Row/column coordinates, inertia, chi-square
- Plot: Biplot of rows and columns
- **Multiple Correspondence Analysis**
- Extension for multiple categorical variables
- Outputs: Variable coordinates, contributions
- Plot: Category space visualization
- **Canonical Correspondence Analysis**
- Relates two sets of variables
- Outputs: Canonical correlations, coefficients
- Plot: Canonical variates plot

Multidimensional Scaling (MDS)

- **Description:** Visualizes dissimilarities in low dimensions
- **Options:**
- Metric MDS (preserves distances)
- Non-metric MDS (preserves rank order)
- **Distance Metrics:** Euclidean, Manhattan, Cosine, Correlation
- **Outputs:** Stress, RSQ, coordinates
- **Plot:** 2D/3D point plot

Principal Components Analysis (PCA)

- **Description:** Reduces dimensionality while preserving variance
- **Outputs:**
- Principal components
- Eigenvalues
- Explained variance ratio
- Component loadings
- **Plots:** Scree plot, biplot, component loadings

Correlation & Regression

Correlation Analysis

- **Pearson Correlation**
- Linear relationship between continuous variables
- Range: -1 to +1
- Outputs: Correlation coefficient, p-value
- **Spearman Correlation**
- Monotonic relationship (rank-based)
- Use for: Non-linear relationships, ordinal data
- **Kendall Correlation**
- Rank correlation (tau)
- More robust than Spearman for small samples
- **Canonical Correlation**
- Relationship between two sets of variables
- Outputs: Canonical correlations, variates
- Plot: Canonical variates

Options: - P-values and significance testing - Confidence intervals - Explained variance

Covariance Analysis

- **Description:** Analyzes variance-covariance matrices
- **Tests Available:**
- Box M Test (homogeneity of covariance matrices)
- Bartlett's Test (sphericity)

- Levene's Test (homogeneity of variance)
- Permutation Test (non-parametric)
- **Fit Measures:**
- Covariance Matrix
- Eigenvalues
- Determinant
- Trace
- Condition Number

Regression Analysis

- **Simple Linear Regression**
- One predictor, one outcome
- Outputs: Coefficients, R^2, p-values, residuals
- Plot: Scatter with regression line
- **Multiple Linear Regression**
- Multiple predictors
- Outputs: Coefficients, adjusted R^2, F-statistic
- Plot: Residual plots, Q-Q plot
- **Generalized Linear Model (GLM)**
- Non-normal outcomes (binomial, Poisson, etc.)
- Outputs: Coefficients, deviance, AIC
- Link functions: Logit, log, identity

Factor Analysis

Exploratory Factor Analysis (EFA)

- **Description:** Identifies latent factors
- **Tests:**
- Bartlett's Test of Sphericity
- Kaiser-Meyer-Olkin (KMO) test
- **Outputs:**
- Factor loadings
- Communalities
- Eigenvalues

- **Rotation:** Varimax, Promax
- **Plots:** Scree plot, factor loadings heatmap

Confirmatory Factor Analysis (CFA)

- **Description:** Tests hypothesized factor structure
- **Features:**
- Specify factor structure
- Define relationships between factors
- Estimate raw or standardized coefficients
- **Outputs:**
- Factor loadings
- Fit indices (CFI, TLI, RMSEA, SRMR)
- Modification indices
- **Plots:** Path diagram with estimates

Mediation Analysis

- **Description:** Tests indirect effects through mediator
- **Model:** X → M → Y
- **Outputs:**
- Direct effects
- Indirect effects
- Total effects
- Sobel test
- **Plot:** Mediation model diagram

PLS-SEM Analysis (Partial Least Squares SEM)

- **Description:** Variance-based structural equation modeling
- **Advantages:**
- Works with small sample sizes
- Handles non-normal data
- Suitable for exploratory research
- **Features:**
- Define measurement model (indicators → constructs)
- Define structural model (construct relationships)
- Bootstrapping for significance testing
- **Outputs:**

- Path coefficients
- R^2 values for endogenous constructs
- Factor loadings
- Construct reliability (Cronbach's α, Composite Reliability)
- AVE (Average Variance Extracted)
- Discriminant validity (Fornell-Larcker, HTMT)
- **Plots:** Path diagram with estimates, reliability charts
- **Use for:** Marketing research, management studies, theory development

Cluster Analysis

K-Means Clustering

- **Description:** Partitions data into K clusters
- **Options:**
- Number of clusters (k)
- Initialization method
- Max iterations
- **Outputs:**
- Cluster assignments
- Cluster centers
- Within-cluster sum of squares
- Silhouette score
- **Plot:** Cluster scatter plot, elbow plot

Hierarchical Clustering

- **Description:** Creates tree of clusters
- **Methods:** Ward, complete, average, single linkage
- **Outputs:**
- Dendrogram
- Cluster assignments at cut height
- Cophenetic correlation
- **Plot:** Dendrogram

Time Series Analysis

Stationarity Tests

- **Augmented Dickey-Fuller (ADF) Test**
- Tests for unit root (non-stationarity)
- Outputs: Test statistic, p-value, critical values
- Interpretation: $p < 0.05$ suggests stationarity
- **KPSS Test**
- Tests null hypothesis of stationarity
- Complementary to ADF test
- Use both for robust conclusion

Seasonal Decomposition

- **Description:** Separates time series into components
- **Components:**
- Trend
- Seasonal pattern
- Residual (irregular)
- **Models:** Additive, multiplicative
- **Plot:** Decomposition plot (4 subplots)

Holt-Winters Method

- **Description:** Exponential smoothing with trend and seasonality
- **Options:**
- Trend type (additive, multiplicative, none)
- Seasonal type (additive, multiplicative, none)
- Seasonal period
- **Outputs:**
- Forecasts
- Fitted values
- MSE, MAE, RMSE
- **Plot:** Actual vs. forecast

Moving Averages

- **Description:** Smooths data using rolling window

- **Options:** Window size
- **Outputs:**
- Smoothed series
- MSE, MAE, RMSE
- **Plot:** Original vs. smoothed

ARMA/ARIMA

- **Description:** Autoregressive Integrated Moving Average models for time series forecasting
- **Models:**
- **ARMA:** For stationary time series (AR + MA components)
- **ARIMA:** For non-stationary time series (includes differencing)
- **Options:**
- AR order (p)
- Differencing order (d)
- MA order (q)
- Seasonal parameters (P, D, Q, S) for SARIMA
- **Outputs:**
- Model coefficients
- AIC, BIC criteria
- Forecasts with confidence intervals
- Residual diagnostics
- **Plots:** Forecast plot, ACF/PACF plots, residual diagnostics

Bi/Multivariate Time Series

- **Description:** Analysis of multiple related time series
- **Methods:**
- **VAR (Vector Autoregression):** Models interdependencies between multiple time series
- **Granger Causality:** Tests if one time series helps predict another
- **Cointegration Analysis:** Tests for long-run equilibrium relationships
- **Impulse Response Functions:** Analyzes dynamic effects of shocks
- **Outputs:**
- VAR coefficients
- Granger causality p-values
- Cointegration test results
- Impulse response plots
- **Use for:** Economic data, financial markets, multivariate forecasting

Machine Learning Menu

Supervised Learning

Logistic Regression

- **Description:** Binary classification using logistic function
- **Options:**
- Dependent variable (binary)
- Independent variables (multiple)
- Train/test split ratio
- Regularization (L1, L2)
- Max iterations
- **Outputs:**
- Coefficients
- Confusion matrix
- Accuracy, precision, recall, F1-score
- Classification report
- ROC curve and AUC
- **Plots:**
- ROC curve
- Confusion matrix heatmap
- Predicted vs. actual
- **Prediction:** Paste new data or upload CSV for predictions

Decision Tree

- **Description:** Tree-based classifier
- **Options:**
- Max depth
- Min samples split
- Criterion (gini, entropy)
- **Outputs:**
- Tree visualization
- Feature importance
- Confusion matrix
- Accuracy metrics

- **Plot:** Decision tree diagram

Random Forest

- **Description:** Ensemble of decision trees
- **Options:**
- Number of trees
- Max depth
- Feature subset size
- **Outputs:**
- Feature importance
- Out-of-bag score
- Confusion matrix
- Accuracy metrics
- **Plot:** Feature importance bar chart

Naive Bayes

- **Description:** Probabilistic classifier based on Bayes' theorem
- **Types:** Gaussian, Multinomial, Bernoulli
- **Outputs:**
- Prior probabilities
- Confusion matrix
- Accuracy metrics

K-Nearest Neighbors (KNN)

- **Description:** Instance-based classifier
- **Options:**
- Number of neighbors (k)
- Distance metric
- Weight function
- **Outputs:**
- Confusion matrix
- Accuracy metrics
- **Plot:** Decision boundaries (2D)

Support Vector Machine (SVM)

- **Description:** Maximum margin classifier

- **Options:**
- Kernel (linear, RBF, polynomial)
- C parameter (regularization)
- Gamma (kernel coefficient)
- **Outputs:**
- Support vectors
- Confusion matrix
- Accuracy metrics
- **Plot:** Decision boundaries with support vectors

Neural Network

- **Description:** Multi-layer perceptron classifier
- **Options:**
- Hidden layer sizes
- Activation function
- Learning rate
- Max iterations
- **Outputs:**
- Loss curve
- Confusion matrix
- Accuracy metrics
- **Plot:** Training loss over iterations

Recommender System (Supervised)

- **Description:** Content-based recommendation using supervised learning
- **Options:**
- Feature columns for item/user profiles
- Target variable (ratings/preferences)
- Model type selection
- **Outputs:**
- Predicted ratings
- Top-N recommendations
- Recommendation accuracy metrics
- **Use for:** Personalized product/content recommendations

Unsupervised Learning

K-Means Clustering

- **Description:** Partitions data into K clusters
- **Options:**
- Number of clusters
- Distance metric
- Initialization method
- **Outputs:**
- Cluster assignments
- Cluster centers
- Silhouette score
- Inertia (within-cluster sum of squares)
- **Plot:** Cluster scatter plot
- **Prediction:** Assign new data points to nearest cluster

Hierarchical Clustering

- **Description:** Builds hierarchy of clusters
- **Options:**
- Linkage method (ward, average, complete, single)
- Distance metric
- **Outputs:**
- Dendrogram
- Cluster assignments
- Cophenetic correlation
- **Plot:** Dendrogram tree
- **Prediction:** Assign new points to nearest cluster centroid

DBSCAN Clustering

- **Description:** Density-based clustering
- **Options:**
- Epsilon (neighborhood radius)
- Min samples (core point threshold)
- **Outputs:**
- Cluster assignments
- Core samples

- Noise points (-1 label)
- **Plot:** Cluster scatter with noise points
- **Prediction:** Label new points based on neighborhood density

Principal Component Analysis (PCA)

- **Description:** Linear dimensionality reduction
- **Options:**
- Number of components
- Standardization
- **Outputs:**
- Transformed data
- Explained variance ratio
- Component loadings
- Eigenvalues
- **Plots:** Scree plot, biplot, component loadings
- **Prediction:** Transform new data into principal component space

Factor Analysis

- **Description:** Latent variable analysis
- **Options:**
- Number of factors
- Rotation method
- **Outputs:**
- Factor loadings
- Communalities
- Factor scores
- **Plot:** Factor loadings heatmap
- **Prediction:** Compute factor scores for new observations

Gaussian Mixture Models (GMM)

- **Description:** Probabilistic clustering using mixture of Gaussians
- **Options:**
- Number of components
- Covariance type
- Max iterations
- **Outputs:**

- Cluster assignments
- Cluster probabilities
- BIC, AIC scores
- Means and covariances
- **Plot:** Cluster scatter with probability contours
- **Prediction:** Predict cluster probabilities for new data

Association Rule Learning (Market Basket)

- **Description:** Discovers relationships between items in transactions using Apriori algorithm
- **Options:**
- Minimum support threshold
- Minimum confidence threshold
- Minimum lift threshold
- **Outputs:**
- Frequent itemsets
- Association rules (antecedent → consequent)
- Support, confidence, lift metrics
- **Plots:** Support vs. confidence scatter, top rules by lift
- **Use for:** Market basket analysis, cross-selling recommendations, inventory management

Recommender System (Collaborative Filtering)

- **Description:** Unsupervised recommendation based on user-item interactions
- **Methods:**
- User-based collaborative filtering
- Item-based collaborative filtering
- Matrix factorization
- **Outputs:**
- Similarity matrices
- Top-N recommendations per user
- Predicted ratings
- **Use for:** Movie/product recommendations, content suggestions

Ensemble, Boosting & Bagging Methods

XGBoost (Extreme Gradient Boosting)

- **Description:** High-performance gradient boosting framework

- **Options:**
- Number of estimators
- Max depth
- Learning rate
- Regularization parameters (L1, L2)
- **Outputs:**
- Feature importance
- Confusion matrix
- Accuracy, precision, recall, F1-score
- **Plot:** Feature importance chart, ROC curve
- **Prediction:** Classify new observations

LightGBM (Light Gradient Boosting)

- **Description:** Fast, distributed gradient boosting framework
- **Options:**
- Number of leaves
- Max depth
- Learning rate
- Feature fraction
- **Outputs:**
- Feature importance
- Confusion matrix
- Accuracy metrics
- **Plot:** Feature importance, training history
- **Use for:** Large datasets, high-dimensional data

AdaBoost (Adaptive Boosting)

- **Description:** Iteratively trains weak learners, focusing on misclassified samples
- **Options:**
- Number of estimators
- Learning rate
- Base estimator type
- **Outputs:**
- Estimator weights
- Feature importance
- Confusion matrix

- Accuracy metrics
- **Plot:** Training error over iterations

Gradient Boosting Classifier

- **Description:** Sequential ensemble that minimizes loss function
- **Options:**
- Number of estimators
- Max depth
- Learning rate
- Subsample ratio
- **Outputs:**
- Feature importance
- Staged predictions
- Confusion matrix
- Accuracy metrics
- **Plot:** Feature importance, learning curve

Voting Ensemble (Multiple Models)

- **Description:** Combines predictions from multiple classifiers
- **Voting Types:**
- Hard voting (majority vote)
- Soft voting (probability averaging)
- **Included Models:** Logistic Regression, Random Forest, SVM, etc.
- **Outputs:**
- Individual model accuracies
- Ensemble accuracy
- Confusion matrix
- **Use for:** Improving prediction robustness

Stacking Ensemble (Meta-Learning)

- **Description:** Two-level learning with meta-classifier
- **Structure:**
- Level 0: Base classifiers (diverse models)
- Level 1: Meta-classifier learns from base predictions
- **Outputs:**
- Base model predictions

- Meta-model accuracy
- Confusion matrix
- **Use for:** Complex classification tasks

Bagging Classifier (Bootstrap Aggregating)

- **Description:** Trains multiple instances on bootstrap samples
- **Options:**
- Number of estimators
- Max samples
- Max features
- Bootstrap with/without replacement
- **Outputs:**
- Out-of-bag score
- Feature importance
- Confusion matrix
- Accuracy metrics
- **Use for:** Reducing variance, improving stability

NLP (Natural Language Processing) Menu

DTM (Document-Term Matrix)

- **Description:** Creates a matrix of document-term frequencies
- **Options:**
- Text column selection
- Minimum document frequency
- Maximum document frequency
- N-gram range
- **Outputs:**
- Document-Term Matrix (sparse matrix)
- Vocabulary list
- Term frequencies
- **Use for:** Text preprocessing for topic modeling, classification

Extract Features

- **Description:** Extracts linguistic and statistical features from text
- **Features Extracted:**
- Word count, sentence count
- Average word length
- Vocabulary richness
- Part-of-speech distribution
- Readability scores
- **Outputs:** Feature matrix with text statistics
- **Use for:** Text classification, author identification

Sentiment Analysis

- **Description:** Classifies text sentiment (positive, negative, neutral)
- **Methods:** VADER, TextBlob
- **Outputs:**
- Sentiment scores (positive, negative, neutral, compound)
- Sentiment classification
- Sentiment distribution statistics
- **Plot:** Sentiment distribution chart

Named Entity Recognition

- **Description:** Identifies and classifies named entities in text
- **Entity Types:** Person, Organization, Location, Date, Money, etc.
- **Outputs:**
- Entity list with types
- Entity frequency counts
- Entity context
- **Plot:** Entity type distribution

Topic Modeling (LDA)

- **Description:** Discovers latent topics in document collection using Latent Dirichlet Allocation
- **Options:**
- Number of topics
- Number of iterations
- Alpha/Beta parameters

- **Outputs:**
- Topic distributions per document
- Top words per topic
- Topic coherence scores
- **Plot:** Topic word clouds, topic distribution

Text Similarity Analysis

- **Description:** Measures similarity between text documents
- **Methods:**
- Cosine similarity
- Jaccard similarity
- TF-IDF based similarity
- **Outputs:**
- Similarity matrix
- Most similar document pairs
- Similarity scores
- **Plot:** Similarity heatmap
- **Use for:** Document clustering, duplicate detection, plagiarism checking

Word Frequency Analysis

- **Description:** Analyzes and visualizes word frequencies in text
- **Options:**
- Remove stopwords
- Minimum word length
- Top N words to display
- **Outputs:**
- Word frequency table
- Cumulative frequency
- **Plots:** Bar chart, word cloud

N-gram Analysis

- **Description:** Analyzes sequences of N consecutive words/characters
- **Options:**
- N-gram size (bigrams, trigrams, etc.)
- Minimum frequency threshold
- Character or word level

- **Outputs:**
- N-gram frequency table
- Top N-grams
- N-gram context
- **Plot:** N-gram frequency bar chart
- **Use for:** Phrase extraction, language modeling, text prediction

---

Meta Analysis Menu

Fixed Effects Model

- **Description:** Assumes all studies share a common true effect size
- **Inputs:**
- Effect sizes (e.g., Cohen's d, correlation r, odds ratio)
- Standard errors or sample sizes
- **Outputs:**
- Pooled effect size
- 95% Confidence interval
- Z-statistic and p-value
- Weights per study
- **Plot:** Forest plot
- **Use for:** Homogeneous studies, single population inference

Random Effects Model

- **Description:** Assumes true effect sizes vary across studies
- **Inputs:**
- Effect sizes
- Standard errors or sample sizes
- **Outputs:**
- Pooled effect size
- 95% Confidence interval
- Tau^2 (between-study variance)
- I^2 (heterogeneity percentage)
- Prediction interval
- **Plot:** Forest plot with prediction interval

- **Use for:** Heterogeneous studies, generalizing across populations

Tests (Submenu)

Heterogeneity Tests

- **Description:** Tests for variability in effect sizes across studies
- **Tests:**
- **Cochran's Q Test:** Chi-square test for heterogeneity
- **I^2 Statistic:** Percentage of variability due to heterogeneity
- **H^2 Statistic:** Ratio of total variability to sampling variability
- **Tau^2 Estimation:** Between-study variance (DerSimonian-Laird, REML, ML)
- **Outputs:**
- Q statistic and p-value
- I^2 with confidence interval
- Tau^2 estimate
- **Interpretation:** $I^2 > 50\%$ suggests substantial heterogeneity

Publication Bias Tests

- **Description:** Detects selective reporting of significant results
- **Tests:**
- **Funnel Plot:** Visual inspection of asymmetry
- **Egger's Test:** Regression test for funnel plot asymmetry
- **Begg's Test:** Rank correlation test
- **Trim and Fill:** Adjusts for missing studies
- **Fail-Safe N:** Number of null studies needed to nullify results
- **Outputs:**
- Test statistics and p-values
- Adjusted effect size (trim and fill)
- Funnel plot
- **Use for:** Assessing reliability of meta-analytic conclusions

Meta-Analytic SEM (MASEM)

- **Description:** Combines meta-analysis with structural equation modeling
- **Features:**
- Pool correlation matrices across studies
- Fit SEM models to pooled matrices

- Test path models and mediation
- **Inputs:**
- Correlation matrices from multiple studies
- Sample sizes per study
- Hypothesized structural model
- **Outputs:**
- Pooled correlation matrix
- Path coefficients
- Fit indices (CFI, RMSEA, SRMR)
- Direct and indirect effects
- **Use for:** Testing theoretical models across multiple studies

Convert Matrix to Long Format

- **Description:** Transforms correlation/covariance matrices for meta-analysis
- **Inputs:** Square correlation matrix
- **Outputs:** Long-format data with variable pairs and correlations
- **Use for:** Preparing data for MASEM, pooling correlations

---

Bibliometrics Menu

Analyze Basic Metrics

- **Description:** Calculates fundamental bibliometric indicators
- **Metrics:**
- Total publications count
- Publication year distribution
- Author productivity statistics
- Journal distribution
- Document types breakdown
- **Outputs:**
- Summary statistics table
- Publication trends over time
- **Plots:** Publication timeline, author distribution

Keyword Analysis

- **Description:** Extracts and analyzes keywords from documents
- **Methods:**
- Author keywords
- Index keywords
- TF-IDF extraction
- **Outputs:**
- Keyword frequency table
- Keyword co-occurrence matrix
- Trending keywords over time
- **Plot:** Keyword cloud, keyword trends

Citation Analysis

- **Description:** Analyzes citation patterns
- **Outputs:**
- Citation counts per author/paper
- H-index
- Citation network
- **Plot:** Citation network graph

Co-Authorship Analysis

- **Description:** Examines collaboration patterns
- **Outputs:**
- Co-authorship network
- Collaboration statistics
- **Plot:** Network graph of authors

Bibliometric Coupling

- **Description:** Measures similarity based on shared references
- **Outputs:**
- Coupling strength matrix
- Document clusters
- **Plot:** Coupling network

Word Count Analysis

- **Description:** Analyzes keyword frequencies
- **Outputs:** Word frequencies, trending terms
- **Plot:** Bar chart, word cloud

Keyword Analysis

- **Description:** Extracts and analyzes keywords from documents
- **Methods:** TF-IDF, keyword extraction
- **Plot:** Keyword cloud

Plots Menu

Basic Plots

- **Histogram:** Distribution of continuous variables
- **Box Plot:** Five-number summary with outliers
- **Scatter Plot:** Relationship between two variables
- **Line Plot:** Trends over time/sequence
- **Bar Chart:** Comparisons across categories
- **Pie Chart:** Part-to-whole relationships

Advanced Plots

- **Heatmap:** Correlation or confusion matrix
- **Pair Plot:** Pairwise relationships (scatter matrix)
- **Violin Plot:** Distribution shape with quartiles
- **3D Scatter:** Three-variable relationships
- **Contour Plot:** 2D density or function values
- **Q-Q Plot:** Tests normality assumption

Help Menu

About SADSA

- **Description:** Version information and credits
- **Developer:** AMCHIK SOLUTIONS
- **License:** Full Version

User Guide

- **Description:** Opens comprehensive documentation
- **Contents:**
- Getting started tutorial
- Menu reference
- Analysis examples
- FAQ

License Information

- **Description:** View license status and validity
- **Actions:**
- Check license
- Activate license
- View machine ID
- License agreement

Contact Support

- **Email:** contact@codingfigs.com
- **Website:** http://codingfigs.com
- **Response Time:** 24-48 hours

Data Management

Supported File Formats

Import Formats

- **CSV** (Comma-separated values)
- **Excel** (.xlsx, .xls)
- **JSON** (JavaScript Object Notation)
- **TSV** (Tab-separated values)
- **Parquet** (Apache Parquet)
- **LibreOffice** (.ods)

Export Formats

- **CSV**
- **Excel** (.xlsx)
- **JSON**
- **HTML** (for reports)
- **PDF** (via report generation)

Data Grid Features

- **Editable Cells:** Double-click to edit
- **Sort:** Click column headers
- **Filter:** Right-click column header
- **Add/Delete Rows:** Context menu
- **Add Columns:** Edit → Add Column
- **Copy/Paste:** Standard keyboard shortcuts

Analysis Workflow

General Steps

1. **Load Data:** File → Open Data File

2. **Explore Data:** View in data grid, check descriptive statistics
3. **Prepare Data:** Apply transformations if needed
4. **Select Analysis:** Choose from menu
5. **Configure Options:** Select variables, parameters, tests
6. **Run Analysis:** Click "Run Analysis"
7. **View Results:** Multiple tabs with tables and plots
8. **Export Results:** Export to CSV or download report

Tips for Best Results

- **Check Assumptions:** Use normality tests, correlation checks
- **Handle Missing Data:** Remove or impute before analysis
- **Scale Variables:** Use standardization for distance-based methods
- **Validate Results:** Check p-values, confidence intervals, fit indices
- **Visualize:** Always inspect plots to understand patterns

Keyboard Shortcuts

Action	Shortcut
Open File	Ctrl+O
Save	Ctrl+S
Copy	Ctrl+C
Paste	Ctrl+V
Delete Row	Delete
Undo	Ctrl+Z
Redo	Ctrl+Y
Find	Ctrl+F
Select All	Ctrl+A

System Requirements

Minimum Requirements

- **OS:** Windows 10 (64-bit)
- **RAM:** 4 GB
- **Storage:** 500 MB free space
- **Display:** 1280x720 resolution

Recommended Requirements

- **OS:** Windows 11 (64-bit)
- **RAM:** 8 GB or more
- **Storage:** 1 GB free space
- **Display:** 1920x1080 resolution
- **Processor:** Multi-core CPU for faster computations

License & Activation

SADSA uses a license-based activation system: - **Trial Version:** 30 days with full features - **Full Version:** Requires license key - **Activation:** Help → License Information → Activate License

To obtain a license key: 1. Visit http://codingfigs.com 2. Purchase license 3. Receive license key via email 4. Activate in SADSA using Help menu

✅ During Trial Period (Days 1-30):

- ✅ **File** - Supports CSV, Excel, and few other data formats
- ✅ **Transformations** - Data Transformations such as variable recoding, computing etc.
- ✅ **Data Simulations** - Data simulations for quit testing learning purposes (supports cholsky, SVD, QR etc).
- ✅ **Data Analytics** - Uni, bi and multivariate analysis (including CA, PCS, MDS, EFA, CFA, Time Series Forecasting etc.)
- ✅ **Machine Learning** - All supervised & unsupervised algorithms
- ✅ **NLP** - Document-term matrix generation & feature extraction
- ✅ **Meta Analysis** - Fixed/random effects, heterogeneity tests
- ✅ **Bibliometrics** - Citation analysis, co-authorship networks

- ✅ **Advanced Plots** - Multi-plot generator with customization
- ✅ **Report Download** - Export to PDF/DOCX
- ✅ **Data Export** - All formats (CSV, Excel, JSON, Parquet, ODS)
- ✅ **File Import** - All supported formats

Trial starts automatically on first launch - no registration required!

⚠️ After Trial Expires (Day 31+):

- ❌ Machine Learning menu - **Disabled** (requires license)
- ❌ NLP menu - **Disabled** (requires license)
- ❌ Meta Analysis menu - **Disabled** (requires license)
- ❌ Bibliometrics menu - **Disabled** (requires license)
- ❌ Plots menu - **Disabled** (requires license)
- ❌ Download Reports - **Disabled** (requires license)
- ⚠️ File Import - **CSV only** (other formats blocked)
- ✅ Basic data viewing, editing and analysis - **Still available**

Full License Activation

Activate a **FULL LICENSE** for: - ✅ **Permanent Access** - Never expires - ✅ **All Features** - No restrictions - ✅ **All File Formats** - Import/export everything - ✅ **Priority Support** - Direct email assistance - ✅ **Free Updates** - Receive new features and improvements

How to Activate:

1. **Get Your Machine ID**:
2. Help → Machine ID Information
3. Copy either "Computer Name" OR "Machine ID"
4. Example: `39FBD9ACCC7D0618`
5. **Request License**:
6. Email: **contact@codingfigs.com**
7. Subject: "SADSA License Request"
8. Include: Your Machine ID or Computer Name
9. Specify: License type needed (Full/Extended Trial)
10. **Activate**:
11. Receive license key via email
12. Help → Activate License

13. Paste the license key
14. Click "Activate License"
15. **Restart SADSA**
16. **Verify**:
17. Title bar shows: "SADSA - Full Version"
18. Help → License Information: "ACTIVE (PERMANENT)"
19. All menus and features enabled

License Status Display

Your license status is visible in multiple locations: - **Title Bar**: `SADSA - Trial (X days remaining)` or `SADSA - Full Version` - **Status Bar**: Shows daily countdown and available features - **Help** → **License Information**: Detailed license status, expiration date, Machine ID

License Agreement

Copyright © 2025 AMCHIK SOLUTIONS. All rights reserved.

This software is licensed, not sold. By installing and using SADSA, you agree to the terms specified in the License Agreement accessible via Help → License Agreement.

Key Terms: - Personal/Academic use permitted - Commercial use requires appropriate license - Redistribution prohibited - Reverse engineering prohibited

For full license terms, see LICENSE.txt or Help → License Agreement within the application.

Support & Contact

Technical Support

- **Email:** contact@codingfigs.com
- **Website:** http://codingfigs.com
- **Response Time:** 24-48 hours on business days

Report Issues

When reporting issues, please include: - SADSA version number - Windows version - Error message (screenshot if possible) - Steps to reproduce - Sample data (if applicable)

Feature Requests

We welcome suggestions for new features! Email us at contact@codingfigs.com with: - Feature description - Use case - Expected benefit

---

Updates & Versions

SADSA receives regular updates with: - New statistical methods - Bug fixes - Performance improvements - UI enhancements

Check for Updates: Help → Check for Updates

Credits

Developed by: Dr. M. Kamakshaiah / AMCHIK SOLUTIONS
Website: Codingfigs
Contact: dr.m.kamakshaiah@gmail.com

Built with: - Python 3.11+ - Tkinter (GUI) - Pandas (Data manipulation) - NumPy (Numerical computing) - SciPy (Scientific computing) - Scikit-learn (Machine learning) - Matplotlib/Seaborn (Visualization) - Statsmodels (Statistical modeling)

Frequently Asked Questions (FAQ)

Q: Can I use SADSA for commercial purposes?
A: Yes, with a commercial license. Contact us for pricing.

Q: Is my data secure?
A: All data processing is done locally on your computer. No data is transmitted to external servers.

Q: Can I import data from databases?
A: Currently, SADSA supports file-based imports. Database connectivity is planned for future versions.

Q: What if I get an error during analysis?
A: Check that your data meets the analysis assumptions (e.g., no missing values, appropriate data types). Contact support if issues persist.

Q: Can I save my analysis workflow?
A: Currently, you must manually repeat analysis steps. Workflow automation is planned for a future release.

Q: Is there a Mac or Linux version?
A: Currently Windows-only. Mac and Linux versions are under consideration.

Thank you for choosing SADSA!

LATEST UPDATES

🗂️ File Menu: Advanced File Import

- **Menu:** `File → Advanced File Import...`
- **Supported Formats:**
- **Standard Formats:** CSV (.csv), Excel (.xlsx, .xls), JSON (.json), Parquet (.parquet), LibreOffice (.ods), Tab-Separated (.tsv)
- **Advanced Formats:** WEKA (.arff), SPSS (.sav, .por), MATLAB (.mat), Tableau (.tds, .twb)
- **Delimited Files:** Custom delimiters (comma, tab, semicolon, pipe, space, colon, or custom)
- **Encrypted Files:** SADSA Encrypted (.sadsa) with optional password protection
- **Databases:** SQLite (.db, .sqlite, .sqlite3), MySQL, PostgreSQL
- **How to use:**
- Click **File** menu
- Select **Advanced File Import...**
- Choose format tab (Standard Formats, Advanced Formats, Delimited Files, Encrypted, Databases)
- Browse, configure options, preview, and import your data

🤖 Machine Learning: Ensemble Methods

- **Menu:** `Machine Learning → Ensemble Methods → [XGBoost, LightGBM, AdaBoost, Gradient Boosting]`
- Unified 2x2 grid interface for all ensemble algorithms
- **Steps:**
- Select target variable
- Select features
- Configure parameters (n_estimators, learning_rate, max_depth, test_size)
- Train and view results

🖱️ Context Menu Integration

- **Feature:** "Open with SADSA" in Windows right-click menu for `.csv` , `.xlsx` , `.xls` , `.txt` , `.tsv` , `.sadsa` , `.dat` , `.data`
- **Install:**
- Run `install_context_menu.bat` (admin)
- Or use `install_context_menu.py` or registry files
- **Uninstall:**
- Use `uninstall_context_menu.bat` , `uninstall_context_menu.py` , or registry uninstaller

🐍 Python Console

- **Menu:** `Tools → Python Console`
- Launch an interactive Python terminal within SADSA.
- Access app objects: `app` , `data` , `pd` , `np` , `plt` , etc.
- Run any Python code, inspect data, and call SADSA methods directly (e.g., `app.perform_correlation_analysis()`).
- Features:
- Command history (Up/Down arrows)
- Themed interface (choose your favorite look)
- Type `help()` for a quick guide, `methods()` for a list of available app methods
- Supports all standard Python imports (e.g., `import seaborn as sns`)
- Great for power users, debugging, and custom analysis!

📚 Other Features

- **Help Menu:** License activation, documentation, support
- **NLP, Meta Analysis, Bibliometrics:** (optional, see deployment guide)
- **Advanced plotting, reporting, and export**

📖 References & Guides

- See the `backup/` folder for detailed markdown guides:
- `ADVANCED_FILE_IMPORT_GUIDE.md`
- `ENSEMBLE_QUICK_START.md`
- `CONTEXT_MENU_GUIDE.md`
- `FINAL_CLEANUP_SUMMARY.md`
- ...and more!

🏁 Quick Start

1. Install Python dependencies: `pip install -r requirements.txt`
2. Run the app: `python pyfda.py`
3. Use the installer for Windows deployment: `SADSA_Installer_*.exe`

💡 Need Help?

- Open the **Help** menu in the app
- See the documentation in the `backup/` folder
- Or contact the developer via GitHub
