


k-diagram: Technical Report (Derivations and Details)

Kouao Laurent Kouadio 

Abstract—This technical report provides the mathematical foundations for the ‘k-diagram’ Python package. It details the core principles of mapping statistical metrics onto a polar coordinate system, which forms the basis for the package’s novel visualizations. We present the formal definitions and equations used to generate key diagnostic plots, including those for prediction interval coverage, error distribution analysis, and forecast horizon evaluation. The aim is to offer a concise reference for users and developers seeking a deeper understanding of the theoretical underpinnings of ‘k-diagram’.

Index Terms—Probabilistic Forecasting, Uncertainty Quantification, Data Visualization, Polar Coordinates, Model Diagnostics, Technical Report.

I. INTRODUCTION

THE ‘k-diagram’ package provides a suite of visualization tools designed to diagnose the performance of probabilistic forecasts in a granular and intuitive manner [1]. While traditional metrics often aggregate performance into single scores, ‘k-diagram’ leverages polar coordinates to reveal deeper structural patterns in forecast uncertainty and error.

This document serves as a supplement to the main JOSS paper. Its purpose is to explicitly state the mathematical formulations that transform raw forecast outputs and observations into the visual diagnostics offered by the package. We will cover the general mapping framework before detailing the specific equations for several key plots.

II. CORE PRINCIPLES

The fundamental innovation of ‘k-diagram’ is the systematic mapping of one-dimensional statistical properties onto a two-dimensional polar space defined by a radius (r) and an angle (θ).

A. General Definitions

Let us define a set of N observations, $Y = \{y_1, y_2, \dots, y_N\}$. For each observation y_i , a probabilistic forecast provides a predictive distribution, which is often summarized by a set of quantiles.

- **Observation:** $y_i \in \mathbb{R}$ is the true value for instance i .
- **Point Forecast:** \hat{y}_i is a point estimate, often the median (0.5-quantile).

K. L. Kouadio is with the School of Geosciences and Info-physics, Central South University, Changsha 410083, China, and the Hunan Key Laboratory of Nonferrous Resources and Geological Hazards Exploration, Changsha, Hunan, 410083, China (e-mail: lkouao@csu.edu.cn).

This document serves as a technical appendix to the JOSS paper “k-diagram: Rethinking Forecasting Uncertainty via Polar-based Visualization.” Version v1.2.3, September 4, 2025.

- **Prediction Interval (PI):** For a nominal coverage level of $(1-\alpha) \times 100\%$, the PI is defined by its lower and upper bounds, $[L_i, U_i]$, which correspond to the $\alpha/2$ and $1-\alpha/2$ quantiles of the predictive distribution, respectively.
- **Error:** The forecast error is defined as $e_i = y_i - \hat{y}_i$.

B. Polar Coordinate Mapping

The core methodology involves two steps:

- 1) **Score Calculation:** For each instance i , a relevant statistical score, S_i , is computed. This score could represent error, interval width, coverage, etc.
- 2) **Mapping to (r, θ) :** The score S_i is mapped to the radius r_i , while the instance index i (or a categorical variable it belongs to) is mapped to the angle θ_i .

Typically, the angle for instance i is distributed uniformly across a specified angular coverage ($A_{cov} \in (0, 2\pi]$) to represent the entire dataset:

$$\theta_i = A_{cov} \cdot \frac{i-1}{N-1}, \quad \text{for } i = 1, \dots, N \quad (1)$$

The radius r_i is often a direct or normalized representation of the score S_i .

III. FORMULATIONS FOR KEY DIAGNOSTIC PLOTS

A. Coverage Evaluation Plot

This plot provides a point-wise diagnostic of whether an observation falls within its predicted interval.

- 1) **Score:** The score is a binary coverage indicator, c_i , for each observation y_i and its corresponding PI, $[L_i, U_i]$. It is defined using the indicator function $\mathbf{1}(\cdot)$:

$$c_i = \mathbf{1}\{L_i \leq y_i \leq U_i\} = \begin{cases} 1 & \text{if } y_i \in [L_i, U_i] \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

- 2) **Mapping:** Each instance i is mapped to an angle θ_i . The value of c_i is then visualized at that angle, typically using color (e.g., green for $c_i = 1$, red for $c_i = 0$) at a fixed radius.

The overall empirical coverage, or Prediction Interval Coverage Probability (PICP), is the mean of these scores:

$$\text{PICP} = \frac{1}{N} \sum_{i=1}^N c_i \quad (3)$$

B. Model Error Distribution (Polar Violin Plot)

This plot compares the complete error distributions of multiple models.

- 1) **Score:** The score is the set of raw errors, $\{e_i\}_{i=1}^N$, for a given model.
- 2) **Mapping:** A model is assigned a specific angular sector. The radial axis within this sector represents the error value e . The shape of the "violin" is determined by the Probability Density Function (PDF) of the errors, estimated non-parametrically using a Kernel Density Estimator (KDE) [2]:

$$\hat{f}(e) = \frac{1}{Nh} \sum_{i=1}^N K\left(\frac{e - e_i}{h}\right) \quad (4)$$

where K is a kernel function (e.g., Gaussian) and h is the bandwidth. The width of the violin at radius $r = e$ is proportional to $\hat{f}(e)$, providing a visual representation of the error density. A distribution centered at $r = 0$ indicates an unbiased model.

C. Forecast Horizon Drift

This diagram visualizes how forecast uncertainty evolves over different forecast horizons (e.g., lead times).

- 1) **Score:** The score is the mean interval width, \bar{w}_j , for each forecast horizon j . Let $w_{i,j} = U_{i,j} - L_{i,j}$ be the interval width for instance i at horizon j . If there are N_j instances for horizon j , the score is:

$$\bar{w}_j = \frac{1}{N_j} \sum_{i=1}^{N_j} w_{i,j} \quad (5)$$

- 2) **Mapping:** This is typically visualized as a polar bar chart. Each horizon j is assigned a unique angle θ_j . A bar is drawn at this angle with a length (radius) equal to its score, \bar{w}_j . This allows for an intuitive comparison of uncertainty across horizons.

D. Probabilistic Calibration (Polar PIT Histogram)

Calibration is assessed by checking if the Probability Integral Transform (PIT) values are uniformly distributed.

- 1) **Score:** The PIT value, p_i , for an observation y_i is calculated using the forecast's cumulative distribution function (CDF), F_i :

$$p_i = F_i(y_i) \quad (6)$$

If the forecast is perfectly calibrated, the set of PIT values $\{p_i\}_{i=1}^N$ will follow a uniform distribution, $p_i \sim \mathcal{U}(0, 1)$.

- 2) **Mapping:** To visualize the distribution, the range $[0, 1]$ is divided into K bins. The angular range $[0, 2\pi]$ is also divided into K sectors. The score for each sector k is the frequency of PIT values falling into the k -th bin:

$$\text{Freq}(k) = \sum_{i=1}^N \mathbf{1}\left\{p_i \in \left[\frac{k-1}{K}, \frac{k}{K}\right)\right\} \quad (7)$$

In the polar histogram, the radius of the bar in sector k is proportional to $\text{Freq}(k)$. A perfectly calibrated forecast would produce bars of equal radius, forming a circle.

IV. RELATIONSHIP AND ERROR VISUALIZATIONS

This section presents plots designed to directly compare model predictions with actual values and to analyze the structure of the resulting forecast errors, particularly their relationship with other variables.

A. Actual vs. Predicted

This plot offers a direct visual comparison between the true values and a model's point predictions (e.g., the median forecast) in a polar format.

- 1) **Input Data:** A set of N true observations, $Y = \{y_1, \dots, y_N\}$, and a corresponding set of predictions, $\hat{Y} = \{\hat{y}_1, \dots, \hat{y}_N\}$.

- 2) **Mapping:**

- **Angle θ_i :** Each data point i is mapped to an angle based on its index, distributing the points over the specified angular coverage A_{cov} :

$$\theta_i = A_{cov} \cdot \frac{i - 1}{N - 1} \quad (8)$$

- **Radius r_i :** The radius directly corresponds to the value of the observation or prediction. Two data series are plotted: the set of polar points (y_i, θ_i) representing the actuals, and (\hat{y}_i, θ_i) representing the predictions.
- **Error Visualization:** To highlight the error for each point, a radial line segment can be drawn at each angle θ_i . The line connects the predicted radius \hat{y}_i to the actual radius y_i , providing an immediate visual representation of the error magnitude, $e_i = y_i - \hat{y}_i$.

B. Conditional Quantile

This plot visualizes how the predicted conditional distribution (represented by quantiles) changes with the magnitude of the true value. It is an effective tool for diagnosing heteroscedasticity, where the uncertainty of the forecast is not constant.

- 1) **Input Data:** True values $\{y_i\}_{i=1}^N$ and a corresponding matrix of K quantile predictions, $\{\hat{y}_i^{(\tau_k)}\}_{k=1}^K$.
- 2) **Sorting:** To create a coherent visualization, all data points are sorted based on the true value, y_i . Let the sorted data points be indexed by j , such that we have $\{y_{(j)}\}$ and the corresponding quantile predictions $\{\hat{y}_{(j)}^{(\tau_k)}\}$.
- 3) **Mapping:**

- **Angle θ_j :** The angle is mapped from the sorted true value, which sweeps from its minimum to its maximum value across the angular span A_{cov} :

$$\theta_j = \text{map}(y_{(j)}, [\min(y), \max(y)], [0, A_{cov}]) \quad (9)$$

- **Radius r_j :** For each angle θ_j , the radii are the set of corresponding quantile predictions, $\{\hat{y}_{(j)}^{(\tau_k)}\}$.

- 4) **Visualization:** The plot typically visualizes prediction intervals as shaded bands. For a given interval (e.g., 90%), the area between the lower quantile forecast (e.g., $\tau = 0.05$) and the upper quantile forecast (e.g.,

$\tau = 0.95$) is filled. By nesting multiple bands, the plot reveals how the shape and width of the predictive distribution evolve as the true value changes.

C. Error and Residual Relationship

These diagnostic plots are crucial for identifying systematic model deficiencies by revealing patterns in the forecast errors (residuals). The goal is to check if errors are independent of other variables; any discernible structure suggests an opportunity for model improvement.

- 1) **Input Data:** True values $\{y_i\}$, predicted values $\{\hat{y}_i\}$, and the resulting errors $e_i = y_i - \hat{y}_i$.
- 2) **Radius Mapping:** Since errors can be negative and polar radii must be non-negative, the error values are shifted to ensure all points can be plotted. An offset, R_{offset} , is calculated from the minimum error: $R_{offset} = |\min(\{e_i\})|$ if $\min(\{e_i\}) < 0$, and 0 otherwise. The radius for each point i is then:

$$r_i = e_i + R_{offset} \quad (10)$$

With this transformation, the zero-error line becomes a reference circle at radius R_{offset} .

- 3) **Angular Mapping:** The angle is determined by the variable against which the error is being checked. Two primary variants exist:
 - **Error vs. True Value:** To check if error magnitude depends on the true value (reveals heteroscedasticity). The angle θ_i is mapped from the true value, y_i .
 - **Residual vs. Predicted Value:** To check if error depends on the model's own prediction magnitude (reveals bias or misspecification). The angle θ_i is mapped from the predicted value, \hat{y}_i .

In both cases, the data is typically sorted by the angular variable before plotting to create a continuous sweep. An ideal forecast would show points randomly scattered around the zero-error circle.

D. Error Bands

This visualization diagnoses systematic biases by plotting the forecast error as a function of another variable (e.g., time of day, seasonality). It separates the mean error (bias) from the error's volatility (uncertainty).

- 1) **Input Data:** A set of forecast errors $\{e_i\}$ and a corresponding angular variable $\{\alpha_i\}$.
- 2) **Binning:** The data is grouped into K angular bins based on the value of α_i . Let B_k be the set of indices for all data points whose mapped angle $\theta_i = \text{map}(\alpha_i)$ falls into the k -th bin.
- 3) **Score Calculation:** For each bin B_k , the mean error (bias) and the standard deviation of the error are calculated:

$$\mu_{e,k} = \frac{1}{|B_k|} \sum_{i \in B_k} e_i \quad (11)$$

$$\sigma_{e,k} = \sqrt{\frac{1}{|B_k| - 1} \sum_{i \in B_k} (e_i - \mu_{e,k})^2} \quad (12)$$

- 4) **Mapping:** The mean error $\mu_{e,k}$ is plotted as a line against the central angle of each bin. A shaded band is drawn between the upper and lower bounds, defined by a multiple (n) of the standard deviation:

$$\text{Band}_k = [\mu_{e,k} - n \cdot \sigma_{e,k}, \mu_{e,k} + n \cdot \sigma_{e,k}] \quad (13)$$

A reference circle at radius $r = 0$ indicates a perfect, unbiased forecast.

V. FORMULATIONS FOR 2D AND DYNAMIC VISUALIZATIONS

Beyond one-dimensional diagnostics, 'k-diagram' provides tools to visualize two-dimensional distributions and dynamic processes, such as vector fields and temporal evolution. These methods rely on binning, kernel density estimation, and vector calculus principles, adapted for polar coordinates.

A. Polar Heatmap (2D Density)

The polar heatmap visualizes the joint distribution of two variables mapped to polar coordinates, revealing density concentrations.

- 1) **Input Data:** A set of N data points, where each point i has a radial component r_i and an angular component α_i .
- 2) **Coordinate Mapping:** The raw angular data α_i is first mapped to the plot's angular span $A_{cov} \in (0, 2\pi]$ to get the plotting angle θ_i .

$$\theta_i = \text{map}(\alpha_i, [\min(\alpha), \max(\alpha)], [0, A_{cov}]) \quad (14)$$

- 3) **Binning:** The polar plane is discretized into a grid of $N_r \times N_\theta$ bins. A bin, indexed by (j, k) , is defined by radial edges $[R_j, R_{j+1}]$ and angular edges $[\Theta_k, \Theta_{k+1}]$.
- 4) **Score:** The score for each bin is the count of data points that fall within its boundaries. The score C_{jk} for bin (j, k) is:

$$C_{jk} = \sum_{i=1}^N \mathbf{1}\{r_i \in [R_j, R_{j+1}] \wedge \theta_i \in [\Theta_k, \Theta_{k+1}]\} \quad (15)$$

where $\mathbf{1}(\cdot)$ is the indicator function. The resulting matrix of counts C is then visualized, with the color of each bin corresponding to its count.

B. Polar Quiver (Vector Fields)

This plot is designed to visualize a vector field in polar coordinates, which is useful for representing gradients or flows.

- 1) **Input Data:** For each point i , the data consists of a polar coordinate anchor (r_i, θ_i) and a corresponding vector representing change. This change vector is given in polar components: a radial component Δr_i and an angular component $\Delta \theta_i$.
- 2) **Mapping:** The plot renders an arrow (a "quiver") at each anchor point (r_i, θ_i) . The arrow's orientation and length are determined by the vector $(\Delta r_i, \Delta \theta_i)$. The color of the arrow can be mapped to an independent

scalar quantity, or by default, to the magnitude of the vector:

$$M_i = \sqrt{(\Delta r_i)^2 + (r_i \Delta \theta_i)^2} \quad (16)$$

Note the r_i term in the angular component, which converts the angular change (in radians) to a length, making the magnitude physically meaningful. The implementation in ‘matplotlib’ [3] handles the transformation of these components into the final visual representation.

C. Velocity Diagram

This plot visualizes the rate of change (velocity) of a forecasted quantity over time for multiple locations or entities.

- 1) **Input Data:** For each of N locations, we have a time series of median forecasts (Q50) across M time steps: $Q_{i,t}$ for location $i \in \{1, \dots, N\}$ and time $t \in \{1, \dots, M\}$.
- 2) **Score Calculation:** The average velocity v_i for each location is computed as the mean of the first differences of the time series:

$$v_i = \frac{1}{M-1} \sum_{t=1}^{M-1} (Q_{i,t+1} - Q_{i,t}) \quad (17)$$

- 3) **Mapping:** This is a polar scatter plot.
 - **Angle θ_i :** The angle corresponds to the location index, distributing all locations around the circle: $\theta_i = A_{cov} \cdot (i-1)/(N-1)$.
 - **Radius r_i :** The radius represents the calculated average velocity v_i . This value is often normalized to the range $[0, 1]$ for stable visualization:

$$r_i = \frac{v_i - \min(\{v_k\})}{\max(\{v_k\}) - \min(\{v_k\})} \quad (18)$$

- **Color c_i :** The color of each point can be mapped to a separate metric, such as the mean absolute magnitude of the quantity over time, $\bar{Q}_i = \frac{1}{M} \sum_{t=1}^M |Q_{i,t}|$.

D. Radial Density Ring

This visualization adapts the standard 1D Kernel Density Estimate (KDE) into a polar format, providing an intuitive view of a variable’s distribution.

- 1) **Input Data:** A one-dimensional array of N data points, $X = \{x_1, \dots, x_N\}$. This data could be a direct measurement, or a derived quantity like an interval width ($U_i - L_i$).
- 2) **Score Calculation:** The score is the probability density function $\hat{f}(x)$ estimated via KDE, as defined in Equation 4, using a kernel such as the Gaussian kernel implemented in libraries like SciPy [4].
- 3) **Mapping:** The density is visualized as a colored ring.
 - **Radius r :** The radial axis represents the value of the variable x .
 - **Angle θ :** The density estimate is constant with respect to angle, so it is replicated across the entire angular span A_{cov} to form a continuous ring.
 - **Color c :** The color at any radial position r is mapped to the normalized density value at that point, $\hat{f}_{norm}(r) = \hat{f}(r) / \max(\hat{f})$.

This creates a plot where high-density regions of the data appear as brightly colored concentric circles.

E. Error Ellipses (2D Uncertainty)

This plot visualizes two-dimensional uncertainty where the errors in the radial and angular directions are distinct. It is ideal for representing anisotropic uncertainty, for instance in tracking or location forecasting.

- 1) **Input Data:** For each point i , the data consists of a mean position in polar coordinates, $(\mu_{r,i}, \mu_{\theta,i})$, and the associated standard deviations of uncertainty in the radial, $\sigma_{r,i}$, and angular, $\sigma_{\theta,i}$ (in radians), directions.
- 2) **Ellipse Definition:** An uncertainty ellipse is drawn for each point. In a local Cartesian frame aligned with the radial direction, the semi-axes of the ellipse are defined by a multiple (n) of the standard deviations:
 - **Semi-axis (radial):** $a_i = n \cdot \sigma_{r,i}$
 - **Semi-axis (tangential):** $b_i = n \cdot (\mu_{r,i} \tan(\sigma_{\theta,i})) \approx n \cdot \mu_{r,i} \sigma_{\theta,i}$. The tangential length is derived from the angular uncertainty and the mean radius.
- 3) **Transformation and Mapping:** To plot the ellipse correctly on the polar axes, its path is calculated parametrically in the local frame, rotated by the angle $\mu_{\theta,i}$, translated to the global Cartesian position corresponding to $(\mu_{r,i}, \mu_{\theta,i})$, and finally converted back into polar coordinates for rendering. This process ensures each ellipse accurately reflects the magnitude and orientation of the 2D uncertainty at its specific location.

VI. COMPREHENSIVE MODEL EVALUATION

The package also includes tools for holistic model comparison, enabling the assessment of multiple models across multiple performance metrics in a single, compact visualization.

A. Regression Performance

This plot, also known as a “Polar Performance Chart,” provides a multi-faceted comparison of regression models. It creates a polar grouped bar chart that visualizes different performance metrics simultaneously, creating a unique “fingerprint” for each model.

- 1) **Input Data:** A set of J models to be evaluated against K performance metrics. The raw score of model j on metric k is denoted S_{jk} .
- 2) **Score Unification:** All metrics are transformed such that a higher value indicates better performance. For error metrics like Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE), where lower is better, the scores are negated (e.g., -MAE). Standard metrics like the Coefficient of Determination (R^2) [5] are used directly.
- 3) **Normalization:** To make disparate metrics comparable, the scores are normalized. A common strategy is “per-metric” normalization, which scales the scores for each metric independently to the range $[0, 1]$:

$$r_{jk} = \frac{S_{jk} - \min_{j'}(S_{j'k})}{\max_{j'}(S_{j'k}) - \min_{j'}(S_{j'k})} \quad (19)$$

In this scheme, the best-performing model on metric k receives a normalized score of 1, and the worst receives a score of 0.

4) Mapping:

- **Angular Groups:** The polar axis is divided into K angular sectors, one for each metric. The central angle for metric k is $\Theta_k = A_{cov} \cdot (k - 1)/K$.
- **Grouped Bars:** Within each metric's sector, a group of J bars is drawn, one for each model. The angle for the bar corresponding to model j within metric group k is offset from the central angle Θ_k .
- **Radius:** The height (radius) of each bar corresponds to its normalized performance score, r_{jk} .

The resulting chart is framed by two circles: an inner "Worst Performance" ring (radius 0) and an outer "Best Performance" ring (radius 1), providing clear visual references for comparison.

B. Pinball Loss Plot

The pinball loss is a proper scoring rule used to evaluate the accuracy of quantile forecasts. This plot visualizes the average pinball loss for each forecasted quantile, providing a detailed assessment of a model's probabilistic calibration and sharpness across the entire distribution.

- 1) **Score Definition:** The pinball loss, L_τ , for a true value y and a τ -quantile forecast $\hat{y}^{(\tau)}$ is defined as:

$$L_\tau(y, \hat{y}^{(\tau)}) = \begin{cases} (y - \hat{y}^{(\tau)})\tau & \text{if } y \geq \hat{y}^{(\tau)} \\ (\hat{y}^{(\tau)} - y)(1 - \tau) & \text{if } y < \hat{y}^{(\tau)} \end{cases} \quad (20)$$

This loss function penalizes over-prediction and under-prediction asymmetrically, with the penalty determined by the quantile level τ .

- 2) **Average Score:** For a dataset of N observations, the plot computes the average pinball loss for each of the K forecasted quantiles, $\{\tau_k\}_{k=1}^K$:

$$\bar{L}_{\tau_k} = \frac{1}{N} \sum_{i=1}^N L_{\tau_k}(y_i, \hat{y}_i^{(\tau_k)}) \quad (21)$$

3) Mapping:

- **Angle θ_k :** The angle is directly proportional to the quantile level, $\theta_k = A_{cov} \cdot \tau_k$. This maps the entire range of quantiles $[0, 1]$ onto the angular span.
- **Radius r_k :** The radius corresponds to the calculated average pinball loss, $r_k = \bar{L}_{\tau_k}$.

A well-calibrated forecast typically yields a U-shaped plot, indicating lower losses (higher accuracy) for central quantiles and higher losses for the more challenging extreme quantiles.

VII. CONCLUSION

This report has provided the essential mathematical expressions underpinning the visualizations in the 'k-diagram' package. By mapping statistical scores to polar coordinates, the package offers a framework for dissecting and understanding the complex behaviors of forecasting models. For implementation details and usage, please refer to the package [documentation](#) and the main JOSS paper [1].

REFERENCES

- [1] K. L. Kouadio, "k-diagram: Rethinking forecasting uncertainty via polar-based visualization," 2025, pre-publication draft. [Online]. Available: <https://github.com/earthai-tech/k-diagram>
- [2] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall/CRC, 1986.
- [3] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
- [4] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Wehrer, J. Huttunen, T. H acklin, S. Straub, J. Pike, A. Petroff, K. Conti, R. Lieberman, J. Pollard, B. Nichols, J. Raimi, E. Hoydu, K. B. Kerkwijk, M. Brett, A. Haldane, J. del R  o, G. Manolopoulos, A. Ivanov, K. J. Millman, M. C. Wehner, S. J. Briggs, N. Yurkewycz, R. Knothe, J. Butterworth, N. Goodman, S. J. de la Vega, O. Routhenko, A. Perron, Z. Rothberg, E. Ryaboy, N. Williams, P. Turrell, S. Behnel, L. Sieber, R. Blake, B. R. Kern, R. Larson, C. A. Edwards, E. Fazio, I. Juric, T. Pitkanen, S. Cijesek, B. E. Taylor, R. T. Miller, E. Jessen, D. Derksen, C. J. Carey, S. Wejchert, S. Olenin, A. Petrov, A. Jachalsky, V. Slavov, O. Bokhove, J. Jung, M. Aivazis, M. D. Hendriksen, K. A. Halderman, A. de Miranda Cardoso, D. Levy, and Y. V  zquez-Baeza, "SciPy 1.0: fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, no. 3, pp. 261–272, 2020.
- [5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Olszewski, M. Blondel, F. Prejbal, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and   . Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.