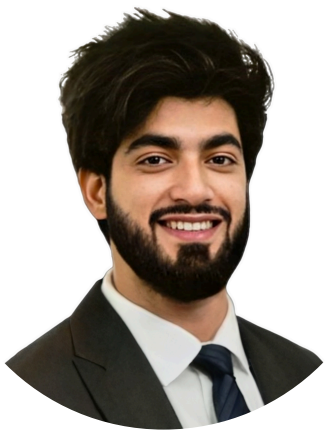

Tajamul Khan

Statistics Cheat Sheet

@Tajamulkhann

Descriptive Statistics

- **mean(data):** Calculate the average of a dataset.
- **median(data):** Find the middle value of sorted data.
- **mode(data):** Identify the most frequently occurring value.
- **variance(data):** Measure the spread of data points.
- **std(data):** Calculate the standard deviation to quantify data dispersion.
- **min(data):** Determine the smallest value in the dataset.
- **max(data):** Identify the largest value in the dataset.
- **quantile(data, q)**: Compute the q-th quantile of the data.

@Tajamulkhann

Probability Concepts

- **binom.pmf(k, n, p):** Probability of k successes in n Bernoulli trials.
- **poisson.pmf(k, λ):** Probability of k events in a given interval.
- **norm.pdf(x, μ, σ):** Probability density of normal distribution @x.
- **norm.cdf(x, μ, σ):** Cumulative probability for a value x in a normal distribution.
- **uniform.pdf(x, a, b):** Uniform distribution probability density at x.
- **expon.pdf(x, λ):** Probability density for exponential distribution.
- **bernoulli.pmf(k, p):** Probability mass for Bernoulli trials.
- **beta.pdf(x, α, β):** Probability density for Beta distribution.

@Tajamulkhann

Inferential Statistics

- **ttest_ind(data1, data2):** Perform an independent t-test.
- **ttest_rel(data1, data2):** Perform a paired t-test.
- **anova(data1, data2, data3):** Conduct ANOVA to compare multiple means.
- **zscore(data):** Standardize data using z-scores.
- **chi2_contingency(table):** Test independence using a Chi-square test.
- **confidence_interval(mean, std, n, confidence=0.95):** Compute confidence intervals.
- **pearsonr(x, y):** Calculate Pearson correlation coefficient.
- **spearmanr(x, y):** Compute Spearman rank correlation.

@Tajamulkhann

Hypothesis Testing

- **p_value(data, null_hypothesis):** Calculate p-value to test null hypothesis.
- **z_test(x̄, μ, σ, n):** Perform a Z-test for population mean.
- **one_sample_ttest(data, μ0):** Compare sample mean to a population mean.
- **two_sample_ttest(data1, data2):** Compare means of two independent samples.
- **wilcoxon(data1, data2):** Non-parametric test for paired samples.
- **f_oneway(data1, data2, data3):** One-way ANOVA for multiple groups.
- **mannwhitneyu(data1, data2):** Test differences between two independent samples.
- **power_analysis(effect_size, n, α):** Determine sample size or test power.

@Tajamulkhann

Correlation & Regression

- **corrcoef(x, y):** Calculate correlation matrix.
- **linregress(x, y):** Perform linear regression analysis.
- **polyfit(x, y, deg=1):** Fit a polynomial regression model.
- **ols(formula, data):** Perform ordinary least squares regression.
- **residuals(y_actual, y_pred):** Calculate regression residuals.
- **r_squared(y_actual, y_pred):** Compute coefficient of determination (R^2).
- **logit(data):** Apply logistic regression analysis.
- **predict(model, new_data):** Use trained model for predictions.

@Tajamulkhann

Statistics Visualization

- **sns.boxplot(data):** Identify outliers and visualize distributions.
- **sns.histplot(data, kde=True):** Visualize data distribution with KDE overlay.
- **sns.scatterplot(x, y):** Analyze relationships between variables.
- **sns.heatmap(corr_matrix, annot=True):** Display correlation matrix with annotations.
- **sns.pairplot(data):** Show pairwise relationships and trends.
- **sns.violinplot(x, y):** Display distribution for categorical comparisons.
- **plt.errorbar(x, y, yerr):** Add error bars to scatterplots.
- **sns.lmplot(x, y, data):** Plot regression line with scatter points.

@Tajamulkhann

Sampling Techniques

- **random.sample(population, k):** Draw a random sample of size k.
- **random.choices(population, weights, k):** Perform weighted sampling.
- **stratified_sample(data, strata, size):** Generate stratified random samples.
- **bootstrapping(data, n_samples):** Create bootstrap samples for confidence intervals.
- **systematic_sample(data, interval):** Select samples systematically.
- **cluster_sample(data, clusters, size):** Perform cluster sampling.
- **oversampling(data, minority_class):** Handle imbalanced datasets.
- **undersampling(data, majority_class):** Reduce majority class size for balance

@Tajamulkhann

Found Helpful ?

Repost

Follow for more!