**Regression Tasks (Predicting Continuous Variables)**

1. **Predict Starting Salary** based on education, skills, internships, and networking score.
2. **Predict Career Satisfaction** using job-related and education-related features.
3. **Predict Years to Promotion** given performance indicators, networking, and job level.
4. **Predict University GPA** from high school GPA, SAT score, university ranking, and field of study.
5. **Predict Job Offers** based on skills, projects, certifications, networking score, and internships.
6. **Predict Soft Skills Score** based on networking score, internships, and field of study.
7. **Predict Networking Score** from soft skills, internships, university ranking, and extracurriculars.
8. **Predict Work-Life Balance** based on career level, job satisfaction, and entrepreneurship.
9. **Predict University Ranking Preference** using SAT score, GPA, and field of study.
10. **Predict the Number of Internships Completed** based on academic performance and networking.

**Classification Tasks (Predicting Categorical Outcomes)**

1. **Classify Job Offers into "High" vs. "Low"** based on skills, internships, and projects.
2. **Classify Career Satisfaction into "Satisfied" vs. "Not Satisfied"** based on job and personal factors.
3. **Classify High School GPA as "Above Average" or "Below Average"** using SAT score and university rank.
4. **Classify Soft Skills Score into "Strong" vs. "Weak"** based on internships, networking, and certifications.
5. **Classify Students as Likely to Become Entrepreneurs or Not** based on their academic and career path.
6. **Classify Employees as "Fast Promoters" vs. "Slow Promoters"** based on job level, networking, and performance.
7. **Classify Work-Life Balance into "Good" vs. "Poor"** using job satisfaction, salary, and career level.
8. **Classify University GPA as "High" vs. "Low"** based on university ranking, SAT score, and study field.
9. **Classify Fields of Study into "Technical" vs. "Non-Technical"** using career level and internships.
10. **Classify Employees as "High Performers" vs. "Average Performers"** based on job level, projects, and networking.

**1. Problem Definition and Project Planning**

1. **Clarify the question or objective**:
   * What exactly are you trying to predict, classify, or cluster?
   * Is the final goal to achieve a certain accuracy, to provide a proof-of-concept, or to explore the dataset for research insights?
2. **Define success metrics and constraints**:
   * How will you measure success (e.g., accuracy, F1-score, RMSE)?
   * What time or computational constraints exist for your analysis?
3. **Set up a project structure**:
   * Plan a folder structure (e.g., data/, scripts/, results/, etc.).
   * Consider version control (e.g., Git) to track experiments and script changes.

**2. Data Acquisition and Loading**

1. **Gather data**:
   * Assemble all relevant data sources (CSV files, SQL databases, sensor data, etc.).
   * Make sure you have the appropriate permissions and understand any data privacy requirements.
2. **Load data into MATLAB**:
   * Use MATLAB functions like readtable, csvread, or specialized data import tools.
   * For large datasets, consider using the **Tall Arrays** feature or **Datastore** to process chunks of data without loading everything into memory at once.

**3. Data Exploration and Cleaning**

1. **Exploratory Data Analysis (EDA)**:
   * Use summary statistics (mean, std, summary) and visual exploration (e.g., histogram, plot, scatter) to understand data distributions, outliers, and relationships.
   * Plot correlations using MATLAB’s built-in functions such as corrplot.
2. **Detect and handle anomalies, missing values, and outliers**:
   * Identify rows with missing values (ismissing), decide whether to remove them or impute.
   * Evaluate strategies for outlier handling (clipping, transformation, or removal).
3. **Check data types and consistency**:
   * Ensure numeric variables are numeric, categorical are categorical, etc.

**4. Data Preprocessing and Feature Engineering**

1. **Transform and scale data**:
   * Normalize or standardize features (e.g., using normalize, zscore, or custom transformations).
   * Encode categorical data (with dummyvar or the **Categorical Arrays** in MATLAB).
2. **Dimensionality reduction (if needed)**:
   * If the dataset is very high-dimensional, consider techniques like PCA (pca), t-SNE (tsne for visualization), or autoencoders (from the **Deep Learning Toolbox**).
3. **Feature engineering**:
   * Create new features from existing data (e.g., date/time transformations, domain-specific calculations).
   * Consider domain knowledge: sometimes manually crafted features can significantly improve model performance.

**5. Model Selection and Development**

1. **Split your data**:
   * Create training, validation, and test sets (common splits include 60/20/20, 70/15/15, etc.).
   * Use MATLAB’s built-in functions like cvpartition for cross-validation.
2. **Initial model selection**:
   * Based on the task (classification, regression, etc.), consider:
     + **Classification**: Logistic regression, SVMs, Decision trees, Random forests, Neural networks.
     + **Regression**: Linear/Nonlinear regression, SVR, Regression trees, Ensembles, Neural networks.
     + **Unsupervised**: k-means, Hierarchical clustering, Gaussian Mixture Models, etc.
3. **Hyperparameter tuning**:
   * Automate hyperparameter tuning using **Bayesian Optimization**, **Random Search**, or **Grid Search** (fitcauto, fitrauto, or the **Hyperparameter Optimization** options in Classification/Regression Learner apps).
   * Use cross-validation to get reliable estimates of model performance.
4. **Iterate**:
   * Try different models, compare metrics, refine your feature set or data preprocessing steps based on the insights.

**6. Model Evaluation and Validation**

1. **Assess performance thoroughly**:
   * For **classification**: Accuracy, precision, recall, F1-score, confusion matrices.
   * For **regression**: MSE, RMSE, MAE, R².
   * For **clustering**: Silhouette score, Calinski-Harabasz, Davies-Bouldin, etc.
2. **Compare multiple models**:
   * Plot learning curves to diagnose bias/variance issues.
   * Use ensemble methods or stacked approaches if single models are insufficient.
3. **Perform error analysis**:
   * Investigate where the model performs poorly.
   * Identify if more data is needed or if further feature engineering could help.

**7. Deployment or Final Presentation**

1. **Model deployment (if applicable)**:
   * MATLAB offers **MATLAB Compiler** and **MATLAB Production Server** for deploying models.
   * Alternatively, export models to other environments if required (e.g., C code generation, Python integration).
2. **Documentation and reproducibility**:
   * Document your code and pipeline so that others can replicate your work.
   * Store final models, hyperparameters, and data-preprocessing pipelines in a well-organized manner.
3. **Presentation of results**:
   * Provide visual aids (graphs, tables, etc.) and clear explanations of findings.
   * Emphasize interpretability and insights gained, especially for advanced machine learning coursework.