

First Project Supervised Learning Regression Problem (Structured Dataset)

**Loading and Exploring Data**:

* + We start by importing important libraries like Pandas, NumPy, Seaborn, and Matplotlib, along with machine learning modules from Scikit-learn.
  + We read a CSV file called 'housing\_data.csv' using Pandas and explore the initial rows, dataset information, and summary statistics.
  + Initial visualizations, like pair plots, correlation matrices, box plots, and histograms, help us understand relationships and distributions among numerical columns.
* **Data Preprocessing**:
  + We encode categorical variables using one-hot encoding with 'get\_dummies'.
  + Selected numerical columns are scaled using Scikit-learn's MinMaxScaler.
  + To handle missing values, we impute the mean and perform Principal Component Analysis (PCA) on these columns.
* **Cleaning Data and Feature Engineering**:
  + We convert the 'date' column into a datetime format.
  + Checking consistency between 'code' and 'area' columns involves a group-wise check for uniqueness.
  + Any conflicts between 'code' and 'area' entries are resolved, assuming a single area for each code.
  + New features like 'year' and 'month' are extracted from the 'date' column.
* **Training and Evaluating Models**:
  + The data is split into features (X) and the target variable (y).
  + Various regression models (Linear Regression, Decision Tree, Random Forest, Gradient Boosting, SVR, KNeighbors) are used to predict 'no\_of\_crimes'.
  + For each model, we evaluate performance metrics like Mean Squared Error (MSE) and R-squared using train-test splits.
* **Improving Our Models**:
  + We fit a RandomForestRegressor model with optimized parameters obtained from earlier evaluations.
  + Using this model, we make predictions for 'no\_of\_crimes' and add these predictions to the housing\_data DataFrame.
* **Hyperparameter Tuning**:
  + Utilizing GridSearchCV, we search for the best hyperparameters for the RandomForestRegressor model.
* **Final Model Evaluation**:
  + We evaluate the RandomForestRegressor model with optimized parameters on a test set, measuring performance using MSE and R-squared metrics.

# **Second Project Supervised Learning Classification Problem (Structured Dataset**)

**Preparing Tools**:

* + We start by bringing in tools that help us handle data and create models.
* **Checking Our Data**:
  + We load a dataset about gender classification.
  + We check what the dataset contains, like the type of information and how much of it is missing.
  + We remove any repeated information in the dataset.
* **Getting Our Data Ready**:
  + We adjust some types of information to make them easier to use.
  + We turn words or labels into numbers so our computer can understand them better.
* **Understanding Our Data**:
  + We create a few graphs to show relationships between different parts of our data.
  + For example, we show how often certain characteristics (like having long hair) appear.
* **Building and Testing Models**:
  + We split our data into parts to train models and check how well they work.
  + We try different methods to predict gender based on the information given.
  + Each method tries to learn patterns in the data and predict gender.
  + We evaluate how accurate each method is at guessing the right gender.
* **Summarizing Our Approach**:
  + We explore various ways to understand the data and predict gender. We test these methods to see which one works better for this specific task.

# Third Project Unsupervised Learning (Structured Dataset)

**Loading and Exploring Data**:

* + We begin by importing Pandas and loading our diabetes dataset named 'patients.csv'.
  + Our initial step involves inspecting the dataset with **info()**, **describe()**, and **isnull().sum()** to understand its structure, summary statistics, and missing values.
  + We also peek into the first few rows and check unique values in the 'Outcome' column, assuming it's our target variable.
* **Data Inspection and Visualization**:
  + We further examine the dataset by checking missing values, summary statistics, and data types using **info()** and **describe()**.
  + Visualizations like a count plot of the 'Outcome' column and a heatmap of feature correlations help us understand data distributions and relationships.
* **Feature Analysis and Visualizations**:
  + We use box plots and pair plots to delve deeper into important features like 'Glucose', 'BMI', 'Age', 'Insulin', and 'BloodPressure'.
* **Data Preprocessing**:
  + Outlier handling specifically for the 'BMI' column involves replacing values exceeding certain boundaries.
  + We print the changes in the 'BMI' column after this handling.
* **Clustering Techniques**:
  + We employ K-Means and DBSCAN clustering methods to identify patterns within the dataset.
  + Visualization of clusters based on 'Glucose' and 'BMI' helps us visualize the effectiveness of clustering.
* **Data Manipulation and Analysis**:
  + We print the entire updated dataset and drop a column named 'cluster' from it.
* **Dimensionality Reduction**:
  + Using t-SNE, we reduce the dataset's dimensions to two components ('TSNE1', 'TSNE2') to aid visualization.
* **Further Exploration**:
  + We investigate unique values within the 'Age' and 'Pregnancies' columns to understand the range and variation present.
* **Independent Component Analysis (ICA)**:
  + Employing FastICA, we decompose original features into statistically independent components ('IC1', 'IC2', 'IC3') to uncover underlying factors.

Each step allows us to understand, manipulate, and visualize different aspects of our diabetes dataset, aiding in comprehensive data exploration and analysis.

# Fourth Project Supervised Learning Classification Problem (Unstructured Dataset - Image Dataset)

* **Data Loading and Preprocessing**:
  + We start by loading images and their respective labels from a directory using OpenCV and NumPy.
  + Images are resized to 64x64 pixels, converted to floating-point format, and normalized to values between 0 and 1.
  + The images and labels are appended to separate lists for further processing.
* **Data Preparation**:
  + We convert the image list into a NumPy array 'X', containing all the images.
  + The labels are encoded using **LabelEncoder()** from Scikit-learn, assigning numerical values to categorical labels.
  + To ensure randomness in the dataset, we shuffle 'X' and 'y' using **shuffle()** from Scikit-learn.
* **Building the Convolutional Neural Network (CNN)**:
  + We create a sequential model using Keras and add convolutional layers, max-pooling layers, and dense layers.
  + The CNN architecture includes multiple convolutional layers with increasing filters and a final dense layer with a softmax activation for multi-class classification.
* **Compiling and Training the Model**:
  + We compile the model using 'adam' optimizer and 'sparse\_categorical\_crossentropy' loss, optimizing for accuracy.
  + Training occurs on the shuffled data for 10 epochs with a batch size of 25, utilizing 20% of the data for validation.
* **Model Evaluation**:
  + We plot the training and validation loss to visualize the model's performance during training.
  + After training, we load test images and labels similarly, prepare the test set, encode the labels, and evaluate the trained model using the test set.
  + The evaluation provides us with the test loss and accuracy metrics to gauge the model's performance on unseen data.
  + Each step in this process contributes to the creation, training, and evaluation of a convolutional neural network designed for image classification.

Project link: https://github.com/Mosyexp/machine2