# Machine Learning

## Methodology

# Description of Preliminary Data Preprocessing

## Kaggle Dataset #1 (U.S. Homicide Reports, 1980 - 2014)

- Filtered ‘State’ column for California.
- Renamed ‘City’ column as ‘County’ in order to enable joining with second dataset.
- Dropped irrelevant columns (‘Agency Code’ and “Record Source’) **WHY IRRELEVANT?**
- Checked for null values (there were none).

## Kaggle Dataset #2 (US Census Demographic Data)

- Filtered ‘State’ column for California.
- Dropped irrelevant columns (‘Professional, ‘Service’, “Office’, ….) **WHY IRRELEVANT?**
- Checked for null values (there were none).

Datasets merged in Postgres; merged dataset uploaded to AWS S3 bucket

# Description of Preliminary Feature Engineering / Selection

- Checked unique values in columns with categorical values for potential bucketing **ELABORATE ON WHY NO BUCKETING WAS REQUIRED**
- Encoded categorical data using OneHotEncoder.
- Merged the encoded features and dropped the originals from the DataFrame.
- Divided the columns into ‘X’ and ‘y’ for ‘features’ and ‘target’, with ‘CrimeSolved’ as the target (what the ML model is being trained to predict) and all other columns as the features.
- Split the data into training and testing datasets.
- Scaled the data, which is recommended for certain ML models including logistic regression.

# Description of How the Data was Split into Training and Testing Datasets

- The dataset was split using train_test_split() function from Python's sklearn library.
- The 'stratify' argument was used to ensure the training and testing datasets have a comparable ratio of both outcomes, given that there are nearly twice as many solved homicide cases in California as unsolved.
- The 'random_state" argument was used to ensure the experiment is reproducible.

# Explanation of Model Choice and Limitations / Benefits

- We are looking for a binary, categorical output (Solved vs. Unsolved) with multiple independent variables of unknown importance to determining the target.
- Therefore, we will be testing for accuracy, as well as training vs. testing scores (to determine potential overfitting) on a basic logistic regression model, a random forest model and a deep neural network model. We will assess the best performing model based on accuracy and training/testing score ratio, with an accuracy cutoff of 75%.

# Limitations / Benefits Continued

- Pros and cons of each model:
  - Logistic Regression:
    - Pros: Least prone to overfitting, least computationally expensive.
    - Cons: Can’t learn on data that isn’t linearly divisible, more prone to bias.
  - Random Forest:
    - Pros: Can rank feature importance and handle nonlinear data, and less prone to overfitting than neural networks.
    - Cons: More parameters to tune and more computationally expensive than logistic regression.
  - Deep Neural Network:
    - Pros: Best suited for identifying complex nonlinear relationships.
    - Cons: Most prone to overfitting, most parameters to tune, most computationally expensive.