Eirene Michella Tjhan

Data Science Portfolio

Undergraduate Student—Data Science Major
Bina Nusantara University

Tangerang, Indonesia
Email: eirene.michella@gmail.com
LinkedIn: linkedin.com/in/eirene-michella-tjhan
GitHub: github.com/eirenemichellatjhan

Table of Contents

About Me

I am a fifth-semester undergraduate student majoring in **Data Science** at **Bina Nusantara University (GPA: 3.80/4.00)** with a strong passion for turning data into actionable insights. Throughout my studies, I have developed solid analytical and programming skills in Python, R, and SQL, and I have gained hands-on experience with frameworks and tools such as TensorFlow, Keras, Scikit-learn, PySpark, and Streamlit.

My academic journey has exposed me to a wide range of topics, including machine learning, deep learning, data mining, visualization, and big data technologies. Beyond the classroom, I actively contribute to the **Data Science Club** and am currently a member of **Data Seekers**, the university's official data science competition team.

Professionally, I have worked part-time at **PT. Pandansari Prima Anugrah**, where I was responsible for financial documentation, transaction processing, and coordination with banking partners, developing my attention to detail and accountability.

I have also led and managed multiple roles in organizations such as **Karsa4Youth** and **Tumpu.id**, where I directed public relations, organized educational programs, and collaborated with local communities to promote health and education initiatives. These experiences strengthened my communication, leadership, and project management abilities.

I am detail-oriented, eager to learn, and currently seeking an internship opportunity to apply my data science skills in a professional environment. I aspire to begin my career as a data scientist, contributing to data-driven solutions that enhance decision-making and innovation.

Technical Skills

Programming Languages:
Python, R, SQL

Frameworks & Libraries:
TensorFlow, Keras, Scikit-learn, PyTorch, Pandas, NumPy

Data Visualization:
Matplotlib, Seaborn, Plotly, Tableau, Power BI

Big Data & Cloud Tools:
PySpark, Hadoop, Google Colab, Docker

Development & Deployment:
Streamlit, Git, Jupyter Notebook, VS Code

Machine Learning & Analytics:
Data Preprocessing, Feature Engineering, Classification, Regression, Clustering, Deep Learning, NLP, Model Evaluation, and Deployment

Soft Skills:
Analytical Thinking, Problem-Solving, Team Collaboration, Leadership, Communication, Time Management

Projects

Lecture-Based Projects

This section highlights the projects I've completed **throughout my academic studies**, each developed as part of course requirements to apply theoretical concepts to real-world problems.

1. Deep Learning

Hourly Air Temperature Prediction using LSTM | Link

This project aimed to forecast the next hour's air temperature (AT) based on the previous 5 hours of air quality and meteorological data. The dataset included multiple environmental factors such as particulate matter (PM2.5, PM10), gases (CO, NOx, SO2), and weather parameters (humidity, wind speed, solar radiation). By combining these features, the model learned short-term temperature patterns useful for urban planning, public health, and energy management.

- **Tools:** Python, TensorFlow, Keras, Pandas, Matplotlib
- **Approach:** Compared three models: a simple LSTM, a bidirectional LSTM, and a tuned version with fewer layers and dropout.
- **Results:** The baseline LSTM achieved the best performance with the lowest MAE and MAPE and highest R^2, outperforming the more complex models.
- **Key Insight:** Good data and clear temporal patterns can make a simple model outperform deeper, more complex ones.

Pistachio Image Denoising using Autoencoders | Link

This project focused on restoring noisy pistachio images using deep learning-based denoising autoencoders. The goal was to reconstruct clean images from noisy inputs, improving visual quality and preserving structural details for better image clarity and feature extraction.

- **Tools**: Python, TensorFlow, Keras, OpenCV, Matplotlib
- **Approach**: Compared three architectures: a baseline autoencoder, a U-Net model, and a tuned U-Net with optimized hyperparameters using GridSearch.
- **Results**: The Tuned U-Net achieved the lowest MSE (0.000230), lowest MAE (0.006506), and highest reconstruction accuracy, outperforming the baseline and standard U-Net.
- **Key Insight**: Careful hyperparameter tuning can significantly improve image reconstruction quality, even when using similar model architectures.

Pistachio Image Generation using GANs | Link

This project explored generative modeling using pistachio images to synthesize realistic samples through adversarial learning. The goal was to train models capable of producing high-quality pistachio images that closely resemble real ones, evaluated using quantitative and perceptual metrics.

- **Tools**: Python, TensorFlow, Keras, Matplotlib, NumPy
- **Approach**: Compared three models: a simple GAN (baseline), a DCGAN with convolutional layers, and a fine-tuned DCGAN with adjusted filters, dropout, and learning rates for better balance between generator and discriminator.
- **Results**: The Fine-Tuned DCGAN achieved the lowest FID score (181.93) on the full dataset, indicating the highest realism. It produced smoother and more visually coherent images compared to the Baseline and DCGAN models.
- **Key Insight**: Fine-tuning hyperparameters such as filter size and learning rate significantly improved visual realism, even when training stability appeared lower.

Pharmaceutical Drug Image Classification using CNN and Transfer Learning | Link

This project aimed to classify synthetic images of pharmaceutical drugs and vitamins using deep learning. The model helps identify drugs from their images, reducing confusion and assisting the public in recognizing medications accurately. The dataset used was the Pharmaceutical Drugs and Vitamins Synthetic Images from Kaggle, consisting of labeled synthetic drug and vitamin images.

- **Tools**: Python, TensorFlow, Keras, Matplotlib
- **Approach**: Implemented transfer learning using EfficientNetB0 and ResNet50V2 with both fully frozen and partially frozen layers. Attention map visualization was applied to analyze which image areas influenced model predictions.
- **Results**: The partially frozen ResNet50V2 achieved the best performance, with accuracy >80% and F1-scores ≥0.80 across all classes. Partial layer unfreezing significantly improved feature adaptation and focus compared to fully frozen models.
- **Key Insight**: Fine-tuning pretrained models (partial freezing) greatly increases performance on domain-specific data, leading to more accurate and stable classification results.

Image Reconstruction using Denoising and Masked Autoencoders (Fashion MNIST) | Link

This project explores how autoencoders can learn efficient visual representations for grayscale fashion images. Using the Fashion MNIST dataset, both Denoising Autoencoder (DAE) and Masked Autoencoder (MAE) models were developed to reconstruct images from incomplete or noisy inputs, demonstrating the ability of deep learning models to capture meaningful structure in limited data.

- **Tools**: Python, TensorFlow, Keras, NumPy, Matplotlib
- **Approach**: Implemented and trained both DAE and MAE architectures for 100 epochs. The DAE learned to remove Gaussian noise from images, while the MAE reconstructed masked image patches, focusing on structure-level understanding.
- **Results**: The DAE achieved an average SSIM of 0.61 (best 0.96), indicating good reconstruction quality despite occasional failure on complex textures. The MAE produced low and stable loss, maintaining high SSIM on most samples and showing strong structural consistency even with partial visual input.
- **Key Insight**: The DAE excelled in pixel-level recovery, while the MAE proved better at learning robust visual representations, showing that self-supervised approaches can perform well even with incomplete data.

Brain Tumor Image Segmentation using U-Net Architecture | Link

This project focused on segmenting brain tumors from MRI scans of patients with Lower-Grade Glioma (LGG) using a U-Net deep learning architecture. The dataset contained MRI images paired with expert-labeled tumor masks outlining tumor regions. The goal was to automatically identify and localize tumors to assist in diagnosis and treatment planning.

- **Tools**: Python, TensorFlow, Keras, NumPy, OpenCV, Matplotlib
- **Approach**: Implemented a U-Net model for pixel-wise tumor segmentation. The model was trained on paired MRI images and masks, with evaluation based on accuracy, Intersection over Union (IoU), and Dice coefficient.
- **Results**: Achieved test Accuracy = 99.70%, Dice = 0.739, and IoU = 0.849, showing strong overall performance. However, because most pixels in MRI scans represent background, high accuracy can be misleading; the model may classify healthy regions perfectly but still miss small tumor areas. Visual inspection confirmed that while the model captured tumor regions well, minor outline mismatches and missed details occurred on complex shapes.
- **Key Insight**: U-Net performs effectively for medical image segmentation but should be evaluated beyond accuracy. Metrics like Dice and IoU, alongside visual inspection, provide a more realistic view of model performance, especially for critical tumor boundary detection.

Amazon Stock Price Prediction using LSTM and Bidirectional RNN | Link

This project focused on forecasting Amazon's stock price using deep learning models on historical data from 2006 to 2018. The dataset included daily open, high, low, close, and volume values. The objective was to predict the next day's closing price and evaluate how well RNN-based models can capture stock market patterns.

- **Tools**: Python, TensorFlow, Keras, NumPy, Scikit-learn, Matplotlib
- **Approach**: Built and compared LSTM and Bidirectional RNN models for time series prediction. Data was normalized, sequenced, and evaluated using RMSE, MAE, and R^2.
- **Results**: The LSTM achieved a strong fit ($R^2 \approx 0.95$ on test data) but showed overfitting with higher test errors. The Bidirectional RNN slightly improved generalization and captured temporal patterns more effectively.
- **Key Insight**: Incorporating bidirectional context helps reduce overfitting and improves forecasting stability compared to a standard LSTM.

Garment Worker Productivity Prediction using Sequential and Functional Neural Networks | Link

This project analyzed productivity in a garment factory to understand how factors like overtime, idle time, team size, incentives, and style changes influence performance. The dataset included both operational and temporal features such as date, day, quarter, and SMV (Standard Minute Value).

- **Tools**: Python, TensorFlow, Keras, NumPy, Pandas, Scikit-learn, Matplotlib
- **Approach**: Built and compared four regression models: two Sequential and two Functional (baseline and modified). The baseline models used ReLU activations, while the modified versions tested ELU to improve stability and avoid dead neurons. Models were evaluated using MAE, MSE, and R^2.
- **Results**: The Sequential baseline model achieved the best balance of accuracy and generalization, with higher R^2 and lower MAE/MSE than other variants. Functional models showed more flexibility but tended to overfit.
- **Key Insight**: While ELU improved training stability, ReLU-based Sequential networks performed best overall. The experiment highlighted that simpler architectures can outperform more flexible designs when data is noisy or limited.

Bone X-ray Classification using AlexNet and EfficientNet | Link

This project focused on classifying bone X-ray images into five categories: *Normal, Doubtful, Mild, Moderate,* and *Severe* osteoarthritis. The dataset consisted of labeled medical images, aiming to automate diagnosis through deep learning.

- **Tools**: Python, TensorFlow, Keras, NumPy, Matplotlib, Scikit-learn
- **Approach**: Built and compared multiple CNN architectures: Baseline AlexNet, Modified AlexNet (BatchNorm + Dropout + Adam), and EfficientNet. Hyperparameter tuning was applied (optimizer, dropout rate, and learning rate). Models were evaluated using accuracy and loss.
- **Results**: The EfficientNet model achieved the best accuracy at 25.6%, outperforming the AlexNet variants (≈20%). However, class imbalance led to bias toward "Moderate" and "Severe" categories.
- **Key Insight**: Architectural improvements and fine-tuning didn't guarantee better performance. Future improvement should focus on balancing classes and improving feature representation rather than deeper models.

Possum Classification Using CNN (Sequential vs Functional Approach) | Link

This project aimed to classify possums into Victoria or Other populations based on 13 biological features (e.g., head length, weight, and tail length). A simple CNN-like dense network was implemented using both the Sequential and Functional APIs in Keras to compare performance and architectural flexibility on tabular data.

- **Tools**: Python, TensorFlow, Keras, NumPy, Scikit-learn
- **Approach**: Developed and compared two models using Keras: a Sequential CNN-based dense network and a Functional model with dual feature paths and concatenation. Both models used Dropout for regularization and ReLU/LeakyReLU activations to capture non-linear relationships in the data.
- **Results**: The Sequential model achieved 90% test accuracy, while the Functional model slightly improved performance to 92%. The Functional model showed better recall for "Victoria" but misclassified a few "Other" samples.
- **Key Insight**: The Functional approach provided better generalization and flexibility, demonstrating that even simple tabular data can benefit from nonlinear architectural designs beyond a purely sequential setup.

2. Model Deployment

Obesity Diagnosis Model with Streamlit Deployment | Link

This project aims to classify individuals into seven obesity levels based on lifestyle and health-related attributes using tabular data. The task focuses on building an accurate machine learning model and deploying it as an interactive web application.

- **Tools**: Python, Scikit-learn, FastAPI, Streamlit, Joblib, Pandas, NumPy
- **Approach**: Compared Random Forest and XGBoost models for multiclass classification. The best-performing model was saved as a .pkl file and deployed using a FastAPI backend connected to a Streamlit frontend with a custom-designed UI.
- **Results**: XGBoost achieved the highest accuracy (96.17%) and near-perfect F1-scores across most classes, effectively identifying high-risk obesity types.
- **Key Insight**: XGBoost proved slightly superior to Random Forest, showing strong generalization and reliability for healthcare risk prediction. The deployment pipeline demonstrates a complete end-to-end system, from model training to real-time prediction.

Streamlit-Based Hotel Booking Cancellation Predictor using Random Forest & XGBoost | Link

This project predicts whether a hotel booking will be canceled based on guest, stay, and financial details. The goal is to help hotels reduce cancellations and optimize room allocation through predictive analytics.

- **Tools**: Python, Scikit-learn, Streamlit, Pickle, Pandas, NumPy
- **Approach**: Developed an end-to-end machine learning pipeline using object-oriented programming (OOP). The workflow includes data preprocessing (encoding, feature scaling, and handling categorical variables), model training (Random Forest and XGBoost), and evaluation. The best model was saved using Pickle and deployed through a Streamlit app with a user-friendly input form for real-time prediction.
- **Results**: The XGBoost model achieved the best accuracy of 94.2%, outperforming Random Forest in both precision and recall. The model successfully identified high-risk bookings and potential cancellations.
- **Key Insight**: Proper data preprocessing and encoding significantly impacted model performance. The project demonstrates a complete deployment cycle, from raw data to an interactive web app, providing actionable insights for hotel booking management.

3. Survey Sampling Methods

Research of Finding a Correlation Between Phone Screen Time and Previous Grade Point Semester (GPS) from Computer Science and Data Science Majors in BINUS University | Link

This project investigated the relationship between university students' smartphone screen time and their academic performance (Grade Point Semester). The survey aimed to understand how screen habits, both quantity and type of usage, affect students' learning focus and outcomes.

- **Tools**: Python, Pandas, Matplotlib, WordCloud, Scikit-learn
- **Approach**: Conducted a student survey via Google Forms, followed by data preprocessing (cleaning nulls, correcting typos, and renaming columns) and exploratory text mining on open-ended responses. Descriptive statistics and correlation analysis (Spearman) were used to explore the link between screen time behavior and academic performance.
- **Results**: Analysis revealed no strong correlation between total screen time and GPA. Most students spent over 8 hours daily on their smartphones regardless of GPA level. However, higher-GPA students balanced productivity and entertainment app use more effectively.
- **Key Insight**: The type and purpose of smartphone use, not its duration, showed stronger links to academic success. Distraction management and app prioritization are more influential than simply limiting screen time.

4. Machine Learning

Heart Attack Risk Prediction: Health Data Analysis for Early Detection | Link

This project focuses on predicting the 10-year risk of coronary heart disease (CHD) based on behavioral, demographic, and medical factors. The goal was to identify at-risk individuals early using a machine learning approach that integrates multiple classification algorithms.

- **Tools**: Python, Scikit-learn, XGBoost, Pandas, NumPy, Matplotlib, Seaborn
- **Approach**: Evaluated multiple supervised learning models: Logistic Regression, Decision Tree, Random Forest, SVM, KNN, Naive Bayes, and XGBoost, using accuracy and classification reports. The top three models (Logistic Regression, Random Forest, SVM) were fine-tuned using GridSearchCV, followed by an ensemble stacking method combining all three optimized models for final evaluation.
- **Results**: The Random Forest and Logistic Regression models achieved the highest individual accuracy (~85%), while the stacking ensemble slightly improved performance (85.08%). However, results also revealed class imbalance, the model performed well on non-risk cases but struggled to identify positive CHD cases accurately.
- **Key Insight:** The ensemble model achieved high accuracy but struggled to detect high-risk patients because the dataset was imbalanced. To improve results, techniques like SMOTE, cost-sensitive learning, or better feature selection could help the model identify heart disease risks more effectively.

Predicting Cirrhosis Stage Using Random Forest & XGBoost | Link

This project aimed to predict the clinical stage of liver disease (cirrhosis) from patient records containing biochemical and clinical features (e.g., Albumin, Bilirubin, Prothrombin, SGOT). The goal was to compare ensemble models and identify which biomarkers are most informative for stage prediction.

- **Tools**: Python, Scikit-learn, Pandas, NumPy, Matplotlib
- **Approach**: Trained and compared Random Forest and XGBoost classifiers. Performed hyperparameter tuning with GridSearchCV (multiple search spaces) and evaluated models on an independent test set using accuracy, precision/recall/F1 per class, and confusion matrices. Feature importance was extracted from the best model (Random Forest).
- **Results**: Random Forest achieved ~50% test accuracy, while XGBoost reached ~44%. Both models struggled with underrepresented classes (notably class 0), performing best on classes 2 and 3. After tuning, Random Forest was selected as the best model. We also gained

information that the most important features were Albumin (0.19), Prothrombin (0.17), Triglycerides (0.158), Alkaline Phosphatase (0.094), and SGOT (0.092).

- **Key Insight**: Albumin and Prothrombin (liver function biomarkers) are the most important features for predicting cirrhosis stage. The model's accuracy was affected by class imbalance and similar feature patterns between stages. Improving the data balance (using SMOTE) and adding more clear features could help the model make better predictions.

Customer Segmentation using K-Means Clustering | Link

This project applies unsupervised learning to segment customers based on demographic, purchasing, and behavioral attributes. The goal was to uncover hidden customer patterns to support targeted marketing and strategy decisions.

- **Tools**: Python, Scikit-learn, Pandas, NumPy, Matplotlib, Seaborn
- **Approach**: The dataset was clustered using the K-Means algorithm, with hyperparameter tuning based on the silhouette score and elbow (distortion) method, both indicating k = 4 as the optimal number of clusters. Each cluster was then analyzed for key characteristics, including age, spending habits, discount usage, payment preferences, and shopping frequency.

- **Results**:
  - Cluster 0 (Older): Seniors (avg. age 65), moderate spenders with high engagement (≈26 past purchases). Prefer Venmo and shop mostly in spring, fall, and summer. Discounts influence 44% of purchases.
  - Cluster 1 (Millennials): Mid-aged group (avg. 38 years) with consistent spending and strong credit card usage. Prefer shopping in winter and often buy annually or monthly.
  - Cluster 2 (Young Adults): The youngest segment (avg. 25 years), frequent shoppers with the highest purchase amount ($60.51). Prefer PayPal and show positive reviews with moderate discount usage.
  - Cluster 3 (Mid-Aged): Middle-aged customers (avg. 52 years) with the highest engagement (25.83 previous purchases). Heavily discount-driven (57.6%) and active during spring–summer seasons.
- **Key Insight**: K-Means successfully identified four distinct customer profiles differing in age, spending behavior, and purchasing preferences. This segmentation highlights opportunities for personalized marketing, such as loyalty programs for middle-aged customers and discount strategies targeting younger shoppers.

Video Game Recommendation System using Content-Based Filtering | Link

This project builds a personalized recommendation system to suggest similar video games based on attributes such as genre, platform, and publisher. The goal was to generate relevant recommendations using content similarity instead of user behavior.

- **Tools**: Python, Scikit-learn, Pandas, NumPy, TF-IDF Vectorizer, Cosine Similarity
- **Approach**: Implemented a content-based recommender system using TF-IDF vectorization to represent text-based features numerically and cosine similarity to compute similarity scores between games. A custom function returns the top 5 most similar titles for a given input.
- **Results**: The model produced relevant and brand-consistent recommendations, demonstrating strong correlation between game details and recommendation quality.
- **Key Insight**: Content-based filtering successfully recommends games with similar characteristics, though expanding to hybrid or user-based methods could improve personalization and reduce redundancy in suggestions.

5. Bayesian Data Analysis

House Price Prediction Using Bayesian Linear vs Beta Regression | Link

This project applies Bayesian regression methods to predict house prices based on key property and location features from the kc_house_data dataset. After sampling 5,000 entries and selecting 11 predictors through correlation analysis, the goal was to identify which housing attributes most strongly influence price.

- **Tools**: R, Bayesian Linear Regression, Bayesian Beta Regression, DIC & WAIC Evaluation
- **Approach**: Two Bayesian models were built: Linear Regression with uninformative Gaussian priors and Beta Regression for scaled price prediction. Both models were run with MCMC (2 chains), and convergence was assessed using Gelman-Rubin, Geweke, autocorrelation, and effective sample size diagnostics to ensure reliable posterior estimates. Model comparison was performed using Deviance Information Criterion (DIC) and Widely Applicable Information Criterion (WAIC).
- **Results**:
  - Bayesian Linear Regression achieved lower DIC (1112.35) and WAIC (397.51), indicating better predictive accuracy than Bayesian Beta Regression (DIC = 2467.76, WAIC = 548.13).
  - Posterior predictive checks showed that the model closely matched observed data, confirming a good fit.
- **Key Insight**: Bayesian Linear Regression provided more accurate and stable predictions of house prices. Variables such as floors, waterfront, view, grade, and latitude showed strong positive influence on price, while bedrooms, bathrooms, and living area had limited impact.

6. Data Mining and Visualization

Causes of Type 2 Diabetes in Pima Indian Women | Link

This project investigates the two main causes of Type 2 diabetes in Pima Indian women: lifestyle habits and genetic predisposition. The goal was to analyze how daily habits and family history influence diabetes risk and provide actionable insights for prevention.

- **Tools**: R, dplyr, ggplot2, stats, Shapiro-Wilk Test, Pearson Correlation
- **Approach**: Performed Exploratory Data Analysis (EDA) to summarize statistics, visualize distributions, and identify anomalies. Conducted statistical analysis to examine correlations between lifestyle factors (BMI, SkinThickness) and genetic factors (DiabetesPedigreeFunction) with diabetes outcomes.
- **Results**: Analysis showed that individuals with higher BMI and SkinThickness, indicating overweight or obesity, are more likely to develop diabetes, highlighting the critical role of lifestyle. Additionally, a high DiabetesPedigreeFunction indicates a genetic predisposition, meaning that even with a healthy lifestyle, family history can increase diabetes ri
- **Key Insight**: Both lifestyle habits and genetic factors contribute to diabetes. Maintaining a healthy lifestyle is crucial to reduce risk, but individuals with a family history of diabetes should monitor glucose levels regularly. Early awareness and proactive management can prevent complications and improve overall health outcomes.

Capstone/Professional Certificate Projects

This section showcases hands-on projects from **professional certification programs**, highlighting practical skills and applied learning.

1. IBM Data Science

SpaceX Launch Performance Analysis and Prediction | Link

This project analyzed SpaceX launch data to understand mission outcomes and predict future launch success. The goal was to identify key factors affecting mission success and develop a predictive model for future launches.

- **Tools**: Python, SQL, Pandas, Scikit-learn, Folium, Plotly Dash
- **Approach**: Collected data via the SpaceX REST API and Wikipedia scraping, cleaned and merged datasets, then performed EDA using SQL and visualizations. Built interactive maps and dashboards to explore launch sites and mission outcomes. Trained and tuned classification models (Logistic Regression, SVM, Decision Tree, KNN) to predict mission success.
- **Results**: Decision Tree and KNN achieved the highest test accuracy (88.89%), with the Decision Tree providing better class separation and fewer misclassifications. Analysis also revealed improved SpaceX success rates over time, especially after 2016.
- **Key Insight**: Predictive modeling combined with interactive visualization provides actionable insights into mission planning, demonstrating that data-driven strategies can enhance launch success rates.

2. Meta Data Analyst

Facebook vs AdWords Ad Performance Analysis | Link

This project analyzes the effectiveness of two online advertising platforms—Google AdWords and Facebook Ads—in driving conversions. The goal was to determine which platform performs better and predict conversions based on ad metrics.

- **Tools**: Excel, Tableau, Statistical Modeling (t-test, Simple Linear Regression)
- **Approach**: Conducted exploratory analysis of AdWords and Facebook Ads data, then applied t-tests and Simple Linear Regression to examine the relationship between Facebook Ad Clicks and Conversions.
- **Results**: Analysis showed a strong positive correlation between Facebook Ad Clicks and Conversions, indicating that increasing clicks leads to higher conversions. Facebook Ads was identified as the more effective platform for this dataset.
- **Key Insight**: Statistical modeling can reveal platform performance differences and guide data-driven decisions for optimizing online advertising strategies.

Competition Projects

This section showcases projects **completed in competition**, focusing on solving real-world problems with data-driven solutions.

Crowd Counting in Public Spaces Using CSRNet | Link

This project develops a deep learning model to estimate crowd density in images captured from surveillance cameras. The goal was to provide real-time crowd estimates to help city planners manage large events safely, prevent overcrowding, and optimize resource allocation.

- **Tools**: Python, TensorFlow, Keras, OpenCV, NumPy, Matplotlib
- **Approach**: Prepared image datasets and corresponding annotations, generated density maps, and trained a CSRNet model with a VGG-16 front-end and dilated convolutional back-end. Model performance was evaluated using Mean Squared Error (MSE) and Pixel Mean Absolute Error (MAE) on both training and validation sets.
- **Results**: The model achieved a final train loss (MSE) of 0.1460, validation loss of 0.3353, train pixel MAE of 0.1115, and validation pixel MAE of 0.2004, indicating effective learning with slight overfitting.
- **Key Insight**: CSRNet can accurately estimate crowd sizes in congested scenes by combining local feature extraction and global context understanding, providing actionable insights for smart city crowd management and public safety planning.

Predicting COPPA Violations in Children's Mobile Apps Using Artificial Neural Networks | Link

This project analyzes the risk of mobile applications violating the Children's Online Privacy Protection Act (COPPA). The goal was to classify apps as "at risk" or "not at risk" and identify key factors contributing to privacy risks.

- **Tools**: Python, TensorFlow, Keras, NumPy, Pandas
- **Approach**: Conducted exploratory data analysis to identify patterns, performed feature selection, encoded categorical data, normalized features, and trained an Artificial Neural Network (ANN) classifier. Model performance was evaluated using accuracy and binary cross-entropy.
- **Results**: The ANN model successfully classified apps based on COPPA risk, highlighting influential features such as app category, user audience, and privacy policy indicators.
- **Key Insight**: Deep learning can effectively detect potential privacy violations in children's apps, enabling safer app development and supporting data-driven policy decisions.

Customer Feedback Prediction Using Machine Learning | Link

This project analyzes customer feedback on online food orders, aiming to predict whether feedback is positive or negative and identify patterns that influence customer satisfaction.

- **Tools**: Python, Scikit-learn, Pandas, NumPy, Matplotlib, Seaborn
- **Approach**: Performed exploratory data analysis to understand customer demographics and order features, preprocessed data, and trained multiple models including Random Forest, XGBoost, Logistic Regression, KNN, and SVM. Models were evaluated using accuracy, F1 Score, confusion matrices, and ROC AUC metrics to handle imbalanced classes effectively.
- **Results**: The SVM model achieved the best balance of performance with 88.46% accuracy, 0.9268 F1 Score, and 0.899 ROC AUC, demonstrating strong precision, recall, and class separation. Random Forest closely matched accuracy and F1 Score, while other models like XGBoost, Logistic Regression, and KNN performed slightly lower but still provided reliable predictions.
- **Key Insight**: SVM with a 4th-degree polynomial kernel effectively captures complex patterns in feedback data, making it the most reliable model for predicting both positive and negative customer responses in imbalanced datasets.