PROJECT ABSTRACT

Startup Success Prediction using Machine Learning

A tabular binary classifier predicting startup outcomes (acquired vs closed) from funding and company features: feature engineering on the public 923-startup dataset, logistic regression vs random forest vs gradient boosting compared honestly, full evaluation metrics, and a single-file demo that trains live in the browser. Delivered built-to-order with code, notebooks, report, PPT and viva kit.

01. Title

Startup Success Prediction using Machine Learning

02. Introduction / Background

Investors, incubators and analysts all wrestle with the same question: which early signals separate the startups that get acquired from the ones that shut down? Human judgment in this domain is famously noisy, and the relevant factors — funding rounds, amounts and timing, team size, sector, geography — interact in ways intuition cannot track. Machine learning treats this as tabular binary classification: learn from hundreds of historical startup outcomes which feature patterns point toward acquisition and which toward closure. This project builds that pipeline on the public Kaggle Startup Success Prediction dataset — 923 startups, 49 raw attributes, with acquired-vs-closed outcomes — compares several classifiers on identical splits, and explains predictions through learned feature weights instead of treating the model as an oracle.

03. Problem Statement

Early-stage investment and incubation decisions are made under extreme uncertainty, and unstructured judgment leads to inconsistent, biased calls. The relevant evidence is tabular and historical: how much was raised, in how many rounds, how fast, by whom, in which sector and city. No accessible student-built tool turns that evidence into a calibrated, explainable prediction with honest evaluation. The project addresses this by building the full pipeline — cleaning and encoding the 49 raw attributes, handling the class imbalance (597 acquired vs 326 closed), comparing logistic regression, random forest and gradient-boosted trees on the same splits, and reporting accuracy, precision, recall, F1, ROC-AUC and the confusion matrix on a held-out test set.

04. Objectives

- Profile the Startup Success Prediction dataset (923 startups, 49 raw attributes) for missing values, outliers and the acquired/closed class balance.
- Build a feature-engineering pipeline: missing-value handling, categorical encoding, scaling and derived funding-timeline features (~32 final features).
- Train logistic regression, random forest and gradient-boosted trees on stratified train/validation/test splits and select the best by validation F1.
- Evaluate on the held-out test set with accuracy, precision, recall, F1, ROC-AUC and the confusion matrix, with imbalance handling documented.
- Build a single-file browser demo that trains logistic regression by gradient descent live, shows the loss curve and weights table, and serves a prediction form with per-feature explanations.
- Deliver the complete student kit: source code, notebooks, report, PPT and viva Q&A.

05. Proposed Methodology

The build follows a standard tabular-ML workflow. (1) Data profiling: pandas explores the 923 rows and 49 columns, quantifying missingness and the 597/326 class split. (2) Preprocessing: numeric features are imputed and scaled, categoricals one-hot or target encoded, and funding-timeline features (rounds count, total raised, pace) are derived. (3) Modeling: logistic regression, random forest and gradient-boosted trees are trained with cross-validation on stratified splits; hyperparameters are tuned on validation F1. (4) Evaluation: the selected model is scored once on the held-out test set with the full metric suite and confusion matrix. (5) Demo: a JavaScript logistic regression with batch gradient descent trains on built-in data in the browser, exposing the loss curve, weight table, confusion matrix and a prediction form.

06. System Architecture / Working

In the Python pipeline, raw CSV data flows through the cleaning/encoding stage into a feature matrix, then through stratified splitting into the three candidate models; the validation-F1 winner is re-evaluated on the test set, and the notebook renders the metric comparison, ROC curves and the confusion matrix. In the browser demo, a built-in 400-row synthetic dataset feeds a from-scratch gradient-descent trainer: each epoch updates weights, the loss curve redraws, and the final weights populate an inspectable table. The prediction form takes a startup's funding and company features, applies the same preprocessing, and returns an acquisition probability alongside the top contributing features with their weights — the explainability story for the viva.

07. Hardware / Software Requirements

- Python 3.10 with scikit-learn, pandas, NumPy, Matplotlib, Seaborn (pipeline and notebooks)
- Jupyter notebook for training, tuning and evaluation

- Kaggle Startup Success Prediction dataset (public download)
- Any modern desktop browser for the single-file HTML/CSS/JS demo app — no server, works offline
- No special hardware; training runs on an ordinary student laptop CPU

08. Expected Outcomes

- Clean feature pipeline turning 49 raw attributes into ~32 model-ready features.
- Honest three-model comparison on identical splits, winner chosen by validation F1.
- Full test-set evaluation: accuracy, precision, recall, F1, ROC-AUC, confusion matrix — numbers produced by the buyer's own build.
- Browser demo with live gradient-descent training, weights table and prediction form.
- Complete documentation: report, PPT, methodology notes and viva Q&A.

09. Applications

- Teaching platform for tabular ML: feature engineering, imbalance, model comparison.
- Reference build for other binary-outcome business datasets (churn, default, hiring).
- Demo of explainable linear models: weight tables as prediction explanations.
- Baseline for more advanced approaches (calibrated models, survival analysis).

10. Future Scope

- Probability calibration (Platt/isotonic) for trustworthy risk scores.
- SHAP-based global and local explanations beyond linear weights.
- Survival-analysis framing: time-to-acquisition or time-to-closure.
- Larger, fresher startup datasets and sector-specific models.

11. References

- Kaggle — Startup Success Prediction dataset (manishkc06).
- scikit-learn documentation — logistic regression, ensembles, metrics.
- He, H. and Garcia, E. A. — Learning from Imbalanced Data (TKDE, 2009).
- Lundberg, S. M. and Lee, S.-I. — A Unified Approach to Interpreting Model Predictions (NeurIPS, 2017).