Machine Learning Final Year Project Ideas PDF

A good machine learning final year project solves one clear problem on a real public dataset, compares at least two models with the right metric, explains the result, and ends with a small working app. Fraud detection, churn prediction, disease risk and demand forecasting are safe, well-documented choices for B.Tech, BCA and MCA students.

Key takeaways

- Pick a problem with a public dataset you can download today, such as UCI or Kaggle data.
- Always compare a simple baseline with one or two stronger models.
- Choose the metric to suit the problem: recall for fraud and disease, RMSE or MAPE for forecasting.
- A small Streamlit or Flask app makes the demo easy to follow in the viva.
- The PDF gives plans and steps, not finished source code.

Who this PDF is for

This PDF is for final-year B.Tech, BCA, MCA and M.Sc students who need a machine learning project that is big enough for a final evaluation but small enough to finish in one semester. Every idea below uses data you can actually get, a tech stack taught in most colleges (Python, pandas, scikit-learn, XGBoost, Streamlit or Flask), and a clear list of modules you can split among 3 or 4 team members.

The projects are grouped by difficulty. Start with the level that matches your team's Python skill, not the one that sounds most impressive.

How to choose your project

- **Data first.** If you cannot download the dataset this week, drop the idea.
- **One measurable goal.** "Predict churn with recall above the baseline" is better than "use AI for telecom".
- **Fit the team.** Each member should own one module: data cleaning, modelling, evaluation or the app.
- **Ask your guide early** whether a dashboard or app is required, and whether IEEE papers must be cited.

Standard level projects

1. Credit Card Fraud Detection

- **Problem:** Flag fraudulent card transactions where frauds are a tiny fraction of all rows.
- **Dataset:** Kaggle "Credit Card Fraud Detection" (European cardholders, PCA features).
- **Tech stack:** Python, pandas, scikit-learn, imbalanced-learn, XGBoost, Streamlit.
- **Main modules:** Data loader, resampling (SMOTE or class weights), model training, threshold tuning, scoring app.
- **Steps:** Explore class imbalance, build a logistic regression baseline, train a tree model, tune the threshold on validation data, report precision-recall.
- **Show in the viva:** Why accuracy is misleading here, the precision-recall curve and the chosen threshold.
- **Extensions:** Add cost-based evaluation or a simple drift check on new data.

2. Telecom Customer Churn Prediction

- **Problem:** Predict which customers are likely to leave so the business can act early.
- **Dataset:** Kaggle "Telco Customer Churn" (IBM sample data).
- **Tech stack:** Python, pandas, scikit-learn, LightGBM, SHAP, Streamlit.
- **Main modules:** Cleaning and encoding, feature engineering, model comparison, explanations, retention dashboard.
- **Steps:** Encode categories, split with stratification, compare logistic regression and LightGBM, explain with SHAP, list top risk customers.
- **Show in the viva:** Top churn drivers from SHAP and how a manager would use the list.
- **Extensions:** Add a what-if slider for contract type and monthly charges.

3. Heart Disease Risk Prediction

- **Problem:** Estimate heart disease risk from clinical measurements.
- **Dataset:** UCI "Heart Disease" dataset (Cleveland subset is most used).
- **Tech stack:** Python, scikit-learn, Flask, Chart.js.
- **Main modules:** Input form, preprocessing pipeline, model, risk explanation page.
- **Steps:** Handle missing values, scale features, compare logistic regression, random forest and SVM with cross-validation, save the best pipeline.
- **Show in the viva:** Recall versus precision for a screening tool, and a clear note that it is not a medical device.
- **Extensions:** Add calibration so the risk score reads as a probability.

4. Bank Marketing Term Deposit Prediction

- **Problem:** Predict whether a customer will subscribe to a term deposit after a call campaign.
- **Dataset:** UCI "Bank Marketing" dataset.
- **Tech stack:** Python, scikit-learn, CatBoost, Plotly.
- **Main modules:** EDA report, leakage check, model training, campaign targeting simulator.
- **Steps:** Remove the call-duration leak, encode features, train CatBoost, rank customers by score, show lift by decile.

- **Show in the viva:** Why the duration column is leakage and how lift charts guide a campaign.
- **Extensions:** Compare results across age bands for a fairness check.

Advanced level projects

5. Retail Sales Demand Forecasting

- **Problem:** Forecast daily store sales to plan stock and staff.
- **Dataset:** Kaggle "Rossmann Store Sales" competition data.
- **Tech stack:** Python, pandas, LightGBM, Prophet, Streamlit.
- **Main modules:** Time features, holiday and promo features, model training, backtesting, forecast dashboard.
- **Steps:** Build lag and rolling features, use time-based splits, compare Prophet with LightGBM, measure RMSPE, plot forecasts by store.
- **Show in the viva:** Why random splits break time series and how backtesting works.
- **Extensions:** Add a forecast for a new store with little history.

6. Movie Recommendation System

- **Problem:** Recommend movies a user is likely to rate highly.
- **Dataset:** MovieLens dataset from GroupLens (100K or 1M version).
- **Tech stack:** Python, Surprise or implicit, FastAPI, React or Streamlit.
- **Main modules:** Popularity baseline, collaborative filtering, content-based model, hybrid ranker, demo UI.
- **Steps:** Split by user and time, train matrix factorisation, add genre similarity, evaluate with precision at k and RMSE.
- **Show in the viva:** The cold-start problem and how the hybrid handles new users.
- **Extensions:** Add diversity so lists are not only blockbusters.

7. Network Intrusion Detection with ML

- **Problem:** Classify network connections as normal or attack types.
- **Dataset:** NSL-KDD dataset from the University of New Brunswick.
- **Tech stack:** Python, scikit-learn, XGBoost, Streamlit.
- **Main modules:** Feature encoding, multi-class model, per-attack metrics, alert viewer.
- **Steps:** Encode protocol and service, train on KDDTrain+, test on KDDTest+, report a confusion matrix per attack family.
- **Show in the viva:** Why the test set contains attack types missing from training.
- **Extensions:** Try an autoencoder for anomaly detection and compare.

8. Income Prediction with Fairness Audit

- **Problem:** Predict whether income exceeds 50K and check the model for bias.
- **Dataset:** UCI "Adult" (Census Income) dataset.
- **Tech stack:** Python, scikit-learn, Fairlearn, Streamlit.
- **Main modules:** Baseline model, fairness metrics by group, mitigation step, comparison report.

- **Steps:** Train gradient boosting, measure demographic parity and equalised odds by sex and race, apply a Fairlearn mitigation, compare accuracy loss.
- **Show in the viva:** The trade-off between accuracy and fairness, with numbers from your run.
- **Extensions:** Add a model card that documents limits and intended use.

Research-oriented projects

9. Air Quality Index Forecasting for Indian Cities

- **Problem:** Forecast next-day AQI from pollutant readings.
- **Dataset:** Kaggle "Air Quality Data in India (2015 - 2020)".
- **Tech stack:** Python, pandas, XGBoost, statsmodels, Plotly.
- **Main modules:** City-wise cleaning, gap filling, feature lags, models per city, map view.
- **Steps:** Fill short gaps, create lag features, compare ARIMA with XGBoost, report MAE per city.
- **Show in the viva:** How missing sensor data was handled and its effect on error.
- **Extensions:** Add weather data as extra features.

10. Student Performance Prediction and Early Warning

- **Problem:** Spot students at risk of failing early in the term.
- **Dataset:** UCI "Student Performance" dataset (Portuguese and Maths).
- **Tech stack:** Python, scikit-learn, SHAP, Flask.
- **Main modules:** Data merge, model with and without earlier grades, explanation, teacher dashboard.
- **Steps:** Train two versions (with and without G1/G2 grades), compare, explain top factors, build a simple alert list.
- **Show in the viva:** How results change when earlier grades are removed, and why.
- **Extensions:** Add a counselling note generator based on top factors.

11. Wine Quality Prediction with Explainable ML

- **Problem:** Predict wine quality from lab tests and explain each prediction.
- **Dataset:** UCI "Wine Quality" dataset (red and white).
- **Tech stack:** Python, scikit-learn, XGBoost, SHAP, LIME.
- **Main modules:** Regression and classification versions, explainers, comparison of explainers.
- **Steps:** Treat quality as a score and as good/bad classes, train both, compare SHAP and LIME on the same rows.
- **Show in the viva:** Where SHAP and LIME agree and where they differ.
- **Extensions:** Write it up as a short conference paper.

12. Hyperparameter Tuning Benchmark

- **Problem:** Compare grid search, random search and Bayesian tuning on the same tasks.
- **Dataset:** Three UCI datasets used above (Adult, Wine Quality, Bank Marketing).
- **Tech stack:** Python, scikit-learn, Optuna, MLflow.
- **Main modules:** Experiment runner, tracking, results tables, charts of score against time.

- **Steps:** Fix seeds and budgets, run each tuner on each dataset, log runs in MLflow, compare best score and time taken.
- **Show in the viva:** Your MLflow runs and the time versus score chart.
- **Extensions:** Add early-stopping tuners such as Hyperband.

Report structure

A final-year report is marked on clarity as much as on code. A structure most universities accept:

1. **Abstract** (150 to 250 words): problem, method, main result.
2. **Introduction:** background, problem statement, objectives and scope.
3. **Literature review:** 8 to 15 papers or existing systems, and the gap your project fills.
4. **Requirements and design:** functional and non-functional requirements, architecture diagram, data flow from raw data to model to app, and the train/validation/test split plan.
5. **Implementation:** modules, tools, key algorithms and screenshots.
6. **Testing and results:** baseline versus final model on the right metric, a confusion matrix or error plot, and an honest limits section.
7. **Conclusion and future scope.**
8. **References** in the format your department asks for (IEEE style is common).

Follow your own university's template if it gives one; it overrides this list.

Common mistakes to avoid

- Reporting accuracy on imbalanced data such as fraud or disease.
- Fitting the scaler or encoder on the full dataset before splitting, which leaks test data.
- Using a random split for time-series data.
- Copying a notebook without being able to explain each cell.

Frequently asked questions

Which machine learning project is best for a final year?

One with a real public dataset, a clear metric and a working demo. Fraud detection, churn prediction, demand forecasting and recommendation systems are reliable choices because data and references are easy to find.

Can I do a machine learning project without a GPU?

Yes. Every project in this PDF uses tabular data and scikit-learn or gradient boosting, which run on a normal laptop or free Google Colab.

How many models should I compare?

At least a simple baseline and one or two stronger models. Examiners want to see why your final choice is better, measured on the same test data.

Is Kaggle data acceptable for a final year project?

Usually yes, if you cite the dataset and its licence and add your own work such as features, evaluation and an app. Check with your guide in case your department wants primary data.

Does this PDF include full source code?

No. It gives the problem, data, stack, modules and steps for each project. You write the code yourself, which is what examiners test in the viva.

Is this PDF free?

Yes. Fill in the short form on this page and the PDF downloads right away.

Sources and further reading

- UCI Machine Learning Repository
- Kaggle Datasets
- GroupLens: MovieLens datasets
- scikit-learn User Guide
- Fairlearn documentation

About this PDF

Prepared by the Uncodemy training team as practice project ideas. Datasets and tools named are public; check each dataset's licence before use. We update it every January. Corrections: info@uncodemy.com.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550