Machine Learning Mini Project Ideas PDF

A machine learning mini project should take one to four weeks, use a small clean dataset, and teach one core idea well, such as classification, regression or clustering. Spam detection, house price prediction, diabetes risk and customer segmentation are good first choices for 2nd and 3rd year students.

Key takeaways

- Keep the scope small: one dataset, one task, two models.
- Use clean public data such as UCI or Kaggle so you spend time learning, not scraping.
- Learn the full loop once: load, clean, split, train, evaluate, explain.
- A Jupyter notebook plus a short report is enough for most mini projects.
- These plans prepare you for a bigger final-year project later.

Who this PDF is for

This PDF is for 2nd and 3rd year B.Tech, BCA and B.Sc students who need a machine learning mini project for a lab, a semester submission or a first GitHub portfolio. Each idea fits in one to four weeks, runs on a normal laptop or Google Colab, and uses Python, pandas and scikit-learn.

The ideas are grouped by level. Do one beginner project fully before you try an intermediate one.

How to choose your project

- **Pick a task type you have studied:** classification, regression or clustering.
- **Prefer small datasets** (under a few lakh rows) so training takes seconds.
- **Plan in weeks:** week 1 data and EDA, week 2 models, week 3 results and report.
- **Write your own notebook.** Use tutorials for ideas, not for copying.

Beginner mini projects (1 to 2 weeks)

1. SMS Spam Classifier

- **Problem:** Classify text messages as spam or not spam.
- **Dataset:** UCI "SMS Spam Collection".
- **Tech stack:** Python, scikit-learn (TF-IDF, Naive Bayes).
- **Main modules:** Text cleaning, TF-IDF features, classifier, test-a-message box.

- **Steps:** Lowercase and clean text, build TF-IDF, train Multinomial Naive Bayes, check precision and recall.
- **Show in the viva:** How TF-IDF turns words into numbers.
- **Extensions:** Compare with logistic regression and a linear SVM.

2. House Price Prediction

- **Problem:** Predict sale price from house features.
- **Dataset:** Kaggle "House Prices - Advanced Regression Techniques".
- **Tech stack:** Python, pandas, scikit-learn.
- **Main modules:** Missing value handling, encoding, linear and tree models, error analysis.
- **Steps:** Fill missing values, log-transform price, compare linear regression, ridge and random forest using RMSE.
- **Show in the viva:** Why the log transform helps and which features matter most.
- **Extensions:** Try gradient boosting and compare.

3. Diabetes Risk Prediction

- **Problem:** Predict diabetes onset from basic health measurements.
- **Dataset:** Pima Indians Diabetes dataset (available on Kaggle).
- **Tech stack:** Python, scikit-learn, matplotlib.
- **Main modules:** Zero-value cleaning, scaling, KNN and logistic regression, confusion matrix.
- **Steps:** Replace impossible zeros, scale, tune k for KNN, compare with logistic regression.
- **Show in the viva:** How k affects KNN and why scaling matters for it.
- **Extensions:** Add a simple input form with Streamlit.

4. Titanic Survival Prediction

- **Problem:** Predict which passengers survived the Titanic.
- **Dataset:** Kaggle "Titanic - Machine Learning from Disaster".
- **Tech stack:** Python, pandas, scikit-learn.
- **Main modules:** Feature engineering (title, family size), decision tree, cross-validation.
- **Steps:** Extract titles from names, create family size, train a decision tree, plot it, cross-validate.
- **Show in the viva:** Reading the decision tree splits aloud.
- **Extensions:** Submit to the Kaggle competition and note your score.

Intermediate mini projects (2 to 3 weeks)

5. Customer Segmentation with Clustering

- **Problem:** Group shoppers by buying behaviour for targeted offers.
- **Dataset:** UCI "Online Retail" dataset.
- **Tech stack:** Python, pandas, scikit-learn (K-Means), Plotly.
- **Main modules:** RFM features, scaling, K-Means, segment profiles.
- **Steps:** Compute recency, frequency and monetary values, choose k with the elbow and silhouette methods, name each segment.
- **Show in the viva:** How you chose k and what each segment means for a marketer.

- **Extensions:** Compare with DBSCAN or hierarchical clustering.

6. Wine Quality Classifier

- **Problem:** Classify wine as good or average from lab tests.
- **Dataset:** UCI "Wine Quality" dataset.
- **Tech stack:** Python, scikit-learn, seaborn.
- **Main modules:** Label creation, correlation study, random forest, feature importance.
- **Steps:** Convert score to classes, check correlations, train random forest, plot importances.
- **Show in the viva:** Which chemical properties matter and how you checked.
- **Extensions:** Treat it as regression and compare.

7. Loan Default Risk Scorer

- **Problem:** Score loan applicants by default risk.
- **Dataset:** UCI "Statlog (German Credit Data)".
- **Tech stack:** Python, scikit-learn, Streamlit.
- **Main modules:** Encoding, cost-sensitive model, score bands, simple app.
- **Steps:** Encode categories, train logistic regression with class weights, convert probabilities into risk bands.
- **Show in the viva:** Why missing a bad loan costs more than rejecting a good one.
- **Extensions:** Add a fairness check by age group.

8. Mobile Price Range Classification

- **Problem:** Predict a phone's price range from its specs.
- **Dataset:** Kaggle "Mobile Price Classification".
- **Tech stack:** Python, scikit-learn.
- **Main modules:** EDA, SVM and random forest, hyperparameter search.
- **Steps:** Explore which specs relate to price, train SVM with grid search, compare with random forest.
- **Show in the viva:** The kernel and C values you chose and why.
- **Extensions:** Add a spec input page.

Stretch mini projects (3 to 4 weeks)

9. Movie Recommender (Item-based)

- **Problem:** Suggest similar movies to one a user liked.
- **Dataset:** MovieLens 100K from GroupLens.
- **Tech stack:** Python, pandas, scikit-learn (cosine similarity).
- **Main modules:** Rating matrix, item similarity, top-N list, simple UI.
- **Steps:** Build the user-item matrix, compute cosine similarity between movies, return top 10 similar titles.
- **Show in the viva:** How cosine similarity works on rating vectors.
- **Extensions:** Add genre filtering.

10. Energy Consumption Forecast

- **Problem:** Forecast household power use for the next hours.
- **Dataset:** UCI "Individual Household Electric Power Consumption".
- **Tech stack:** Python, pandas, scikit-learn, statsmodels.
- **Main modules:** Resampling to hourly, lag features, regression model, plots.
- **Steps:** Resample, create lag and hour features, train with a time-based split, compare with a naive forecast.
- **Show in the viva:** Why you beat (or did not beat) the naive forecast.
- **Extensions:** Try daily forecasting with ARIMA.

11. Mushroom Edibility Classifier

- **Problem:** Classify mushrooms as edible or poisonous from physical traits.
- **Dataset:** UCI "Mushroom" dataset.
- **Tech stack:** Python, scikit-learn.
- **Main modules:** One-hot encoding, decision tree, rule extraction.
- **Steps:** Encode all categorical columns, train a shallow tree, export readable rules.
- **Show in the viva:** The few rules that separate the classes.
- **Extensions:** Compare tree depth against accuracy in a chart.

12. Iris and Beyond: Model Comparison Notebook

- **Problem:** Compare six classic classifiers on three small datasets.
- **Dataset:** UCI "Iris", "Wine" and "Breast Cancer Wisconsin (Diagnostic)".
- **Tech stack:** Python, scikit-learn, matplotlib.
- **Main modules:** Shared pipeline, cross-validation loop, results table, decision boundary plots.
- **Steps:** Write one pipeline, run KNN, Naive Bayes, SVM, tree, forest and logistic regression, tabulate scores.
- **Show in the viva:** Why no single model wins every dataset.
- **Extensions:** Add timing and model size columns.

Mini project report structure

A mini project report is usually 10 to 20 pages. Keep it short and clear:

1. **Title page and certificate** in your department's format.
2. **Introduction:** the problem in one paragraph and your objectives as a list.
3. **Tools used:** Python version, libraries and where you ran the code (laptop or Colab).
4. **Design:** one block diagram or flowchart, plus the train/test split you used.
5. **Implementation:** what each module does, with 3 to 6 screenshots.
6. **Results:** a table of metrics for each model and one chart.
7. **Conclusion and future scope,** then references.

Ask your guide for the exact format before you start writing.

Common mistakes to avoid

- Choosing a huge dataset that takes hours to load.
- Testing on the training data and reporting perfect scores.
- Skipping EDA and missing obvious data errors.
- Writing the report on the last day without screenshots.

Frequently asked questions

What is a good machine learning mini project for beginners?

SMS spam detection, house price prediction and diabetes risk prediction are good starts. They use small public datasets and teach classification or regression clearly.

How long should a mini project take?

Usually one to four weeks. Plan one week for data and EDA, one for models and one for results and the report.

Do I need deep learning for a mini project?

No. Classic scikit-learn models are enough and easier to explain. Try deep learning later in a bigger project.

Can a mini project become my final year project?

Yes. Many students extend a mini project with more data, better models and an app. Tell your guide what is new in the final version.

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
- scikit-learn tutorials
- GroupLens: MovieLens datasets

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