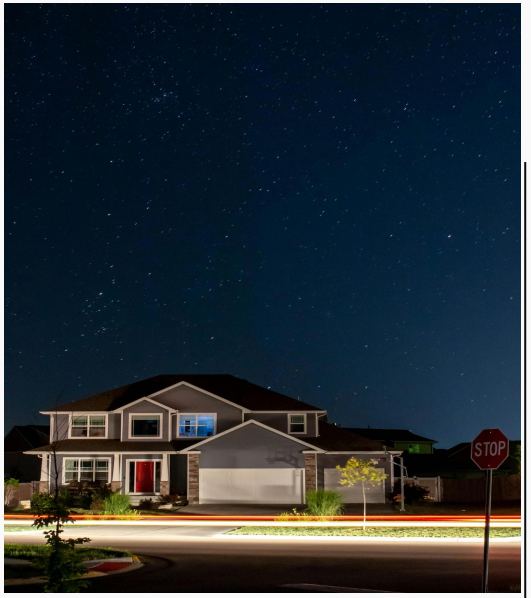
Machine Learning 2 -
Housing Price & Drinking Behaviour

Zahra Eshtiaghi (476679)
Tsoi Kwan Ma (476914)

House Prices Regression (Problem & Data)



Goal

Predict each home's SalePrice from house attributes.
I train on labeled homes and generate predictions for the Kaggle test set.

Dataset snapshot:

- train.csv: 1460 rows × 81 columns (includes target SalePrice)
- test.csv: 1459 rows × 80 columns (no SalePrice)
- Explanatory variables: 79
- Mix of numeric + categorical features
- Train/Validation split used in the notebooks (80/20):
  - X_train: 1168 × 80
  - X_val : 292 × 80

Evaluation metric

RMSE on log(SalePrice) → "log-RMSE / RMSLE-style".
Why: house prices are skewed; log makes errors more relative than absolute.

I model in log-space

...then convert back for final submission.

Workflow Overview

How the work is split across 3 notebooks

1) Preprocessing

- Missing values + types
- Skewness handling
- Encoding + scaling
- Save prepared_data.pkl

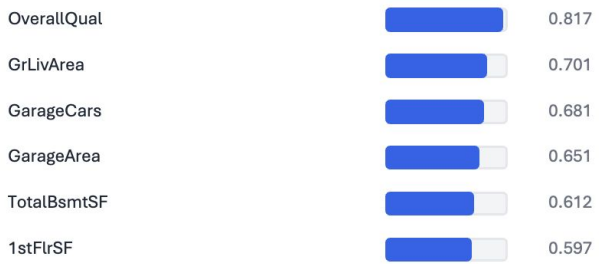
2) Feature engineering

- Correlation checks
- Fix encoding issues
- Create new features
- Save engineered dataset

3) Modeling

- LASSO feature selection
- Train (XGB/LGBM/Cat/RF/Ensemble)
- Compare + validate
- Create submission file

Top numeric predictors (corr with log SalePrice)



Engineered features (examples)

I added 12 engineered features (age, totals, interactions, ratios).
A few of the strongest new features by |correlation|:

HouseAge (|r|≈0.421)
RemodAge (|r|≈0.420)
GarageAge (|r|≈0.269)
TotalPorch (|r|≈0.202)
Result: richer signals before modeling.

Sanity checks (example)
CentralAir initially showed a negative correlation.
I checked value counts + mean price by category,
then flipped the encoding so “1 = has AC”.
After the fix, correlation became positive (as expected).

Notebooks: 1- regression.ipynb • 2- regression.ipynb • 3-regressionC.ipynb

Modeling Results

LASSO feature selection + boosting models

Modeling approach (Notebook 3)

Prepared + engineered features

↓

LASSO feature selection

↓

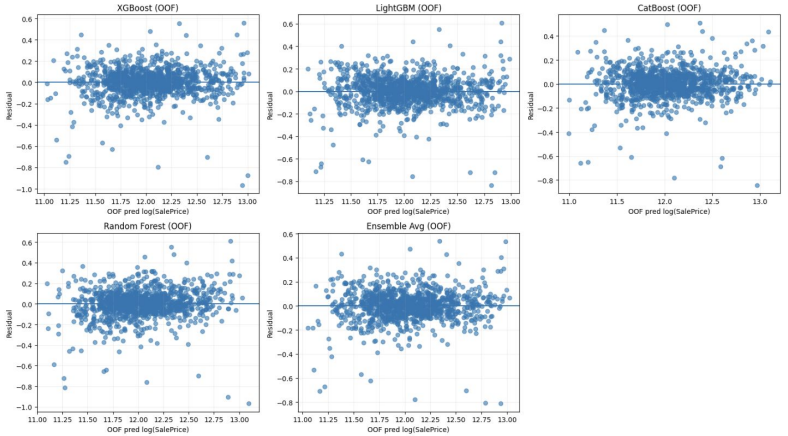
Training five models: XGBoost, LightGBM, CatBoost, RandomForest, Ensemblig Avg3boosting models

↓

Predict log(SalePrice) → XGBosst

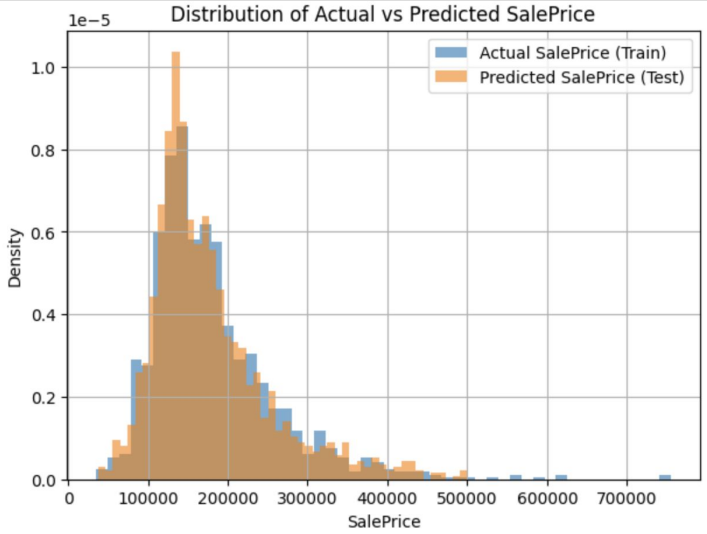
Best tuned CV log-RMSE: 0.1243

Residual plots (OOF 5-Fold)



	Model	Train log-RMSE (mean)	Validation log-RMSE (mean)	Overfit Gap (Val - Train)	Validation Std (stability)
0	CatBoost	0.0753	0.1209	0.0456	0.0163
1	Ensemble (avg 3 boosting)	0.0876	0.1236	0.0359	0.0170
2	XGBoost	0.1046	0.1302	0.0257	0.0183
3	LightGBM	0.0952	0.1304	0.0352	0.0176
4	Random Forest	0.0687	0.1384	0.0697	0.0170

Results and final model



Actual prices go up to ~755,000 Predicted prices go up to ~500,000

This shows that the model avoids extreme predictions. This is a sign of controlled overfitting, not a mistake.

In real applications, it is usually better to:

slightly under-predict extreme values than to overfit noise.

Id	SalePrice
1461	118634.41
1462	153415.58
1463	171287.02
1464	190779.58
1465	189893.95
1466	171330.95
1467	172103.92
1468	169181.38
1469	194125.58
1470	122829.68
1471	203992.7
1472	95045.086

Statistic	Actual (Train)	Predicted (Test)
Mean	180,921	177,665
Median	163,000	158,304

Statistic	Actual	Predicted
Std	79,443	75,251
25%	129,975	128,404
75%	214,000	206,768

Why Xgboost?

- Smallest overfit gap(val – train) across folds
- Stable performance vs Catboost/ Ensemble
- Residual plots looked more compact/ centerd
- Final outpoot: textprediction_xgboost.csv

Classification - Drinking Behaviour

Objective: Predict **drinking behaviour (Yes / No)** using various body signals (e.g. blood pressure / liver enzymes)

- Data: National Health Insurance Service in Korea
- Train: 594,807 rows (60%)
- Validation: 198,269 rows (20%)
- Test: 198,270 rows (20%)

To avoid data leakage, data split done beforehand
- Total observations: 991,346
- Total columns: 24 (including target variable)

Data Preprocessing & EDA
- No missing values
- Balanced target distribution (49 / 51)
- Lots of outliers on numeric features so abnormally distributed + highly skewed

Feature Selection & Engineering
- Spearman correlation (ordinal & numeric)
- ANOVA (target with numeric features)
- Cramér's V (target and nominal variables)
- Log transformation on skewed numerics

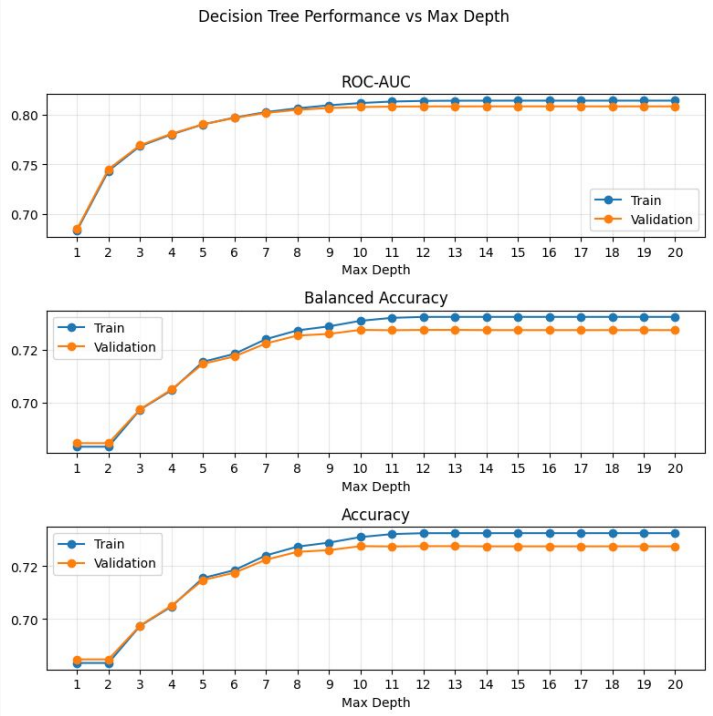
Evaluation Metrics on Models

1. ROC-AUC: whether well-separate between drinkers and non-drinkers across all thresholds
2. BA: assess performance at fixed threshold while ensuring equal importance of both classes

Model Training

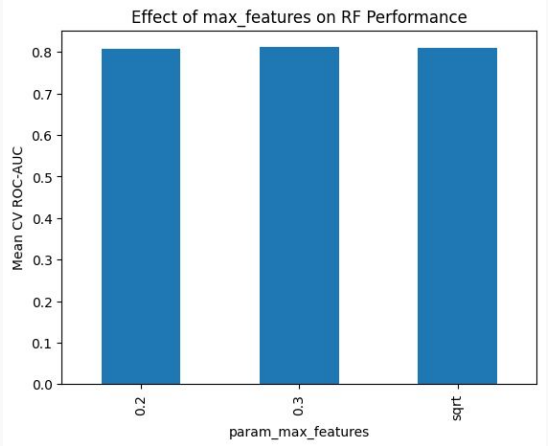
Classification Decision Tree (Baseline - CART)

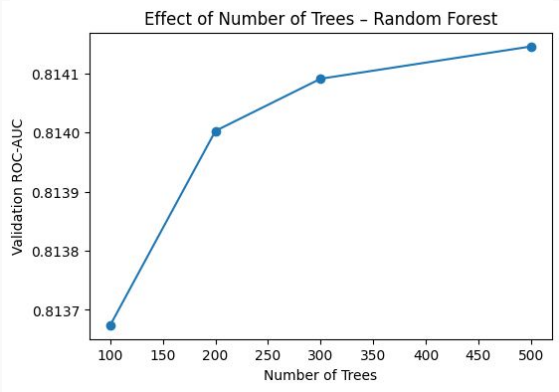
```
dt_parameters = {
    "criterion": ["gini", "entropy"],
    "max_depth": list(range(4, 21)) + [None], # pre-pruning
    "min_samples_split": [200, 500, 1000, 1200, 1500, 2000], # pre-pruning
    "min_samples_leaf": [100, 200, 500, 800, 1000, 1200] # pre-pruning
}
```



Random Forest (subsample ~ 200k)

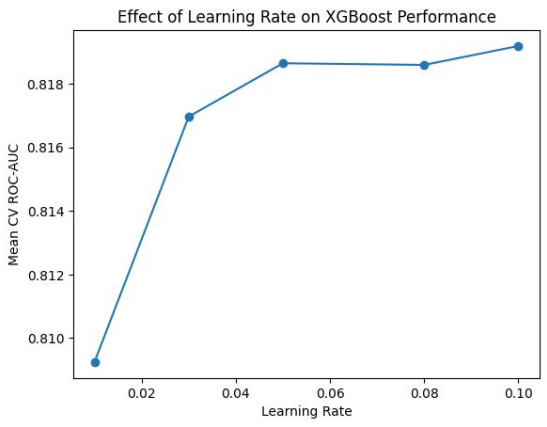
OOB (ROC-AUC): 0.7325

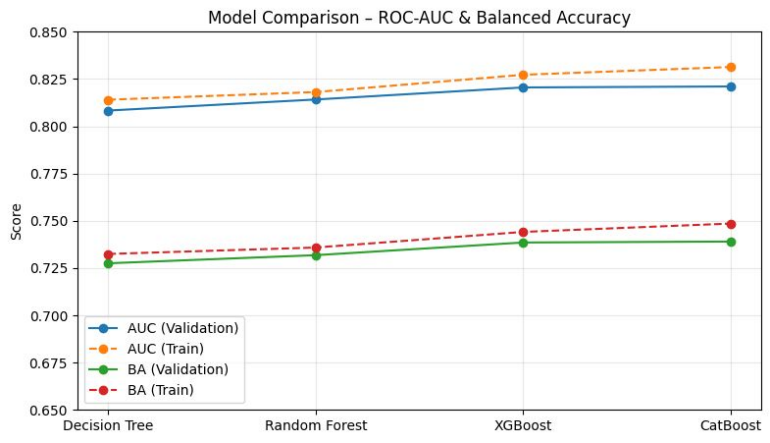


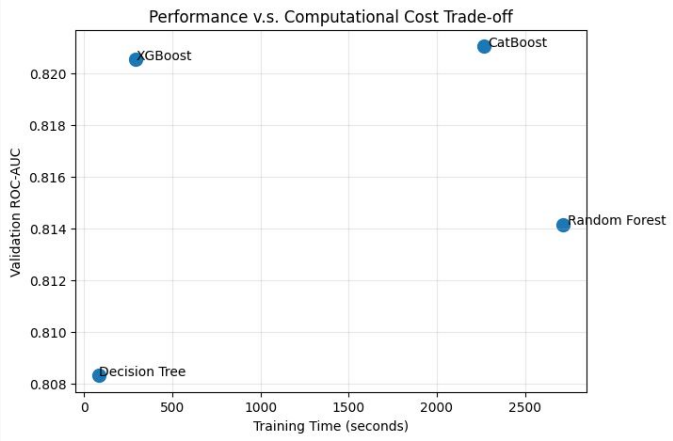


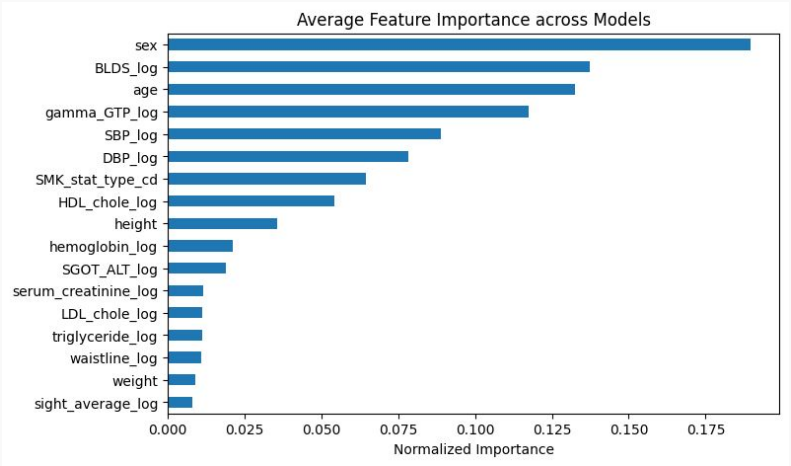
Model Training & Evaluation

XGBoost / CatBoost



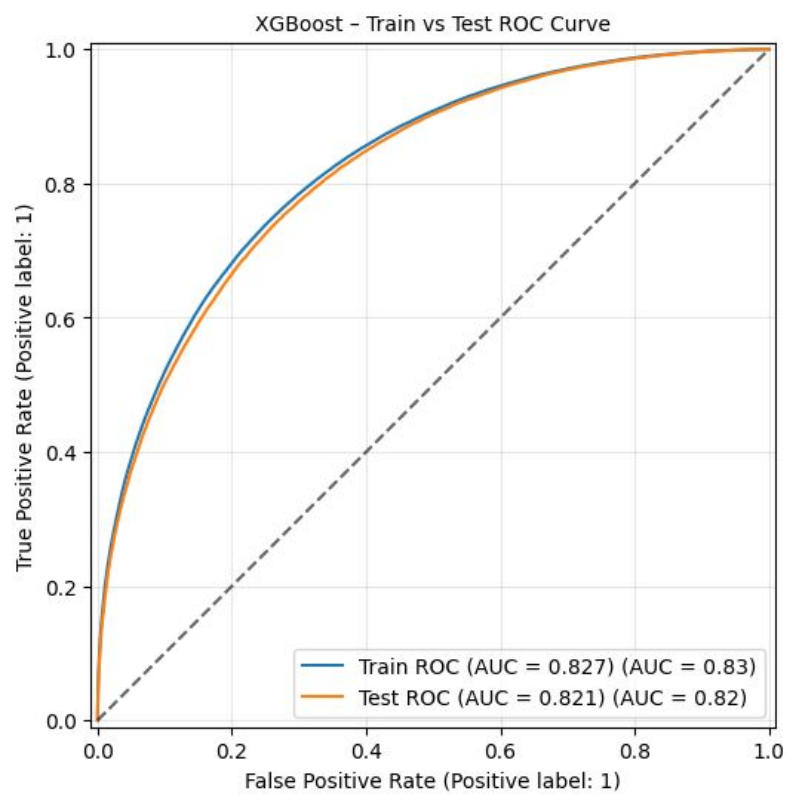


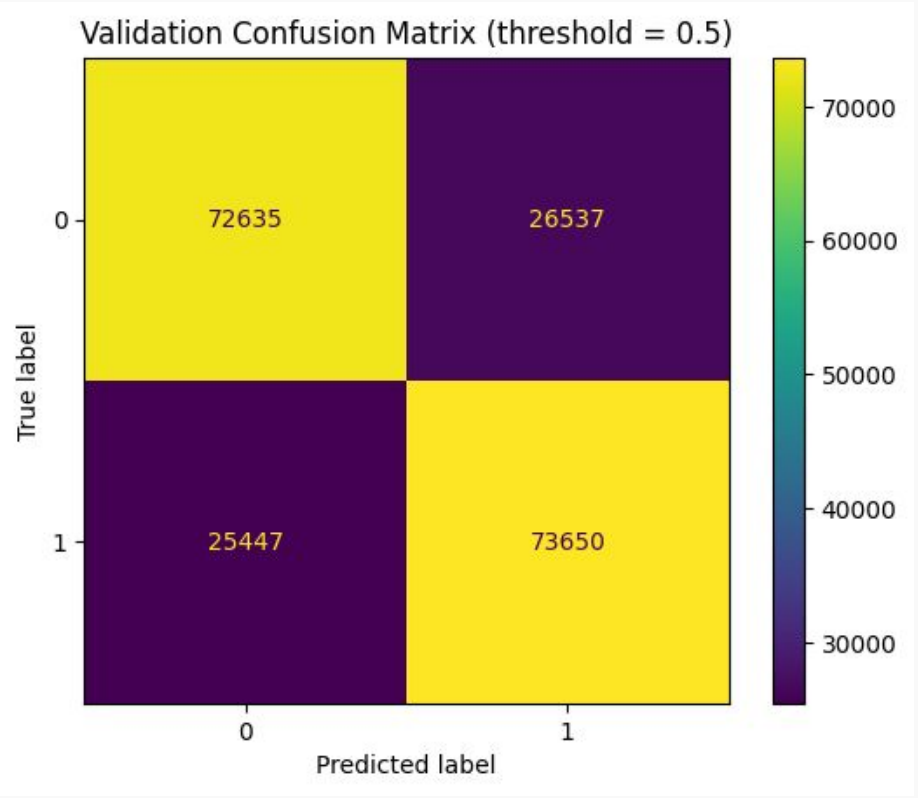




Model Final Test - XGBoost

TRAIN - ROC-AUC: 0.8272, Balanced Acc: 0.7441, Acc: 0.7441, Prec: 0.7411, Recall: 0.7502
TEST - ROC-AUC: 0.8199, Balanced Acc: 0.7378, Acc: 0.7378, Prec: 0.7351, Recall: 0.7432





The End