DSO 530 Final Project

Insurance Loss Analytics Project

Presented by:
Adam Young, Dominic He, Emma Liu,
Danny Ko, Min Yu Ho, Zitong Wang

Team Members

Adam Young

Dominic He

Emma Liu

Danny Ko

Min Yu Ho

Zitong Wang

Project Background

Current scenario:

Automobile Insurance companies have been using

predictive analytics to:

- Manage Risk
- Forecast Claims
- Identify Fraud
- Optimize Costs

Project Breakdown

LC (Loss Cost)

- Understand how risky a particular segment is
- **High** → high severity of claims, underpriced
- **Low** → lower risk, potential for competitive pricing

HALC (Historical LC)

- Standardized way to compare policy holders, controlling time
- **High** → frequent claims, higher risk
- **Low** →safer drivers, more favorable risk

CS (Claim Status)

- Claim frequency calculation, predicting likelihood of future claims. Important in determining risk.
- **1** → a claim is made
- **0** → no claim

Variable Selection Predictions

Important	Less Important
X.25: Market Value higher value - more expensive claims **X.8: Years with Insurer** more experience = safer behavior	**X.13: Last Payment Method** yearly or half-yearly - do they pay on time **X.26: Number of Doors** redundant to weight

Note:

- Lots of variables account for the same issues
  - X.23-28: Car features
  - X.8-11: Loyalty
- Assumption: Redundant variables can increase noise → LASSO feature selection should be considered

Data Wrangling

Model Preprocessing

Dataset Preparation

Columns Included:

- X.7 - X.14, X.19 - X.29, Age, License_years, Policy_years
- X.27: rows with missing(NA) values were removed

Added columns:

- **Age, License_years, Policy_years**: Derived from date columns by calculating the number of years since birth, license issuance, and policy start

Data Classification:

- Numerical variables: All other continuous variables
- Categorical variables: X.7, X.13, X.19, X.20, X.21, X.27

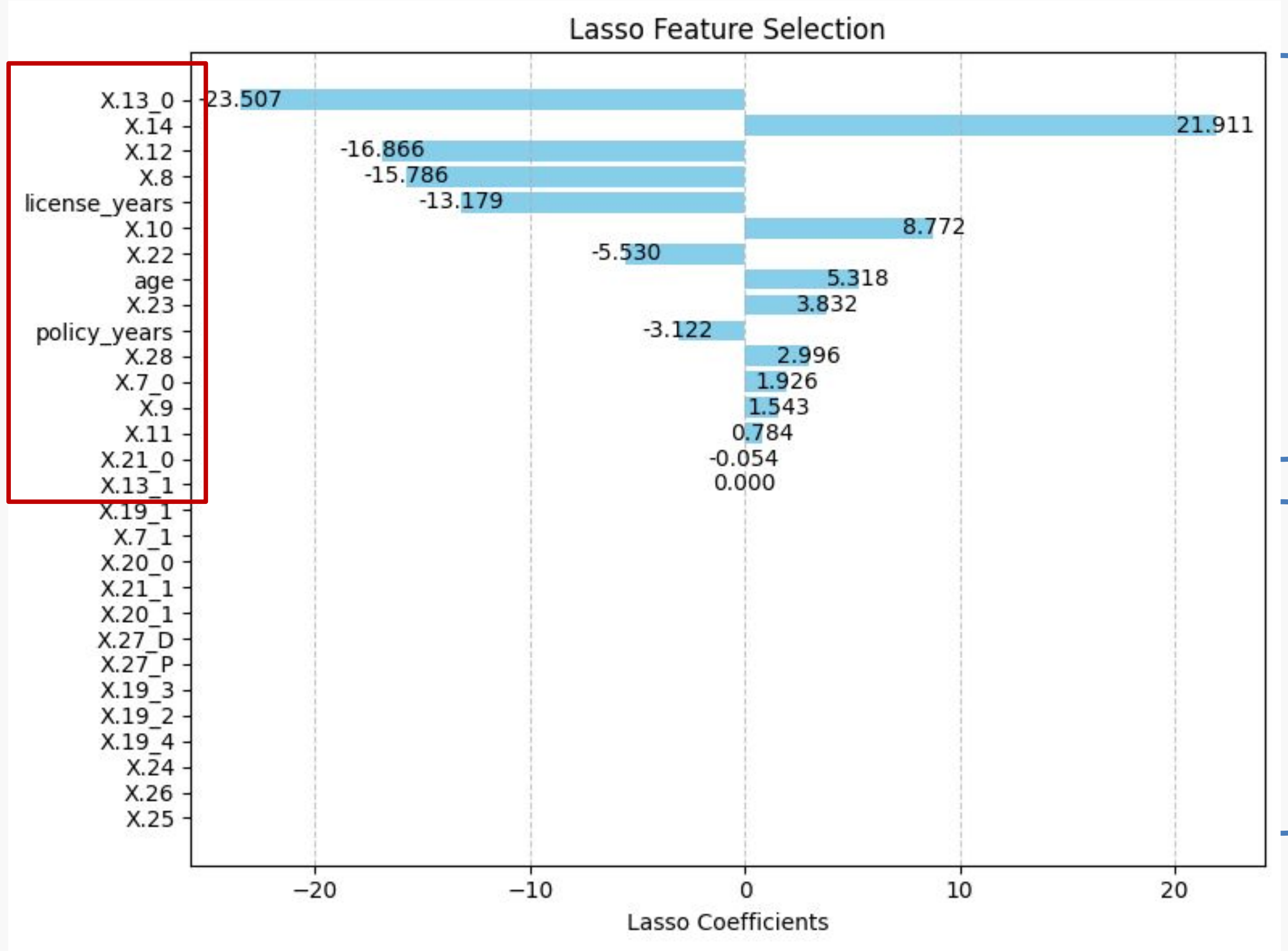
Preprocessing:

- Numerical variables: Standardize with **StandardScaler()** to handle variable magnitude
- Categorical variables: Use **OneHotEncoder(handle_unknown='ignore')** to create dummy variable

Feature Selection – Shrinkage Methods (Lasso)

GridSearchCV for 5-folds
alpha : {0.01, 0.1, 0.05, 5.0, 10.0}

- ***# of Selected Features: 13***
- ***# of Eliminated Features: 16***



Highlights more on Policy and Driver-Related Features

- X.13_0: Last payment method (half-yearly vs. annual)
- X.14: Net premium amount during the current year
- X.12: Number of policies canceled or terminated during the current year
- X.8: Insurance Years
- Driver license years

Shrinkage of Vehicle-Related Features Coefficients

- X.24: Cylinder capacity of the vehicle
- X.25: Market value of the vehicle as of 31/12/2019
- X.26: Number of vehicle doors

Limitation:

- **Linear Assumption Between Y and X**
- **Variance-Bias Trade-Off**

Models Overview

**all features included for model performance*

Predicting LC & HALC (Task 1)

- In the dataset, over 89% of policyholder did not file a claim, making LC and HALC highly **right skewed**; hence **Tweedie regression** is used to predict these variables

- Below are different Tweedie model we tried:
  - TweedieRegressor
  - LightGBM + Tweedie
  - XGBoost + Tweedie
  - Two-step model

Predicting Claim Status (Task 2)

- To predict whether the policyholder filed a claim or not during the year, we used different **classification models** to predict the binary variable

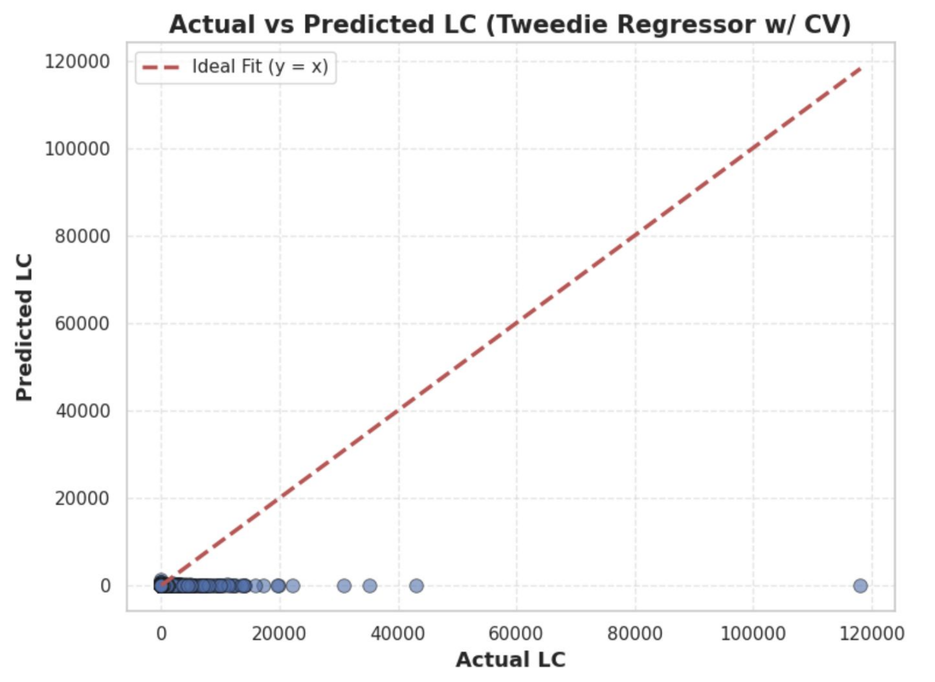
- Below are models we tried:
  - Random Forest
  - LightGBM
  - XGBoost

Task 1

Model Selection – LC

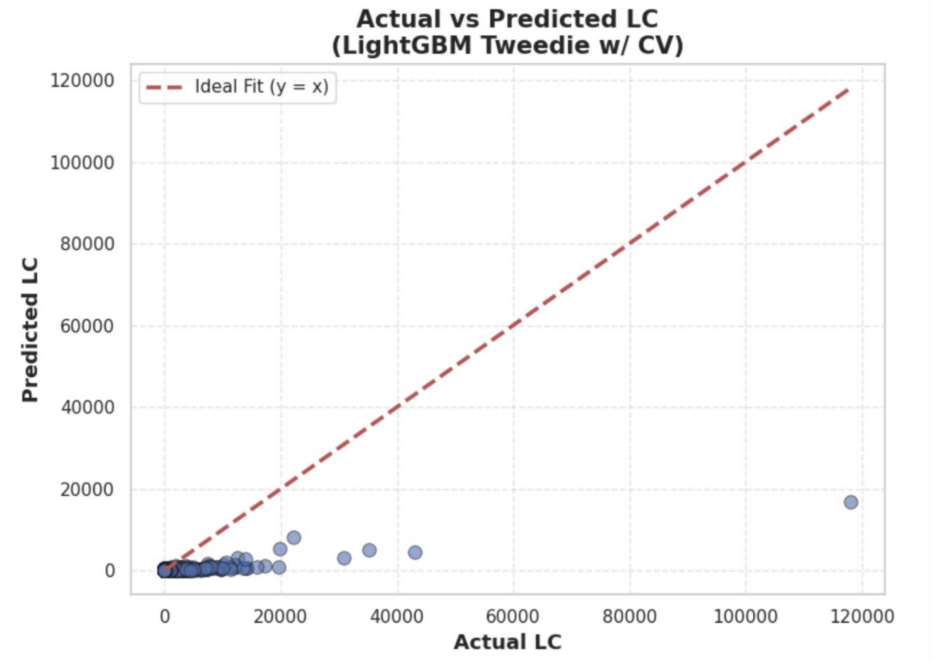
*all LC model parameters are tuned with **5-fold Cross Validation**

Tweedie Regressor



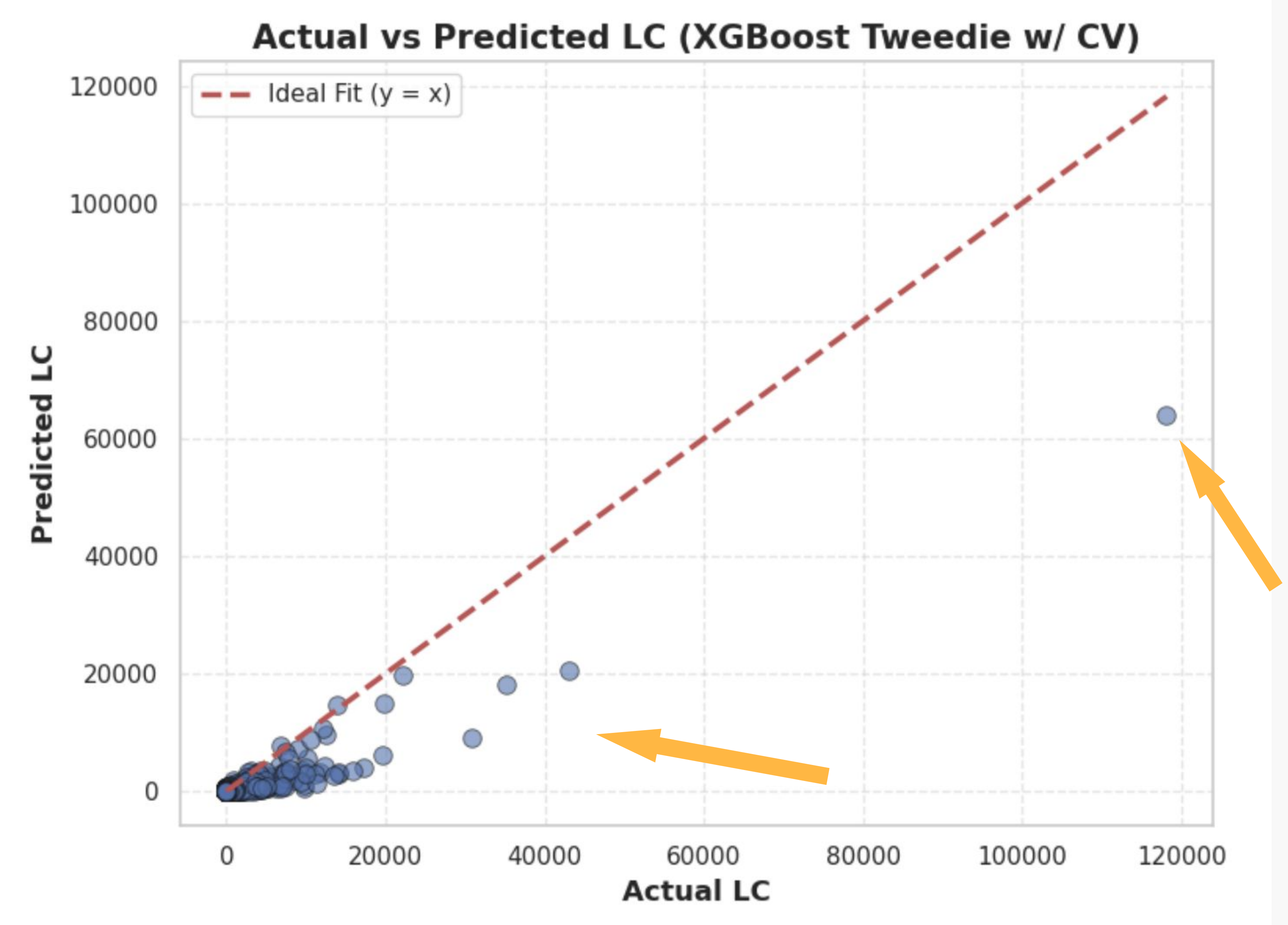
- **MSE**: 712,989
- Overestimates zero-claim losses
- Struggles due to lack of model complexity

LightGBM Tweedie



- **MSE**: 549,996
- Better fit than linear Tweedie
- Still underperforms on high-loss cases

XGBoost Tweedie - Tuned



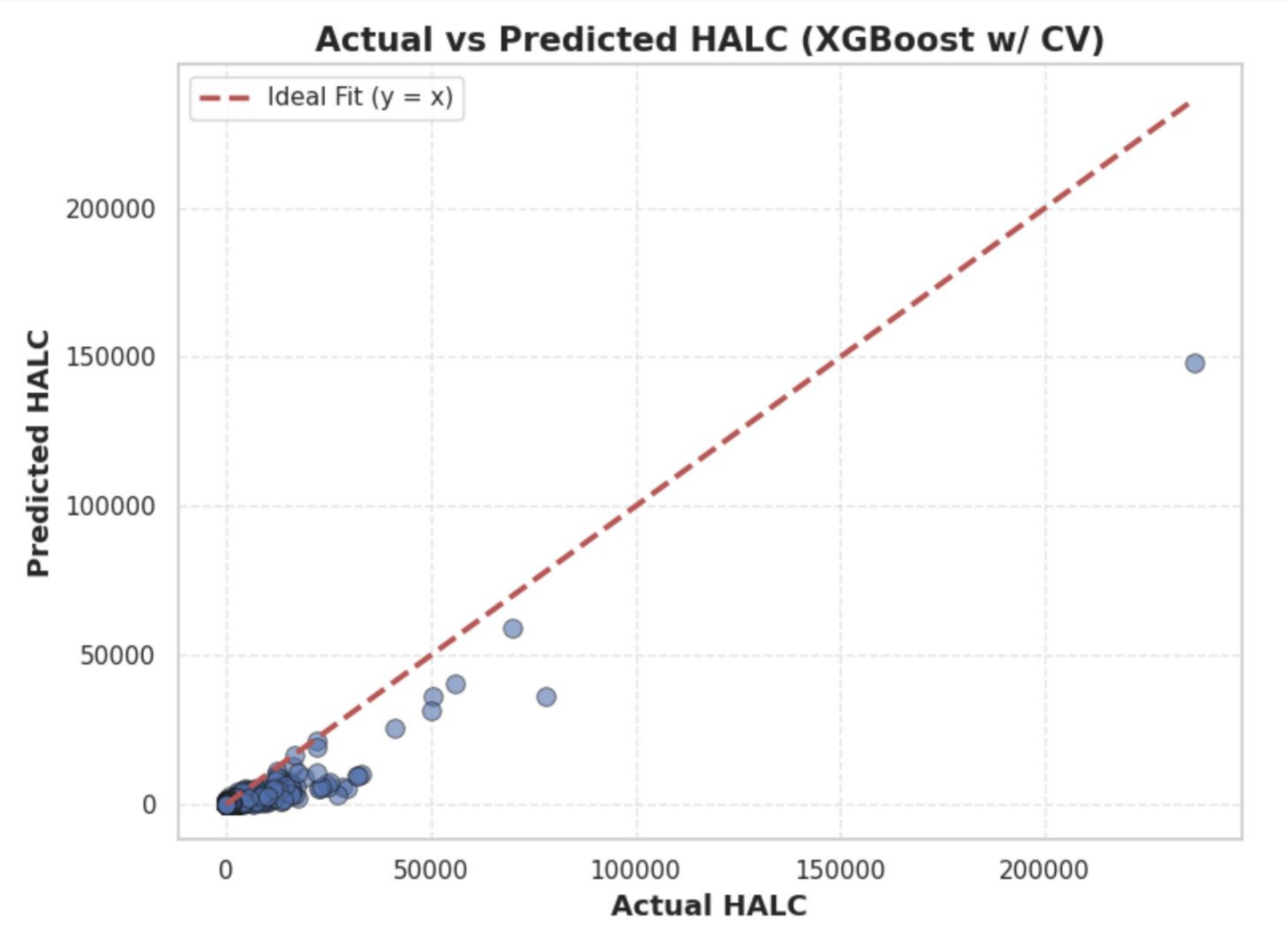
- **MSE**: 217,392
- Best fit across full LC range
- ~**60%** improvement over LightGBM Tweedie
- Best params: α = 1.0, variance power = 1.5

Task 1

Model Selection – HALC

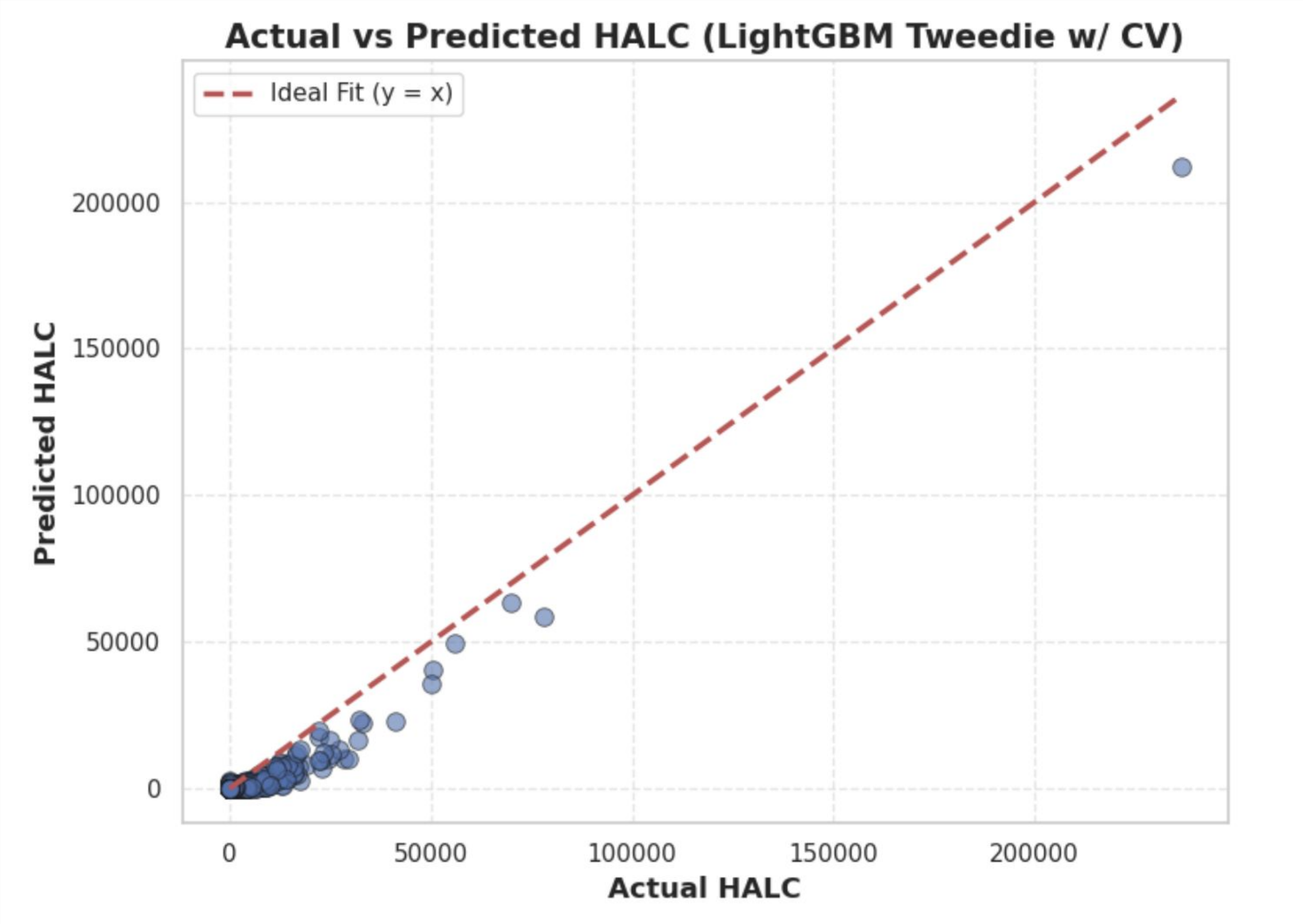
**all HALC model parameters are tuned with 5-fold Cross Validation*

XGBoost Tweedie - Tuned



- **MSE**: 653,948
- Less consistent fit across the full HALC spectrum

LightGBM Tweedie - Tuned



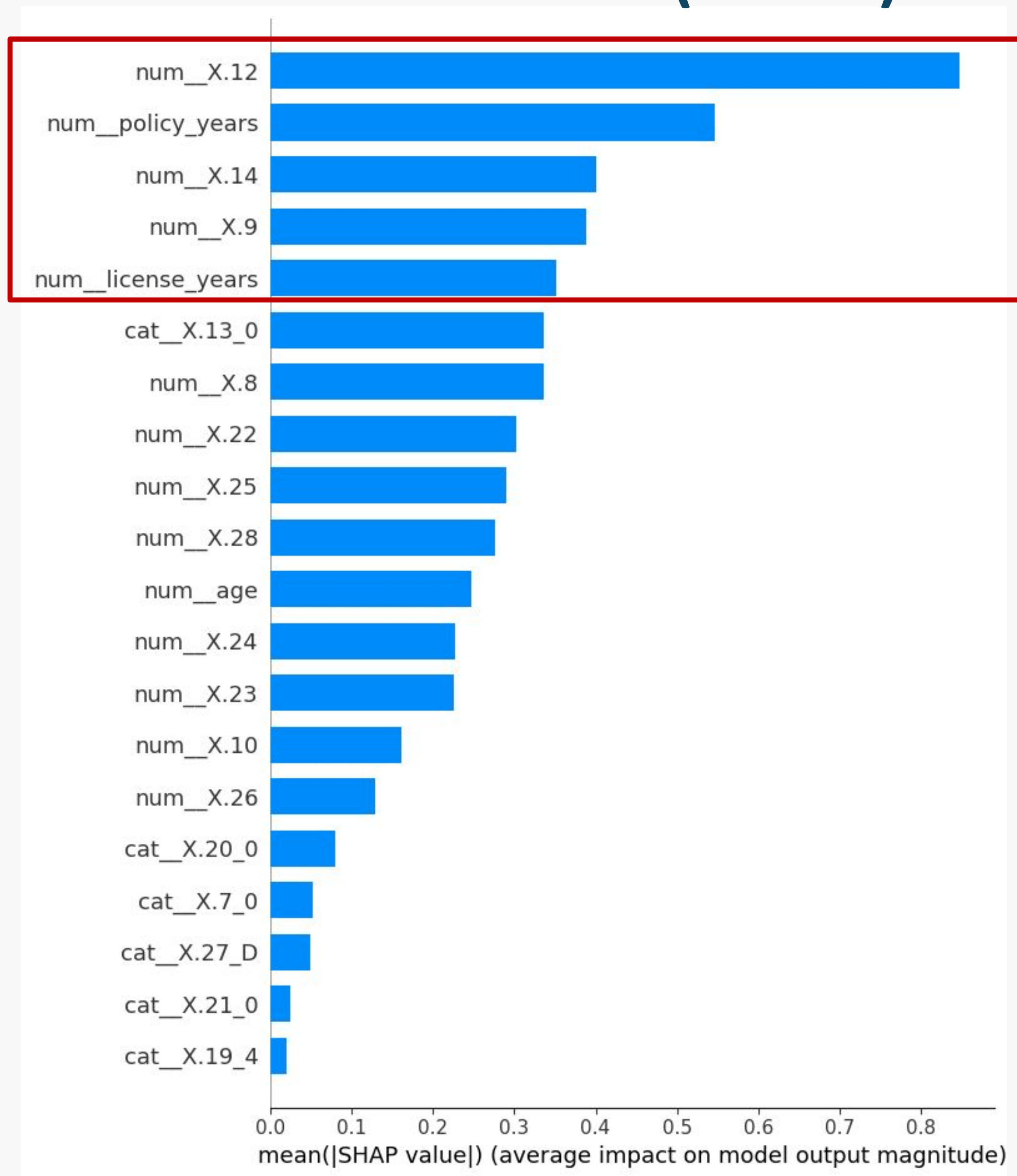
- **MSE**: 360,881
- Best fit across full HALC range
- Performs well across both low and moderately high HALC values
- Best params: α = 0.5, variance power = 1.1

Task 1

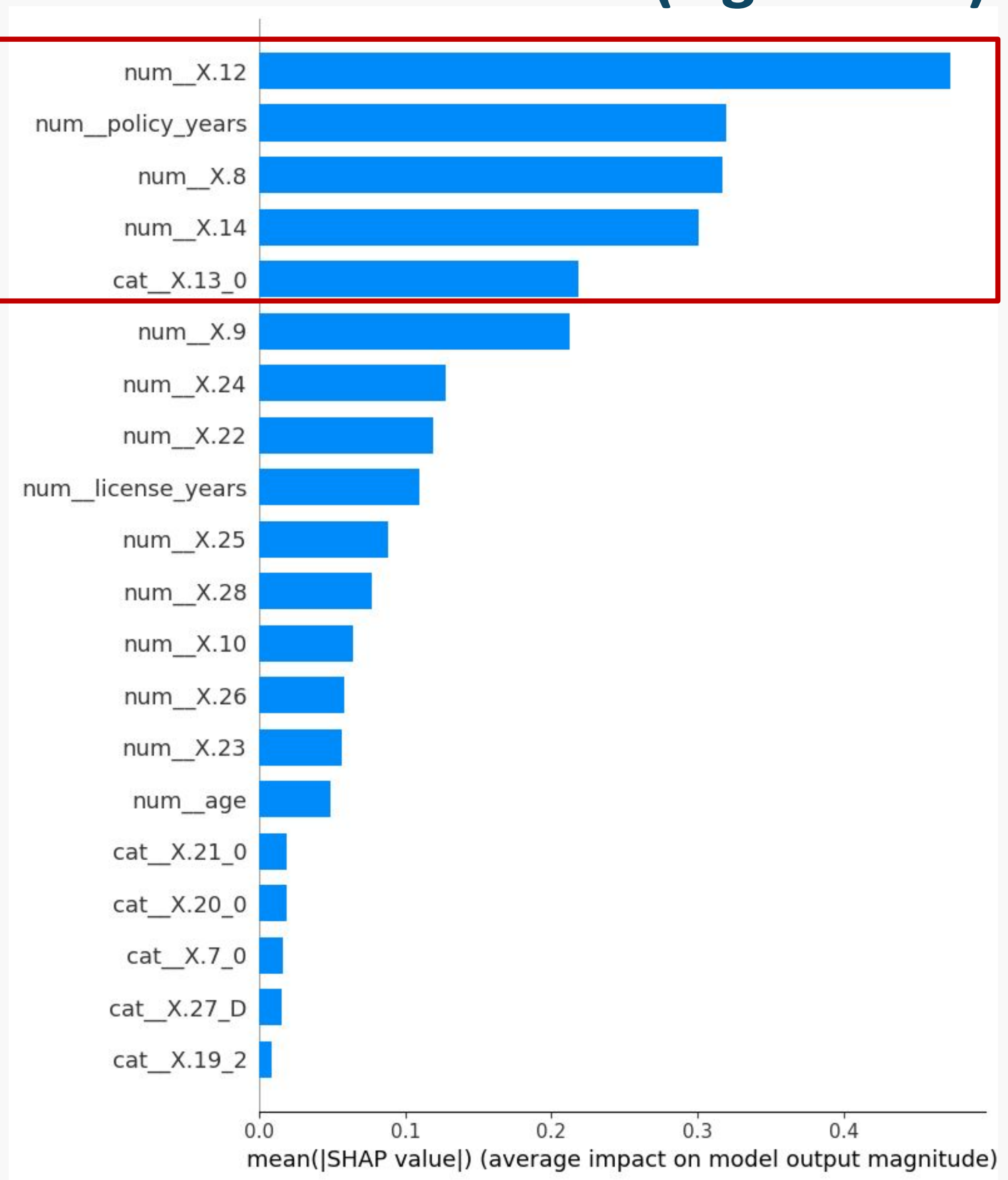
Model Interpretation – LC vs. HALC

Magnitude of Feature Impact on Prediction

SHAP Value for LC (XGBM)



SHAP Value for HALC (LightGBM)



Top Features for Predicting LC

- X.12: # of policies canceled or terminated
- Policy years
- X.14: Net premium amount
- X.9: # of policies held during the current year
- Driver license years

Top Features for Predicting HALC

- X.12: # of policies canceled or terminated
- Policy years
- X.8: Insurance years
- X.14: Net premium amount
- X.13_0: Last payment method (half-yearly vs. annual)

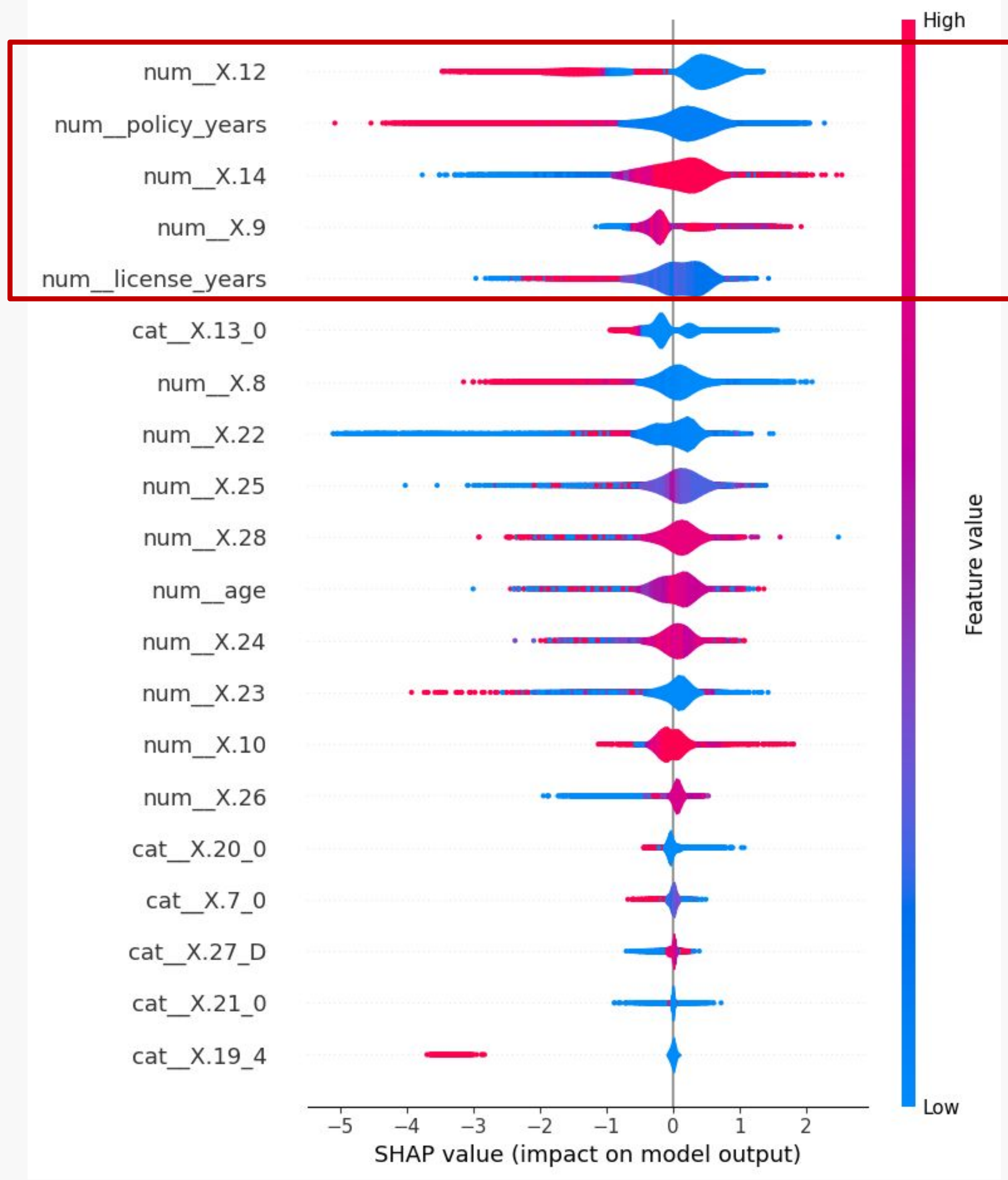
- **Model for LC is focused on short-term risk exposure related to drivers**
- **Model for HALC is more affected by long-term factors**

Task 1

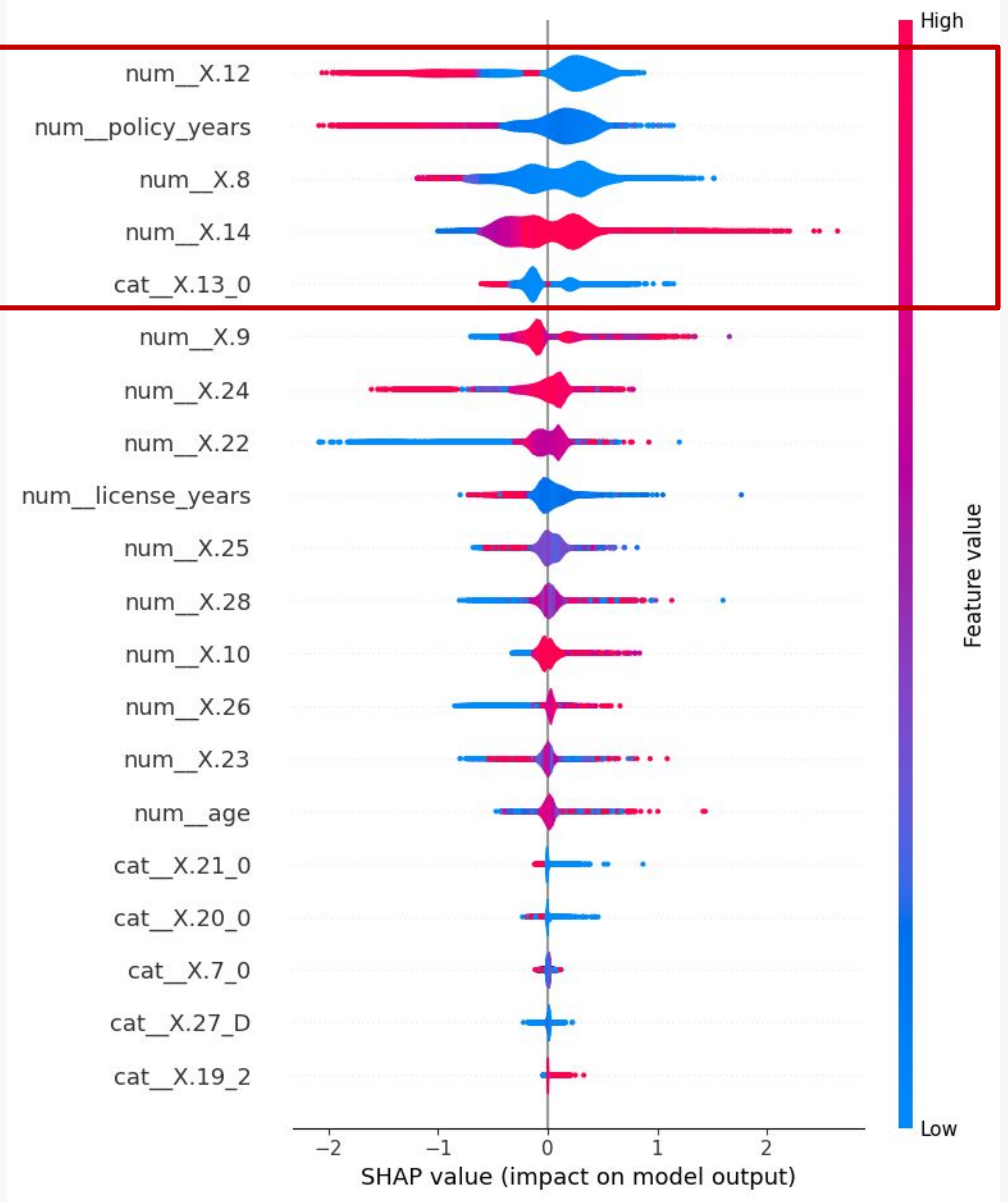
Model Interpretation – LC vs. HALC

Direction of Feature Impact on Prediction

Violin Chart for LC (XGBM)



Violin Chart for HALC (LightGBM)



Top Features for Predicting LC

X.12: # of policies canceled or terminated	Negative
Policy Years	Negative
X.14: Net premium amount	Positive
X.9: # of policies held during the current year	Positive
Driver license years	Negative

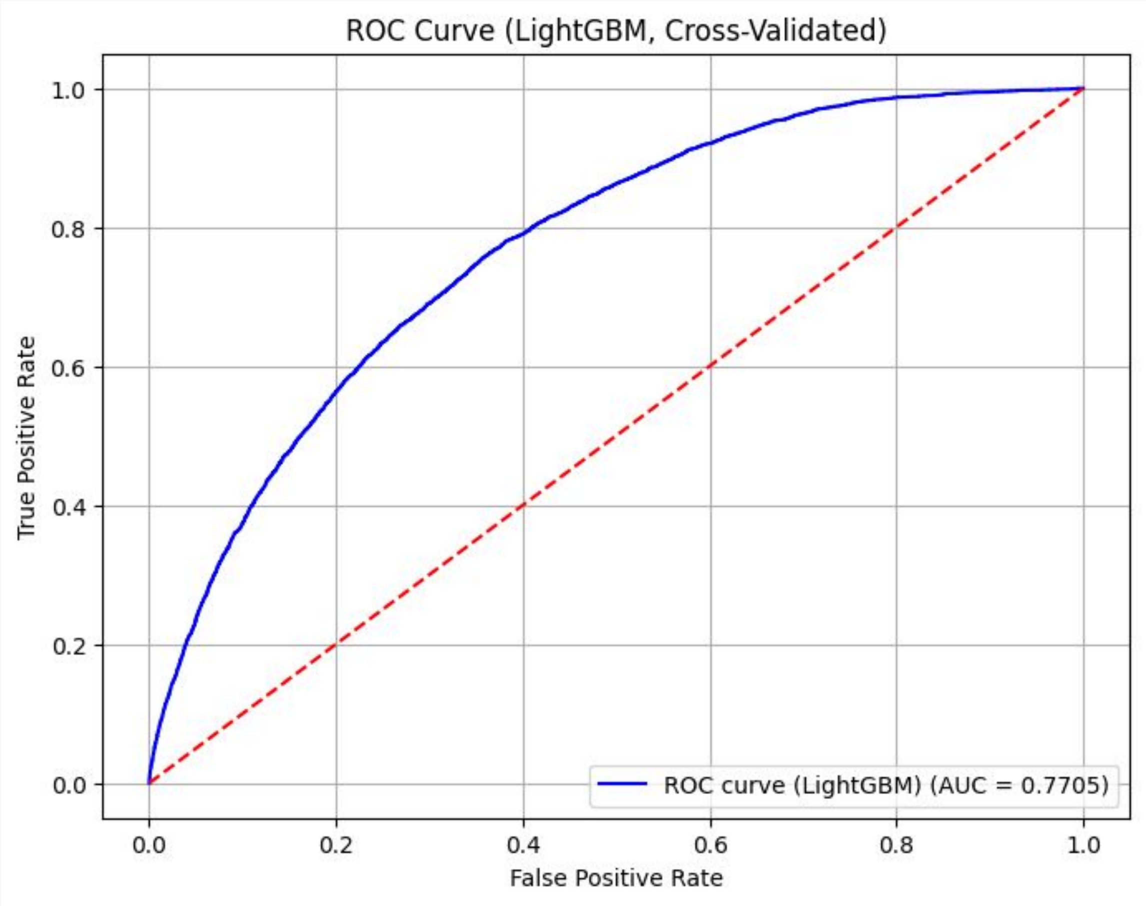
Top Features for Predicting HALC

X.12: # of policies canceled or terminated	Negative
Policy Years	Negative
X.8: Insurance years	Negative
X.14: Net premium amount	Positive
X.13_0: Last payment method (half-yearly vs. annual)	Negative

Task 2

Model Selection – Claim Status

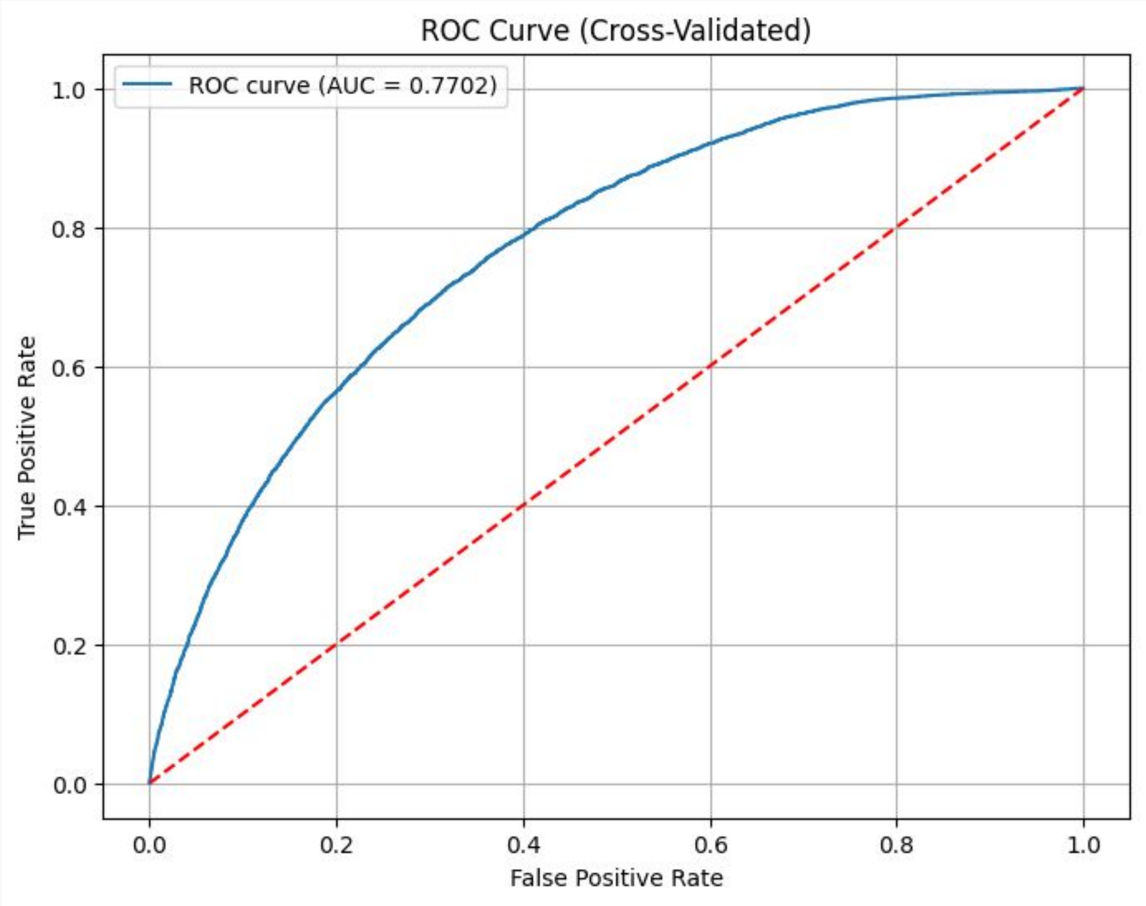
Light Gradient Boosting Machine (LightGBM)



Best Parameters:

- learning_rate = 0.05
- max_depth = -1
- n_estimators = 100
- num_leaves = 31

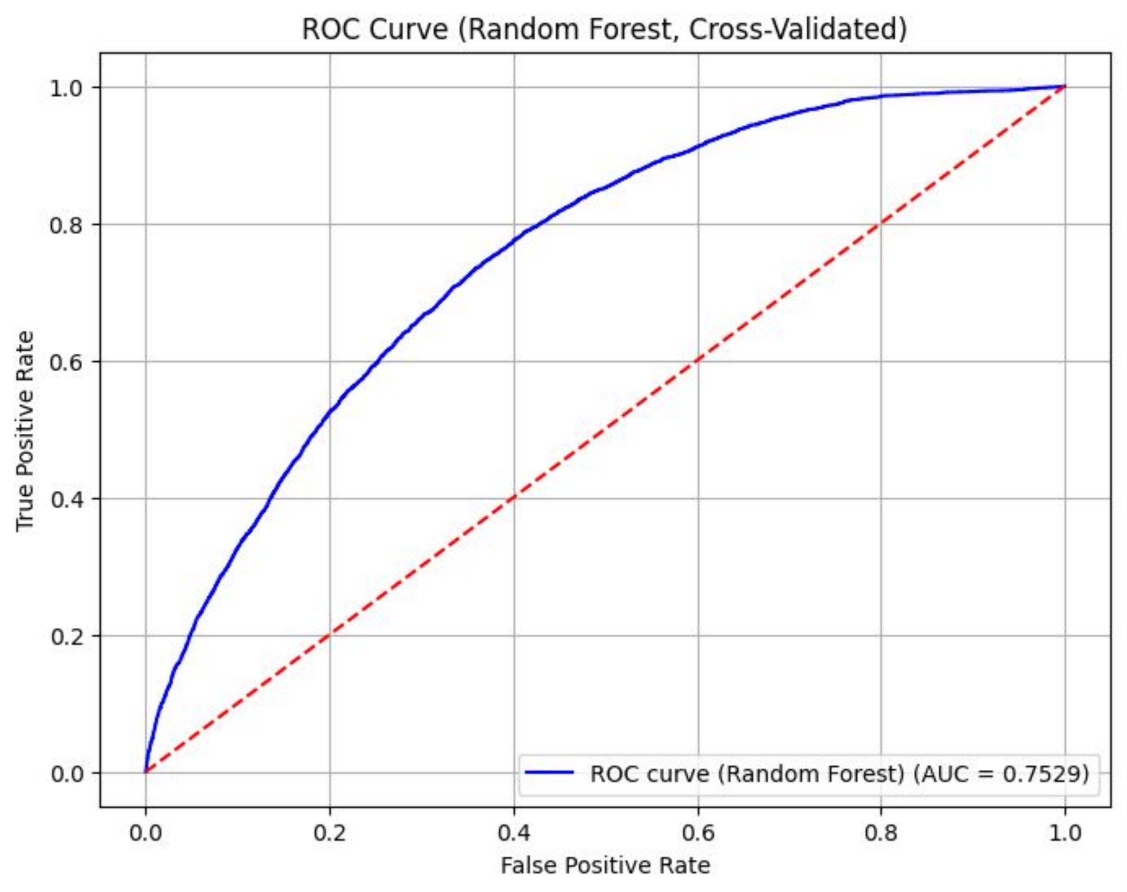
Extreme Gradient Boosting (XGBoost)



Best Parameters:

- gamma = 0.1
- learning_rate = 0.05
- max_depth = 5
- n_estimators = 200

Random Forest Classifier



Best Parameters:

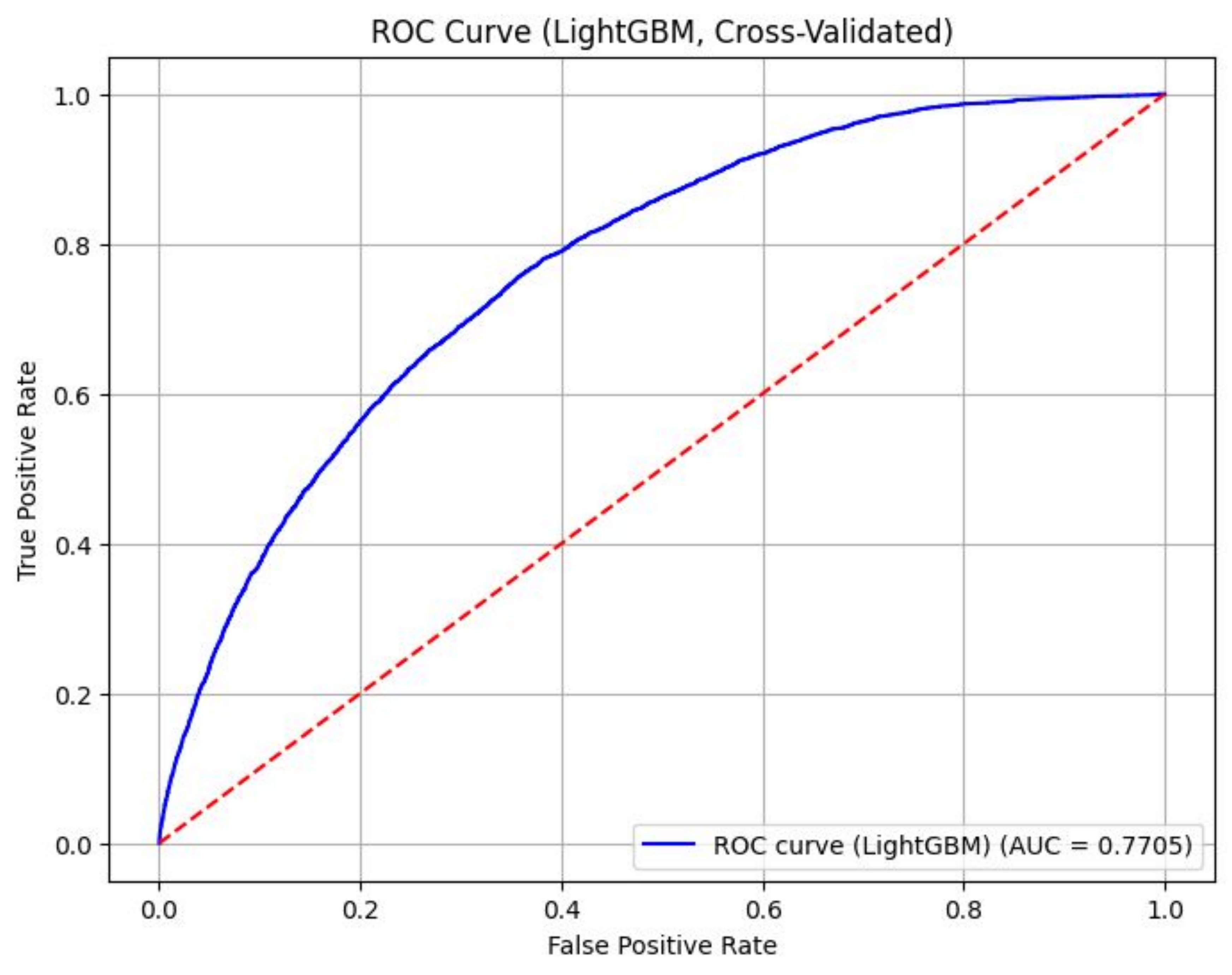
- bootstrap = False
- max_depth = 10
- min_samples_leaf = 2
- min_samples_split = 5
- n_estimators = 200

*all results are tuned

Task 2

Model Evaluation – Claim Status

Light Gradient Boosting Machine (LightGBM)



Best Parameters:
- learning_rate: 0.05
- max_depth: -1
- n_estimators: 100
- num_leaves: 31

Validation method:

For each model, we've used 5-fold cross-validation to find out the best parameters for each model

Evaluation method:

By comparing the ROC AUC score across each models' best tuned version, we see that LightGBM has the highest AUC (0.7705)

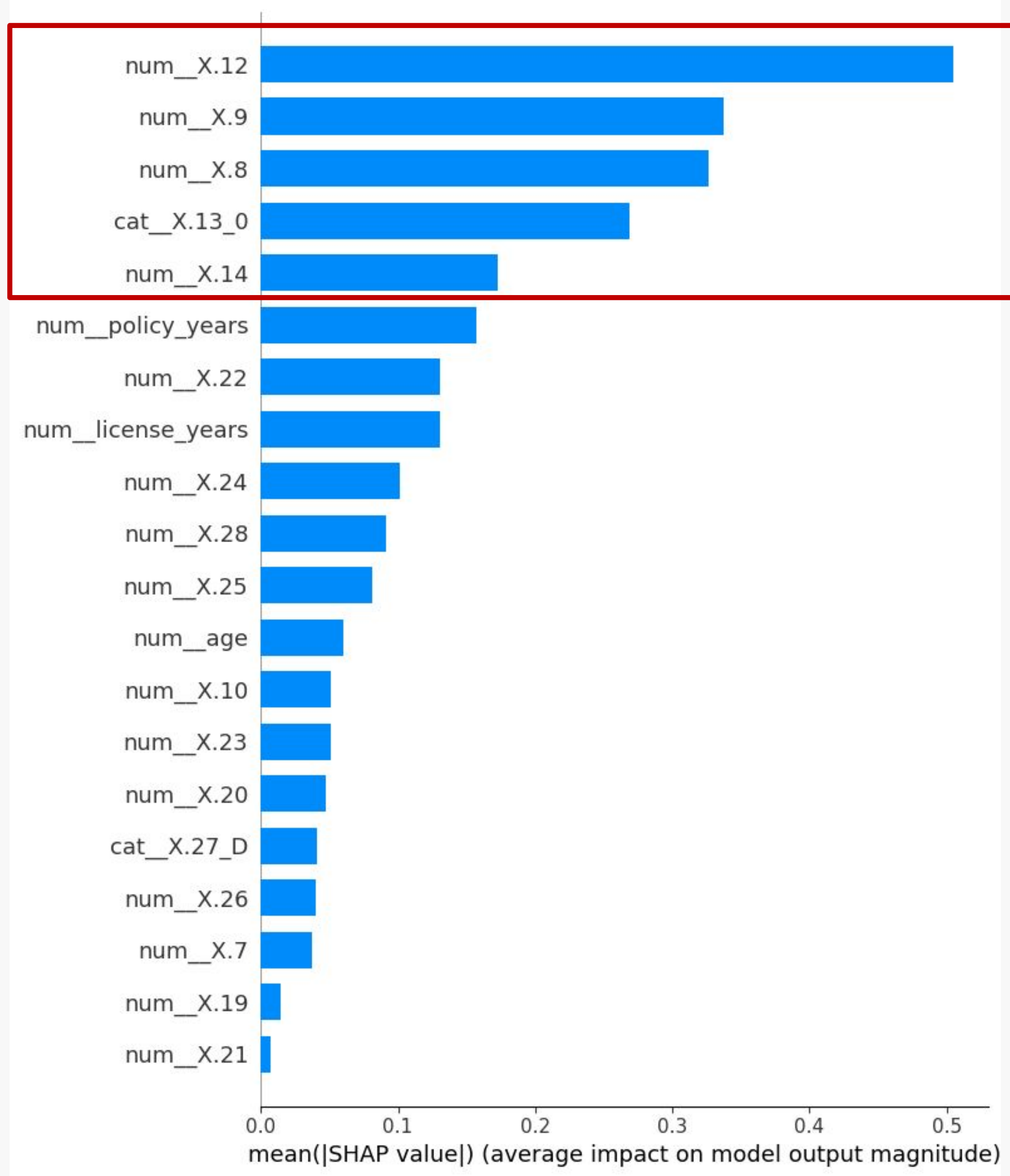
➔ ***LightGBM offers an excellent trade-off between training efficiency and prediction accuracy***

Task 2

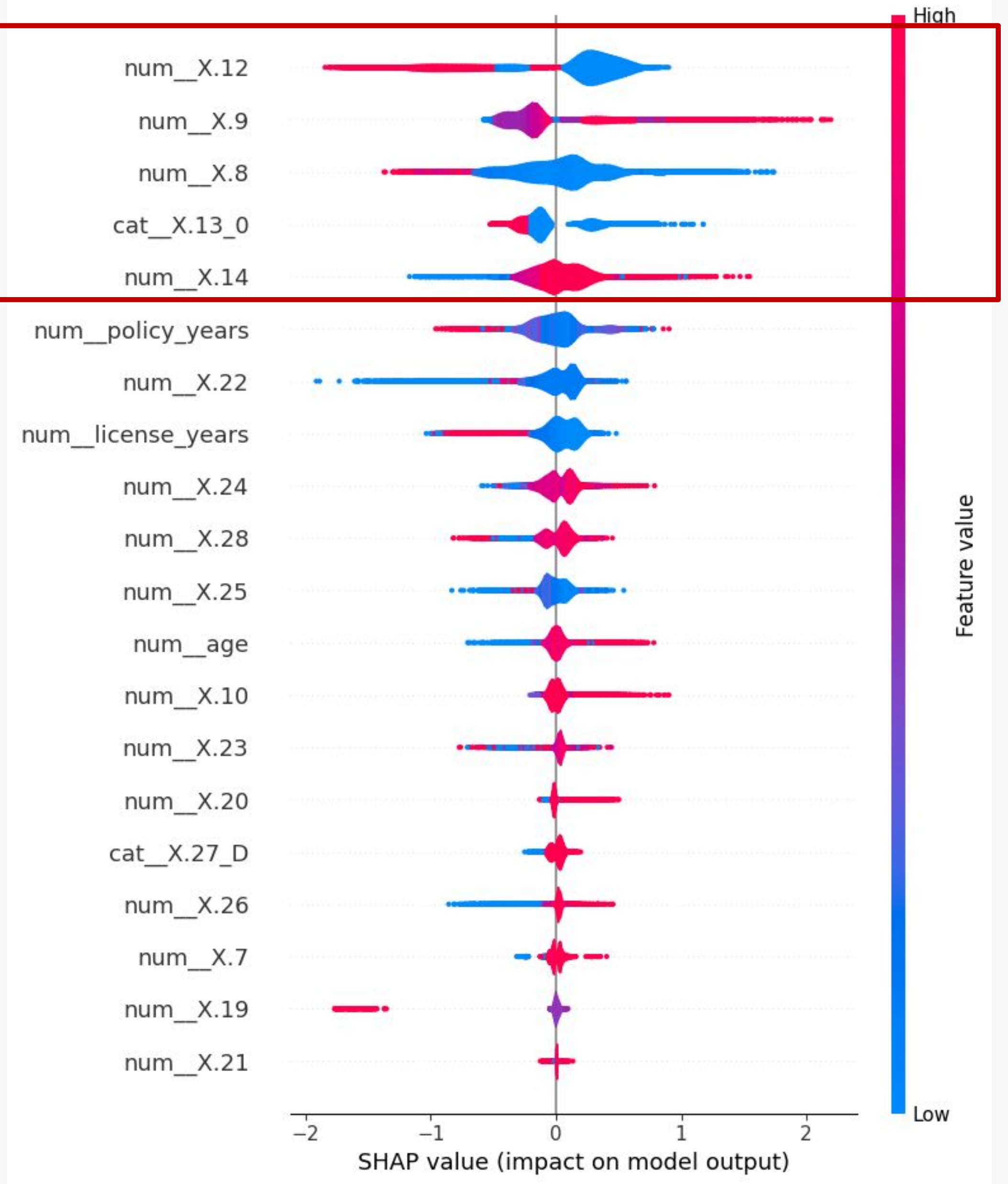
Model Interpretation – Claim Status

Magnitude & Direction of Feature Impact on Prediction

SHAP Value for CS (LightGBM)



Violin Chart for CS (LightGBM)



Top Features for Predicting Claim Status

X.12: # of policies canceled or terminated	Negative
X.9: # of policies held during the current year	Positive
X.8: Insurance years	Negative
X.13_0: Last payment method (half-yearly vs. annual)	Negative
X.14: Net premium amount	Positive

- The number of policies becomes 2nd important feature for predicting claim status

Innovation: Adversarial Validation

What is Adversarial Validation?

- A technique used to check **whether your training and test datasets come from the same distribution**
  - avoid overfitting

Steps:

1. Label training data as **0**, test data as **1**
2. Combine both datasets and shuffle
3. Train a **classifier** (like XGBoost or Logistic Regression) to predict the label

Why is it relevant?

- Issues that may lead to performance gap
  - data **leakage**
  - **distribution shift** between training and test sets
  - **non-representative** test data
- Adversarial Validation helps test this hypothesis

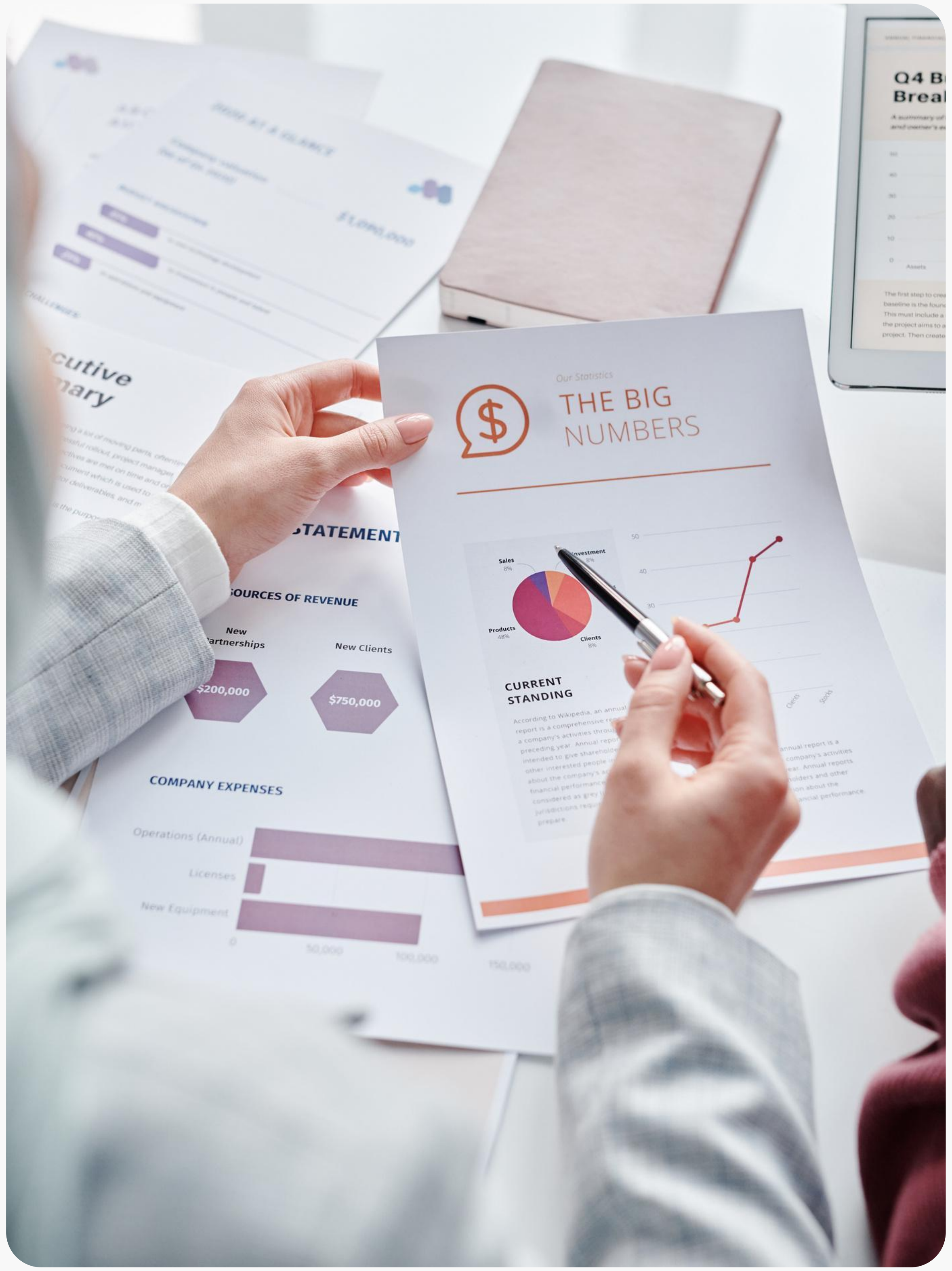
Our findings

Logistic Regression	LightGBM Classifier
AUC ≈ 0.51 → model could not distinguish between train and test sets	Train AUC = 0.78, but Validation AUC = 0.51 → model overfit, no real separation

Interpretation:

- **Train and test sets share similar distributions**
- **No strong evidence of data leakage or distribution shift**

Business Questions



Accuracy or Interpretation?

- **Prediction accuracy is slightly more important**, especially for predicting *Claim Status (CS)*
  - Accurate CS predictions help flag high-risk policyholders → Enables risk-adjusted pricing
- However, **interpretation is still critical** in regulated industries like insurance.
- **Balance is key.** We optimized accuracy **without sacrificing explainability**.

Using our model to directly predict loss cost for car insurance policies?

- **Caution is warranted.**
- **89%** of policyholders had **zero claims** → Highly imbalanced and skewed data.
- Tweedie regression helps but doesn't eliminate rare-event risk.

Thank you