Alfiyyah Ajeng Nurardita

Digital Skola

Car Insurance Prediction

CONTENT

Why Traditional Insurance Strategies Fail to Attract Good Customers?

Conventional insurance strategies rely on **broad group statistics**, for example assuming all young male drivers share the same risk profile.

This strategy limited because it **fails to assess risk on an individual level**, leading to unfair pricing. As a result, companies struggle to attract good customers and waste money on inefficient, untargeted marketing.

Given these limitations, there's a need for a smarter solution. Our solution uses a **predictive model** to analyze a wide range of customer data. This allows us to move beyond broad assumptions and **accurately predict which customers are likely to be interested, enabling a more precise, individualized approach.**

Methodology

Dataset

Using PredictionInsurance.csv to predict car insurance

Preprocessing

One Hot Encoding
Ordinal Encoding

Model Development

Decision Tree Classifier
Random Forest Classifier
XGBoost Classifier

Pipeline

Preprocessing - SMOTE - Model

Deployment

Using Streamlit

Attributes

Variable	Description
Gender	Customer gender (e.g., 'Male' or 'Female').
Age	Customer age in years.
Driving_License	Driving license ownership status (0 for do not have, 1 for have).
Previosly_Insured	Status whether the customer has had insurance before (0 for not having, 1 for having).
Vehicle_Age	The age of the vehicle owned by the customer (e.g., 'New', '1-2 Years', 'More than 2 Years').
Vehicle_Damage	Indication of whether the customer's vehicle has experienced previous damage (e.g., 'Yes' or 'No').
Response	Customer response to insurance offer (0 for not interested, 1 for interested).

Data Cleaning

Checking **Missing Values**

```
1 data.isna().sum()
```

	0
Gender	0
Age	0
Previously_Insured	0
Driving_License	0
Vehicle_Age	0
Vehicle_Damage	0
Response	0

dtype: int64

Checking **Data Types**

```
1 data.dtypes
```

	0
Gender	object
Age	int64
Previously_Insured	int64
Driving_License	int64
Vehicle_Age	object
Vehicle_Damage	object
Response	int64

dtype: object

Data Preprocessing

Ordinal Encoder for **Gender** and **Vehicle_Damage** Variable

```
#Ordinal Encoder
gender = OrdinalEncoder(categories=[['Male', 'Female']])
data['Gender'] = gender.fit_transform(data[['Gender']])
damage = OrdinalEncoder(categories=[['No', 'Yes']])
data['Vehicle_Damage_Encoded'] = damage.fit_transform(data[['Vehicle_Damage']])
```

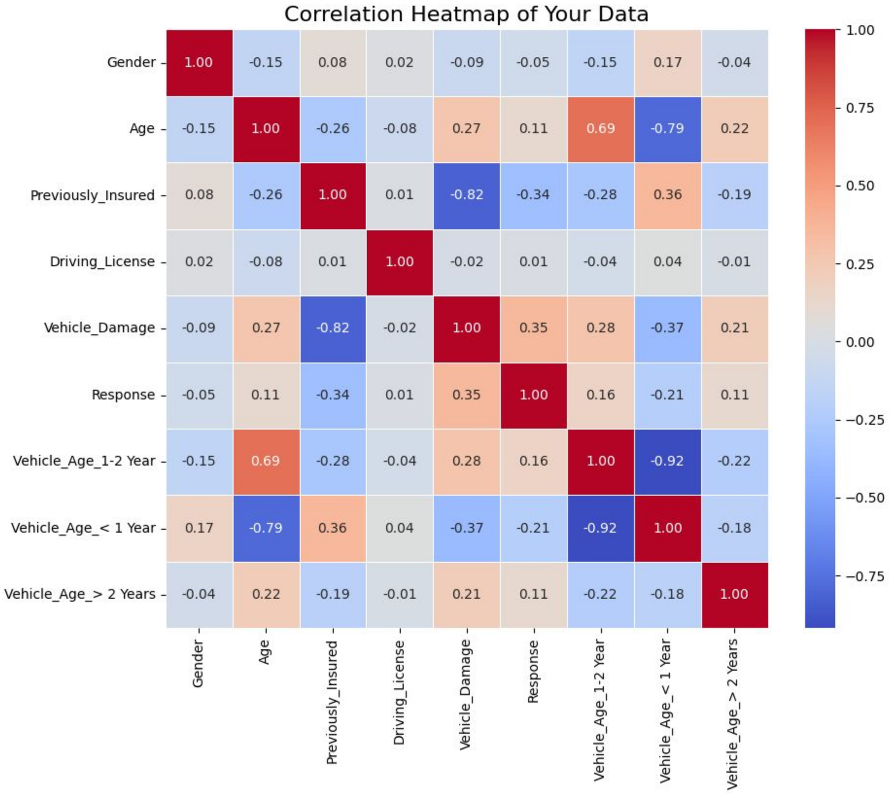
One Hot Encoding for **Vehicle_Age** Variable

```
#OHE
data = pd.get_dummies(data, columns=['Vehicle_Age'], dtype=int)
```

Implementing **SMOTE** for balancing dataset

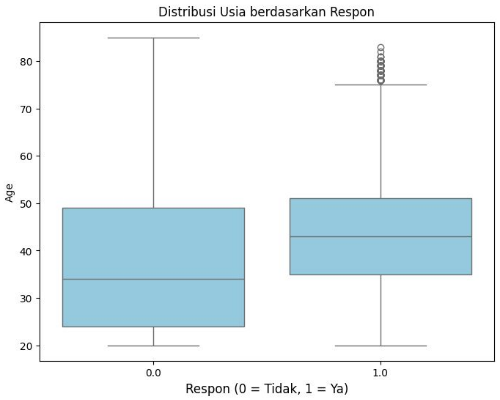
```
smote = SMOTE(sampling_strategy='minority')
X, y = smote.fit_resample(X, y)
y.value_counts()
```

Insight



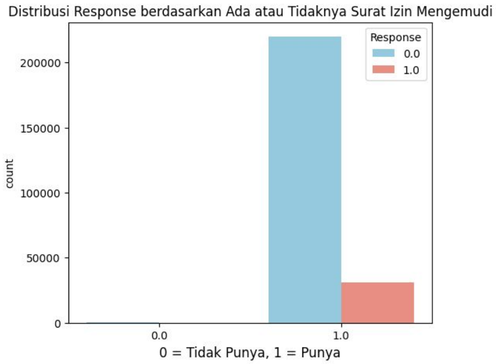
- Past **vehicle damage** drives **interest**
  Customers whose vehicles have experienced damage (Vehicle_Damage) show a stronger tendency to respond positively to insurance offers. This indicates that direct exposure to risk increases awareness of the need for protection.
- **Previously Insured** customers are **less** interested
  The variable Previously_Insured shows a negative correlation with customer response (Response). Customers who already hold insurance are generally less inclined to purchase an additional policy. This segment is less promising as a primary target.
- **Customer** age and **vehicle** age **move together**
  There is a strong correlation between Age and Vehicle_Age categories: older customers tend to own older vehicles. This relationship can help tailor products—for instance, offering specialized insurance for older vehicles.
- **Gender** has **little** impact
  Gender (Gender) has almost no correlation with Response. Therefore, gender-based segmentation is not a meaningful driver of purchase decisions in this context.

Insight



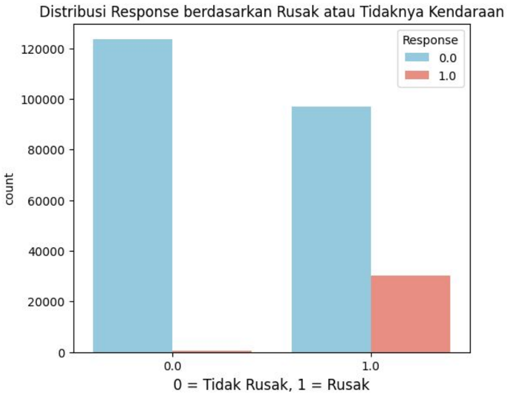
Older customers are more likely to show interest in insurance offers, making age an important factor for targeting.

The visualization shows that customers who responded positively (Response = 1) tend to be older, with a median age around the early 40s. In contrast, those who were not interested (Response = 0) are generally younger, with a median closer to the early 30s and a wider spread across age groups.



Driving license ownership is not a differentiating factor, since almost everyone has one.

The chart shows that nearly all customers have a driving license (1), while only a negligible number fall into the “no license” (0) category. Among licensed customers, the majority did not respond positively to the insurance offer (Response = 0), with only a smaller portion showing interest (Response = 1).



A vehicle's condition is a major factor in whether an insurance claim is filed.

The vast majority of non-damaged cars (0) do not result in a claim (response 0), with a minimal number of exceptions. In contrast, while many damaged cars (1) still have no claim, there is a substantial increase in the number of claims filed (response 1). This highlights that damage is a key driver for filing an insurance claim.

Model Development

- Data split with ratio 80:20
- The predictive models were developed using three different algorithms to ensure robust performance.

1. **Decision Tree** models decisions by creating a tree-like structure. It repeatedly splits the data into smaller groups based on the most informative features, forming branches. This process continues until a final prediction is reached at the "leaves" of the tree. It's an intuitive model that mimics human thought processes.
2. **Random Forest** is an ensemble model that builds many independent Decision Trees. It works by:

- Bootstrapping: Taking random subsets of the data.
- Random Feature Selection: Choosing a random subset of features for each tree.
- Aggregation: All trees make a prediction, and the final result is determined by combining their outcomes (e.g., majority vote for classification, average for regression). This method reduces overfitting and improves accuracy.

3. **XGBoost** is a powerful ensemble model that builds trees sequentially. Each new tree corrects the errors made by the previous ones. It works by:

- Iterative Correction: Starting with a simple prediction, it builds one tree at a time, with each new tree learning from the residual errors of the combined trees that came before it.
- Boosting: This process of learning from past mistakes is called boosting.
- Regularization: It includes built-in regularization to prevent overfitting. The final prediction is the sum of the predictions from all trees in the sequence.

- It’s also implementing **SMOTE** since the Response Variable has imbalanced values between 0 and 1. SMOTE works by creating new, synthetic data points for the minority class, rather than simply duplicating existing ones.

- To optimize the models, Hyperparameter Tuning also implemented using **Bayesian Optimization**, a sequential strategy for finding the best possible output of a function. Instead of trying every possible combination, BO uses a probabilistic model to intelligently guide its search.

Model Evaluation

The models are evaluated with confusion matrix and specially highlighting recall (true positive rate), a metric that measures models ability to find all the positive cases in dataset, for calculating

MODELS	ACCURACY	RECALL		PRECISION	
		0	1	0	1
Models with SMOTE	80	90	74	67	92
Decision Tree	82	90	30	90	31
Random Forest	87	88	40	98	7
XGBoost	87	88	45	99	3
Decision Tree - Tuned	87	88	45	100	0.1
Random Forest - Tuned	88	88	0	100	0
XGBoost - Tuned	86	89	35	96	17

* every algorithms for model with SMOTE dataset had a same accuracy either tuned or not

Classification Matrix

Baseline Model

classification report for Decition Tree

	precision	recall	f1-score	support
0	0.90	0.90	0.90	66360
1	0.31	0.30	0.30	9862
accuracy			0.82	76222
macro avg	0.60	0.60	0.60	76222
weighted avg	0.82	0.82	0.82	76222

classification report for Random Forest

	precision	recall	f1-score	support
0	0.98	0.88	0.93	74449
1	0.07	0.40	0.12	1773
accuracy			0.87	76222
macro avg	0.53	0.64	0.53	76222
weighted avg	0.96	0.87	0.91	76222

classification report for XGBoost

	precision	recall	f1-score	support
0	0.99	0.88	0.93	75594
1	0.03	0.45	0.06	628
accuracy			0.87	76222
macro avg	0.51	0.66	0.49	76222
weighted avg	0.99	0.87	0.93	76222

Models with SMOTE

	precision	recall	f1-score	support
0	0.67	0.90	0.77	49810
1	0.92	0.74	0.82	83950
accuracy			0.80	133760
macro avg	0.80	0.82	0.79	133760
weighted avg	0.83	0.80	0.80	133760

Tuned Model

classification report for Decision Tree

	precision	recall	f1-score	support
0	1.00	0.88	0.93	75911
1	0.01	0.45	0.03	311
accuracy			0.87	76222
macro avg	0.51	0.66	0.48	76222
weighted avg	0.99	0.87	0.93	76222

classification report for Random Forest

	precision	recall	f1-score	support
0	1.00	0.88	0.93	76222
1	0.00	0.00	0.00	0
accuracy			0.88	76222
macro avg	0.50	0.44	0.47	76222
weighted avg	1.00	0.88	0.93	76222

classification report for XGBoost

	precision	recall	f1-score	support
0	0.96	0.89	0.92	71606
1	0.17	0.35	0.23	4616
accuracy			0.86	76222
macro avg	0.56	0.62	0.58	76222
weighted avg	0.91	0.86	0.88	76222

Conclusion

From all models, it can be concluded that the model trained with **SMOTE-processed data** yields the **best performance** for both recall and precision. Thus, the pipeline used is:

Full Preprocessing + SMOTE +Modeling

In the context of car insurance prediction, the model's performance metrics indicate different implications for business risk management. While both classes are important, it is generally more critical to prioritize **Class 1** (claim customers) due to the **potential financial impact**.

Class 1 (Response: Yes)

- **Precision (0.92)**, among all customers predicted as claim, 92% were truly claim customers.
- **Recall (0.74)**, the model correctly identified 74% of actual claim customers.
- **F1-Score (0.82)**, strong overall balance, but recall is the weaker link compared to precision.

From these models, we tried to estimate the average financial loss and income that may occur due to model predictions.
Annual Premium : 30.564 USD
Average Claim assumption: 15.000 USD

	Pred 0	Pred 1
Actual 0	TN = 787	FP = 13
Actual 1	FN = 52	TP = 148

Estimated Loss

Estimated Loss = (FN x Average Claim) + (FP x Premi)
Estimated Loss = (52 x 15.000) + (13 x 30.564)
Estimated Loss = 1,18 million USD

Estimated Income

Estimated Income = (TP x TN) x Premi
Estimated Income = (148+787) x 30.564
Estimated Income = 28,55 million USD

Net Expected Profit

Net Expected Profit = Estimated Income - Estimated Loss
Net Expected Profit = 28,55 - 1,18
Net Expected Profit = 27,37 million USD

Thank You !

Explore my projects and insights, or let's connect to discuss data-driven solutions.

linkedin.com/in/ajeng-nurardita | *github.com/alfiyyahajeng*