Predicting Health Insurance Policyholders Interest in Vehicle Insurance

By Brandon Chuck, Yiming Huang, Yujia Zhang

Project Problem & Goals:

Problem:

★ Company wants to determine if current policyholders are interested in purchasing vehicle insurance.

Goals:

★ Test various classification models and find the best for predicting if current health insurance policyholders will buy vehicle insurance.
★ Predict the vehicle insurance Response based on the most impactful features.
★ Identify best predictors for a positive Response of 1.

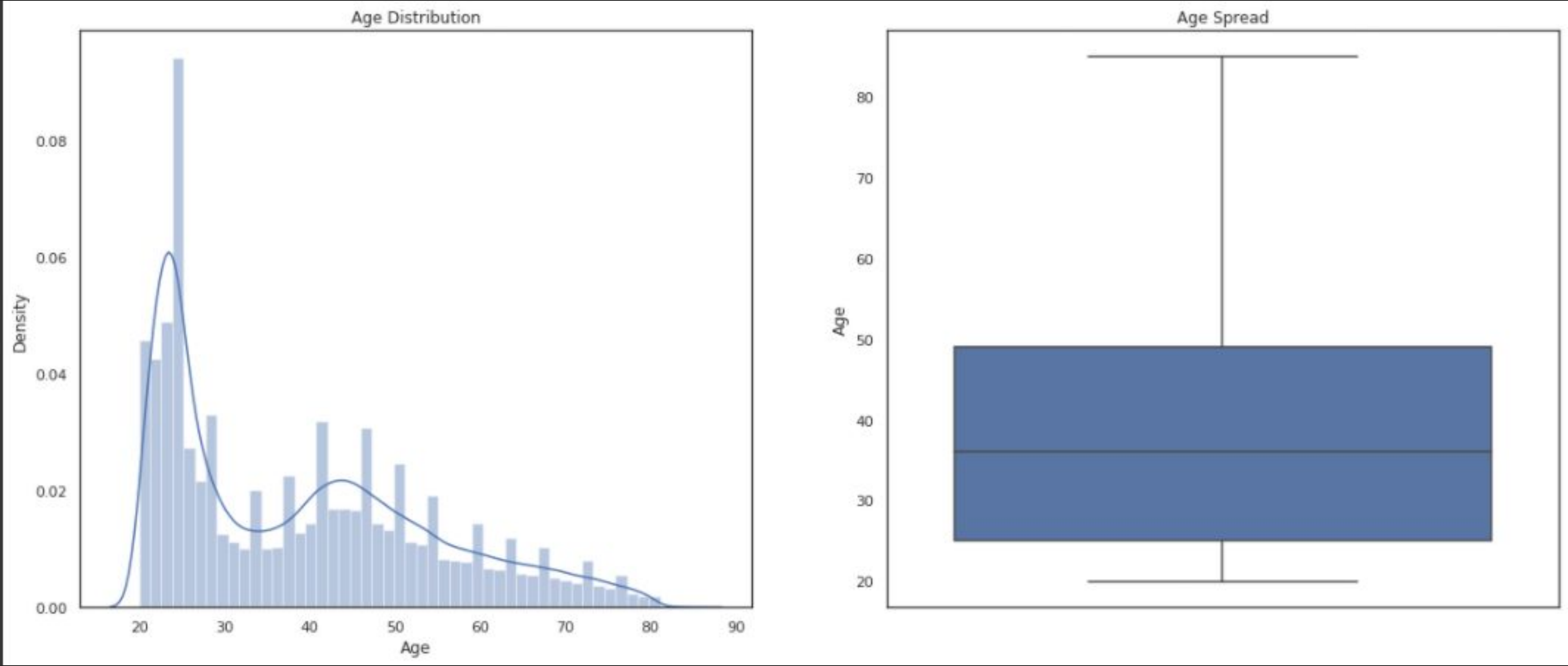
Browse Dataset: Existing Policyholder Features

- Continuous Features:
  - Annual_Premium
  - Vintage
  - Age
- Nominal Features:
  - Vehicle_Age (<1 yr, 1-2 yrs, >2 yrs)
  - Gender (Male or Female)
  - Previously_Insured (0 or 1)
  - Vehicle_Damage (Y or N)
  - Driving_License (Y or N)
  - Policy_Sales_Channel
  - Region_Code
- Predicting For:
  - Response (0 or 1)
    - 0 - Not Interested
    - 1 - Interested

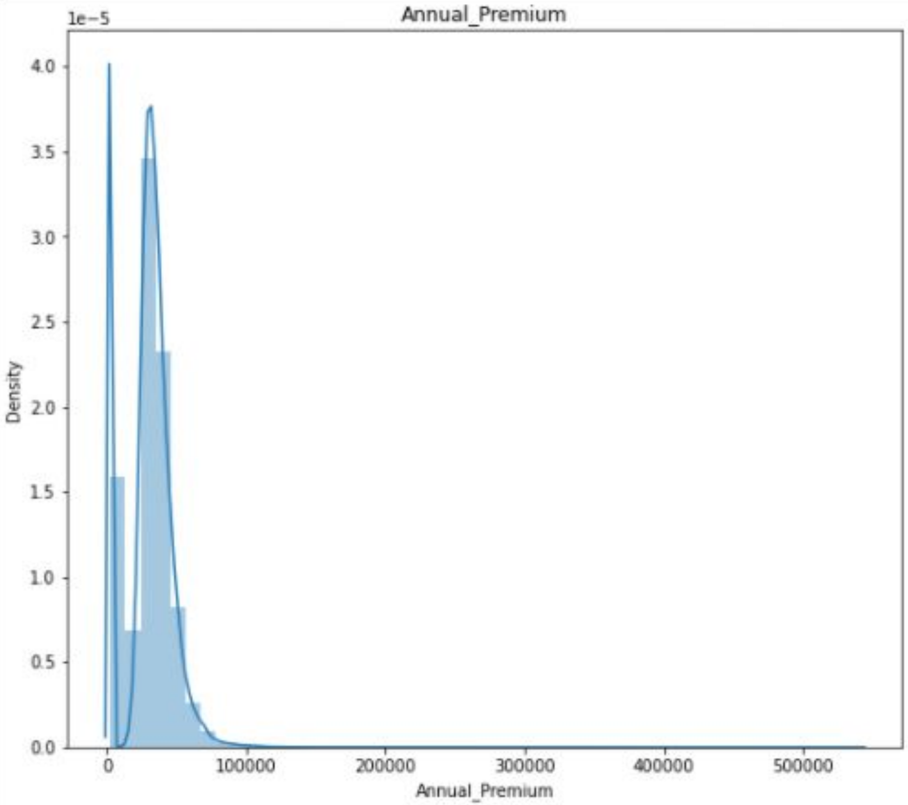
Feature Clarifications:

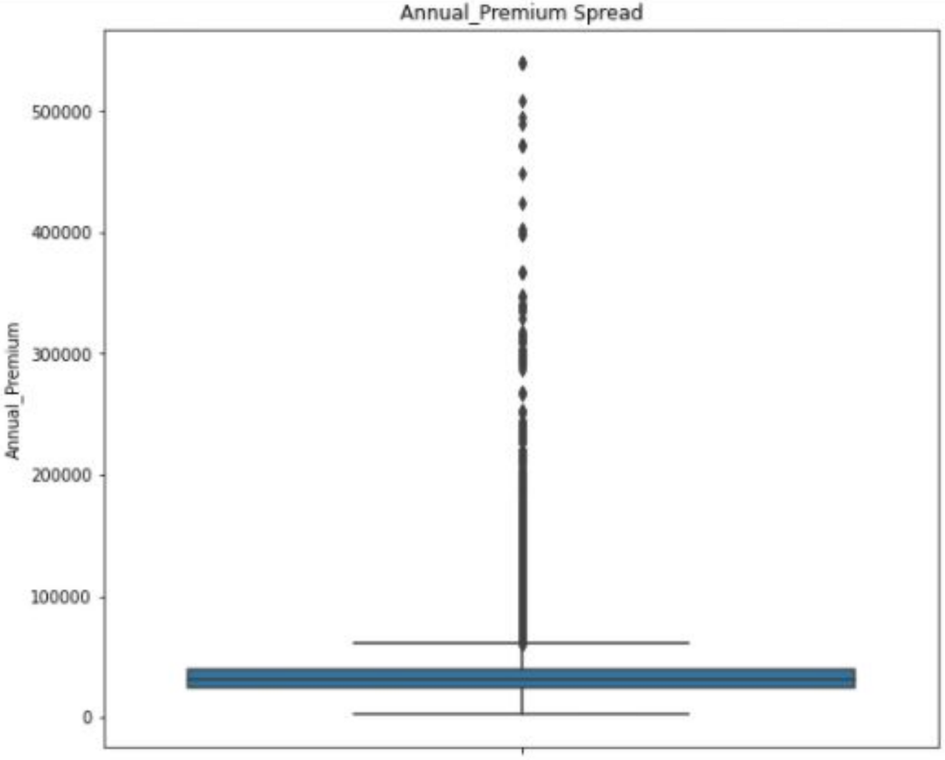
- Region_Code:
  - Unique code for the region of the customer
- Policy_Sales_Channel:
  - Anonymised code for the channel of outreaching to the customer ie. Different Agents, By-Mail, By-Phone, In-Person, etc.
- Vintage:
  - Number of days customer has been associated with the company

EDA Pt.1 - Age distribution



EDA Pt2 - Annual Premium



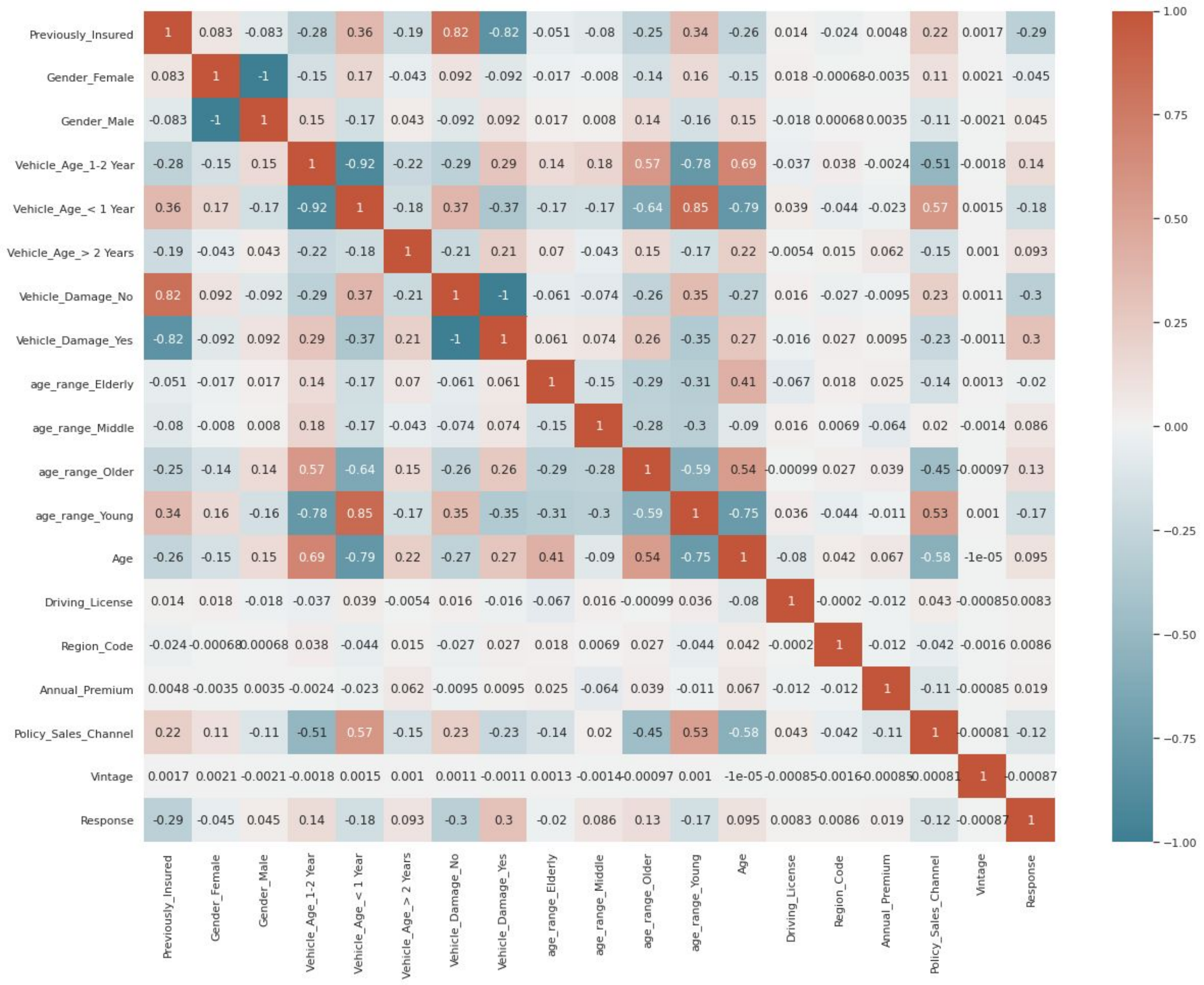


Correlation Matrix

Highly Correlated Features:

- Response:
  - Vehicle_Damage_Yes
  - Vehicle_Damage_No
  - Previously_Insured

- Previously_Insured:
  - Vehicle_Damage_Yes
  - Vehicle_Damage_No



Feature Engineering: Create New Features based on existing features

New Features:

1. ‘insured_with_no_damage’

```
df_numeric['insured_with_no_damage'] = df_numeric['Previously_Insured'] *df_numeric['Vehicle_Damage_No']
```

2. ‘not_insured_with_damage’

```
df_numeric['not_insured_with_damage'] = df_numeric['Previously_Insured'].apply(lambda x: 1 if x == 0 else 0) * df_numeric['Vehicle_Damage_Yes']
```

Preprocessing Steps:

1. Combine each dataset (train.csv, test.csv, sample_submission.csv) into single dataframe
2. Create a new feature ‘age_range’ w/ 4 categories: Young (20-30), Middle (30-40), Older (40-65), and Elderly (65-85).
3. Drop feature ‘id’ that have unique values, ‘Driving_License’, and ‘Vintage’ which seems to not affect the model.
4. One-hot encode categorical features: ‘Gender’, ‘Previously_Insured’, ‘Vehicle_Age’, ‘Vehicle_Damage’, ‘age_range’.
5. Apply RobustScaler() from Scikit-Learn in the pipeline models to Annual_Premium to handle outliers.
    - * Only apply StandardScaler() to ‘Age’ when training non-tree models.

Models:

- Baseline Used → Mode
  - Since we are dealing with a binary classification problem, we decided to compare our predictions with the mode.

- Pipeline Models
  - Logistic Regression + GridSearchCV
  - Decision Tree Classifier + GridSearchCV
  - Random Forest Classifier + GridSearchCV
  - KNN Classifier + GridSearchCV
  - XGBoost Classifier + GridSearchCV

Fitting Models with Raw Dataset

Baseline Accuracy= 0.91

Train.csv

test.csv

Sample_submission.csv

Raw dataset

Create age_range to young/middle/older/elder

Drop id, Driving_lRaw dataset icense, vintage

One hot encoding
'Gender', 'Previously_Insured', 'Vehicle_Age', 'Vehicle_Damage', 'age_range'.

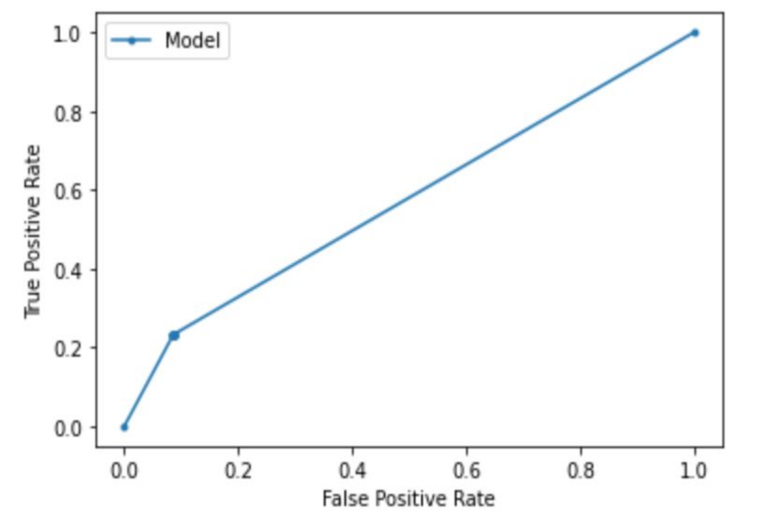
Create two new features:
Insured_with_no_damage = Previously_Insured * Vehicle_Damage_No
Not_insured_with_damage = Previously_Insured * Vehicle_Damage_Yes

For non_tree model:
StandardScaler() -> Age
RobustScaler() -> Annual Premium

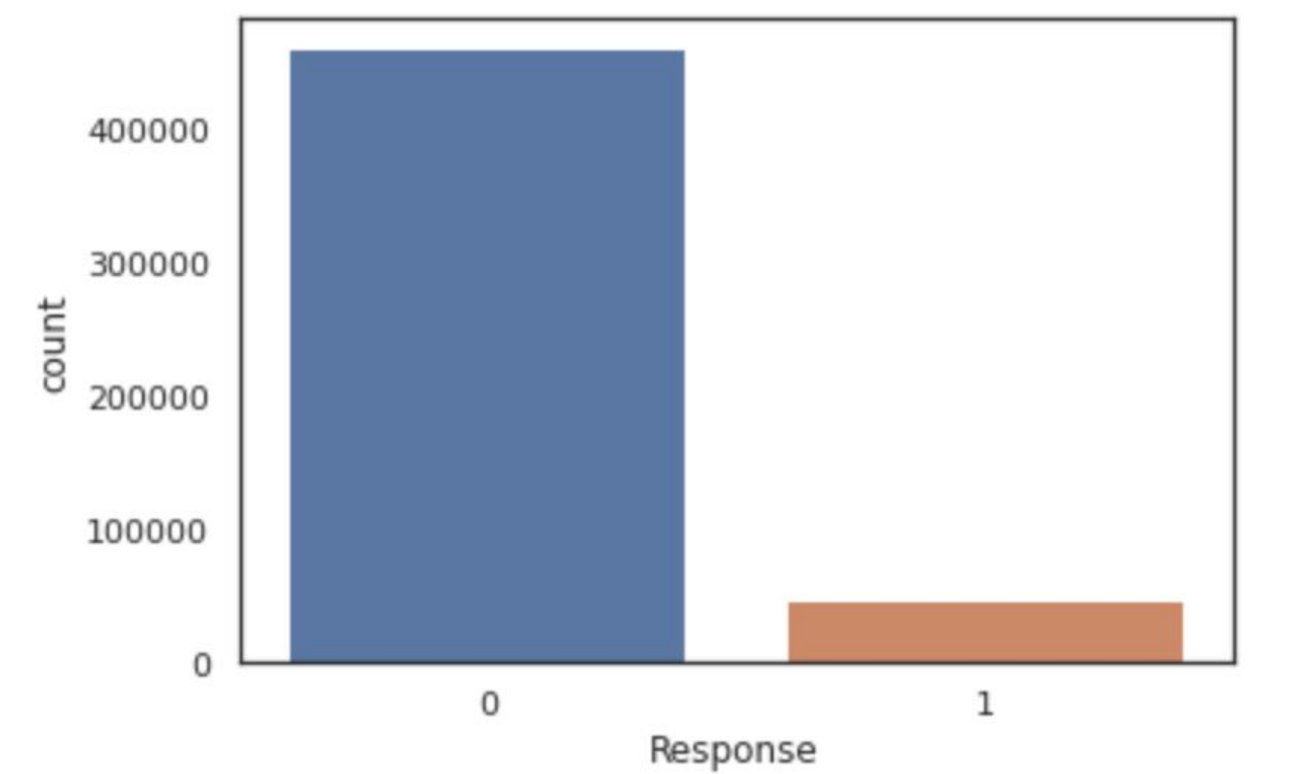
For tree model:

RobustScaler() -> Annual Premium

Models	F1_Train	F1_Val	ROC_train	ROC_val
Decision Tree_CV	1	0.85	1	0.573
Random Forest_CV	1	0.9	1	0.829
Logistic Regression_CV	0.91	0.91	0.827	0.824
KNN_CV	1	0.88	1	0.574



Problems with Dataset:



- Number of Responses were highly unbalanced which affected our model's predicting capabilities.
- Baseline mode accuracy rate is 0.9

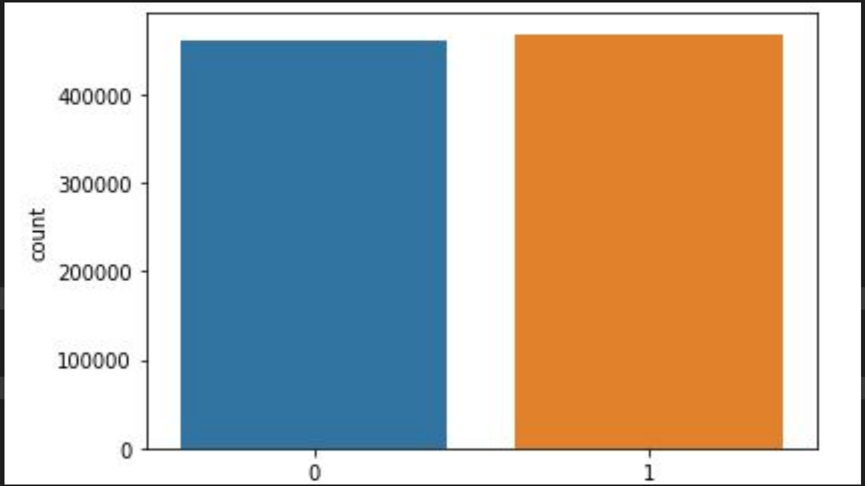
Balancing the Data

- To remedy this, we applied the ADASYN model which was designed to balance datasets being used for classification.

```
[ ] # Use ADASYN to balance the dataset
    from imblearn.over_sampling import ADASYN

    # create ADASYN model
    adasyn = ADASYN()

    # balance data
    df_X_adasyn, df_y_adasyn = adasyn.fit_resample(df_X, df_y)
```

```
[ ] new_df_y = pd.DataFrame(df_y_adasyn, columns=['Response'])
```

```
[ ] # Confirming that we have balanced Responses after ADASYN
    new_df_y_adasyn.value_counts()

    Response
    1          467827
    0          461436
    dtype: int64
```



Improved Results: After Applying ADASYN

Baseline Accuracy= 0.5

Train.csv

test.csv

Sample_submission.csv

Raw dataset

Create age_range to young/middle/older/elder

Drop id, Driving_lRaw dataset icense, vintage

One hot encoding
‘Gender’, ‘Previously_Insured’, ‘Vehicle_Age’, ‘Vehicle_Damage’, ‘age_range’.

Create two new features:
Insured_with_no_damage = Previously_Insured * Vehicle_Damage_No
Not_insured_with_damage = Previously_Insured * Vehicle_Damage_Yes

For non_tree model:
StandardScaler() -> Age
RobustScaler() -> Annual Premium

For tree model:
RobustScaler() -> Annual Premium

Applying ADASYN

Models	F1_Train	F1_Val	F1_Test	ROC_train	ROC_val	ROC_test
Decision Tree	0.93	0.91	0.91	0.988	0.961	0.959
Decision Tree_CV	0.98	0.92	0.92	0.999	0.938	0.935
Random Forest	0.99	0.93	0.93	0.999	0.98	0.98
Random Forest_CV	0.99	0.93	0.93	0.999	0.981	0.98
Logistic Regression_CV	0.78	0.78	0.78	0.829	0.83	0.828
KNN_CV	0.99	0.88	0.88	0.998	0.938	0.937
XGBoost	0.91	0.91	0.91	0.976	0.976	0.976
XGBoost_CV	0.92	0.92	0.92	0.98	0.979	0.979

Conclusion: Best Model -- Random Forest On Test Set

ROC AUC=0.980

F1 score metrics

	Precision	recall	f1-score	support
0	0.92	0.93	0.93	46165
1	0.93	0.92	0.93	46762
accuracy			0.93	92927
macro avg	0.93	0.93	0.93	92927
weighted avg	0.93	0.93	0.93	92927

Client base population
508,146

90% not interested in auto insurance
461,436

10% interested in auto insurance
46,710

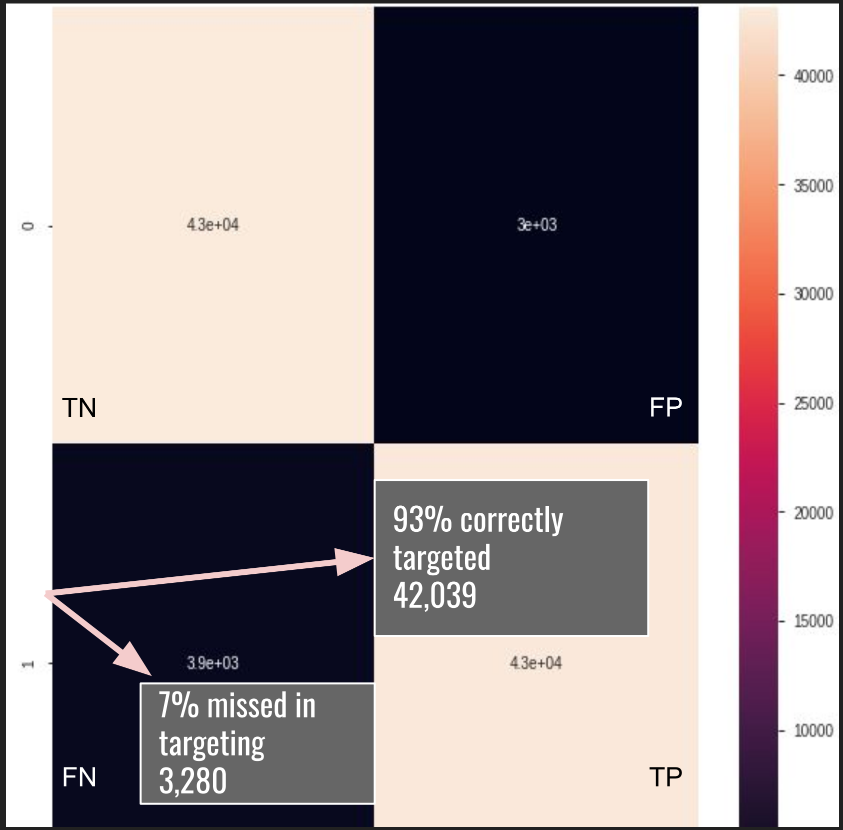
Confusion Matrix

Predicted not to buy

Predicted to buy

Actual Not Buying

Actual buying



Most Impactful Features:

Feature	Importance Score	Feature In-Detail
not_insured_with_damage	0.12845653	Previously insured but car damage
Region_Code	0.12077327	The region of the customer
Previously_Insured	0.10421228	Previously insured with auto insurance
age_range_Elderly	0.08769957	Age 65-85
Vehicle_Damage_Yes',	0.08241084	Car previously damaged
Annual_Premium	0.07346153	The amount customer needs to pay as premium in the year
Vehicle_Age_1-2 Year	0.05321889	Car 1-2 year
Policy_Sales_Chanel	0.04980615	Different Agents, Over Mail, Over Phone, In Person,

Thank you!