Auto Insurance Claim Data Prediction

Shaily Prajapati, Halim Dagher, Girik Nagpal, Brian Ortega, Amulya

MSBA Capstone QNT 795

Introduction

Insurance is a growing and vital industry that protects companies and individuals from economic uncertainty. “An insurance company offers risk management in the form of a contract” [1]. In other words, insurance companies protect companies and individuals from future potential losses by agreeing to pay any costs caused by an accident in exchange for premiums. A premium is the amount of money an insurer pays periodically to keep their insurance policy (contract) active. These premiums vary from person to person and are typically calculated by numerous variables that are stored in a model. These models are used to rate a policy based on its risk characteristics and are constantly changing and evolving as insurance companies try to leverage newer data. For insurance companies it is critical for them to adequately price out risks to cover their losses and generate revenue. Although all insurance companies offer to protect against future losses not all insurance companies offer the same products.

There are two types of insurance companies' life insurance companies and property and casualty insurance companies. Life insurance companies mainly issue policies that payout for death benefits, essentially when the insured person dies, the named beneficiaries will receive a sum of money. Property and casualty insurance companies protect personal or business against accidents. This includes loss coverage for things such as workers compensation, transportation, corporate buildings, personal home and auto etc. Although all risks are different the main goal of these insurance companies is to provide coverage when things don’t go right. In this project we will primarily focus on property and casualty insurance, particularly one major industry that insurance companies insure the auto industry.

Auto insurance protects an insurer from paying large amounts of cash if they are liable for an accident or an accident that occurs due to theft, a tree falling on your auto or weather-

related events. Auto coverage usually includes liability coverage, uninsured/underinsured, medical coverage, collision, comprehensive, etc. The auto insurance industry has faced some profitability challenges for numerous years and has seen improvements, but the latest trend sees loss ratios deteriorate in recent years [2]. Loss ratio are losses incurred (dollars) over premiums earned [3], losses is what an insurance company incurs from paying out claims. A claim is a request for coverage or payment from their insurance company due to an accident that has occurred. Claims without a doubt are the biggest cost component of an insurance company. After a claim is investigated, settled, and paid out it can account for 70% of premiums collected [4] if not more if a policy wasn't adequately priced for their risks. Essentially the more claims an insurance company has the more losses they incur and must pay out. One possible way insurance companies can help reduce costs is by leveraging predictive modeling to predict future claims. By leveraging this information, insurance companies can update their pricing via underwriting to ensure they cover future losses based on their portfolio (policy distribution), adjust their pricing in their models to improved pricing segmentation, or they can add rules and restrictions in their models to limit the number of policies that can be written that contain certain risky characteristics (known based on predictive models similar to the one in this article).

In this project we are going to try to build a model that leverages policy holder data and car characteristics data to predict whether a policyholder will file a claim or not within 6 months. By building such model a company can leverage this information to help mitigate risk by build strategy around what risk they are willing to accept as new business or automatically decline, increase underwriting rigor, and/or increase pricing to help them make the right investments to ensure their company remains profitable.

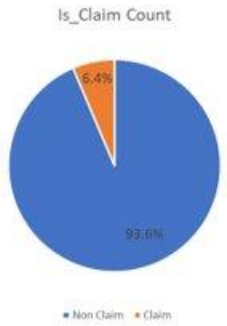
Exploratory Data Analysis

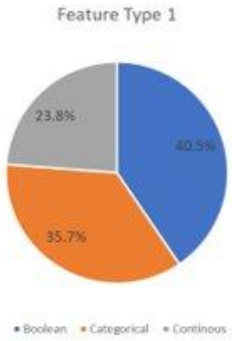
Data Sourcing

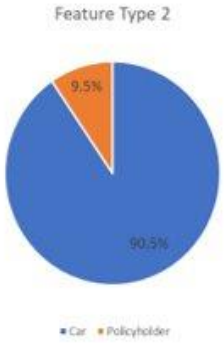
The data for this project was found in Kaggle with title “Car Insurance Claim Prediction” [7]. The data consisted of 44 columns and 58, 592 rows. Our target feature is called “is_claim” which is a Boolean that flags whether a particular policy had a claim or not.

Data Inspection & Cleaning

There were no duplications in the data and some of continuous variables were already normalized (age of policyholder, age of car, and policy tenure). Our target variable count was 54,844 (93.6%) non-claim and 3,748 (6.4%) claims which in the insurance world is typically the norm. However, when running our models, we noticed that our data was unbalanced due to the results and would have to balance out our data and rerun our model (more detail on this in predictive model section of paper). There are 3 types of features in our data. The first is Boolean feature which is 17 of 42 or 40.5%, then Categorical features 15 of 42 or 35.7%, lastly continuous features 10 of 42 or 23.8% of columns. For our Boolean features we change our values from yes/no to 1/0. For our categorical features we used the lift technique to change them to numeric, more to come on this technique later this paper. Finally, one other feature we shared is the break down between car features and policyholder feature. For background, in the insurance industry insurance companies leverage many variables to rate a given policy or risk. Most of these variables are related to the policyholder themselves such as where they live, credit score, motor history, age, tenure, etc. In this dataset we have more vehicle features 38 of 42 or 90.5% than we do policy holder feature 4 of 42 or 9.5%. This plays an interesting roll in predicting claims since most of features is information about characteristics about the car not the insured.







Data Dictionary

Boolean

Feature	Definition
Is_parking_camera	Boolean flag indicating whether parking camera is present or not
is_parking_sensors	Boolean flag indicating whether parking sensors are available or not
Is_brake_assist	Boolean flag indicating whether brake assistance feature is available or not
Is_central_locking	Boolean flag indicating whether the break assistance feature is available or not
Is_speed_alert	Boolean flag indicating whether the speed alert system is available in the car or not

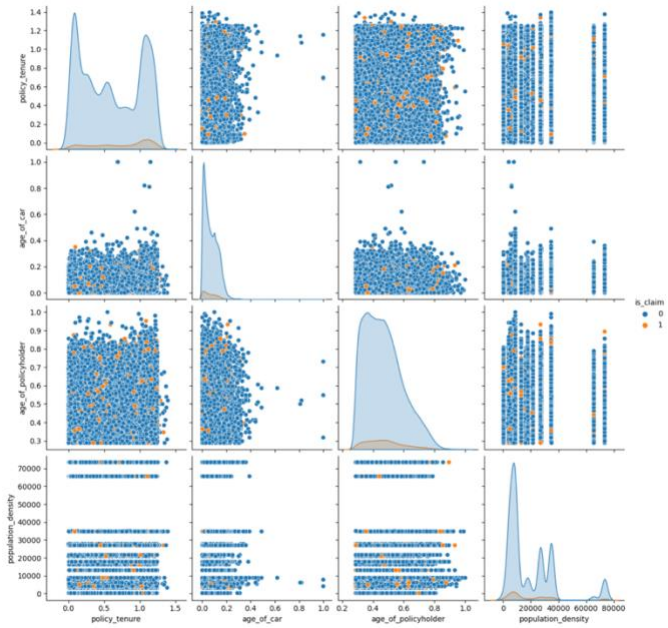
Categorical

Feature	Definition
Make	Make of the vehicle
Model	Model of vehicle
Max_power	Maximum power generated by car (bhp@rpm)
Engine_type	Type of engine used in the car
Ncap_rating	Safety rating given by NCAP (out of 5)

Continuous

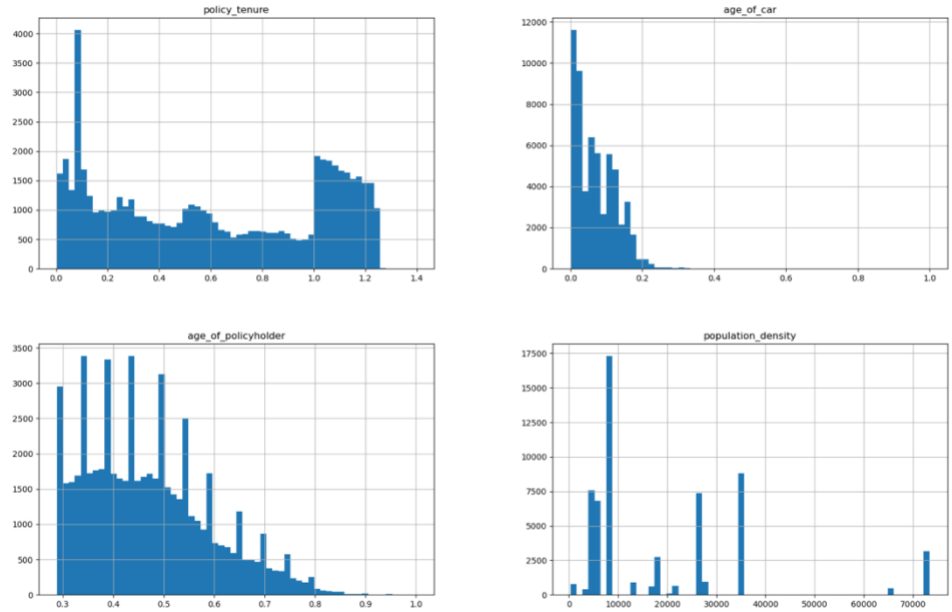
Feature	Definition
Policy_tenure	Time period of the policy
age_of_car	Normalized age of the cars in years
age_of_policyholder	Normalized age of policyholder in years
population_density	Population density of the city (policyholder city)
turning_radius	The space a vehicle needs to make a certain turn(meters)

Data Visualizations

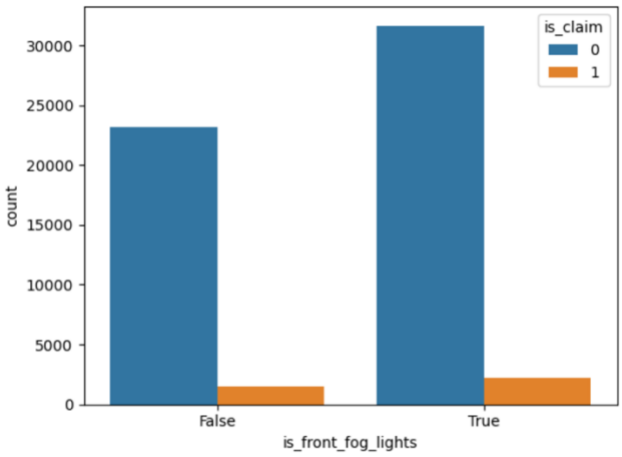


The pair plot helps understand the best set of features to explain the pairwise relationship between two sets of variables (regarding claim received- 0,1).

1. **Histogram (Matplotlib)**

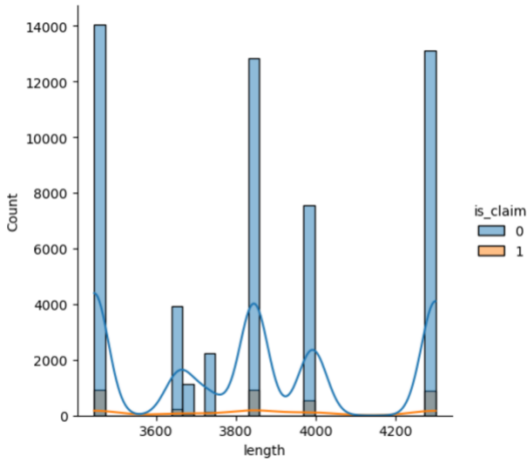


2. **Bar Graphs (Seaborn)**



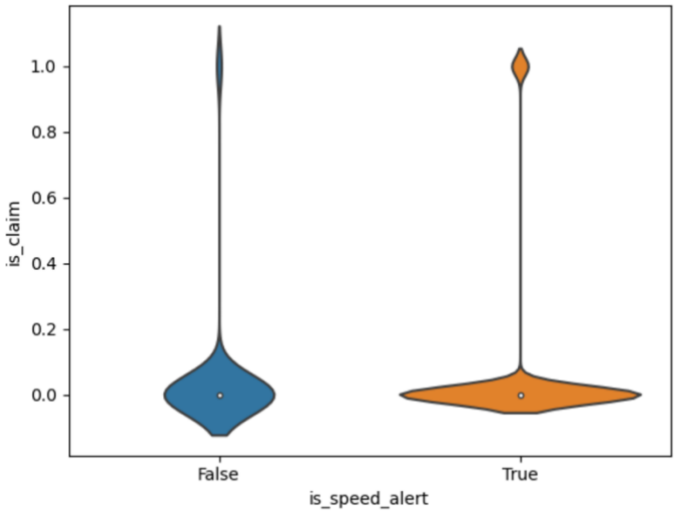
The categorical/ Boolean data was mostly represented by bar graphs. The above example shows that those cars with front fog lights installed, had less claims come in.

3. **Displot (Seaborn/ Matplotlib)**



The displot showed the variation in the data distribution.

4. **Violin Graphs (Seaborn)**



The violin plot, a mix of box plots and kernel density plot shows the peaks in our data using density curves. The width of the data represents the frequency of the data. In the above example of is_speed_alert, we can see that those cars which had speed alert systems in place, had less claims come in.

Feature Selection

The process of reducing the number of input variables when developing a predictive model is known as feature selection. Feature selection allows us to select a portion of relevant and/or important features from a large set of features in a dataset using certain techniques. It is preferable to reduce the number of input variables in order to reduce modelling computational costs and improve model performance. There are various methods that can be used in feature selection, filter-based feature selection methods score the correlation or dependence between input variables, which can then be filtered to select the most relevant features. Statistical measures for feature selection must be carefully selected based on the data types of the input and output variables. We used multiple methods outlined below for our feature selection analysis.

For the correlation method, we used the correlation coefficient to eliminate highly correlated features. For all other methods, we obtained the top 15 features that were deemed significant by the models.

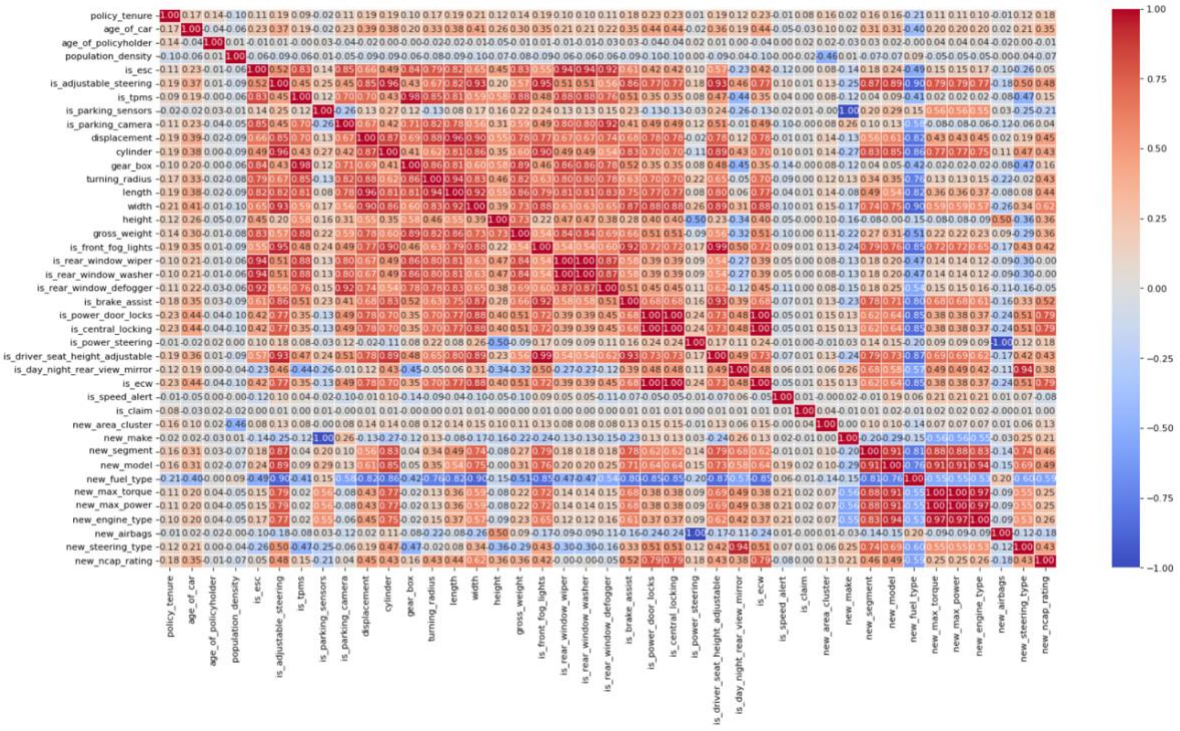
Correlation

Correlation measures the association or how closely related the values of two variables are. Its measures can help you understand the strength and direction, whether positive or negative, of a relationship between two variables. A positive correlation exists when the values of the two variables increase or decrease together, while a negative correlation occurs when one variable increase while the other decreases. Correlation can be measured by a correlation coefficient, which ranges from -1 to 1, with values closer to 1 indicating a stronger positive correlation, values closer to 0 indicating no relationship, and values being closer to -1 indicating strong negative correlation. The use of correlation can help identify patterns and trends that can be useful for analysis and feature selection. For example, based on the Pearson's correlation, we found the max torque, max power and engine type were highly correlated. We see the correlation between max torque and max power is 1 while the correlation between engine type and both max power and max torque was 0.97. According to the finding in our analysis we decided to keep engine type and drop both max torque and max power since they were both perfectly correlated, and highlight correlated to engine type. Based on just correlation alone, we eliminated 14 features.

Features Eliminated:

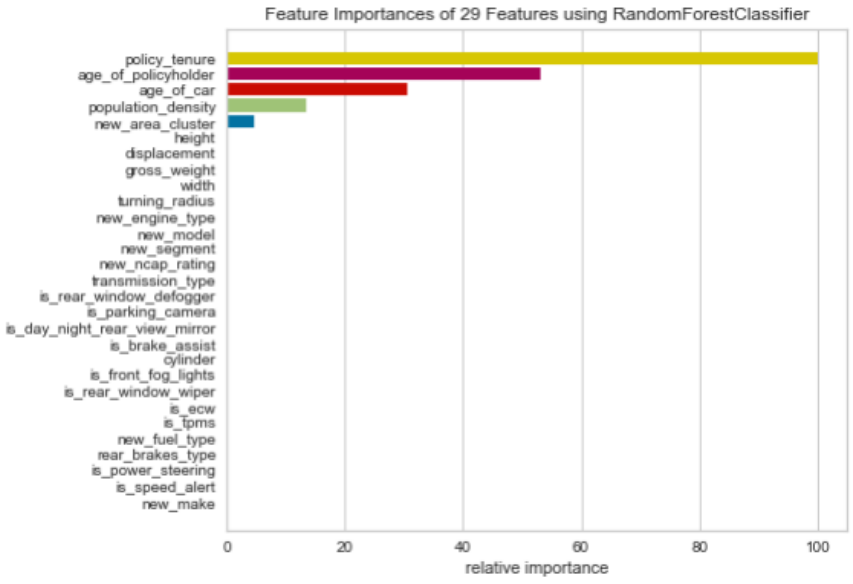
1. Deleting policy_id as it does not help us with data analysis
2. Deleting is_rear_window_washer as it is perfectly correlated with is_rear_window_wiper; wiper is more frequently used in cars than washer
3. Deleting is_power_door_locks as it is perfectly correlated with is_central_locking, they are essentially the same in definition

4. Deleting is_parking_sensors as it is perfectly correlated with new_make, the make would include the parking sensor aspect in the make type. Each make would have the same type of parking sensor
5. Deleting is_max_torque as it is perfectly correlated with is_max_power
6. Deleting is_airbags as it is perfectly correlated with is_power_steering, power steering is more relevant to claims than airbags in respect to the 'car' itself
7. Deleting is_central_locking as it is perfectly correlated with is_ecw, keeping ecw as it is more relevant in terms of target variable of claims. Hence, looking back to bullet number 2, we are essentially choosing ecw over two variables- power door locks and central locking
8. Deleting is_driver_seat_height_adjustable and keeping is_front_fog_lights based on sum comparison
9. Deleting gear_box and keeping is_tpms based on sum comparison
10. Deleting new_max_power and keeping new_engine_type based on sum comparison
11. Deleting length and keeping displacement based on sum comparison
12. Deleting is_adjustable_steering and keeping cylinder based on sum comparison
13. Deleting new_steering_type and keeping day_night_rear_view_mirror based on sum comparison
14. Deleting is_esc and keeping and keeping is_rear_window_wiper based on sum comparison



Random Forest

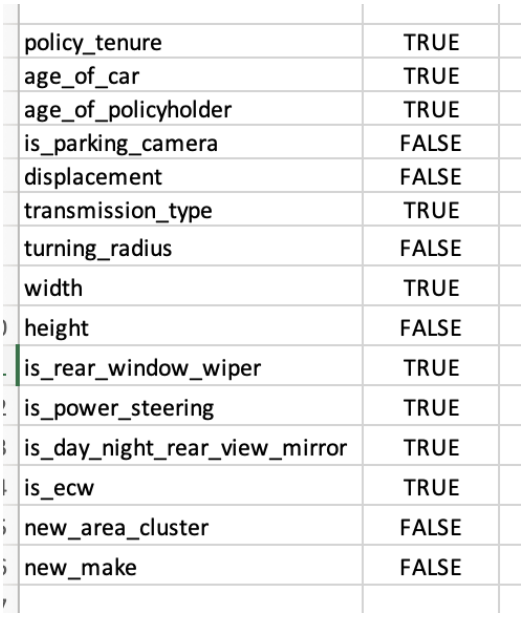
One of the most popular machine learning algorithms, Random Forest is a great tool for selecting the most important features. The most important features can be visualized and used to choose the collection of features that is the most informative. When used correctly, feature significance can provide us with excellent deliverables (the bar plot) and effective optimization (feature selection). The feature importance can be measured as the average impurity decrease computed from all decision trees in the forest. A random extraction of the observations from the dataset and a random extraction of the features are used to build each of the hundreds of decision trees that make up a random forest. Because some trees don't see all the features or all the data, the trees are guaranteed to be de-correlated and therefore less likely to overfit. We saw 5 out of 29 features showed great significance using Random Forest.



Lasso

Lasso is a machine learning technique frequently used for regression tasks. It's a useful technique that can help recognize the most significant features in the dataset, and discard the less important ones, leading to a more explainable and accurate model. Lasso works by constraining the coefficients of variables through regularization, which allows the algorithm to learn the underlying relationships between variables and the target variable in a more effective way. Lasso is also known for its ability to handle multicollinearity, a situation where some features are highly correlated with each other. By shrinking the coefficients of the redundant features to zero, lasso can effectively select one representative feature instead of many highly correlated ones. Lasso remains a widely used technique for regression tasks and feature selection.

Output.



policy_tenure	TRUE
age_of_car	TRUE
age_of_policyholder	TRUE
is_parking_camera	FALSE
displacement	FALSE
transmission_type	TRUE
turning_radius	FALSE
width	TRUE
height	FALSE
is_rear_window_wiper	TRUE
is_power_steering	TRUE
is_day_night_rear_view_mirror	TRUE
is_ecw	TRUE
new_area_cluster	FALSE
new_make	FALSE

Sequential forward selection (SFS) and backward selection (SBS)

SFS and SBS are used in machine learning to select the most relevant features/variables in the dataset that is being studied. In SFS the algorithm selects one feature at a time and adds them to an empty set which makes the model efficient. While SBS starts with a full set and removes the worst features one at a time until the determined number of variables is reached.

SFS wd 10 fw	SFS wd 10 bk	SBS wd 10 fw	SBS wd 10 bk
policy_tenure	policy_tenure	policy_tenure	policy_tenure
age_of_car	age_of_car	age_of_car	age_of_car
population_density	population_density	population_density	population_density
is_tpms	is_tpms	is_tpms	is_tpms
is_parking_camera	is_parking_camera	is_parking_camera	is_parking_camera
rear_brakes_type	rear_brakes_type	rear_brakes_type	rear_brakes_type
displacement	displacement	displacement	displacement
cylinder	cylinder	cylinder	cylinder
transmission_type	transmission_type	transmission_type	transmission_type
turning_radius	turning_radius	turning_radius	turning_radius
width	width	width	width
height	height	height	height
gross_weight	gross_weight	gross_weight	gross_weight
is_front_fog_lights	is_front_fog_lights	is_front_fog_lights	is_front_fog_lights
is_rear_window_wiper	is_rear_window_wiper	is_rear_window_wiper	is_rear_window_wiper

Mutual Information

Mutual Information is a statistical method used to measure the dependence between two variables. It provides a measure of how much information one variable provides about the other variable. In the context of the above dataset, mutual information can help to determine which features have a strong association with the target variable, and which features are less informative.

To calculate mutual information, we need to consider the joint distribution of the target variable and each feature. We can then compare this to the expected joint distribution if the variables were independent. The difference between the two distributions gives us the mutual information between the target variable and the feature.

If we assume that the target variable is some measure of the car's safety or performance, we could use mutual information to determine which features are most informative for predicting this measure. For example, we could calculate the mutual information between the target variable and each feature to determine which features are most strongly associated with car safety or performance.

Overall, mutual information can be a useful tool for feature selection and understanding the relationship between variables. However, it requires careful consideration of the target variable and the joint distribution of the features, and it is important to ensure that the results are interpreted correctly.

Output.

1		
2	policy_tenure	0.0030
3	is_speed_alert	0.01124
4	rear_brakes_type	0.011199
5	is_power_steering	0.010746
6	is_front_fog_lights	0.009916
7	is_brake_assist	0.009252
8	is_ecw	0.008377
9	cylinder	0.00818
10	new_segment	0.004924
11	is_parking_camera	0.004517
12	new_fuel_type	0.00437
13	is_rear_window_defogger	0.004052
14	new_area_cluster	0.003889
15	transmission_type	0.003663
16	is_day_night_rear_view_mirror	0.003581
17	new_engine_type	0.003541
18	gross_weight	0.002691
19	new_ncap_rating	0.002616
20	is_tpms	0.002561
21	new_model	0.002306
22	is_rear_window_wiper	0.002267
23	turning_radius	0.002263
24	age_of_car	0.001835
25	width	0.001721
26	population_density	0.001644
27	displacement	0.001407
28	height	0.001329
29	age_of_policyholder	0.00117
30	new_make	0.000822
31		

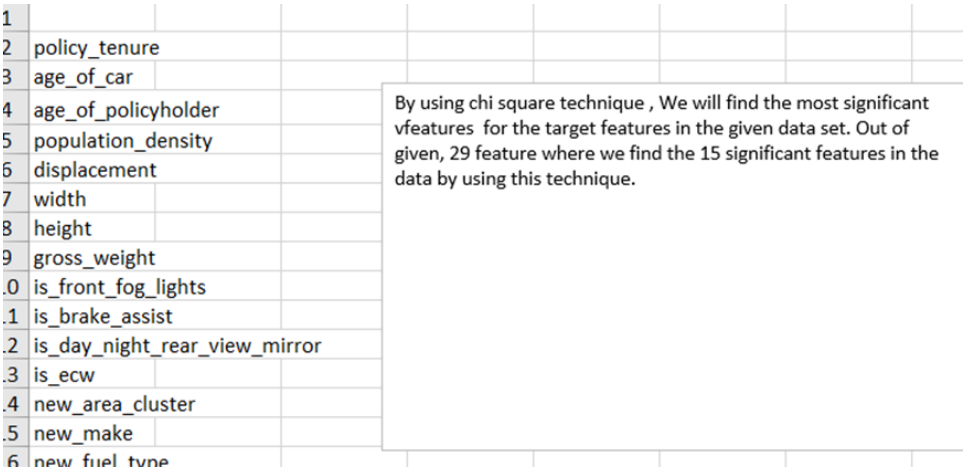
Chi-Square

Chi-square is a statistical test used to determine if there is a significant association between two categorical variables. Chi-square is typically used when we have two categorical variables, and we want to determine if there is a significant association between them. For example, if we had data on car brand and car color, we could use chi-square to determine if there is a significant association between the two variables, i.e., if certain car brands tend to be of certain colors. The given dataset appears to be a list of numerical values for various features of a car.

For our feature selection process, we used multiple methods/techniques. Based on our analysis and output of these techniques, we cut down on the number of features which we'll use to run our models. In the beginning of our analysis, we leveraged the correlation heat map to

identify features that were either perfectly positively or negatively correlated (i.e had a value of -1 or 1) or were strongly positively or negatively correlated. These features were eliminated from our dataset. To continue with our feature selection analysis, we leveraged other techniques such as Random Forest, Lasso, and Sequential Feature Selection to eliminate more features from our dataset. Although we leveraged multiple techniques most of the weight was put on Random Forest and Lasso techniques, with some consideration to Sequential Feature selection.

Output.



Features Selected

Based on just correlation alone, we eliminated 14 features. In order to eliminate more features, we found the most significant features based on Random Forest, Lasso, Mutual Information, SFS, SBS and Chi Sqaure methods. Out of the significant features, we considered the top 15 and compared them to the other methods. As shown in the screenshot below, 1 determines if the method selected the feature in its top 15 features and 0 determines if the method did not select the feature as its top 15 significant features. We can find average of the total (based on 1 and 0.) Based on these average values, we decided to keep the features in dark

yellow. Additionally, since both Random Forest and Lasso selected the features highlighted in light yellow, we decided to include these as well.

	Random Forest	Ranking	Lasso	Fw(5)	Fw(10)	Bk (5)	Bk (10)	Total	Chi Square	Mutual Info
is_parking_camera	0.077615	15	1	1	1	1	1	3	0	0
transmission_type	0.068958	16	1	1	1	1	1	3	0	0
policy_tenure	100	1	1	1	1	1	0	2.5	1	1
new_area_cluster	4.362871	5	1	1	0	1	1	2.5	1	1
is_day_night_rear_view_mirror	0.06078	19	1	1	0	1	1	2.5	1	1
is_rear_window_wiper	0.05107	22	1	1	1	0	1	2.5	0	0
age_of_policyholder	52.969807	2	1	0	1	0	1	2	1	1
turning_radius	0.183629	10	1	0	1	0	1	2	0	0
is_rear_window_defogger	0.067399	17	0	1	1	1	1	2	0	0
cylinder	0.064635	18	0	1	1	1	1	2	0	0
rear_brakes_type	0.026971	26	0	1	1	1	1	2	0	0
new_make	0.018916	27	1	1	1	0	0	2	1	1
is_power_steering	0.018058	28	1	0	0	1	1	2	0	0
age_of_car	30.805515	3	1	0	1	0	0	1.5	1	1
displacement	0.204827	8	1	0	1	0	0	1.5	1	1
is_brake_assist	0.059964	20	0	1	0	1	1	1.5	1	1
is_ecw	0.053597	21	1	0	0	1	0	1.5	1	1
is_front_fog_lights	0.0504	23	0	1	0	1	1	1.5	1	1
is_tpms	0.03097	25	0	1	1	0	1	1.5	0	0
is_speed_alert	0.011783	29	0	1	1	1	0	1.5	0	0
population_density	13.219836	4	0	1	1	0	0	1	1	1
height	0.274621	6	1	0	0	0	0	1	1	1
width	0.224655	7	1	0	0	0	0	1	1	1
gross_weight	0.198296	9	0	0	0	0	1	0.5	1	1
new_segment	0.115881	14	0	0	0	0	1	0	0.5	0	0
new_fuel_type	0.037286	24	0	0	0	0	1	0	0.5	1	1
new_engine_type	0.162545	11	0	0	0	0	0	0	0	0	0
new_model	0.157746	12	0	0	0	0	0	0	0	0	0
new_ncap_rating	0.120154	13	0	0	0	0	0	0	0	0	0

Based on the outputs from all methods other than correlation, we chose to eliminate the following features:

1. Is_front_fog_lights and is_driver_seat_height_adjustable
2. is_tpms and gear_box
3. new_max_power and new_engine_type
4. displacement and length
5. is_adjustable_steering and cylinder
6. is_day_night_rear_view_mirror and new_steering_type
7. is_esc and is_rear_window_wiper

Prediction Models

Machine learning is a subset of artificial intelligence that allows computer systems to learn and improve from experience without being explicitly programmed. It involves the use of algorithms and statistical models to analyze and make predictions or decisions based on data. Machine learning can be applied to various fields, including car insurance claim prediction. In this context, machine learning algorithms can be trained on historical data of car insurance

claims, along with various other features such as driver age, vehicle make and model, location, etc. The algorithm can then use this training data to identify patterns and make predictions on future claims, such as the likelihood of a claim being filed and the expected cost of the claim. This can help insurance companies assess risk, set premiums, and make more informed decisions about claims.

Lift technique was performed to bin the data into groups thus transforming our categorical variables into continuous features. Binning the variables allowed us to improve the performance of the model that we wanted to create, and it allowed us to identify any missing values or outliers.

Following are the predictive models we ran to see which one fits our data the best: Logistic Regression, Decision Tree, XGBoost, Random Forest, Naïve Banes Model, Support Vector Machine.

Balancing the data: As 93.6% of our data comprises of No Claim (0) data, our models gave us a biased output and mostly predicted a No Claim (0) output. We balanced the data and ran our models the second time to have a more accurate predictive model.

Logistic Regression Model

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X. In our model, Claim = 1; No Claim = 0.

Using the Logistic Regression Model, we tried to optimize the accuracy of using it as a predictive machine learning model. It helps us predict the dependent binary variable of is_claim with potential outcomes of either No Claim (0) or Claim (1). We tried using the class_weight = 'balanced' but it gave us very low accuracy in the model. Thus, we chose to use the data as is without classifying class weight as balanced. *The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y)).*

The logistic regression function $p(\boldsymbol{x})$ is the sigmoid function of $f(\boldsymbol{x})$: $p(\boldsymbol{x}) = 1 / (1 + \exp(-f(\boldsymbol{x}))$. As such, it's often close to either 0 or 1. The function $p(\boldsymbol{x})$ is often interpreted as the predicted probability that the output for a given $\boldsymbol{x}$ is equal to 1. Therefore, $1 - p(x)$ is the probability that the output is 0.

Some quick observations:

- Accuracy score: 0.9355 (very high).
- No Claim (0) Precision: 0.94
- Claim (1) Precision: 0.00
- Confusion Matrix also only predicted No Claim outputs.

Analyzing Cumulative Gains Curve/ Lift Curve

The Cumulative Gain curve shows on y-axis, the amount of gain we would have by approaching a certain percentage of our data on x-axis that affect the binary outcome of Claim or No Claim. Because, the model is not efficient (no positive/ true outcomes predicted), we have poorly reflecting gains curve which does not provide us much insight.

The lift curve chart is derived from the cumulative gains curve chart. The x-axis remains the same being the percentage of sample. Instead of the Gains on the y-axis, we

have the ratio of our model's gains to the gains of a random model, which informs us on how much better our model predicts than randomly guessing.

As the lift curve is influenced by how good the gains curve is, we have a poorly formed lift curve very near to the dotted baseline, not offering much insight.

Evaluating ROC Curve

The ROC Curve (Receiver Operating Characteristic Curve) plots the false-positive rate on the x-axis versus the true positive rate on the y-axis for several different candidate threshold values between 0.0 and 1.0.

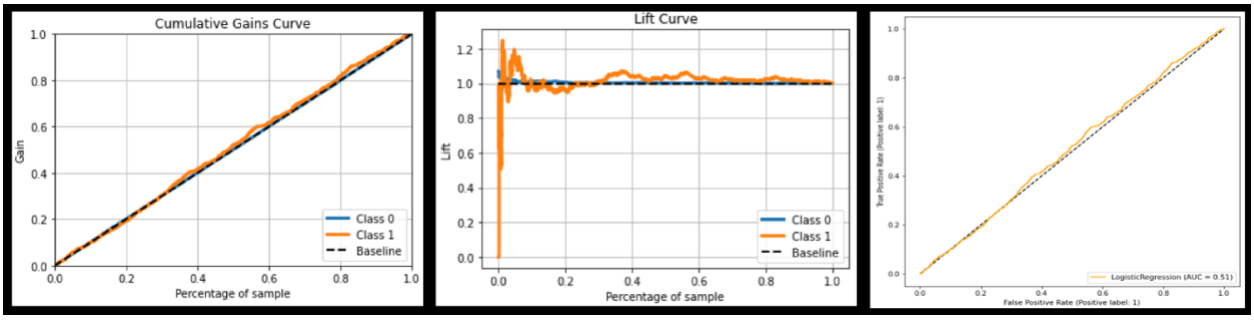
Because our model does not predict anything to be true i.e., predicting outcome of Claim (1), there are no true positives. Thus, sensitivity is zero and essentially, there should not be a curve generated but as Python does some internal smoothing, it generates a line just above the dotted baseline.

A model with no skill at each threshold is represented by a diagonal line from the bottom left of the plot to the top right and has an AUC of 0.5. Hence, our model with AUC of 0.51 has no skill and is not a great choice for predictive modeling in relevance to our data.

Output:

	precision	recall	f1-score	support
No Claim	0.94	1.00	0.97	16444
Claim	0.00	0.00	0.00	1134
accuracy			0.94	17578
macro avg	0.47	0.50	0.48	17578
weighted avg	0.88	0.94	0.90	17578

Graphs:



Next, we now updated the code by balancing the data using pd.concat to consider 'Claim' (1) data as previously the data was biased towards 'No Claim' (0) data and kept predicting 'No Claim' (0) outcomes.

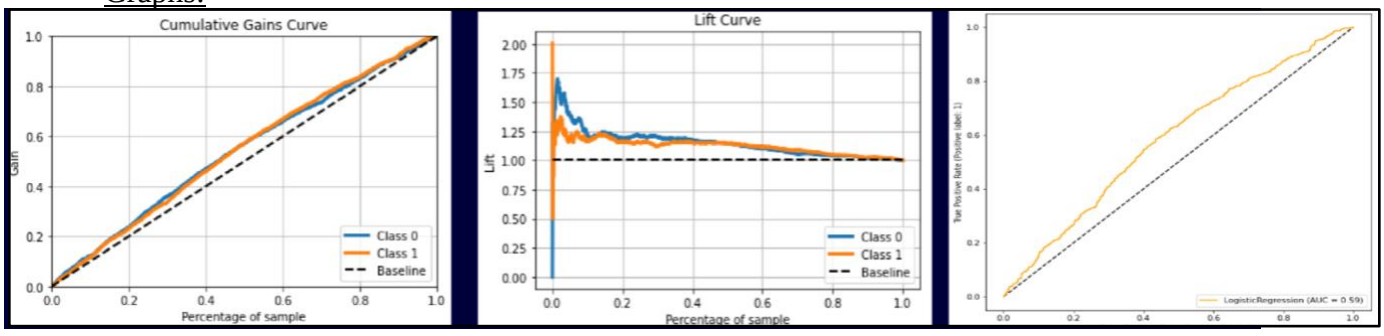
New observations:

- Accuracy score: 0.4984 (lower).
- No Claim (0) Precision: 0.58
- Claim (1) Precision: 0.57
- Confusion Matrix now predicted both Claim (1) and No Claim (0) outcomes.
- AUC: 0.59 (Better); Graphs also slightly better.

Output:

	precision	recall	f1-score	support
No Claim	0.58	0.55	0.56	1128
Claim	0.57	0.59	0.58	1121
accuracy			0.57	2249
macro avg	0.57	0.57	0.57	2249
weighted avg	0.57	0.57	0.57	2249

Graphs:



**ROC/ AUC are a good measure to evaluate performance between different models and*

Cumulative Gains/ Lift Curves as good for seeing how the model responds to the data.

Looking at all our observations and graphs using the Logistic Regression model, it is not the most efficient predictive model for our data with the highest AUC.

Decision Tree Model

Background. Decision tree models are a type of supervised machine learning technique based on information gain. Data is split until 'pure' leaf nodes are obtained. Although pure leaf nodes are ideal, the way in which they are divided isn't always efficient. The goal of a decision tree is to choose splits that result in subsets of examples that have less impurity in the class distribution. The dominance of one or more classes over the other classes is growing concurrently. A decision tree is a graphical structure with a hierarchical organization that aids in carefully examining and correctly categorizing a model. A decision tree can be used to visualize probable outcomes, identify prospective occurrences, and explain decisions. The analysis aids the decision-making process. In this manner, after a few splits, you can locate a subset that only includes instances of one class. Because they can be generalized more effectively than trees with high-cardinality attributes, binary trees are preferred for decision trees.

Information Gain. Entropy is a term coined by Claude Shannon to describe the impurity of an input set. Entropy is a term used to describe the unpredictability or impurity in a system. It alludes to the impurity in a set of instances in information theory. Entropy decrease is referred to as information gain. Based on the attribute values, information gain calculates the difference between average entropy before the split and before the split of the dataset. Information gain is used in the decision tree method ID3 (Iterative Dichotomiser).Gain Ratio

The characteristic with several outcomes is biased in information gain. It implies that it favors the attribute with a lot of distinct values. Take a property with a unique identifier, like customer_ID, for instance, that contains zero info(D) due to pure partition. This increases information acquisition while producing pointless partitioning.

$$Info(D) = -\sum_{i=1}^{m} pi \log_x pi$$

$$Info1(D) = -\sum_{k=1}^{n} \frac{|Dk|}{|D|} X\, Info(D_k)$$

$$Gain(A) = Info(D) - Info1(D)$$

Where:

- Info(D) is the average amount of information needed to identify the class label of a tuple in D.
- |Dk|/|D| acts as the weight of the kth partition.
- Info1(D) is the expected information required to classify a tuple from D based on the partitioning by '1'

The attribute A with the highest information gain, Gain(A), is chosen as the splitting attribute at node N().

Gini Index. A variable's likelihood of being incorrectly categorised when it is randomly selected is measured by the Gini Index or Gini impurity. It can be referred to as pure if each of the components belongs to a single class. Gini Index ranges between 0 and 1. The CART (Classification and Regression Tree) algorithm uses the Gini method to create split points.

$$Gini(D) = 1 - \Sigma \sum_{i=1}^{m} Pi^2$$

Where pi is the probability that a tuple in D belongs to class Ci.

For each attribute, the Gini Index takes a binary split into account. A weighted total of each partition's impurity can be calculated. If a binary split on attribute A partitions data D into D1 and D2, the Gini index of D is:

$$Gini_A(D) = \frac{|D1|}{|D|} Gini(D_1) + \frac{|D2|}{|D|} Gini(D_2)$$

When choosing a splitting attribute for a discrete-valued attribute, look for the subset that provides the lowest gini index for that attribute. For continuous-valued attributes, the technique is to choose a point with a smaller gini index as the splitting point after selecting each pair of neighboring values as a potential split point.

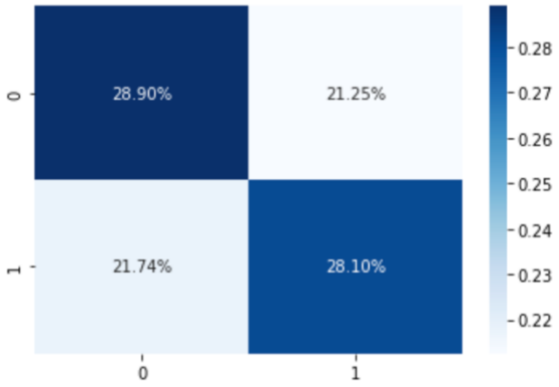
Using the Model. We use decision trees to get a general idea of the data model and to evaluate other models by comparison and contrast. We attempt to reduce overfitting of the data by setting the max depth of the tree as 3. We first ran a decision tree model with all the variables where we get an accuracy of 86.94%. Then, we remove the root node and run the model again. We obtained an accuracy of 86.93%. Although this model isn't ideal, it is easy to manipulate and learn about the data through this model. Due to the skewed nature of our data – about 8% of our data gives 1 for the target variable while the rest returns 0, the precision and accuracy of our data is not up to mark.

Output

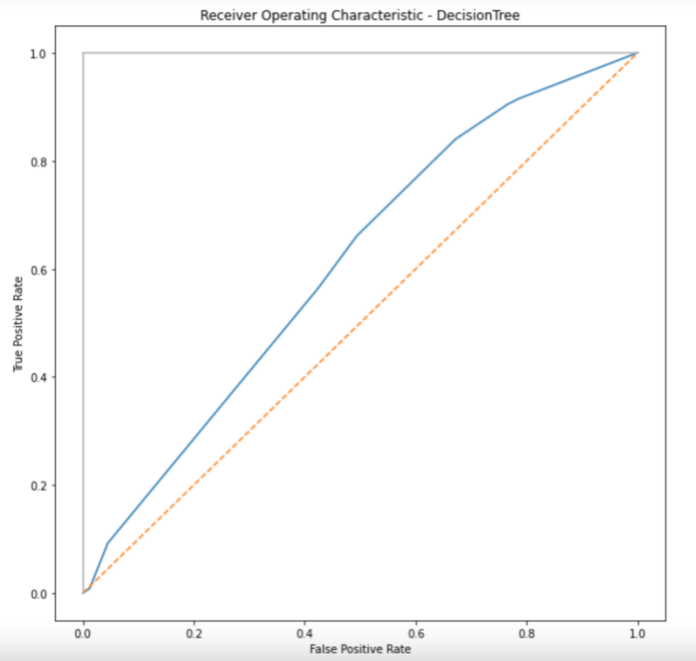
```
Classification report:
Accuracy:  0.5700311249444198
              precision    recall  f1-score   support

           0       0.58      0.57      0.57      1139
           1       0.56      0.57      0.57      1110

    accuracy                           0.57      2249
   macro avg       0.57      0.57      0.57      2249
weighted avg       0.57      0.57      0.57      2249
```

Confusion Matrix

```
Decision Tree Confusion Matrix

<AxesSubplot:>
```

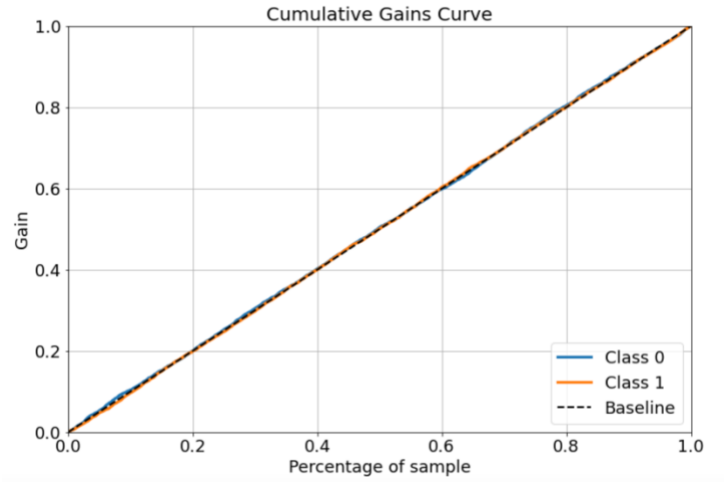


Receiver Operating Characteristic Curve



Area under the curve for Decision Tree: 0.60779857

Cumulative Gains Curve



Random Forest Model

Background. Using a variety of modeling techniques or training data sets, ensemble modeling is the process of building numerous varied models to predict a result. The ensemble model then combines each base model's forecast into a single overall prediction for the unobserved data. The goal of employing ensemble models is to lower the prediction's generalization error. The ensemble technique reduces the model's prediction error as long as the base models are varied and independent. The strategy looks on the collective knowledge of people to make a forecast. The ensemble model behaves and functions as a single model even when it has numerous basis models. Most of the practical data mining solutions utilize ensemble modeling techniques.

A random subset of the data and a random subset of the characteristics are used to build each tree in the forest. After that, the combined projections of all the forest's trees yield the ultimate prediction. Compared to one decision tree, this method provides the following benefits:

- Less Overfitting: Decision trees are prone to overfitting, which causes them to perform badly on fresh data because they fit the training data too well. By averaging the predictions of several trees, Random Forest helps prevent overfitting and produces predictions that are more stable and trustworthy.

- Greater Accuracy: Random Forest frequently outperforms a single decision tree in accuracy, especially for complex datasets. This is due to the fact that the random selection of data and characteristics aids in the capture a larger variety of connections and patterns in the data.

-Outlier Robustness: Decision trees have a tendency to be sensitive to outliers, which can result in incorrect predictions. Because Random Forest combines the predictions of many different trees, it is more resilient to outliers and less affected by any given outlier.

-Scalability: Random Forest is scalable to huge datasets since it is readily parallelized. This is so that every tree in the forest might be constructed separately and concurrently.

Random Forest is a potent machine learning method that can handle complicated datasets, lessen overfitting, and increase accuracy in comparison to a single decision tree.

Random forest is a machine learning algorithm used for both classification and regression tasks. In our case the purpose of the random forest model is to provide an accurate and robust prediction of insurance claims in our data set. The random forest model showed an accuracy of ~ 87% which makes us comfortable to use this model in our predictions. However, the Random Forest classifier showed an AUC measurement of 0.54 which makes us conclude that our model cannot identify accurately our target variable. One of the reasons for those values above is that we might have underfitted our data.

Processing:

The following parameters were used:

```
In [15]: paramGrid = {'n_estimators': [10, 100, 250, 500, 1000],
                      'max_features': ['sqrt', None]}
```

The max_features hyperparameter in Random Forest controls the number of features that are randomly selected at each node to determine the best split. Moreover, the reason we used

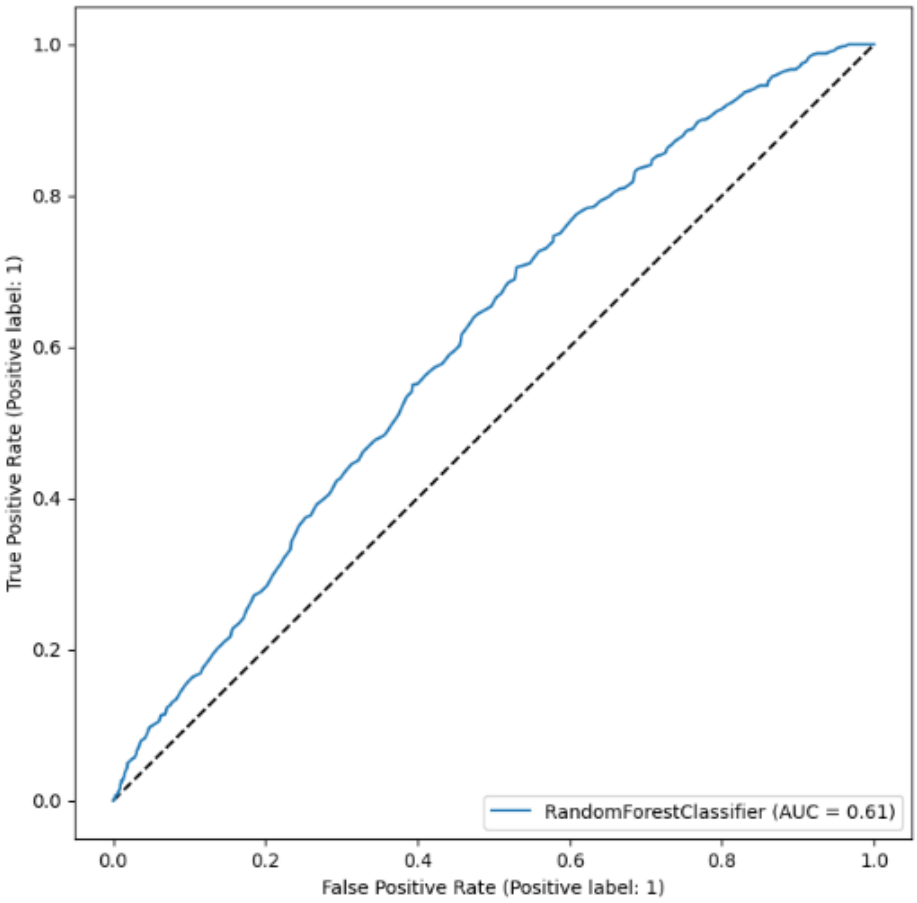
squared root parameter for max_features is that the square root of the total number of features tends to work well in practice. This is because, on average, this will result in a subset of features that is roughly half the size of the full set of features.

After balancing our data the following can be concluded:

New Findings:

Our data was skewed (8% of our data gives were positive for claims “0” and the rest negative for claims “1”, thus our predictive models were yielding inaccurate predictions and were only predicting non-claims. Thus, we had to balance our data. The new balanced data gave us for the Random Forest Model an AUC of .61 and accuracy of .56

```
In [41]: accuracy_score(y_test, testPredsCustomCutoff)
Out[41]: 0.5606666666666666
```



-

Pros & Cons of Random Forest:

Pros:

- highly accurate and robust
- Can be used with different types of data
- Generates feature importance measures

Cons:

- Because of the increase in accuracy of the model, more trees are required to create the model this slowing down the model.
- Can't automatically describe the correlation or relation between the variables.
- Overfitting
- Can struggle with imbalanced data

XGBoost Model

Background

XGBoost (Extreme Gradient Boosting) is a strong machine learning method used for supervised learning issues such as regression and classification. It is a gradient boosting approach that employs decision trees as base learners. We use XGBoost for two main purposes – first for its speed and second for its model performance. When compared to other implementations of gradient boosting, the XGBoost model is very fast. Using the XGBoost model, we obtained an accuracy of 58.34% with an area under the curve at 0.61.

Features of XGBoost:

1. Gradient boosting: The XGBoost technique uses gradient boosting to train a sequence of decision trees to predict the target variable. It employs an additive technique in which each tree is constructed to repair the mistakes of the preceding tree.
2. Regularization: To minimize overfitting, XGBoost employs two types of regularization: L1 regularization (Lasso) and L2 regularization (Ridge). Regularization techniques penalize big coefficients, which helps to minimize model complexity.

3. Feature Impotance: XGBoost delivers a feature significance score that reflects how effective each feature is in predicting the target variable. This can be used to discover the most significant elements in a dataset and increase the model's accuracy.
4. Handling missing values: During training, XGBoost learns the best imputation approach for missing values in the dataset.
5. Parallel processing: XGBoost may use multi-core CPUs to parallelize the training process, dramatically reducing training time for large datasets.
6. Cross-validation: XGBoost offers cross-validation to aid in tuning the model's hyperparameters and preventing overfitting.
7. Early stopping: XGBoost enables early stopping, which allows the model to cease training when the validation set performance no longer improves. This can assist prevent overtraining and cut training time.
8. Multiple programming languages supported: XGBoost supports a variety of programming languages, including Python, R, Java, Scala, and Julia.

In summary, XGBoost is a sophisticated machine learning method that employs gradient boosting as a base learner with decision trees as the foundation learner. It includes, among other things, regularization, feature importance, missing value management, parallel processing, cross-validation, and early termination. Its appeal to data scientists stems from its capacity to handle enormous datasets, give reliable findings, and provide insights on feature relevance.

OBSERAVATIONS USING FULL DATA:

- Accuracy score: 0.59
- No Claim (0) Precision: 0.58
- Claim (1) Precision:

- Confusion Matrix also only predicted No Claim outputs.
- AUC: 0.61 •Graphs unsatisfactory

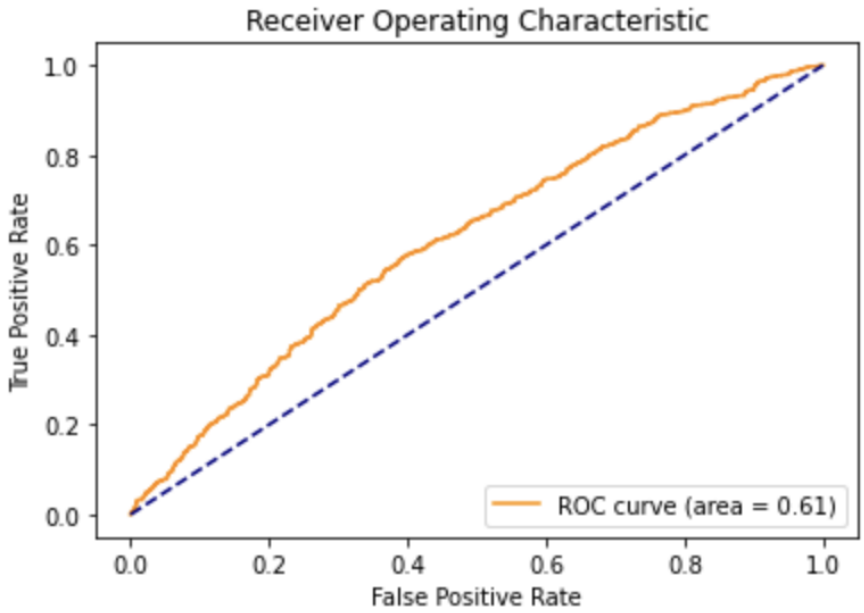
OBSERVATIONS USING BALANCED DATA

- Accuracy score: 58.34
- No Claim (0) Precision: 0.59
- Claim (1) Precision: 0.58
- Confusion Matrix predicted both Claim (1) and No Claim (0) outcomes.
- AUC: 0.61 Graphs also slightly better.

Result. Using the XGBoost model, we obtained an accuracy of 58.34% with an area under the curve at 0.61.

	precision	recall	f1-score	support
0	0.59	0.57	0.58	1128
1	0.58	0.59	0.59	1121
accuracy			0.58	2249
macro avg	0.58	0.58	0.58	2249
weighted avg	0.58	0.58	0.58	2249

Receiver Operating Characteristic Curve



Naïve Bayes Model

The Naive Bayes algorithm is a probabilistic classification machine learning algorithm. This algorithm is based on Bayes' theorem that gives probabilities to each possible event or outcome based on prior knowledge of conditions that may be related to the event.

The Naive Bayes algorithm assumes that the features are independent of one another, meaning that if the value of one feature is changed then it does not directly influence or change the value of any other feature used in the algorithm. This assumption makes it easier to compute the probability of a given event or outcome given a set of features and makes the algorithm simpler and more efficient.

Based on the training set Naive Bayes computes the prior probabilities of each class then calculates the likelihood of each class given the values of its features for each new instance. Later the algorithm combines the prior probabilities and likelihoods to calculate the posterior probabilities of each class. The highest probability is then chosen to be the predicted class for the new instance.

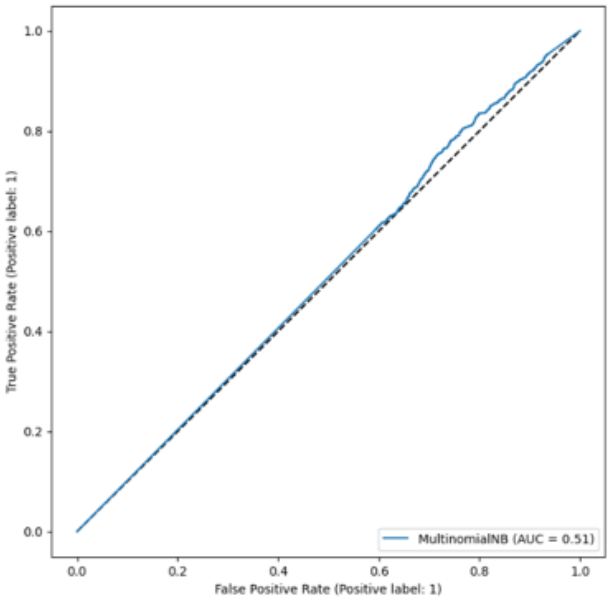
There are three main types of Naive Bayes classifiers Gaussian, Multinomial, and Bernoullli. Gaussian is used to estimate the probability distribution of each feature by calculating the mean and standard deviation of each feature for each class label. This is mainly used for continuous variables thus we won't be using this classifier for our project due to us having both categorical and continuous features. Multinomial is used to categorize documents or text into multiple classes given that some of our features are categorical the Multinomial classifier will be suitable to use. Bernoulli is another suitable classifier since our predicted outcome "is_claim' is a binary output.

For our model we decided to run Multinomial and Bernoulli classifier while using the grid search technique to consider all possible combinations of hyperparameters and find the best hyperparameters for our model. For our model we will use roc_auc for our scoring within the grid search as well as balanced_accuracy due to our data being unbalanced. The accuracy score and roc_auc score are as followed.

Model	**Grid Search Scoring**	**Accuracy Score**	**ROC_AUC**
Multinomial	Balanced Accuracy	40.6246%	51.0662%
Multinomial	AUC_ROC	40.6133%	51.0554%
Bernoulli	Balanced Accuracy	82.2562%	56.4236%
Bernoulli	AUC_ROC	93.5488%	56.4236%

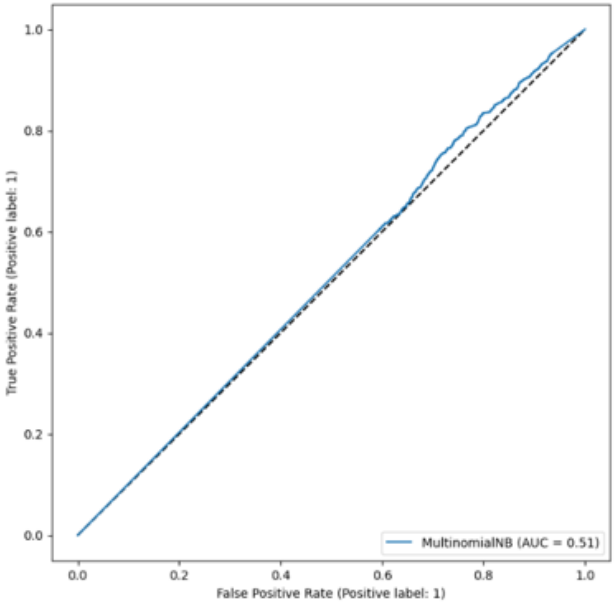
Multinominal ROC_AUC Scoring

	precision	recall	f1-score	support
No Claim: 0	0.94	0.39	0.55	16444
Claim: 1	0.07	0.62	0.12	1134
accuracy			0.41	17578
macro avg	0.50	0.50	0.34	17578
weighted avg	0.88	0.41	0.52	17578



Multinominal Balanced Accuracy Scoring

	precision	recall	f1-score	support
No Claim: 0	0.94	0.39	0.55	16444
Claim: 1	0.07	0.62	0.12	1134
accuracy			0.41	17578
macro avg	0.50	0.50	0.34	17578
weighted avg	0.88	0.41	0.52	17578

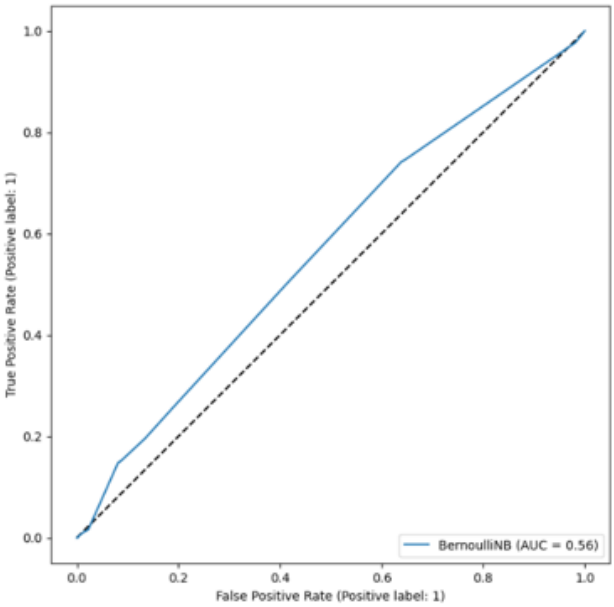


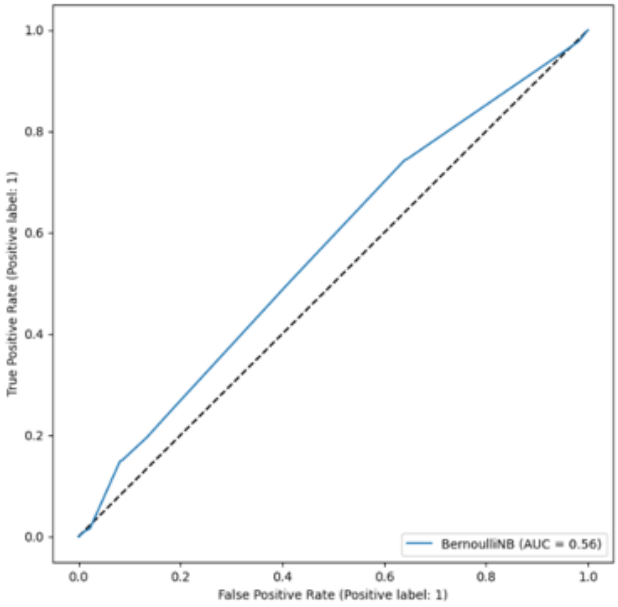
Bernoulli ROC_AUC Scoring

	precision	recall	f1-score	support
No Claim: 0	0.94	1.00	0.97	16444
Claim: 1	0.00	0.00	0.00	1134
accuracy			0.94	17578
macro avg	0.47	0.50	0.48	17578
weighted avg	0.88	0.94	0.90	17578

Bernoulli Balanced Accuracy Scoring

	precision	recall	f1-score	support
No Claim: 0	0.94	0.87	0.90	16444
Claim: 1	0.09	0.20	0.12	1134
accuracy			0.82	17578
macro avg	0.52	0.53	0.51	17578
weighted avg	0.89	0.82	0.85	17578





The AUC score for both models combined with the different grid search scoring all had a similar score around 50 to mid-50%. The roc score is considerably low and is concerning, AUC score around 50% are considered to be ok or “good as random” while 1 is considered to be the best at classifying. The accuracy scores for Bernoulli Naïve Bayes are significantly higher than the accuracy scores for Multinomial. Bernoulli Naïve Bayes with grid search scoring of AUC_ROC has an accuracy score of 93.55% which is significantly high, however the output of the confusion matrix shows that the model is predicting all claims to be 0 or non-claims. We’ve got to note that our data is imbalanced with only 7% of records being claims thus predicting all records to be non-claims is significantly inflating the accuracy score and not an accurate representation of the model. Bernoulli with balanced accuracy scoring had a respectable accuracy score of 82.26% however had a concerning AUC score of 56.42%. Overall due to the

imbalance of the data and considering the low accuracy scores and AUC scores I would suggest that the Naïve Bayes algorithm with either classifier type is not a suitable model to predict our target feature" is_claims".

After running the Naïve Bayes models, we noticed that some Accuracy Scores and AUC scores were relatively low. In addition, some of the models were predicting all claims to be non-claim due to the imbalance of our data with our target variable "is_claim" being 6-7% of our overall data. As a result, we've decided to balance our data and use the new balanced dataset as our starting point for running our models. To balance our data, we kept all of our "is_claim" records (3748) and randomly selected 3748 non claim records to have a balanced 50/50 dataset (50/50). We reran our models and reproduced our Accuracy Score and AUC Score hoping to get some improvement.

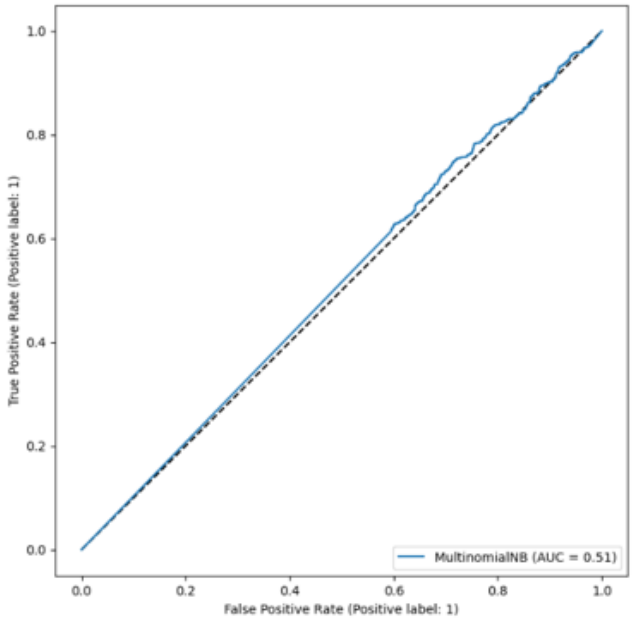
Model	**Grid Search Scoring**	**Accuracy Score**	**ROC_AUC**
Multinomial	Balanced Accuracy	50.9560%	51.1078%
Multinomial	AUC_ROC	50.9560%	51.1078%
Bernoulli	Balanced Accuracy	54.6910%	55.0765%
Bernoulli	AUC_ROC	54.6910%	55.0765%

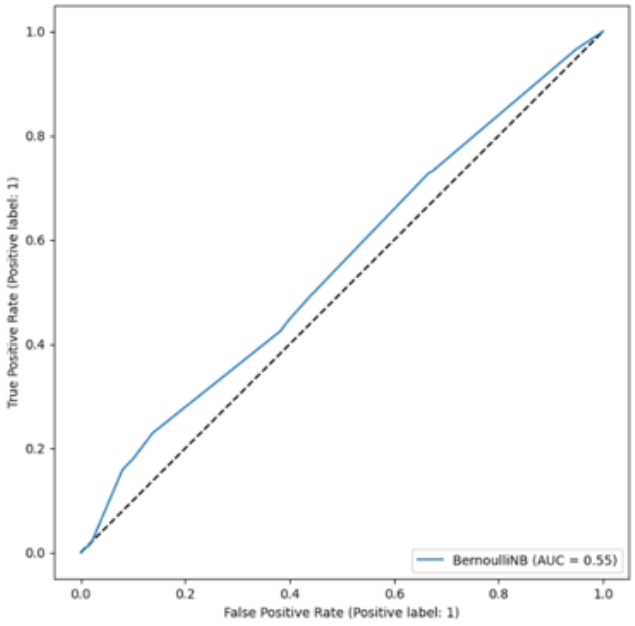
Multinomial

	precision	recall	f1-score	support
No Claim: 0	0.51	0.39	0.44	1128
Claim: 1	0.51	0.63	0.56	1121
accuracy			0.51	2249
macro avg	0.51	0.51	0.50	2249
weighted avg	0.51	0.51	0.50	2249

Bernoulli

	precision	recall	f1-score	support
No Claim: 0	0.53	0.86	0.66	1128
Claim: 1	0.62	0.23	0.34	1121
accuracy			0.55	2249
macro avg	0.58	0.55	0.50	2249
weighted avg	0.58	0.55	0.50	2249





For both Multinomial Naïve Bayes models we had the same Accuracy Score approx. 51% which was a 10pt improvement from our previous dataset. AUC score stayed relatively flat for both models. Accuracy score for both Bernoulli Naïve Bayes models decreased significantly this is due to us balancing our data to be 50/50 claim/non-claim and the model predicting more records to be claims. Reminder in our previous model with previous unbalanced dataset Bernoulli models predicted all records to be non-claim (93% of our data) which results in relatively high accuracy scores. Overall Accuracy score for Bernoulli models went down 30-40 pts from previous run. AUC Scores went down 1pt from 56% to 55%. In summary, considering we balanced our dataset we still produced relatively low Accuracy Scores and AUC Scores for

all Naïve Bayes Models. I would suggest that the Naïve Bayes algorithm with either classifier type is not a suitable model to predict our target feature" is_claims".

Support Vector Machine Model

Background

Support Vector Machines (SVMs) are a type of machine learning algorithm used for classification and regression analysis. SVMs can be used for both linear and nonlinear classification, as well as regression tasks. The main idea behind SVMs is to find the hyperplane that separates the data points into different classes with the maximum margin, i.e., the distance between the hyperplane and the nearest data point of each class is maximized. SVMs are widely used in various fields, including computer vision, natural language processing, and bioinformatics, among others. SVMs are known for their high accuracy, robustness, and effectiveness in dealing with high-dimensional data.

Support Vector Machines (SVM) is a popular supervised machine learning algorithm used for classification and regression tasks. It works by finding an optimal hyperplane that best separates the data points into different classes or predicts the target variable.

SVM KERNELS:

Linear Kernel:

- When the data is linearly separable, the linear kernel is the simplest kernel to utilize. It operates by locating the hyperplane in the original feature space that best separates the classes

Polynomial Kernel:

- Using a polynomial function, the polynomial kernel transfers the input data into a higher dimensional feature space. It comes in handy when the data has non-linear boundaries.

Radial Basis Function (RBF) Kernel:

- The RBF kernel is a prominent kernel that is used when the data cannot be separated linearly. It uses a Gaussian function to translate the input data into a high-dimensional feature space.

SVMs work well in the following scenarios:

SVMs work best when the data is linearly separable, i.e., the data points of different classes can be perfectly separated by a straight line or hyperplane

Binary Classification: SVMs are primarily used for binary classification tasks where the goal is to classify data points into one of two classes. SVMs can handle imbalanced datasets, where the number of samples in different classes is uneven, by adjusting the class weights or using techniques such as oversampling or undersampling.

However, there are scenarios where SVMs may not perform as well:

1. Large Datasets: SVMs can become computationally expensive and time-consuming for large datasets, as the training time increases with the size of the dataset. In such cases, other algorithms such as logistic regression or gradient boosting may be more efficient.

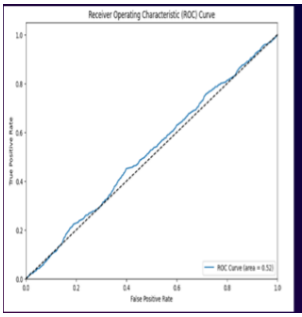
2. Overlapping Classes: SVMs may not perform well when the classes overlap or when there is no clear margin of separation between the classes. In such cases, SVMs may misclassify data points, leading to lower accuracy.

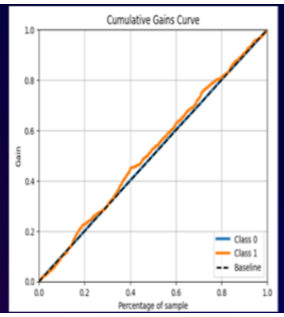
ROC:

Most of the inputs of this dataset are categorical means some of them have yes/no values , some of them have 0/1, and some of them have true/false value. If I am going to discuss about the model is suitable for this dataset or not then we need to understand that An accuracy of 0.50 and an ROC curve area of 0.51 for a Support Vector Machine (SVM) model may suggest that the model is not performing well and may not be suitable for this dataset.

An accuracy of 0.50 means that the model is making correct predictions only half of the time, which is essentially the same as random chance. An ROC curve area of 0.51 indicates that the model's ability to discriminate between positive and negative instances is only slightly better than random chance (which would have an area of 0.50).

FINAL RESULTS:
OBSERVATIONS USING FULL DATA



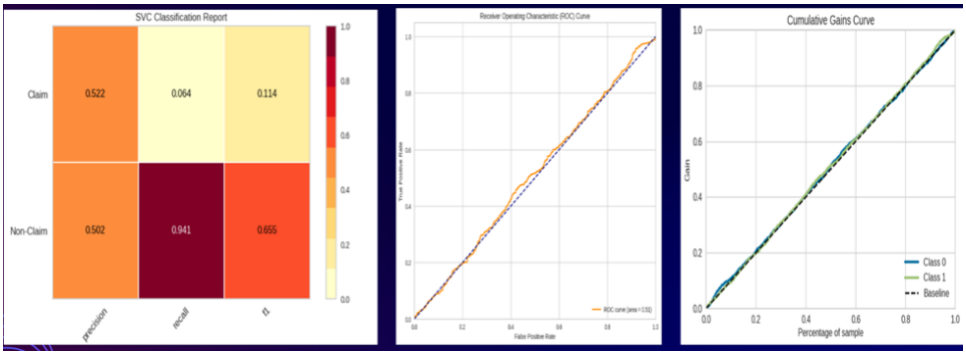


Classification Report:

	precision	recall	f1-score	support
0	0.94	0.99	0.96	10963
1	0.07	0.01	0.02	756
accuracy			0.92	11719
macro avg	0.50	0.50	0.49	11719
weighted avg	0.88	0.92	0.90	11719

•Accuracy score: 0.91
•No Claim (0) Precision: 0.94
•Claim (1) Precision:0.07
•AUC: 0.52
•Graphs unsatisfactory

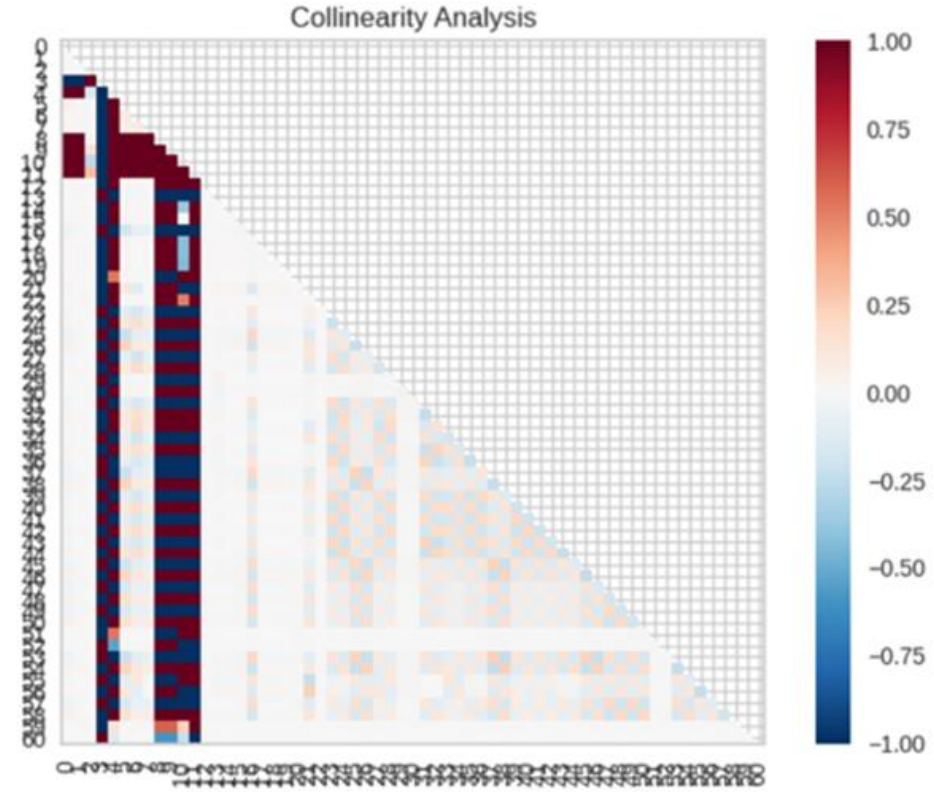
OBSERVATIONS USING BALANCED DATA:



• Accuracy score:0.53
•No Claim (0) Precision: 0.52
• Claim (1) Precision: 0.50
•AUC: 0.51
•Graphs also slightly better.

COLLINEARITY ANALYSIS:

this diagram shows the relation between pairs of values, and well they are co-related



<Axes: title={'center': 'Collinearity Analysis'}>

Conclusion

Model	***Accuracy (w balanced data)***	***AUC***
Logistic Regression	*0.4984*	*0.59*
Decision Tree	*0.5700*	*0.61*
XGBoost	*0.5834*	*0.61*
Random Forest	*0.5607*	*0.61*
Naïve Bayes Model	*0.5469*	*0.55*
Support Vector Machine	*0.53*	*0.51*

In conclusion, after our analysis the model that yielded the best results for predicting our target variable "is_claim" was XG Boost. XG Boost was the best model after we balanced our data, 50/50 claim and non-claim, due to our data having the majority (93.6%) tagged as non-claims. Due to the unbalanced data some of our models (Linear regression, Naïve Bayes, etc.) were predicting all the data to be non-claims. After balancing our data XG Boost gave the best accuracy score of 0.5834 and best AUC score of 0.61.

Insurance companies leverage many data points to price a single risk. One recommendation to increase the model's accuracy is to leverage policyholder data. As mentioned earlier in the paper this dataset had more car features (38 of 42) compared to policyholder features (4 of 42). Features like policyholder credit score and prior claim history can be high indicators of future potential losses. Lastly, leveraging telematic data can help improve our models since its driver behavior data which can be highly indicative of future potential losses. Telematic data such as miles driven, acceleration grade, braking grade and much more can help increase accuracy score of any given model.

Ethics

The insurance sector must address the moral questions raised using artificial intelligence (AI) as it adopts technology to enhance operations and the customer experience. AI can improve the efficacy and efficiency of the insurance sector, but it can also be used to make judgments that have a significant influence on people's lives, raising questions about justice, prejudice, accountability, and transparency.

Fairness is one moral dilemma that comes from the application of AI in insurance. To decide who to insure, how much to charge for premiums, and how much to pay out in claims, insurance firms utilize algorithms and machine learning models. Certain groups of people, such as those who belong to a particular race or gender, may be unfairly disadvantageous if the algorithms and models are biased or discriminatory.

The insurance industry is ever evolving by continuously adjusting their models to adequately price risk, predict claims, identify potential fraudulent claims etc. With the ease and accessibility to regularly leverage big data, insurance companies must be aware of the risk associated with using this data. There are certain ethical and security concerns that issuance companies need to be aware and cognizant of while using this data. From an ethical perspective one concern is the use of data to unintentionally discriminate against certain groups of people such as those with pre-existing conditions or certain races that can potentially lead to unequal treatment and access to insurance. In the article “Insurance Regulators Pledge to Address Racism And Discrimination Within The Industry”[5] it states that the states of California, Hawaii, and Massachusetts prohibited the use of credit history in the calculation of auto insurance premiums. Additionally, New York disallowed the use of education level and occupation as pricing factors. A second concern to keep in mind is invasion of privacy. A major topic in recent times is the use

of personal data and getting consent from the individual to gather and use that data. A notable example of this is the use of telematics, which is technology to gather driver behavior. This data is collected via a plug in or an app. Driver data such as speed, miles driven, acceleration, braking and much more is being collected and used in ratings by insurance companies. The third concern is lack of transparency when using this data. Insurance companies use multivariate models to price risks and predict claims, these models can be very technical and complex which leading to difficulty of understanding from the side of the consumer. Fortunately, there is a department of insurance for all states that analysis things from increase/decrease in pricing, model changes, new data adaptations etc. These departments are there to protect the consumer from unfair price increases and misuse of data that would create unfair treatment. Lastly, insurance companies should always be concerned about data quality/basis and unintended consequences. If a model is fed bias or substandard quality data, then the results from our model would be biased or we will result in poor analytics.

As insurance companies continue to leverage big data, they need to think of the ethical use of data and its security. All companies are prone to cyber security attacks, "Over two-thirds (68%) of organizations say they suffered a cyberattack at some point during the last twelve months." [6] Cyberattacks can lead to serious financial damage, damage reputation and can cause customer churn which will lead to a decrease in revenue. Insurance companies collect substantial amounts of sensitive data from social security numbers, phone numbers, addresses and bank information which makes them vulnerable to cyberattacks and data breaches. Some of this big and sensitive data is acquired directly from the customer or from third party vendors. Many insurance companies work with third party vendors to get data and use it in rate in hopes of getting some competitive advantage. The transfer of this data can expose both companies to

data breaches thus both companies must ensure that strong and strict data protocols to guarantee safe transaction of data. Lastly, data retention is another security concern any company must keep in mind. Insurance companies must find a secure way to keep their customers' historical data safe and secure. Once the data is no longer in use, they must find a way to safely purge and delete this data from their database to ensure this information is not being used or shared.

Now more than ever the importance of data ethics, compliance, and security is at its highest due to the accessibility of big data. Insurance companies must ensure ethical use of data so no one group is favored or discriminated against. To ensure fairness is pricing and accessibility of insurance. Data compliance with legal and regulatory requirements is necessary to avoid legal penalties and reputational damage. Lastly, data security is vital to any organization to guarantee protection from cyberattacks and data breaches which can lead to high financial loss and reputational damage. By focusing on data ethics, compliance, and security insurance companies can build trust, accountability, and transparency with their customers on the way they collect, manage, and use data leading to greater outcomes.

References

1. https://www.investopedia.com/ask/answers/051915/how-does-insurance-sector-work.asp
2. https://www.claimsjournal.com/app/uploads/2021/09/AutoInsuranceClaimsCostsIncreasing-678.pdf#:~:text=Higher%20severity%20along%20with%20rising%20claims%20frequency%20are,amid%20severe%20shortages%20of%20new%20and%20used%20vehicles
3. https://www.investopedia.com/terms/l/loss-ratio.asp
4. https://www2.deloitte.com/us/en/insights/industry/financial-services/insurance-claims-transformation.html
5. https://www.forbes.com/sites/advisor/2020/07/23/insurance-regulators-pledge-to-address-racism-and-discrimination-within-the-industry/?sh=2ad1b82f2c00
6. https://www.msn.com/en-us/money/other/more-companies-are-being-hit-by-cyberattacks-than-ever-before/ar-AA1a4bmy