Liberty Insurance Case Study

Dev: Mateus Broilo da Rocha

Statement of the Problem

Problem:

Presently Liberty Insurance Co. has a very complicated problem regarding lack of trustful knowledge on claims that a policyholder is most likely to file during the exposure time policy.

Formal Statement:

Build a model able to predict the total number of claims a customer is going to file with the company.

Liberty Insurance

Is there any dataset available?

Yes!

A dataset containing policy info of motor insurance customers and the total claims they have filed.

More Specifically:

- *This dataset initially contains 12 columns and 678013 rows.*
- *The total number of claims per customer is represented by claim_count.*
  - *This column is the TARGET feature.*
  - *The expected frequency of claims is calculated by using num_feature (which is an offset variable)*
  - *All the rest can be viewed as RESOURCE features.*

	policy_desc	claim_count	cat_areacode	num_vehicleAge	num_noClaimDiscountPercent	cat_carBrand	num_populationDensitykmsq	cat_Region	ord_vehicleHP	num_exposure	cat_fuelType	num_driverAge
0	1	1	D	0	50	B12	1217	R82	5	0.1	Regular	55
1	3	1	D	0	50	B12	1217	R82	5	0.77	Regular	55
2	5	1	B	2	50	B12	54	R22	6	0.75	Diesel	52
3	10	1	B	0	50	B12	76	R72	7	0.09	Diesel	46
4	11	1	B	0	50	B12	76	R72	7	0.84	Diesel	46
...	...	...	...	...	...	...	...	...	...	...	...	...
678008	6114326	0	E	0	50	B12	3317	R93	4	0.002739726	Regular	54
678009	6114327	0	E	0	95	B12	9850	R11	4	0.002739726	Regular	41
678010	6114328	0	D	2	50	B12	1323	R82	6	0.002739726	Diesel	45
678011	6114329	0	B	0	50	B12	95	R26	4	0.002739726	Regular	60
678012	6114330	0	B	6	54	B12	65	R72	7	0.002739726	Diesel	29

678013 rows × 12 columns

Liberty Insurance.

Tell me more about this data!

Sure thing!

Variables:

- ***Target:***
  - *claim_count: Total Claims (This is the response you should predicting); Numeric Variable*
- ***Offset:***
  - *num_exposure: Exposure time of policy. Time period within which the claims were made; Numerical Variable*
- ***Independent:***
  - *policy_desc: Policy Identifier; Primary Key which is unique for every policy*
  - *cat_areacode: Area Code; Categorical Variable*
  - *num_vehicleAge: Age of the vehicle; Numeric Variable*
  - *num_noClaimDiscountPercent: Percentage of discount applied to policy premium based on claim history. If value is greater than 100 then policy premium was increased, if it's less than 100 a discount was applied. A value of 100 means the premium remain unchanged;*
- ***Numerical Variable***
  - *cat_carBrand: Insured Vehicle Brand; Categorical Variable*
  - *num_populationDensitykmsq: Population density of the city the policy holder lives in; Numerical Variable*
  - *cat_Region; Region of the country the policy holder lives in; Categorical Variable*
  - *ord_vehicleHP: Vehicle Horsepower; This feature is anonymised but maintains the same ordinality; Ordinal Variable*
  - *cat_fuelType: Insured Vehicle Fuel Type; Categorical Variable*
  - *num_driverAge: Age of the Policy Holder; Numerical Variable*

Liberty Insurance.

Anything else?

Yes!

cat_fuelType → 22 null entries (or missing blank), *i.e. 0.003%*

num_driverAge → 14 null entries (or missing blank), *i.e. 0.002%*

	COLUMN_NAME	COLUMN_DTYPE	#_NULL	#_NON_NULL	%_NULL	%_NON_NULL	UNIQUE_VALUES
0	policy_desc	int64	0	678013	0.000	100.000	678013
1	claim_count	int64	0	678013	0.000	100.000	11
2	cat_areacode	object	0	678013	0.000	100.000	6
3	num_vehicleAge	int64	0	678013	0.000	100.000	78
4	num_noClaimDiscountPercent	int64	0	678013	0.000	100.000	115
5	cat_carBrand	object	0	678013	0.000	100.000	11
6	num_populationDensitykmsq	int64	0	678013	0.000	100.000	1607
7	cat_Region	object	0	678013	0.000	100.000	22
8	ord_vehicleHP	int64	0	678013	0.000	100.000	15
9	num_exposure	object	0	678013	0.000	100.000	184
10	cat_fuelType	object	22	677991	0.003	99.997	4
11	num_driverAge	object	14	677999	0.002	99.998	165

And...

Some data type irregularities

num_exporuse → was loaded as an object, however it must be a numerical variable (more specifically float64).

num_driveAge → was loaded as an object, however it must be a numerical variable.

Moreover

num_driveAge → there're string entries as well as numeric entries.

- There're also Not A Number entries (nan) and missing values (string spaces).
- String spaces corresponds to 24 entries, which is 0.007% of the dataset (splitted).
- Nan corresponds to 0.011%.

(*) The effect of removing these rows most likely won't interfere in the final result.

```
df.loc[(df['num_driverAge'] == ' ') | (df['num_driverAge'] == '  ')]
```

executed in 146ms, finished 21:51:51 2021-09-29

	policy_desc	claim_count	cat_areacode	num_vehicleAge	num_noClaimDiscountPercent	cat_carBrand	num_populationDensitykmsq	cat_Region	ord_vehicleHP	num_exposure	cat_fuelType	num_driverAge
258	546	1	D	0	50	B12	1489	R52	5	0.76	Regular	
105969	1032478	0	C	12	68	B6	238	R54	8	1	Diesel	
165364	1148598	0	D	3	50	B2	1054	R24	9	0.92	Regular	
212991	2026077	0	D	0	50	B12	999	R73	7	0.57	Diesel	
217702	2041616	0	E	0	50	B12	4128	R52	7	0.005479452	Diesel	
233954	2073771	0	C	4	50	B2	175	R24	5	0.8	Diesel	
268441	2136673	0	E	7	85	B2	2767	R52	8	0.5	Diesel	
329623	2262769	0	B	5	68	B3	58	R24	6	0.5	Regular	
343375	2276521	0	C	9	50	B1	242	R26	7	0.5	Regular	
364615	3017281	0	C	2	58	B12	377	R93	4	0.53	Regular	
376870	3037006	0	E	1	64	B12	4128	R52	4	0.09	Diesel	
377929	3038864	0	B	1	50	B12	66	R91	12	0.03	Regular	
436711	3158719	0	B	4	50	B3	56	R73	7	1	Diesel	
447620	3172897	0	A	7	54	B1	42	R24	7	0.002739726	Regular	
530452	4121630	0	D	10	50	B11	564	R11	7	1	Regular	
557180	4148358	0	C	13	51	B2	115	R24	7	0.74	Regular	
566668	4157846	0	C	3	50	B2	141	R82	5	0.62	Diesel	
570981	4162159	0	B	1	50	B1	57	R24	4	1	Regular	
621480	5065996	0	A	1	60	B12	25	R91	4	0.87	Diesel	
624661	5073421	0	F	9	80	B3	27000	R11	9	0.008219178	Regular	
624718	5073597	0	B	3	50	B12	80	R54	4	0.7	Regular	
635940	5100367	0	D	0	50	B12	657	R11	7	0.04	Diesel	
650714	6043152	0	E	3	50	B12	2563	R11	6	0.74	Regular	
676421	6112739	0	D	2	50	B12	918	R11	4	0.32	Regular	

And...

Some data type irregularities

num_exporuse → was loaded as an object, however it must be a numerical variable (more specifically float64).

num_driveAge → was loaded as an object, however it must be a numerical variable.

Moreover

num_exporuse → was loaded as an object, however it must be a numerical variable (more specifically float64).

- There were entries in num_exposure which ended with "years".

cat_fuelType → `array(['Regular', 'Diesel', nan, 'Electric'], dtype=object)`

```
* Regular: 51.01%
* Diesel: 48.99%
* Electric: 0.002%
```

(*) The nan entries will be replaced by Not Informed (NI)

```
df.num_exposure.unique()
```

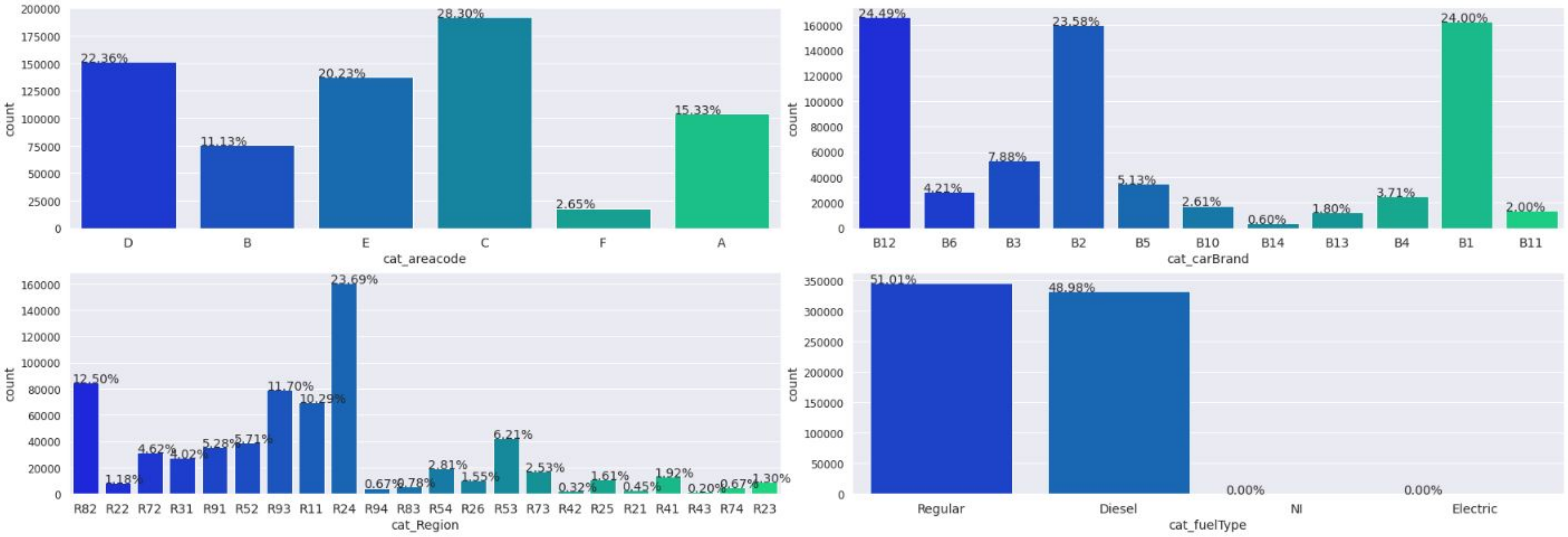
xecuted in 80ms, finished 21:51:52 2021-09-29

```
array(['0.1', '0.77', '0.75', '0.09', '0.84', '0.52', '0.45', '0.27',
       '0.71', '0.15', '0.87', '0.81', '0.05', '0.76', '0.34', '0.74',
       '0.03', '0.06', '0.55', '0.19', '0.01', '0.79', '0.04', '0.8',
       '0.07', '0.39', '0.47', '0.69', '0.16', '0.12', '0.41', '0.46',
       '0.82', '0.11', '0.08', '0.02', '0.72', '0.14', '0.5', '0.92',
       '0.9', '0.78', '0.83', '0.67', '0.13', '0.59', '0.21', '0.65',
       '0.42', '0.85', '0.73', '0.23', '0.48', '0.37', '0.86', '0.17',
       '0.24', '0.63', '0.51', '0.64', '0.66', '0.18', '0.31', '0.35',
       '0.25', '0.62', '0.22', '0.7', '0.58', '0.28', '0.61', '0.53',
       '0.2', '0.008196721', '0.005464481', '0.43', '0.33', '0.57',
       '0.36', '0.56', '0.3', '0.6', '0.68', '0.97', '0.54', '0.44',
       '0.49', '0.29', '0.32', '0.26', '0.00273224', '0.96', '0.98',
       '0.88', '0.4', '0.89', '0.38', '0.95', '0.94', '0.93', '0.91', '1',
       '0.99', '23years', '0.005479452', '0.002739726', '0.008219178',
       '10years', '37years', '1.99', '1.17', '1.12', '1.48', '1.5',
       '1.53', '1.01', '1.03', '1.09', '1.56', '1.07', '1.43', '1.34',
       '1.46', '1.37', '1.41', '1.05', '1.04', '1.02', '1.08', '1.06',
       '1.15', '1.13', '1.2', '1.11', '1.14', '1.16', '1.18', '1.22',
       '1.23', '1.26', '1.1', '1.3', '1.38', '1.29', '1.25', '1.32',
       '1.45', '1.21', '1.33', '1.27', '1.24', '1.36', '1.19', '1.4',
       '1.65', '1.98', '1.35', '1.28', '1.51', '1.52', '1.49', '1.31',
       '1.74', '1.44', '2.01', '1.93', '2', '1.75', '1.54', '1.69',
       '1.88', '1.9', '1.67', '1.82', '1.6', '1.85', '1.92', '1.63',
       '1.71', '1.39', '1.7', '1.64', '1.55', '1.62'], dtype=object)
```

Liberty Insurance

Story time!
Once upon a time a dataset...

Univariate EDA



Liberty Insurance

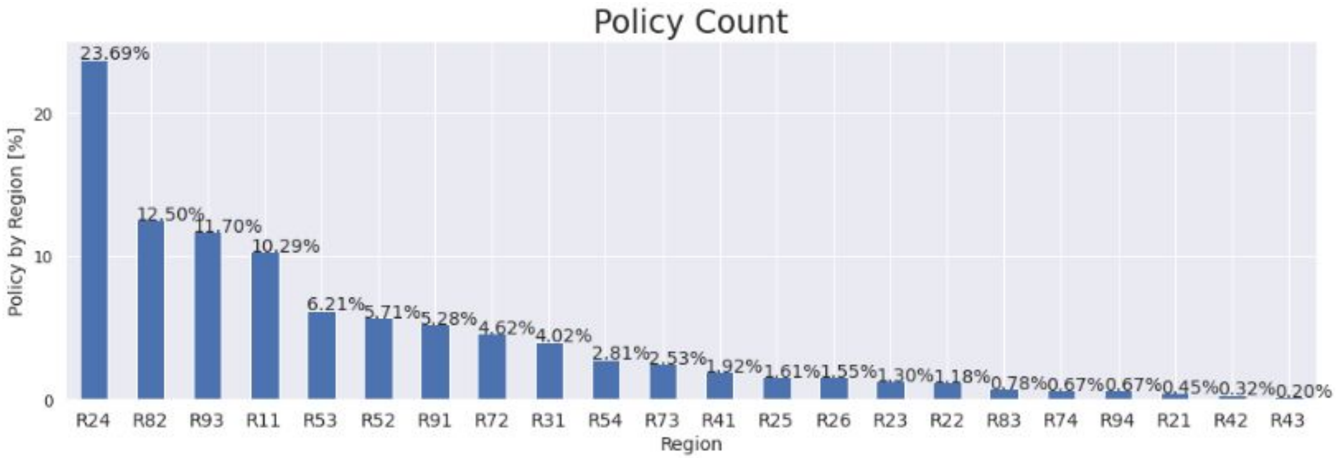
Univariate EDA

- **Area: *cat_areacode***
  - Area C + D + E correspond to approximately 71% of the whole entries in the dataset.
  - But there's one area (F) that has a very low presence (3%).
  - The remaining A + B areas states for 26%

- **Region: *cat_Region***
  - There'a a pick in Region R24 which corresponds to ~24% of the whole entries.
  - However, regions R11 + R24 + R82 + R93 account to approximately 58% of the dataset.
  - The remaining regions account to approximately 42%.

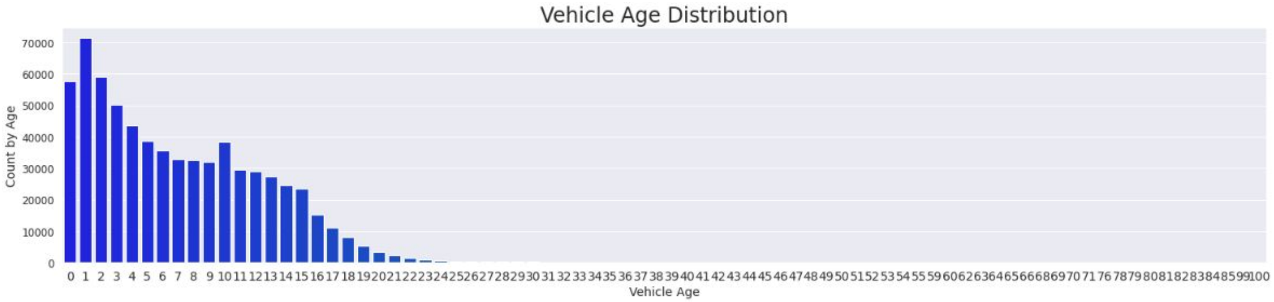
- **Fuels: *cat_fuelType***
  - Fuel Diesel + Regular takes over most of the dataset, ~99.9%.
  - There're some entries which corresponds to Electric, but is very "insignificant" in comparison with Diesel and Regular.
  - Moreover, there were some missing values, which were marked as Not Informed (NI), but it also corresponds to a insignificant amount of entries.

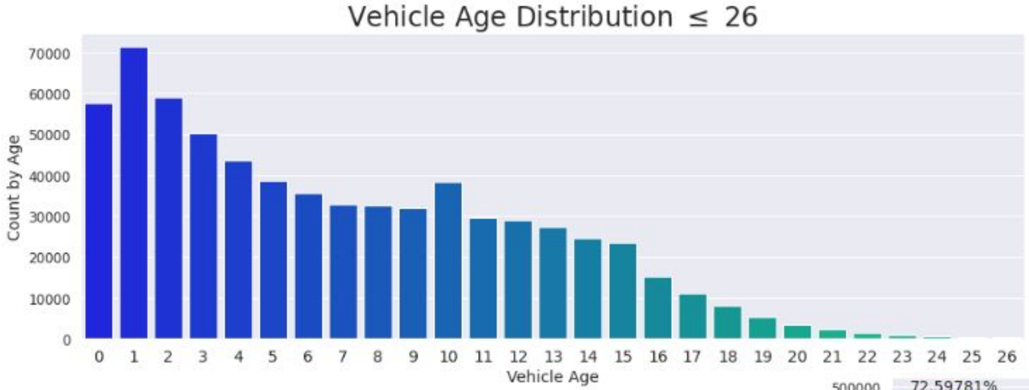
- **Brand: *cat_carBrand***
  - Brand B1 + B2 + B12 correspond to approximately 72.0% of the whole entries in the dataset.
  - Brand B3 + B5 group is the second most significant in size, but it's just 13%.
  - The rest of the brands correspond to ~15%



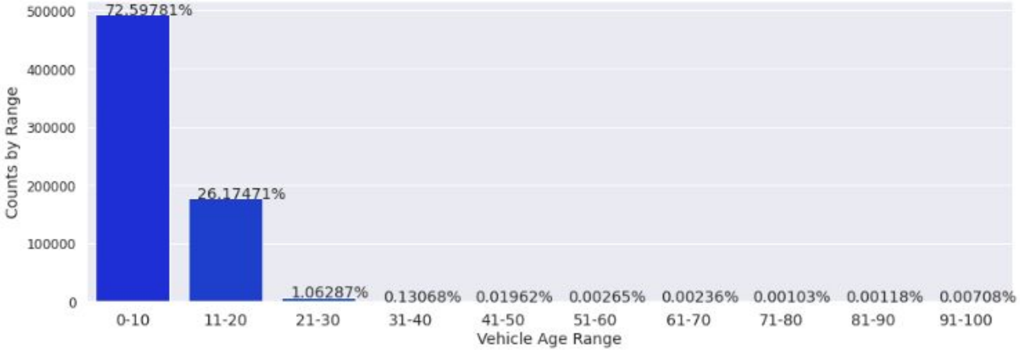
Liberty Insurance.

Univariate EDA





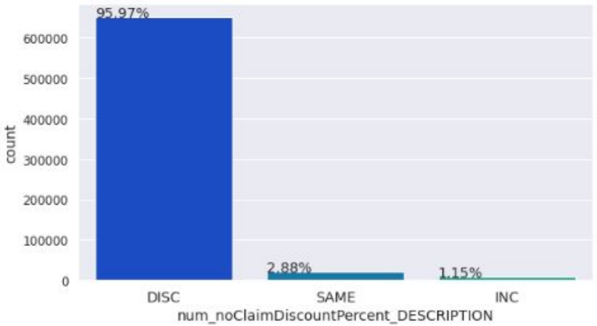
Most of the vihicles (~73%) appears to be concentrated in the age range 0 .LE. age .LE. 10.

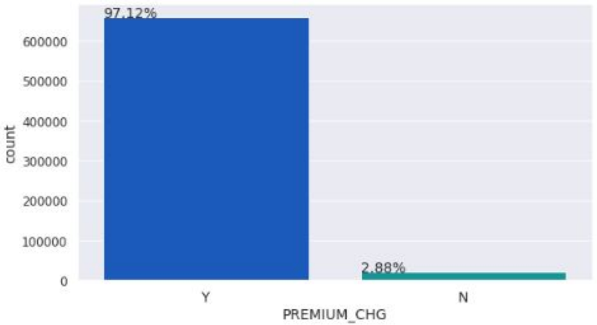


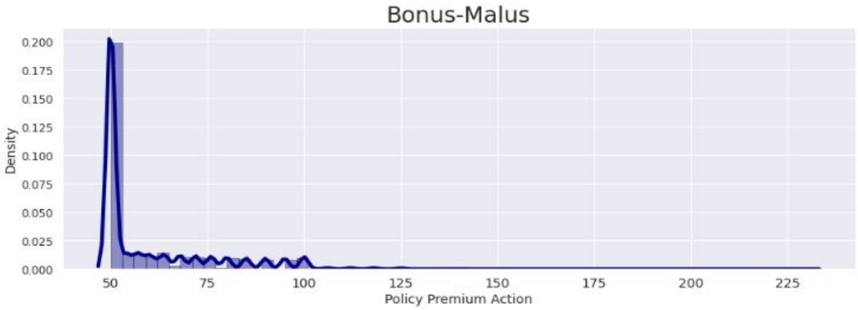
~27% are in the interval 10 .LE. age .LE. 20.

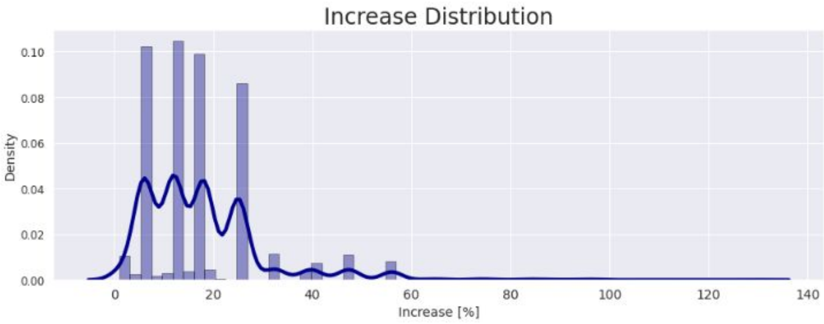
Liberty Insurance.

Univariate EDA





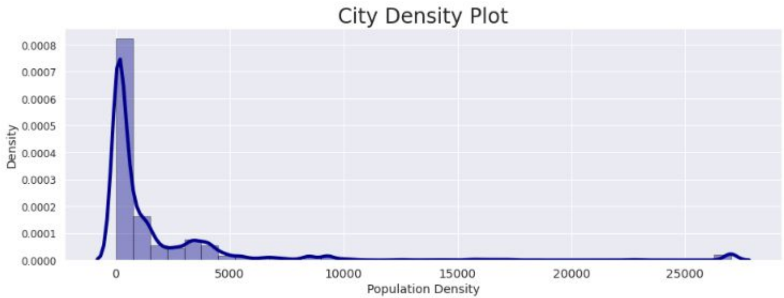


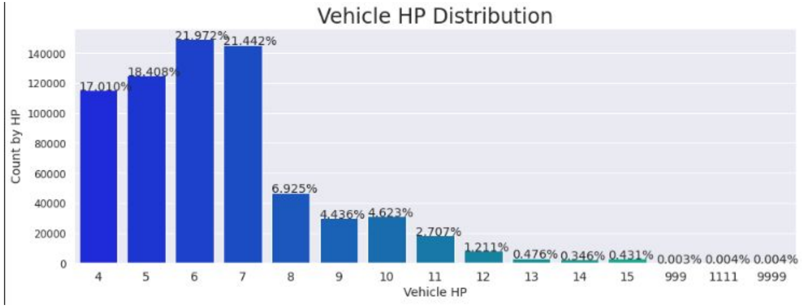


- Almost 96% corresponds to discounts applied to policy premium.
- ~3% of the policy premium remain unchanged.
- And just ~1.2% actually had an increase.
- Therefore, 97% of the dataset had some kind of applied action.
- It was created 5 new resources:
    - num_noClaimDiscountPercent_DESCRIPTION: specifies the action applied
    - num_PREMIUM_DISC: by how much whether the action was a discount
    - num_PREMIUM_INC: by how much whether the action was an increase
    - PREMIUM_CHG: wheter there was an action

Liberty Insurance.

Univariate EDA

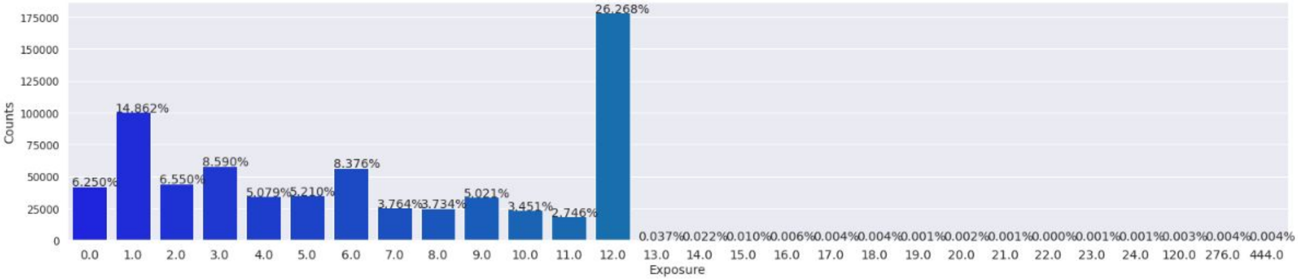


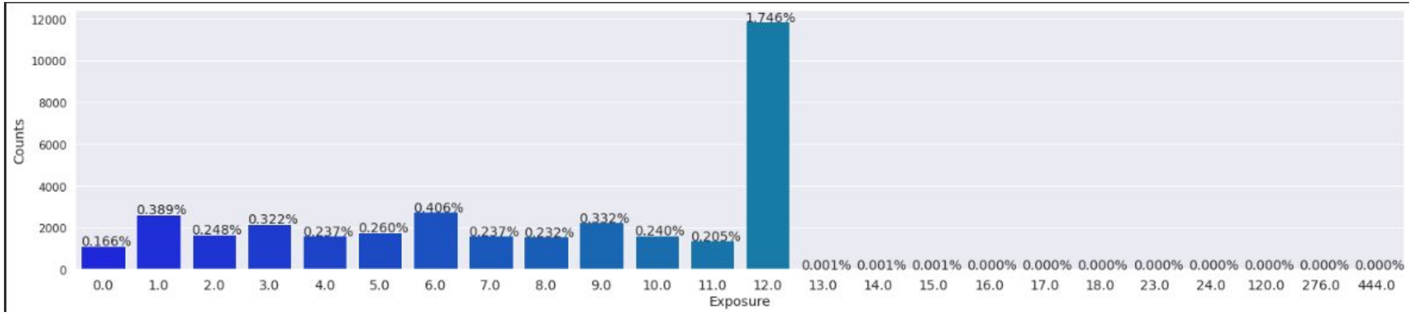


- ➔ Almost 95% of the customers have no claims at all
- ➔ Only ~5% of them have one claims
- ➔ And less then 0.5% have 2 or more claims.
- ➔ Created a new resource which states whether there was a claims: IS_CLAIM
    - ◆ This could be interesting to use in a classification problem. For instance, verify the propensity of a given customer made a claim. However, this would imply in a highly unbalanced dataset. One possible way out could be by means of data augmentation (unsing Random Oversampling in the lowest classification, Over and Under Sampling, or using a sintetic algorithm that generates data, *e.g.* SMOTE. On the other hand, depending on the ML-classifier, for example XGBoost has a hyperparameter which deals with unbalanced data.)

Liberty Insurance

Univariate EDA



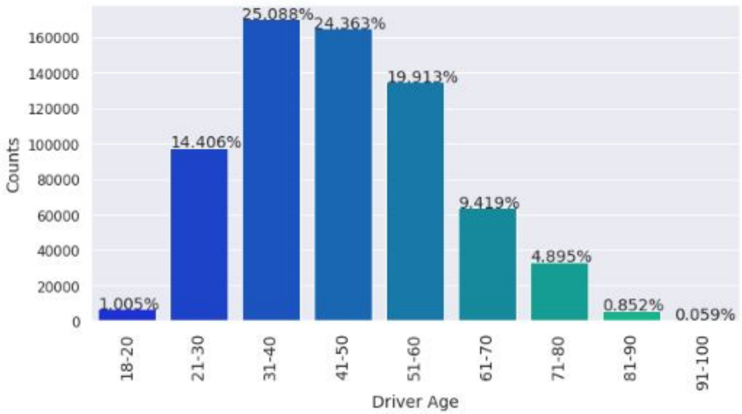


- ➔ I think that I didn't quite get this info. I mean, let's work out an example:
  - ◆ So, taking into account that num_exposure is, by definition, the exposure time of the policy.

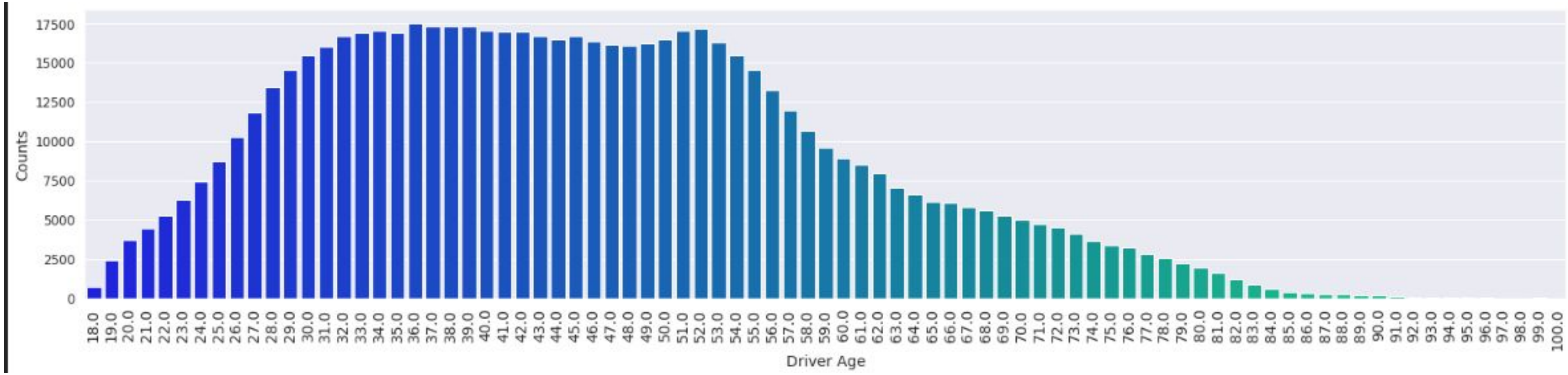
(*) Then, in the new column where it is represented the exposure by months, in the 12th month, which is where 26% of the customers have their exposure time policy insurance, approximately 1.7% of them actually made some claim due to any loss?

Liberty Insurance.

Univariate EDA



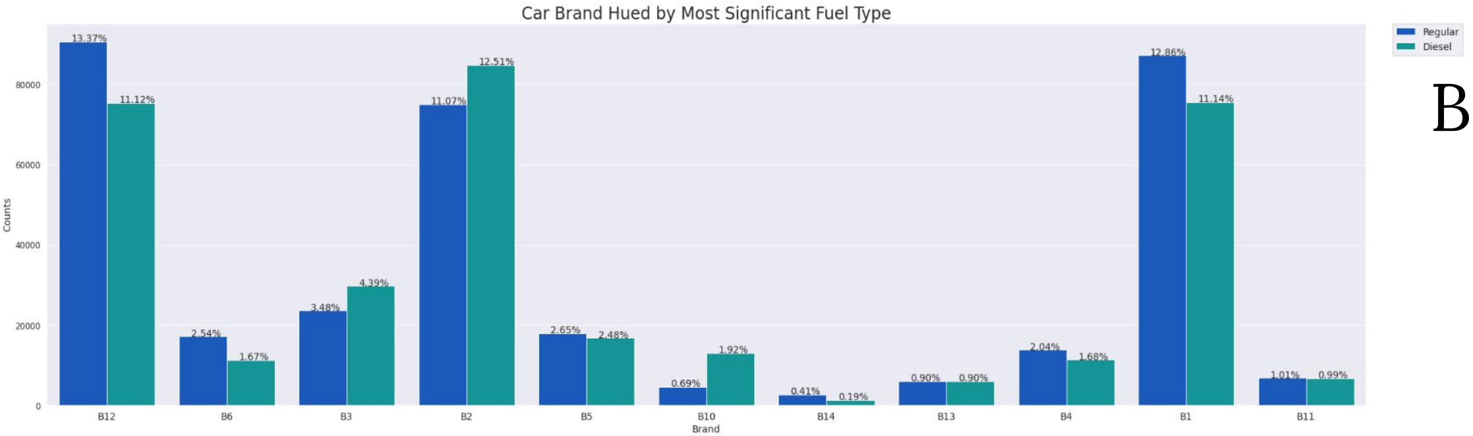
- Most of the customers, ~49%, appears to be betwqeen 31 and 50 yearls old.
  - It was created a two new resource which is:
  - The driver age range: num_driverAge_RANGE

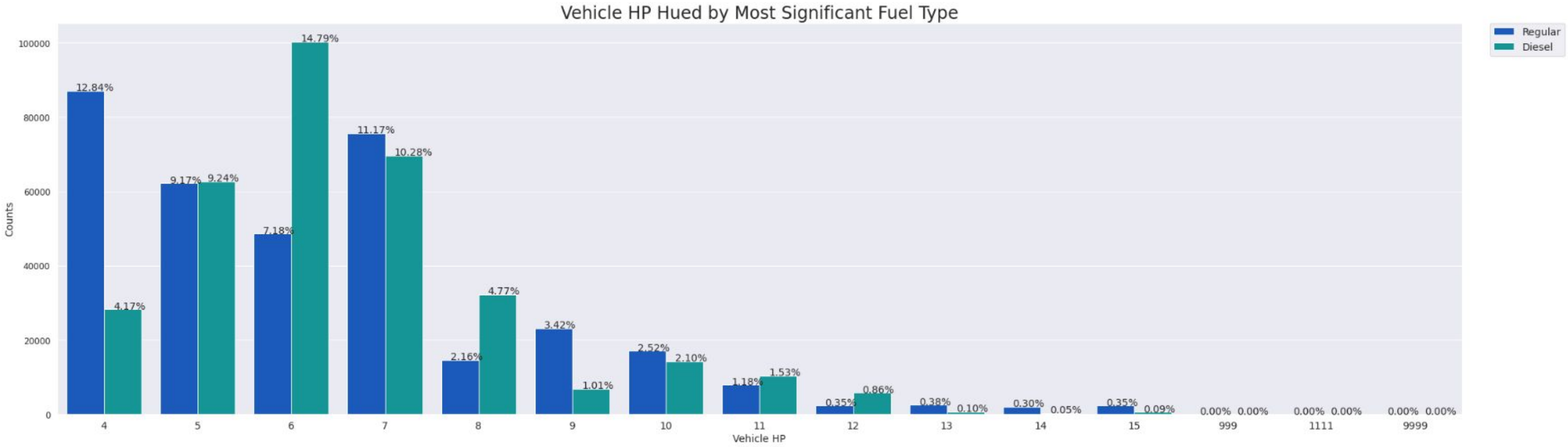


Bivariate EDA

	policy_desc	claim_count	num_vehicleAge	num_noClaimDiscountPercent	num_populationDensitykmsq	ord_vehicleHP	num_exposure	num_driverAge	num_PREMIUM_DISC	num_PREMIUM_INC	num_exposure_Months
policy_desc	1.000000	-0.127934	-0.156920	-0.005584	0.070751	0.002783	-0.127131	0.063696	-0.019385	-0.012801	-0.129044
claim_count	-0.127934	1.000000	-0.021770	0.050545	0.010757	-0.000676	0.055094	0.011390	-0.018478	0.049288	0.055288
num_vehicleAge	-0.156920	-0.021770	1.000000	0.079912	-0.090439	-0.000512	0.119968	-0.059206	0.017600	0.016707	0.121810
num_noClaimDiscountPercent	-0.005584	0.050545	0.079912	1.000000	0.077704	-0.001386	-0.145788	-0.479965	0.184404	0.374665	-0.145430
num_populationDensitykmsq	0.070751	0.010757	-0.090439	0.077704	1.000000	0.001269	-0.055912	-0.004688	0.020273	0.010292	-0.056289
ord_vehicleHP	0.002783	-0.000676	-0.000512	-0.001386	0.001269	1.000000	-0.002053	0.000289	-0.002200	-0.000703	-0.002022
num_exposure	-0.127131	0.055094	0.119968	-0.145788	-0.055912	-0.002053	1.000000	0.137534	-0.044023	-0.013605	0.999010
num_driverAge	0.063696	0.011390	-0.059206	-0.479965	-0.004688	0.000289	0.137534	1.000000	-0.198562	-0.079382	0.136557
num_PREMIUM_DISC	-0.019385	-0.018478	0.017600	0.184404	0.020273	-0.002200	-0.044023	-0.198562	1.000000	-0.289523	-0.043500
num_PREMIUM_INC	-0.012801	0.049288	0.016707	0.374665	0.010292	-0.000703	-0.013605	-0.079382	-0.289523	1.000000	-0.013719
num_exposure_Months	-0.129044	0.055288	0.121810	-0.145430	-0.056289	-0.002022	0.999010	0.136557	-0.043500	-0.013719	1.000000

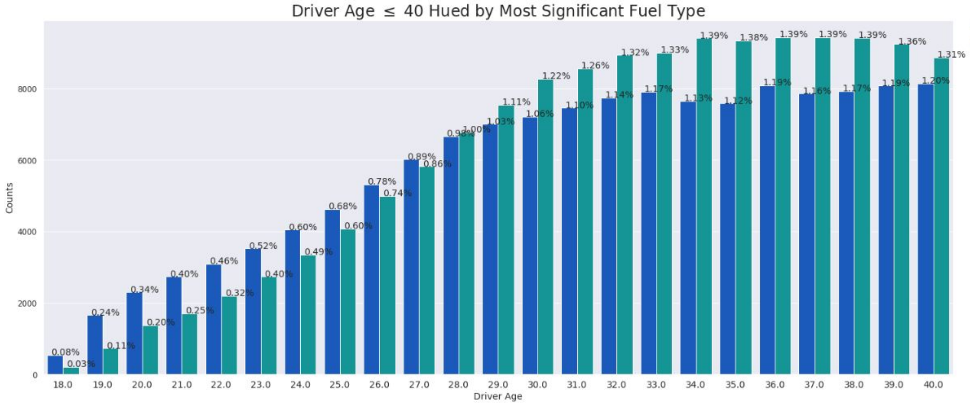
Bivariate EDA

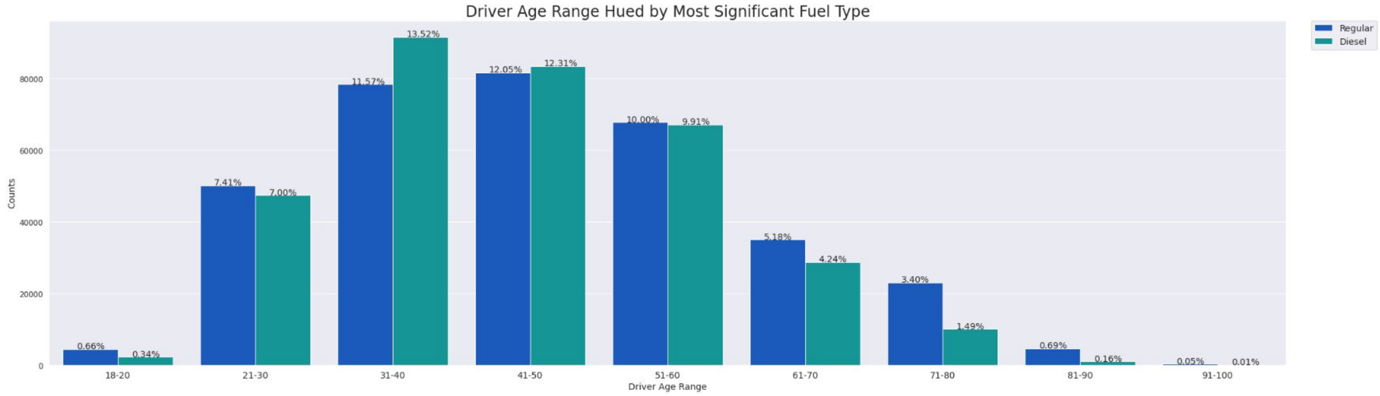




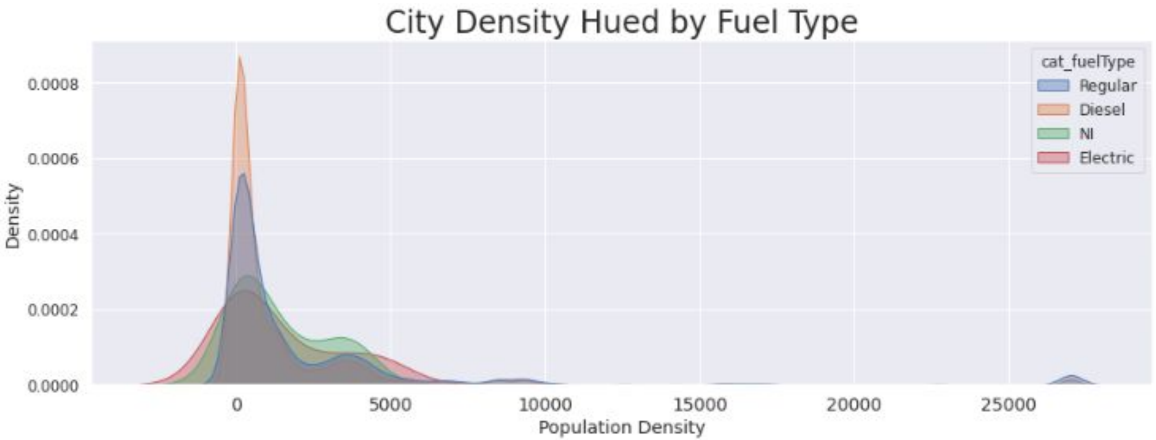
Liberty Insurance.

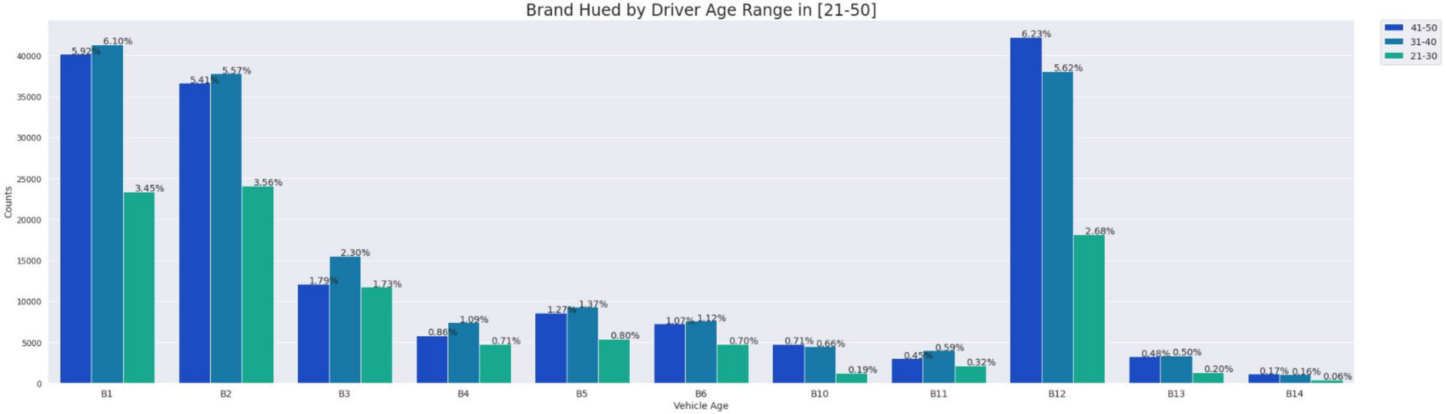
Bivariate EDA





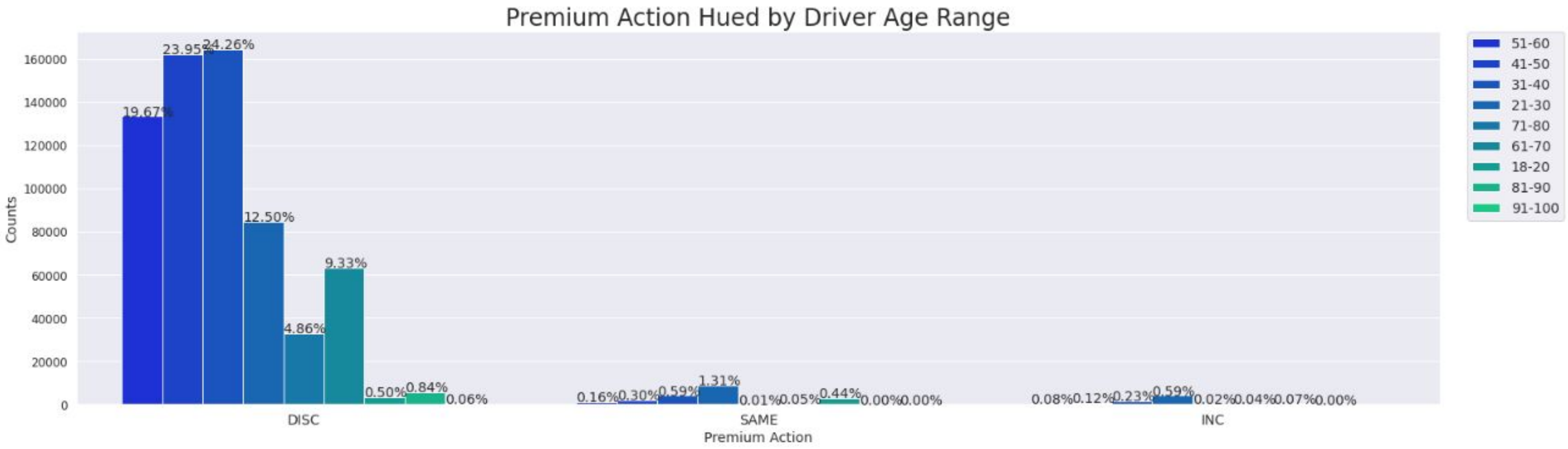
Bivariate EDA





Liberty Insurance.

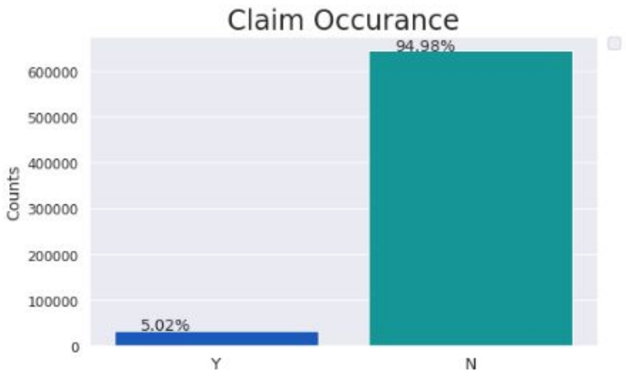
Bivariate EDA



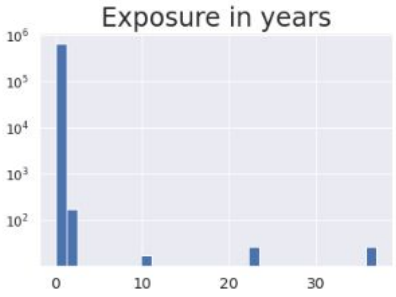
- ➔ Notice that in bivariate analysis the EDA could still spend so much time. So far, the bivariate analysis was performed using some basic key-insigths in way to gather some kind of information.
- ➔ What else could be done in Bivariate and Multivariate EDA?
  - ◆ The limit is the imagination! We could try scatter plots to verify if there's some kind of preferred direction say Drivers' Age VS Exposure?
  - ◆ Check the correlation between resources. Find out if there're correlated or anticorrelated behaviours (Probably both).
  - ◆ And so much more.
  - ◆ But, let's focus on the target analysis. And the correlation between the resources.

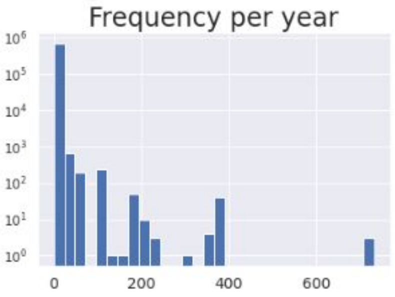
Research

- This isn't exactly a pricing model, however it's the first step towards one.
- What is known as loss cost model is simply predicting claim frequencies, meaning the rate of expected number of total claims by customer per unit of exposure time policy. The common trend of analyses in insurence predicting claim model shows that the distribution usually follows a Poissonian one. For this reason, the natural choice and most simple one is to start with a GLM with Poisson distribution and log-like funtion.
- On one hand, following this trend the Poisson distribution, by definition, has mean equals variance. On the other hand, in real data, this isn't a given. But, and there'a a big BUT here... Let's take it for granted! :D

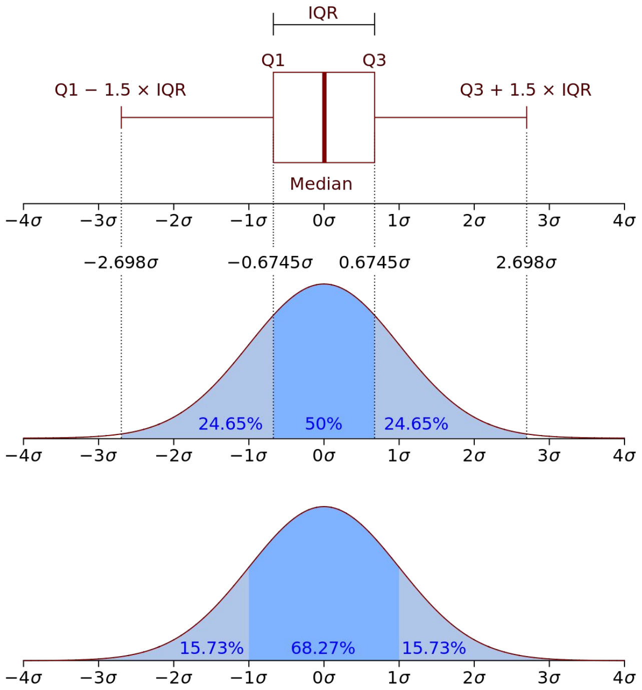


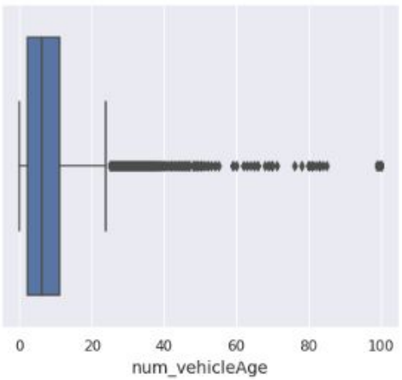
- Claim occurance is highly skewed, *i.e.* ~95% of policies don't have any claim at all.
  - Some possible reason to this behaviour usually might be associated with nonreporting claims.

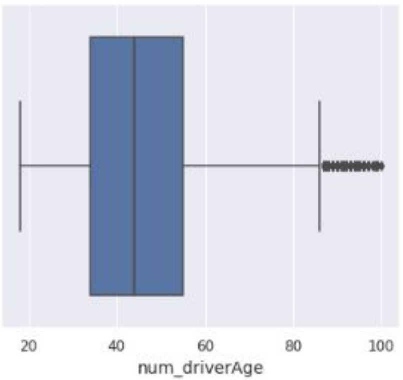




Check Outliers





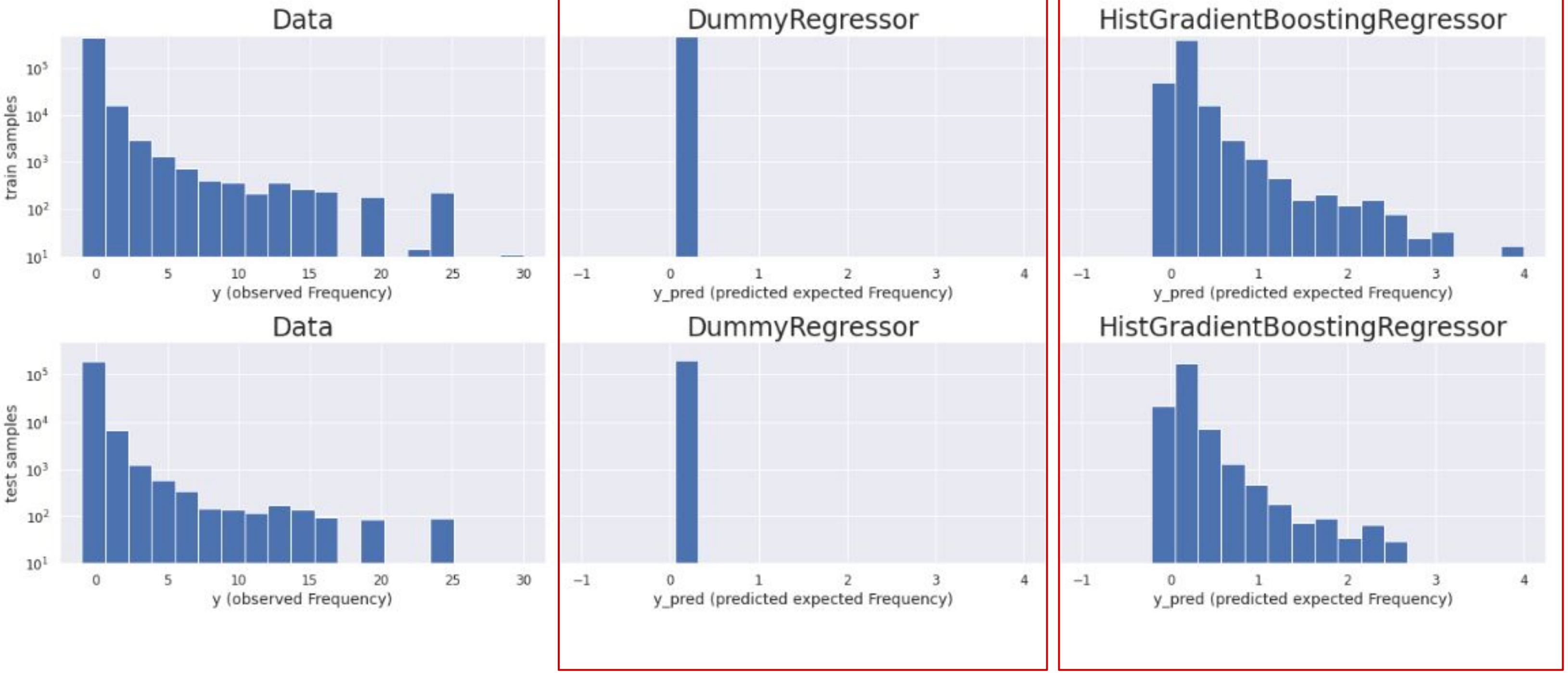


	PERCENTILE	FREEZED VALUE	COUNT
num_vehicleAge	99.9	33.0	433
num_driverAge	99.9	89.0	413

- ➔ The above figure is a very common way to search for outliers. This technique is called Turkey's Method and can be used to exclude data by setting as threshold the inner or outter fences:
  - ◆ Inner: 1.5 * Interquartile-Interval
  - ◆ Outter: 3.0 * Interquartile-Interval
- ➔ Another method could me by means of z-scores analysis, where z is defined by $(data_{i} + data_{mean})/data_{std}$
  - ◆ Z-scores simply means how far away we are from the mean value in a given feature.

Dummy Regressor
Gradient Boosting Regression Trees for Poisson Regression

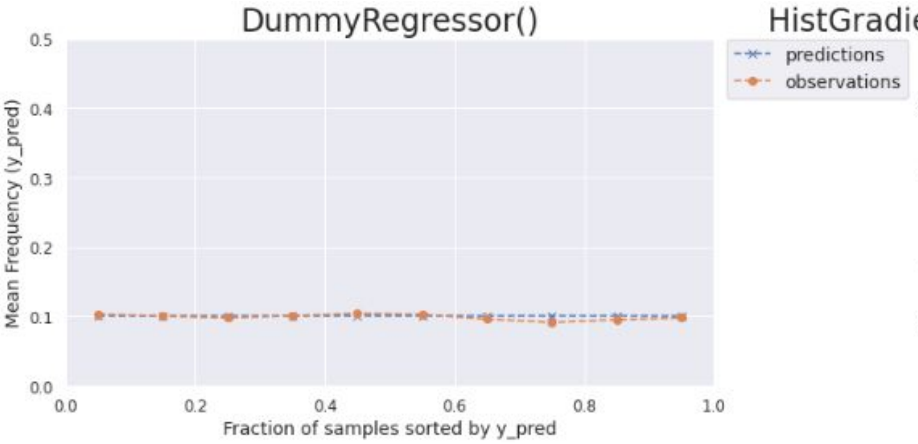
Modelling

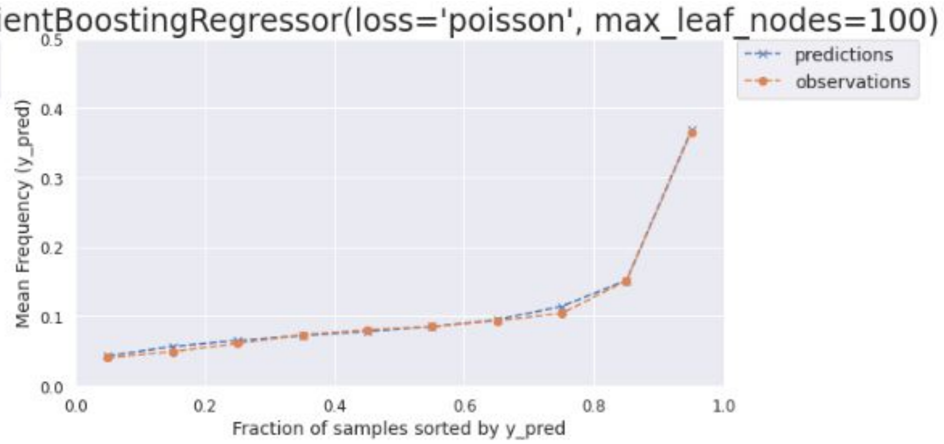


Liberty Insurance

Dummy Regressor
Gradient Boosting Regression Trees for Poisson Regression

Modelling





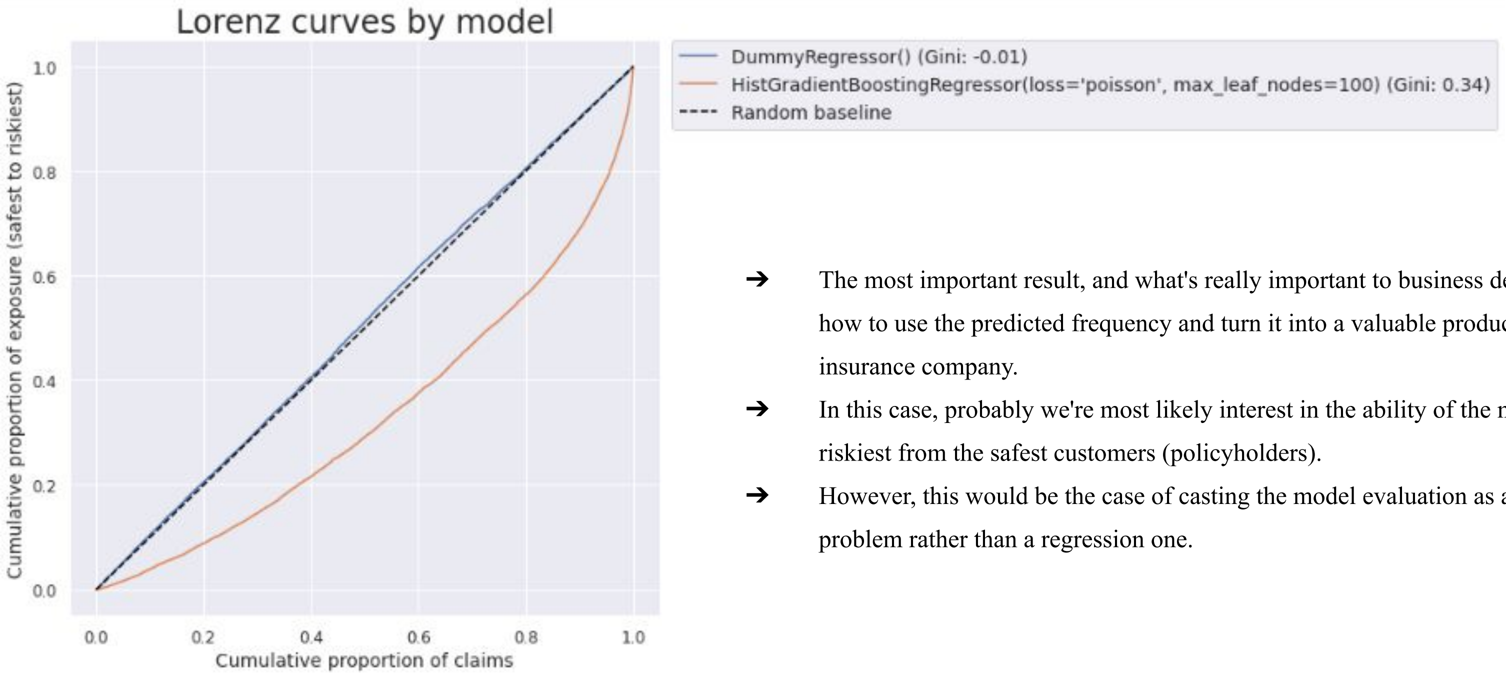
Meaning...?
Ok, but why is it so important?

➔ Dummy Regressor:
  - Predicts a constant frequency of claims by policy holder.

➔ Gradient Boosted Tree Regressor:
  - Shows a much better consistency between observed targets (our ground truth) and the predicted ones (our expectation).

Dummy Regressor
Gradient Boosting Regression Trees for Poisson Regression

Modelling



- ➔ The most important result, and what's really important to business decisions, is how to use the predicted frequency and turn it into a valuable product for tha insurance company.
- ➔ In this case, probably we're most likely interest in the ability of the model to rank riskiest from the safest customers (policyholders).
- ➔ However, this would be the case of casting the model evaluation as a ranking problem rather than a regression one.