MA-326 Project: Creating a pure premium pricing Model using GLMs

Shubham Singhal Madison Torres Jason Yin

December 2025

1 Introduction

In this project, we develop a pricing model that predicts auto insurance premiums using Generalized Linear Models (GLMs). Our approach involves decomposing pure premium into two distinct components: claim frequency and claim severity.

$$Pure\ Premium = Frequency * Severity \tag{1}$$

We model the frequency of claims using a Poisson GLM, which naturally handles count data and captures the expected number of claims per policy. For severity, we employ a Gamma GLM with a log link function to estimate the average cost per claim, given that a claim occurs.

This distribution is particularly well-suited for modeling positive, right-skewed claim amounts typical in insurance data. We incorporate a variety of predictors such as 'Age, no. of claims, claims history, etc. across policyholders.

To address the challenge of extreme outliers in claim amounts, we apply winsorization techniques which involves capping the positive severities at the 99.5th percentile to reduce outlier influence during model training. Our final pure premium estimates are obtained by multiplying the predicted frequency and severity for each policy, following the fundamental actuarial principle that expected cost equals expected frequency times expected severity.

We evaluate our models using multiple metrics including Mean Squared Error (MSE) and Mean Absolute Error (MAE),

$$MSE = \frac{1}{n}\Sigma(Y_i - \hat{Y}_i)^2 \tag{2}$$

$$MAE = \frac{1}{n}\Sigma|y_i - \hat{y}_i| \tag{3}$$

and assess calibration through decile-based analysis and visualization. We chose MSE as it penalizes large errors, which is helpful in insurance applications where large errors can cause significant financial payment. We chose MAE as it weighs all of the deviations linearly, allowing to measure the error without as

much influence from outliers. They provide a good measure of model accuracy and stability for the frequency and severity components. The dataset we work with contains cleaned motor insurance records with policy-level information, comprehensive driver and vehicle characteristics, and historical claims data.

2 Methodology

2.1 Overall Approach

We made a Generalized Linear Model (GLM) for frequency and another frequency. Combined, making predictions for each of these can be used with their product being the final prediction for pure premium.

2.2 Data Parsing

The data set[1] was in the form of semicolon separated values (csv format). We used the Pandas library to load the information into a pandas dataframe. Then, we normalized the data. For example, a column in the dataset was date of birth, so we converted it into years. For booleans we converted True to 1 and False to 0, and and for categorical variables we used one-hot encoding. Table 1 in the Appendix describes the types of data we used in the predictors.

2.3 Usage of scikit

In our project, we used statsmodels as our primary statistical toolkit for implementing generalized linear models (GLMs)[2]. Statsmodels automated the underlying mathematics and estimation procedures so we could focus on model design. For the frequency component, we utilized statsmodels.api.GLM[3] with a Poisson family and log link function to model claim counts, while for the severity component, we employed a Gamma family with a log link to model positive claim amounts.

2.4 Winsorizing

In our severity modeling approach, we employed a statistical technique called winsorizing to mitigate the influence of extreme outliers. Essentially, winsorizing caps extreme values at a specified percentile threshold rather than removing them entirely in order to reduce their disproportionate impact on model estimation. For example, a 90% winsorization would see all data below the 5th percentile set to the 5th percentile, and all data above the 95th percentile set to the 95th percentile.

While regularizers like Ridge or Lasso are powerful for preventing overfitting by penalizing large coefficients, we chose winsorizing instead because we were

[1] `https://data.mendeley.com/datasets/5cxyb5fp4f/2`
[2] `https://www.statsmodels.org/stable/examples/notebooks/generated/glm.html`
[3] `https://www.codecademy.com/resources/docs/python/statsmodels/glm`

addressing a fundamentally different problem: data-level outliers rather than parameter-level instability. So we were trying to make our data more 'nicer' for our severity model to provide reasonable estimates, rather having to adjust the severity model itself.

2.5 Mathematical Concepts: GLM

A Generalized Linear Model (GLM) goes past ordinary linear regression by allowing the response variable to follow a distribution from a chosen family and connecting the mean of this to a linear predictor through a link function. For our models, we chose the Poisson and Gamma distributions as described above.

We used GLMs to build a two-part frequency-severity model to predict auto insurance pure premiums. We fit a Poisson GLM using intercept-only design matrices to model claim counts, compute overdispersion, and generate predicted frequencies for the test set.

To generate predicted severities, we preprocessed data and winsorized outliers to fit a Gamma GLM to our model.

In the statsmodel, we used the .glm function to create these models, with the statistical model for each observation i assumed to be:

$$Y_i \tilde{F}_{EDM}(.|\theta, \phi, w_i)(4)$$

and

$$\mu_i = E[Y_i]|x_i] = g^{-1}(x`_i\beta) \tag{5}$$

where g is the link function and $\mathtt{F_{EDM}}(.|\theta, \phi, \mathtt{w_i})$ is a distribution of the family of exponential dispersion models (EDM). The density is

$$f_{EDM}(.|\theta, \phi, w_i) = c(y, \phi, w) \cdot exp(\frac{y\theta - b(\theta)}{\phi} w) \tag{6}$$

It also follows that

$$\mu = b'(\theta)\ Var[Y|x] = \frac{\phi}{w} b''(\theta) \tag{7}$$

The inverse of the first equation gives us the function of the expected value $\theta(\mu)$ such that

$$Var[Y_i|x)i] = \frac{\phi}{w_i} v(\mu_i) \tag{8}$$

giving $v(\mu) = b''(\theta(\mu))$, so the GLM is determined by the link function g and the variance function $v(\mu)$. Our specific Poisson function is received from μ, a variance of $\mu, \theta(\mu) = log(\mu)$, and $b(\theta) = e^{\theta}$. Our specific Gamma function is received from μ, a variance of μ^2, $\theta(\mu) = \frac{-1}{\mu}$, and $b(\theta) = -log(-\theta)$. [4]

Intuition: In a Generalized Linear Model, three fundamental components work together to describe the relationship between predictors and outcomes: the link function, the mean, and the variance. The link function g serves as a

[4] `https://www.statsmodels.org/stable/glm.html`

transformation that connects the linear predictor (input variables times coefficients) to the scale of the expected value, essentially translating between the model's mathematical framework and the real-world measurement scale.

The mean μ represents the expected value of the response variable, while the variance describes how spread out the observations are around this mean. An important mathematical property emerges from the exponential family structure: the mean is the first derivative of a function b with respect to the natural parameter θ (that is, $\mu = b'(\theta)$), and the variance is proportional to the second derivative of this same function ($Var[Y|x] = \frac{\phi}{w} b''(\theta)$).

This function b is indeed the cumulant generating function, which is simply the logarithm of the MGF. This makes sense since we are dealing with exponentials. Also, The scale parameter (φ) controls the overall variability or "noise" in our data. The natural parameter (θ) is a transformation of the distribution's mean that puts an exponential family distribution into its canonical form. For example, for poisson $\theta = log(\lambda)$ where λ is the mean.

2.6 Diagnostics

In order to get a better idea of what training data our severity model was going to be trained on, we decided to print out the number of zeros and non-zero severity amounts to see what might be affecting our model training. This allowed us to make the decision of winsorizing our data for the severity model.

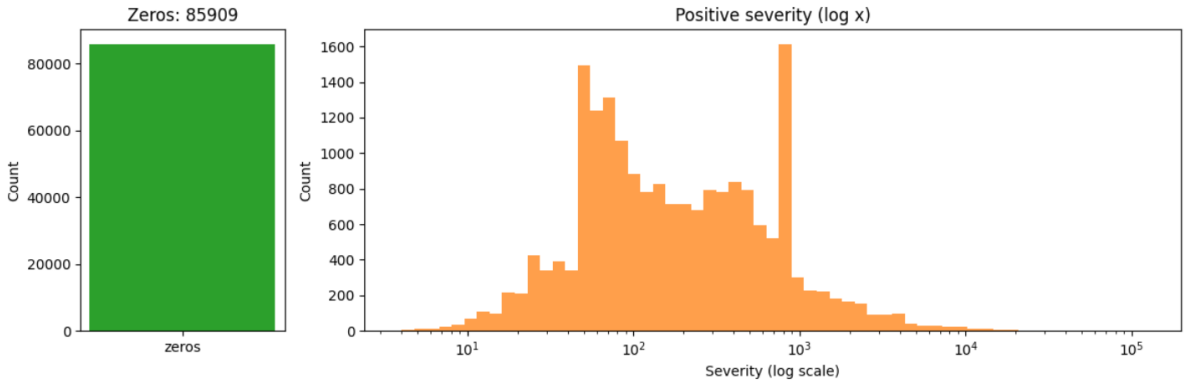

Figure 1: Severity data column from our dataset

2.7 Feature Selection

In our modeling approach for the frequency and severity GLMs, we initially selected the same core predictors for both models: Age,N_claims_history and R_Claims_history. It provides an indication of the policy's claims frequency history.), Power, and Value_vehicle.

For the frequency model, we expanded beyond this initial set by incorporating additional categorical features including Type_fuel, Area, Type_risk, Distribution_channel, and Second_driver. These categorical predictors were one-hot encoded to integrate them into the Poisson GLM framework.

In contrast, for the severity model, we retained only the original predictors. Although we initially considered categorical variables such as Type_fuel, Area, and Type_risk, we ultimately chose not to include them because we concluded these features did not provide meaningful explanatory power for claim severity once the numeric predictors were accounted for, and their inclusion risked overfitting due to the smaller sample size of policies with observed severity values.

2.8 Data Preparation

We split the datasets between training and testing data using a 40/60 train/test split. The frequency model uses the full dataset that was created with numeric and categorical variables converted to integers through binary dummy variables. The severity model uses a subset of the dataset that drops any rows that have missing predictors or severity due to severity only defined for policies with at least one claim. It follows the same procedure as the frequency model with using numerical and categorical variables and sets the severity variable as the dependent variable. We used a random_state equal to 42 so the random seed can help the results be reproducible.

2.9 Frequency Model

For the GLM Models, we started with the Frequency model. First, we converted the test and training arrays to NumPy arrays to avoid issues. We used the .glm function to create a generalized linear model object for the frequency with Poisson regression, since Poisson is for modeling frequency. We fitted the model to estimate the regression coefficients and printed out a summary of the results. Next, we computed overdispersion to verify if the Poisson assumptions hold. Overdispersion is the variance of the data and a check to see if this variance is larger than the variance predicted by the model indicating if the Poisson model is too restrictive, done by

$$(ResidualDeviance)/(DegreesofFreedom) > 1 \tag{9}$$

The Poisson assumptions are that the outcome is a count with non-negative integer values, the mean is equal to the variance, and that each policy is assumed to have independent claim counts. These assumptions should hold due to the claim occurrences being rare and a claim on one policy not affecting whether there is a claim on another. Therefore, overdispersion can tell us if the variance exceeded expectation and whether the Poisson model was correct before predicting on the test set. After this is verified, we went on to predict the Poisson mean predictions on the test set.

2.10 Severity Model

For the Severity model, we added an intercept column of 1's to the training and test matrices to allow for the model to shift. We then, converted the arrays to NumPy arrays like above to avoid issues.

We used a Gamma GLM for the severity, and since there can only be positive response values, we created a boolean mask selecting rows where the severity is greater than 0. We proceeded to winsorize the positive severities by finding the 99.5th percentile severity and capping the extreme outliers.

This is due to Gamma being sensitive to extreme values and winsizoring the reduces the influence of these outliers. The .glm function created the generalized linear model for the severity with Gamma regression and fit it using the positive winsorized severities and the predictors. We used this to predict on the test set.

2.11 Pure Premium Model

We originally tried the same approach by simply multiplying the results for our frequency and severity models together and make that our output pure_premium_pred variable. However, we ran into some issues about shapes not matching as well as not being able to compute the MSE or MAE. So, we had to flatten the arrays into 1-D arrays using .ravel().

When arrays aren't flattened to 1-D, three critical operations break: sklearn metrics interpret 2-D arrays as multi-output problems rather than single predictions, causing mean_squared_error() to either throw a ValueError about mismatched outputs or compute metrics across the wrong axis. Second, boolean masking creates 2-D boolean arrays when filtering (e.g., ys_obs > 0 produces shape (n, 1) instead of (n,)), which causes indexing errors or selects elements in unexpected ways.

Finally, matrix multiplication triggers numpy's broadcasting rules incorrectly, so $freq\_pred * sev\_pred$ with shapes (1000, 1) and (300, 1) might create a (1000, 300) matrix instead of element-wise multiplication, or raise a broadcasting error. The .ravel() call ensures all arrays are simple 1-D vectors with shape (n,), making sklearn interpret them as single-output predictions, boolean masks work as direct element selectors, and multiplication performs element-wise operations as intended.

3 Results

3.1 GLMs

These are our statistics for our Frequency and Severity GLMs:

Generalized Linear Model Regression Results

Dep. Variable:	y	No. Observations:	42222
Model:	GLM	Df Residuals:	42209
Model Family:	Poisson	Df Model:	12
Link Function:	Log	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-32091.
Date:	Tue, 02 Dec 2025	Deviance:	44584.
Time:	20:30:23	Pearson chi2:	5.93e+04
No. Iterations:	24	Pseudo R-squ. (CS):	0.3581
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.0313	0.061	-33.170	0.000	-2.151	-1.911
x1	-0.0068	0.001	-10.415	0.000	-0.008	-0.006
x2	0.0004	0.000	1.111	0.267	-0.000	0.001
x3	6.671e-06	1.33e-06	5.010	0.000	4.06e-06	9.28e-06
x4	0.0882	0.001	100.729	0.000	0.086	0.090
x5	0.3412	0.002	144.552	0.000	0.337	0.346
x6	-0.0381	0.019	-2.027	0.043	-0.075	-0.001
x7	-0.0111	0.018	-0.632	0.528	-0.046	0.023
x8	0.7486	0.056	13.462	0.000	0.640	0.858
x9	0.5632	0.053	10.628	0.000	0.459	0.667
x10	-23.5110	1.18e+04	-0.002	0.998	-2.32e+04	2.31e+04
x11	0.1949	0.017	11.811	0.000	0.163	0.227
x12	0.3219	0.022	14.634	0.000	0.279	0.365

Figure 2: Frequency GLM Statistics

Generalized Linear Model Regression Results

Dep. Variable:	y	No. Observations:	7858
Model:	GLM	Df Residuals:	7852
Model Family:	Gamma	Df Model:	5
Link Function:	log	Scale:	3.5348
Method:	IRLS	Log-Likelihood:	-56778.
Date:	Tue, 02 Dec 2025	Deviance:	13375.
Time:	20:30:23	Pearson chi2:	2.78e+04
No. Iterations:	14	Pseudo R-squ. (CS):	0.008905
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	6.0145	0.118	50.805	0.000	5.783	6.247
x1	-0.0004	0.002	-0.254	0.799	-0.004	0.003
x2	-0.0362	0.005	-8.008	0.000	-0.045	-0.027
x3	0.0269	0.022	1.227	0.220	-0.016	0.070
x4	0.0010	0.001	1.103	0.270	-0.001	0.003
x5	4.778e-06	3.61e-06	1.323	0.186	-2.3e-06	1.19e-05

Figure 3: Severity GLM Statistics

```
Frequency MSE: 168.67593956501122
Frequency MAE: 0.6345687757467177
Severity MSE: 2519750.6484724223
Severity MAE: 462.62006468727236
```

Figure 4: MAE and MSE for Models

The coefficient values indicate how impactful each factor is. A higher magnitude means that a factor matters is taken into account more in the GLM and thus has a higher impact on the resulting prediction.

For frequency, the top 3 highest magnitude coefficients (and thus most impactful) were x10 (-23.51), x8 (0.75), and x9 (0.56). These coefficients all correspond to the type of risk based on vehicle – in this case, x8 maps to vans, x9 to passenger cars, and x10 to agricultural vehicles. Vehicle type risks having the highest magnitude suggests that the type of vehicles involved in the crash are very important to the frequency. For example, -23.51 for agricultural vehicles indicates that if the vehicle is an agricultural one, it will be involved in less crashes. That makes sense because there are far less agricultural vehicles on the road (ex. tractors) than any other type. Outside of vehicle type risk, the next three highest magnitude coefficients belong to x5 (0.34), x12 (0.32), and x11 (0.19). The variable x5 corresponds to ratio of claims to policy duration,

x12 corresponds to second driver, and x11 corresponds to contract's distribution channel.

For severity, the magnitude coefficients in order from greatest to least are x2 (-0.036), x3 (0.027), x4 (0.001), x1(-0.0004), and x5 (4.8e-6). Those variables correspond to the total claims during the current year, ratio of claims to policy duration, vehicle horsepower, age, and market value of the vehicle. Here, total claims during the current year had the most impact. A correlation implied here is that drivers that get into many accidents will also be getting into more severe accidents.

In terms of Mean Squared Error (MSE) and Mean Absolute Error (MAE), the frequency model was much better than severity due to an abundance of zeros in our severity predictors resulting in it being very right-skewed.

3.2 Comprehensive Predictions

The following graph shows the our predicted pure premiums compared to the actual pure premiums.

Under the pure premium peformance cell, we computed the MSE and MAE respectively for our pure premium model. We got Pure Premium MSE: 7392818.33 and Pure Premium MAE: 293. MSE is not that great of an indicator for our model's validity as it is driven by extreme errors on high-value claims (squares amplify large errors). MAE is more fair indicator, the average absolute error of $293 per policy. If we assume avg. auto insurance premium to be roughly $1500, then our error is about 19.5% of the premium. This error might also be due to the fact that we are computing a 'pure premium', which is simply the premium required for the insurance to break-even so it would make sense that insurance policies would be higher in order for the insurance company to make a profit.

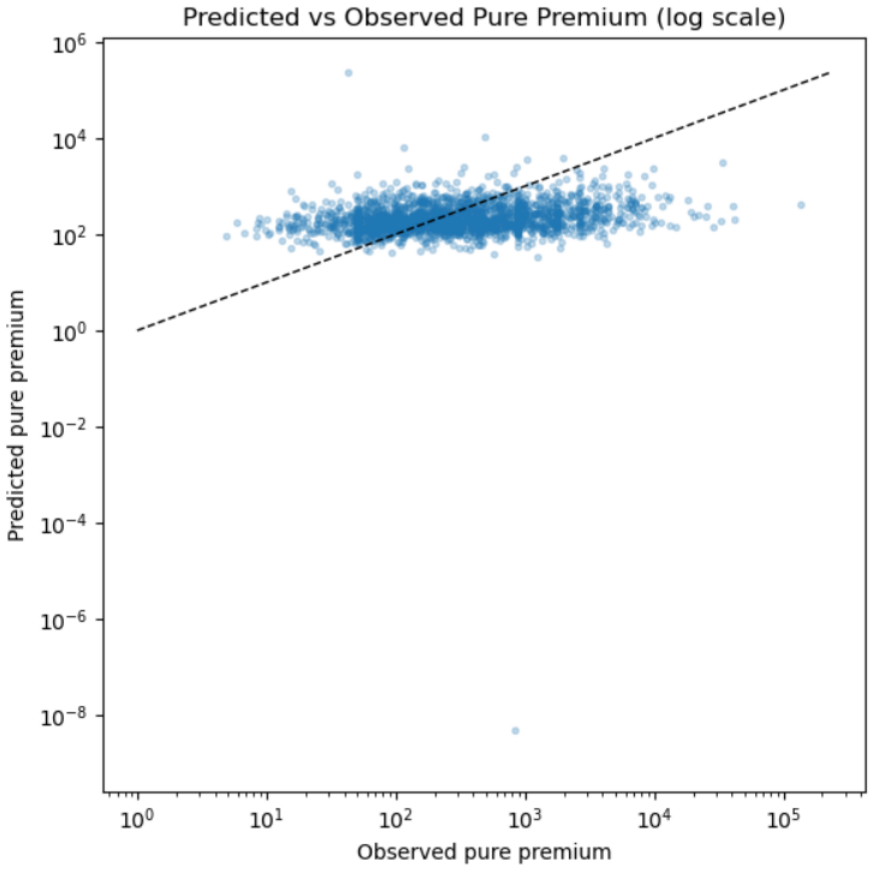

Figure 5: Predicted vs Actual Pure Premiums

4 Discussion

4.1 Possible Improvements

While our frequency model did fairly well as seen with it's low MAE and MSE, our severity model struggled despite winsorizing the data. We could have tried further winsorizing the data into a multi-threshold system: Cap at 95th percentile for high-risk segments, 99th for medium, 99.5th for low-risk.

We had also tried to implement other distributions such as the Tweedie Distribution: one can directly model the total loss with a unique Compound Poisson Gamma generalized linear model (with a log link function). This model is a special case of the Tweedie GLM with a "power" parameter $p \in (1, 2)$. See (`https://scikit-learn.org/stable/auto_examples/linear_model/plot_tweedie_regression_insurance_claims.html#`). We weren't able to get it working. We could also have tried finding categorical features with regularization for our severity model to allow for more exposure and potentially better results.

Appendices

Predictor	Description
Age	Calculated from Date_birth.
N_claims_history	Total number of claims filed throughout the entire duration of the insurance policy.
R_claims_history	Ratio of the number of claims filed for the specific policy to the total duration (whole years) of the policy in force. It provides an indication of the policy's claims frequency history.
Power	Vehicle power measured in horsepower.
Value_vehicle	Market value of the vehicle on 31/12/2019.
Type_fuel	Specific kind of energy source used to power a vehicle. Petrol (P) or Diesel (D).
Area	Dichotomous variable indicates the area. 0 for rural and 1 for urban (more than 30,000 inhabitants) in terms of traffic conditions.
Type_risk	Type of risk associated with the policy. Each value corresponds to a specific risk type: 1 for motorbikes, 2 for vans, 3 for passenger cars and 4 for agricultural vehicles
Distribution_channel	Classifies the channel through which the policy was contracted. 0 for Agent and 1 for Insurance brokers.
Second_driver	1 if there are multiple regular drivers declared, or 0 if only one driver is declared.

Table 1: Description of Predictors