Annals of Actuarial Science (2024), 1–31
doi:10.1017/S1748499524000058

ORIGINAL RESEARCH PAPER

Interpretable zero-inflated neural network models for predicting admission counts

Alex Jose[1,2], Angus S. Macdonald[1,2], George Tzougas[1,2] and George Streftaris[1,2]

[1]School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK; and [2]Maxwell Institute for Mathematical Sciences, Edinburgh, UK
Corresponding author: Alex Jose; Email: aj61@hw.ac.uk

(Received 04 August 2023; revised 12 December 2023; accepted 22 February 2024)

Abstract

In this paper, we construct interpretable zero-inflated neural network models for modeling hospital admission counts related to respiratory diseases among a health-insured population and their dependants in the United States. In particular, we exemplify our approach by considering the zero-inflated Poisson neural network (ZIPNN), and we follow the combined actuarial neural network (CANN) approach for developing zero-inflated combined actuarial neural network (ZIPCANN) models for modeling admission rates, which can accommodate the excess zero nature of admission counts data. Furthermore, we adopt the LocalGLMnet approach (Richman & Wüthrich (2023). *Scandinavian Actuarial Journal*, 2023(1), 71–95.) for interpreting the ZIPNN model results. This facilitates the analysis of the impact of a number of socio-demographic factors on the admission rates related to respiratory disease while benefiting from an improved predictive performance. The real-life utility of the methodologies developed as part of this work lies in the fact that they facilitate accurate rate setting, in addition to offering the potential to inform health interventions.

Keywords: Predictive modeling; neural network; actuarial; morbidity; zero-inflated neural network; admission rates

1. Introduction

Over the last few decades, one of the most imperative tasks of actuaries working within the life and non-life sectors is to construct models with high predictive ability which can efficiently capture the stylized characteristics of claim count and severity data. Even today, traditional regression models, particularly generalized linear models (GLMs) as introduced by Nelder & Wedderburn (1972), are one of the widely adopted approaches for handling such data sets. Instances of application of GLMs in an insurance context could be found in Ohlsson & Johansson (2010), De Jong *et al.* (2008), Haberman & Renshaw (1996), and Frees (2009). Furthermore, morbidity studies encompassing regression models under a Bayesian framework have also been considered by Arık *et al.* (2021), and Ozkok *et al.* (2014).

As far as modeling of hospital admission frequency data is concerned, which is the main focus of this study, it should be noted that one of the main issues associated with this and similar type of rare event count data has been the zero-inflated nature of the data leading to overdispersion. Although regression models with an underlying negative binomial distributional assumption could address the issue of overdispersion, they fall short of effectively addressing the issue of excess zeros (Gurmu & Trivedi, 1996). The problem of excess zeros magnifies once we start looking at cause-specific events. For instance, in our context, we are interested in admissions to hospitals or healthcare facilities, specifically due to respiratory diseases. Even though many individuals could

suffer respiratory diseases, admission to hospital implies a severe condition and could be less frequent. This highlights the difficulty and necessity for accurately modeling admission rates and analyzing the impact of different risk characteristics on the medical condition. In addition to the above-mentioned insurance context, numerous instances from other fields of research also deal with zero-inflated data. Irrespective of the area of research, for any studies based on rare events data, examples of which could be found in Ridout *et al.* (1998), the issue of excess zeros presents a challenge.

Different methodologies have been proposed to handle rare event data, out of which zero-inflated Poisson regression modes, proposed by Lambert (1992), and the hurdle model by Mullahy (1986), are two of the most popular approaches. The underlying principle behind both approaches is similar. In essence, both approaches set forth a mixture distribution with two components: the zero component and the count component. The zero component is used to model the zeros (in case of hurdle models) or the excess zeros (in case of zero-inflated models), and the count component is for the frequency data of the event of interest. Several variants and extensions have been developed ever since, and the application of the same under different contexts has also been considered. Lambert (1992) considered the application to defects in manufacturing, whereas Gurmu (1997) used a semi-parametric version of the hurdle model for Medicaid utilization. Famoye & Singh (2006) proposed a generalized variant of the zero-inflated model in the context of domestic violence data. In an actuarial setting, Yip & Yau (2005) used zero-inflated models for modeling insurance claim frequency data. Ridout *et al.* (1998) provide details regarding other similar instances. For additional information regarding the zero-adjusted models for handling data sets with excess zeros, we refer to the textbook of Cameron & Trivedi (2013).

The advancing field of deep learning, a subset of artificial intelligence, embodies a modern approach to designing and training neural networks (NNs) – computerized systems inspired by the human brain that learn from complex data to make predictions or decisions. This methodology has garnered remarkable successes in fields like computer vision, natural language processing, and speech recognition. Furthermore, it is attracting growing attention within the actuarial community, as evidenced by works both in the academic research field (Wüthrich & Merz, 2019; Denuit *et al.*, 2019; Wüthrich, 2020; Denuit *et al.*, 2021 and Wüthrich & Merz, 2023) and in the insurance business sector. One drawback of neural network (NN) models is the lack of explainability and interpretability of the results due to the inherent black box nature arising from their complex structure. Several techniques, such as SHapley Additive exPlanations by Lundberg & Lee (2017), LocalGLMet from Richman & Wüthrich 2023), locally interpretable model-agnostic explanation proposed by Ribeiro *et al.* (2016) etc., have been developed to address the interpretability issue. Out of these different approaches, here we adopt and extend the LocalGLMnet approach, owing to its ease of implementation and the likeness that it provides to interpretations derived from traditional regression models. Further criticism for NN models relates to the potential failure to maintain the balance property, thereby leading to bias at the population or portfolio level. (see Wüthrich, 2020, 2022). Several bias regularization approaches, such as those discussed in the aforementioned articles, have been proposed to address this problem.[1] Also, autocalibration, which was considered by Denuit *et al.* (2021), is a technique that can be employed to mitigate population-level bias, and for this reason, it has gained wide popularity recently.

The main focus of this study is to enhance the predictive performance of zero-inflated models utilized in the context of modeling admission rates associated with respiratory diseases in the US-insured population, characterized by a significant prevalence of zero counts. This will be achieved by constructing interpretable zero-inflated Poisson NN models capable of effectively analyzing count data, where there is a high prevalence of zero occurrences, as is the case with the data used

[1]The approach of Wüthrich (2020) has also been considered in the context of the data used in this study in our previous work Jose *et al.* (2022).

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

herein. This is important since accurate forecasting of hospital admission rates would assist hospitals in anticipating fluctuations in demand and thereby enhancing the overall quality of patient care. Furthermore, in addition to improving predictive performance, retaining interpretability is crucial for investigating the impact of covariates or input variables (see Table 2) on the admission counts.

The primary motivation for considering admissions-related respiratory diseases is that they remain one of the leading cause of mortality and morbidity. According to World Health Organization (WHO), worldwide, hundreds of millions of people suffer from preventable chronic respiratory diseases (CRDs) such as asthma, chronic obstructive pulmonary disease (COPD), and occupational lung diseases, and around four million deaths occur each year due to CRDs (Bousquet *et al*., 2007). Recent statistics indicate that each year 3 million people die from COPD, which comes to nearly 6% of all the deaths worldwide, and globally 262 million people suffer from asthma (WHO, 2022). The significance of having an efficient predictive model for analyzing the admissions related to respiratory diseases arises from the fact that most CRDs are preventable through early detection and intervention. This demands precise identification and understanding of significant risk attributes of respiratory diseases. The alternative models which we consider in this study facilitate more accurate modeling of admission rates compared to the ZIP model, while also analyzing the impact of different risk attributes on these diseases. In particular, the main contributions of this paper are outlined below.

- Firstly, following the CANN approach of Wüthrich & Merz (2019), we construct a Zero-inflated Poisson Combined Actuarial Neural network (ZIPCANN) model for modeling hospital admission frequency data. By embedding a zero-inflated Poisson regression into the CANN framework, we can explore the gains in predictive power compared to conventional regression models, and at the same time, we can accommodate the high presence of zeros in the count data. Additionally, it is worth noting that the CANN model and its extensions are part of the Residual Neural Network (ResNet) family, which employs skip connections, as discussed in He *et al*. (2016). These skip connections enable the model to address the vanishing gradient effects commonly associated with deep NNs.
- Secondly, we interpret the results of the ZIPNN model using the LocalGLMnet approach. It should be noted that interpreting results from a ZIP distributional assumption is more complex compared to the Poisson distributional assumption counterpart. This complexity arises due to the model's mixture nature, wherein input features are incorporated through the rate and probability parameters. Furthermore, the interpretations obtained from the NN-based model are compared to those derived with the coefficient estimates of the ZIP regression model. This comparison showcases the impact of potential non-linear interactions captured by the NN on the final results.

With the above contributions, we have taken into account all of the alternative approaches that have been developed in the recent actuarial literature concerning network-based models. For instance, Noll *et al*. (2020) and Gao *et al*. (2019) consider NN models in the context of motor insurance. Hejazi & Jackson (2016) propose a network-based approach for the valuation of large portfolios of variable annuities, while Kuo (2019) considers a deep-learning approach to loss reserving. Additionally, Hainaut (2018) adapts NNs for mortality forecasting. Richman (2021) provides a detailed review of recent advances of Artificial Intelligence in actuarial science. For more details regarding the application of NNs in an actuarial context, please refer to textbooks Denuit *et al*. (2019) and Wüthrich & Merz (2023). Moreover, it is worth noting that the above approaches and their combinations are used for the first time in the literature regarding the zero-inflated Poisson model for the case of hospital admission data.

The rest of this article is organized as follows: Section 2 provides a general description of the data, along with descriptive statistics and data considerations that were carried out prior to modeling. The different models considered as part of this work, i.e., the ZIP, ZIPNN, and ZIPCANN

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Table 1. Frequency of number of admissions related to respiratory diseases

No. of admissions	Frequency
0	2,046,167
1	3527
2	292
3	77
4	14
5	23

models, are discussed in Section 3. Details of the different hyper-parameter assumptions associated with NN model fitting are given in Section 4. Section 5 describes the LocalGLMnet approach and its extension for the ZIPNN model. A comparison of different models in terms of predictive performance and interpretations derived is given in Section 6. Finally, Section 7 contains the concluding remarks.

2. Data

The Merative (previously called IBM Watson Health) provides the admission data for the US population. The data set was constructed by combining the enrollment information and the admission details from different tables in the Commercial Claims and Encounters Database of Merative MarketScan Research Databases. The data contain individual-level demographic and employment-related information sourced from various employers and health plans around the US. The International Statistical Classification of Diseases and Related Health Problems 10th Revision – Clinical Modification (ICD-10-CM) codes were used for defining the underlying cause of admission facilitating the categorization of the admissions based on the disease (CDC, 2016). Also, multiple admission records of an individual due to a particular disease within a period of two days were treated as a single admission. As our focus is predominantly on the working population, we consider individuals in the age range [30, 65] for the year 2016. Different data consideration steps were undertaken to create the final data set. For instance, a variable (INDSTRY) which contained information regarding the industry in which the individual is employed, was excluded due to a high proportion of missingness. For the rest of the data variables, only the complete cases are used as the proportion of missingness was less than 2%. Additionally, variables, such as HLTHPLAN and DATATYP, are excluded due to the high level of relationship with other variables. The HLTHPLAN variable indicates whether the employer or a health plan provides the data, and the DATATYP variable indicates whether the individual's plan is on a reimbursement or capitation basis. The HLTHPLAN variable is strongly associated with employment-related variables such as EECLASS and EESTATU, and the DATATYP variable is correlated with the PLANTYP variable, which details the type of health plan the individual is part of. A description of the various variables within the final data set is given in Table 2. Furthermore, the ceiling for the number of admissions for an individual was set as five to address the unusually high numbers of individual admissions, possibly due to the common healthcare data practices. The same data set was used in earlier work (Jose *et al.*, 2022), which contains a more detailed description of the above-mentioned data considerations and the preliminary exploratory analysis that was carried out. Here, the REGION variable is chosen over a more granular geographical variable EGEOLOC which was used in Jose *et al.* (2022). The final data set contains 2,050,100 records with 4513 admissions related to respiratory diseases (see Table 1). For a further in-depth understanding of the data set and its analysis, we refer the reader to earlier work by Jose *et al.* (2022).

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Table 2. Description of variables in the admission data set

Variable	Description	Comment	Categories
ENROLID	Unique ID for individual	ID variable	-
AGE	Age of the last birthday of the individual	$\in[30, 65]$	-
SEX	Gender of the individual	Factor w/2 categories	1:Male, 2:Female
UR	Urban/ rural indicator based on individual's residence	Factor w/2 categories	1:Rural, 2:Urban
REGION	Geographical region of residence	Factor w/5 categories	1: Northeast, 2:North Central, 3: South, 4: West, 5: Unknown
EECLASS	Employee classification	Factor w/9 categories	1:Salary Non-union, 2:Salary Union, 3:Salary Other, 4:Hourly Non-union, 5:Hourly Union, 6:Hourly Other, 7:Non-union, 8:Union, 9:Unknown
EESTATU	Status of employment	Factor w/9 categories	1:Active Full Time, 2:Active Part Time or Seasonal, 3:Early Retiree, 4:Medicare Eligible Retiree, 5:Retiree (status unknown), 6:Comprehensive Omnibus Budget Reconciliation Act (COBRA) Continuee, 7:Long-Term Disability, 8:Surviving Spouse/Depend, 9:Unknown
EMPREL	Relation to the primary beneficiary	Factor w/3 categories	1:Employee, 2:Spouse, 3:Child/Other
PLANTYP	Type of health plan individual is part of	Factor w/8 categories	2:Comprehensive Plan, 3:Exclusive Provider Organization Plan, 4:Health Maintenance Organization Plan, 5:Non-Capitated (Non-Cap) Point-of-Service, 6:Preferred Provider Organization Plan, 7:Capitated (Cap) or Partially Capitated (PartCap) Point-of-Service Plan, 8:Consumer-Driven Health Plan, 9:High-Deductible Health Plan

2.1 Data pre-processing

Prior to carrying out any modeling, two data pre-processing steps were undertaken, the details of which are outlined below.

(1) One-hot encoding: for all categorical variables with more than two levels, one-hot encoding was applied. One-hot encoding represents a categorical variable with l categories $c_1, c_2, \ldots, c_l$ using a l dimensional feature vector which is of the form

$$x_j \mapsto \left(\mathbb{1}_{\{x_j=c_1\}}, \ldots, \mathbb{1}_{\{x_j=c_l\}}\right)^T \in \mathbb{R}^l. \tag{1}$$

(2) Min-max scaler: a min-max scaler which transforms the variable to a $[-1, 1]$ scale was implemented for the numerical and binary variables. The transformation was applied using the formula

$$x_j \mapsto x_j^* = \frac{2(x_j - m_j)}{M_j - m_j} - 1 \in [-1, 1] \tag{2}$$

where m_j and M_j represent the minimum and maximum values of variable x_j.

A more detailed description of the application of both one-hot encoding and min-max scaler on the admission data set under consideration is given in Jose *et al.* (2022). Following the data pre-processing steps, a 90:10 random split of the entire data set was created to be used as the learning $\mathcal{D}$ and testing $\mathcal{T}$ data sets. All the models were fitted using the learning data set, and the performance of the models on the testing data set was used to compare their performances.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

3. Models

The main models discussed as part of this work are the ZIPNN and ZIPCANN models, along with the extension of the ZIPNN model developed by incorporating the LocalGLMnet approach, which facilitates the interpretation of the results. A zero-inflated Poisson (ZIP) regression model is also considered for comparing the predictive performance between traditional regression modeling and network-based approaches. Furthermore, although not explicitly described here, instances of traditional Poisson regression, NN, and CANN models as detailed in Jose *et al.* (2022) are also considered for comparison. Before extending the LocalGLMnet approach to ZIPNN model, the same was implemented for the NN model as well.

3.1 Zero-inflated Poisson regression

As mentioned earlier, zero-inflated Poisson models are based on a mixture distribution comprising two components (Lambert, 1992; Cameron & Trivedi, 2013). In practice, a count distribution such as the Poisson or negative binomial distribution is used to represent the count component, and a Bernoulli distribution is assumed for the zero component. Hence a zero-inflated Poisson (ZIP) regression model is of the form

$$Y_i \sim \begin{cases} 0 & \text{with probability } \pi_i \\ Poisson(\lambda_i e_i) & \text{with probability } (1-\pi_i) \end{cases} \quad \dots \text{ for } i = 1, \dots, n \tag{3}$$

with $\mu_i = \lambda_i e_i$ being the mean of the Poisson part of the ith record with exposure e_i and rate parameter λ_i, while π_i represents the probability of only having zero admissions. Zeros arise from both the zero component and the count component, with the two components having probability π_i and $(1-\pi_i)$, respectively. Hence, the probability mass function of the ZIP mixture distribution is

$$Pr(Y_i = y_i) = \begin{cases} \pi_i + (1-\pi_i)e^{-\mu_i}, & y_i = 0 \\ (1-\pi_i)\frac{e^{-\mu_i}\mu_i^{y_i}}{y_i!} & y_i > 0 \end{cases} \tag{4}$$

and the corresponding mean and variance are given by

$$E(Y_i) = (1-\pi_i)\mu_i, \qquad V(Y_i) = (1-\pi_i)\mu_i(1+\pi_i\mu_i). \tag{5}$$

The ZIP model allows both μ_i and π_i to be modeled using a set of covariates. This would be of the form :

$$\log(\mu_i) = o_i + \beta_0 + \boldsymbol{\beta}_{reg}^{\top} x_i = o_i + \beta_0 + \langle \boldsymbol{\beta}_{reg}, x_i \rangle \tag{6}$$

and

$$logit(\pi_i) = \gamma_0 + \boldsymbol{\gamma}^{\top} w_i \tag{7}$$

where $o_i = \log(e_i)$ is the offset term, β_0, γ_0 the intercept terms and $\{\boldsymbol{\beta}_{reg}^{\top}, \boldsymbol{\gamma}^{\top}\} = (\beta_1, \dots, \beta_q, \gamma_1, \dots, \gamma_s)$ being the unknown vector of coefficients to be estimated corresponding to the sets of covariates $x_i = (x_{i,1}, \dots, x_{i,q})^{\top}$ and $w_i = (w_{i,1}, \dots, w_{i,s})^{\top}$ considered in the two regression functions with dimensions $q \times 1$ and $s \times 1$, respectively. Regarding the treatment of exposure within the ZIP model, it is possible that the exposure or period at risk could potentially influence the probability of admission, as well as the rates of admission. Consequently, the exposure could be factored into both components, as detailed in Feng (2022). Here, we follow the general practice of treating it as an offset term in the regression function for μ, as discussed in Lee *et al.* (2001).

The model selection procedure was carried out for both count and zero components of the ZIP regression model. All variables entered the model for the count component, and thus, a full model with all the covariates was utilized for the Poisson regression function. Similarly, for

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Table 3. Summary of the variable selection process of the logistic component of the ZIP regression model

Model	Degrees of freedom	BIC
Step 1: ∼1		
Null model	34	53,847.78
+ AGE	35	**53,829.5**
+ REGION	38	53,890.38
+ SEX	35	53,860.14
+ UR	35	53,862.2
+ EECLASS	42	53,945.26
+ EESTATU	42	53,925.05
+ EMPREL	36	53,867.1
+ PLANTYP	41	53,910.87
Step 2: ∼AGE		
+ REGION	39	53,873
+ SEXRT	36	53,840.2
+ URRT	36	53,843.82
+ EECLASS	43	53,927.63
+ EESTATU	43	53,914.32
+ EMPREL	37	53,844.43
+ PLANTYP	42	53,898.87

the regression function associated with the zero component, a forward step-wise variable selection process based on the Bayesian information criterion (BIC) was carried out(Vrieze, 2012). As we are using logistic regression to model the probability parameter, we refer to it as the logistic component from here on. The summary of the variable selection process of the logistic component of the model is given below in Table 3. Under the forward step-wise variable selection process, we start from a model without any terms in the logistic component and consider adding one variable at a time. The outcome of any particular step is the model that yields the lowest BIC value, and the whole process is repeated until there is no further reduction in the BIC value. The model thus identified from the variable selection process had the vector of coefficients $\{\beta_0, \boldsymbol{\beta}_{reg}, \gamma_0, \boldsymbol{\gamma}\} = (\beta_0, \beta_1, \ldots, \beta_q, \gamma_0, \gamma_1)$ corresponding to the complete set of covariates as presented in Table 4. The ZIP regression model was fitted using the `zeroinfl()` function in the `pscl` package which employs the `optim()` function to estimate the model parameters in both components simultaneously, by maximum likelihood, using optimization algorithms such as Nelder-Mead or a quasi-Newton method (Zeileis *et al.*, 2008).

3.2 Zero-inflated Poisson neural network (ZIPNN) model

A generic feed forward NN comprises an input layer, followed by multiple hidden layers and then the output layer. The structure of a feed forward NN is first discussed using a NN model with an underlying Poisson distribution assumption (model 3 in Table 5). A feature space $\mathcal{X}$ is taken as the input layer with dimension q_0. Assuming a network architecture of $d \in \mathbb{N}$ hidden layers with $q_m \in \mathbb{N}, 1 \leq m \leq d$ neurons in each of the layers then a neuron $z_j^{(m)}, 1 \leq j \leq q_m$, in the m^{th} hidden layer $\mathbf{z}^{(m)}$ is given by

$$z_j^{(m)}(\mathbf{z}) = \psi\left(\langle \boldsymbol{\beta}_j^{(m)}, \mathbf{z} \rangle, \right) \tag{8}$$

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Table 4. List of covariates and the corresponding coefficient parameters in both the components of the ZIP regression model

Covariate	Coefficient	Description
Count component		
Intercept	β_0	$\beta_{intercept}$
AGE	β_1	β_{age}
REGION	$\{\beta_2, \ldots, \beta_6\}$	β_{region_a} $a = 1, \ldots, 5$
SEX	β_7	β_{sex}
UR	β_8	β_{ur}
EECLASS	$\{\beta_9, \ldots, \beta_{17}\}$	$\beta_{eeclass_b}$ $b = 1, \ldots, 9$
EESTATU	$\{\beta_{18}, \ldots, \beta_{26}\}$	$\beta_{eestatu_c}$ $c = 1, \ldots, 9$
EMPREL	$\{\beta_{27}, \ldots, \beta_{29}\}$	β_{empred_d} $d = 1, \ldots, 3$
PLANTYP	$\{\beta_{30}, \ldots, \beta_{37}\}$	$\beta_{plantyp_e}$ $e = 1, \ldots, 8$
Logistic component		
Intercept	γ_0	$\gamma_{intercept}$
AGE	γ_1	γ_{age}

Table 5. Predictive performance of Poisson and ZIP regression models and network-based models with and without an additional layer for interpretation based on NLL. The empirical mean of the observed data is 0.0027

Model no.	Model	Learning loss	Testing loss	Average fitted mean
Model 1	Pois.reg	28,524.2	3016.8	0.0027
Model 2	ZIP.reg	26,662.3	2822.2	0.0028
w/o attention layer				
Model 3	NN(20,15,10)	28,202.3	3010.5	0.0026
Model 4	CANN(20,15,10)	28,220.6	3005.1	0.0025
Model 5	ZIPNN(20,15,10)	26,364.5	2808.3	0.0027
Model 6	ZIPCANN(20,15,10)	26,399.7	2809.4	0.0027
w/ attention layer				
Model 7	NN(20,15,10)	28,217.9	3000.2	0.0024
Model 8	ZIPNN(20,15,10)	26,424.2	**2807.8**	0.0026

with $\mathbf{z}^{(m)}$ represented as

$$\mathbf{z}^{(m)}\colon \mathbb{R}^{q_{m-1}} \to \mathbb{R}^{q_m}, \quad \mathbf{z} \mapsto \mathbf{z}^{(m)}(\mathbf{z}) = (1, z_1^{(m)}(\mathbf{z}), \ldots, z_{q_m}^{(m)}(\mathbf{z}))^\top, \tag{9}$$

inclusive of the intercept component, where $\boldsymbol{\beta}_j^{(m)} = (\beta_{l,j}^{(m)})_{0 \le l \le q_{m-1}}^\top \in \mathbb{R}^{q_{m-1}+1}$ are the network parameters and $\psi\colon \mathbb{R} \to \mathbb{R}$, the activation function. The network parameters corresponding to the hidden layer $\mathbf{z}^{(m)}$ are the $\left(\boldsymbol{\beta}_1^{(m)}, \ldots, \boldsymbol{\beta}_{q_m}^{(m)}\right) \in \mathbb{R}^{q_m}$. The comprehensive set of network parameter is given by $\boldsymbol{\beta} = \left(\boldsymbol{\beta}_1^{(1)}, \ldots, \boldsymbol{\beta}_{q_d}^{(d)}, \boldsymbol{\beta}^{(d+1)}\right) \in \mathbb{R}^r$, where the dimension is $r = \sum_{m=1}^d q_m \left(q_{m-1} + 1\right) + (q_d + 1)$. The predictor of the NN obtained from the output layer is then of the form

$$(o_i, x_i) \mapsto \log\left(\mu_i^{NN}\right) = o_i + \langle \boldsymbol{\beta}^{(d+1)}, (\mathbf{z}^{(d)} \circ \cdots \circ \mathbf{z}^{(1)})(x_i) \rangle, \tag{10}$$

for $i = 1, \ldots, n$, where $\boldsymbol{\beta}^{(d+1)} \in \mathbb{R}^{q_d+1}$ are the weights associated with the output layer which connects the last hidden layer $\mathbf{z}^d$ to the output layer $\mathbb{R}_+$.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

For a ZIPNN model, the feature space $\mathcal{X}$ with dimension q is taken as the input layer, i.e., $q_0 = q$. The construction of ZIPNN models involves incorporating zero-inflation considerations through the establishment of a distribution layer within the NN. This distribution layer, referred to as a lambda layer[2] in machine learning terminology, is a component in a NN that facilitates the integration of custom operations by defining a function applied to input data. In particular, this functionality allows for the inclusion of specific computations not covered by standard layers. In the specific context of zero-inflated NNs, the distribution lambda layer is employed to integrate a zero-inflated distribution assumption into the NN, enabling the model to capture complex relationships in the data. Also, we take $q_d = 2$ with $\psi: f(x) = x$. In other words, a dense layer with two neurons without activation is defined before the distribution layer.

The output of these two neurons is then used to calculate the probability (π) and rate (λ) parameters of the zero-inflated Poisson distribution constructed using the distribution layer that follows. The distribution layer thus created has an underlying zero-inflated Poisson mixture distribution. The model output is then derived as the mean of the distribution given by

$$E(Y_i) = (1 - \pi_i^{zipnn})\mu_i^{zipnn} \tag{11}$$

where π_i^{zipnn} and μ_i^{zipnn} are the rate and probability parameters of the underlying zero-inflated Poisson distribution. As in the case of a ZIP model, the exposure is taken as an offset term in the count component of the ZIPNN model. An exponential transformation is applied to the sum of the offset term and the output from the neuron associated with λ. A sigmoid function is applied to the output from the neuron associated with π to restrict the value of π to $[0, 1]$ interval. The μ_i^{zipnn} is then given by

$$\log(\mu_i^{zipnn}) = z_1^{(d)}(\mathbf{z}) = \psi\left(\langle \boldsymbol{\beta}_1^{(d)}, \mathbf{z}\rangle\right) + o_i = \langle \boldsymbol{\beta}_1^{(d)}, \mathbf{z}\rangle + o_i \tag{12}$$

or equivalently

$$\log(\mu_i^{zipnn}) = o_i + \langle \boldsymbol{\beta}_1^{(d)}, (\mathbf{z}^{(d-1)} \circ \cdots \circ \mathbf{z}^{(1)})(x_i)\rangle. \tag{13}$$

Similarly, π_i^{zipnn} is of the form

$$logit(\pi_i^{zipnn}) = z_2^{(d)}(\mathbf{z}) = \psi\left(\langle \boldsymbol{\beta}_2^{(d)}, \mathbf{z}\rangle\right) = \langle \boldsymbol{\beta}_2^{(d)}, \mathbf{z}\rangle. \tag{14}$$

$\boldsymbol{\beta}_1^{(d)}$ and $\boldsymbol{\beta}_2^{(d)}$ represent the network weights associated with the two neurons in the last hidden layer corresponding to probability and rate parameters. In essence the Equations (6) and (7) are replaced by Equations (13) and (14). Alternatively, we could also define

$$logit(1 - \pi_i^{zipnn}) = z_2^{(d)}(\mathbf{z}) = \psi\left(\langle \boldsymbol{\beta}_2^{(d)}, \mathbf{z}\rangle\right) = \langle \boldsymbol{\beta}_2^{(d)}, \mathbf{z}\rangle. \tag{15}$$

Since the zero-inflated distribution is defined as a mixture distribution using the distribution layer, the models could also be constructed using the $p_i = (1 - \pi_i)$ parameter with ease. This means that instead of π, the probability of zero component, the distribution layer considers p, which is the probability of count component or, in other words, the probability of having a non-zero admission. The only difference is how the mixture distribution is defined. For ease of interpretation, while using the attention layers, we have adopted the latter approach. A schematic representation of a ZIPNN with 20,15,10,2 neurons (model 5 in Table 5) in each dense layer is shown in Fig. 1. Code for implementing the same is given in Listing 3 and additional details regarding the approach can be found in Dürr *et al.* (2020). The structure of the ZIPNN model in terms of the connection between layers, the shape of input and out of each layer, and the number of parameters

[2]For more details regarding the lambda layer, refer to https://keras.io/api/layers/core_layers/lambda/.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

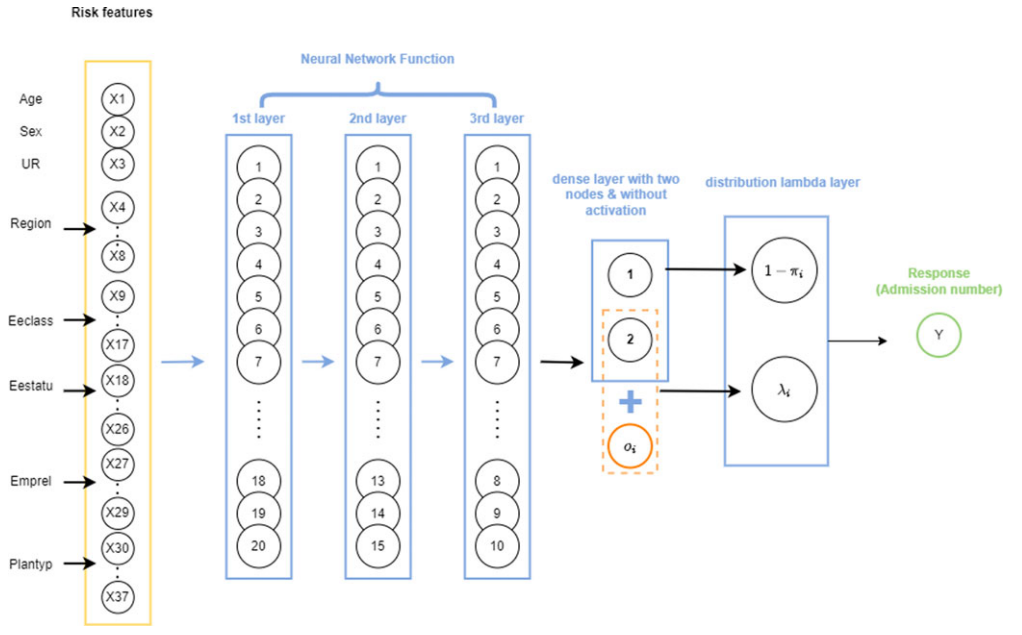

Figure 1. An illustration of a sample ZIPNN model with three hidden layers and 20,15,10,2 neurons in each layer.

Listing 1. Structure of the *ZIPNN* model.

```
Model: "ZIPNN model"
_________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
=================================================================================
 Design (InputLayer)         [(None, 37)]              0         []
 hidden1 (Dense)             (None, 20)                760       ['Design[0][0]']
 dropout_2 (Dropout)         (None, 20)                0         ['hidden1[0][0]']
 hidden2 (Dense)             (None, 15)                315       ['dropout_2[0][0]']
 dropout_1 (Dropout)         (None, 15)                0         ['hidden2[0][0]']
 hidden3 (Dense)             (None, 10)                160       ['dropout_1[0][0]']
 dropout (Dropout)           (None, 10)                0         ['hidden3[0][0]']
 Net (Dense)                 (None, 2)                 22        ['dropout[0][0]']
 LogVol (InputLayer)         [(None, 1)]               0         []
 concate1 (Concatenate)      (None, 3)                 0         ['Net[0][0]','LogVol[0][0]']
 distribution_lambda         (None, None)              0         ['concate1[0][0]']
 (DistributionLambda)
=================================================================================
Total params: 1,257
Trainable params: 1,257
Non-trainable params: 0
```

is shown in Listing 1. The final layer of the model is the distribution layer, which follows a zero-inflated Poisson distribution. The mean of the distribution is taken as the output or the response, which, in our case, is the admission count.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

3.3 Zero-inflated Poisson combined actuarial neural network (ZIPCANN) model

The ZIPCANN model contains additional skip connections, as in the case of a CANN model. In a CANN model (model 4 in Table 5), a skip connection connects the input features directly to the output layer. In effect, the predictor of a CANN model contains an additional regression function compared to the predictor of a NN and is of the form

$$(o_i, x_i) \mapsto \log(\mu_i^{CANN}) = o_i + \langle \boldsymbol{\beta}^{reg}, x_i \rangle + \langle \boldsymbol{\beta}^{(d+1)}, (\mathbf{z}^{(d)} \circ \cdots \circ \mathbf{z}^{(1)})(x_i) \rangle, \tag{16}$$

with the parameters from the regression function or skip connection represented by $\boldsymbol{\beta}^{reg}$ (Schelldorfer & Wuthrich, 2019; Jose *et al*., 2022). As detailed in Schelldorfer & Wuthrich (2019), various versions of CANNs exist depending on whether the weights in the regression part are updated or not whilst training the model. More specifically, one can either estimate the weights in the skip connection during training, or alternatively, keep the weights of the regression component fixed as the iterated weighted least squares estimates from the corresponding regression model. In this work, we adopt the former approach for all models containing skip connections.

For the ZIPCANN model, two skip connections are used; one for the rate parameter and one for the probability parameter. Instead of the output layer, the skip connection is made to the two neurons in the last hidden layer. In order to be consistent with the ZIP model, only the age variable was used in the skip connection associated with π, whereas all the features were used in the skip connection for μ. Consequently, the predictor for the ZIPCANN model is of the form

$$E(Y_i) = (1 - \pi_i^{zipcann})\mu_i^{zipcann} \tag{17}$$

where $\mu_i^{zipcann}$ is given by

$$\log(\mu_i^{zipcann}) = \langle \boldsymbol{\beta}^{reg}, x_i \rangle + \langle \boldsymbol{\beta}_1^{(d)}, \mathbf{z} \rangle + o_i \tag{18}$$

and $\pi_i^{zipcann}$ is given by

$$logit(\pi_i^{zipcann}) = \gamma_{age} x_i^{age} + \langle \boldsymbol{\beta}_2^{(d)}, \mathbf{z} \rangle \tag{19}$$

or alternatively,

$$logit(1 - \pi_i^{zipcann}) = \gamma_{age} x_i^{age} + \langle \boldsymbol{\beta}_2^{(d)}, \mathbf{z} \rangle. \tag{20}$$

Similarly to the ZIPNN model, the final output layer of the ZIPCANN model is also a distribution layer. The code for fitting the ZIPCANN model is given in Listing 4, and a diagrammatic representation of a sample model with one-hot encoding, skip connections, and 20,15,10,2 neurons in each of the layers (model 6 in Table 5), is shown in Fig. 2. The structure of the ZIPCANN model is given in Listing 5.

4. Model fitting

The ZIP regression model was fitted using `zeroinfl()` function in the `pscl` package in RStudio (Zeileis *et al*., 2008; Jackman, 2020; R Core Team, 2021; RStudio Team, 2021). The `zeroinfl()` utilizes the Nelder and Mead optimization method for estimating the model coefficients. For the network-based implementation of the ZIP model, we mainly employed the `tfprobability` package (Keydana, 2022; Dillon *et al*., 2017) in addition to the `keras` (Allaire & Chollet, 2021) and `tensorflow` (Allaire & Tang, 2021) packages used for constructing NN models. The `tfprobability` package allows to define probability distributions within a deep-learning model using a distribution layer. The main parts of the code used for building the network-based models are given in Appendix A.2.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

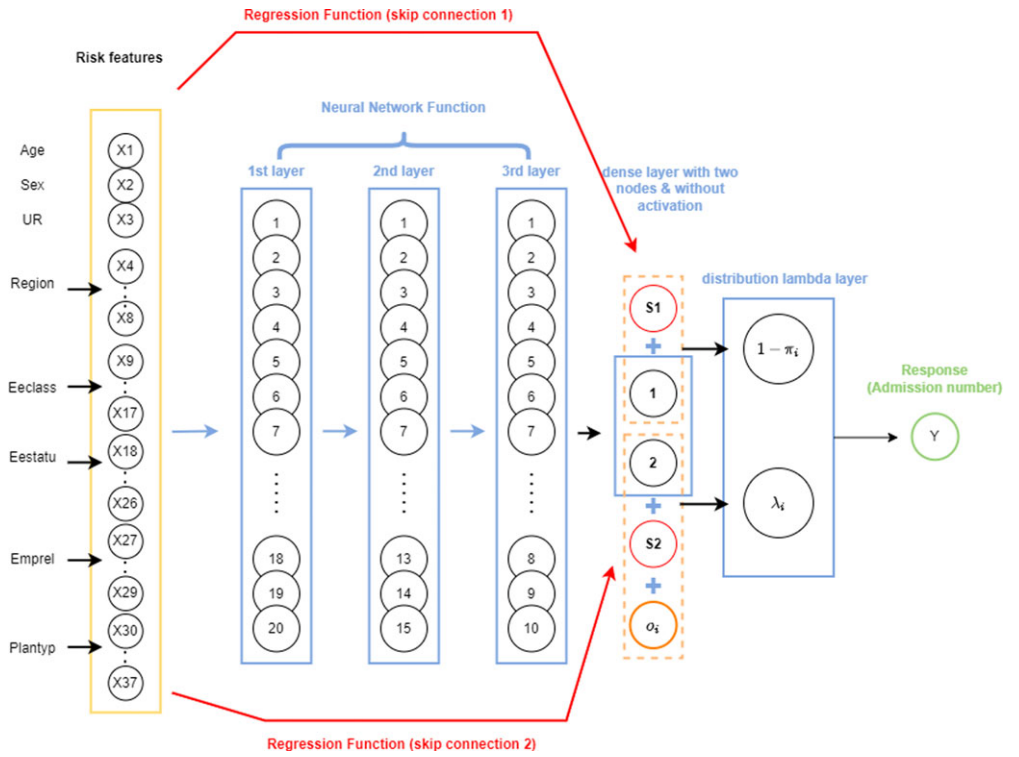

Figure 2. An illustration of a sample ZIPCANN model with skip connections, one-hot encoding, and 20,15,10,2 neurons in each of the layers.

4.1 Hyper-parameters

The fitting of NN models requires determining a number of associated hyper-parameters, including the depth of the network, choice of activation function, loss function, and gradient descent method. The hyper-parameter options identified and adopted in Jose *et al.* (2022) have also been used here, with details given below.

- Gradient descent method (GDM): the Nesterov-accelerated adaptive moment estimation (Nadam) method was used as the choice of the Gradient descent optimization algorithm for estimating the model weights.
- Network architecture: for NN and CANN models, a network architecture with three hidden layers with (20,15,10) neurons in each layer was considered. As mentioned earlier, the ZIPNN and ZIPCANN models have an additional layer with two neurons. In addition to the layers mentioned above, an attention layer is used to interpret the results from NN models (see Section 5). Also, for the ZIPNN and ZIPCANN models, alternate architectures with different numbers of neurons in the first three hidden layers were considered. The results, as given in Table A.1, indicate that the choice of (20,15,10) architecture has the best predictive performance compared to other architectures in the case of the ZIPNN model. Moreover, for the ZIPCANN model, the (35,25,20) architecture only minimally outperforms the (20,15,10) architecture. Hence, the combination of (20,15,10) neurons in the initial three hidden layers is adopted for both models.
- Batch size and epochs: a combination of 30,000 batch size and 500 epochs were used for fitting the NN models. The 30,000 batch size was identified as a reasonable choice in earlier work by

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Jose *et al.* (2022), and the large epoch was used since the early stopping approach implemented using the callback method will restrict the model from overfitting.

- Validation split: a further 80:20 split of the learning set was used as the training set, $\mathcal{D}^{(-)}$, and a validation data set $\mathcal{V}$.
- Loss function: the negative log-likelihood (NLL) was used as the objective function, which the GDM algorithm minimizes for estimating the model weights. The log-likelihood of ZIP mixture distribution is given by

$$l(\pi, \mu; y) = \sum_{i=1}^{n} u_i \ln\left(\pi_i + (1-\pi_i)\exp(-\mu_i)\right) + (1-u_i)\left[\ln(1-\pi_i) - \mu_i + y_i \ln(\mu_i) - \ln(y_i!)\right], \tag{21}$$

where $u_i = I(y_i = 0)$, μ_i is the mean of the Poisson component, π_i the probability parameter and y_i is the response variable. It is worth noting that in the case of the zero-inflated models, the saturated model reduces to the saturated version of the count component (Martin & Hall, 2016). This follows from the fact that a saturated model has the mean responses equivalent to the data, i.e., $E(y_i) = y_i$. For the zero-inflated models, this is true when we set $\mu_i = y_i$ and replace π_i with $u_i = I(y_i = 0)$ since $E(y_i) = (1-\pi_i)\mu_i$. Hence, the saturated model for ZIP mixture distribution will be a saturated Poisson distribution.

As previously mentioned, for the ZIPNN and ZIPCANN models, the output layer is a distribution layer (see Listing 1 and A5) with underlying Zero-inflated Poisson distribution and the mean of the distribution is taken as the model outcome or response which is the admission number. More precisely the distribution layer is considered as the model output. The NLL is the negative sum of the logarithm of the likelihood assigned to the observed count by the conditional probability distribution ($-\sum log(P(y|x, w))$). As the model's output is the distribution layer, we can obtain this by defining the loss function as shown in Listing 2.

Listing 2. Negative log-likelihood loss function used for training zero-inflated neural network models.

```
#negative log-likelihood loss function
nll<-function(y_true,y_pred)
{
  -y_pred$log_prob(k_reshape(y_true,shape = c(-1)))# k_shape for flattening
      the tensor
  #y_pred is the ZIP distribution layer
  #y_true is the observed admission count
}
```

Additionally, the early stopping and dropout model improvement approaches, as detailed in Jose *et al.* (2022), were used to avoid overfitting for all the network-based models.

5. Interpreting network-based models using the LocalGLMnet approach

The underlying principle of the LocalGLMnet approach, as proposed by Richman & Wüthrich 2023), is to use a NN for estimating the model coefficients of a GLM, as indicated by the name local generalized linear model network or LocalGLMnet. In this case, the coefficients $\boldsymbol{\beta}$ are replaced by feature-dependent non-linear functions $\boldsymbol{\beta}(x)$. The implementation of this approach involves creating an additional network layer, called the attention layer, with the same dimension as that of the feature space $q = q_0$, containing the so-called regression attention $\boldsymbol{\beta}(x)$. The regression attention terminology follows from its similarity to the attention weights proposed by Bahdanau *et al.*

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

(2014) and Vaswani *et al.* (2017). The motivation behind the attention weights is to give sufficient weight (attention) to the different features reflecting their significance. Richman & Wüthrich 2023) further extended this to develop LocalGLMnet to derive explanations from the results similar to a GLM. The key assumption behind this is the additive decomposition of the predicted value in terms of the features, as shown in Equation (22). In other words, the predicted value could be represented in terms of the weighted sum of features. Thus, in the case of a LocalGLMnet, the predictor is of the form

$$\begin{aligned} x_i \mapsto g(\mu_i) = g(\mu(x_i)) &= \beta_0 + \langle \boldsymbol{\beta}(x_i), x_i \rangle + o_i \\ &= \beta_0 + \left(\beta_1(x_i)x_{i1} + \ldots + \beta_q(x_i)x_{iq}\right) + o_i \end{aligned} \tag{22}$$

where

$$x_i \mapsto \boldsymbol{\beta}(x_i) = z^{(d:1)}(x_i) = \left(z^{(1)} \circ \ldots . \circ z^{(d)}\right)(x_i) \tag{23}$$

with $g()$ being the link function and μ_i being the mean of the distribution underlying the NN regression function. Interpretations are then obtained by considering the covariate contribution $\beta_j(x_i)x_{ij}$ associated with each of the features $x_{ij}, j = 1, \ldots, q_0$, obtained by extracting the component-wise product. The layer thus containing the component-wise product was termed the LocalGLM layer by Richman & Wüthrich 2023). As $\beta_j(x_i)$ depends on the features $x_i = (x_{i1}, \ldots, x_{i,j})$, $\beta_j(x_i)x_{ij}$ vary for each of the record and interpretations are derived for each feature x_j by looking at the $\beta_j(x_i)x_{ij}$ for $i = 1, \ldots, n$. Prior to extending the approach to the ZIPNN model, we implemented it for a NN model, as proposed by Richman & Wüthrich 2023). This highlights the merits of the LocalGLMnet approach, as well as its potential to be extended for more complex models. A diagrammatic representation of a sample LocalGLMnet with three hidden layers and (20,15,10) neurons in each of the layers (model 7 in Table 5) is shown in Fig. 3 and the code for implementing it is given in Listing 6.

To demonstrate the process of deriving interpretations we consider, as an example, the covariate contributions for the variables AGE and SEX. These are shown in Fig. 4, while the crude admission rates are shown in Fig. 5. Detailed analysis and interpretations for other covariate contributions is given later using the ZIPNN model.

The crude admission rates clearly show that the rates increase with age for both males and females. Also, the female population generally has higher rates compared to the male population. The same conclusion can be inferred from the covariate contributions. The median value of covariate contributions for females (SEX = 2) is higher than for males (SEX = 1), indicating higher admission rates. Similarly, the covariate contribution for the AGE variable implies an increasing trend of admission rates with age, with more variability at younger and older ages.

5.1 Interpreting the ZIPNN model

In order to interpret the results obtained from ZIPNN model, we extend the additive decomposition assumption to the μ and $p = (1 - \pi)$ parameters of the underlying ZIP mixture distribution as shown in Equation (4). The regression weights corresponding to each feature are calculated separately for μ and p. Hence, the dimension of the layer analogous to the LocalGLM layer with the component-wise product of regression weights and features is $2 \times q_0$. Among these $2 \times q_0$ weights, q_0 weights are associated with μ and the rest with p.

5.1.1 Regression attention for ZIPNN

The ZIPNN model, as defined in Section 3.2, consists of an input layer, multiple hidden layers, with the last hidden layer containing two neurons, followed by the distribution layer and the output layer. To create the interpretable ZIPNN model, the attention layer is introduced before

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

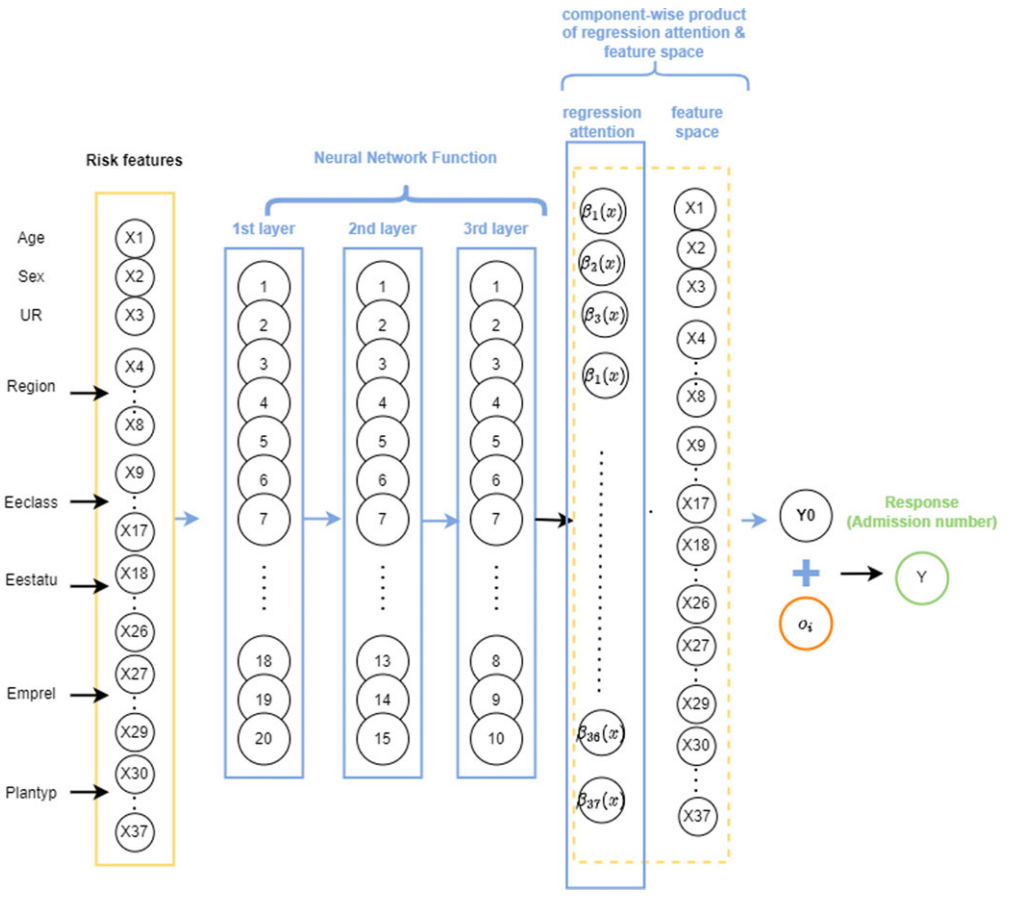

Figure 3. An illustration of a sample LocalGLMnet model with LocalGLM layer, one-hot encoding, and 20,15,10 neurons in each of the layers.

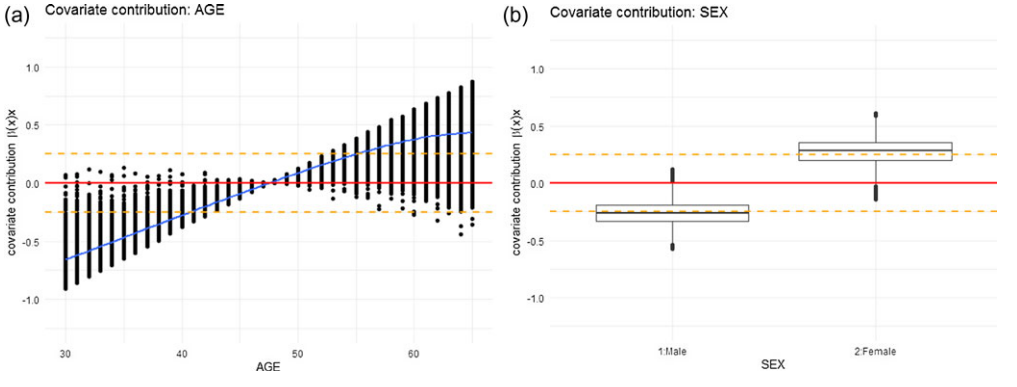

Figure 4. Graphical representation of covariate contribution from LocalGLMnet model for the testing set (a) age variable; (b) male, female; the blue line indicates a spline fit approximate curve and the yellow line has been added as a reference line at levels -0.25 and 0.25.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

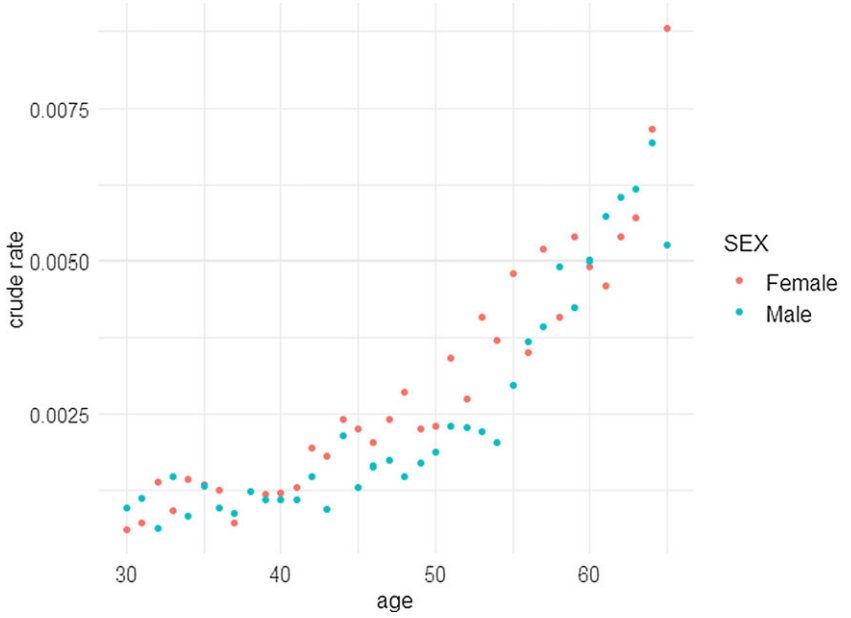

Figure 5. Age-wise crude rates of admission due to respiratory diseases for male and female patients for the entire data set.

the distribution layer, effectively replacing the last hidden layer of the ZIPNN model with two neurons (see Fig. 6).

Hence, the rate and probability parameters defined in Equations (13) and (14) are replaced by

$$\log(\mu_i^{zipnn}) = \beta_{0\mu} + \langle \boldsymbol{\beta}_{(q_0+1:2q_0)}(x_i), x_i \rangle + o_i \tag{24}$$

and

$$logit(p_i^{zipnn}) = \beta_{0\pi} + \langle \boldsymbol{\beta}_{(1:q_0)}(x_i), x_i \rangle \tag{25}$$

with $x_i \mapsto \boldsymbol{\beta}(x_i) = z^{(d:1)}(x_i) = \left(z^{(1)} \circ \ldots \circ z^{(d)}\right)(x_i)$. The attention weights $\boldsymbol{\beta}_{(1:q_0)}(x)$ and $\boldsymbol{\beta}_{(q_0+1:2q_0)}(x)$ are associated with parameters p and μ respectively. The interpretations or the impact of the different features on parameters μ and p are derived by analyzing the covariate contributions: $\beta_j(x_i)x_{ij}$ for p_i and $\beta_{q_0+j}(x_i)x_{ij}$ for μ_i, where $j = 1, \ldots, q_0$. Sample code for creating an interpretable ZIPNN model (model 8 in Table 5) is given in Listing 7. The structure of the interpretable ZIPNN model is given in Listing 8.

Although it is possible to create interpretable versions of both CANN and ZIPCANN models similar to those for the NN and ZIPNN models, the introduction of the attention layer nullifies one of the primary purposes of skip connections, which is to distinguish the main effects from the complex effects. This is because the attention layer will integrate the main effect represented by the skip connection and with the complex effect from the network. Hence, in this work, we do not discuss interpretable versions of the CANN and ZIPCANN models.

6. Comparison of models

In this section, we compare the different models discussed so far. The models are compared in terms of predictive performance and the insights obtained from the model results.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

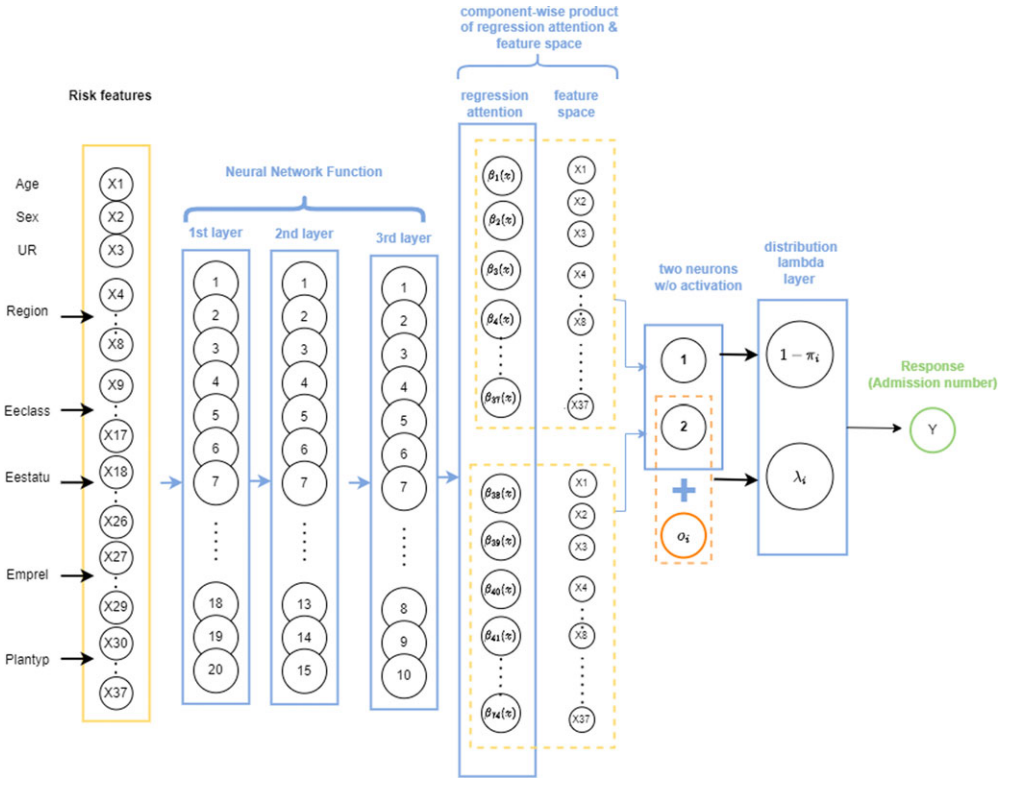

Figure 6. An illustration of an interpretable ZIPNN model with regression attention, one-hot encoding, and 20,15,10 neurons in each of the layers.

6.1 Predictive performance

The NLL was used to compare the predictive performance of the implemented models. While other comparison measures are available, such as the mean absolute error and the mean square error, these focus on the deviation of the predicted value from the actual observed value and do not fully account for the considered probability distribution when using a probabilistic model. Also, other alternative measures, such as the predicted probability of a zero count, fall short of providing a comprehensive comparison as they only consider the probability of admission as a dichotomous event, and fail to consider the admission rate. The NLL of a model has the form:

$$\mathcal{L}_{\mathcal{A}}(\boldsymbol{\beta}) = -\log\left(L_f\right) \tag{26}$$

where $\mathcal{A}$ represents the data set and L_f gives the likelihood of the fitted model (e.g. Equation (21) for the log-likelihood of a ZIP mixture distribution). Performance was compared based on the NLL $\mathcal{L}_{\mathcal{T}}$ of different models for the testing set $\mathcal{T}$. Values of the NLL for both the testing and training data sets, for all considered models, are shown in Table 5. Table 5 also contains the average fitted mean $\hat{\mu}$ for each model for the full data set. The difference between the average fitted mean of a model and that of the empirical mean of observed data indicates population-level bias.

The results indicate that the ZIPNN model has the lowest testing loss compared to other models. In general, the models with an underlying zero-inflated mixture distribution assumption performed better than those with a basic Poisson distribution assumption. The ZIPNN and ZIPCANN models performed better than the traditional ZIP regression model. Similarly, the NN

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

and CANN models performed better than the conventional Poisson regression model. As the difference between the testing loss of ZIPNN and ZIPCANN models is small, it is not possible to conclusively state the supremacy of one over the other due to the inherent randomness in NN results. The different attributes leading to the randomness in NN model results have been extensively discussed by Richman & Wüthrich (2020). Approaches such as nagging predictor and k-fold validation could be adopted to address this randomness, as done in previous work by Jose *et al.* (2022). More than the inherent randomness, the underlying architecture sways the model performance on a larger scale. Hence we analyzed this impact by considering different architectures of varying complexity. As shown in Table A.1, the combination of 20,15,10 neurons in the first three hidden layers is a suitable choice for both ZIPNN and ZIPCANN models.

6.2 Model interpretability

In order to compare the interpretations derived from the model results, we consider the coefficient estimates from the ZIP regression model and the covariate contributions from the ZIPNN model. The coefficient estimates ($\boldsymbol{\beta}_{reg}, \boldsymbol{\gamma}$) from the ZIP regression model are given in Table 6. For the ease of interpretation of the results, a sum-to-zero constraint was applied to the coefficient estimates of different levels of all the non-binary categorical variables.

The interpretation of coefficient estimates from a ZIP distributional assumption is more complex compared to a conventional regression model, because it refers to a mixture distribution. For the ZIP regression model considered here, the count component considers all features, whereas the zero component only considers the age variable. Additionally, in the case of the ZIPNN model, an attention layer is used for both λ and p parameters, which means that the covariate contributions of different variables for λ need to be considered in conjunction with those for the p parameter. This makes interpreting the results and comparison with ZIP regression model less straightforward. The comparison was carried out between the most significant coefficient estimates in the ZIP regression model and the corresponding covariate contribution using the ZIPNN model.

The plots showing the covariate contributions were produced using the test data. Figs. 7 and 7 (continued) show the covariate contributions for different variables associated with the rate (λ) parameter, while Figs. 8 and 8 (continued) show the covariate contributions for the p parameter. The horizontal yellow line in the plots was added as a reference line at levels -0.25 and 0.25. This can help to compare the magnitude of the covariate contributions and how they are associated with the parameters.

The analysis of the covariate contributions using the ZIPNN model for the AGE variable for both the probability and rate parameters (Figs. 7 and 8) indicates an increasing trend over age. This suggests a positive relationship showing that as age increases, both the probability of having an admission and the rate of admission increase. The spline fit (blue line) indicates an almost linear trend, with the greatest variability at younger and older ages. In both cases, this suggests that the impact of the age variable on the parameters varies at different ages. The variation appears to be lowest around the age of 47 before increasing for older ages as well. The highly significant positive coefficient estimate for the AGE variable in the count component of the ZIP regression model (see Table 6) indicates that the rate of admission increases with age. The negative estimate in the zero component associated with the π parameter also suggests that the probability of having excess zero decreases with age. In other words, the probability of having non-zero admissions increases with age.

The covariate contributions under the ZIPNN model for the SEX variable indicate a higher probability and rate of admission for females (SEX = 2) compared to males (Figs. 7 and 8). The significant positive coefficient estimate in the count component of the ZIP regression model also implies the same. This is in line with the pattern displayed by the crude rates (see Fig. 5). The low

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Table 6. Coefficient estimates based on the ZIP regression model with the significance codes ("***,""**,""*," ".,"" ") indicating the level of significance of the estimates at levels (0, 0.001,0.01, 0.05, 0.1, 1)

Coefficient	Estimate	Std. error	*z* value	Pr(> |*z*|)	Signi.
Count component					
$\beta_{intercept}$	−1.7956	0.3522	−5.0990	0.0000	***
β_{age}	0.0268	0.0054	4.9380	0.0000	***
$\beta_{region_1:northeast}$	−0.2113	0.1084	−1.9490	0.0513	.
$\beta_{region_2:northcentral}$	−0.0911	0.1070	−0.8520	0.3945	
$\beta_{region_3:south}$	−0.0959	0.1047	−0.9160	0.3598	
$\beta_{region_4:west}$	−0.5460	0.1109	−4.9230	0.0000	***
$\beta_{region_5:unknown}$	0.9444	0.4050	2.3320	0.0197	*
$\beta_{sex:female}$	0.1022	0.0363	2.8170	0.0048	**
$\beta_{ur:urban}$	−0.1491	0.0502	−2.9720	0.0030	**
$\beta_{eeclass_1:salary\ non-union}$	−0.2894	0.0530	−5.4610	0.0000	***
$\beta_{eeclas_2:salary\ union}$	−0.0385	0.1510	−0.2550	0.7985	
$\beta_{eeclass_3:salary\ other}$	−0.2526	0.0910	−2.7760	0.0055	**
$\beta_{eeclass_4:hourly\ non-union}$	0.1980	0.0537	3.6870	0.0002	***
$\beta_{eeclass_5:hourly\ union}$	0.2905	0.0579	5.0200	0.0000	***
$\beta_{eeclass_6:hourly\ other}$	0.2276	0.0875	2.6010	0.0093	**
$\beta_{eeclass_7:non-union}$	0.1227	0.0535	2.2920	0.0219	*
$\beta_{eeclass_8:union}$	−0.0624	0.0951	−0.6560	0.5117	
$\beta_{eeclass_9:unknown}$	−0.1958	0.0513	−3.8200	0.0001	***
$\beta_{eestatu_1:active\ full\ time}$	−0.3383	0.0650	−5.2010	0.0000	***
$\beta_{eestatu_2:active\ part\ time\ or\ seasonal}$	−1.2290	0.2237	−5.4950	0.0000	***
$\beta_{eestatu_3:early\ retiree}$	−0.1872	0.0777	−2.4100	0.0160	*
$\beta_{eestatu_4:medicare\ eligible\ retiree}$	−0.1774	0.1340	−1.3230	0.1857	
$\beta_{eestatu_5:retiree\ (status\ unknown)}$	−0.2135	0.2855	−0.7480	0.4545	
$\beta_{eestatu_6:COBRA\ continuee}$	0.3664	0.1607	2.2810	0.0226	*
$\beta_{eestatu_7:long\ term\ disability}$	1.0403	0.1598	6.5120	0.0000	***
$\beta_{eestatu_8:surviving\ spouse/depend.}$	0.5113	0.1943	2.6310	0.0085	**
$\beta_{eestatu_9:unknown}$	0.2274	0.0771	2.9510	0.0032	**
$\beta_{emprel_1:employee}$	−0.5073	0.0831	−6.1010	0.0000	***
$\beta_{emprel_2:spouse}$	−0.2611	0.0836	−3.1240	0.0018	**
$\beta_{emprel_3:child/other}$	0.7684	0.1622	4.7360	0.0000	***
$\beta_{plantyp_2:comprehensive\ plan}$	0.0854	0.0898	0.9500	0.3420	
$\beta_{plantyp_3:EPO\ plan}$	0.0834	0.1741	0.4790	0.6318	
$\beta_{plantyp_4:HMO\ plan}$	0.0846	0.0595	1.4220	0.1550	
$\beta_{plantyp_5:Non-Cap\ PoS\ plan}$	−0.1646	0.0725	−2.2680	0.0233	*
$\beta_{plantyp_6:PPO\ plan}$	0.0898	0.0423	2.1240	0.0336	*
$\beta_{plantyp_7:Cap\ or\ Part\ Cap\ PoS\ plan}$	0.1335	0.1284	1.0390	0.2986	
$\beta_{plantyp_8:CDHP}$	0.0243	0.0636	0.3830	0.7021	
$\beta_{plantyp_9:HDHP}$	−0.3364	0.0786	−4.2820	0.0000	***
Zero component					
$\gamma_{intercept}$	6.1238	0.2899	21.1240	< 2e-16	***
γ_{age}	−0.0314	0.0052	−6.0520	0.0000	***

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

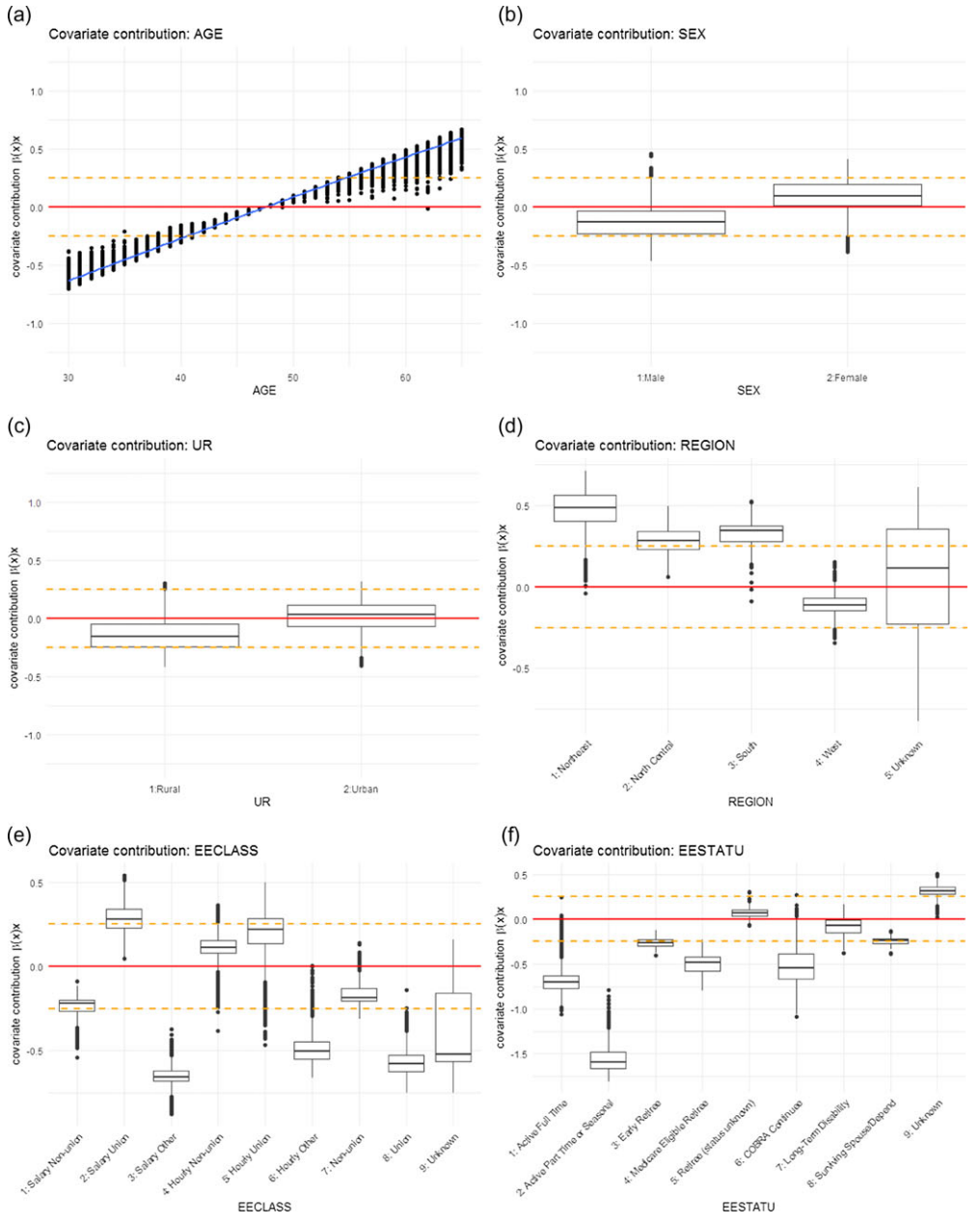

Figure 7. Graphical representation of covariate contributions for parameter λ in the ZIPNN model: (a) AGE; (b) SEX; (c) UR; (d) REGION; (e) EECLASS; (f) EESTATU.

variability in both cases suggests that gender has a consistent and explicit impact on admissions related to respiratory diseases.

Regarding the UR variable, the ZIP regression model suggests a lower admission rate for individuals in the urban area (UR = 2), based on its significant negative coefficient estimate (Table 6). This contradicts the pattern observed under the ZIPNN model. Although not considerably different, the covariate contributions suggest that people in urban areas are more prone to respiratory disease-related admissions (Figs. 7 and 8). This difference could be arising due to the

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

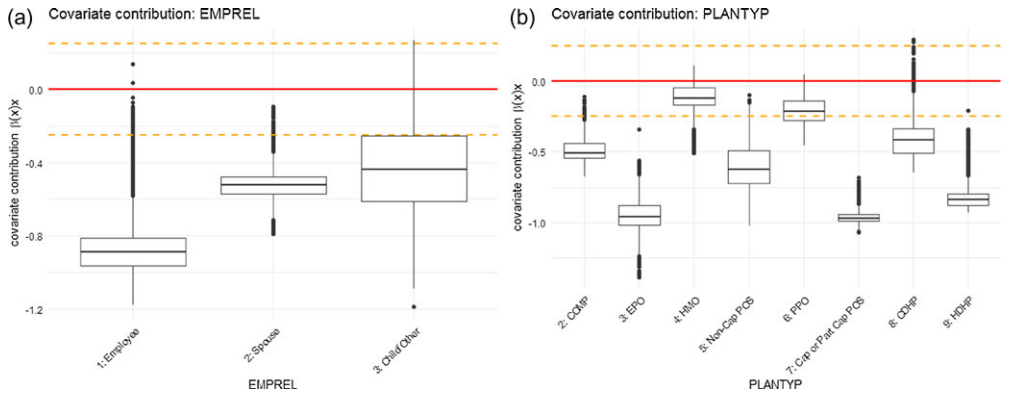

Figure 7. (continued). Graphical representation of covariate contributions for parameter λ in the ZIPNN model: (a) EMPREL; (b) PLANTYP.

fact that the network architecture captures complex interactions between different features, which is accounted for while estimating the non-linear regression weights, $\boldsymbol{\beta}(x)$. For instance, there may exist an interaction between the age and UR variables, due to the possibility that retired or older individuals are more likely to reside in rural areas.

The coefficient estimates based on the ZIP regression model for the REGION variable ($\beta_{region_z}, z = 1, \ldots, 5$), given in Table 6, indicate a higher rate of admission when the REGION variable is unknown (REGION = 5). However, the estimate is not significant. The high variability of covariate contribution under the ZIPNN model for level 5 of the region variable is also pointing toward this uncertainty (see Fig. 7). On the other hand, the "west" region (REGION = 4) has a significant negative coefficient estimate under the ZIP regression model, implying a lower admission rate for individuals from the western region. The covariate contribution in the ZIPNN model corresponding to the λ parameter for the REGION variable also suggests the same (Fig. 7).

In the case of the EECLASS variable, a significantly higher rate of admission is inferred from the coefficient estimates of the ZIP regression model for "hourly non-union" and "hourly union" (levels 4 and 5) and significantly low rates for individuals with unknown (level 9) employee classification (Table 6). A comparable trend is observed when the covariate contributions for both rate and probability parameters under the ZIPNN model are considered together, which is essential as interpreting the results based on a single parameter could be unclear. For example, the covariate contributions for the "salary union" level (level 2) of the EECLASS variable related to the rate parameter (see Fig. 7) suggest that the admission rate is high for individuals belonging to that employee classification cohort. However, the covariate contribution associated with the probability parameter (Fig. 8) indicates that the probability of having an admission is very low for individuals in that group. In contrast, for individuals with unknown employee classification, the covariate contributions not only indicate a lower probability of admission but also imply a lower admission rate.

Similarly, for the EESTATU variable, the combined analysis of covariate contributions of both parameters (Figs. 7 and 8) points toward a higher propensity for admission related to respiratory diseases for individuals with employment status categorized as "long-term disability" (level 7), whereas it is low for individuals with employment status "Active Part Time or Seasonal" (level 2). The same pattern is inferred from the associated coefficient estimates in the ZIP regression model. It is reasonable to conclude that the higher admission rates for people with long-term disability are as anticipated. Drawing further inferences regarding the relationship between employment details of individuals and hospital admission for respiratory diseases is difficult in the current context of admission data. The data provider has compiled the data set

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

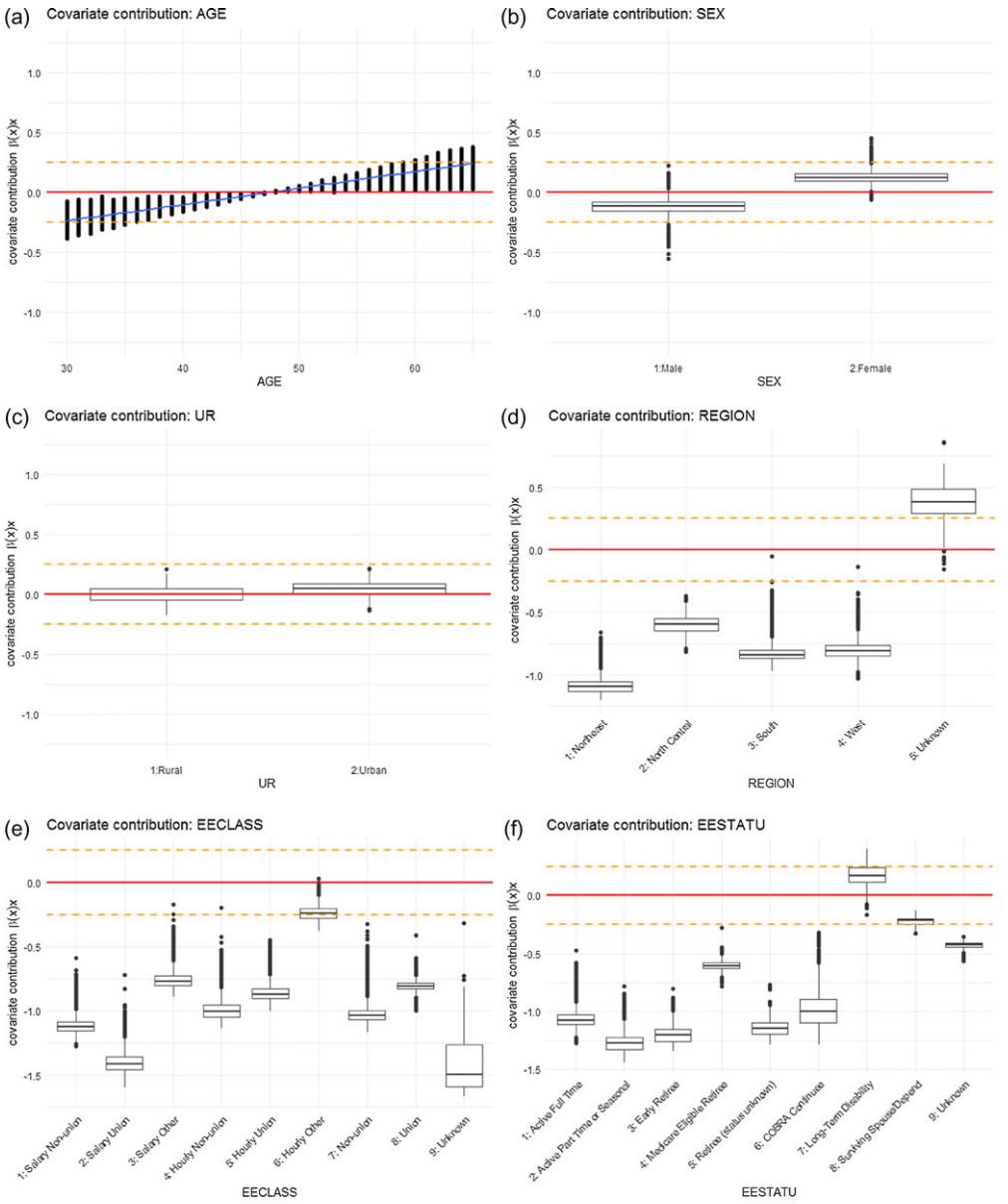

Figure 8. Graphical representation of covariate contributions for parameter p in the ZIPNN model: (a) AGE; (b) SEX; (c) UR; (d) REGION; (e) EECLASS; (f) EESTATU.

by assigning the employment-related information of the primary beneficiary/employee to their dependents as well. In other words, the employment information of those individuals classified as dependents (EMPREL as "spouse" (level = 2) or "child/other" (level = 3)) in the data set does not accurately represent their actual employment status. For instance, an unemployed dependent of an employee/primary beneficiary will also have the same employment-related information as the employee. Due to these inaccuracies within the data set, it becomes challenging to draw conclusive associations between employment information and the rate of admission due to respiratory diseases.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

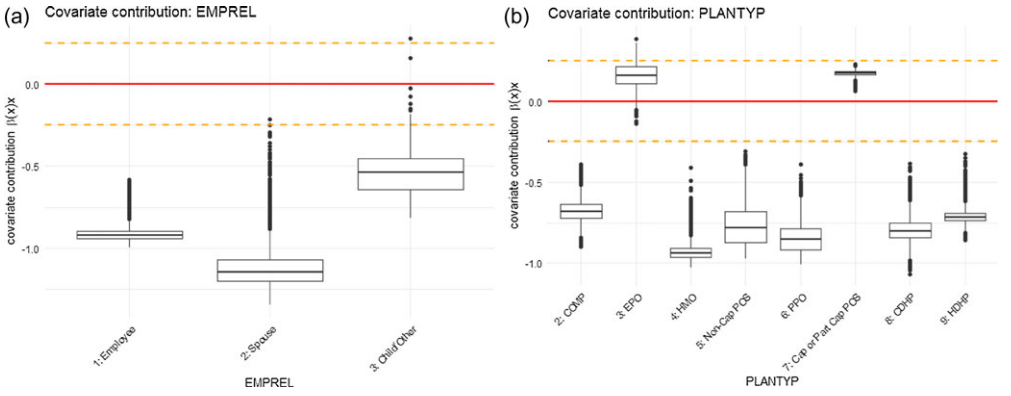

Figure 8. (continued). Graphical representation of covariate contributions for parameter p in the ZIPNN model: (a) EMPREL; (b) PLANTYP.

For the EMPREL variable, the pattern is slightly more evident. The coefficient estimates based on the ZIP regression model indicate a significant difference in the admission rates for different levels of EMPREL covariate (Table 6). The individuals with EMPREL as "child/other" (level = 3) have the highest rate of admission, followed by "spouse"(level = 2) and "employee"(level = 1). Although the same pattern is observed in the covariate contribution related to the λ parameter (Fig. 7 (continued)), the covariate contribution associated with p (Fig. 8 (continued)) indicates a lower probability of admission for individuals with EMPREL categorized as "spouse" compared to "employee."

The coefficient estimates for the different levels of the PLANTYP variable in the ZIP regression model indicate that the admission rates are significantly lower for the High-Deductible Health Plan (HDHP) (level = 9) than for other types of plans. This trend is also supported by the ZIPNN model. The covariate contributions not only indicate a lower probability of admission (Fig. 8 (continued)) but also suggest lower admission rates (Fig. 7 (continued)) for individuals under the "HDHP" plan type. This suggests that individuals who are covered under HDHP plans may have a lower likelihood of getting hospitalized for respiratory diseases, and if they do, it occurs at a lower rate. This might be due to the fact that the HDHP plan incentivizes individuals to be more proactive in managing their health and seek care only when essential or have more restrictive coverage.

7. Concluding remarks

In this article, we have considered the combined actuarial neural network (CANN) approach for constructing a Zero-inflated Poisson Combined Actuarial Neural Network (ZIPCANN) model for modeling admission rates related to health insurance. We also employed the LocalGLMnet approach to interpret the findings under the ZIPNN model and compared them with those obtained from the ZIP regression model. The comparisons demonstrated the capability of NN-based models to capture potential non-linear interactions among risk factors for both the probability and rate parameters of the mixture model (see Figs. 7 and 8, respectively). Furthermore, from a practical standpoint, the separate attention layers that we included in the NN structure for the two model components, enabled us to analyze the impact of important risk factors on both the probability and rate of admissions.

The results indicate that the ensemble of models developed in this work benefit from superior predictive performance compared to conventional regression models. Furthermore, the underlying zero-inflated mixture distribution allows the ZIPNN and ZIPCANN models to accommodate

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

the excess zero nature of the data, while also benefiting from improved predictive performance offered by NN methodologies. Moreover, the network architecture enables the model to capture potential interactions between features without the need to manually identify and specify them, as required in regression models. Additionally, the attention layer approach facilitates deriving interpretations from the model, allowing for an in-depth understanding of the various risk attributes on admissions related to respiratory diseases.

An interesting approach for further research would be to consider bivariate or multivariate versions of the developed models, aiming to address rates related to comorbidities. These models could extensively aid in devising healthcare intervention programs and investigating admissions related to other diseases. They could also be easily extended and adapted for other rate-setting problems within the insurance sector. Finally, it is worth noting that other zero-inflated models, such as a zero-inflated negative binomial model, can be implemented under the proposed framework, while extensions to bivariate and/or mutlivariate versions of such zero-inflated models for dealing with different types of claims are possible.

Acknowledgments. Certain data used in this study were supplied by Merative as part of one or more Merative MarketScan Research Databases. Any analysis, interpretation, or conclusion based on these data is solely that of the authors and not Merative.

We would also like to thank Ian Duncan of the University of California, Santa Barbara for his immense support and guidance.

Data availability statement. Restrictions apply to the availability of these data. We are unable to provide the data due to the data use agreement with Merative, who provided the data.

Funding statement. This research was funded by the Society of Actuaries, https://www.soa.org under a CAE research grant on "Predictive modelling for medical morbidity trends related to insurance".

References

Allaire, **J.**, & **Chollet**, **F.** (2021). keras: R Interface to 'Keras'. R package version 2.7.0.

Allaire, **J.**, & **Tang**, **Y.** (2021). tensorflow: R Interface to 'TensorFlow'. R package version 2.6.0.

Arık, **A.**, **Dodd**, **E.**, **Cairns**, **A.**, & **Streftaris**, **G.** (2021). Socioeconomic disparities in cancer incidence and mortality in england and the impact of age-at-diagnosis on cancer mortality. *Plos One*, **16**(7), e0253854.

Bahdanau, **D.**, **Cho**, **K.**, & **Bengio**, **Y.** (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv: 1409.0473.

Bousquet, **J.**, **Khaltaev**, **N. G.**, & **Cruz**, **A. A.** (2007). Global surveillance, prevention and control of chronic respiratory diseases. World Health Organization. https://apps.who.int/iris/handle/10665/43776

Cameron, **A. C.**, & **Trivedi**, **P. K.** (2013). *Regression analysis of count data*, vol. **53**. Cambridge University Press.

CDC. (2016). Icd- 10 - cm international classification of diseases, tenth revision. clinical modification (icd-10-cm). https://www.cdc.gov/nchs/icd/icd-10-cm.htm.

De Jong, **P.**, & **Heller**, **G. Z.** (2008). *Generalized linear models for insurance data*. Cambridge University Press.

Denuit, **M.**, **Charpentier**, **A.**, & **Trufin**, **J.** (2021). Autocalibration and tweedie-dominance for insurance pricing with machine learning. *Insurance: Mathematics and Economics*, **101**, 485–497.

Denuit, **M.**, **Hainaut**, **D.**, & **Trufin**, **J.** (2019). *Effective statistical learning methods for actuaries III: Neural networks and extensions*. Springer International Publishing.

Dillon, **J. V.**, **Langmore**, **I.**, **Tran**, **D.**, **Brevdo**, **E.**, **Vasudevan**, **S.**, **Moore**, **D.**, **Patton**, **B.**, **Alemi**, **A.**, **Hoffman**, **M.**, & **Saurous**, **R. A.** (2017). Tensorflow distributions. arXiv preprint arXiv: 1711.10604.

Dürr, **O.**, **Sick**, **B.**, & **Murina**, **E.** (2020). *Probabilistic deep learning: With Python, Keras and TensorFlow probability*. Manning Publications.

Famoye, **F.**, & **Singh**, **K. P.** (2006). Zero-inflated generalized poisson regression model with an application to domestic violence data. *Journal of Data Science*, **4**(1), 117–130.

Feng, **C.** (2022). Zero-inflated models for adjusting varying exposures: A cautionary note on the pitfalls of using offset. *Journal of Applied Statistics*, **49**(1), 1–23.

Frees, **E. W.** (2009). *Regression modeling with actuarial and financial applications*. Cambridge University Press.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Gao, **G.**, **Meng**, **S.**, & **Wüthrich**, **M. V.** (2019). Claims frequency modeling using telematics car driving data. *Scandinavian Actuarial Journal*, **2019**(2), 143–162.

Gurmu, **S.** (1997). Semi-parametric estimation of hurdle regression models with an application to medicaid utilization. *Journal of Applied Econometrics*, **12**(3), 225–242.

Gurmu, **S.**, & **Trivedi**, **P. K.** (1996). Excess zeros in count models for recreational trips. *Journal of Business & Economic Statistics*, **14**(4), 469–477.

Haberman, **S.**, & **Renshaw**, **A. E.** (1996). Generalized linear models and actuarial science. *Journal of the Royal Statistical Society: Series D (The Statistician)*, **45**(4), 407–436.

Hainaut, **D.** (2018). A neural-network analyzer for mortality forecast. *ASTIN Bulletin: The Journal of the IAA*, **48**(2), 481–508.

He, **K.**, **Zhang**, **X.**, **Ren**, **S.**, & **Sun**, **J.** (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hejazi, **S. A.**, & **Jackson**, **K. R.** (2016). A neural network approach to efficient valuation of large portfolios of variable annuities. *Insurance: Mathematics and Economics*, **70**, 169–181.

Jackman, **S.** (2020). pscl: Classes and Methods for R Developed in the Political Science Computational Laboratory. United States Studies Centre, University of Sydney, Sydney, New South Wales, Australia. R package version 1.5.5.

Jose, **A.**, **Macdonald**, **A. S.**, **Tzougas**, **G.**, & **Streftaris**, **G.**, Interpretable zero-inflated neural network models for predicting admission counts. *Annals of Actuarial Science*, **1-8**.

Jose, **A.**, **Macdonald**, **A. S.**, **Tzougas**, **G.**, & **Streftaris**, **G.** (2022). A combined neural network approach for the prediction of admission rates related to respiratory diseases. *Risks*, **10**(11), 217.

Keydana, **S.** (2022). tfprobability: Interface to 'TensorFlow Probability'. R package version 0.15.0.

Kuo, **K.** (2019). Deeptriangle: A deep learning approach to loss reserving. *Risks*, **7**(3), 97.

Lambert, **D.** (1992). Zero-inflated poisson regression, with an application to defects in manufacturing. *Technometrics*, **34**(1), 1–14.

Lee, **A. H.**, **Wang**, **K.**, & **Yau**, **K. K.** (2001). Analysis of zero-inflated poisson data incorporating extent of exposure. *Biometrical Journal*, **43**(8), 963–975.

Lundberg, **S. M.**, & **Lee**, **S.-I.** (2017). A unified approach to interpreting model predictions. In *Advances in neural information processing systems, 30*.

Martin, **J.**, & **Hall**, **D. B.** (2016). R 2 measures for zero-inflated regression models for count data with excess zeros. *Journal of Statistical Computation and Simulation*, **86**(18), 3777–3790.

Mullahy, **J.** (1986). Specification and testing of some modified count data models. *Journal of Econometrics*, **33**(3), 341–365.

Nelder, **J. A.**, & **Wedderburn**, **R. W.** (1972). Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, **135**(3), 370–384.

Noll, **A.**, **Salzmann**, **R.**, & **Wuthrich**, **M. V.** (2020). Case study: French motor third-party liability claims. Available at SSRN 3164764.

Ohlsson, **E.**, & **Johansson**, **B.** (2010). *Non-life insurance pricing with generalized linear models*, vol. **2**. Springer.

Ozkok, **E.**, **Streftaris**, **G.**, **Waters**, **H. R.**, & **Wilkie**, **A. D.** (2014). Modelling critical illness claim diagnosis rates i: Methodology. *Scandinavian Actuarial Journal*, **2014**(5), 439–457.

R Core Team (2021). R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria.

Ribeiro, **M. T.**, **Singh**, **S.**, & **Guestrin**, **C.** (2016). "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1135–1144).

Richman, **R.** (2021). Ai in actuarial science–a review of recent advances–part 2. *Annals of Actuarial Science*, **15**(2), 230–258.

Richman, **R.**, & **Wüthrich**, **M. V.** (2020). Nagging predictors. *Risks*, **8**(3), 83.

Richman, **R.**, & **Wüthrich**, **M. V.** (2023). LocalGLMnet: Interpretable deep learning for tabular data. *Scandinavian Actuarial Journal*, **2023**(1), 71–95.

Ridout, **M.**, **Demétrio**, **C. G.**, & **Hinde**, **J.** (1998). Models for count data with many zeros. In *Proceedings of the XIXth international biometric conference*, vol. 19 (pp. 179–192). International Biometric Society Invited Papers Cape Town, South Africa.

RStudio Team (2021). RStudio: Integrated Development Environment for R. RStudio, PBC, Boston, MA.

Schelldorfer, **J.**, & **Wuthrich**, **M. V.** (2019). Nesting classical actuarial models into neural networks. Available at SSRN 3320525.

Vaswani, **A.**, **Shazeer**, **N.**, **Parmar**, **N.**, **Uszkoreit**, **J.**, **Jones**, **L.**, **Gomez**, **A. N.**, **Kaiser**, **Ł.**, & **Polosukhin**, **I.** (2017) Attention is all you need. In *Advances in neural information processing systems, 30*.

Vrieze, **S. I.** (2012). Model selection and psychological theory: A discussion of the differences between the Akaike information criterion (aic) and the Bayesian information criterion (bic). *Psychological Methods*, **17**(2), 228–243.

WHO. (2022). Heath topics- chronic respiratory diseases. https://www.who.int/health-topics/chronic-respiratory-diseasestab_1.

Wüthrich, **M. V.** (2020). Bias regularization in neural network models for general insurance pricing. *European Actuarial Journal*, **10**(1), 179–202.

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Wüthrich, **M. V.** (2022). The balance property in neural network modelling. *Statistical Theory and Related Fields*, **6**(1), 1–9.
Wüthrich, **M. V.**, & **Merz**, **M.** (2019). Yes, we cann!. *ASTIN Bulletin: The Journal of the IAA*, **49**(1), 1–3.
Wüthrich, **M. V.**, & **Merz**, **M.** (2023). *Statistical foundations of actuarial learning and its applications*. Springer Nature.
Yip, **K. C.**, & **Yau**, **K. K.** (2005). On modeling claim frequency data in general insurance with extra zeros. *Insurance: Mathematics and Economics*, **36**(2), 153–163.
Zeileis, **A.**, **Kleiber**, **C.**, & **Jackman**, **S.** (2008). Regression models for count data in R. *Journal of Statistical Software*, **27**(8).

A. Appendix

A. 1. Tables

Table A.1. The testing loss, learning loss, and average fitted mean of the ZIPNN and ZIPCANN models with (50,35,25), (35,25,20), (25,20,15), (20,15,10), and (15,10,5) neurons in the initial three hidden layers.

Model	Learning loss	Testing loss	Average fitted mean
ZIPNN(50,35,25)	26,191.8	2823.2	0.0026
ZIPNN(35,25,20)	26,218.9	2827.9	0.0025
ZIPNN(25,20,15)	26,319.2	2811.8	0.0026
ZIPNN(20,15,10)	26,428.3	2803.3	0.0026
ZIPNN(15,10,5)	26,430.2	2806.1	0.0026
ZIPCANN(50,35,25)	26,358.8	2815.1	0.0027
ZIPCANN(35,25,20)	26,319.3	2810.1	0.0028
ZIPCANN(25,20,15)	26,286.8	2816.5	0.0027
ZIPCANN(20,15,10)	26,366.3	2813.6	0.0027
ZIPCANN(15,10,5)	26,383.0	2817.5	0.0027

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

A.2. Codes and model structure

Listing A3. Code for implementing *ZIPNN*.

```
# Main architecture with 3 hidden layers with q1,q2,q3 nodes in each layer
Network<- Design %>%
  # 1st hidden layer
  layer_dense(units = q1, activation = 'tanh', name = 'hidden1') %>%
      layer_dropout(rate = p) %>%
   # 2nd hidden layer
  layer_dense(units = q2, activation = 'tanh', name = 'hidden2') %>%
      layer_dropout(rate = p) %>%
  # 3rd hidden layer
  layer_dense(units = q3, activation = 'tanh', name = 'hidden3') %>%
      layer_dropout(rate = p) %>%
  # provide two neuron in the intermediate layer
  layer_dense(units = 2, name = 'Net')

Network<-list(Network,LogVol) %>% layer_concatenate(name='concate1')# adding
    offset  node
#function used in lambda layer to add the exposure and to define the zero
    inflated distribution
zero_inf<-function(out){
  # rate/lambda= exp(node 1 + offset) and the squeeze function is used to
      flatten the tensor
  rate=list(out[,1],out[,3]) %>% layer_add(name = 'Add') %>% k_exp() %>%
      tf$squeeze()
  #probability of count component = node 2 and the sigmoid function ensures
      value between 0 and 1
  s = tf$math$sigmoid(out[,2]) %>% layer_flatten()
  probs = layer_concatenate(list(1-s,s),axis = -1) # probabilities of two
      components in the mixture distribution

# Defining zero inflated distribution as a mixture distribution
  return (tfd_mixture(  #(I)
    cat = tfd_categorical(probs = probs), #(II)
    components = list(
      tfd_deterministic(loc = tf$zeros_like(rate)), #(III)
      tfd_poisson(rate = rate)) #(IV)
  )
  )
}
--------------------------------------------------------------------------
#(I): Creates a mixture distribution with two components: one for zero and
#one for a Poisson distribution.
#(II): Represents the probabilities of these two components using an
#instance of a categorical distribution.
#(III) and (IV) correspond to the two components:
#(III): Represents a definite value of zero using a scalar deterministic
#distribution.
#(IV): Represents a value derived from a Poisson distribution.
--------------------------------------------------------------------------

#negative loglikelihood loss function
nll<-function(y_true,y_pred)
{
  -y_pred$log_prob(k_reshape(y_true,shape = c(-1)))# k_shape for flattening
      the tensor
  #y_pred is the ZIP distribution layer
  #y_true is the observed admission count
}
--------------------------------------------------------------------------
```

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

```
#------------------Defining the distribution layer--------------------------
#Model is defined in such a way that the output layer is distribution layer.
#The 'convert_to_tensor_fn' is set to 'tfd_mean' in order to extract the
#fitted mean E(y) from the distribution.
#Samples from the distribution could be generated using 'tfd_sample'.
#Input to the 'make_distribution_fn' is the function used to define the ZIP
#distribution.

#Constructs a distribution layer with an underlying ZIP distribution
#assumption, which is defined as a mixture distribution.

p_y_zi =layer_distribution_lambda(make_distribution_fn = zero_inf,
    convert_to_tensor_fn = tfd_mean)#tfd_sample for sample
------------------------------------------------------------------------------

# ------------------ Model configuration and fitting -------------------------
# Model assembly
model_zi = keras_model(inputs= c(Design,LogVol), outputs=p_y_zi(Network))
summary(model_zi)

#callback to avoid overfitting
CBs<-callback_model_checkpoint("path0",monitor = "val_loss",save_best_only =
    TRUE,                verbose = 1, save_weights_only = TRUE)

# Model configuration
model_zi %>% compile(
  loss = nll, # set poisson NLL  loss function as the objective loss function
  optimizer = 'nadam'
)
# Model fitting by running gradient descent method to minimize the objective
    loss function
  fit <- model_zi %>% fit(
    list(Design.learn,LogVol.learn), # all predictors and the offset term
    Ylearn, # response
    verbose = 1, # verbose = 0 silences the progress bar for the process
    # verbose = 1 shows the fitting process, incl. learning loss and
        validation loss, epoch by epoch
    epochs = epochs, # epochs = 250
    batch_size = batchsize, # batchsize = 30,000
    validation_split = 0.2, # 20% as validation set
    callbacks = CBs)
  load_model_weights_hdf5(model_zi,"path0")
```

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Listing A4. Code for implementing *ZIPCANN*.

```
#skip connection for lambda
Skip_1<-Design %>% layer_dense(units = 1, activation = 'linear', name = '
    Skip_1')
#skip connection for probability
Age<-layer_input(shape = c(1), dtype = 'float32', name = 'Age')
Skip_2<-Age %>% layer_dense(units = 1, activation = 'linear', name = 'Skip_2')

add_skip<-function(out,skip_1,skip_2){
  r<-list(out[,1],skip_1) %>% layer_add(name = 'rate_skip')
  s<-list(out[,2],skip_2) %>% layer_add(name = 'p_skip')
  net_skip<-list(r,s) %>% layer_concatenate(name='net_skip')
  return(net_skip)
}
Network<-add_skip(Network,Skip_1,Skip_2)
Network<-list(Network,LogVol) %>% layer_concatenate(name='concate1')# adding
    offset  node
# Model assembly
model_zi = keras_model(inputs= c(Design,LogVol,Age), outputs=p_y_zi(Network))
```

Listing A5. Structure of the *ZIPCANN* model.

```
Model: " ZIPCANN model"
_________________________________________________________________________
 Layer (type)                Output Shape Param#  Connected to
=========================================================================
 Design (InputLayer)         [(None, 37)] 0       []
 hidden1 (Dense)             (None, 20)   760     ['Design[0][0]']
 dropout_2 (Dropout)         (None, 20)   0       ['hidden1[0][0]']
 hidden2 (Dense)             (None, 15)   315     ['dropout_2[0][0]']
 dropout_1 (Dropout)         (None, 15)   0       ['hidden2[0][0]']
 hidden3 (Dense)             (None, 10)   160     ['dropout_1[0][0]']
 dropout (Dropout)           (None, 10)   0       ['hidden3[0][0]']
 Net (Dense)                 (None, 2)    22      ['dropout[0][0]']
 Age (InputLayer)            [(None, 1)]  0       []
 tf.__operators__.getitem    (None,)      0       ['Net[0][0]']
 (SlicingOpLambda)
 Skip_1 (Dense)              (None, 1)    38      ['Design[0][0]']
 tf.__operators__.getitem_1  (None,)      0       ['Net[0][0]']
 (SlicingOpLambda)
 Skip_2 (Dense)              (None, 1)    2       ['Age[0][0]']
 rate_skip (Add)             (None, 1)    0       ['tf.__operators__.
                                                  getitem[0][0]',
                                                  'Skip_1[0][0]']
 p_skip (Add)                (None, 1)    0       ['tf.__operators__.
                                                  getitem_1[0][0]',
                                                  'Skip_2[0][0]']
 net_skip (Concatenate)      (None, 2)    0       ['rate_skip[0][0]',
                                                   'p_skip[0][0]']
 LogVol (InputLayer)         [(None, 1)]  0       []
 concate1 (Concatenate)      (None, 3)    0       ['net_skip[0][0]',
                                                    'LogVol[0][0]']
 distribution_lambda         (None, None) 0       ['concate1[0][0]']
 (DistributionLambda)
=========================================================================
Total params: 1,297
Trainable params: 1,297
Non-trainable params: 0
```

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Listing A6. Code for implementing *LocalGLMnet*.

```
# Main architecture with 3 hidden layers and attention layer
Attention <- Design %>%
# first three hidden layers
  layer_dense(units = q1, activation = 'tanh', name = 'hidden1') %>%
      layer_dropout(rate = p) %>%
  layer_dense(units = q2, activation = 'tanh', name = 'hidden2') %>%
      layer_dropout(rate = p) %>%
  layer_dense(units = q3, activation = 'tanh', name = 'hidden3') %>%
      layer_dropout(rate = p) %>%
  #layer for attention weights
  layer_dense(units = q0, activation = 'linear', name = 'Attention')

Network<-list(Design,Attention) %>%
  # taking dot product of attention weights and feature set
  layer_dot(name='localglm',axes = 1) %>%
  # provide one neuron in the output layer
  layer_dense(units = 1, activation = 'linear', name = 'Network')
Response = list(Network, LogVol) %>%
        # add the exposure and the last neuron
        layer_add(name = 'Add') %>%
        # give the response
        layer_dense(units = 1,
                    activation = 'exponential',
                    name = 'Response',
                    trainable = FALSE,
                    weights = list(array(1, dim = c(1,1)), array(0, dim = c(1)
                        )))

model <- keras_model(inputs = c(Design,LogVol), outputs = c(Response))
```

Listing A7. Code for interpretable *ZIPNN* model.

```
# Main architecture with 3 hidden layers
Attention<- Design %>%
  layer_dense(units = q1, activation = 'tanh', name = 'hidden1') %>%
      layer_dropout(rate = p) %>%
  layer_dense(units = q2, activation = 'tanh', name = 'hidden2') %>%
      layer_dropout(rate = p) %>%
  layer_dense(units = q3, activation = 'tanh', name = 'hidden3') %>%
      layer_dropout(rate = p) %>%
  #layer for attention weights with dimension= 2*q_0
  layer_dense(units = c(2*q0), activation = 'linear', name = 'Attention')

# rate parameter from attention weights
lamda_local<-list(Design,Attention[,1:q0])%>% layer_dot(name='localglm_1',axes
     = 1) %>%    layer_dense(units = 1, activation = 'linear', name = '
    lamda_local')

# probability parameter from attention weights
p_local   <-list(Design,Attention[,(q0+1):(2*q0)])%>% layer_dot(name='
    localglm_2',axes = 1)  %>%  layer_dense(units = 1, activation = 'linear',
    name = 'p_local')

Network<-list(lamda_local,p_local,LogVol) %>% layer_concatenate(name='concate1
    ')
#the rest of the steps is same as that of the ZIPNN model without attention
    layer
```

https://doi.org/10.1017/S1748499524000058 Published online by Cambridge University Press

Listing A8. Structure of the interpretable *ZIPNN* model.

```
Model: "Interpretable ZIPNN model"
__________________________________________________________________________
 Layer (type)                Output Shape  Param #  Connected to
==========================================================================
 Design (InputLayer)         [(None, 37)]  0        []
 hidden1 (Dense)             (None, 20)    760      ['Design[0][0]']
 dropout_2 (Dropout)         (None, 20)    0        ['hidden1[0][0]']
 hidden2 (Dense)             (None, 15)    315      ['dropout_2[0][0]']
 dropout_1 (Dropout)         (None, 15)    0        ['hidden2[0][0]']
 hidden3 (Dense)             (None, 10)    160      ['dropout_1[0][0]']
 dropout (Dropout)           (None, 10)    0        ['hidden3[0][0]']
 Attention (Dense)           (None, 74)    814      ['dropout[0][0]']
 tf.__operators__.getitem    (None, 37)    0        ['Attention[0][0]']
 (SlicingOpLambda)
 tf.__operators__.getitem_1  (None, 37)    0        ['Attention[0][0]']
 (SlicingOpLambda)
 localglm_1 (Dot)            (None, 1)     0        ['Design[0][0]',
                                                     'tf.__operators__.
                                                    getitem[0][0]']
 localglm_2 (Dot)            (None, 1)      0       ['Design[0][0]',
                                                     'tf.__operators__.
                                                      getitem_1[0][0]']
 lamda_local (Dense)         (None, 1)      2       ['localglm_1[0][0]']
 p_local (Dense)             (None, 1)      2       ['localglm_2[0][0]']
 LogVol (InputLayer)         [(None, 1)]    0       []
 concate1 (Concatenate)      (None, 3)      0       ['lamda_local[0][0]',
                                                      'p_local[0][0]',
                                                      'LogVol[0][0]']
 distribution_lambda         (None, None)   0       ['concate1[0][0]']
 (DistributionLambda)
==========================================================================
Total params: 2,053
Trainable params: 2,053
Non-trainable params: 0
```

Cite this article: Jose A, Macdonald AS., Tzougas G and Streftaris G (2024). Interpretable zero-inflated neural network models for predicting admission counts, *Annals of Actuarial Science*, 1–31. https://doi.org/10.1017/S1748499524000058