MSBA Capstone Summary Report

Cohort A Team 7
Elmira Ushirova, Ziqin Ma, Shihan Li, Qiaoling Huang, Chenran Peng

Github Repository Access:
https://github.com/ziqinm/MSBA-A7-Public.git

Introduction and Problem Statement

This is the capstone project summary report of MSBA Cohort A Team 7 at Boston University. Our project is focusing on homeowner insurance. Recommended and required in some regions, house insurance covers losses and damages to an individual's house and assets in the house. In general, the losses can be categorized into two types: catastrophic and non-catastrophic damages. Catastrophic losses come from damages from severe natural disasters, such as earthquakes, volcano eruptions, storms, and floods. The loss amount is estimated by existing math models, and the amount paid is limited below a maximum.

In this project, we mainly focused on non-catastrophic property and liability losses. Examples of non-catastrophic damage include non-catastrophic flood, fire, and any other damage that occurred on the property (liability). Our main goal is to find exciting features and patterns of the factors related to damage loss. From the insurance company's perspective, the pricing of the insurance policies is based on the numbers predicted by existing models. Thus, it is crucial to update the existing models with new information frequently, so the price of the insurance packages can reflect the current situation on the market.

With such interests and goals, we finalized our business problem: exploring new features from external data sources and evaluating their performance in house damage loss prediction. We want to build a predictive model for homeowner insurance loss, taking the predictions of an existing model into account and our newly-generated features from outside sources (not included in the existing model). We analyzed the data on the zip code level, and the scope of implementation will be Northeastern 5 States in the U.S.: Massachusetts, Connecticut, New Jersey, New Hampshire, Pennsylvania.

Datasets and sources

1. Insurance losses data from an Insurance Company

During this project, we were working closely with a fast-growing home insurance company that provides services in the New England area. This company provided us with the anonymized

data of actual loss data for five years (2013 - 2018) and the predicted loss amounts from their existing models, on the zip code level. The provided data also includes Earned Exposures for each address, showing the proportion of the year that a specific address stayed with a particular insurance policy. Besides the preliminary data, the company provided a supplement dataset with population, area, and the number of properties in each zip code.

Source: anonymized Company-owned data.

2. Crime data from the FBI

We found an official crime statistics from the FBI website. This crime data includes the numbers of different crime categories by State and city. There are 20 separate excel tables, each table representing one-year statistics from 1999 to 2018. The crime categories cover Violent crimes (Murder and nonnegligent manslaughter, Rape, Robbery, Aggravated assault) and Property crimes (Burglary, Larceny- theft, Motor vehicle theft, and Arson).

While exploring original tables, we observed that the number of Violent crimes is the sum of Murder and nonnegligent manslaughter, Rape, Robbery, Aggravated assault, and the number for Property crime is the sum of Burglary, Larceny- theft, Motor vehicle theft, Arson. To prepare the data for programming, we manually cleaned up the formats and changed the format to csv files.

Sources (links are for the recent 3 years; already obtained all years since 1999):
[Introduction] [2018] [2017] [2016]

3. Lightning data from NOAA

We obtained raw lightning data from the Google Cloud Platform's public datasets. The full dataset consists of 32 tables, each representing one-year data from 1987 to 2019. For our analysis, we focused only on the recent 20 years (1999 - 2018). We did not download 2019 data, because the 2019 table only had December data, at the moment of our progress. Each table contains a daily number of strikes and geopoints (around US territory). The tables can be extracted from GCP BigQuery.

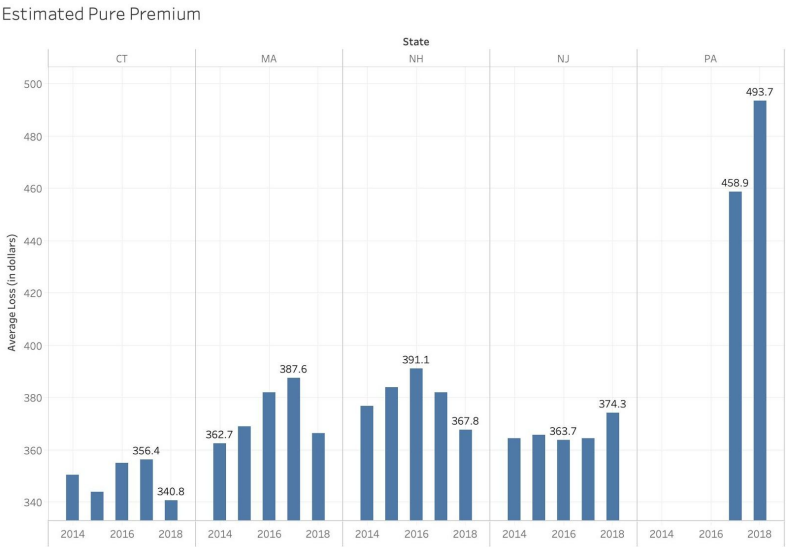
Source:
https://console.cloud.google.com/marketplace/details/noaa-public/lightning?filter=solution-type:dataset&filter=category:climate&id=d18e2712-bc50-471a-bf22-a2de3d9489d9&authuser=1

Exploratory Data Analysis

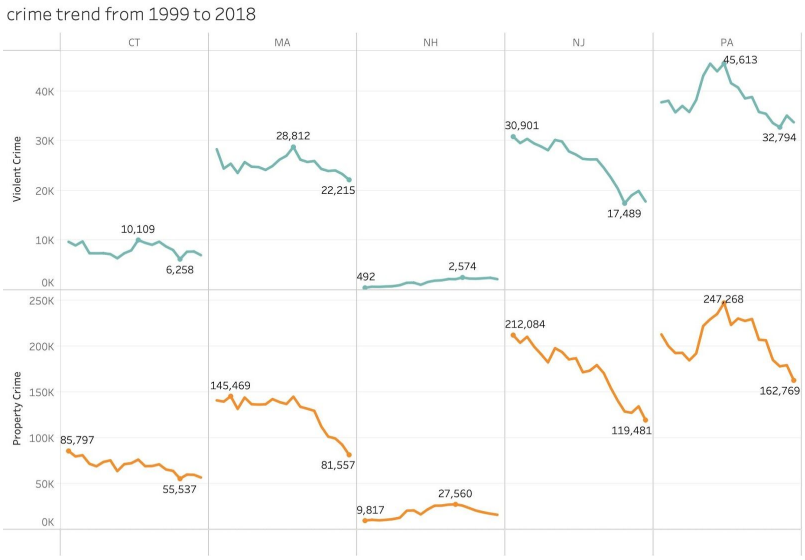
1. Insurance company data

Based on the Company data, we calculated Pure Premiums for each State and each year, dividing actual total loss by the total earned exposure. Pure Premium is usually used as a base for the insurance policy pricing.

From the graph to the right, we can see that Pennsylvania has the highest Pure Premium, which indicates that there were more home damaging issues than in the other four States. The less risky State with the lowest Pure Premium over the years among all five States is Connecticut.



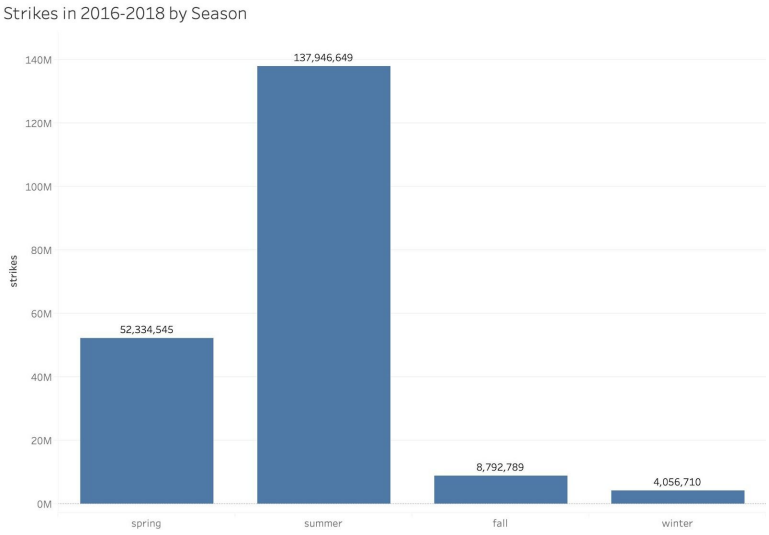
2. Crime data



We summarised the number of violent and property crimes over the 1999-2018 period for 5 States in the graph to the left. From the graph, we noticed that Pennsylvania has the highest number of crimes, while New Hampshire has the lowest number of crimes. Overall, the total amount of crimes over the years is significantly decreasing.

3. Lightning data

For our analysis, we grouped the lightning tables by geopoints and summarised the total number of strikes, number of events (count of days), and the name of seasons during which the lightning happened.



On the graph to the left, we saw that over three years (2016-2018) within the U.S. territory, the lightning took place mostly in summer and spring.

How we tackled the problem

Our methodology to building a model and testing significant features followed the below 6 steps:

Step 1 - NA imputation

As our models only process data with no NAs, we came up with our unique approach to NA imputation. After merging original crime and lightning data to company data by State, city, and zip code, we got different sources of NAs: NAs in crime data, NAs in lightning data, and NAs in the population column in company supplement data.

For crime data, we replaced NAs in the original data by computing the mean of the variable in each State and multiplying the mean by the population ratio of the missing city (i.e., State average number * city population/State population).

However, some cities in company data did not have recorded population values. We checked and confirmed that those cities did not have any crime data information either (the whole row after merging was NA). We also searched for the official statistics for those cities and found that most of the places are small towns and even part of towns, so their population statistics were

not collected. Therefore, we assumed that those places without population information are similar to the cities without any crime information, implying that cities might be too small for the FBI to track. Then we replaced the missing populations with the mean population of other cities without any criminal record and used that population to fill the NAs in crime variables.

For lightning data NAs, we assumed that if there exists any NA after merging data, there were no strikes in that location. Because our original lightning data only consisted of occurred strikes, it is safe to assume that it did not happen if it was not recorded.

Step 2 - Data feature engineering

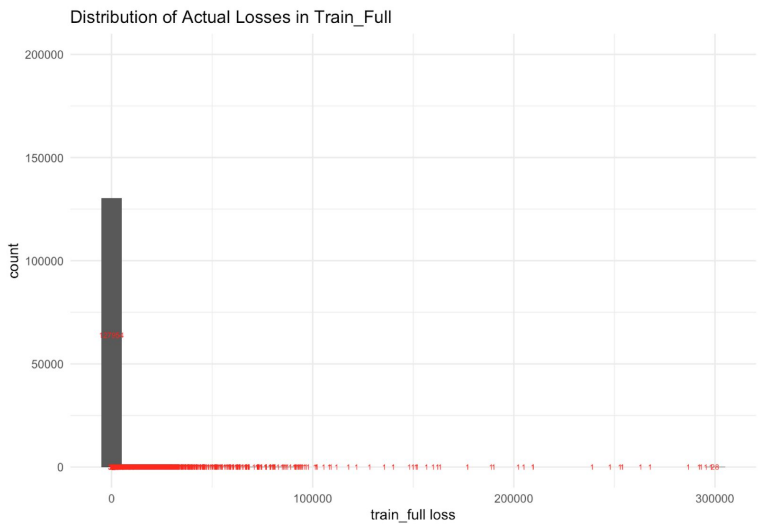
After NA imputation, we started the data feature engineering process. We created two sets of new variables, separately for crime and lightning data.

In crime, we aggregated the data on recent 1, 3, 5, 10, and 20 years, getting five aggregated levels by years. For each of the year aggregated level, we calculated the sum of original numbers, those sums against State sums (State level), those sums against national sum (national level), and ratios such as Violent crime and Property crime against Total crime, each type of Violent crime against total Violent crime, and each type of Property crime against total Property crime. In the end, we got 205 new features for the crime data.

In lightning, we applied the same aggregation level by years and divided each year level by season. Then, for each aggregated year and season level, we calculated the sum of original numbers, State and national level numbers, as well as ratios such as Strikes and Events per square mile, per population, and per property number. In the end, we got 315 new variables for lightning.

We treated our new variables by applying limits of 1 and 99 percentile to avoid the effects of extreme outliers if they were any.

Step 3 - Sample design



We used years 2016-2018 for our in-time training and test sets, and years 2014-2015 for out-of-time validation. We applied a 70%/30% split for the in-time train and test. Moreover, we stratified the sample by year, State, and loss. To ensure an equal division of loss, we applied a cap of $300,000 (99.3 percentile) and categorized loss as zero (loss = 0),

low (loss < 3,000), medium (3,000<= loss < 15,000), and high (15,000 <= loss <= 300,000).

In addition, we noticed that our data is very unbalanced in loss distribution: many zeros against the actual losses. Therefore, we decided to do downsampling to balance zero and non-zero losses, by randomly dropping 50% of zero losses in the full train and test sets. At the end of this step, we got five samples for modeling: Train_Reduced, Test_Reduced datasets, Train_Full, Test_Full, and validation.

Step 4 - Variable selection

For calculation convenience, we used Python to perform variable selection. We used a backward selection approach and got only 33 important variables out of all 520 new variables:

Crime			
Robbery at State level for 1 year	Larceny-theft /Property crime at 20 years level	Murder/Property crime at 10 years level	Motor vehicle theft at State level 10 years
Motor vehicle theft/Property crime 3 years level	Aggravated assault /Total Violent crime 1 year level	Motor vehicle theft /Property crime at 20 years level	Larceny-theft /Property crime at 10 years level
Property crime at State level 10 years	Robbery at State level 5 years	Motor vehicle theft at State level 3 years	Burglary at State level 5 years
Property crime /Total crime 10 years level	Burglary at State level 10 years	Property crime at State level 5 years	Burglary/Property crime 3 years level
Aggravated assault /Total Violent crime 5 year level	Property crime at nation level 10 years	Burglary at State level 20 years	Larceny-theft/Property crime 3 years level
Larceny-theft at State level 5 years	Violent crime/Total crime 10 years level	Larceny-theft at State level 20 years	

Lightning			
Event number in fall for 3 years	Total strikes for 5 years	Strikes in winter for 5 years	Event number in winter for 5 years
Total strikes for 10 years	Total events in 10 years	To total strikes in fall for 10 years	Strikes in winter for 10 years
Event number at nation level for 10 years	Event number in summer for 3 years		

The feature selection results indicate that crime data have more predicting power than lightning, since there are more crime variables than lightning.

Step 5 - Modeling

With our prepared samples and selected variables, we started model design. We decided to use four different models: linear-based (GLM), tree-based (Random Forest and XGBoost), and Neural Network. All models were trained on the Train_Reduced set (the downsampled one) and predictions were made on the rest 4 samples (Test_Reduced, Train_Full, Test_Full, and validation).

We also wanted to see whether our new generated variables have any predictive power (better than random prediction) and whether they have any additional predictive power when combined with the existing model prediction (oldmodel variable). So we repeated the train and fit process twice: one with only new variables and the other with new variables and oldmodel predictions.

In GLM, we used the default parameters for both cases of with and without oldmodel.

In Random Forest, we used three trees and other default parameters for both cases.

In XGBoost, we faced similar overfitting issues; however, we were able to readjust by tweaking the model parameters. For without the oldmodel case, we used 1 fold and 1 round. For with the oldmodel case, we used 5 folds and 10 rounds as model parameters. We used “reg:squarederror” as the objective in both cases.

In Neural Network, we used a Python deep learning package called Keras and applied a basic structure of the neural network model provided by this package. The Neural Network model we used consists of 3 ‘Dense’ layers and were fitted to our data by 3 epochs with a batch size of 33. (Source: https://keras.io/)

Step 6 - Model performance evaluation

In order to choose the most appropriate measure to evaluate model performances, we consulted with an industry expert from the company data owner. We learned that in practice, they are using a slightly modified AUC (Area Under the Curve) score. Accordingly, we obtained a step-by-step calculation of this measure from the industry expert and applied it to our models.

In the case of new variables without the `oldmodel` variable, we compared our results to a random prediction AUC that equals 0.5. If a model's AUC is higher than 0.5, it behaves better than randomness, and the variables do have some predicting power.

In the case of new variables with the ` oldmodel` variable, we calculated AUC for the `oldmodel` only and used those numbers as benchmarks. If a model's AUC is higher than the benchmark, our new variables have additional predictive power.

Model Results and Discussion

Models with new variables only:

Results for models without oldmodel					
Model	Train_reduced AUC	Test_reduced AUC	Train_full AUC	Test_full AUC	Validation AUC
Benchmark (randomness)	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**
Simple GLM	0.631	0.516	0.600	0.606	0.48
Random Forest	0.834	0.481	0.719	0.759	0.523
XGBoost	**0.777**	**0.605**	**0.668**	**0.672**	**0.554**
Neural Network	0.551	0.526	0.528	0.579	0.540

In this table, almost all of our models performed well on each of the sample sets. While choosing the best model, our focus was mainly on beating the benchmark and having the highest scores on Train_Full and Test_Full.

Regarding the benchmark, simple GLM could not beat the benchmark in the validation set; Random Forest could not beat the benchmark in the Test_Reduced set. Compared to Neural Network, XGBoost has higher scores in Train_Full and Test_Full sets. Based on the scores, we chose the XGBoost model as the best in this case.

Feature	Importance %
Motor vehicle theft/Property crime 3 years level	56%
Larceny-theft/Property crime 3 years level	14%
Larceny-theft at State level 5 years	7%
Strikes in winter for 5 years	6%
Robbery at State level for 1 year	4%
Total strikes for 5 years	3%

Knowing that our new 33 variables do have some power in loss prediction for homeowner insurance, we looked at feature importance in the XGBoost model to see which variable had the most significant effect on prediction. The most significant variables are `Motor vehicle theft/Property crime 3 years level` (56%),` Larceny-theft/Property crime 3 years level` (14%), `Larceny-theft at State level 5 years` (7%), `Strikes in winter for 5 years` (6%), `Robbery at State level for 1 year` (4%), and `Total strikes for 5 years` (3%). Other variables contribute around ~1%.

Models with `oldmodel` variable:

Results for models with oldmodel					
Model	Train_reduced AUC	Test_reduced AUC	Train_full AUC	Test_full AUC	Validation AUC
Benchmark (oldmodel)	**0.572**	**0.610**	**0.598**	**0.554**	**0.641**
Simple GLM	0.649	0.619	0.643	0.651	0.600
Random Forest	0.931	0.538	0.808	0.802	0.592
XGBoost	**0.735**	**0.607**	**0.699**	**0.679**	**0.608**
Neural Network	0.607	0.552	0.582	0.621	0.547

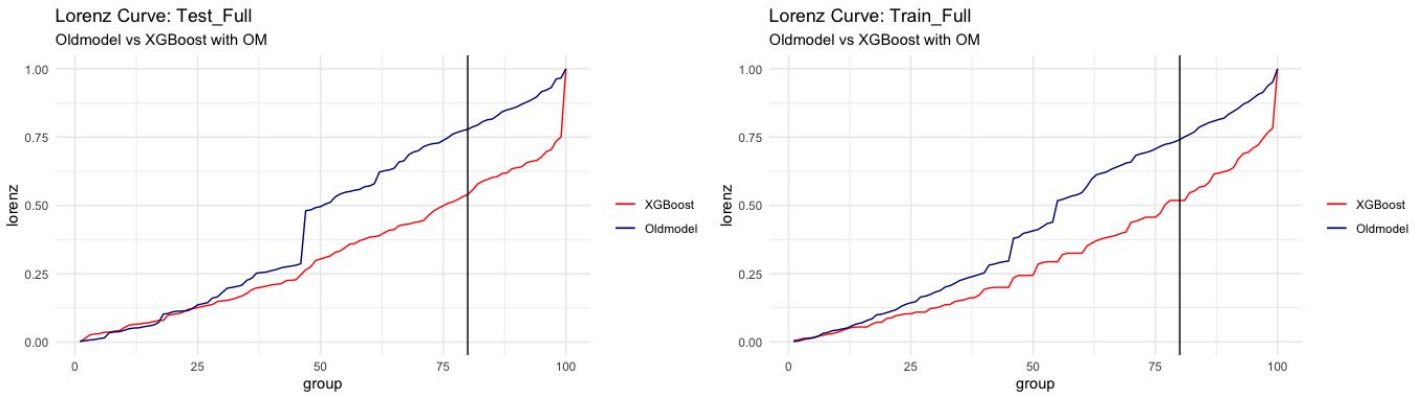
In this case, we encountered the main issue of overfitting of Random Forest, which was using only 3 trees. Even with 1 tree, the overfitting was still apparent. One possible explanation we came up with is the unbalanced structure of our data that might not work well with Random Forest. Therefore, we decided to not consider Random Forest as a candidate for the best model.

When choosing the best model, we focused on the same standards as in the previous case. From the table, Neural Network could not outperform the benchmark in most of the sample sets. We assumed that our input data is too small for a sophisticated deep learning structure. Compared to GLM, XGBoost has higher scores in Train_Full and Test_Full. Thus, we chose XGBoost as the best model in this case, too, even though GLM was not too far behind.

We were aware that none of our models actually beat the benchmark in the validation set, where the benchmark itself was the highest compared to other benchmarks. Given that the validation set consists of older products changing every year, it is reasonable for models to underperform in older years because they are not quite relevant anymore. Nevertheless, our chosen best model has the highest score in the validation set among all four models.

Feature	Importance %
Oldmodel	81%
Number of strikes for recent 5 years	14%
Violent crime /Total crime ratio for recent 10 years	1%

The most significant variables here are `oldmodel`(81%), `number of strikes for the recent 5 years` (14%), and `Violent crime over Total crime ratio for the recent 10 years` (1%). Other variables' contribution was less than 1%.



The graphs above show the Lorenz curves of Oldmodel and XGBoost with the `oldmodel` variable, for both Train_Full and Test_Full sample. The x-axis represents the percentage of policies weighted by earned exposure, and the y-axis represents the cumulative actual losses over the groups. The curve itself measures the actual percentage of losses attributed to a certain percentage of policies. We took a threshold of 80% of the company market in both samples and observed that our models have lower Lorenz scores. In the Train_Full sample, the actual loss percentage is 0.74 for Oldmodel and 0.52 for XGBoost; the risk is reduced by ~30%. In the Test_Full sample, the actual loss percentage for Oldmodel is 0.78 and 0.54 for XGBoost; the risk is reduced by ~31%.

AUC scores for our models have proven that our new variables can add predictive power to the existing company model. In the business context, the loss paid to reported damages is improved by 30% using our model. In other words, our new variables added to the model would adjust the pure premiums charged for clients; at the 80% threshold, the loss paid would decrease.

Criticism of the results and future work

One thing we kept in mind is that the company data itself was a sample from a bigger dataset; thus, the addition of the same variables to a bigger dataset may result in different AUC scores. Nevertheless, the results we got now have implied that these variables have a prediction power. With limited information, we cannot say how powerful quantitatively they would behave. In the future implementation, the additional power can be precisely measured with the company variables and its model.

When training models, we applied basic machine-learning models, mostly with default parameters. We realized then that adjusting some parameters, for example, the underlying distribution inside R functions, might give better results. Over 90% of the actual loss in the data are zeros, and the distribution between zero-loss and non-zero loss looks more like discrete rather than continuous. With the non-gaussian shape of dependent variables, changing the default Gaussian distribution inside R functions to other types (Poisson, Tweedie) might better catch the shape of data, hence improving the model fitting and prediction.

Besides, we noticed that different methods of feature selection bring different sets of variables. We used only one method - backward selection, though other methods (tree-based methods, for example) are likely to return different sets of variables. To that end,the results might shift a little with another set of inputs. Further, we trained with the initially selected 33 variables, while not all 33 were important based on the variable importance score from XGBoost. If we could re-train the model with only important ones, the scores would be expected to increase.

During the data exploration, we found other interesting datasets containing features, including earthquake events, solar radiation, and gas leaking. We inspected these data, and they hold some information with a likelihood to cause house damages. However, our preliminary data from the company only includes information in Northeastern five States, while these data, if projected to these States, might be less effective. As a result, we did not select those sources. In the future, if the data collection range would be expanded, these data sources can definitely be absorbed into the model.

Conclusion

Our main goal for the project was to find additional variables that might help to predict homeowner insurance loss and test their additional predictive power. According to our results, we have found some new features related to crime and lightning data, which do have predictive power by themselves, as well as carrying additional power when used together with existing model predictions. In terms of business application, our variables can reduce the risk of loss

payment by 30%. We acknowledge that our chosen model has constraints, but meanwhile, we believe that continuing work will optimize the variable usage.

Finally, we would like to extend our gratitude to the company representative and the MSBA program faculty and staff for the opportunity, guidance, and support along with the project.

Reference

https://www.investopedia.com/terms/h/homeowners-insurance.asp
https://www.investopedia.com/insurance/homeowners-insurance-guide/
https://www.investopedia.com/terms/c/catastrophe-insurance.asp
https://towardsdatascience.com/how-to-determine-the-best-model-6b9c584d0db4
https://www.math.arizona.edu/~jwatkins/R-01.pdf
https://towardsdatascience.com/feature-selection-with-pandas-e3690ad8504b
https://github.com/josephlee94/intuitive-deep-learning/tree/master/Part%201:%20Predicting%20House%20Prices
https://medium.com/analytics-vidhya/build-your-first-neural-network-model-on-a-structured-dataset-using-keras-d9e7de5c6724
https://keras.io/
https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114