Medical Cost Prediction

Madhura Prashant Vaidya, Rajkumar Baskar, Shreeyash Amit Yende
Khoury College of Computer Sciences, Northeastern University

Abstract

One of the very basic needs almost every individual needs access to is the Healthcare offered by their country. As such, a considerable portion of money gets spent on the insurance, which could become a potential problem if everyone has to pay same monthly premium. This project aims to determine a fair price that can be set by the insurance provider so that they are both affordable and curated to the needs of the individuals based on their medical history. In this project, a wide variety of regression techniques will be utilized to predict the cost, and their metrics will be compared.

Introduction

The United States' national health expenditure (NHE) grew 5.8% to \$3.2 trillion in 2015 (i.e., \$9,990 per person), which accounted for 17.8% of the nation's gross domestic product (GDP). With this increasing growth and the constant need for access to primary healthcare, predicting the healthcare cost for individuals as accurate as possible is one of the major challenges that could benefit more than one stakeholder, not limited to just the medical insurers. These forecasts would also greatly benefit the insured candidates since they could plan for their yearly medical costs that could arise due to unforeseeable incidents and choose the insurance type in advance wisely.

In this project, for medical cost prediction, we will be exploring in detail some of the regression models for prediction to evaluate best the medical cost for individuals based on their age, medical conditions, and other commonly available factors.

Background

Data prediction is one of the common tasks that possesses a high importance in almost all the fields. The predicted data could be utilized in many ways as to what to do or not do with it, by analyzing all the key and required aspects from it. Stock Markets, Retail stores and Weather Monitoring Systems are few of the major areas that widely use prediction on a day-to-day basis.

There are a variety of prediction models available, of which, the majorly explored in here are:

- Linear Regression
- Bayesian Regression

Linear regression is the process of associating a model's output in a linear relationship with the input features. A simple linear regression model assumes that a single input feature is linearly related to the output, so as to predict the outcome for any given value of the input. When there are multiple input features, the method is referred to as multiple linear regression.

Naive Bayes, also known as Simple Bayes or Independence Bayes, is a simple classifier / predictor, which makes use of the Bayesian theorem for prediction of the output. That is, all of the features contribute individually to the probability of being in a class, even if they are related.

Aside from these two models, there are other regression models that are widely used in the field of machine learning. Some of them include - Ridge Regression, Lasso Regression, SVR, Decision Tree Regression, Random Forest Regression, Extra Trees Regression and Gradient Boosting Regression.

Ridge regression addresses some of the problems of Ordinary Least Squares by imposing a penalty on the size of the coefficients. The ridge coefficients minimize a penalized residual sum of squares.

The Lasso is a linear model that estimates sparse coefficients. It is useful in some contexts due to its tendency to prefer solutions with fewer non-zero coefficients, effectively reducing the number of features upon which the given solution is dependent. For this reason, Lasso and its variants are fundamental to the field of compressed sensing.

Support Vector Regression depends only on a subset of the training data, because the cost function ignores samples whose prediction is close to their target. This is an extension of Support Vector Classification to solve regression problems. There are three different implementations of Support Vector Regression: SVR, NuSVR and LinearSVR. LinearSVR provides a faster implementation than SVR but only considers the linear kernel.

Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the features. A tree can be seen as a piece-wise constant approximation.

In random forests, each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. Furthermore, when splitting each node during the construction of a tree, the best split is found either from all input features or a random subset. The purpose of these two sources of randomness is to decrease the variance of the forest estimator. In practice the variance reduction is often significant hence yielding an overall better model.

In extremely randomized trees, randomness goes one step further in the way splits are computed. As in random forests, a random subset of candidate features is used, but instead of looking for the most discriminating thresholds, thresholds are drawn at random for each candidate feature and the best of these randomly-generated thresholds is picked as the splitting rule. This usually allows to reduce the variance of the model a bit more, at the expense of a slightly greater increase in bias.

Gradient Boosting Regression builds an additive model in a forward stage-wise fashion. It allows optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function.

Related Work

There are quite a few AI projects using both simple and complex supervised algorithms to achieve a better prediction capability, especially in the line of healthcare and insurance. However, depending on the ever increasing needs and demands, and newly arising medical conditions, the accurate prediction of premium proves to be a challenging task even today. There have been many projects utilizing various combinations of algorithms and inclusion of multiple features in order to better predict the cost for an individual's insurance, all depending on the data being utilized for the prediction task. Hence, there is no common algorithm or model that applies in general to the premium prediction of insurance data.

Methodology

Data Cleanup and Extraction

No data is perfect unless analyzed and cleaned up. As such, the dataset used here is no exception. The insurance dataset used here will be checked for missing values, duplicate entries and abnormalities / outliers. Also, any categorical feature will be converted to numeric feature since it is a prediction task, thus the need for transformation of data input.

Input Dataset

Here, we use the Kaggle dataset for insurance forecast, consisting of about 1338 records, each with seven features. This file is a direct copy of the Insurance.csv file from the Machine Learning course website: `http://cox.csueastbay.edu/~esuess/stat6620/#week-6` (Spring 2017) taught by Professor Eric Suess at `http://cox.csueastbay.edu/~esuess/classes/Statistics_6620/Presentations/ml10/insurance.csv`

The dataset contains seven features include – age, sex, BMI (body mass index), children count, smoking status, region, and premium/charges. The goal would be to accurately predict the charges for unseen data based on the given information

The dataset will be split into two with 75% training data and 25% testing data.

Model Training

There are two basic models with different supporting information that are explored. They are:

- Linear Regression model.
- Bayesian Regression model.

Along with these two, few other regression models are also used to compare and evaluate performances. They include:

- Ridge Regression
- Lasso Regression
- SVR
- Decision Tree Regression
- Random Forest Regression
- Extra Trees Regression and
- Gradient Boosting Regression.

Code

The ipython project / jupyter notebook file can be found at `https://github.com/rajkumar-b/medical-cost-prediction`. The code does not involve heavy external imports, except for seaborn (for plotting), pandas, numpy and scikit-learn (for ML models). All the other modules are readily available on installing Jupyter notebook for the first time.

The initial part of the code involves all the imports and a function to view/display pandas dataframes. Then the data set is viewed with just few records to get a glimpse of its structure. Then the data set shape and each feature's data types are obtained to have a general idea of what to explore.

The first part of the code involves data cleaning. This basically involves major two tasks - checking for null values and removing duplicates, both of which are performed here. An optional step of removing outlier can be performed, however ignored in this case.

The second part of the code involves data visualization and obtaining statistics. All the code are self explanatory with comments wherever required. This includes describing the input, dividing features into numerical and categorical types and viewing their statistics accordingly, where the former involves data distribution using histograms and boxplots, while the latter involves a pie-chart exploration. Also, the correlation matrix among all features is viewed as a heatmap. Finally, each feature is plotted against the destination column for prediction - charges, with age in the heatmap setting.

The third part involves observations from input and some useful deductions. The interesting factor being BMI against charges, with smoking status playing a key role. Also the charges distribution is explored for smokers and non-smokers as well, further down dividing by gender and comparing, only to find that the distributions are almost similar. Finally, the box-plot for charges is explored since it happens to have the most outliers, and none of the features individually contributing to the high volume.

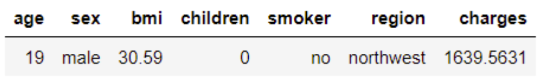
The fourth part of the code involves data preprocessing, where the data is split into training and testing set, and scaled using a min-max scaler.

The last and the most important part of the code involves the model training. The initial model exploration involves Bayesian regression and Linear regression models, both compared against each other, calculating various metrics. Post that, seven other regression models are taken, with few random initial settings and used with Grid Search to find out the best possible setting for that model. Post the analysis, they are used to create the best evaluation model for the considered dataset, which is then used to evaluate the test dataset and find the accuracy, to conclude on the best model among the available ones.

Experiments and Results

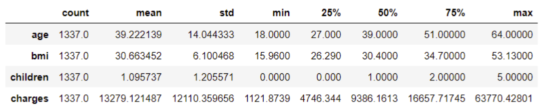
The main objective of this project is to predict the premium for an individual based on readily available information that could be utilized to find a pattern from the previous years' data. Here, we use the Kaggle dataset for insurance forecast, consisting of about 1338 records, each with seven features. The seven features include – age, sex, BMI (body mass index), children count, smoking status, region, and premium/charges.

During analysis, we found that there are no missing values. However, there was one duplicate entry, which was removed.



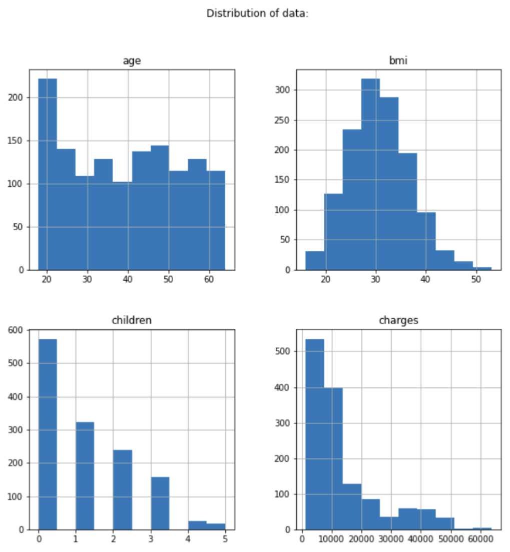
age	sex	bmi	children	smoker	region	charges
19	male	30.59	0	no	northwest	1639.5631

The numerical features of input data were described to get basic info.

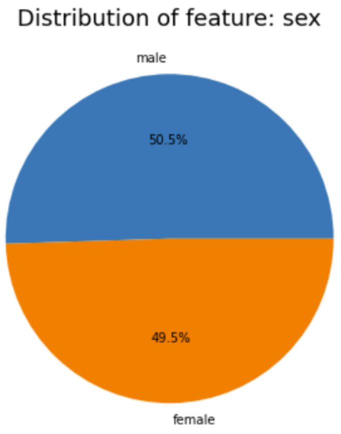


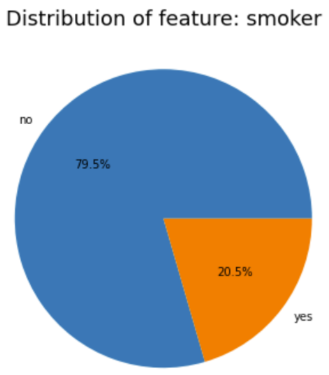
	count	mean	std	min	25%	50%	75%	max
age	1337.0	39.222139	14.044333	18.0000	27.000	39.0000	51.00000	64.00000
bmi	1337.0	30.663452	6.100468	15.9600	26.290	30.4000	34.70000	53.13000
children	1337.0	1.095737	1.205571	0.0000	0.000	1.0000	2.00000	5.00000
charges	1337.0	13279.121487	12110.359656	1121.8739	4746.344	9386.1613	16657.71745	63770.42801

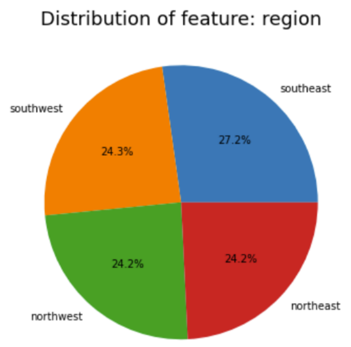
Also, corresponding distribution data and box plots were obtained.



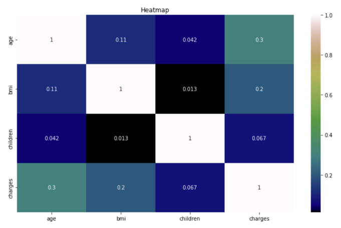
The categorical features were also explored as pie charts.







And the correlation between each feature is explored as a heat map.

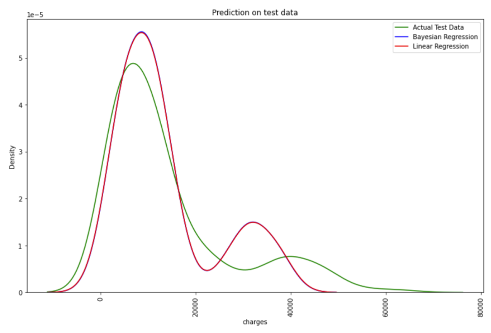


Basic Regression

The basic regression types such as Bayesian Regression and Linear regression are compared initially against each other and found to be behaving almost similar everywhere.

```
Bayesian Regressor

Training score : 0.7294309024962361
Testing score : 0.7958355932839731

Adjusted R2 Score - Training Data: 0.7277993300489772
Adjusted R2 Score - Test Data: 0.7921008785269726

Training data
MSE: 37225970.88231629
MAE: 4208.366784902875
RMSE: 6101.308948276287

Testing data
MSE: 35302029.73843685
MAE: 4063.000193051111
RMSE: 5941.551122260655


Linear Regression

Training score : 0.7294365233101179
Testing score : 0.7962578620326625

Adjusted R2 Score - Training Data: 0.7278049847572141
Adjusted R2 Score - Test Data: 0.7925308717039916

Training data
MSE: 37225197.548421815
MAE: 4210.575616626599
RMSE: 6101.245573522001

Testing data
MSE: 35229015.3273374
MAE: 4063.0773793105764
RMSE: 5935.403552188966
```

And the charge density plot also fall over one other.



As it can be seen above, both the models are almost overlapping with each other. Although the prediction curves are somewhat overlapping with the actual data, they are a bit squeezed on the X-axis, which is the reason for an accuracy of less than 80

Regression Comparison

The linear regression model is then compared against other models on various data parameters and finding out the best model for evaluation. The results are as below.

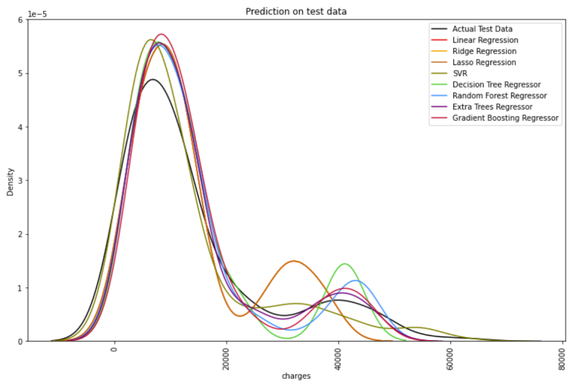
```
Best parameters for Linear Regression:  {'copy_X': True, 'fit_intercept': True, 'normalize': True}

Best parameters for Ridge Regression:  {'alpha': 0.001, 'fit_intercept': True, 'solver': 'sag'}

Best parameters for Lasso Regression:  {'alpha': 1, 'fit_intercept': True}

Best parameters for SVR:  {'C': 1000.0, 'gamma': 'scale', 'kernel': 'poly'}

Best parameters for Decision Tree Regressor:  {'max_depth': 5, 'max_features': 'auto', 'max_leaf_nodes': None, 'min_samples
_leaf': 2, 'min_weight_fraction_leaf': 0.1, 'splitter': 'best'}

Best parameters for Random Forest Regressor:  {'max_depth': 4, 'max_features': 'auto', 'n_estimators': 100}

Best parameters for Extra Trees Regressor:  {'max_depth': 6, 'max_features': 'auto', 'n_estimators': 300}

Best parameters for Gradient Boosting Regressor:  {'learning_rate': 0.03, 'max_depth': 5, 'n_estimators': 100, 'subsample':
0.5}
```

Result Interpretation and Discussion

The accuracy data of the compared regression models is given as below.

	Accuracy
Random Forest Regressor	0.885752
Extra Trees Regressor	0.884520
Gradient Boosting Regressor	0.880323
Decision Tree Regressor	0.864458
SVR	0.862101
Linear Regression	0.796258
Ridge Regression	0.796233
Lasso Regression	0.796148

It is clear that the basic regressors - linear, lasso and ridge, have an accuracy of only 79%, while the complicated regressors like random forest, extra trees and gradient boosting regressors have a much higher accuracy of about 88%. Decision Trees and SVRs lie in the middle with about 86% accuracy.



The density plot above shows how close the regressors fare against the actual test data. It is very evident that linear regressor has the worst performance among other regressors just by looking at the density plot. Note: This is only a density plot and not the actual predicted data. Hence, this is not a metric that is ideal for comparison of performances. It is just to visualize the distribution of the predicted charges for each of the regressors.

Future Work

The regression models used in this project cover the widely available popular models in the field of machine learning. However, with the increasing popularity of neural networks used everywhere, the concept of regression can be expanded to this area as well. A simple Feed Forward Neural Network with one or two hidden layers composed purely of linear and normalization functions might prove to be a better estimator than the basic regression techniques used here. The concept can be expanded to the area of Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) to achieve a better performance by mixing and matching various combination of neurons and layers. However, this is left for exploration in the future. This project can also be adjusted to the settings of machine learning methodologies on various medical health-related data sets in future using nature-inspired and meta-heuristic algorithms.

Conclusion

When it comes to utilizing historical data, machine learning (ML) is one component of computational intelligence that can use a variety of applications and systems to tackle a variety of problems. In the healthcare sector, predicting medical insurance costs is a challenge that has to be looked into and resolved. In this study, computational intelligence is used to forecast healthcare insurance expenditures utilizing a series of ML algorithms. The insurance data from KAGGLE repository was used here to mock the medical insurance dataset, which was then used to train and test the machine learning algorithms such as Linear Regression, Bayesian Regression, Ridge Regression, Lasso Regression, Support Vector Regression, Stochastic Gradient Boosting, Decision Tree, Random Forest Regression, and Extra Trees Regression. This data set went through preprocessing, feature engineering, data splitting, regression fitting and evaluation processes before being subjected to regression analysis. Out of the explored algorithms, Random Forrest Regression model had a high accuracy of 88.57%, according to the final results.

Additional Resources

The ipython project / jupyter notebook file can be found at `https://github.com/rajkumar-b/medical-cost-prediction` to invoke the above simulations and perform comparisons or to validate the results.

References

- National Health Expenditure Data. (2016). Retrieved from The Centers for Medicare & Medicaid Services (CMS): `https://www.cms.gov/research-statistics-data-and-systems/statistics-trends-and-reports/nationalhealthexpenddata`
- Agarwal, N., & Lahiri, B. (2014). Predicting healthcare expenditure increase for an individual from medicare data. Retrieved from Proceedings of the ACM SIGKDD Workshop on Health Informatics: `https://scholar.google.com/scholar_lookup?journal=Proceedings+of+the+ACM+SIGKDD+Workshop+on+Health+Informatics&title=Predicting+healthcare+expenditure+increase+for+an+individual+from+medicare+data&author=C+Lahiri&author=N+Agarwal&publication_year=2014&`
- Choi, M. (2018). Medical Cost Personal Datasets (insurance.csv). Retrieved from Kaggle: `https://www.kaggle.com/datasets/mirichoi0218/insurance`
- Morid, M. A., Kawamoto, K., Ault, T., Dorius, J., & Abdelrahman, S. (2018). Supervised Learning Methods for Predicting Healthcare Costs: Systematic Literature Review and Empirical Evaluation. Retrieved from National Library of Medicine: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5977561/#r1-2731392`
- Uzila, A. (2021). Medical Cost Prediction. Retrieved from Towards Data Science: `https://towardsdatascience.com/medical-cost-prediction-4876e3449adf`

Annex-A

All the members have equally contributed to the project. Highlighted are the areas where one had a significant contribution over others, however, each of the mentioned areas have been individually explored and analyzed by every person in this project group.

- Research on previous papers related to Healthcare and Insurance - Madhura Prashant Vaidya
- Identification of data set for the project - Shreeyash Amit Yende
- Topic analysis and abstract composition - Madhura Prashant Vaidya
- Research on related work and model choosing - Madhura Prashant Vaidya
- Detailed analysis on chosen models and parameter tuning - Shreeyash Amit Yende
- Data download, parsing and cleanup - Shreeyash Amit Yende
- Basic Exploratory analysis - Madhura Prashant Vaidya
- Statistics and Derivation - Rajkumar Baskar
- Seaborn plots and Analysis - Shreeyash Amit Yende
- Useful deductions and conclusions from data - Madhura Prashant Vaidya
- Linear and Bayesian Regression code compilation - Rajkumar Baskar
- Various regression model implementations and comparisons - Shreeyash Amit Yende
- Code accumulation and merge - Madhura Prashant Vaidya
- Report write-up and resource gathering - Madhura Prashant Vaidya, Rajkumar Baskar
- Slide preparation and Code compilation - Madhura Prashant Vaidya
- Video Presentation - Rajkumar Baskar