Spring 2019
CMPE 255
Data Mining

Project Report
San Francisco Fire Suppression Personnel Predictor

Team Kagglers

Presented to :
Dr. David Anastasiu

Group Members :
Anjali Bandaru (ID : 013854126)
Arkil Thakkar (ID :013825292)
Mrunali Sanjay Khandat (ID : 013723177)

Section 1. Introduction

1.1 Motivation

Emergency situations like fires can occur at any time. While there are certain things we can take care of to avoid the situation of a fire, we cannot control it. Effects of fire can be mitigated if it is stopped at its origin. Since fires spread rapidly, the time frame to act is small. The narrow window of time needs to be utilized by firefighters to tackle the situation. The damage caused by fires is manyfold- loss of human lives, property loss, environmental damage.
The only way to mitigate the effects of fire is by sending help at the right time. If help is sent at the right time then not only all human lives can be saved, but property loss can also be minimized. After getting a call for fire rescue, the fire department sends its firefighters- fire suppression personnel immediately. But how many people are to be sent remains unclear. At times, firefighters reach at the emergency situation and then ask for extra help. But by the time help arrives fire has already spread. And then the situation goes out of control causing many civilian and firefighters fatalities, injuries; not to mention property damage.
We chose San Francisco Fire Incidents dataset to predict the number of fire suppression personnel to be sent in case of a fire emergency. Sending the correct number of firefighters will stop the fire spread with almost no loss of human lives, minimum property damage and less damage to the environment as well.

1.2 Objective

San Francisco fire dataset comprises of data of prior fire incidents in the city of San Francisco. The aim is to analyze the dataset and predict the number of fire suppression personnel to be sent in case of fire situation depending upon the parameters like the area of fire origin, property use of the affected building, primary situation.

Section 2. System Design & Implementation Details

2.1 Algorithm(s) considered/selected (and why)

<u>Lasso Regression-</u>
Our dataset consisted of 63 columns and over 490K instances. We aim at predicting the fire suppression personnel using this data. After preprocessing, we reduced our columns to over 50% of the current columns. When we plotted correlation of all the remaining columns with ‘Suppression Personnel’, not all the columns were strongly associated with ‘Suppression Personnel’. Since Lasso regression discards the variables not strongly associated with the predictor variable, we decided to go for Lasso Regression.

<u>Gradient Boosting-</u>

It is observed that Gradient Boosting algorithms give good results in all Kaggle competitions. It is also a robust algorithm that gives results by combining results from weak learners too. That's why we decided to implement a Gradient Boosting Regression algorithm to check its performance on our dataset.

Random Forest Regression-
Because of the large size of the dataset, Random forest algorithm was one of our first choices. After preprocessing the data, we realized that there are only a few columns that were leading our prediction. This would introduce a bias in our results. To reduce this bias and variance, Random forest algorithm was chosen.

Ridge Regression-
Ridge regression is an extension of linear regression that includes bias in it. Ridge regression was chosen for its simplicity. We imputed missing values in the data. If a lot of missing values are imputed then multicollinearity possibility of the dataset increases. Ridge Regression is the best solution to avoid multicollinearity situation.

2.2 Technologies & Tools Used

Conda
Conda is used to install libraries and packages required for the execution of Python scripts. It is easy to use

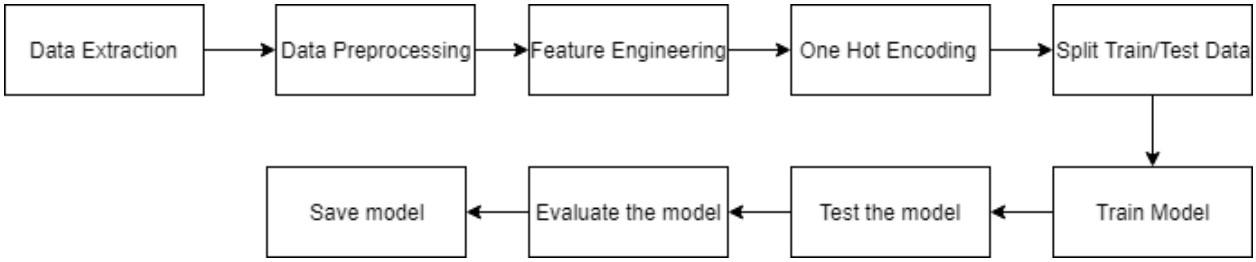
Python
We used the Python programming language to implement our project.

Jupyter Notebook

Libraries
- SciPy
- NumPy
- Pandas
- Scikit-learn
- Matplotlib

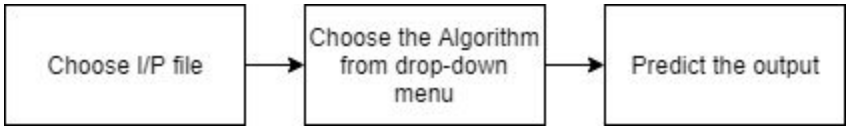
2.3 System Workflow

System workflow for our project is shown in the diagram below. This workflow was followed for all the algorithms implemented.

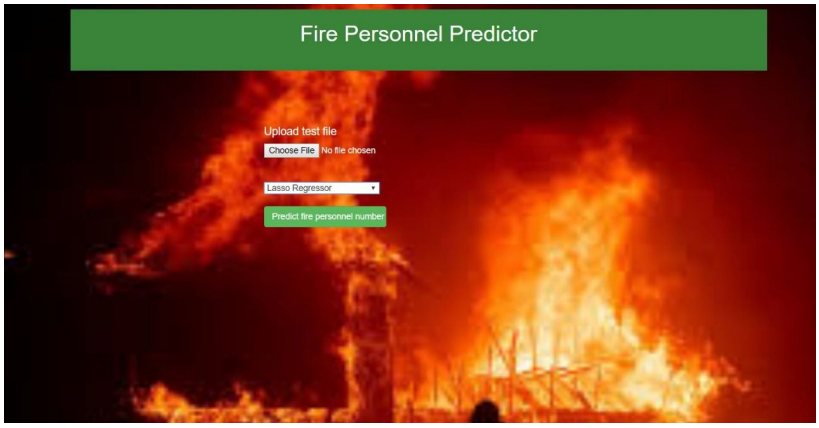


- While actually executing the project, we went from ‘Evaluate the model’ to ‘Data Preprocessing’ for many times until we got a satisfactory result for our models.

- To predict the output using UI,



2.4 GUI Screenshots



Section 3. Experiments

3.1 Dataset

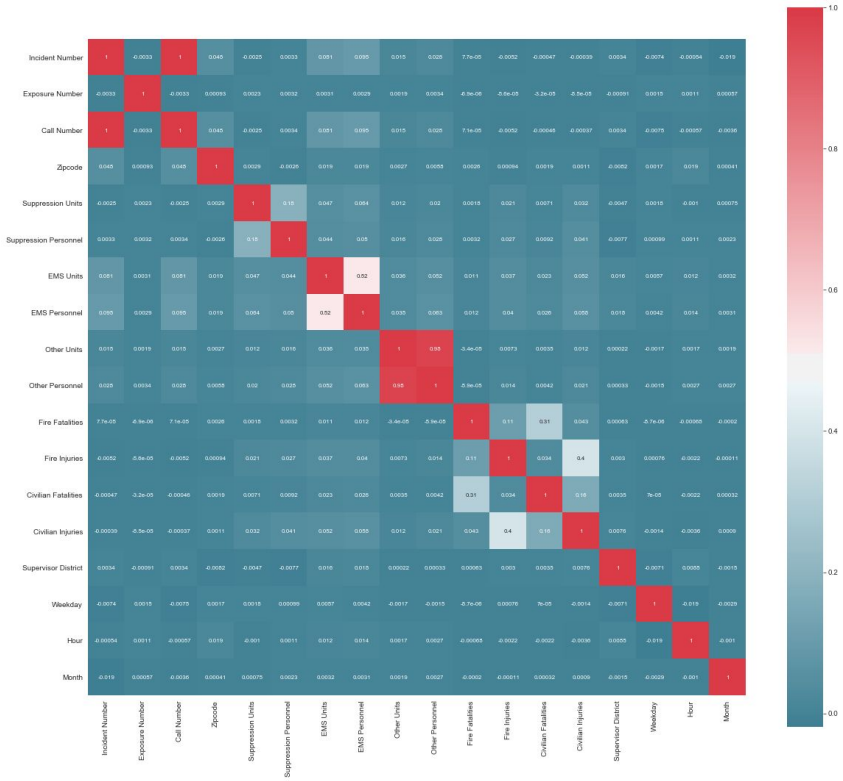
The dataset used is Fire Incidents in the city of San Francisco.

- The link for a dataset: https://data.sfgov.org/Public-Safety/Fire-Incidents/wr8u-xric
- It has 490K instances, each containing 63 attributes.
- The data is in tsv format.
- It describes fire incidents that the department responded to, giving details like the call number, incident number, address, number and type of each unit responding, call type

(as determined by dispatch), prime situation (field observation), actions taken, and property loss.

3.2 Data Preprocessing

- Explore the dataset and drop the columns with no relevance to the problem statement
- Out of the remaining columns, drop the columns with less than 50% instances
- Impute the missing values in remaining columns, using interpolate library function
- Convert categorical variables in important columns to numerical variables. For example, 'Action Taken Primary' column contains values such as "86 - investigate"; these values are converted to 86.
- In these important columns, define another integer to categorize unknown values. For example, defined '650' as value for unknown values in the 'Primary Situation' column
- Convert 'Alarm DtTm' column in 'Weekday', 'Hour' and 'Month'.
- Use backward fill method to fill all 'Nan' values per column.
- Define a function that checks for columns with missing values and returns the count of missing values per column.



3.3 Methodology

<u>Lasso Regression-</u>

After preprocessing of data, one hot encoding was used on the data to convert categorical variables. Output variable in our case- ‘Suppression Personnel’ is stored in another data frame. ‘Suppression Personnel’ is then dropped from the dataset. The dataset was then converted to

the sparse matrix- CSR. After that, the dataset was split into training and test dataset. We have defined a function 'lasso' that takes training dataset, test dataset, training output and test output as input. In this method, the Lasso Regression model is trained using alpha as 0.1. This model is then fitted to the training dataset. Using test dataset as input, predicted 'Suppression Personnel' is found out. Mean Squared Error of the model is then calculated. K-fold cross-validation is applied to the training dataset and training output to calculate cross-validation scores. The value of K is 10. The method 'lasso' returns cross-validation scores and Mean Squared Error(MSE) of the model. Mean Squared Error using Lasso Regression technique was 33.98.

Gradient Boosting-

Preprocessing removes columns with less meaningful data, columns with missing values, duplicate data values. After preprocessing One Hot Encoding is used to make categorical variable columns 'Zipcode' and 'Battalion' into numerical variable columns. This increases the number of columns of the dataset. The dataset is then split into output data and remaining dataset. The remaining dataset is converted to sparse CSR matrix. This dataset is then split into training and testing dataset in the ratio of 80:20. Gradient Boosting Regressor model is then trained using various values of the parameters- 'n_estimators', 'learning_rate', 'max_depth', 'random_state' and 'loss'. By tuning these parameters to get better results, we got Mean Squared Error of 28.01.

Ridge Regression-

Once PreProcessing done, converted to data frames to CSR matrix. Tried using SVD to reduce the dimension of the dataset. Split the data into train and test(validation) in 80:20 ratio. Applied Ridge regression on the matrix obtain. Used different values of Alpha to tune the parameters. Used K-Fold Cross Validation to split data into train/test sets for k consecutive folds. Calculated Cross_val_score for k-folds and compared them. Used MSE as a metric to evaluate the performance of the model.

Random Forest Regression-

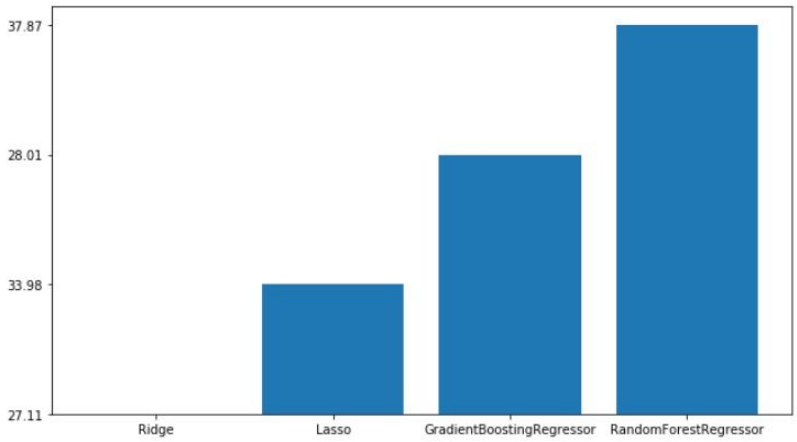
After preprocessing, some categorical value columns are converted into numerical variables. ‘Suppression Personnel’ column is dropped from the dataset and stored as training output. The dataset is converted to CSR matrix. The CSR matrix and training output is split in training and test data. Random Forest Regressor model is prepared, trained and tested using training and test data, training and test output data. MSE for this algorithm turned out to be 37.87.

Display the results on User Interface-

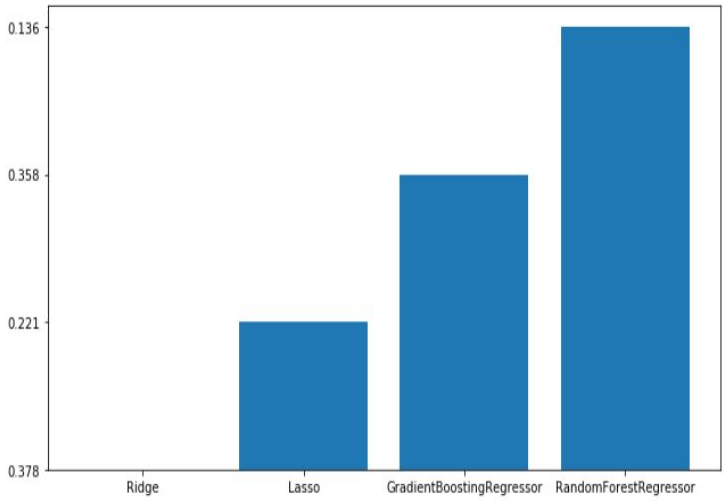
After running each algorithm, its model is saved as a .pkl file. On UI, a file is given as input. This input file contains some test instances. These test instances are raw data. These are preprocessed in the background using the same preprocessing methods. On the UI, there is an option to select an algorithm to predict the result. On selecting a particular algorithm, its model gets loaded and the results are displayed on the UI.

3.4 Results

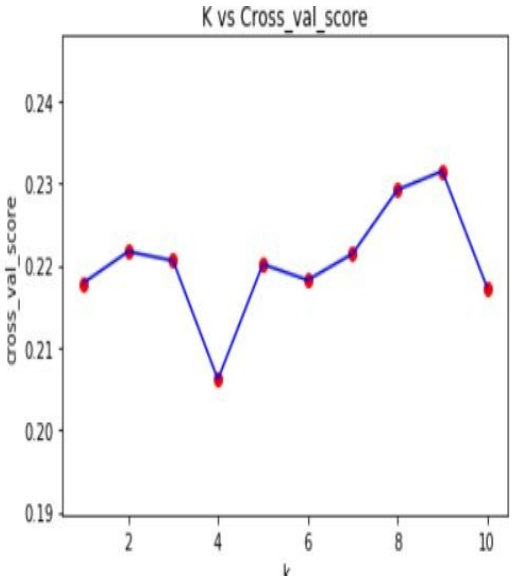
- Following bar chart shows Mean Squared Errors of all the algorithms we implemented



- Following bar chart shows R-2 score of all the algorithms we implemented-



- Following line chart shows Cross Validation Scores of Lasso Regression for different values of K. K here is 10.



3.5 Analysis of Results

- We used Mean Squared Error as a parameter to evaluate algorithms.
- Best algorithm- With 27.11 as Mean Squared Error, Ridge Regression turned out to be the best algorithm.
- Worst algorithm- With 37.87 as Mean Squared Error, Random Forest Regression algorithm was the worst method in terms of evaluation parameter. Random Forest Regression also took more time for execution than other algorithms.
- We also tried these algorithms with and without One Hot Encoding. There was not significant change in MSE in both methods.

Section 4. Discussion & Conclusion

4.1 Decisions Made

- Preprocessing approach needed to be changed. The first MSE score we got was approximately 1176 with Random Forest Regression. After adding a few important columns, we got it to 297. With more focus on improving preprocessing, final MSE for Random Forest Regression was 33.98.
- We tuned the output variable- ‘Suppression Personnel’. Values of ‘Suppression Personnel’ from 1 to 100 were only considered. Some of the outliers had value of 3000.
- We decided to convert data to sparse CSR matrix instead of scaling it.
- Columns relevant to the problem statement from the original dataset were decided.

4.2 Difficulties Faced

- The dataset contained many missing values, outliers that made preprocessing difficult and timetaking.
- The model for prediction, even after saving the model as .pkl file, executed during runtime. This was solved later by splitting preprocessing method and algorithm implementation.
- As compared to standard scaler, operations on CSR matrix took a lot of time.
- Earlier preprocessing and Random Forest Regression model gave MSE of 1176.

4.3 Things That Worked

- MSE improved significantly after using CSR matrix
- Preprocessing change gave change of MSE from 1176 to 37.87 for Random Forest Regression
- Splitting preprocessing as a method and importing it before implementing algorithm

4.4 Things That Didn't Work

- Standard scaler didn't work
- SVD didn't work, change of MSE from 27 to 39 in Ridge Regression
- Changing the value of alpha did not give a significant improvement in MSE

4.5 Conclusion

- Hypertuning of parameters could provide better results.
- Dataset exploration and preprocessing are more important. Even if these are the first steps of the data science problem solving, this needs to be revisited again and again to improve accuracy.
- Clean dataset is important before beginning data mining.

Section 5. Project Plan & Task Distribution

5.1 Who was assigned to what task

- We had decided to split our work after preprocessing initially.
- After preprocessing, everyone will select one algorithm of their choice and use Mean Squared Error as evaluation metric to get results.

5.2 Who ended up doing what task

- After implementing algorithms once, we realized that we needed to preprocess data even more. Then each of us followed our own approach for data exploration and preprocessing.
- We then shared our preprocessing ideas with each other, after improving on MSE.
- Following table shows what each of us ended up doing-

Task	Subtask	Assignee
Project Topic	Each of us came up with one topic	All
Data Analysis		All
Data Preprocessing		All
Regression Algorithms	Ridge	Arkil
	Random Forest	Anjali
	Lasso	Mrunali
	Gradient Boosting	All
UI Part		All
Report		All
Presentation		All

Section 6. References

https://www.fireapparatusmagazine.com/articles/print/volume-17/issue-2/departments/chief-concerns/fire-departments-most-important-function.html

https://work.chron.com/importance-firefighter-19431.html

https://hackernoon.com/practical-machine-learning-ridge-regression-vs-lasso-a00326371ece

https://www.kaggle.com/rnepal2/complete-solution-analysis-and-regression-models