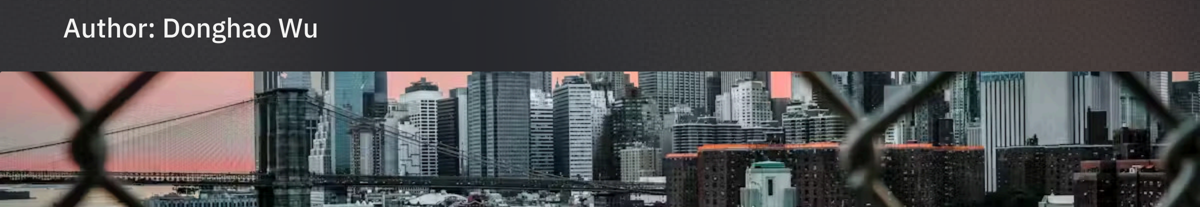
Crime Data & Machine Learning

Author: Donghao Wu



Overview & Introduction

> *Studies have shown that higher levels of inequality, unemployment, and poverty are strongly associated with increased crime rates (de Nadai et al., 2020). Beyond socio-economic factors, research has also highlighted the impact of environmental conditions, such as temperature. McCord and Ratcliffe (2018) found that higher temperatures contribute to an increase in aggressive crimes. Additionally, studies suggest that higher rates of gun ownership are linked to increased violent crime (Bhattacharya, 2020)*

Obejctive of the Project

- Predict the number of crime incidents using various machine learning models
- Compare model performance and identify potential improvements to the models

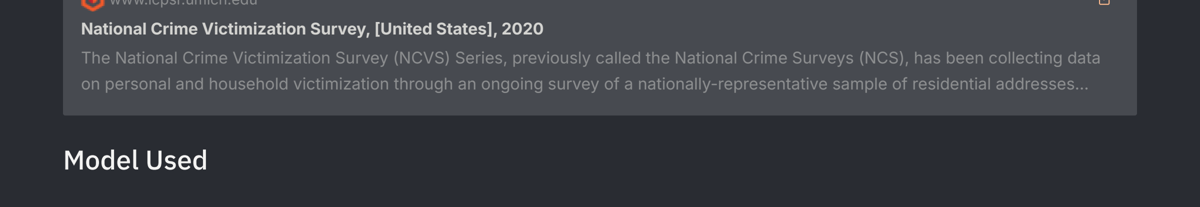
DataSet

- **Source:** National Crime Victimization Survey (NCVS)
- **8043** Observations + **81** features

www.icpsr.umich.edu
National Crime Victimization Survey, [United States], 2020
The National Crime Victimization Survey (NCVS) Series, previously called the National Crime Surveys (NCS), has been collecting data on personal and household victimization through an ongoing survey of a nationally-representative sample of residential addresses...

Model Used

- **Linear Regression + L1 & L2 Regularization**
- **Decision Tree Regressor**
- **Random Forest Regressor**



Data Exploration & Model Building

Full data cleaning and manipulation can be seen in the notebook, which include recode, outlier handling, and regular data cleaning process

Feature Selection

```
selected_features = ["URBANICITY", "GATED", "MARITAL", "AGE", "SEX", "RACE", "ED", "INCOME", "REGION", "WEAPON", "NUM_INCIDENTS"]
```

Target Variable:

NUM_INCIDENTS: Numeric value, representing the number of incidents happened to an individual in the given year.

The features I select from the data includes:

- URBANICITY : includes three different types of living area (Urban, Suburban, and Rural)
- GATED : represents if the individual lives in a community with a gate or not
- MARITAL: represents individual's marital status (1-Married 2-Widowed 3-Divorced 4-Separated 5-Never married)
- AGE: The age of the individual
- SEX: Binary classification (male or female)
- RACE: Race of the individual, which includes 17 different options, detailed description will be in the code notebook
- ED: Education of the individual
- INCOME: 17 different income levels
- REGION: Geographic region of the individual (e.g. East, West)
- WEAPON: Whether the incident involved weapon

	URBANICITY	GATED	MARITAL	AGE	SEX
count	8043.000000	8043.000000	8043.000000	8043.000000	8043.000000
mean	1.965063	1.906503	2.912843	45.248415	1.539351
std	0.567632	0.291146	1.775581	16.952285	0.498480
min	1.000000	1.000000	1.000000	12.000000	1.000000
25%	2.000000	2.000000	1.000000	31.000000	1.000000
50%	2.000000	2.000000	3.000000	44.000000	2.000000
75%	2.000000	2.000000	5.000000	58.000000	2.000000
max	3.000000	2.000000	8.000000	90.000000	2.000000

	RACE	ED	INCOME	REGION	WEAPON
count	8043.000000	8043.000000	8043.000000	8043.000000	8043.000000
mean	1.543827	35.419993	11.844088	2.815989	1.900908
std	1.606739	11.443292	4.443971	1.019681	0.298805
min	1.000000	0.000000	1.000000	1.000000	1.000000
25%	1.000000	28.000000	9.000000	2.000000	2.000000
50%	1.000000	40.000000	13.000000	3.000000	2.000000
75%	1.000000	42.000000	15.000000	4.000000	2.000000
max	20.000000	98.000000	18.000000	4.000000	2.000000

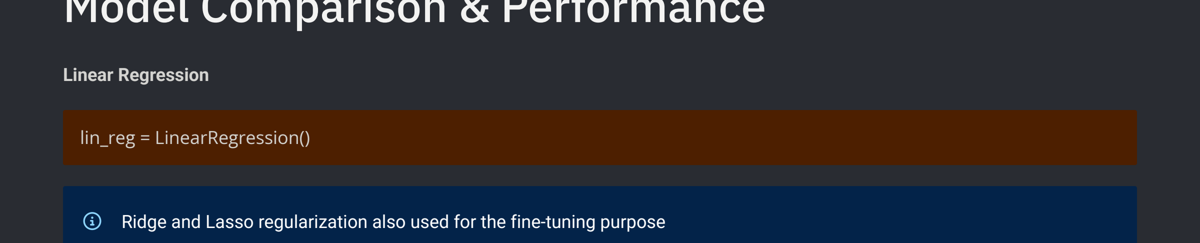
	NUM_INCIDENTS
count	8043.000000
mean	1.611463
std	1.208176
min	1.000000
25%	1.000000
50%	1.000000
75%	2.000000
max	10.000000

EDA of the features and target variable

Train-Test-Validation Split

```
# Split dataset into training (70%), validation (15%), and testing (15%)
train_ncvs, temp_ncvs = train_test_split(ncvs_new_encoded, test_size=0.3, random_state=42)
val_ncvs, test_ncvs = train_test_split(temp_ncvs, test_size=0.5, random_state=42)

#output
{'Training Set': (5577, 40), 'Validation Set': (1195, 40), 'Test Set': (1196, 40)}
```



Model Comparison & Performance

Linear Regression

```
lin_reg = LinearRegression()
```

Ridge and Lasso regularization also used for the fine-tuning purpose

Decision Tree Regressor

```
dt_reg_ase = DecisionTreeRegressor(random_state=42, criterion="absolute_error", max_depth = 30)
```

Changed criterion to "sqaure-error" and "poisson" as well as change the max_depth for fine-tuning purpose

Random Forest Regressor

```
RandomForestRegressor(random_state=42, n_estimators=500, criterion="poisson")
```

Also changed criterion, n_estimator for fine-tuning purpose

Statistical Performance Comparison (training and validation data)

Model	Performance(higher is better)	MSE(lower is better)	R-square(higher is better)
Linear Regression	0.088	0.602	0.077
Decision Tree Regressor(ASE)	0.99	0.378	0.421
Random Forest Model	0.948	0.232	0.645

Final Model Selection

Random Forest Model wins the competiton !!!

Reason:

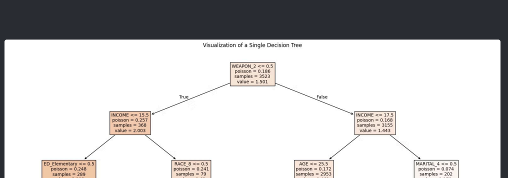
- Linear regression model's performance and all relevant statistics are quite far away from satisfying, probably due to the nature that linear regression model can only capture linear relationship, which is not suitable in this project
- Decision tree model even though improves a lot of the performance, MSE, and R-square score, it remains the overfitting problem, which can be solved by using the random forest model
- With the random forest model, it not only resolve the problem of overfitting, but also improved relevant statistics, which seems like the best model for the data

Result Analysis

> After determining the final model for the data, it is finally the time to touch test data

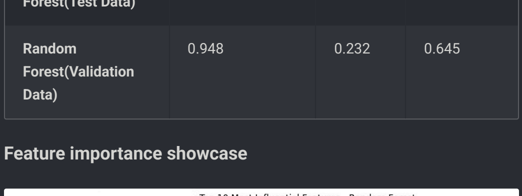
```
final_model = RandomForestRegressor(random_state=42, n_estimators=500, criterion="poisson")
```

```
y_pred_final = final_model.predict(X_test)
```



Model	Performance	MSE	R-Square Score
Random Forest(Test Data)	0.948	0.208	0.692
Random Forest(Validation Data)	0.948	0.232	0.645

Feature importance showcase



Error Analysis

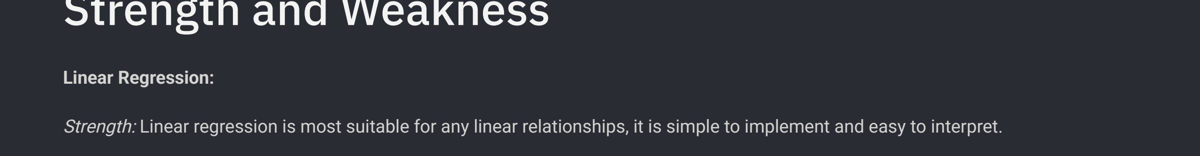
> Apparently, none of the model is going to be perfect, especially in my project

Even though the final model chose in this project is performing the best among all the three models, it still has some potential issues

Error Comparison Across Models:

	Model	MSE	R² Score
0	Linear Regression	0.602	0.077
1	Decision Tree Regressor	0.412	0.368
2	Final Random Forest	0.208	0.692

	Actual	Predicted	Error	Model
3058	1.0	3.4985	2.4985	Linear Regression
1019	1.0	3.4970	2.4970	Linear Regression
2717	1.0	3.4950	2.4950	Linear Regression
7207	1.0	3.4950	2.4950	Linear Regression
2369	3.5	1.0060	2.4940	Linear Regression
2590	3.5	1.0000	2.5000	Decision Tree Regressor
2245	1.0	3.5000	2.5000	Decision Tree Regressor
6769	3.5	1.0000	2.5000	Decision Tree Regressor
1917	3.5	1.0000	2.5000	Decision Tree Regressor
694	1.0	3.5000	2.5000	Decision Tree Regressor
809	3.5	1.1640	2.3360	Random Forest Regressor
810	3.5	1.1640	2.3360	Random Forest Regressor
811	3.5	1.1640	2.3360	Random Forest Regressor
1384	1.0	2.7390	1.7390	Random Forest Regressor
1327	3.0	1.4340	1.5660	Random Forest Regressor



Strength and Weakness

Linear Regression:

Strength: Linear regression is most suitable for any linear relationships, it is simple to implement and easy to interpret.

Weakness: In this project, linear regression will assume a linear relationship between the target variable and all the features selected, which is not the case in for my dataset. Therefore, the result obtained from linear regression will underfit the data and lead to high error.

Decision Tree Regressor:

Strength: Unlike linear regression model, decision tree regressor can capture a more complicated relationship between target variable and features. Unlike random forest, decision tree regressor is easier to visualize and understood, since there will be only one tree for the relationship.

Weakness: The weakness for decision tree regressor is that it is very likely to generate a overfitting model before regularization. Even after regularization, the model will not be generalizable enough for all the testing method. This will result in an underfitting problem

Random Forest:

Strength: Random forest model in this project has the lowest error and highest R-square score, which indicates that the model fits the data and predicts the relationship much better than other models. Moreover, due to the nature of random forest, it is more generalizable than otehr models and reduce the overfitting problem.

Weakness: However, the computing expense for random forest is much higher than the other two models and the training process will be rather long if the dataset is too big. Moreover, unlike decision tree regressor, random forest model is harder to interpret because the randome forest model is more like a black box model. Finally, the random forest model is nearly impossible to visualiza, since there will be hundreds of trees in the process.

Improvements

Linear Regression:

- **Feature Engineering:** I can add an interaction term in the model such as(AGE * INCOME) to improve the model. Moreover, using log transformation can also handle the skewed distributions.
- **Polynomial Regression:** Furthermore, I can use quadratic or cubic terms to model non-linearity, this should partly solve the issue that the linear regression model can only capture the linear relationship.
- **Remove Outlier:** Linear models are sensitive to outliers, identify and remove outliers from the dataset should improve the performance of the model

Decision Tree Regressor:

- **Pre-prune and post-prune the tree:** To avoid overfitting, I can use post or pre-pruning to resolve the issue. For example, I can set restrictions to max_depth, min_samples_leaf to prevent deep trees.
- **Feature Selection:** Identify and remove irrelevant features to prevent the tree from memorizing the noise

Randome Forest Regressor:

- **Find the Balance:** adjuest the n_estimator to improve the speed of computing, without reduce the performance of the model drastically. Increase min_samples_split to prevent deep tree.
- **Consider boosting methods:** There are some boosting methods such as Gradient boosting, XGboost for potentially better accuracy.

Feedback from the lab

Should recode some of the variabls in the dataset, some may use one-hot coding, some use ordinal coding