CPSC-6300 Project:Insurance Fraud Detection

Dineshchandar Ravichandran [1*] **Archana Lalji Saroj** [1*] **Prashanth Reddy Kadire** [1*]
[1]Clemson University
{dravich,asaroj,pkadire}@g.clemson.edu

1 Introduction & Problem statement

The insurance industry consists of over 7,000 companies that collect over $1 trillion in premiums yearly. The massive size of the industry contributes significantly to the cost of insurance fraud by providing more opportunities and bigger incentives for committing illegal activities. The total cost of insurance fraud (non-health insurance) is estimated to be more than $40 billion annually. That means Insurance Fraud costs the average U.S. family between $400 and $700 per year in the form of increased premiums. Insurance fraud also steals at least $308.6 billion yearly from American consumers. For this project, we will focus on Automobile-insurance fraud where 25%-33% of insurance claims have an element of fraud. Automobile claim fraud and buildup added $5.6 billion-$7.7 billion in excess payments to auto-injury claims paid in the U.S. in 2012. 21 % of bodily injury (B.I.) claims and 18 % of personal injury protection (PIP) claims closed with payment had the appearance of fraud and/or buildup. Buildup involves inflating otherwise legitimate claims. The government and other organizations are responding to this by investing in technology to detect fraudulent claims, 21% of insurance institutes plan to invest in A.I. (Artificial Intelligence) in the next two years.

2 Motivation Goals

We will construct a supervised machine learning model-based Fraud Detection system to predict the chances of a fraudulent insurance claim. This system aims to analyze the insurance claim data set containing 38 distinctive features and generate a classification logic to identify genuine and fraudulent claims. By constructing such a fraud detection system, we can alert the insurance institutes and allow them to take necessary action against fraudulent claims.

3 Data Exploration

3.1 Data Summary

We collected the insurance claims data set from Kaggle "insurance_claims_data." This data set comprises 1000 total data points. The data set is imbalanced since there are a total of 247 fraud claims and 753 genuine claims, according to the preliminary data exploration.

3.2 Unit of Analysis

For this project, the accuracy, Precision, and F1 Score will be calculated on each model. These measures are considered as our unit of analysis for selecting the best model.

- **F1 score**: This is the harmonic mean of precision and recall and gives a better measure of the incorrectly classified cases than the accuracy matrix.

$$F1 - score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}$$

- **Precision**: It is implied as the measure of the correctly identified positive cases from all the predicted positive cases. Thus, it is useful when the costs of False Positives are high.

$$Precision = \frac{TruePositive}{(TruePositive + FalsePositive)}$$

- **Accuracy**: One of the more obvious metrics is the measure of all the correctly identified cases. It is most used when all the classes are equally important.

$$Accuracy = \frac{TruePositive + TrueNegative}{(TruePositive + FalsePositive + TrueNegative + FalseNegative)}$$

3.2.1 Observations in data set

We have a data set containing 1000 insurance claims and 38 features pertaining to it. In addition, we also have a column stating which of these insurance claims are fraudulent and which are genuine, as illustrated below:

3.2.2 Unique observations

In our dataset, every claim is unique, though we do not have a unique identifier like claim I.D., the policy numbers for all the claims are unique.

3.2.3 Time period is covered

Our dataset contains claims with incidents occurring from 1st Jan 2015 to 1st March 2015.

4 Data cleaning

Three columns had '?' Missing value:

1. Collision Type.
2. Property Damage.
3. Police Report Available.

The missing values are imputed using KNN imputer. The categorical features have been transformed into numerical features using one-hot encoding and label encoding.

```
df_insurance.shape
✓ 0.5s
(1000, 39)
```

Figure 1: Sample size

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip	...	witnesses	police_report_available
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132	...	2	YES
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176	...	0	?
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632	...	3	NO
3	256	41	227811	1990-05-25	IL	250/500	2000	1415.74	6000000	608117	...	2	NO
4	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91	6000000	610706	...	1	NO
5	256	39	104594	2006-10-12	OH	250/500	1000	1351.10	0	478456	...	2	NO
6	137	34	413978	2000-06-04	IN	250/500	1000	1333.35	0	441716	...	0	?
7	165	37	429027	1990-02-03	IL	100/300	1000	1137.03	0	603195	...	2	YES
8	27	33	485665	1997-02-05	IL	100/300	500	1442.99	0	601734	...	1	YES
9	212	42	636550	2011-07-25	IL	100/300	500	1315.68	0	600983	...	1	?

total_claim_amount	injury_claim	property_claim	vehicle_claim	auto_make	auto_model	auto_year	fraud_reported
71610	6510	13020	52080	Saab	92x	2004	Y
5070	780	780	3510	Mercedes	E400	2007	Y
34650	7700	3850	23100	Dodge	RAM	2007	N
63400	6340	6340	50720	Chevrolet	Tahoe	2014	Y
6500	1300	650	4550	Accura	RSX	2009	N
64100	6410	6410	51280	Saab	95	2003	Y
78650	21450	7150	50050	Nissan	Pathfinder	2012	N
51590	9380	9380	32830	Audi	A5	2015	N
27700	2770	2770	22160	Toyota	Camry	2012	N
42300	4700	4700	32900	Saab	92x	1996	N

Figure 2: Sample head

5 Data visualization

5.1 Description of the dataset:

Column Name	Data Type	Description
months_as_customer	int64	No.of months customer has been a customer to the insurance organization.
age	int64	Age of the customer.
policy_number	int64	Policy no.of the customer's insurance.
policy_bind_date	object	The moment when the insurance coverage goes into force, it's date and time sp
policy_state	object	State in which the policy was procured.
policy_csl	object	Combined single limits are a provision of an insurance policy limiting coverag
policy_deductable	int64	The amount customers have to pay for covered services before their insurance
policy_annual_premium	float64	Annual payment for the policy.
umbrella_limit	int64	Extra insurance that provides protection beyond existing limits and coverages
insured_zip	int64	N.A
insured_sex	object	Male/ Female.
insured_education_level	object	Education level of the customer.
insured_occupation	object	Occupation of the customer.
insured_hobbies	object	Customers' hobbies.
insured_relationship	object	Relationship of the person involved in the incedent to the actual insurance hold
capital-gains	int64	Increase in a capital asset's value.
capital-loss	int64	Loss in a capital asset's value.
incident_date	object	Date of incident.
incident_type	object	Type of incident (Single Vehicle Collision, Vehicle Theft, Multi-vehicle Collis
collision_type	object	Type of collision (Side Collision, Rear Collision, and Front Collision).
incident_severity	object	The severity of the incident classified as per damage (Major Damage, Minor D
authorities_contacted	object	Type of authorities contacted after the incident.
incident_state	object	U.S. state where the incident occurred.
incident_city	object	City in which the incident occurred.
incident_location	object	Address of the incident.
incident_hour_of_the_day	int64	Time of incendent in 24hrs.
number_of_vehicles_involved	int64	No.of vehicles involved with the incident.
property_damage	object	If 'YES,' then the insurance carrier helps pay to repair the damage customer ca
bodily_injuries	int64	No.of bodily injuries.
witnesses	int64	No.of witnesses to the incident.
police_report_available	object	If a police report exists of the incident.
total_claim_amount	int64	The total amount claimed by the customer fot the incedent.
injury_claim	int64	The portion of the total claim requested for the injury claim.
property_claim	int64	The portion of the total claim requested to pay for property damages.
vehicle_claim	int64	The portion of the total claim requested for vehicle damage.
auto_make	object	Manufacturer of the customer's vehicle that was involved in the incident.
auto_model	object	The specific automobile model of the customer's that was involved in the incid
auto_year	int64	Model year
fraud_reported	object	Determining whether a claim is fraudulent or not.

5.1.1 Point of interest data distribution

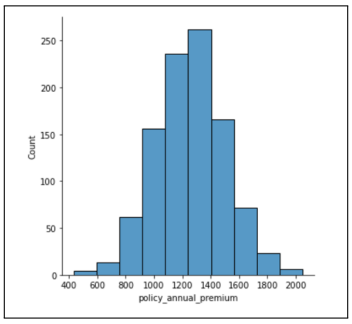

Figure 3: Annual policy premium distribution

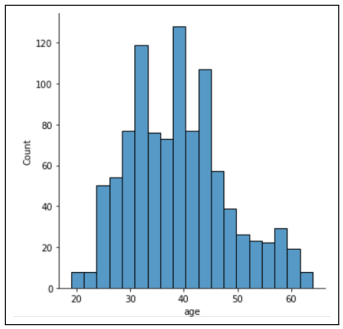

Figure 4: Age distribution of policyholders

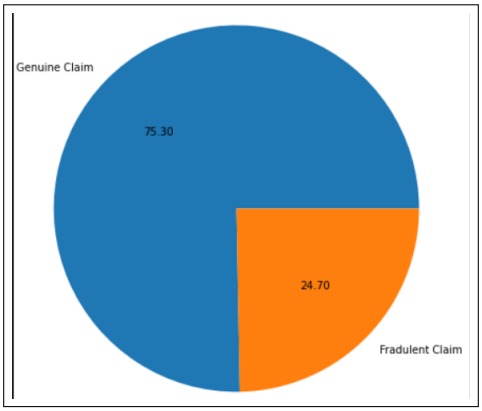

Figure 5: Genuine claims VS Fraudulent claims comparisons

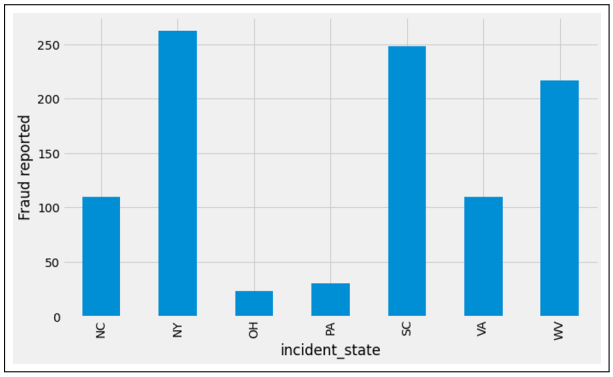

Figure 6: Incident state-wise fraudulent claims

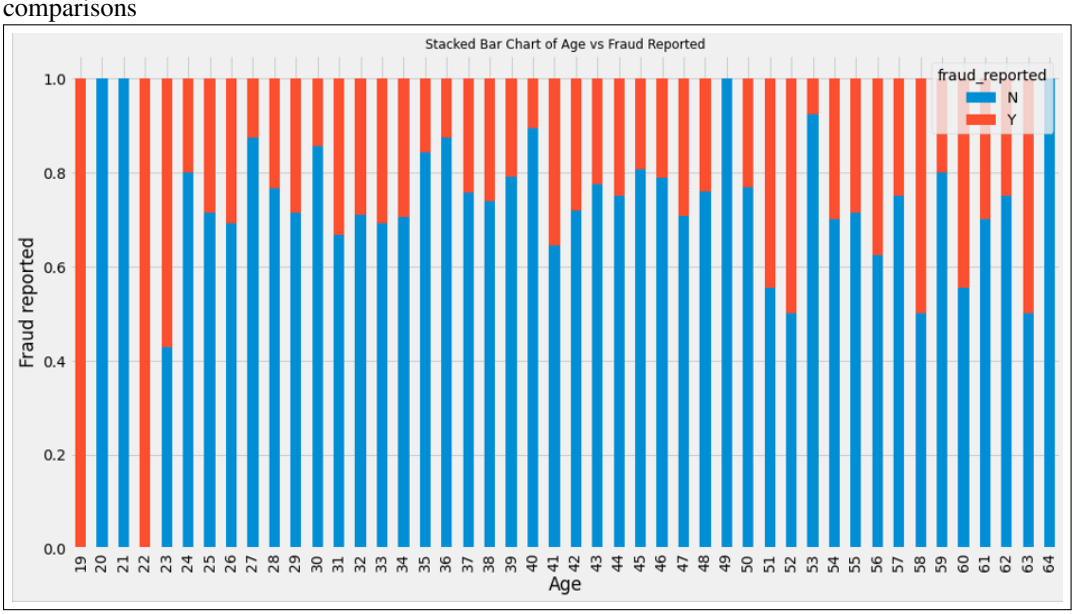

Figure 7: Age-wise fraudulent claims

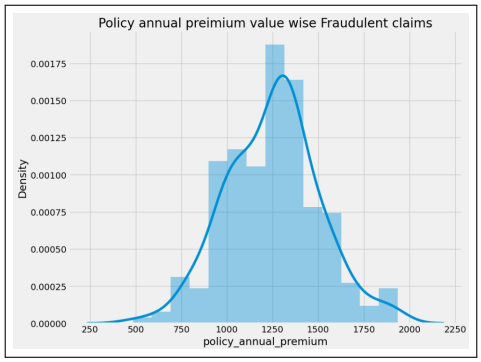

Figure 8: Policy premium relation with fraudulent claims

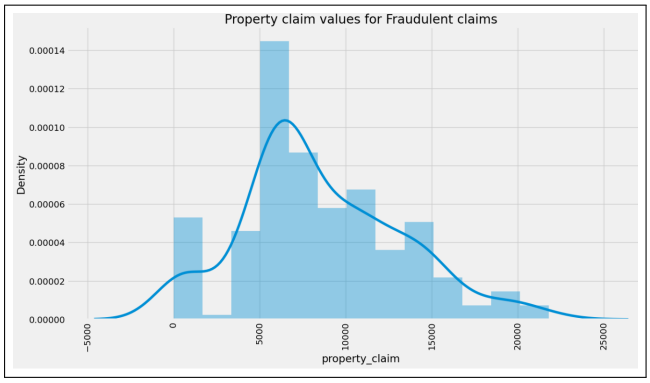

Figure 9: Property claim relation with fraudulent claims

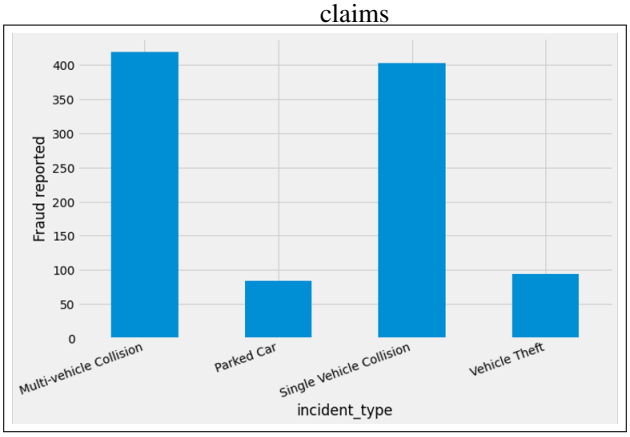

Figure 10: Incident type related to fraudulent claims

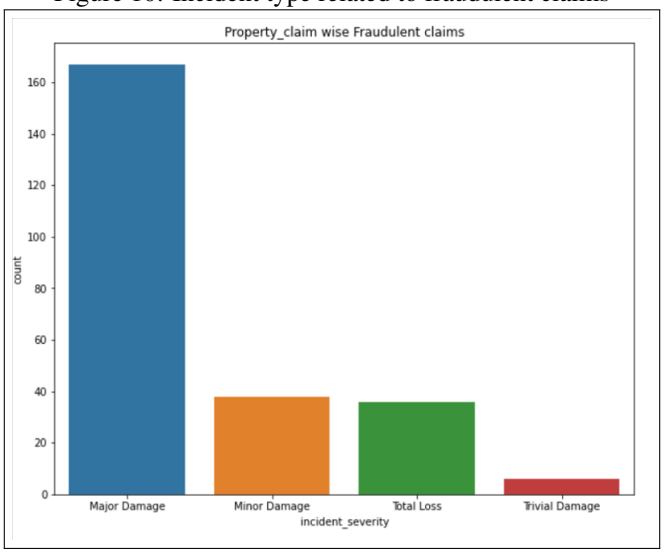

Figure 11: Damage severity related to fraudulent claims

6 Model choice justification

Our fraudulent claim data set comprises 1000 data points where only 24.70% of data are fraudulent, making the data set imbalanced. Furthermore, we also have labels stating which claims are fraudulent. As stated in checkpoint1, our project aims to create a classification model to classify genuine and fraudulent claims. Post going through a few of the existing works pertaining to statistical analysis and prediction on tabular data, as well as machine learning-based fraud detection models. We have narrowed our initial model implementation and prediction to the following:

1. XGBoost.
2. SVM.
3. Logistic regression.

6.1 XG-Boost

XG-boost is widely used for statistical analysis and prediction, often time outperforming deep learning models1. Furthermore, we referred to similar works on fraud detection, like credit-card fraud detection2 and auto-fraud detection paper. Which had also selected XG-Boost as it is one of the most efficient implementations of gradient-boosted decision trees and has been regarded as one of the best machine-learning algorithms, often used in Kaggle competitions3. Specifically designed to optimize memory usage and exploit the hardware computing power, XG-boost decreases the execution time with increased performance. The main idea of boosting is to sequentially build sub-trees from an original tree such that each subsequent tree reduces the errors of the previous one. In such a way, the new sub-trees will update the previous residuals to reduce the error of the cost function. The following properties of XG boot further make it a staple for these classifications:

1. XG-boost is also a highly scalable end-to-end tree-boosting system.
2. The justified weighted quantile sketch is used for efficient proposal calculation.
3. The sparsity-aware algorithm is developed for parallel tree learning.
4. An effective cache-aware block structure is implemented for out-of-core tree learning.

Due to these pros, XG-boost outperforms most other machine learning algorithms in speed and accuracy.

6.2 SVM:

Support vector machines (SVMs) are supervised machine learning models that apply classification methods to two-group classification issues. The support vector machine algorithm aims to locate a hyperplane in N-dimensional space (N is the number of features) that categorizes the data points. Furthermore, SVMs are relatively memory efficient, do not experience overfitting, and perform well when there is a distinct indication of class separation. When the total number of samples is fewer than the total number of dimensions, SVM can be used and does well in terms of memory. Various algorithms are used in machine learning for classification; however, SVM is better than most others since it produces results with higher accuracy. SVM Classifier, compared to other classifiers, has better computational complexity. Even if the number of positive and negative examples are not the same, SVM can be used as it can normalize the data or project into the space of the decision boundary separating the two classes. Due to these pros, we have chosen the SVM as one of the models for fraud detection.

6.3 Logistic regression:

Logistic regression is one of the most common algorithms used for classification models. It is a mathematical model used in statistics to estimate the probability of an event occurring, having been given some previous data. We are using insurance data to classify claims as fraudulent or non-fraudulent operating factors like claim amount, claim type, gender, age, and others. Unlike decision trees or support vector machines, the logistic regression approach enables models to be quickly changed to reflect new data. It is possible to update using stochastic gradient descent. Logistic regression is less prone to overfitting in a low-dimensional dataset with sufficient training

examples. Logistic regression proves very efficient when the dataset has linearly separable features. The predicted parameters (trained weights) give an inference about the importance of each feature. The direction of the association, i.e., positive or negative, is also given. So, we can use logistic regression to determine the features' relationship. Due to these pros, we have chosen the Logistic Regression model as one of the models for fraud detection.

7 Chosen model's test error rates

As illustrated below XG-boost has better scores for training data; hence we will perform further training and testing using XG-Boost:

```
xgb = XGBClassifier()
logreg= LogisticRegressionCV(solver='lbfgs', cv=10)
knn = KNeighborsClassifier(5)
svcl = SVC()
adb = AdaBoostClassifier()
dt = DecisionTreeClassifier(max_depth=5)
rf = RandomForestClassifier()
lda = LinearDiscriminantAnalysis()
gnb = GaussianNB()

# prepare configuration for cross validation test harness
seed = 7
# prepare models
models = []
models.append(('LR', LogisticRegressionCV(solver='lbfgs', max_iter=5000, cv=10)))
models.append(('XGB', XGBClassifier()))
models.append(('SVM', SVC(gamma='auto')))


# evaluate each model in turn
results = []
names = []
scoring = 'accuracy'
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=None)
    cv_results = model_selection.cross_val_score(model, x_train_scaled, y_train, cv=kfold, scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)

# boxplot algorithm comparison
plt.rcParams['figure.figsize'] = [15, 8]
fig = plt.figure()
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()
```

```
LR: 0.831667 (0.057951)
XGB: 0.830000 (0.066999)
SVM: 0.775000 (0.057373)
```

Figure 12: Chosen model's score

8 Predictions of model

We will be checking the performance of predictions of the XG-Boost based on: F1-score, recall, precision, F-beta, and confusion matrix: As illustrated above, XG-boost has a precision of 61% and an F-1 score of 68% for fraudulent claims. Additionally, from the confusion matrix, we can observe that the model categorizes 7% of fraudulent claims as genuine claims and has an f-beta score of 72.7% on recall. These scores show that the model can detect most fraudulent cases.