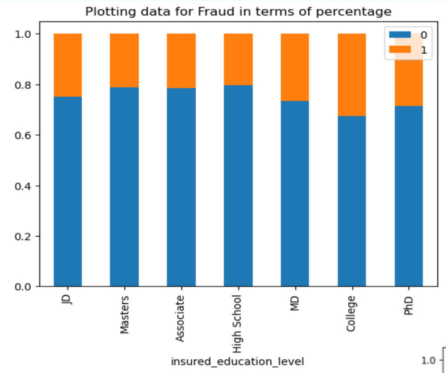
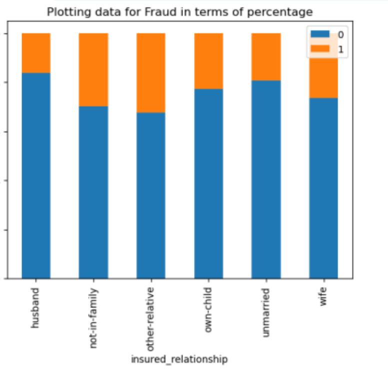
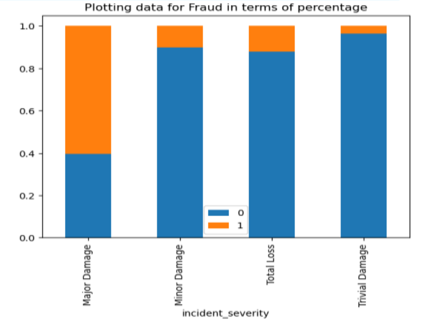
Case Study- Fraudulent Claim Detection

By: Ilan
Sachin
Disha

Summary







Objective:

- Building a model to classify claims as fraudulent or legitimate early in the approval process.

Approach:

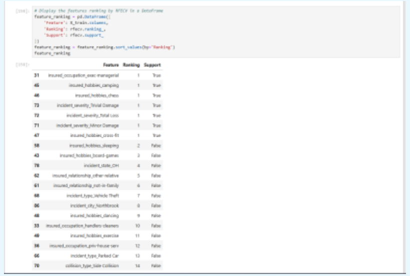
- Data cleaning has been done by dropping blank rows and columns and imputing junk values.
- Univariant and Bivariate Analysis has been conducted between target variable and categorical variables and Correlation Analysis for numerical feature.
- Class imbalance was addressed by RandomOverSampler method.
- Feature Reduction is done by combining the existing feature
- RFE and Cross Validation methods were used in Logistic Regression for feature selection
- Grid Search for hyperparameter tuning in Random Forest

Findings:

Higher proportion if fraud claims observed among:

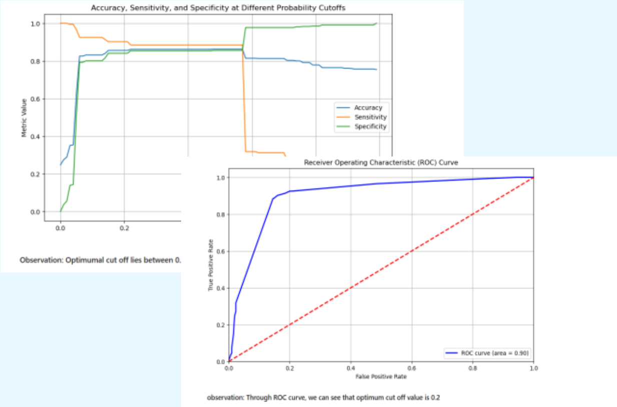
- Males
- Education Level: College students
- Occupation as: exec-managerial, farming/fishing
- Hobbies- Chess, Crossfit
- Insured relationship: Other relative
- Incident Type: Multi-vehicle and single-vehicle collision
- Incident Severity: Major Damage
- Incident State: Ohio
- Incident City: Arlington
- Number of vehicles involved: 4
- Auto Make: Dodge, Ford, Volkswagen
- Auto Model: A5, Silverado, X6 etc.

Logistic Regression



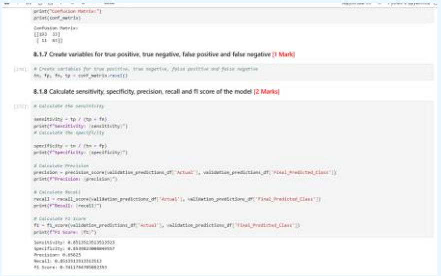
```
# Display the features ranking by RFECV in a DataFrame
feature_ranking = pd.DataFrame({
    'Feature': X_train.columns,
    'Ranking': rfecv.ranking_,
    'Support': rfecv.support_
})
feature_ranking = feature_ranking.sort_values(by='Ranking')
feature_ranking
```

	Feature	Ranking	Support
31	insured_occupation_exec-managerial	1	True
45	insured_hobbies_camping	1	True
46	insured_hobbies_chess	1	True
73	incident_severity_Trivial Damage	1	True
72	incident_severity_Total Loss	1	True
71	incident_severity_Minor Damage	1	True
47	insured_hobbies_cross-fit	1	True
58	insured_hobbies_sleeping	2	False
43	insured_hobbies_board-games	3	False
78	incident_state_OH	4	False
62	insured_relationship_other-relative	5	False
61	insured_relationship_not-in-family	6	False
68	incident_type_Vehicle Theft	7	False
86	incident_city_Northbrook	8	False
48	insured_hobbies_dancing	9	False
33	insured_occupation_handlers-cleaners	10	False
49	insured_hobbies_exercise	11	False
36	insured_occupation_priv-house-serv	12	False
66	incident_type_Parked Car	13	False
70	collision_type_Side Collision	14	False



Observation: Optimumal cut off lies between 0.

observation: Through ROC curve, we can see that optimum cut off value is 0.2



```
print("Confusion Matrix:")
print(conf_matrix)
```

```
Confusion Matrix:
[[193  33]
 [ 11  63]]
```

8.1.7 Create variables for true positive, true negative, false positive and false negative [1 Mark]

```
# Create variables for true positive, true negative, false positive and false negative
tn, fp, fn, tp = conf_matrix.ravel()
```

8.1.8 Calculate sensitivity, specificity, precision, recall and f1 score of the model [2 Marks]

```
# Calculate the sensitivity

sensitivity = tp / (tp + fn)
print(f"Sensitivity: {sensitivity}")
# Calculate the specificity

specificity = tn / (tn + fp)
print(f"Specificity: {specificity}")

# Calculate Precision
precision = precision_score(validation_predictions_df['Actual'], validation_predictions_df['Final_Predicted_Class'])
print(f"Precision: {precision}")

# Calculate Recall
recall = recall_score(validation_predictions_df['Actual'], validation_predictions_df['Final_Predicted_Class'])
print(f"Recall: {recall}")

# Calculate F1 Score
f1 = f1_score(validation_predictions_df['Actual'], validation_predictions_df['Final_Predicted_Class'])
print(f"F1 Score: {f1}")
```

```
Sensitivity: 0.8513513513513513
Specificity: 0.8539823008849557
Precision: 0.65625
Recall: 0.8513513513513513
F1 Score: 0.7411764705882353
```

Feature Selection:

Insured_occupation_exec-managerial, insured_hobbies_camping, insured_hobbies_chess, incident_severity_Trivial Damage, incident_severity_Total Loss, incident_severity_Minor Damage, insured_hobbies_cross-fit were selected as the high contributing features for the fraud by the Recursive Feature Elimination method.

Optimal Cutoff

ROC Curve area=0.90,
The cut-off point lies between 0.2 and and 0.3 which is acceptable

Sensitivity, specificity, precision, recall and F1-score

Sensitivity: 0.8513513513513513

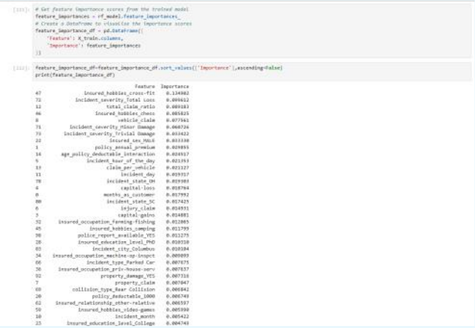
Specificity: 0.8539823008849557

Precision: 0.65625

Recall: 0.8513513513513513

F1 Score: 0.7411764705882353

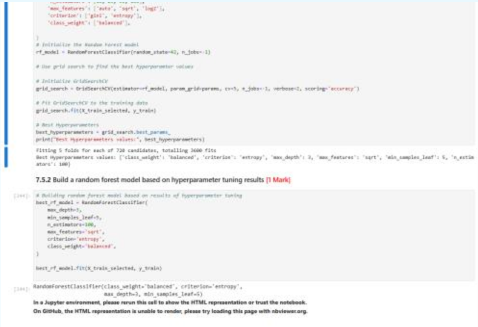
Random Forest



```
[111]: # Get feature importance scores from the trained model
       feature_importances = rf_model.feature_importances_
       # Create a DataFrame to visualize the importance scores
       feature_importance_df = pd.DataFrame({
           'Feature': X_train.columns,
           'Importance': feature_importances
       })

[112]: feature_importance_df=feature_importance_df.sort_values(['Importance'],ascending=False)
       print(feature_importance_df)

                                  Feature  Importance
       47             insured_hobbies_cross-fit    0.134982
       72          incident_severity_Total Loss    0.099612
       12                     total_claim_ratio    0.089183
       46                 insured_hobbies_chess    0.085825
       8                          vehicle_claim    0.077561
       71        incident_severity_Minor Damage    0.060726
       73      incident_severity_Trivial Damage    0.033422
       22                     insured_sex_MALE    0.033330
       1                  policy_annual_premium    0.029855
       14     age_policy_deductable_interaction    0.024517
       5               incident_hour_of_the_day    0.021353
       13                     claim_per_vehicle    0.021127
       11                          incident_day    0.019317
       78                     incident_state_OH    0.019303
       4                           capital-loss    0.018764
       0                     months_as_customer    0.017992
       80                     incident_state_SC    0.017425
       6                           injury_claim    0.014931
       3                          capital-gains    0.014881
       32    insured_occupation_farming-fishing    0.012865
       45               insured_hobbies_camping    0.011799
       98           police_report_available_YES    0.011275
       28           insured_education_level_PhD    0.010310
       83                incident_city_Columbus    0.010184
       34  insured_occupation_machine-op-inspct    0.009093
       66              incident_type_Parked Car    0.007675
       36    insured_occupation_priv-house-serv    0.007637
       92                   property_damage_YES    0.007316
       7                         property_claim    0.007047
       69         collision_type_Rear Collision    0.006842
       20                policy_deductable_1000    0.006749
       62   insured_relationship_other-relative    0.006597
       59           insured_hobbies_video-games    0.005990
       10                        incident_month    0.005422
       23       insured_education_level_College    0.004749
```



```
    'max_features': ['auto', 'sqrt', 'log2'],
    'criterion': ['gini', 'entropy'],
    'class_weight': ['balanced'],

}
# Initialize the Random Forest model
rf_model = RandomForestClassifier(random_state=42, n_jobs=-1)

# Use grid search to find the best hyperparameter values

# Initialize GridSearchCV
grid_search = GridSearchCV(estimator=rf_model, param_grid=params, cv=5, n_jobs=-1, verbose=2, scoring='accuracy')

# Fit GridSearchCV to the training data
grid_search.fit(X_train_selected, y_train)

# Best Hyperparameters
best_hyperparameters = grid_search.best_params_
print("Best Hyperparameters values:", best_hyperparameters)

Fitting 5 folds for each of 720 candidates, totalling 3600 fits
Best Hyperparameters values: {'class_weight': 'balanced', 'criterion': 'entropy', 'max_depth': 3, 'max_features': 'sqrt', 'min_samples_leaf': 5, 'n_estim
ators': 100}
```

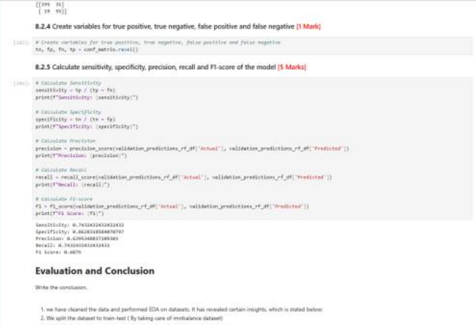
7.5.2 Build a random forest model based on hyperparameter tuning results [1 Mark]

```
[244]: # Building random forest model based on results of hyperparameter tuning
       best_rf_model = RandomForestClassifier(
           max_depth=3,
           min_samples_leaf=5,
           n_estimators=100,
           max_features='sqrt',
           criterion='entropy',
           class_weight='balanced',
       )

       best_rf_model.fit(X_train_selected, y_train)

[244]: RandomForestClassifier(class_weight='balanced', criterion='entropy',
                              max_depth=3, min_samples_leaf=5)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.



```
[[195  31]
 [ 19  55]]
```

8.2.4 Create variables for true positive, true negative, false positive and false negative [1 Mark]

```
[182]: # Create variables for true positive, true negative, false positive and false negative
       tn, fp, fn, tp = conf_matrix.ravel()
```

8.2.5 Calculate sensitivity, specificity, precision, recall and F1-score of the model [5 Marks]

```
[186]: # Calculate Sensitivity
       sensitivity = tp / (tp + fn)
       print(f"Sensitivity: {sensitivity}")

       # Calculate Specificity
       specificity = tn / (tn + fp)
       print(f"Specificity: {specificity}")

       # Calculate Precision
       precision = precision_score(validation_predictions_rf_df['Actual'], validation_predictions_rf_df['Predicted'])
       print(f"Precision: {precision}")

       # Calculate Recall
       recall = recall_score(validation_predictions_rf_df['Actual'], validation_predictions_rf_df['Predicted'])
       print(f"Recall: {recall}")

       # Calculate F1-score
       f1 = f1_score(validation_predictions_rf_df['Actual'], validation_predictions_rf_df['Predicted'])
       print(f"F1 Score: {f1}")

       Sensitivity: 0.7432432432432432
       Specificity: 0.8628318584070797
       Precision: 0.6395348837209303
       Recall: 0.7432432432432432
       F1 Score: 0.6875
```

Evaluation and Conclusion

Write the conclusion.

1. we have cleaned the data and performed EDA on datasets. It has revealed certain insights, which is stated below:
2. We split the dataset to train-test (By taking care of imnbalance dataset)

Feature Selection:
Selected features with importance score greater than 0.02 and they have been listed below, Insured_hobbies_cross-fit, incident_severity_Total Loss, total_claim_ratio, insured_hobbies_chess, vehicle_claim, incident_severity_Minor Damage, incident_severity_Trivial Damage, insured_sex_MALE, policy_annual_premium, age_policy_deductable_interaction,incident_hour_of_the_day, claim_per_vehicle

Hypertuning:

max_depth=3, min_samples_leaf=5, n_estimators=100, max_features='sqrt', criterion='entropy', class_weight='balanced',

Sensitivity, specificity, precision, recall and F1-score:

Sensitivity: 0.8513513513513513
Specificity: 0.8539823008849557
Precision: 0.65625
Recall: 0.8513513513513513
F1 Score: 0.7411764705882353

Recommendations

- Prioritize Camping and Dancing demography over the others in the insured_hobbies, as they show least fraud rate compared to Chess and Cross-fit which shows highest fraud rate in the feature.
- Investigate Multi-vehicle and Single Vehicle Collisions over Vehicle Theft and Parked Cars in incident_type feature.
- The fraud rate increases with the number of vehicles involved in the accident hence, more investigation is needed with increase in the number of vehicles in an accident.
- College seems to have high fraud rate and other than that, the fraud rate increases with the degrees that needs more number of days to complete.
- Exec-managerial and armed-forces has high fraud rate and has to be investigated further.
- Males tend to have a high fraud rate compared to females, even when the class imbalance is observed in favor of females.

Business Implications

- Chess and Cross fit has high rate of fraud. Usually these activities doesn’t require costly equipment and they’r build quality is strong, so the possibility of a high failure rate is not possible hence, it requires more check for these cases.
- Multi-vehicle or Single-vehicle collision has high rate of fraud, maybe due to the lenient inquiries or legal implications for these cases than a stolen car.
- With the increase in the number of days required to complete a degree, the fraud rate increases this may be due to high dropout rate.
- Exec-managerial and armed-forces may have a high fraud rate, possibly due to their influence over the higher authorities.

Questions

- How can we analyse historical claim data to detect patterns that indicate fraudulent claims?
  Yes, it is possible to analyze historical claim data to identify patterns indicative of fraudulent activity. The process begins with data cleaning and imputation to address inconsistencies. This is followed by exploratory data analysis (EDA) to understand data trends and detect imbalances. Feature engineering can then be applied to correct these imbalances and enhance the dataset. Based on the insights gained, an appropriate model can be selected—often, unsupervised learning models are used for such fraud detection tasks.

- Which features are the most predictive of fraudulent behavior?
  The feature **insured_hobbies_cross-fit** is the strongest predictor of fraudulent behavior, with an importance score of **0.134902**.

- Based on past data, can we predict the likelihood of fraud for an incoming claim?
  Yes, it is possible to predict the likelihood of fraud for an incoming claim. After training the model, we can evaluate its performance on test data using metrics such as recall, precision, sensitivity, specificity, and accuracy. These metrics help assess the model’s ability to detect potential fraud.

- What insights can be drawn from the model that can help in improving the fraud detection process?

1. Higher fraud rates were associated with certain demographics, including:

   - Males
   - Customers with specific educational backgrounds (College students)
   - Certain occupations (e.g., exec-managerial, armed-forces)
   - Certain hobbies (e.g., chess, cross-fit)

2. Incident types such as Multi-vehicle and Single Vehicle Collisions showed higher fraud likelihood.

3. The number of vehicles involved and specific auto makes/models also correlated with fraudulent claims.