Capstone 1 Final Project

Caravan Insurance Prediction with Imbalanced Data

Abstract

Policy ownership prediction of CARAVAN insurance policies based on customer demographics and other policy ownership.

Introduction

Welcome to my project from the 2000 CARAVAN insurance challenge.

One of the main reasons why I chose this dataset is the fact that the data is heavily imbalanced and provided an opportunity to see how to address issue in problems more common problems in the real world. This data was officially released in 2000 as part of a challenge to see who could determine the most likely customer to purchase a Caravan Insurance policy.

The data is anonymous and is split up between customer demographics and current policy ownership. Additionally, it has been coded and a dictionary was included for interpretation.

A link to the dataset can be found here:

https://www.kaggle.com/kushshah95/the-insurance-company-tic-benchmark

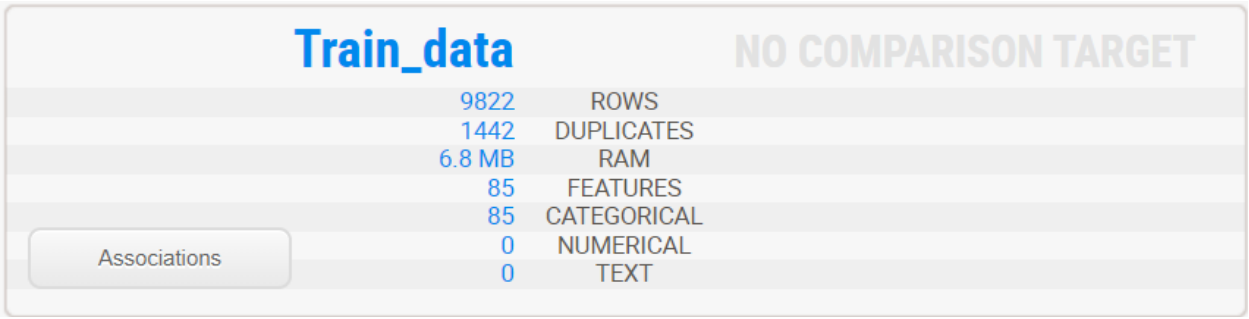
Problem Statement

Given the customer demographics and current policy ownership, who would be most likely to purchase a caravan policy?

Data Description

Data presented in 2 csv files and a txt file

tic_2000_train_data.csv & tic_2000_eval_data.csv: information includes pre encoded customer demographics and policy ownership.



According to the Insurance challenge the data is split into 2 sections: Demographics 0-42 and Policy ownership 43-86. The latter includes our target. Additionally, the features are based on the zipcodes which is unknown. These features are all categorical in nature. The policy ownerships features are coded by percentile levels demonstrated below:

```
L3:

0 0%
1 1 - 10%
2 11 - 23%
3 24 - 36%
4 37 - 49%
5 50 - 62%
6 63 - 75%
7 76 - 88%
8 89 - 99%
9 100%


L4:

0 f 0
1 f 1 – 49
2 f 50 – 99
3 f 100 – 199
4 f 200 – 499
5 f 500 – 999
6 f 1000 – 4999
7 f 5000 – 9999
8 f 10.000 - 19.999
9 f 20.000 - ?
```

The features Subtype, Maintype, Age, and Roman Catholic percentage also had a subcategories that can be viewed in the dictionary provided in file.

Data Wrangling Details

Thankfully the dataset was extremely clean and did not require any data wrangling for the most part. For better readability and ease of maneuvering through the dataset, I concatenated both csv files and then renamed all the columns into abbreviations according to the dictionary provided.

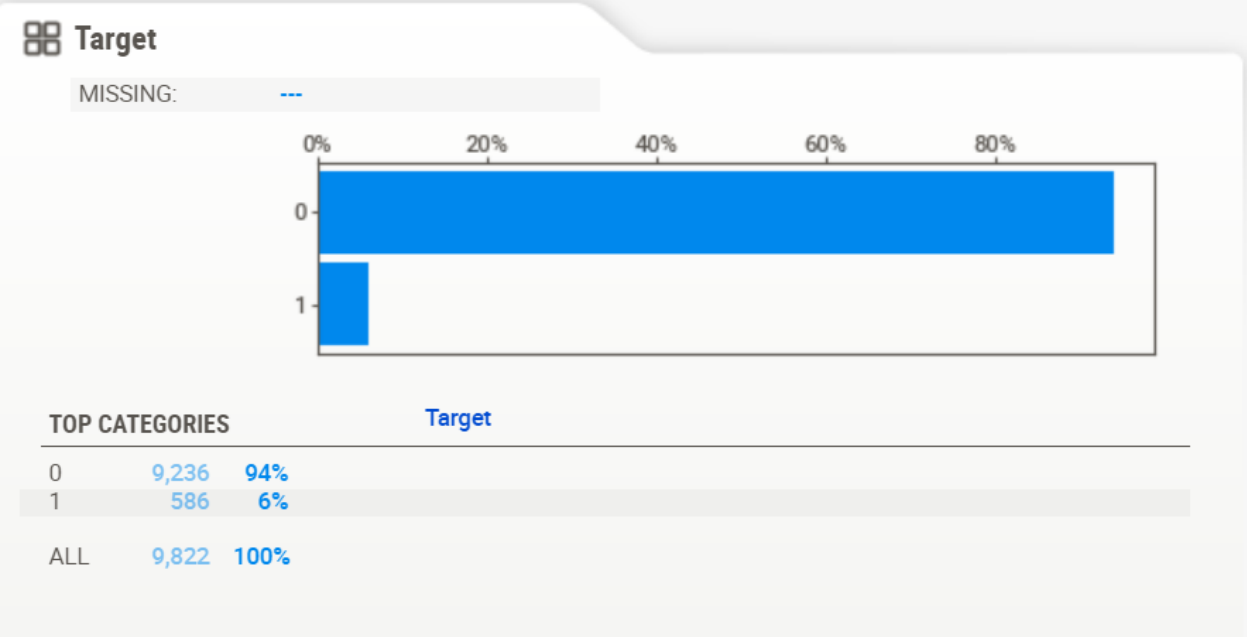
```
pd.set_option('display.max_columns', None)
df_td = pd.read_csv('tic_2000_train_data.csv')
eval_df = pd.read_csv('tic_2000_eval_data.csv')#CARAVAN is renamed 'Target' in this set

#renaming the training data to match the test data.
df_td.rename(columns={'MOSTYPE': 'subtype_L0', 'MAANTHUI':'Num_houses', 'MGEMOMV' : 'Avg_hh_size',
                  'MGEMLEEF':'age_L1', 'MOSHOOFD': 'maintype_L2', 'MGODRK': 'romcath_L3',
                  'MGODPR': 'Protestant','MGODOV' : 'O_religion', 'MGODGE': 'N_religion','MRELGE' :'Married',
                  'MRELSA' : 'Living_together','MRELOV' : 'O_relation','MFALLEEN' : 'Singles','MFGEKIND' : 'hh_wo_child',
                  'MFWEKIND' : 'hh_w_child','MOPLHOOG' : 'H_lvl_edu','MOPLMIDD' : 'M_lvl_edu',
                  'MOPLLAAG' : 'L_lvl_edu','MBERHOOG' : 'H_status','MBERZELF' : 'Entrepreneur','MBERBOER' : 'Farmer',
                  'MBERMIDD' : 'Mid_management','MBERARBG' : 'Skld_labor','MBERARBO' : 'Unskld_labor',
                  'MSKA' : 'Soc_cls_A','MSKB1' : 'Soc_cls_B1','MSKB2' : 'Soc_cls_B2','MSKC' : 'Soc_cls_C',
                  'MSKD' : 'Soc_cls_D','MHHUUR' : 'R_house','MHKOOP' : 'O_house','MAUT1' : '1_car','MAUT2' : '2_cars',
                  'MAUT0' : 'N_car','MZFONDS' : 'Nat_Hlth_Serv','MZPART' : 'Prv_Hlth_Insur','MINKM30' : 'Inc_u_30k',
                  'MINK3045' : 'Inc_btw_30_45k','MINK4575' : 'Inc_btw_45_75k','MINK7512' : 'Inc_75_122k','MINK123M' : 'Inc_ovr_123k',
                  'MINKGEM' : 'Avg_inc','MKOOPKLA' : 'PP_cls','PWAPART' : 'Contri_prv_3p_insur','PWABEDR' : 'Firm_Contri_3p_ insur',
                  'PWALAND' : 'Ag_Contri_3p_insur','PPERSAUT' : 'Contri_car_pol','PBESAUT' : 'Contri_deliv_van_pol',
                  'PMOTSCO' : 'Contri_motorcycle/scooter_pol','PVRAAUT' : 'Contri_lorry_pol','PAANHANG' : 'Contri_trailer_pols',
                  'PTRACTOR' : 'Contri_tractor_pol','PWERKT' : 'Contri_ag_machine_pol','PBROM' : 'Contri_moped_pol',
                  'PLEVEN' : 'Contri_life_insur','PPERSONG' : 'Contri_prv_accid_insur_pol',
                  'PGEZONG' : 'Contri_fam_accid_insur_pol','PWAOREG' : 'Contri_disabl_insur_pol','PBRAND' : 'Contri_fire_pol',
                  'PZEILPL' : 'Contri_surfb_pol','PPLEZIER' : 'Contri_boat_pol','PFIETS' : 'Contri_bike_pol',
                  'PINBOED' : 'Contri_prop_insur_pol','PBYSTAND' : 'Contri_ss_insur_polo','AWAPART' : 'Num_prv_3p_insur',
                  'AWABEDR' : 'Num_firm_3p_insur','AWALAND' : 'Num_ag_3p_insur','APERSAUT' : 'Num_car_pol',
                  'ABESAUT' : 'Num_deliv_van_pol','AMOTSCO' : 'Num_motorcycle/scooter_pol', 'AVRAAUT' : 'Num_lorry_pol','AAANHANG': 'Num_trailer_pol',
                  'ATRACTOR' : 'Num_tractor_pol','AWERKT' : 'Num_ag_machines_pol','ABROM' : 'Num_moped_pol',
                  'ALEVEN' : 'Num_life_insur_pol', 'APERSONG' : 'Num_prv_accid_insur_pol','AGEZONG' : 'Num_fam_ccid_insur_pol',
                  'AWAOREG' : 'Num_disabl_insur_pol','ABRAND' :'Num_fire_pol','AZEILPL' :'Num_surfb_pol','APLEZIER' :'Num_boat_pol',
                  'AFIETS' :'Num_bike_pol','AINBOED' :'Num_prop_insur_pol','ABYSTAND' :'num_ss_insur_pol', 'CARAVAN' : 'Target'},
          inplace=True)
eval_df.rename(columns={'MOSTYPE': 'subtype_L0', 'MAANTHUI':'Num_houses', 'MGEMOMV' : 'Avg_hh_size',
                  'MGEMLEEF':'age_L1', 'MOSHOOFD': 'maintype_L2', 'MGODRK': 'romcath_L3',
                  'MGODPR': 'Protestant','MGODOV' : 'O_religion', 'MGODGE': 'N_religion','MRELGE' :'Married',
                  'MRELSA' : 'Living_together','MRELOV' : 'O_relation','MFALLEEN' : 'Singles','MFGEKIND' : 'hh_wo_child',
                  'MFWEKIND' : 'hh_w_child','MOPLHOOG' : 'H_lvl_edu','MOPLMIDD' : 'M_lvl_edu',
                  'MOPLLAAG' : 'L_lvl_edu','MBERHOOG' : 'H_status','MBERZELF' : 'Entrepreneur','MBERBOER' : 'Farmer',
                  'MBERMIDD' : 'Mid_management','MBERARBG' : 'Skld_labor','MBERARBO' : 'Unskld_labor',
                  'MSKA' : 'Soc_cls_A','MSKB1' : 'Soc_cls_B1','MSKB2' : 'Soc_cls_B2','MSKC' : 'Soc_cls_C',
                  'MSKD' : 'Soc_cls_D','MHHUUR' : 'R_house','MHKOOP' : 'O_house','MAUT1' : '1_car','MAUT2' : '2_cars',
                  'MAUT0' : 'N_car','MZFONDS' : 'Nat_Hlth_Serv','MZPART' : 'Prv_Hlth_Insur','MINKM30' : 'Inc_u_30k',
                  'MINK3045' : 'Inc_btw_30_45k','MINK4575' : 'Inc_btw_45_75k','MINK7512' : 'Inc_75_122k','MINK123M' : 'Inc_ovr_123k',
                  'MINKGEM' : 'Avg_inc','MKOOPKLA' : 'PP_cls','PWAPART' : 'Contri_prv_3p_insur','PWABEDR' : 'Firm_Contri_3p_ insur',
                  'PWALAND' : 'Ag_Contri_3p_insur','PPERSAUT' : 'Contri_car_pol','PBESAUT' : 'Contri_deliv_van_pol',
                  'PMOTSCO' : 'Contri_motorcycle/scooter_pol','PVRAAUT' : 'Contri_lorry_pol','PAANHANG' : 'Contri_trailer_pols',
                  'PTRACTOR' : 'Contri_tractor_pol','PWERKT' : 'Contri_ag_machine_pol','PBROM' : 'Contri_moped_pol',
                  'PLEVEN' : 'Contri_life_insur','PPERSONG' : 'Contri_prv_accid_insur_pol',
                  'PGEZONG' : 'Contri_fam_accid_insur_pol','PWAOREG' : 'Contri_disabl_insur_pol','PBRAND' : 'Contri_fire_pol',
                  'PZEILPL' : 'Contri_surfb_pol','PPLEZIER' : 'Contri_boat_pol','PFIETS' : 'Contri_bike_pol',
                  'PINBOED' : 'Contri_prop_insur_pol','PBYSTAND' : 'Contri_ss_insur_polo','AWAPART' : 'Num_prv_3p_insur',
                  'AWABEDR' : 'Num_firm_3p_insur','AWALAND' : 'Num_ag_3p_insur','APERSAUT' : 'Num_car_pol',
                  'ABESAUT' : 'Num_deliv_van_pol','AMOTSCO' : 'Num_motorcycle/scooter_pol', 'AVRAAUT' : 'Num_lorry_pol','AAANHANG': 'Num_trailer_pol',
                  'ATRACTOR' : 'Num_tractor_pol','AWERKT' : 'Num_ag_machines_pol','ABROM' : 'Num_moped_pol',
                  'ALEVEN' : 'Num_life_insur_pol', 'APERSONG' : 'Num_prv_accid_insur_pol','AGEZONG' : 'Num_fam_ccid_insur_pol',
                  'AWAOREG' : 'Num_disabl_insur_pol','ABRAND' :'Num_fire_pol','AZEILPL' :'Num_surfb_pol','APLEZIER' :'Num_boat_pol',
                  'AFIETS' :'Num_bike_pol','AINBOED' :'Num_prop_insur_pol','ABYSTAND' :'num_ss_insur_pol'},
          inplace=True)

df_list = [df_td, eval_df]
df = pd.concat(df_list)
```

Because some of the features had subcategories within them, I decided to create a dummy table of the entire dataset and leave out the target variable. I did this in order to better capture which features had greater impacts on the target variable. I did this however after further data exploration.

I also used under sampling techniques in order to compensate for the data imbalance.

Exploratory Data Analysis and Feature Selection

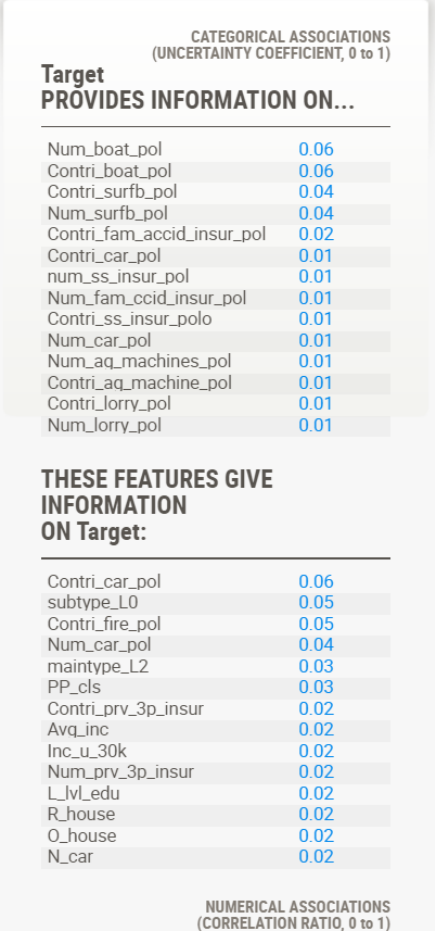
In exploring the data I confirmed that the dataset with extremely imbalanced.



Additionally, I was able to quickly narrow down where to start looking as far as feature extraction was concerned. There were in total 28 features which related to my target variable based on the uncertainty coefficient. I found there was also some heavy inner correlation among features, which I could intuitively rule out.

For example: Contributions to Car Policies and Number of Car Policies.

Because of that, I dropped many of features and focused on the ones with a direct relationship to the target feature which can be seen on the following page.



CATEGORICAL ASSOCIATIONS
(UNCERTAINTY COEFFICIENT, 0 to 1)

Target
PROVIDES INFORMATION ON...

Num_boat_pol	0.06
Contri_boat_pol	0.06
Contri_surfb_pol	0.04
Num_surfb_pol	0.04
Contri_fam_accid_insur_pol	0.02
Contri_car_pol	0.01
num_ss_insur_pol	0.01
Num_fam_ccid_insur_pol	0.01
Contri_ss_insur_polo	0.01
Num_car_pol	0.01
Num_aq_machines_pol	0.01
Contri_aq_machine_pol	0.01
Contri_lorry_pol	0.01
Num_lorry_pol	0.01

THESE FEATURES GIVE INFORMATION ON Target:

Contri_car_pol	0.06
subtype_L0	0.05
Contri_fire_pol	0.05
Num_car_pol	0.04
maintype_L2	0.03
PP_cls	0.03
Contri_prv_3p_insur	0.02
Avg_inc	0.02
Inc_u_30k	0.02
Num_prv_3p_insur	0.02
L_lvl_edu	0.02
R_house	0.02
O_house	0.02
N_car	0.02

NUMERICAL ASSOCIATIONS
(CORRELATION RATIO, 0 to 1)

I then encoded the entire dataset, leaving out my target variable, and ran a LassoCV across the features. I ended up with 59 features in total which can be seen below.

```
#Using LassoCV, we now have our new features
new_df = post_enc_df[['Contri_boat_pol_0', 'Avg_inc_0', 'maintype_L2_10', 'Contri_fire_pol_2',
       'Contri_ss_insur_polo_0', 'maintype_L2_5', 'maintype_L2_4', 'R_house_2',
       'L_lvl_edu_7', 'H_lvl_edu_3', 'H_lvl_edu_0', 'L_lvl_edu_6',
       'maintype_L2_3', '1_car_4', '1_car_2', 'L_lvl_edu_9', 'Avg_inc_2',
       'Contri_fire_pol_1', 'Avg_inc_3', 'Contri_fire_pol_6', 'R_house_5',
       'PP_cls_3', 'PP_cls_5', 'Contri_prv_3p_insur_1', 'Num_fire_pol_2',
       'maintype_L2_1', 'maintype_L2_2', 'Contri_disabl_insur_pol_6',
       'H_lvl_edu_5', 'Num_fire_pol_1', '1_car_5', '1_car_6', 'PP_cls_4',
       'H_lvl_edu_4', 'H_lvl_edu_6', 'Contri_fire_pol_5', 'Avg_inc_5',
       'maintype_L2_8', 'R_house_1', '1_car_9', 'L_lvl_edu_8', 'R_house_6',
       'L_lvl_edu_2', 'maintype_L2_9', 'Contri_fire_pol_3', 'Avg_inc_4',
       'Avg_inc_7', 'L_lvl_edu_1', 'PP_cls_8', 'R_house_0', 'L_lvl_edu_0',
       '1_car_7', 'Contri_prv_3p_insur_2', 'H_lvl_edu_7',
       'Contri_ss_insur_polo_4', 'Contri_fire_pol_4', 'PP_cls_7',
       'Contri_car_pol_6', 'Contri_fam_accid_insur_pol_3','Target']]
```

Prediction Machine Learning

For this problem I will be using Native Bayes, Ensemble methods, decision trees, and a neural net to see what works best with the data. I still have to address the imbalance, but I want to see how these models handle the data with the selected features.

I have decided to use these particular models as this is a classification and prediction problem. NativeBayes offers a particular advantage in that it assumes a level of independence between variables and allows us to better implement it into the test data.

One of the more interesting models I will be using is the ComplementNB(), the primary purpose of it is applying native bayes specifically to compensate for imbalanced data.

I am incorporating many metrics in the model in order to compare how they interpret and score the results.

Metric for judging models:

1) Accuracy_score - accuracy score as the name implies, measures how accurate the model is at matching the predictions with the true values. However its draw back is that the model can simply memorize all the pathways. As with GBC, it has an extremely high accuracy score, however it absolutely failed at successfully sorting occurances of our Target variable as can be seen in the confusion matrix. Because of this, I will not be relying so heavily on the accuracy score for measuring the model's fit.

2) Confusion_matrix - the confusion matrix which measures how successfully the model has classified the instances of the predictions will be one of the primary metrics in how I measure how well these models perform. The higher instances of True Positives and True negatives, the better the model. The ROC curve will be used to confirm this trend.

3) ROC - This is the more suitable version of accuracy score, specifically when addressing imbalanced datasets. A drawback is that ROC AUC doesn't tell you anything about the costs of different kinds of errors. So this I will be using this as a secondary scoring metric.

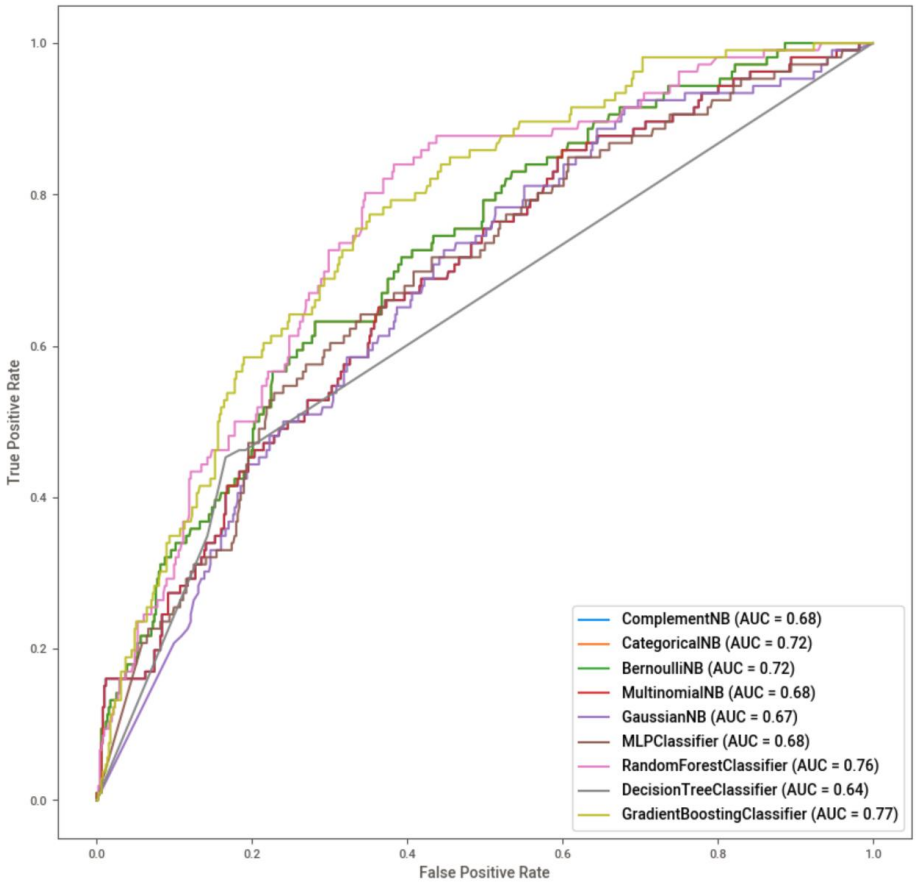
4) F1 score and classification report - you will notice hace the a traing & testing F1 score, and a classification report listed. I included these because I wanted to be sure that the models did not have an extremely violent drop off from the training data to the testing data. By including both of these F1 metrics, I got a better sense of how each model truly handled the data. Additionally, the classification report provided further information on the test set to confirm how the results played out.

5) Precision/Recall - This is the other primary metric I used to ensure how the models were operating. Having a higher precision was my main focus because my main question is who is most likely to purchase a policy. In this instance, being able to correctly sort the instances is more important.

6) Balance_accuracy_score - I included this to compare how accurate each model was at balancing the data itself. I included this mainly to compare the ComplementNB model to the rest of the classifiers used.

Under sampling data:

I then used an under sampling method to better fit my data. I tried to get as close to a 5-1 ratio as possible as was suggested by my mentor. I ended up using 2500 instances of the majority class after playing around with different numbers as this created some optimal results.


True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ComplementNB (AUC = 0.68)
CategoricalNB (AUC = 0.72)
BernoulliNB (AUC = 0.72)
MultinomialNB (AUC = 0.68)
GaussianNB (AUC = 0.67)
MLPClassifier (AUC = 0.68)
RandomForestClassifier (AUC = 0.76)
DecisionTreeClassifier (AUC = 0.64)
GradientBoostingClassifier (AUC = 0.77)

```
GradientBoostingClassifier()
The Training F1 Score is 0.3656957928802589
The Ttest F1 Score is 0.26027397260273977
accuracy score
 0.8252427184466019
balanced_accuracy_score
 0.5702094103215631
model confusion matrix
 [[0.79449838 0.02265372]
 [0.15210356 0.03074434]]
classification_report
              precision    recall  f1-score   support

           0       0.84      0.97      0.90       505
           1       0.58      0.17      0.26       113

    accuracy                           0.83       618
   macro avg       0.71      0.57      0.58       618
weighted avg       0.79      0.83      0.78       618
```

```
ComplementNB()
The Training F1 Score is 0.4276169265033407
The Ttest F1 Score is 0.4110429447852761
accuracy score
 0.6893203883495146
balanced_accuracy_score
 0.6519057215456059
model confusion matrix
 [[0.58090615 0.23624595]
 [0.07443366 0.10841424]]
classification_report
              precision    recall  f1-score   support

           0       0.89      0.71      0.79       505
           1       0.31      0.59      0.41       113

    accuracy                           0.69       618
   macro avg       0.60      0.65      0.60       618
weighted avg       0.78      0.69      0.72       618
```

```
RandomForestClassifier(max_depth=3, max_features=None, n_estimators=615,
                       n_jobs=1, random_state=0, verbose=False)
The Training F1 Score is 0.11328124999999999
The Ttest F1 Score is 0.06666666666666665
accuracy score
 0.8187702265372169
balanced_accuracy_score
 0.5147288180145448
model confusion matrix
 [[0.81229773 0.00485437]
 [0.1763754  0.00647249]]
classification_report
              precision    recall  f1-score   support

           0       0.82      0.99      0.90       505
           1       0.57      0.04      0.07       113

    accuracy                           0.82       618
   macro avg       0.70      0.51      0.48       618
weighted avg       0.78      0.82      0.75       618
```

Conclusion

ComplementNB, as the documentation suggested, handles heavily imbalanced data much better at balancing than the other algorithms. However, its scores were not too notable compared to the other native bayes models. The Categorical/BernoulliNB both did equally well in that there was no distinction between the two and handled the best out of the models used thus far. The other models used were primarily to play around and see what really worked. I was curious to see how a neural net would handle the dataset, I found this required a lot of hyper parameter tuning in which I attempted using Hyperopt-sklearn in the tuningparams doc. I ended up resorting to manual tuning as this seemed to better capture the information I was seeking. I incorporated the different ensemble models to see how they were able to sort the data but was met with disappoint results. I even tuned the hyper parameters for the RandomForestClassifier but still had an overfitting problem. GradientBoosting clearly was met with an overfitting problem which I learned that it would have been better to use if I had a much larger dataset.

Looking back at the dataset, the strongest predictors for who would purchase a car policy ended up being the purchasing power of the consumer, and whether or not they bought a fire or car policy.

With a much larger dataset, I believe many of the issue I faced in this problem may have been overcome. The biggest issue was having so few instances that it made it harder to implement more complex models without overfitting the problem entirely.