Warren Ball
Christopher Pineda
Jason Wang
Andreas Constantinou

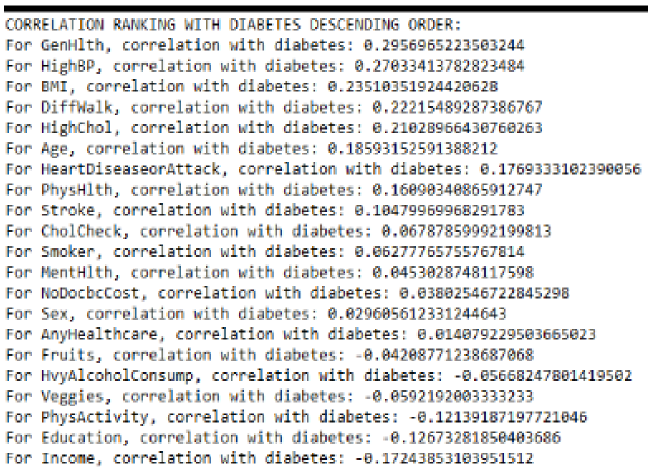
Data Mining Report

<u>Diagnosing Diabetes</u>

For our final project we decided to modify a dataset on diabetes. Diabetes has been the major cause of issues in many family households. Our goal for this dataset is to figure out what influences diabetes. We believe that by finding what manipulates diabetes, we may provide advertisers data and capture a target audience for people that may be at risk.

Our dataset consists of multiple types of data such as binary, numeric, and discretized. We used a correlation matrix to figure out which columns correlated best with individuals who were diagnosed with diabetes. We then took the top five columns which were GenHlth, HighBP, BMI, DiffWalk, and HighChol and kept adding more columns until the overall score eventually started decreasing.

There was only one issue with our dataset, and that issue was that our dataset was imbalanced in a sense where most people were recorded as negative for diabetes rather than positive. Due to such imbalance, we applied the following strategies to our training data: oversampling, undersampling, SMOTE, and balancing weights toward the minority class. The models we decided on using were naive bayes, logistic regression, decision trees, and random forest. We would announce the best model depending on how precise and large the score, recall, true positives, and true negatives were, along with the best balancing method applied to the model. Lastly, we would use the most accurate model to draw logical conclusions and predict diabetes depending on the features.



```
CORRELATION RANKING WITH DIABETES DESCENDING ORDER:
For GenHlth, correlation with diabetes: 0.2956965223503244
For HighBP, correlation with diabetes: 0.27033413782823484
For BMI, correlation with diabetes: 0.23510351924420628
For DiffWalk, correlation with diabetes: 0.22215489287386767
For HighChol, correlation with diabetes: 0.21028966430760263
For Age, correlation with diabetes: 0.18593152591388212
For HeartDiseaseorAttack, correlation with diabetes: 0.17693331023900056
For PhysHlth, correlation with diabetes: 0.16090340865912747
For Stroke, correlation with diabetes: 0.10479969968291783
For CholCheck, correlation with diabetes: 0.06787859992199813
For Smoker, correlation with diabetes: 0.06277765755767814
For MentHlth, correlation with diabetes: 0.0453028748117598
For NoDocbcCost, correlation with diabetes: 0.03802546722845298
For Sex, correlation with diabetes: 0.029605612331244643
For AnyHealthcare, correlation with diabetes: 0.014079229503665023
For Fruits, correlation with diabetes: -0.04208771238687068
For HvyAlcoholConsump, correlation with diabetes: -0.05668247801419502
For Veggies, correlation with diabetes: -0.0592192003333233
For PhysActivity, correlation with diabetes: -0.12139187197721046
For Education, correlation with diabetes: -0.12673281850403686
For Income, correlation with diabetes: -0.17243853103951512
```

Before applying any balancing methods, the results were heavily skewed towards the majority class, where the recall for positives was way lower than of people tested negative. Although this made the model score 10-15% better, it would never predict anyone to have diabetes for any model.

Warren's NB Report

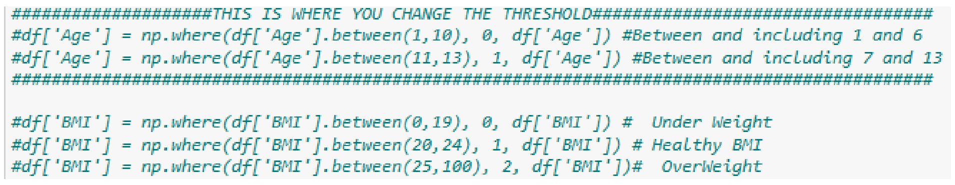
I remembered near the beginning that you exaggerated that we could look into the real world cost of misdiagnosing healthy people to catch more sick people. However, in my experience, it was impossible to move any amount of False Negatives to True Positives without bringing over 10 or 20 times as many False Positives. In the end, I believe it was because each feature had such a low correlation score for being a predictor of diabetes. The highest correlating feature, General Health, couldn't even breach 30%. There was also the fact that the accuracy and recall would only be enhanced by the top 6 most correlated features, and any more would bring those scores down. I would also argue that these features aren't too effective at diagnosing diabetes in the first place. Every one of the features were very non-invasive, unlike a feature that would need some kind of facility to test out, such as insulin level or blood sugar. All of these features could be found by a surveyor (note high blood pressure and high cholesterol were yes or no questions not measured levels. ie "Do you have high blood pressure?"). Due to these constraints, I dropped the hope of trying to significantly minimize False Positives beyond what I had already accomplished.

Naive Bayes was interesting in that it had a surprisingly unchanging score. Whether I were to balance out the weights, use oversampling, undersampling, or even SMOTE oversampling, the amount of True Positives, recall, and accuracy values would remain close to identical.

All accuracy hovered at 72% for features selected and 74% for all features. The amount of True Positives were around 3435 (3330 for all features). And recall was at a constant .77 (.73 for all features). The only exception was SMOTE oversampling, which dropped in accuracy to .71 (.72 for all features), had an increase in True Positives at 3492 (3214 for all features), and increased in recall at .78 (.71 for all features). But this is hardly a change at all. In my estimation, this is due to Naive Bayes being a simplistic model to begin with. Given that there are only a few numbers that get plugged in, (ie the prior, the likelihood, and the evidence) I think that whatever way I manipulated the data, it would ultimately be read the same way.

<u>Christopher's Logistic Regression Report</u>

With Logistic Regression performing best with categorical data, binarizing numerical values such as BMI and Age did not aid in improving the performance of the model. Age and BMI both have a decent correlation with diabetes, so an approach that I took was to see how the model would react if I binarized certain age categories as well as BMI. The results of binarizing made it perform slightly worse to the point where in reality it changed nothing.



```
####################THIS IS WHERE YOU CHANGE THE THRESHOLD##################################
#df['Age'] = np.where(df['Age'].between(1,10), 0, df['Age']) #Between and including 1 and 6
#df['Age'] = np.where(df['Age'].between(11,13), 1, df['Age']) #Between and including 7 and 13
############################################################################################

#df['BMI'] = np.where(df['BMI'].between(0,19), 0, df['BMI']) #  Under Weight
#df['BMI'] = np.where(df['BMI'].between(20,24), 1, df['BMI']) # Healthy BMI
#df['BMI'] = np.where(df['BMI'].between(25,100), 2, df['BMI'])#  OverWeight
```

Before trying every procedure to consider the imbalanced dataset, each procedure resulted in having the same result where more data made the model perform better (or slightly better for less correlated values). Certainly removing the best correlated values made it perform worse, so in general feature selection was not very friendly for this model. Although it did not improve the model, feature selection was tested out on all four methods to make sure if it performs differently against one another.

Setting balanced_weight = ‘balanced” performed the worst compared to the other methods, especially if feature selection is applied. With feature selection the amount of true positives along with the recall dropped, with 300 true positives less than without feature selection. As for oversampling, undersampling, and SMOTE, the results were very head-on to the point where classifying the best method was a challenge. SMOTE was valued as the best due to its better reaction to feature selection, because removing irrelevant features did not hurt or improve the model, which should make more sense considering the features were badly correlated. The recall, score, and true positives/negatives ratio was overall better than the other two, with no binarizing since it made it worse.

Jason’s Random Forest Report

Random Forest is one of the most versatile algorithms out there, being able to handle binary features, categorical features, and numerical features. Since the dataset is mostly binary, one of the approaches we did was binarizing Age and BMI before using a random forest classifier and not regressor because it is mostly binary data. The result was mixed because it did slightly perform better with oversampling, class_weight and SMOTE, but it performed worse with undersampling, so I didn’t binarize Age and BMI when using undersampling. As mentioned before, our dataset is imbalanced with more people having no diabetes than having diabetes, so using oversampling, SMOTE, balanced weight, and undersampling to fix the imbalancing certainly did improve random forest’s performance, with undersampling performing the best.

Before getting into the result after feature selection, I want to talk about the performance of random forest before feature selection, as this is where undersampling stands out the most from the rest. Starting with setting class_weight = “balanced”, it got a high score of 80.62%, but it has a recall of 0.27 and 1589 true positives. Oversampling performed slightly better with a lower score of 78.5%, recall of 0.4, and 2420 true positives, showing that we focus more on recall and true positives than the score. After applying undersampling to the model, we got the lowest score of 70.34%, but it got the highest recall and true positives out of the 4 methods with 0.76 on the recall and 4636 true positives. Even though logistic regression with smote got the best performance overall, random forest got the worst performance with smote when k nearest neighbors is set to 15. For instance, smote got the highest score of 82.42%, but it got 0.24 on the recall and 1423 on the true positives, which is lower than setting class_weight to balanced.

However, after feature selection, we will see better performance in class_weight = balanced, oversampling, and smote. Just to clarify, the columns I will be using after feature selection would be the top 5 columns based on correlation since it seems like I get the best performance using those columns. For balanced weight, the score dropped ~8%, but the recall went from 0.27 to 0.73, and the true positive increased to 4352, and we will see this pattern for oversampling and smote. Oversampling's score dropped ~6% and the recall and true positives are roughly the same as class_weight = balanced, and that same goes for smote, but the score dropped ~10% instead. Based on this pattern, I thought undersampling would have similar results, but after feature selection, undersampling performed slightly worse. The score increased by 0.5% and the recall stayed the same, but the true positives decreased by ~130. Although for undersampling there is a slight decrease when it features selects, it still has a better recall and true positive than other methods even after feature selection, which shows that undersampling performs best with and without feature selection unlike the oversampling, smote, and class_weight = balanced. Most people view random forests as a better version of decision trees, so we also want to explore that algorithm and see if it performs similar results.

Andreas Constantinou Decision Trees Report

Decision trees are always a good choice when it comes to picking a good data mining technique due to the fact that they can typically work with almost any type of data. A decision tree is a type of technique used for building classification models. It constantly asks the dataset questions which depending on the answer, separates the data into different groups that form the "branches" of the decision tree. Due to this it typically tends to form a tree like structure hence the name "Decision tree".

I ended up making four decision tree models with each one using a different balancing technique. My first model was using balancing weights, my second model used oversampling, my third model used undersampling, and my last model used SMOTE. While using these four models I also made two datasets in each model with one using all the features, and the other using the top 5 best features in the dataset based off of the correlation matrix.

For balancing weight my overall score was 78.02 percent from using all the features, and 70.79 percent after using feature selection. When it came to recall I had a very bad score of 31 percent when I used all the features and a good score of 72 percent after feature selection. Lastly my true positives came out to be better after feature selection with a score of 4272 compared to before feature selection which was 1880. Out of all my models this one had the highest overall score which I found to be very interesting, but its true positives and recalls were not really that accurate at all.

For oversampling, my overall score was 77.75 percent before feature selection and decreased to 71.56 percent after feature selection. The recall score was a horrifying 33 percent before feature selection and an ok 70 percent after feature selection. Lastly, my true positives for before feature selection were really low at 1975 and very good at 4256 after feature selection.

My oversampling scores improved on the balanced weight score barely. Although it may be better than balanced weights, it is still not my most accurate model.

For undersampling my overall score was the worst by far at 65.62 percent before feature selection and a very average score at 71.64 percent after feature selection. My recall score was low at 64 percent before feature selection and average at 71 percent after feature selection. Lastly, my true positives were both very average at 3873 before feature selection and 4289 after feature selection. Unlike my previous two models, undersampling is very one sided with every score just being better after feature selection. That being said the score is still pretty low compared to my other models so this is still not my most accurate model.

For SMOTE my overall score before feature selection was 77.55 percent before feature selection and 77.39 after feature selection. My recall score is very low at 64 percent before feature selection and an average 71 percent after feature selection. Lastly, the true positives are 2030 before feature selection and 3202 after feature selection. In my last model we notice that the difference in overall score is almost non-existent, the recall score is pretty average compared to my other models, and my true positives are low for before feature selection and average for after feature selection. With all that being said, I do still believe that this is my best model due to the fact that the score is high for both models, and that the recall and true positive are both very good compared to my other models. It has a good balance of all the important information which is why it is my best model.

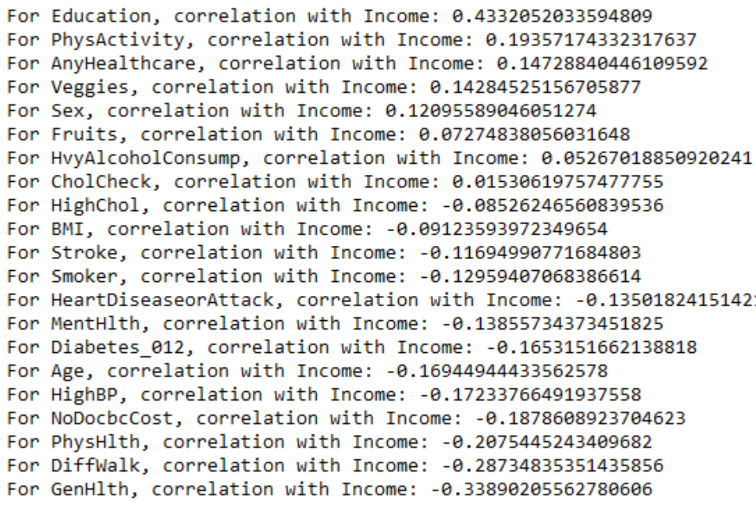
After each observation, we recorded the best model to be logistic regression using SMOTE for our imbalanced dataset. We feel like this may be due to the dataset being consistent of almost all binary/categorical data. The best model was classified as the best being it had a good overall score,recall, and true positive/true negative ratio. Using the best model, we are going to draw logical conclusions and conduct an exploratory analysis to see what information we can scrape, as well as what we can do with the information discovered.

Warren’s health conclusions

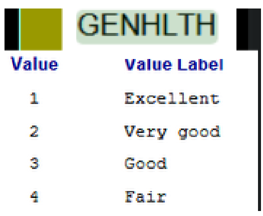
My question was if the combined health categories (general, physical, and mental) could be used to meaningfully predict diabetes. The answer I was immediately confronted with was a resounding no. Immediately plugging in the features to the logistic model, the overall accuracy was 63%. But that was nowhere near the terrible False Positive score of 12,494 (as opposed to the True Negative score of 19,453). Given that I already knew how hard it was to control the movement of False Positives, and that mental health and physical health had super low correlation scores (4% and 15% respectively), I wasn’t expecting anything to come of this. Also at this point, the slides were getting a little too numerous, so I discontinued researching this question.

Christopher’s Income Conclusions

One of the interesting correlations according to the ranker code was income, which was negatively correlated with diabetes. Since income does not have a direct purpose to diabetes, examining associations with income could help understand its negative correlation. We see that income correlates well with education, which may be because income affords higher level education (undergrad/grad). An important note is that we see general health being negatively correlated, which does not mean that as income goes up, general health gets worse, but rather with higher income, general health improves, and



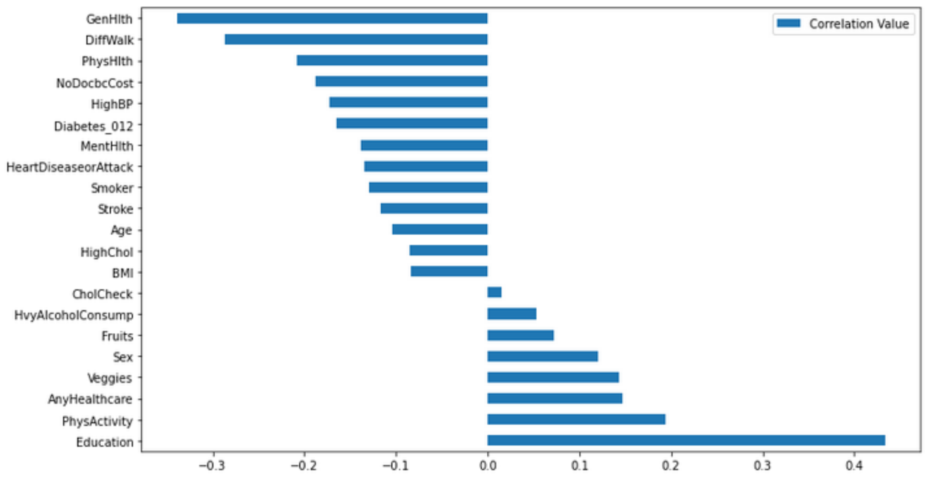
```
For Education, correlation with Income: 0.4332052033594809
For PhysActivity, correlation with Income: 0.19357174332317637
For AnyHealthcare, correlation with Income: 0.14728840446109592
For Veggies, correlation with Income: 0.14284525156705877
For Sex, correlation with Income: 0.12095589046051274
For Fruits, correlation with Income: 0.07274838056031648
For HvyAlcoholConsump, correlation with Income: 0.05267018850920241
For CholCheck, correlation with Income: 0.015306197574777555
For HighChol, correlation with Income: -0.08526246560839536
For BMI, correlation with Income: -0.09123593972349654
For Stroke, correlation with Income: -0.11694990771684803
For Smoker, correlation with Income: -0.12959407068386614
For HeartDiseaseorAttack, correlation with Income: -0.1350182415142
For MentHlth, correlation with Income: -0.13855734373451825
For Diabetes_012, correlation with Income: -0.1653151662138818
For Age, correlation with Income: -0.16944944433562578
For HighBP, correlation with Income: -0.17233766491937558
For NoDocbcCost, correlation with Income: -0.1878608923704623
For PhysHlth, correlation with Income: -0.2075445243409682
For DiffWalk, correlation with Income: -0.28734835351435856
For GenHlth, correlation with Income: -0.33890205562780606
```



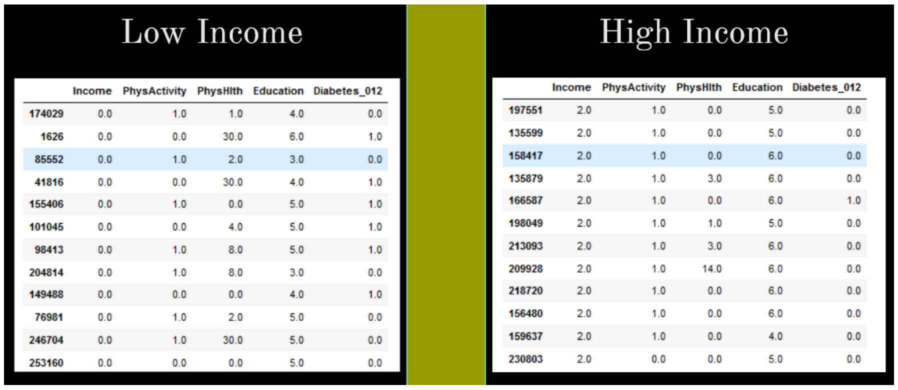
GENHLTH

Value	Value Label
1	Excellent
2	Very good
3	Good
4	Fair

that is because the dataset was categorized in a way where a lower number score indicates good health (same with physical health).



Other features such as difficulty walking, physical health, high blood pressure, and vegetables certainly are features that can be manipulated by money, whereas one can buy certain medicine/supplements (like insulin) to regulate their health to prevent diabetes, afford vegetables on their plate, and attend gyms as well as purchase tools for exercise to aid in better physical health.



Low Income

	Income	PhysActivity	PhysHlth	Education	Diabetes_012
174029	0.0	1.0	1.0	4.0	0.0
1626	0.0	0.0	30.0	6.0	1.0
85552	0.0	1.0	2.0	3.0	0.0
41816	0.0	0.0	30.0	4.0	1.0
155406	0.0	1.0	0.0	5.0	1.0
101045	0.0	0.0	4.0	5.0	1.0
98413	0.0	1.0	8.0	5.0	1.0
204814	0.0	1.0	8.0	3.0	0.0
149488	0.0	0.0	0.0	4.0	1.0
76981	0.0	1.0	2.0	5.0	0.0
246704	0.0	1.0	30.0	5.0	0.0
253160	0.0	0.0	0.0	5.0	0.0

High Income

	Income	PhysActivity	PhysHlth	Education	Diabetes_012
197551	2.0	1.0	0.0	5.0	0.0
135599	2.0	1.0	0.0	5.0	0.0
158417	2.0	1.0	0.0	6.0	0.0
135879	2.0	1.0	3.0	6.0	0.0
166587	2.0	1.0	0.0	6.0	1.0
198049	2.0	1.0	1.0	5.0	0.0
213093	2.0	1.0	3.0	6.0	0.0
209928	2.0	1.0	14.0	6.0	0.0
218720	2.0	1.0	0.0	6.0	0.0
156480	2.0	1.0	0.0	6.0	0.0
159637	2.0	1.0	0.0	4.0	0.0
230803	2.0	0.0	0.0	5.0	0.0

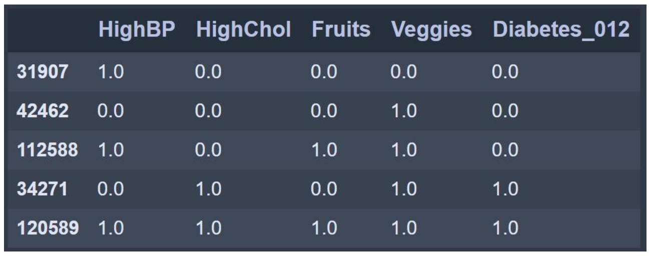
Just from randomly sampling twelve participants, we see everyone except one has done physical activity in the past thirty days. Compared to the low income sampling, we see the number of people with a good physactivity record decreases, as well as their physhlth trait. From seeing the high income sample, we notice that out of twelve random samples, only one recorded having diabetes.

When predicting with the logistic regression model, we conducted more observations to indicate someone with diabetes based on their income and education. When inputting someone who would have low income and never attended school, they were predicted to have diabetes, and the same goes for someone with little education but high income. However when inputting someone with an education of undergrad with little income, they were predicted to not have diabetes. A result like this can possibly be explained where education provides the knowledge of diabetes, information as to how to avoid it, and its leading causes (observations such as this assist in such knowledge).

Information like this can be provided to advertisers so that they can sell affordable medicine to people with lower income as their target audience (since the lower the income the higher chance of diabetes). Other options could include affordable gym memberships and healthcare which would benefit both the patient and the seller.

Jason's Dietary Conclusions

Diet is one of those topics where we think if we start eating healthy, it could lower your chance of getting diabetes, and that is partially true. Assuming that you have decent to healthy scores on other columns that are more correlated to diabetes, diet plays a small role, but your diet could affect other columns within the dataset. For instance, certain fruits and vegetables can lower your blood pressure, and blood pressure plays a big role in getting diabetes.



	HighBP	HighChol	Fruits	Veggies	Diabetes_012
31907	1.0	0.0	0.0	0.0	0.0
42462	0.0	0.0	0.0	1.0	0.0
112588	1.0	0.0	1.0	1.0	0.0
34271	0.0	1.0	0.0	1.0	1.0
120589	1.0	1.0	1.0	1.0	1.0

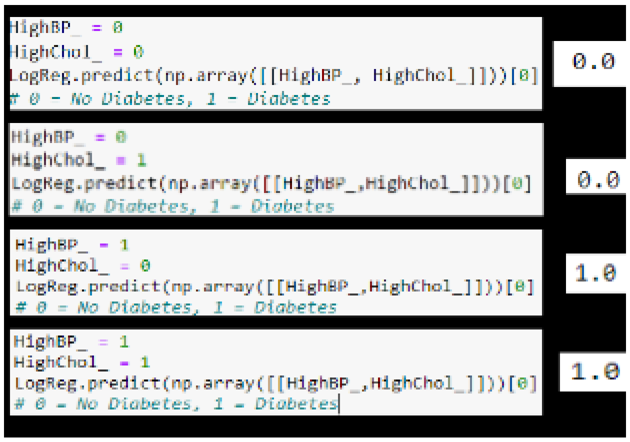
However, eating healthy doesn't guarantee that you won't get diabetes or a healthy blood pressure score, so physical health is also an important factor to consider, as diet alone can't prevent diabetes. Just want to mention that fruit and veggies have poor correlation with other columns since it doesn't play a big role with them.

After deciding that logistic regression with smote has the best performance for our dataset, the question I chose was does diet affect your chance of getting diabetes. The two columns that are considered dietary were fruit and veggies, so I used them for my predictions. Using the model, it predicted that if you eat fruit and veggies, you won't have diabetes, but if you don't eat both fruit and veggies, you most likely have diabetes. However, these predictions can be inaccurate since the model is only assessing fruit and veggies. Because it only uses two columns (fruits and veggies), the score is only ~55%, recall being 0.5, and only 3015 true positives.

In general, having a well-balanced diet can lower your chance of getting diabetes because a good diet could also very slightly affect your score in cholesterol, blood pressure, or BMI. Eating one fruit/veggies a day won't increase your chance of having diabetes, if anything, the benefits outweigh the negatives.

Andreas's Conclusions

After everyone in the group did their own model we decided that logistic regression using SMOTE was the most accurate model. Once we figured out which model was the best we proceeded to use that model to help answer our questions.The question that I(Andreas) chose was if high blood pressure and high cholesterol affected individuals into getting diabetes. Originally I believed that both features would equally affect diabetes. To be more specific I believed that having only high blood pressure, only high cholesterol, or both would increase the chances of someone having diabetes. But after using the predict and sample feature in python, I figured out that high cholesterol did not have any impact on diabetes whatsoever. In fact, the only time that diabetes was ever present was only if high blood pressure was there as well. So due to this I figured out that high cholesterol is nowhere near as effective on diabetes as blood pressure was.



```
HighBP_  = 0
HighChol_  = 0
LogReg.predict(np.array([[HighBP_, HighChol_]]))[0]
# 0 - No Diabetes, 1 - Diabetes
```

```
0.0
```

```
HighBP_  = 0
HighChol_  = 1
LogReg.predict(np.array([[HighBP_,HighChol_]]))[0]
# 0 - No Diabetes, 1 - Diabetes
```

```
0.0
```

```
HighBP_  = 1
HighChol_  = 0
LogReg.predict(np.array([[HighBP_,HighChol_]]))[0]
# 0 = No Diabetes, 1 = Diabetes
```

```
1.0
```

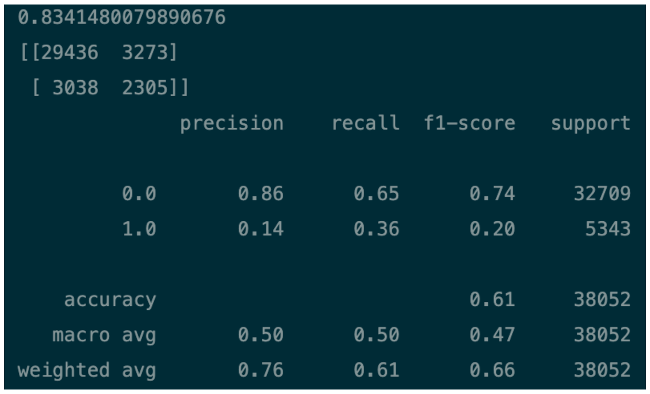
```
HighBP_  = 1
HighChol_  = 1
LogReg.predict(np.array([[HighBP_,HighChol_]]))[0]
# 0 - No Diabetes, 1 - Diabetes
```

```
1.0
```

Applying machine learning to this dataset has enabled us to discover different findings that we did not expect, as well as obtain knowledge. Our understanding of preprocessing, feature selection, and balancing-imbalance techniques have advanced from this project. We hope that this information would appeal to others in a way that would benefit them positively moving forward.

Before any balanced-imbalanced method was applied

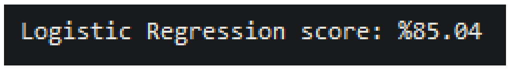
Naive Bayes



```
0.8341480079890676
[[29436  3273]
 [ 3038  2305]]
              precision    recall  f1-score   support

         0.0       0.86      0.65      0.74     32709
         1.0       0.14      0.36      0.20      5343

    accuracy                           0.61     38052
   macro avg       0.50      0.50      0.47     38052
weighted avg       0.76      0.61      0.66     38052
```

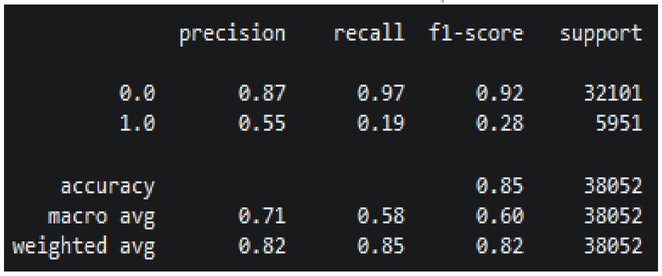
Highest precision, but at the cost of recall and true positives

Logistic Regression



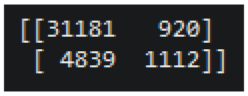
```
Logistic Regression score: %85.04
```

Classification report



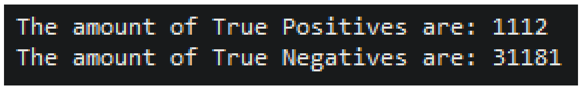
```
              precision    recall  f1-score   support

         0.0       0.87      0.97      0.92     32101
         1.0       0.55      0.19      0.28      5951

    accuracy                           0.85     38052
   macro avg       0.71      0.58      0.60     38052
weighted avg       0.82      0.85      0.82     38052
```

Classification matrix



```
[[31181   920]
 [ 4839  1112]]
```

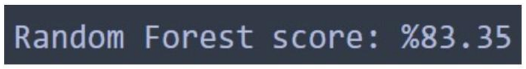
True Positives/True Negatives



```
The amount of True Positives are: 1112
The amount of True Negatives are: 31181
```

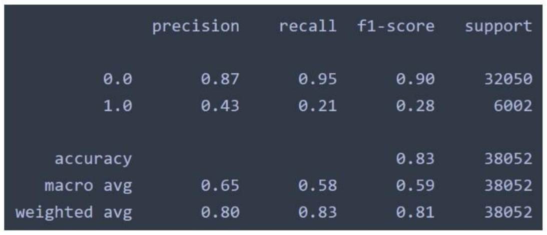
Applying no balancing gives a bad recall score and # of true positives since the data is mostly consisted of recorded non-diabetic individuals

Random Forest



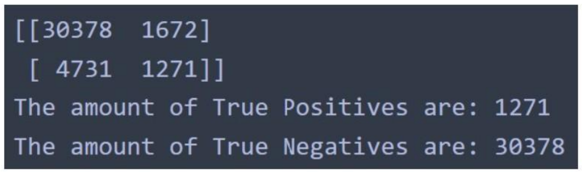
```
Random Forest score: %83.35
```

Classification Report



```
              precision    recall  f1-score   support

         0.0       0.87      0.95      0.90     32050
         1.0       0.43      0.21      0.28      6002

    accuracy                           0.83     38052
   macro avg       0.65      0.58      0.59     38052
weighted avg       0.80      0.83      0.81     38052
```

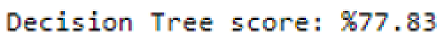
Classification Matrix (TP / TN)



```
[[30378  1672]
 [ 4731  1271]]
The amount of True Positives are: 1271
The amount of True Negatives are: 30378
```

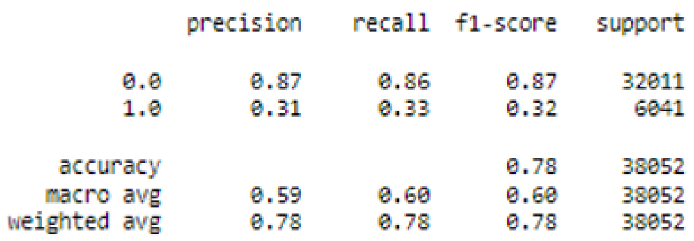
Applying no balance gave me the highest score, but has the worst recall and # of true positives, and inaccurate predictions

Decision Trees



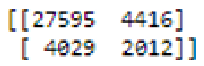
```
Decision Tree score: %77.83
```

Classification Report



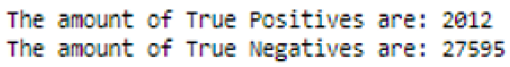
```
              precision    recall  f1-score   support

         0.0       0.87      0.86      0.87     32011
         1.0       0.31      0.33      0.32      6041

    accuracy                           0.78     38052
   macro avg       0.59      0.60      0.60     38052
weighted avg       0.78      0.78      0.78     38052
```

Classification Matrix



```
[[27595  4416]
 [ 4029  2012]]
```

True Positive/True Negative

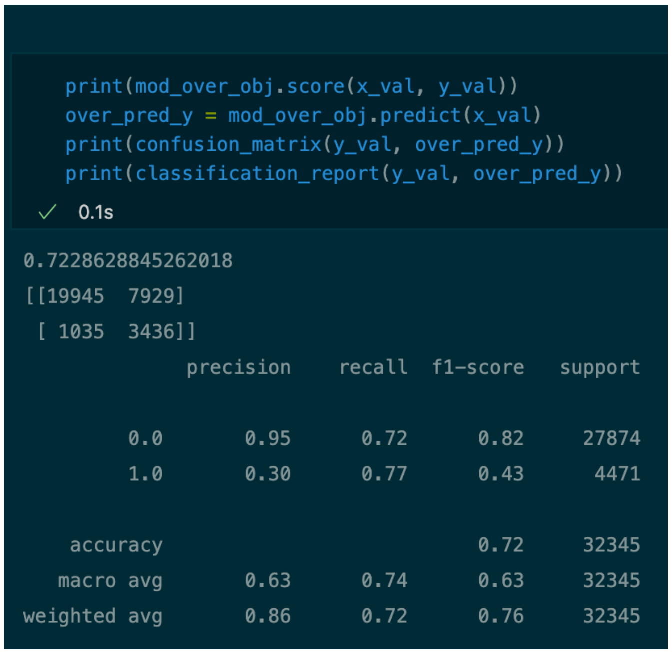


```
The amount of True Positives are: 2012
The amount of True Negatives are: 27595
```

Applying no balancing gave me the second highest overall score, horrible recall, and very low true positives

Warren’s Images (features selected)

Oversampling

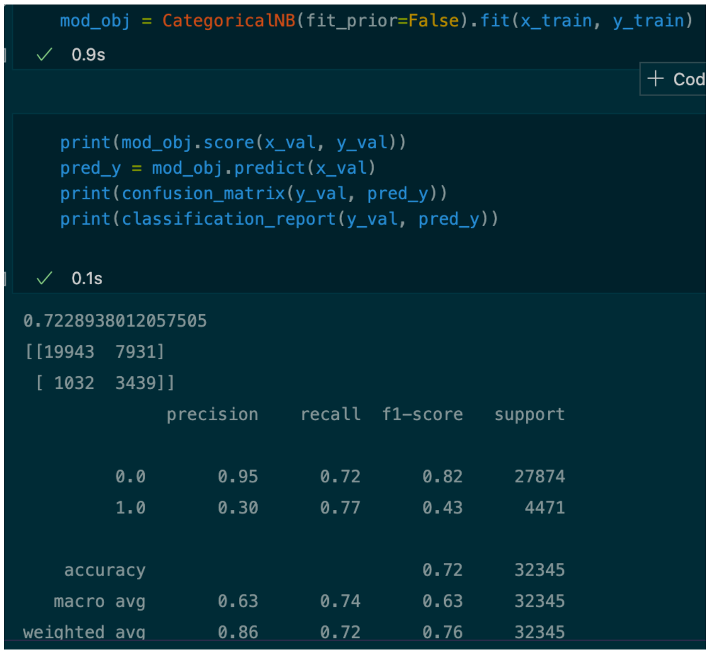


```
print(mod_over_obj.score(x_val, y_val))
over_pred_y = mod_over_obj.predict(x_val)
print(confusion_matrix(y_val, over_pred_y))
print(classification_report(y_val, over_pred_y))
```

✓ 0.1s

```
0.7228628845262018
[[19945  7929]
 [ 1035  3436]]
              precision    recall  f1-score   support

         0.0       0.95      0.72      0.82     27874
         1.0       0.30      0.77      0.43      4471

    accuracy                           0.72     32345
   macro avg       0.63      0.74      0.63     32345
weighted avg       0.86      0.72      0.76     32345
```

Balanced Weights



```
mod_obj = CategoricalNB(fit_prior=False).fit(x_train, y_train)
```

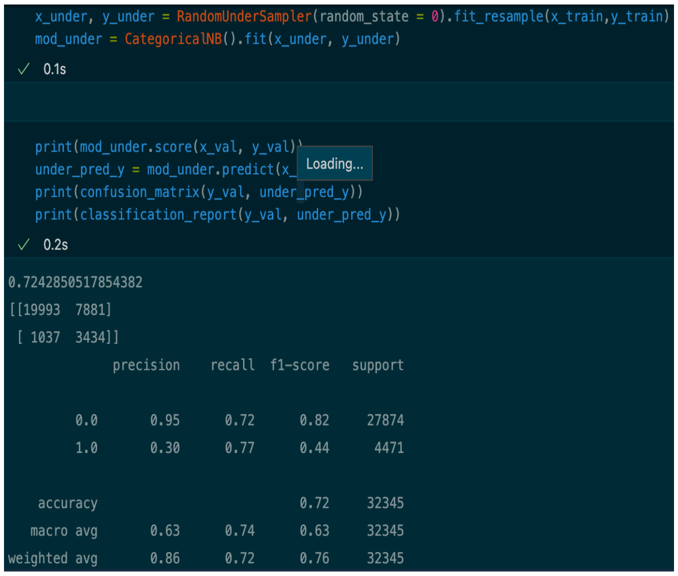
✓ 0.9s

```
print(mod_obj.score(x_val, y_val))
pred_y = mod_obj.predict(x_val)
print(confusion_matrix(y_val, pred_y))
print(classification_report(y_val, pred_y))
```

✓ 0.1s

```
0.7228938012057505
[[19943  7931]
 [ 1032  3439]]
              precision    recall  f1-score   support

         0.0       0.95      0.72      0.82     27874
         1.0       0.30      0.77      0.43      4471

    accuracy                           0.72     32345
   macro avg       0.63      0.74      0.63     32345
weighted avg       0.86      0.72      0.76     32345
```

Undersampling



```
x_under, y_under = RandomUnderSampler(random_state = 0).fit_resample(x_train,y_train)
mod_under = CategoricalNB().fit(x_under, y_under)
```

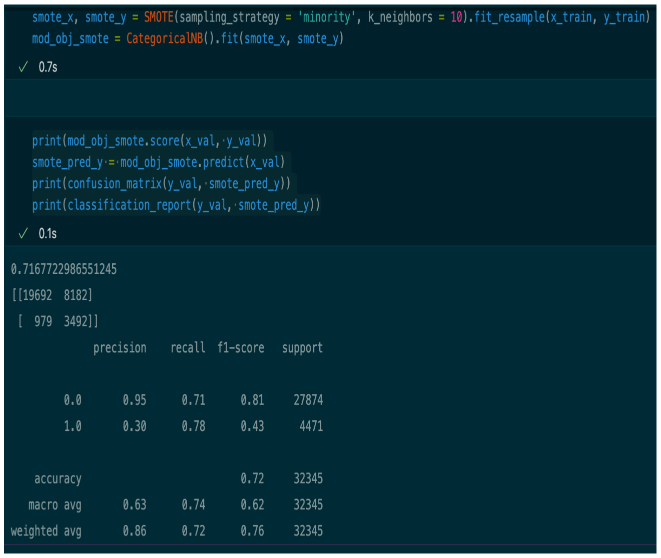
✓ 0.1s

```
print(mod_under.score(x_val, y_val))
under_pred_y = mod_under.predict(x_
print(confusion_matrix(y_val, under_pred_y))
print(classification_report(y_val, under_pred_y))
```

✓ 0.2s

```
0.7242850517854382
[[19993  7881]
 [ 1037  3434]]
              precision    recall  f1-score   support

         0.0       0.95      0.72      0.82     27874
         1.0       0.30      0.77      0.44      4471

    accuracy                           0.72     32345
   macro avg       0.63      0.74      0.63     32345
weighted avg       0.86      0.72      0.76     32345
```

SMOTE oversampling



```
smote_x, smote_y = SMOTE(sampling_strategy = 'minority', k_neighbors = 10).fit_resample(x_train, y_train)
mod_obj_smote = CategoricalNB().fit(smote_x, smote_y)
```

✓ 0.7s

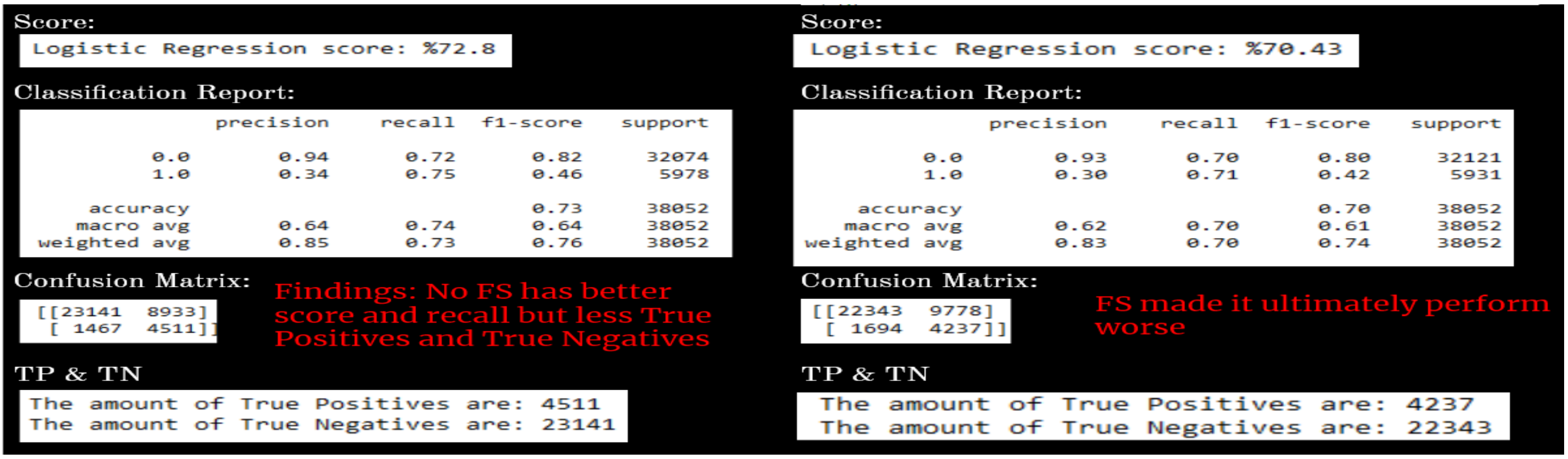
```
print(mod_obj_smote.score(x_val, y_val))
smote_pred_y = mod_obj_smote.predict(x_val)
print(confusion_matrix(y_val, smote_pred_y))
print(classification_report(y_val, smote_pred_y))
```

✓ 0.1s

```
0.7167722986551245
[[19692  8182]
 [  979  3492]]
              precision    recall  f1-score   support

         0.0       0.95      0.71      0.81     27874
         1.0       0.30      0.78      0.43      4471

    accuracy                           0.72     32345
   macro avg       0.63      0.74      0.62     32345
weighted avg       0.86      0.72      0.76     32345
```

NO FEATURE SELECTION Balanced Weights FEATURE SELECTION

Christopher Pineda's images



Score:

```
Logistic Regression score: %72.8
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.94      0.72      0.82     32074
         1.0       0.34      0.75      0.46      5978

    accuracy                           0.73     38052
   macro avg       0.64      0.74      0.64     38052
weighted avg       0.85      0.73      0.76     38052
```

Confusion Matrix:

```
[[23141  8933]
 [ 1467  4511]]
```

Findings: No FS has better score and recall but less True Positives and True Negatives

TP & TN

```
The amount of True Positives are: 4511
The amount of True Negatives are: 23141
```

Score:

```
Logistic Regression score: %70.43
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.93      0.70      0.80     32121
         1.0       0.30      0.71      0.42      5931

    accuracy                           0.70     38052
   macro avg       0.62      0.70      0.61     38052
weighted avg       0.83      0.70      0.74     38052
```

Confusion Matrix:

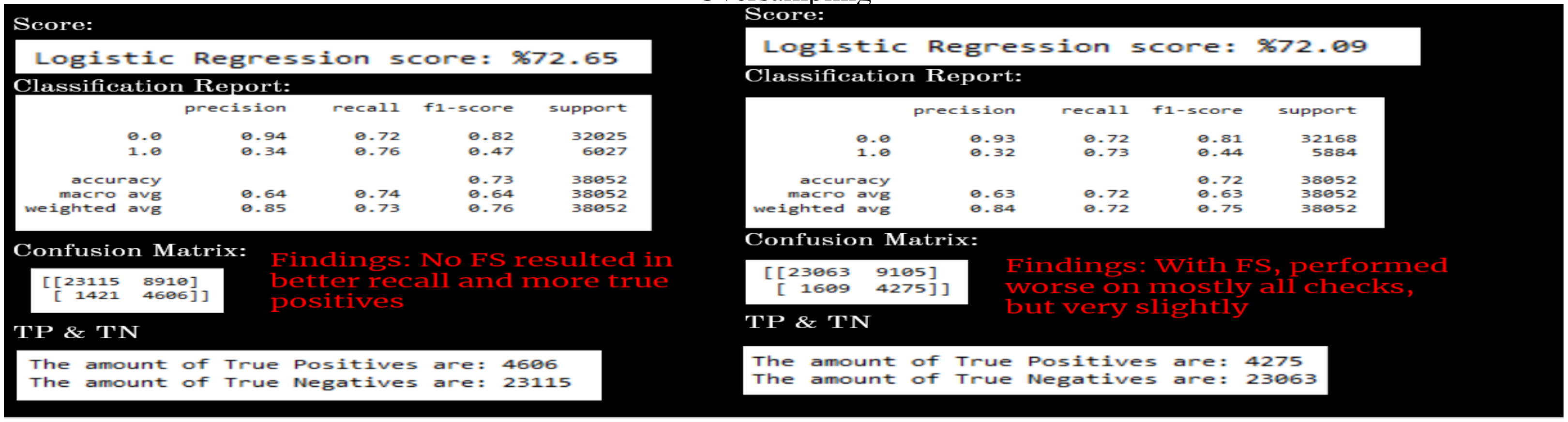
```
[[22343  9778]
 [ 1694  4237]]
```

FS made it ultimately perform worse

TP & TN

```
The amount of True Positives are: 4237
The amount of True Negatives are: 22343
```

Oversampling



Score:

```
Logistic Regression score: %72.65
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.94      0.72      0.82     32025
         1.0       0.34      0.76      0.47      6027

    accuracy                           0.73     38052
   macro avg       0.64      0.74      0.64     38052
weighted avg       0.85      0.73      0.76     38052
```

Confusion Matrix:

```
[[23115  8910]
 [ 1421  4606]]
```

Findings: No FS resulted in better recall and more true positives

TP & TN

```
The amount of True Positives are: 4606
The amount of True Negatives are: 23115
```

Score:

```
Logistic Regression score: %72.09
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.93      0.72      0.81     32168
         1.0       0.32      0.73      0.44      5884

    accuracy                           0.72     38052
   macro avg       0.63      0.72      0.63     38052
weighted avg       0.84      0.72      0.75     38052
```

Confusion Matrix:

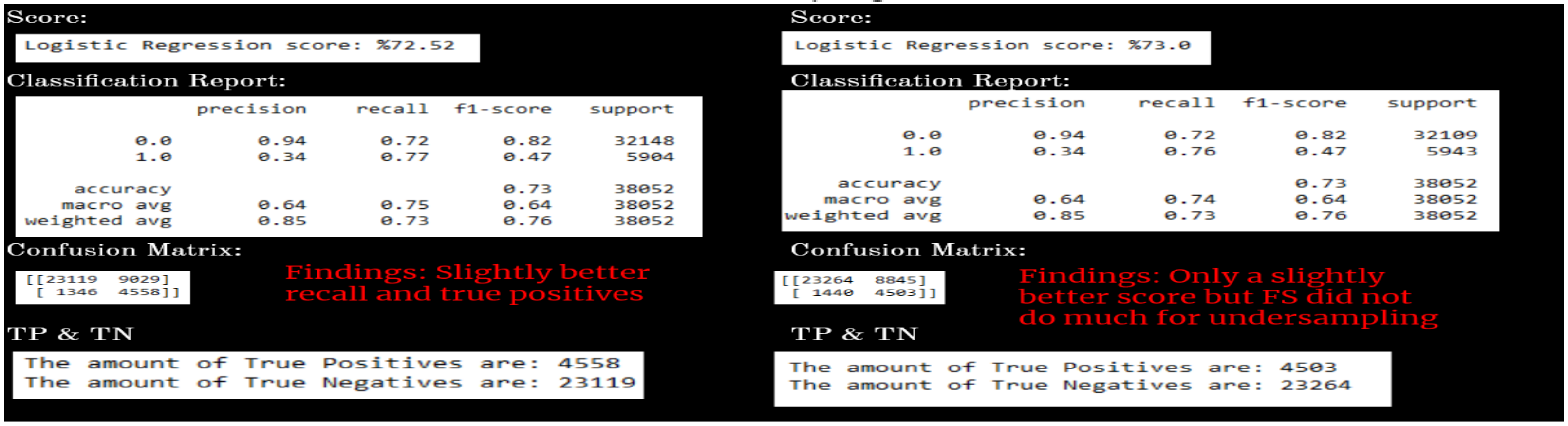
```
[[23063  9105]
 [ 1609  4275]]
```

Findings: With FS, performed worse on mostly all checks, but very slightly

TP & TN

```
The amount of True Positives are: 4275
The amount of True Negatives are: 23063
```

Undersampling



Score:

```
Logistic Regression score: %72.52
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.94      0.72      0.82     32148
         1.0       0.34      0.77      0.47      5904

    accuracy                           0.73     38052
   macro avg       0.64      0.75      0.64     38052
weighted avg       0.85      0.73      0.76     38052
```

Confusion Matrix:

```
[[23119  9029]
 [ 1346  4558]]
```

Findings: Slightly better recall and true positives

TP & TN

```
The amount of True Positives are: 4558
The amount of True Negatives are: 23119
```

Score:

```
Logistic Regression score: %73.0
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.94      0.72      0.82     32109
         1.0       0.34      0.76      0.47      5943

    accuracy                           0.73     38052
   macro avg       0.64      0.74      0.64     38052
weighted avg       0.85      0.73      0.76     38052
```

Confusion Matrix:

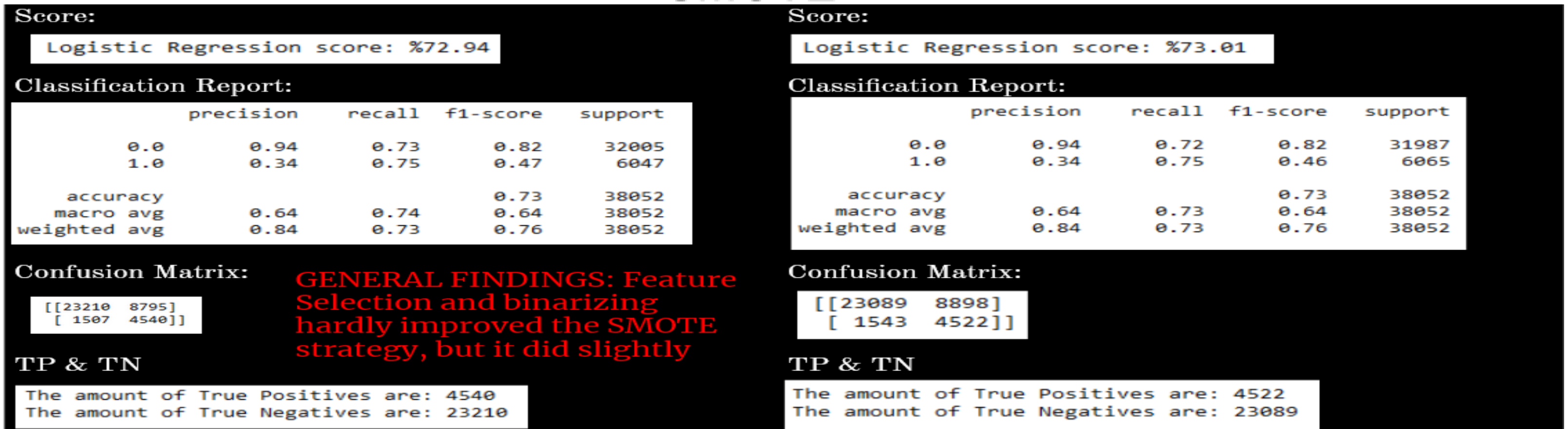
```
[[23264  8845]
 [ 1440  4503]]
```

Findings: Only a slightly better score but FS did not do much for undersampling

TP & TN

```
The amount of True Positives are: 4503
The amount of True Negatives are: 23264
```

SMOTE



Score:

```
Logistic Regression score: %72.94
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.94      0.73      0.82     32005
         1.0       0.34      0.75      0.47      6047

    accuracy                           0.73     38052
   macro avg       0.64      0.74      0.64     38052
weighted avg       0.84      0.73      0.76     38052
```

Confusion Matrix:

```
[[23210  8795]
 [ 1507  4540]]
```

GENERAL FINDINGS: Feature Selection and binarizing hardly improved the SMOTE strategy, but it did slightly

TP & TN

```
The amount of True Positives are: 4540
The amount of True Negatives are: 23210
```

Score:

```
Logistic Regression score: %73.01
```

Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.94      0.72      0.82     31987
         1.0       0.34      0.75      0.46      6065

    accuracy                           0.73     38052
   macro avg       0.64      0.73      0.64     38052
weighted avg       0.84      0.73      0.76     38052
```

Confusion Matrix:

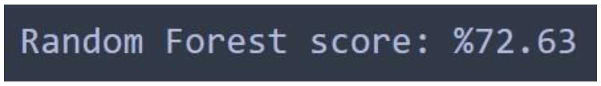
```
[[23089  8898]
 [ 1543  4522]]
```

TP & TN

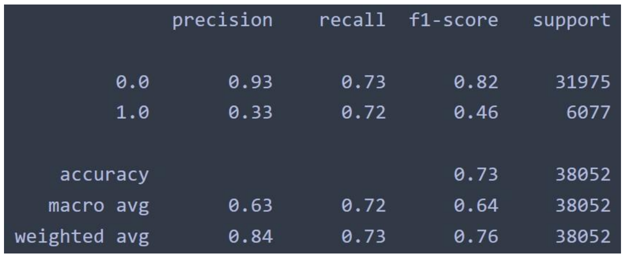
```
The amount of True Positives are: 4522
The amount of True Negatives are: 23089
```

Jason's Images (best result for each method)

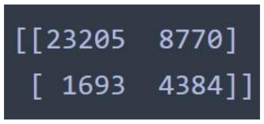
Oversampling(feature selection)



```
Random Forest score: %72.63
```

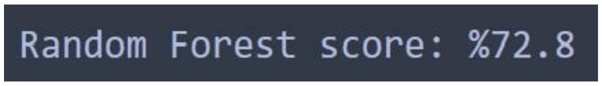


```
              precision    recall  f1-score   support

         0.0       0.93      0.73      0.82     31975
         1.0       0.33      0.72      0.46      6077

    accuracy                           0.73     38052
   macro avg       0.63      0.72      0.64     38052
weighted avg       0.84      0.73      0.76     38052
```

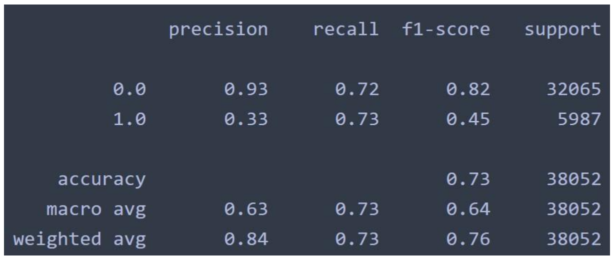


```
[[23205  8770]
 [ 1693  4384]]
```

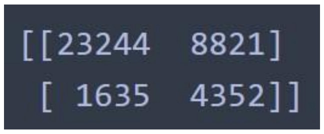
Balanced Weights(feature selection)



```
Random Forest score: %72.8
```

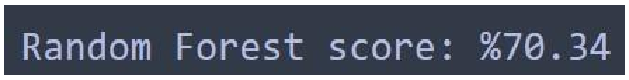


```
              precision    recall  f1-score   support

         0.0       0.93      0.72      0.82     32065
         1.0       0.33      0.73      0.45      5987

    accuracy                           0.73     38052
   macro avg       0.63      0.73      0.64     38052
weighted avg       0.84      0.73      0.76     38052
```

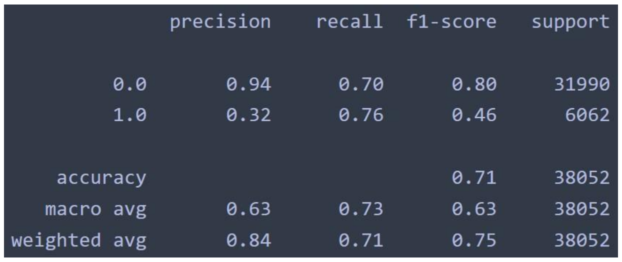


```
[[23244  8821]
 [ 1635  4352]]
```

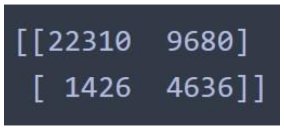
Undersampling (no feature selection)



```
Random Forest score: %70.34
```

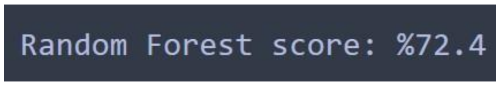


```
              precision    recall  f1-score   support

         0.0       0.94      0.70      0.80     31990
         1.0       0.32      0.76      0.46      6062

    accuracy                           0.71     38052
   macro avg       0.63      0.73      0.63     38052
weighted avg       0.84      0.71      0.75     38052
```

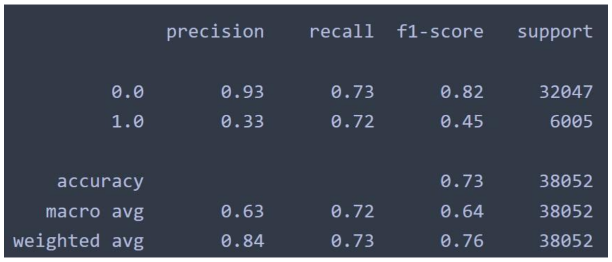


```
[[22310  9680]
 [ 1426  4636]]
```

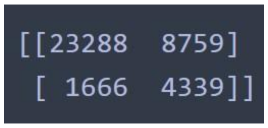
SMOTE oversampling (feature selection)



```
Random Forest score: %72.4
```

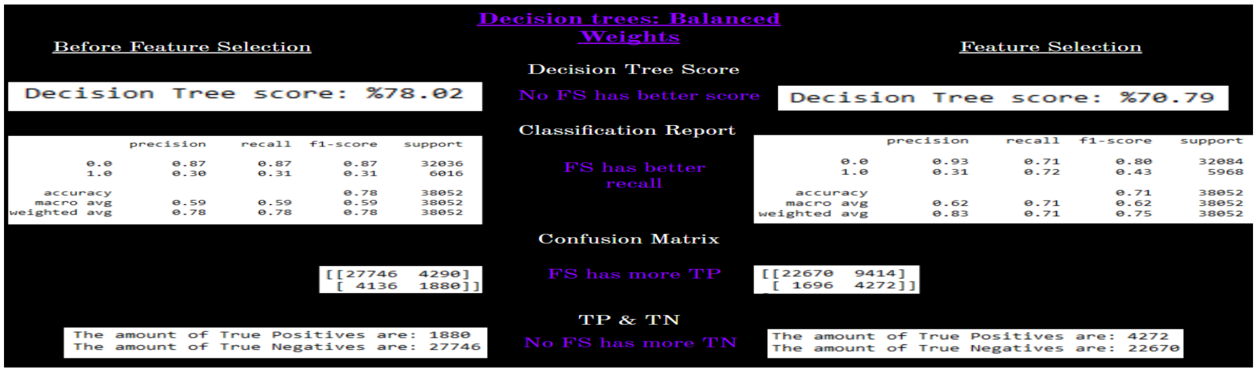


```
              precision    recall  f1-score   support

         0.0       0.93      0.73      0.82     32047
         1.0       0.33      0.72      0.45      6005

    accuracy                           0.73     38052
   macro avg       0.63      0.72      0.64     38052
weighted avg       0.84      0.73      0.76     38052
```



```
[[23288  8759]
 [ 1666  4339]]
```

Andreas's Images (Top Results)



Decision trees: Balanced Weights

Before Feature Selection

```
Decision Tree score: %78.02

              precision    recall  f1-score   support

         0.0       0.87      0.87      0.87     32036
         1.0       0.30      0.31      0.31      6016

    accuracy                           0.78     38052
   macro avg       0.59      0.59      0.59     38052
weighted avg       0.78      0.78      0.78     38052

[[27746  4290]
 [ 4136  1880]]

The amount of True Positives are: 1880
The amount of True Negatives are: 27746
```

Feature Selection

```
Decision Tree score: %70.79

              precision    recall  f1-score   support

         0.0       0.93      0.71      0.80     32084
         1.0       0.31      0.72      0.43      5968

    accuracy                           0.71     38052
   macro avg       0.62      0.71      0.62     38052
weighted avg       0.83      0.71      0.75     38052

[[22670  9414]
 [ 1696  4272]]

The amount of True Positives are: 4272
The amount of True Negatives are: 22670
```

Decision Tree Score

No FS has better score

Classification Report

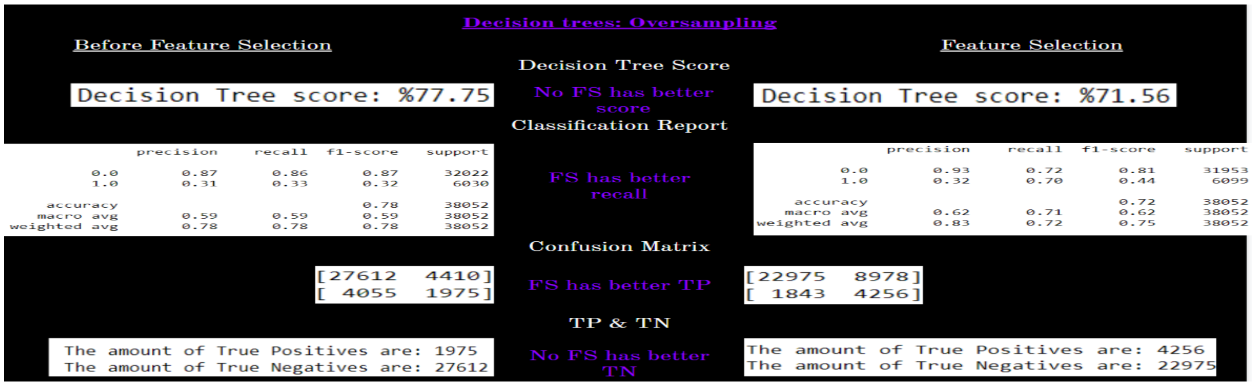
FS has better recall

Confusion Matrix

FS has more TP

TP & TN

No FS has more TN



Decision trees: Oversampling

Before Feature Selection

```
Decision Tree score: %77.75

              precision    recall  f1-score   support

         0.0       0.87      0.86      0.87     32022
         1.0       0.31      0.33      0.32      6030

    accuracy                           0.78     38052
   macro avg       0.59      0.59      0.59     38052
weighted avg       0.78      0.78      0.78     38052

[27612   4410]
[ 4055   1975]

The amount of True Positives are: 1975
The amount of True Negatives are: 27612
```

Feature Selection

```
Decision Tree score: %71.56

              precision    recall  f1-score   support

         0.0       0.93      0.72      0.81     31953
         1.0       0.32      0.70      0.44      6099

    accuracy                           0.72     38052
   macro avg       0.62      0.71      0.62     38052
weighted avg       0.83      0.72      0.75     38052

[22975   8978]
[ 1843   4256]

The amount of True Positives are: 4256
The amount of True Negatives are: 22975
```

Decision Tree Score

No FS has better score

Classification Report

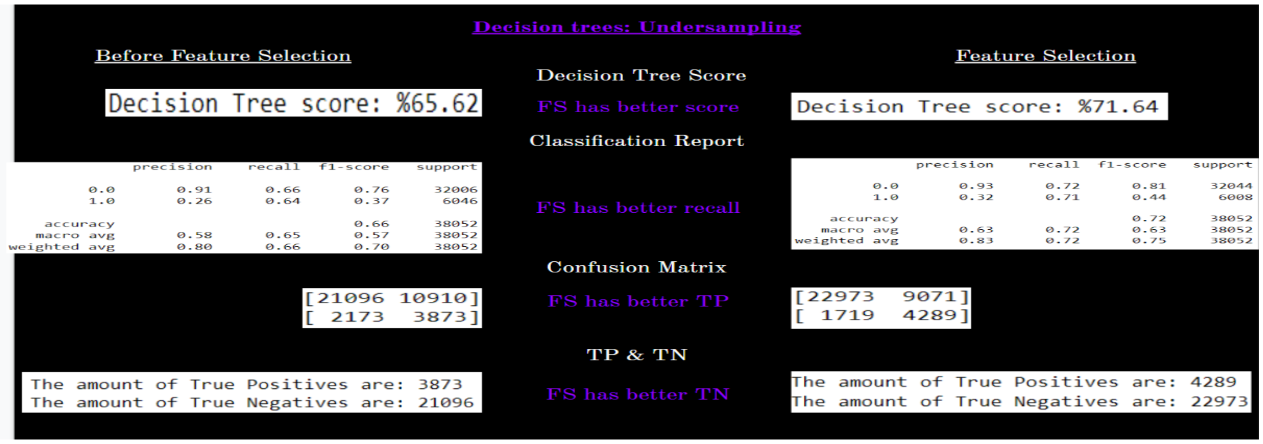
FS has better recall

Confusion Matrix

FS has better TP

TP & TN

No FS has better TN



Decision trees: Undersampling

Before Feature Selection

```
Decision Tree score: %65.62

              precision    recall  f1-score   support

         0.0       0.91      0.66      0.76     32006
         1.0       0.26      0.64      0.37      6046

    accuracy                           0.66     38052
   macro avg       0.58      0.65      0.57     38052
weighted avg       0.80      0.66      0.70     38052

[21096 10910]
[ 2173   3873]

The amount of True Positives are: 3873
The amount of True Negatives are: 21096
```

Feature Selection

```
Decision Tree score: %71.64

              precision    recall  f1-score   support

         0.0       0.93      0.72      0.81     32044
         1.0       0.32      0.71      0.44      6008

    accuracy                           0.72     38052
   macro avg       0.63      0.72      0.63     38052
weighted avg       0.83      0.72      0.75     38052

[22973   9071]
[ 1719   4289]

The amount of True Positives are: 4289
The amount of True Negatives are: 22973
```

Decision Tree Score

FS has better score

Classification Report

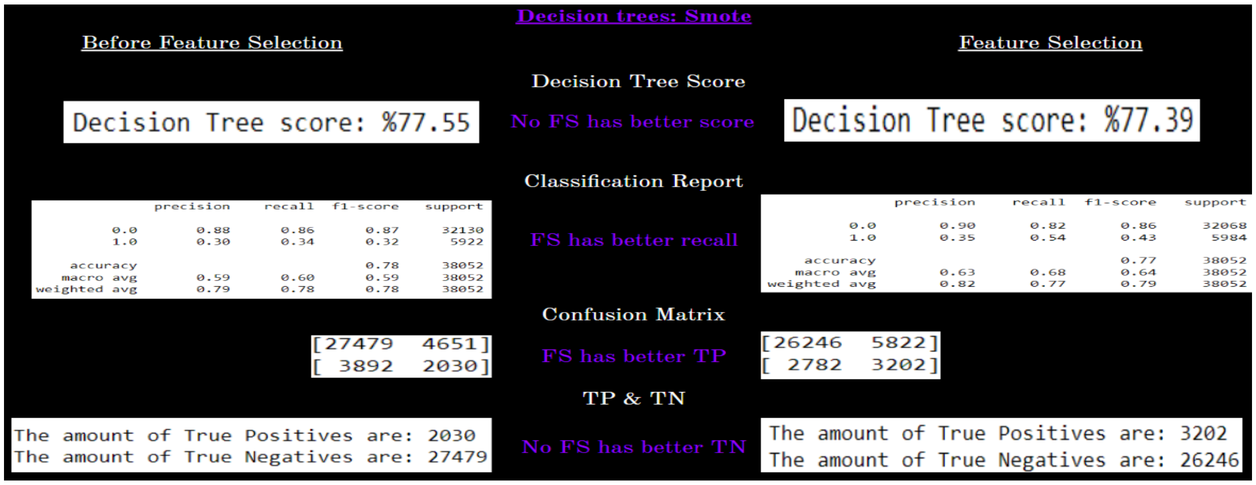
FS has better recall

Confusion Matrix

FS has better TP

TP & TN

FS has better TN



Decision trees: Smote

Before Feature Selection

```
Decision Tree score: %77.55

              precision    recall  f1-score   support

         0.0       0.88      0.86      0.87     32130
         1.0       0.30      0.34      0.32      5922

    accuracy                           0.78     38052
   macro avg       0.59      0.60      0.59     38052
weighted avg       0.79      0.78      0.78     38052

[27479   4651]
[ 3892   2030]

The amount of True Positives are: 2030
The amount of True Negatives are: 27479
```

Feature Selection

```
Decision Tree score: %77.39

              precision    recall  f1-score   support

         0.0       0.90      0.82      0.86     32068
         1.0       0.35      0.54      0.43      5984

    accuracy                           0.77     38052
   macro avg       0.63      0.68      0.64     38052
weighted avg       0.82      0.77      0.79     38052

[26246   5822]
[ 2782   3202]

The amount of True Positives are: 3202
The amount of True Negatives are: 26246
```

Decision Tree Score

No FS has better score

Classification Report

FS has better recall

Confusion Matrix

FS has better TP

TP & TN

No FS has better TN

Sources

- Dataset: https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset
- (Used for Christopher's conclusion)

  https://www.healthline.com/health/diabetes/medications-list#type-1-diabetes
- (Used for Christopher's LR)
- Dataset: https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset
- (Used for Andreas code)
- https://scikit-learn.org/stable/modules/tree.html
- (Used for Jason's conclusion)

  https://www.medicalnewstoday.com/articles/322284#fifteen-foods-that-help-to-lower-blo od-pressure
- (Used for Jason's RF)

  https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifi er.html
- (Warren's most frequently used source)
- https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.CategoricalNB.html

  Generally used sources
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split. html
- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.htm l
- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html