Individual Final Report
DATS 6103-10, Group Four
Kristin Levine's Report

Introduction:

Our project looked at MMR vaccination data. I wanted to examine the data using MMR rate as a target, Ben decided to use state as a target, and Russell wanted to run a regression.

Description of the Individual Work:

I decided to run the decision tree, random forest, and KNN algorithms on this data in order to classify whether a not a specific school had reached 95% MMR vaccination. In other words, when provided values for city, state/county, enrollment, type of school, and exemption records, I wanted to classify whether or not the school would have an MMR vaccination rate >= the recommended 95%. In this case, I was using the MMR rate as the target. I also ran the data using a >= 90% target rate.

Describe the portion of the work you did in detail:

Vaccination Rate as Target

I wrote all the code to preprocess the data to run the decision tree, random forest, and KNN algorithms using the MMR rate as the target variable. While WSJ was looking for a complete list of all schools, I wanted to look at schools for which we had data. The first course of action was to eliminate schools that possessed no enrollment data.

```
# Eliminate schools with no enrollment data
m = m.loc[(m['enroll' > 0)]
```

Because states reported vaccination data in different ways, there were two columns in the original file: MMR vaccination rate and overall vaccination rate (includes MMR and other diseases). Sometimes both values were provided and sometimes only one was given. Because I was focused specifically on MMR rates, I chose to take MMR when available and overall vaccination rate otherwise.

```
# Combine MMR and overall vaccination rates; use MMR if given and overall otherwise

m['vac_rate'] = m['mmr']

m['vac_rate'] = m['mmr'].where(m['mmr'] > 0, m['overall'])
```

The WSJ appeared to use -1 as a replacement for missing values in certain places. I eliminated all schools with a negative vaccination rate, since it is impossible to have a negative vaccination rate.

```
# Eliminate schools with a negative vaccination rate

m = m.loc[(m['vac_rate'] >= 0)]
```

The World Health Organization recommends a 95% vaccination rate to prevent measles; others set the value at 90%. For our target variable, I created new columns for both cases and set the variable type to Boolean.

```
# Is the vac_rate >= 95%?

m['at_least_95'] = (m['vac_rate'] >= 95)

# Is the vac_rate >= 90%?

m['at_least_90'] = (m['vac_rate'] >= 90)
```

The state column was grouped to compute the mean vaccination rate for each state.

```
# Find the mean vac_rate per state
```

```
state_mean = m[['state', 'vac_rate']]
state_mean = m.groupby('state').agg({'vac_rate': 'mean'}).reset_index()
state_mean = state_mean.rename(columns= {'vac_rate': 'state_mean'})
state_mean = state_mean.round(decimals=1)
```

Then, I wanted to substitute the mean values computed above back into the original frame to take the place of the state names. This was performed by converting state_mean to a dictionary and using replace().

```
# Add state_mean to original df
# First need to convert state_mean df to a dictionary, then use replace()
sm_dict = dict(zip(state_mean['state'], state_mean['state_mean']))
m['state_mean'] = m['state']
m = m.replace({'state_mean': sm_dict})
```

The same action was performed for the city and county features; if missing values remained they were replaced with zeros.

```
# Fill county NaN values with zero
m['county_mean'] = m['county_mean'].fillna(0)
```

Next, I looked at the type feature. A dictionary was created for values in the type feature, using zero for missing values.

```
# Rename type column
m = m.rename(columns={'type': 'type_of_school'})
# Enter school type variables as binary
ts_dict = {'Public': 1, 'Charter': 2, 'Private': 3, 'Kindergarten': 4}
```

```
m = m.replace({'type_of_school': ts_dict})
# Fill type_of_school NaN values with zero
m['type_of_school'] = m['type_of_school'].fillna(0)
```

In the file containing data for all states, the exemption data is separated into religious, medical and personal exemptions (xrel, xmed and xper, respectively). For the purpose of simplification, missing values were replaced with zeros and the exemption features were combined to measure total exemptions.

```
# Fill exemption values with zero
m['xrel'] = m['xrel'].fillna(0)
m['xmed'] = m['xmed'].fillna(0)
m['xper'] = m['xper'].fillna(0)
# Add all different types of exemptions together
m['xtotal'] = m['xrel'] + m['xmed'] + m['xper']
```

The state mean, city mean, county mean, school type, enrollment, xtotal, at least 95 and at least 90 features were selected and converted to binary for use in decision tree and random forest analysis.

```
# Select columns to use for DT
m_tree = m[['state_mean', 'city_mean', 'county_mean', 'type_of_school', 'enroll',
'xtotal', 'at_least_95', 'at_least_90']]
# Check to see if df has any NaN values
print(m_tree.isnull().sum())
print(m_tree.dtypes)
# Converting variables to binary for use in analysis
```

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

m_tree['state_mean'] = le.fit_transform(m_tree['state_mean'])

m_tree['city_mean'] = le.fit_transform(m_tree['city_mean'])

m_tree['county_mean'] = le.fit_transform(m_tree['county_mean'])

m_tree['type_of_school'] = le.fit_transform(m_tree['type_of_school'])

m_tree['enroll'] = le.fit_transform(m_tree['enroll'])

m_tree['xtotal'] = le.fit_transform(m_tree['xtotal'])

m_tree['at_least_95'] = le.fit_transform(m_tree['at_least_95'])

m_tree['at_least_90'] = le.fit_transform(m_tree['at_least_90'])
```

GUI Code:

I took the sample file from Prof. Jarafi's GitHut and tweaked it to work with my data. I did all the GUI code for the algorithms using MMR vaccinate rate (both 95 and 90) as the target rate.

We all contributed to writing our individual sections of the report and creating the slides for our presentation.

Results:

Decision Tree Results 95% target rate

In our test dataset of size 0.3, there were actually 6422 schools with an MMR rate >= 95 and 3166 schools with an MMR rate < 95.

This model found 6339 of the schools with 95% MMR rates and 1116 of the schools with lower vaccination rates. It mislabeled 83 schools with rates >= 95% as not meeting the 95% threshold; it also mislabeled 2050 schools as meeting the 95% threshold when they did not.

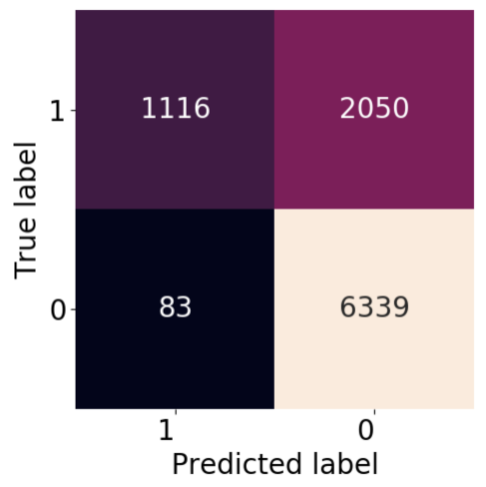
This model had more False Positive (Type I errors) than False Negative (Type II errors.)

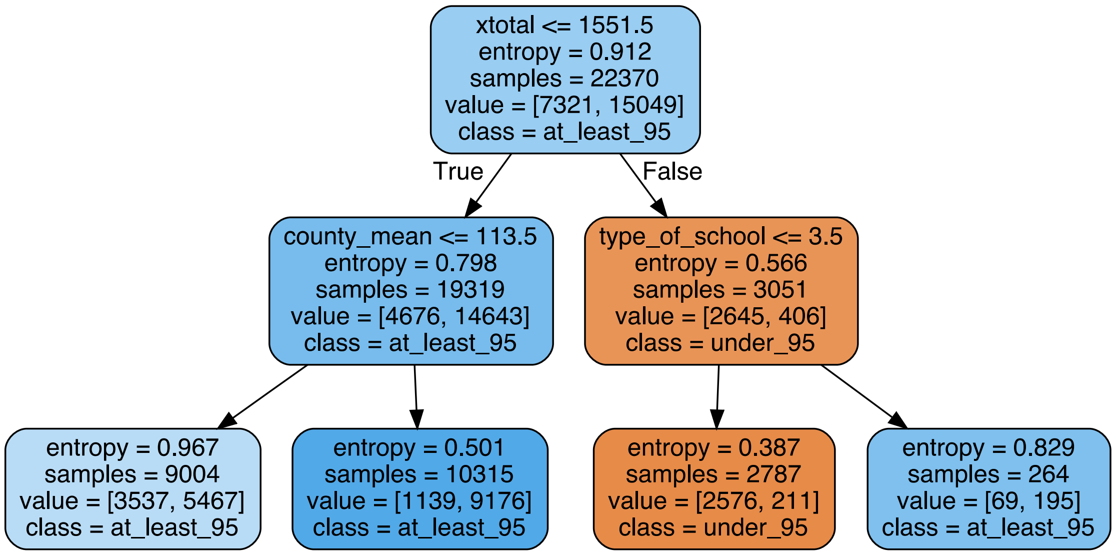
Results Using Entropy:

Classification Report:

```
    precision   recall  f1-score   support

0      0.93      0.35     0.51      3166

1      0.76      0.99     0.86      6422
```

```
   accuracy                          0.78      9588
  macro avg       0.84      0.67      0.68      9588
weighted avg       0.81      0.78      0.74      9588
Accuracy :  77.75344180225282
```





Decision Tree Results 90% target rate

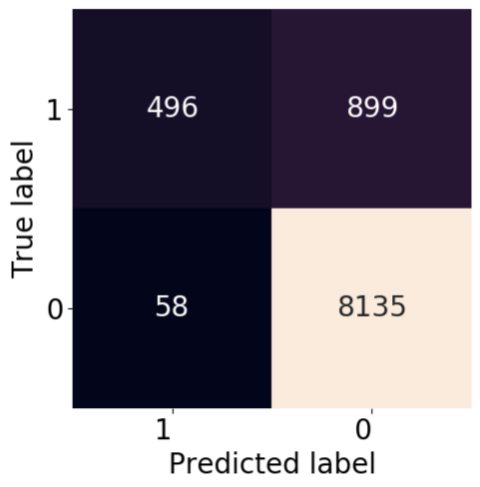
If we lowered the target rate to >= 90, there were actually 8193 schools that met this target, and 1395 that did not.

In this case, the model found 8135 of the schools that met the target rate and correctly labeled 496 schools that did not.

It mislabeled 58 schools with rates over 90% as not meeting the threshold; it also mislabeled 899 schools as meeting the 90% threshold when they did not.

This model had more False Positive (Type I errors) than False Negative (Type II errors.)

```
Results Using Entropy:

Classification Report:

              precision    recall  f1-score   support

           0       0.90      0.36      0.51      1395

           1       0.90      0.99      0.94      8193

    accuracy                           0.90      9588

   macro avg       0.90      0.67      0.73      9588

weighted avg       0.90      0.90      0.88      9588

Accuracy :  90.01877346683355
```



Random Forest Results 95%

For our Random Forest Algorithm, the results were similar: this model found 5655 of the schools with 95% MMR rates and 2072 of the schools with lower vaccination rates. It mislabeled 767 schools with rates >= 95% as not meeting the 95% threshold; it also mislabeled 1094 schools as meeting the 95% threshold when they did not.

Compared to the Decision Tree, the Random Forest reduced the number of Type 1 errors, however, it increased the number of Type II errors. However, overall, this model still had more False Positive (Type I errors) than False Negative (Type II errors.)

Results Using All Features:

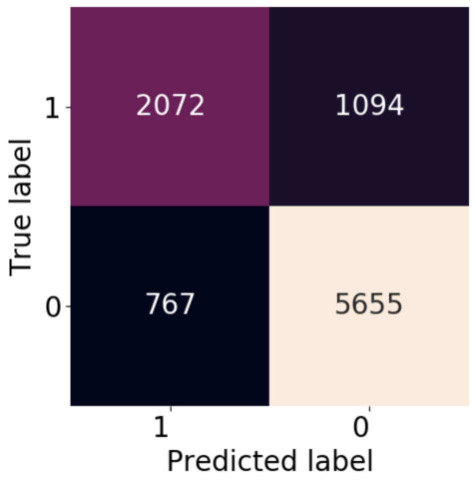
Classification Report:

```
              precision    recall  f1-score   support

           0       0.73      0.65      0.69      3166

           1       0.84      0.88      0.86      6422

    accuracy                           0.81      9588

   macro avg       0.79      0.77      0.78
9588

weighted avg       0.80      0.81      0.80
9588

Accuracy :  80.70504797663747

ROC_AUC :  86.5299946114637
```

Results Using K features:

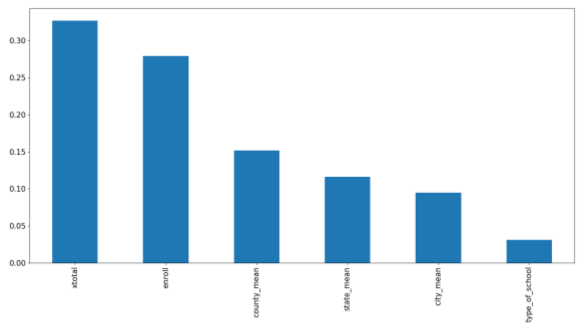
Classification Report:

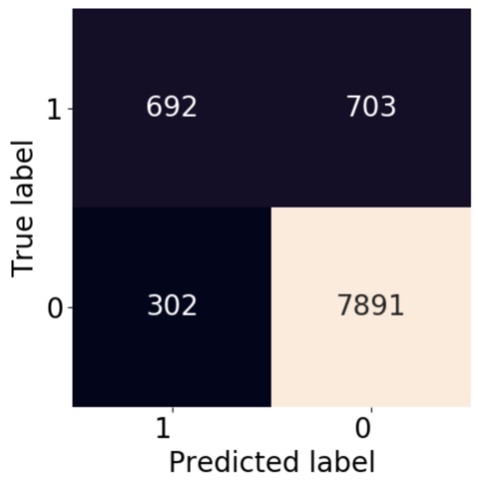
```
              precision    recall  f1-score   support

           0       0.73      0.65      0.69      3166

           1       0.84      0.88      0.86      6422

    accuracy                           0.81      9588

   macro avg       0.78      0.77      0.77
9588

weighted avg       0.80      0.81      0.80
9588

Accuracy :  80.59032123487692

ROC_AUC :  86.63533813507853
```



Random Forest 90%

With a 90% threshold, the results were very similar. It’s interesting to note, that the most important feature was the total exemption rate; enrollment was second. The type of school was the least important.





Results Using All Features:

Classification Report:

```
              precision    recall  f1-score   support

           0       0.69      0.50      0.58      1395

           1       0.92      0.96      0.94      8193

    accuracy                           0.89      9588

   macro avg       0.80      0.73      0.76
9588

weighted avg       0.88      0.89      0.89
9588
```

Accuracy : 89.3825615352524

ROC_AUC : 87.01679508733525

Results Using K features:

Classification Report:

```
              precision    recall  f1-score   support

           0       0.70      0.50      0.58      1395

           1       0.92      0.96      0.94      8193

    accuracy                           0.90      9588

   macro avg       0.81      0.73      0.76
9588

weighted avg       0.89      0.90      0.89
9588
```

Accuracy : 89.51814768460575

ROC_AUC : 86.84485882038474

The KNN results were very similar to the Random Forest

KNN 95%

Classification Report:

```
              precision    recall  f1-score   support

           0       0.71      0.64      0.67      3146

           1       0.83      0.87      0.85      6442

    accuracy                           0.80      9588

   macro avg       0.77      0.76      0.76
9588

weighted avg       0.79      0.80      0.79
9588
```

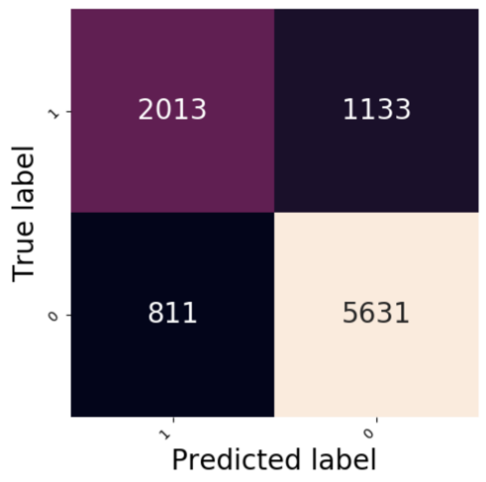
Accuracy : 79.72465581977471

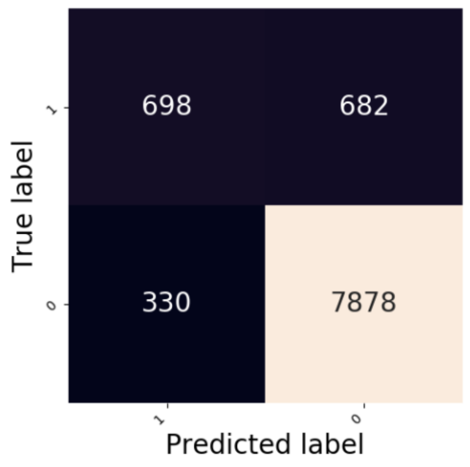
KNN 90%

Classification Report:

```
              precision    recall  f1-score   support

           0       0.68      0.51      0.58      1380

           1       0.92      0.96      0.94      8208

    accuracy                           0.89      9588

   macro avg       0.80      0.73      0.76
9588

weighted avg       0.89      0.89      0.89
9588
```

Accuracy : 89.44513975803086





In fact, all three models (Decision Tree, Random Forest, and KNN) were quite similar. With the 95% threshold, our accuracy ranged from 77.8% to 80.7% across the three models. With the 90% threshold, our accuracy ranged from 89.4% to 90.0%

Comparing Accuracy of All Three Models

	95% Threshold	**90% Threshold**
Decision Tree	77.8	90.0
Random Forest	80.7	89.4
KNN	79.7	89.4

Conclusions:

In the case of decision tree, random forest and KNN using vaccination rates as the target, one interesting conclusion was that the total number of exemptions was the most important feature when looking at whether or not a vaccination rate would be over a certain threshold. While this isn't exactly surprising, it does provide evidence that state laws probably do influence overall vaccination rates. If a state only allowed medical exemptions, they would probably have a higher vaccination rate than a state that made it easy to request a religious or personal exemption. For future research, I'd be interested in tracking down state laws and seeing if this is indeed the case. Additionally, it would be interesting to track down vaccination data from states not listed in this sample and see how that data works with these models. We had originally planned to use data from the University of Pittsburg as well, but we ended up taking a different approach. Also, while the data looking at vaccination rates over 95% was fairly balanced, when we changed the cutoff point to over 90% it became much less balanced. Finally, it would be interesting to look at school with a lower vaccination rate (perhaps under 80%) to see what they have in common.

Challenges:

Because we couldn't get together in person, it made it hard to share ideas and offer suggestions. We tried on WebEx calls, but it was still difficult to see and share code, even when we were shaping our screens. I wish we could have collaborated more on cleaning the data. Ben took a different approach (sometimes imputing rather than eliminating values) and I think it would have been great to combine our approaches.

The other challenge we faced was that I prefer to get things done in advance, while it seemed like my group members needed to wait until the last minute. I got a lot of my codes done in advance, but I think my group didn't have time to look at them until the very end. Again, I think this was made worse by our inability to meet in person. It did all come together in the end though!

I have to say, I really enjoyed and learned a lot from this project! The GUI was especially a challenge, but it was very satisfying when I got it to run. Despite the frustration of being unable to get together in person, I think everyone did a pretty good job working on their different pieces of the project.

Calculate the percentage of the code that you found or copied:

I'm not exactly sure how to do this. I wrote all the code (98 lines – All_measles_rates_Explore) to explore the data set. I wrote all the code for cleaning the data. (138 lines Export_clean_dataframe) I used the examples of the algorithms we went over in class and tweaked them to work with my cleaned data. (Saved as different files for Decision Tree, Random Forest, and KNN, for both a 95% and 90% target.)

For the GUI, I took the sample project, cut the parts I didn't need, and added copies of the algorithms to use with both a 95% and 90% target. That ended up being 1823 lines of code, though a lot of that was tweaking items from the sample project provided.

References:

Info about French Vaccine

http://www.rfi.fr/en/science-and-technology/20200227-france-s-pasteur-institute-develop-coronavirus-vaccine-candidate

Data Mining Class GitHub

https://github.com/amir-jafari/Data-Mining

Wall Street Journal Article

https://www.wsj.com/graphics/school-measles-rate-map/

Wall Street Journal Dataset

https://github.com/WSJ/measles-data