PREDICTING HEART DISEASE IN SENIORS USING FRUIT AND VEGETABLE INTAKE

MAY 13, 2017

RUSHABH RAMBHIA

1. Problem Statement

Can we predict chances of Heart Disease in seniors based on their daily fruit and vegetable intake?

2. Hypothesis

Seniors who eat more fruits and vegetables on a daily basis have lesser chances of Heart Disease.

3. Previous Studies

On initial research, I found two previous studies on this topic

- Study 2: [Published in 1996 in JAMA]
    - This study concluded that there was an inverse association between **fiber** intake and **Myocardial Infarction** in **male health professionals**
- Study 1: [Published in 2002 in American Society of Clinical Nutrition]
    - This study concluded that there was an inverse association of fruit and **vegetable** intake with the risk of cardiovascular disease in **general** US population
    - Conducted on **time series** data for 9,608 adults and data was collected from **1971 to 1975**

These studies confirm that there is existing scientific basis for my hypothesis. At this point, I think it is important for me to mention that my analysis is different from the above two in the following ways:

- Focus on Seniors (65+)
- Cross Sectional Data
- Data Source: BRFSS
- Feature of interest is daily fruit and vegetable intake
- Include other variables that could impact either Fruit Intake or Heart Disease in seniors
- Outcome is Categorical ie Heart Disease = Y or N

4. Data Set

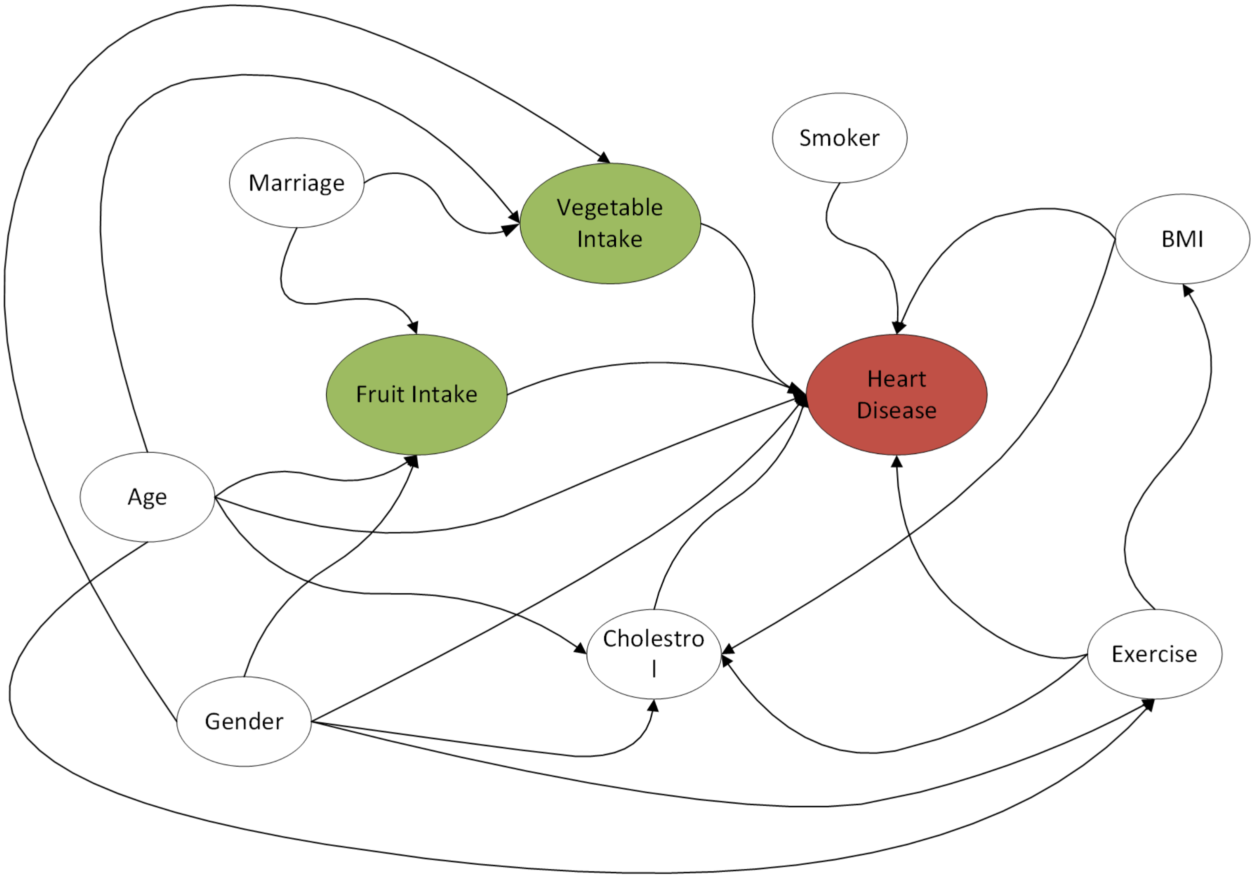
I am using 2015 BRFSS data set. This data is publicly available and can be downloaded from CDC website.

Here are key highlights of 2015 data set

- Cross Sectional data
- Over 441,000 people interviewed across 50 states in US
- Each interviewee answered over 120 mandatory and optional questions about their habits and chronic conditions
- There are over a total of 330 columns that include original responses to the interview questions along with calculated variables based on those responses
- Data feed available in TEXT as well as SAS format
- Data size is little less than 1 GB

5. Identify Key Columns

There are 330 columns in original dataset. I created a web of causation to minimize the list of columns to the ones that are relevant to this study.



6. Data Extraction

I followed below steps to prepare preliminary data set for my analysis.

1. Download SAS file from [CDC website]
2. Import SAS file using read_sas function
3. Extract columns relevant to my analysis. See Appendix A for list of columns selected from original data set
4. Save this data into a CSV which can be used in later steps. This way I do not have to load the original dataset which is huge and can take a lot of memory

7. Preliminary Data Preparation & Cleaning

In this phase I reviewed documentation of all fields I have selected for my analysis and create either calculated variables or dummy variables depending on how I thought I would use those fields in my analysis. See Appendix B for calculated and dummy variable list and logic.

Additionally I excluded records that met the following criteria:

- No response or NaN for response variable (_MICHD)
- No response to age (_AGEG5YR >= 14)
- Less than 65 years of age (AGEG5YR)
- Null values for cholesterol question (_RFCHOL)
- Null values for exercise (EXERANY2)
- Null values for dailyFruit (FRUIT1)
- Null value for dailyVeggie (FVGREEN)

I was left with over 127K records out of original 441K records.

8. Data Exploration

8.1 Response Variable:

My model will have one response variable (HD). It is a binary response variable with two possible values: 0 = doesn't have heart disease; 1 = has heart disease.

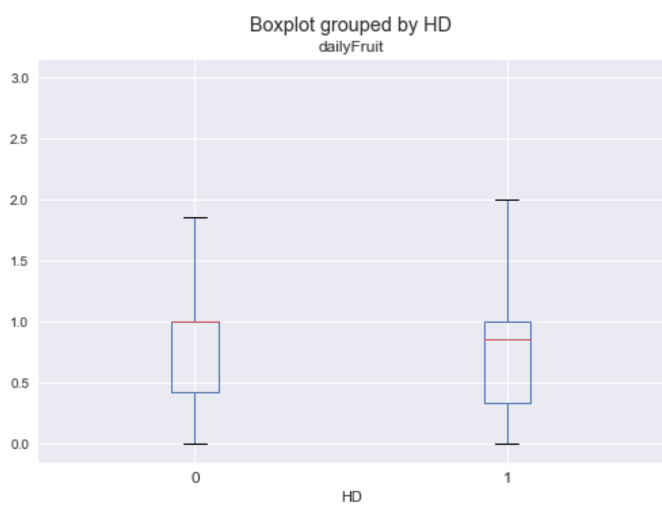
21,764 records out of 127,038 records have heart disease ie 17.13% have a positive result.

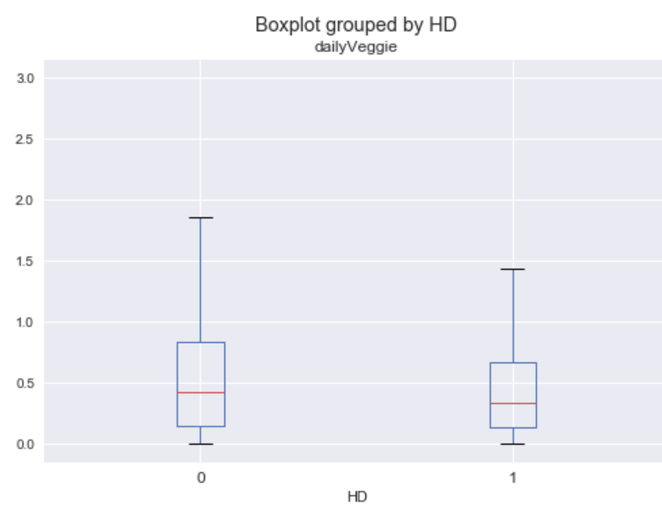
8.2 Features:

The two main features of my analysis are dailyFruit and dailyVeggie. I have calculated these features from original columns FRUIT1 and FVGREEN respectively. The functions used to calculate these features can be found in Appendix B.

These two features are continuous numbers and they range from 0 to 99. I have decided to exclude Nan values as well as outliers (values > 3) for both these features.

Below box plots show data ranges for dailyFruit and dailyVeggie features for HD=0 and HD=1 populations.

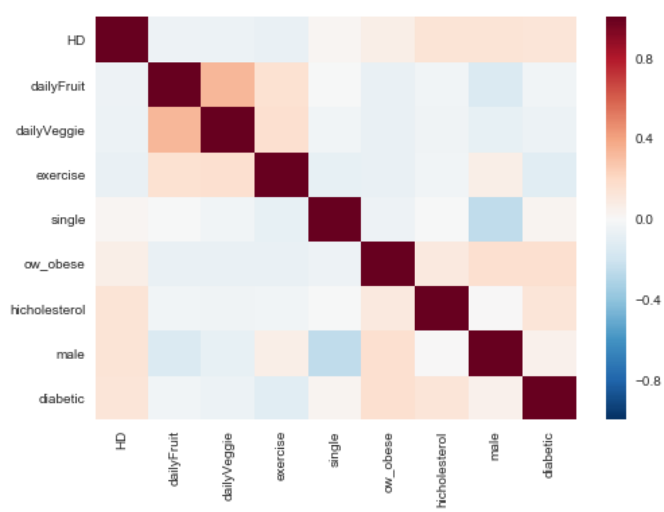




8.3 Correlation between variables:

Observations from attached correlation heat map:

- ❖ Columns dailyFruit and dailyVeggie are highly correlated.
- ❖ Columns diabetic, male and hicholesterol are correlated with response variable HD
- ❖ Columns ow_obese is correlated with male and diabetic



9. Modelling Process

9.1 Imbalanced data

On running Logistic Regression and using "accuracy" score to validate results, I observed that model's accuracy was as good as Null accuracy (83%). I also checked "recall" score of my initial model, and it was zero, thus indicating that the model was unable to predict any HD = 1.

```
# Initial analysis with Logistic Regression
feature_cols = ["dailyFruit","dailyVeggie"]
X = df[feature_cols]
y = df.HD
logReg = LogisticRegression()
print X.shape, y.shape
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)
logReg.fit(X_train,y_train)
y_pred_class = logReg.predict(X_test)
print "Accuracy: ", metrics.accuracy_score(y_test, y_pred_class)
print "Recall: ", metrics.recall_score(y_test, y_pred_class)
print metrics.confusion_matrix(y_test, y_pred_class)
```

```
(127038, 2) (127038L,)
Accuracy:  0.829722921914
Recall:  0.0
[[26352     0]
 [ 5408     0]]
```

I used KNN on this data and observed that recall was the highest (13.3%) with accuracy (75%) lower than Null accuracy for K = 1.

```
# Initial analysis with KNN
feature_cols = ["dailyFruit","dailyVeggie"]
X = df[feature_cols]
y = df.HD
print X.shape, y.shape
knn = KNeighborsClassifier(n_neighbors= 1)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)
knn.fit(X_train,y_train)
y_pred_class = knn.predict(X_test)
print "Accuracy: ", metrics.accuracy_score(y_test, y_pred_class)
print "Recall: ", metrics.recall_score(y_test, y_pred_class)
print metrics.confusion_matrix(y_test, y_pred_class)
```

```
(127038, 2) (127038L,)
Accuracy:  0.750409319899
Recall:  0.133136094675
[[23113  3239]
 [ 4688   720]]
```

Typically such problem arises due to imbalanced data. So I decided to apply following tactics:

Under sampling

I decided to balance my data so that number of records with HD = 0 is the same as the number of records with HD = 1. This means that null accuracy of this subset would be 50%.

Use of Cost Sensitive Measure for Validation

I also decided to use f1 score as it is important to know what percentage of actual positives is the model able to predict correctly. F1 score is combination of precision and recall. The greater the value the better the model prediction.

9.2 Using KNN estimator with dailyFruit and dailyVeggie

I used cross validation on under sampled data with KNN estimator and observed f1 score of 0.47 for K = 1.

```
# Analysis with KNN on Under sampled data
feature_cols = ["dailyFruit","dailyVeggie"]
X = df[feature_cols]
y = df.HD
#print X.shape, y.shape
scores_df = pd.DataFrame(columns = ("neighbors","metric"))

for i in range(1,10):
    knn = KNeighborsClassifier(n_neighbors= i)
    scores = cross_val_score(knn, X, y, cv=10, scoring='f1')
    scores_df.loc[i] = [i,scores.mean()]
#print scores_df
print scores_df.loc[scores_df['metric'].idxmax()]
```

```
neighbors    1.000000
metric       0.471976
Name: 1, dtype: float64
```

9.3 Using KNN estimator with dailyFruit only

Using the estimator with only one feature dailyFruit, I got a f1 score if 0.38. My conclusion was that it dailyFruit and dailyVeggie used together gave better prediction compared to using only dailyFruit.

```
# Analysis with KNN on Under sampled data
feature_cols = ["dailyFruit"]
X = df[feature_cols]
y = df.HD
#print X.shape, y.shape
scores_df = pd.DataFrame(columns = ("neighbors","metric"))

for i in range(1,10):
    knn = KNeighborsClassifier(n_neighbors= i)
    scores = cross_val_score(knn, X, y, cv=10, scoring='f1')
    scores_df.loc[i] = [i,scores.mean()]
#print scores_df
print scores_df.loc[scores_df['metric'].idxmax()]
```

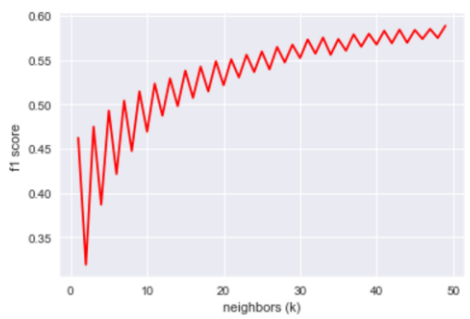
```
neighbors    1.000000
metric       0.386601
Name: 1, dtype: float64
```

9.4 Adding more features to KNN estimator and optimizing for better f1 score & accuracy

As I added more variables and tested for higher values of K, the f1 score as well as accuracy got better. I observed the best value of f1 score at 0.60 as well as accuracy = 0.60 at k = 50; after which I observed diminishing returns for both f1 score and accuracy.

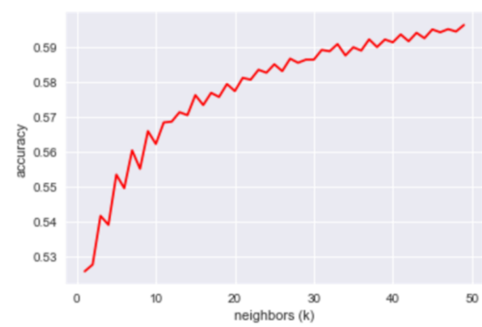
```
plt.plot(scores_df.neighbors, scores_df.metric, color='red')
plt.xlabel('neighbors (k)')
plt.ylabel('f1 score')
```

```
<matplotlib.text.Text at 0xf594f98>
```




```
plt.plot(scores_df.neighbors, scores_df.metric, color='red')
plt.xlabel('neighbors (k)')
plt.ylabel('accuracy')
```

```
<matplotlib.text.Text at 0xefe84a8>
```




```
# Analysis with KNN on under sampled data
feature_cols = ["dailyFruit","dailyVeggie","exercise","diabetic","hicholesterol"]
X = df[feature_cols]
y = df.HD
#print X.shape, y.shape
scores_df = pd.DataFrame(columns = ("neighbors","metric"))

for i in range(1,50):
    knn = KNeighborsClassifier(n_neighbors= i)
    scores = cross_val_score(knn, X, y, cv=10, scoring='f1')
    scores_df.loc[i] = [i,scores.mean()]
#print scores_df
print scores_df.loc[scores_df['metric'].idxmax()]
```

```
neighbors    49.000000
metric        0.588544
Name: 49, dtype: float64
```

Using the same estimator with train test split method gives 0.60 f1 score.

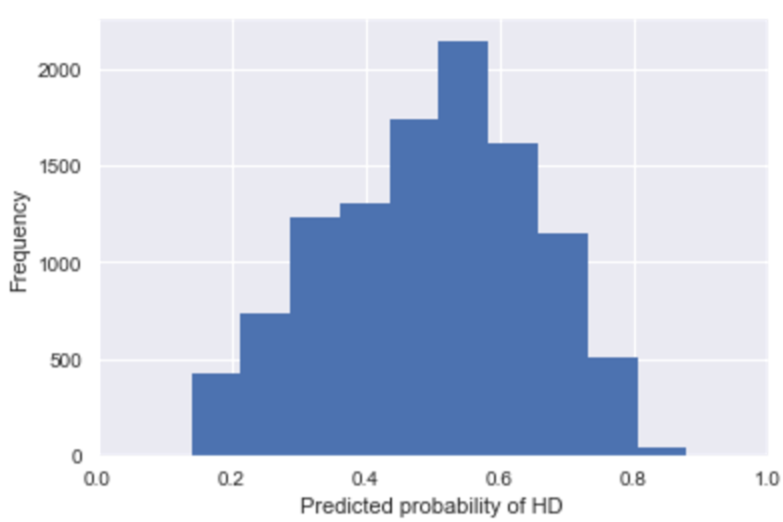
```
# Try predicting using train test split
feature_cols = ["dailyFruit","dailyVeggie","exercise","diabetic","hicholesterol"]
X = df[feature_cols]
y = df.HD
knn = KNeighborsClassifier(n_neighbors= 50)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=2)
knn.fit(X_train,y_train)
y_pred_class = knn.predict(X_test)
#print metrics.accuracy_score(y_test, y_pred_class)
print metrics.f1_score(y_test, y_pred_class)
print metrics.accuracy_score(y_test, y_pred_class)
print metrics.confusion_matrix(y_test, y_pred_class)
```

```
0.60447761194
0.600624885131
[[3215 2139]
 [2207 3321]]
```

The probability distribution of HD = 1 can be seen below for this model.

```
y_pred_prob = knn.predict_proba(X_test)[:, 1]
plt.hist(y_pred_prob)
plt.xlim(0, 1)
plt.xlabel('Predicted probability of HD')
plt.ylabel('Frequency')
```

```
<matplotlib.text.Text at 0xc155748>
```

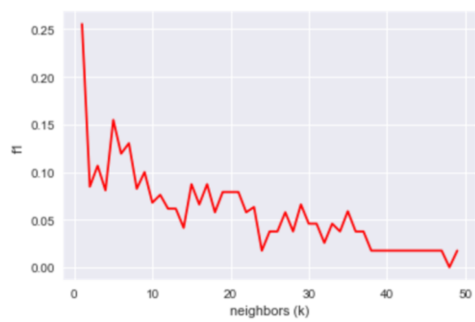


9.5 Estimating without dailyFruit and dailyVeggie features

I created a model with only diabetic, hicholsterol and exercise features (excluding dailyFruit and dailyVeggie features. This model had best f1 score of 0.25 at k = 1 and the score deteriorated with k > 1. Accuracy for this model was the best at 0.52 when k = 5.

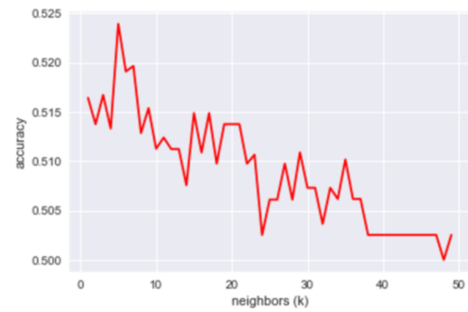
```
plt.plot(scores_df.neighbors, scores_df.metric, color='red')
plt.xlabel('neighbors (k)')
plt.ylabel('f1')
```

```
<matplotlib.text.Text at 0xf7f14e0>
```




```
plt.plot(scores_df.neighbors, scores_df.metric, color='red')
plt.xlabel('neighbors (k)')
plt.ylabel('accuracy')
```

```
<matplotlib.text.Text at 0xf4d7f60>
```



10. Conclusion

Based on my observations in sections 9.2, 9.4 and 9.5, I conclude that dailyFruit and dailyVeggie are better estimators of heart disease as compared to features such as diabetes, hi cholesterol or exercise. Yet best results are achieved only when all these 5 features are used together for predicting heart disease in seniors.

Appendices A:

List of columns from original data source.

<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>_MICHD</td><td></td></tr>
<tr><td>SEX</td><td></td></tr>
<tr><td>MARITAL</td><td>
Marital Status
Section: 7.6 Demographics
Type: Num
Column: 157
SAS Variable Name: MARITAL
Prologue:
Description: Are you: (marital status)
<table>
<tr><th>Value</th><th>Value Label</th><th>Frequency</th><th>Percentage</th><th>Weighted Percentage</th></tr>
<tr><td>1</td><td>Married</td><td>233,210</td><td>52.83</td><td>50.46</td></tr>
<tr><td>2</td><td>Divorced</td><td>59,406</td><td>13.46</td><td>10.79</td></tr>
<tr><td>3</td><td>Widowed</td><td>56,481</td><td>12.79</td><td>6.89</td></tr>
<tr><td>4</td><td>Separated</td><td>8,968</td><td>2.03</td><td>2.54</td></tr>
<tr><td>5</td><td>Never married</td><td>67,668</td><td>15.33</td><td>23.90</td></tr>
<tr><td>6</td><td>A member of an unmarried couple</td><td>12,627</td><td>2.86</td><td>4.71</td></tr>
<tr><td>9</td><td>Refused</td><td>3,096</td><td>0.70</td><td>0.70</td></tr>
</table>
</td></tr>
<tr><td>_RFBMI5</td><td>
Overweight or obese calculated variable
CalculatedVariables: 7.20 Calculated Variables
Type: Num
Column: 1993
SAS Variable Name: _RFBMI5
Prologue:
Description: Adults who have a body mass index greater than 25.00 (Overweight or Obese)
<table>
<tr><th>Value</th><th>Value Label</th><th>Frequency</th><th>Percentage</th><th>Weighted Percentage</th></tr>
<tr><td>1</td><td>No
Notes: 1200 <= _BMI5 < 2500 (_BMI5 has 2 implied decimal places)</td><td>138,130</td><td>31.29</td><td>32.31</td></tr>
<tr><td>2</td><td>Yes
Notes: 2500 <= _BMI5 < 9999</td><td>266,928</td><td>60.47</td><td>58.87</td></tr>
<tr><td>9</td><td>Don’t know/Refused/Missing
Notes: _BMI5 = 9999</td><td>36,398</td><td>8.24</td><td>8.82</td></tr>
</table>
</td></tr>
<tr><td>DIABETE3</td><td>
Ever told) you have diabetes
Section: 6.12 Chronic Health Conditions
Type: Num
Column: 117
SAS Variable Name: DIABETE3
Prologue:
Description: (Ever told) you have diabetes (If "Yes" and respondent is female, ask "Was this only when you were pregnant?". If Respondent says pre-diabetes or borderline diabetes, use response code 4.)
<table>
<tr><th>Value</th><th>Value Label</th><th>Frequency</th><th>Percentage</th><th>Weighted Percentage</th></tr>
<tr><td>1</td><td>Yes</td><td>57,256</td><td>12.97</td><td>10.48</td></tr>
<tr><td>2</td><td>Yes, but female told only during pregnancy—Go to Section 07.7.1 SEX</td><td>3,608</td><td>0.82</td><td>0.95</td></tr>
<tr><td>3</td><td>No—Go to Section 07.7.1 SEX</td><td>372,104</td><td>84.29</td><td>86.77</td></tr>
<tr><td>4</td><td>No, pre-diabetes or borderline diabetes—Go to Section 07.7.1 SEX</td><td>7,690</td><td>1.74</td><td>1.60</td></tr>
<tr><td>7</td><td>Don’t know/Not Sure—Go to Section 07.7.1 SEX</td><td>598</td><td>0.14</td><td>0.16</td></tr>
<tr><td>9</td><td>Refused—Go to Section 07.7.1 SEX</td><td>193</td><td>0.04</td><td>0.04</td></tr>
<tr><td>BLANK</td><td>Not asked or Missing</td><td>7</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

_RFCHOL

High Cholesterol Calculated Variable

CalculatedVariables: 5.2 Calculated Variables **Type:** Num

Column: 1898 **SAS Variable Name:** _RFCHOL

Prologue:

Description: Adults who have had their cholesterol checked and have been told by a doctor, nurse, or other health professional that it was high

Value	Value Label	Frequency	Percentage	Weighted Percentage
1	No Notes: BLOODCHO=1 and TOLDHI2 = 2	218,771	57.22	62.97
2	Yes Notes: BLOODCHO=1 and TOLDHI2 = 1	159,970	41.84	36.19
9	Don't know/Not Sure Or Refused/Missing Notes: BLOODCHO=1 and TOLDHI2 = 7 or 9 or Missing	3,561	0.93	0.84
BLANK	Missing Notes: BLOODCHO = 2 or 7 or 9 or Missing	59,154		

EXERANY2

Exercise in Past 30 Days

Section: 11.1 Exercise (Physical Activity) **Type:** Num

Column: 227 **SAS Variable Name:** EXERANY2

Prologue:

Description: During the past month, other than your regular job, did you participate in any physical activities or exercises such as running, calisthenics, golf, gardening, or walking for exercise?

Value	Value Label	Frequency	Percentage	Weighted Percentage
1	Yes	296,020	72.91	72.36
2	No	107,444	26.46	25.60
7	Don't know/Not Sure	602	0.15	0.13
9	Refused	1,946	0.48	1.91
BLANK	Not asked or Missing	35,444		

FRUIT1

How many times did you eat fruit?

Section: 10.2 Fruits & Vegetables **Type:** Num

Column: 212-214 **SAS Variable Name:** FRUIT1

Prologue:

Description: During the past month, not counting juice, how many times per day, week, or month did you eat fruit? Count fresh, frozen, or canned fruit.(Read only if necessary: "Your best guess is fine. Include apples, bananas, applesauce, oranges, grape fruit, fruit salad, watermelon, cantaloupe or musk melon, papaya, lychees, star fruit,)

Value	Value Label	Frequency	Percentage	Weighted Percentage
101 - 199	Times per day	183,688	44.55	42.08
201 - 299	Times per week	94,755	22.98	24.36
300	Less than one time per month	368	0.09	0.08
301 - 399	Times per month	109,859	26.65	26.41
555	Never	16,204	3.93	4.31
777	Don't know/Not sure	5,139	1.25	1.08
999	Refused	2,293	0.56	1.68
BLANK	Not asked or Missing	29,150		

FVGREEN

_AGEG5YR

Appendices B:

List of calculated and dummy variables

<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>HD</td><td>This is response variable. All records with _MICHD = 1 where set as 1 and all other values of _MICHD were set to 0.</td></tr>
<tr><td>dailyFruit</td><td>

```python
def fruit2daily_fruit(row):
    if row['FRUIT1'] >= 100 and row['FRUIT1'] < 200:
        val = row['FRUIT1']-100
    elif row['FRUIT1'] >= 200 and row['FRUIT1'] < 300:
        val = (row['FRUIT1']-200)/7
    elif row['FRUIT1'] == 300:
        val = 0.02
    elif row['FRUIT1'] > 300 and row['FRUIT1']<400:
        val = (row['FRUIT1'] - 300)/30
    elif row['FRUIT1'] == 555:
        val = 0
    else:
        val = float('NaN')
    return val
```

</td></tr>
<tr><td>dailyVeggie</td><td>

```python
def fvgreen2daily_veggie(row):
    if row['FVGREEN'] >= 100 and row['FVGREEN'] < 200:
        val = row['FVGREEN']-100
    elif row['FVGREEN'] >= 200 and row['FVGREEN'] < 300:
        val = (row['FVGREEN']-200)/7
    elif row['FVGREEN'] == 300:
        val = 0.02
    elif row['FVGREEN'] > 300 and row['FVGREEN']<400:
        val = (row['FVGREEN'] - 300)/30
    elif row['FVGREEN'] == 555:
        val = 0
    else:
        val = float('NaN')
    return val
```

</td></tr>
<tr><td>single</td><td rowspan="7">

```python
df['single'] = np.where(df.MARITAL==1,0,np.where(df.MARITAL==6,0,1))
df['diabetic'] = np.where(df.DIABETE3==1,1,np.where(df.DIABETE3==2,1,0))
df['male'] = np.where(df.SEX == 1,1,0)
df['hicholestrol'] = np.where(df._RFCHOL == 2, 1, 0)
df['ow_obese'] = np.where(df._RFBMI5 == 2, 1, 0)
df['exercise'] = np.where(df.EXERANY2 == 1, 1, 0)
df['senior'] = np.where(df._AGEG5YR >= 10, 1, 0)
```

</td></tr>
<tr><td>diabetic</td></tr>
<tr><td>hicholesterol</td></tr>
<tr><td>exercise</td></tr>
<tr><td>male</td></tr>
<tr><td>ow_obese</td></tr>
<tr><td>senior</td></tr>
</table>