```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


#651 pm
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.utils import resample
import matplotlib.pyplot as plt


# In[2]:



# Read in train and test data
df_train =
pd.read_csv(r'C:\Users\jeann\Downloads\BodySystemTrainTable.csv',
on_bad_lines='skip')
df_test =
pd.read_csv(r'C:\Users\jeann\Downloads\BodySystemTestTable.csv',
on_bad_lines='skip')


# In[3]:


# Remove the 'id', 'testtrain', and 'bs15lr' columns from the train set
df_train = df_train.drop(['id', 'TestTrain', 'bs15lr'], axis=1)

# Remove the 'id', 'hf', 'vcode', and 'bs15lr' columns from the test set
df_test = df_test.drop(['id', 'hf', 'vcode', 'bs15lr'], axis=1)


# In[4]:


# Fill the null values with 1s
df_train = df_train.fillna(1)
df_test = df_test.fillna(1)


# In[5]:



# Separate the dependent variable from the independent variables
X_train = df_train.drop('dm', axis=1)
y_train = df_train['dm']
```

```
X_test = df_test.drop('dm', axis=1)
y_test = df_test['dm']


# In[6]:



# Upsample the training dataset
X_train_pos = X_train[y_train == 1]
y_train_pos = y_train[y_train == 1]
X_train_neg = X_train[y_train == 0]
y_train_neg = y_train[y_train == 0]
X_train_pos_upsampled, y_train_pos_upsampled = resample(X_train_pos,
                                                        y_train_pos,
                                                        replace=True,

n_samples=X_train_neg.shape[0],
                                                        random_state=123)
X_train_upsampled = pd.concat([X_train_pos_upsampled, X_train_neg])
y_train_upsampled = pd.concat([y_train_pos_upsampled, y_train_neg])


# In[7]:


# Split the training dataset into training and validation sets
X_train_final, X_val, y_train_final, y_val =
train_test_split(X_train_upsampled,

y_train_upsampled,

test_size=0.2,

random_state=123)


# In[8]:


# Standardize the data
scaler = StandardScaler()
X_train_final_scaled = scaler.fit_transform(X_train_final)
X_val_scaled = scaler.transform(X_val)
X_test_scaled = scaler.transform(X_test)


# In[93]:


# Fit a logistic regression model with Lasso regularization to the
training data using cross-validation
model = LogisticRegression(max_iter=10000, penalty='l1', C=0.00002,
random_state=123, solver='saga')
```

```
model.fit(X_train_final_scaled, y_train_final)



# In[94]:


# Predict on the validation and test sets
y_val_pred = model.predict(X_val_scaled)
y_test_pred = model.predict(X_test_scaled)


# In[98]:


# Use the model to predict the incidence of diabetes in the test set
y_pred = model.predict(X_test_scaled)


# In[101]:


# Get the absolute values of the coefficients and their corresponding
feature names
coef_abs = abs(model.coef_[0])
feature_names = X_train.columns

# Combine the coefficients and feature names into a DataFrame
coef_df = pd.DataFrame({'feature': feature_names, 'coef': coef_abs})

# Sort the DataFrame by absolute coefficient value in descending order
coef_df = coef_df.sort_values(by='coef',
ascending=False).reset_index(drop=True)

# Print the features with the highest absolute coefficient values, there
are only 19 left in the list
print(coef_df.head(20))


# In[55]:


##now use the method
#that professor
#showed us to manually determine the contributing variables for each of
the direct variables. You can then
#create a network diagram based on that temporal analysis.

#From The BC with Time and Order spreadsheet I learned that:

#Circulatory (bs7lr) is predicted by nothing. (No other variable ever
occurs before bs7lr.)

#Endocrine (bs3lr) is only predicted by circulatory.
```

#Mental disorders (bs5lr) is only predicted by circulatory .

#Nervous system (bs6lr) is predicted by circulatory, repiratory, digestive, genitourniary, musculoskeletal and v_codes.

#Digestive (bs9lr) is predicted by nothing else.

#Neoplasms (bs2lr) is predicted by endocrine, mental, circulatory, respiratory, digestive, genitourinary, pregnancy, musculoskeltal.

#Injury and poisoning (bs17lr) is predicted by v_codes, Rare DX.

#Respiratory (bs8lr) is predicted by digestive, musculoskeletal.

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]: