In [1]:
```
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

In [2]:
```
#  import required packages
from consort import ConsortGraph
from matplotlib.lines import Line2D
from scipy.stats import chi2_contingency
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import json
import sklearn
from sklearn.preprocessing import LabelEncoder
import networkx as nx
from sklearn.model_selection import train_test_split
import xgboost as xgb
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from matplotlib.pylab import rcParams
from itertools import combinations
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
```

In [3]:
```
class PredictaVie_Preprocess:
    def __init__(self, dataframe):
        self.dataframe = dataframe

    def event_pie_chart(self, column_name):
        counts = self.dataframe[column_name].value_counts()
        total_count = counts.sum()
        ratios = counts / total_count
        significant_events = ratios[ratios > 0.01].index.tolist()
        other_ratio = ratios[ratios <= 0.01].sum()
        plt.figure(figsize=(8, 6))
        plt.pie(ratios[ratios > 0.01].values.tolist() + [other_ratio], labels=si
        plt.title('Distribution of ' + column_name + ' in the target dataset:')
        plt.show()

    def filter_data_by_significant_events(self, column_name):
        significant_events = self.dataframe[column_name].value_counts(normalize=
        significant_events_data = self.dataframe[self.dataframe[column_name].isi
        self.filtered = significant_events_data
        return significant_events_data


    def ind_event_sequence(self, person_id):
        df = self.dataframe
        df['event_date'] = pd.to_datetime(df['event_date'])
        df0 = df[df['person_id'] == person_id]
        df1 = df0.sort_values(by='event_date')
        df2 = df1[df1['event'] != df1['event'].shift(-1)]
        df2.reset_index(drop=True, inplace=True)
        return df2
```

```
    def plot_journey(self, person_id):
        df = self.dataframe
        df['event_date'] = pd.to_datetime(df['event_date'])
        person_data = df[df['person_id'] == person_id]
        person_data_sorted = person_data.sort_values(by='event_date')
        person_data_unique = person_data_sorted[person_data_sorted['event'] != p
        person_data_unique.reset_index(drop=True, inplace=True)
        G = nx.DiGraph()
        for _, row in person_data_unique.iterrows():
            G.add_node(row['event'])
        for i in range(len(person_data_unique) - 1):
            current_event = person_data_unique.iloc[i]['event']
            next_event = person_data_unique.iloc[i + 1]['event']
            G.add_edge(current_event, next_event)
        pos = nx.circular_layout(G)
        nx.draw(G, pos, with_labels=True, node_size=2000, node_color='skyblue',
                arrows=True, arrowsize=20)
        plt.title('Event Sequence for Person ID: ' + str(person_id))
        plt.show()

    def create_event_sequence(self):
        if not hasattr(self, 'filtered'):
            raise AttributeError("The 'filtered' data has not been generated. Pl

        event_sequence_df = pd.DataFrame(columns=['person_id', 'gender'])
        last_event_df = pd.DataFrame(columns=['person_id'])
        for person_id, group in self.filtered.groupby('person_id'):
            gender = group['gender'].iloc[0]
            events = group.sort_values(by='event_date')['event'].tolist()
            previous_event = None
            event_list = []
            for event in events:
                if event != previous_event:
                    event_list.append(event)
                previous_event = event
            event_sequence = {'person_id': person_id, 'gender': gender}
            sequence_count = 0
            for i in event_list[:-1]:
                sequence_count += 1
                event_sequence[f'condition_{sequence_count}'] = i
            event_sequence_df = pd.concat([event_sequence_df, pd.DataFrame(event
                                           ignore_index=True)
            last_event_df = pd.concat([last_event_df, pd.DataFrame({'person_id':
                                                                    'last_condit
                                       ignore_index=True)
        all_together = pd.merge(event_sequence_df, last_event_df, on='person_id'
        self.event_sequence = all_together
        return all_together

    def filter_long_journey(self, top_numbers=2000):
        if not hasattr(self, 'event_sequence'):
            raise AttributeError("The 'event_sequence' attribute has not been ge

        data = self.event_sequence
        data['null_count'] = data.isnull().sum(axis=1)
        sort = data.sort_values(by='null_count', ascending=True)
        sort.reset_index(drop=True, inplace=True)
        result = sort.iloc[:, :-1].head(top_numbers)
        self.long_journey = result
        return result
```

```
    def generate_pairs(self):
        if not hasattr(self, 'long_journey'):
            raise AttributeError("The 'long_journey' attribute has not been gene

        df = self.long_journey
        pairs = []
        for index, row in df.iterrows():
            person_id = row['person_id']
            gender = row['gender']
            last = row['last_condition']
            conditions = [col for col in row['condition_1':'last_condition'] if

            for pair in combinations(conditions, 2):
                if pair[0] != pair[1]:  # avoid having same conditions paired
                    pair_str = ' -> '.join(pair)
                    pair_str = pair_str.replace('[', '').replace(']', '')
                    pairs.append({'person_id': person_id, 'gender': gender, 'pai
                                  'last_condition': last})

        pairs_df = pd.DataFrame(pairs)
        self.pairs = pairs_df
        return pairs_df

    def pairs_to_bin(self):
        if not hasattr(self, 'pairs'):
            raise AttributeError("The 'pairs' attribute has not been generated.

        df = self.pairs
        pair_type = df['pair'].unique()
        pair_bin_df = pd.DataFrame(columns=['person_id', 'MALE', 'FEMALE'] + lis

        for person_id, group in df.groupby('person_id'):
            gender = group['gender'].iloc[0]
            last = group['last_condition'].iloc[0]
            pair_bin = {'person_id': person_id, 'last_condition': last}

            for i in ['MALE', 'FEMALE']:
                pair_bin[i] = 0
                if i == gender:
                    pair_bin[i] = 1

            for i in pair_type:
                pair_bin[i] = 0
                if i in group['pair'].values:
                    pair_bin[i] = 1

            pair_bin_df = pd.concat([pair_bin_df, pd.DataFrame(pair_bin, index=[

        return pair_bin_df

    def pairs_to_count(self):
        if not hasattr(self, 'pairs'):
            raise AttributeError("The 'pairs' attribute has not been generated.

        df = self.pairs
        pair_types = df['pair'].unique()
        pair_count_df = pd.DataFrame(columns=['person_id', 'MALE', 'FEMALE'] + l

        for person_id, group in df.groupby('person_id'):
```

```
            gender = group['gender'].iloc[0]
            last = group['last_condition'].iloc[0]

            pair_counts = group['pair'].value_counts().to_dict()

            pair_count = {'person_id': person_id, 'last_condition': last}

            for i in ['MALE', 'FEMALE']:
                pair_count[i] = 0
                if i == gender:
                    pair_count[i] = 1

            for pair_type in pair_types:
                pair_count[pair_type] = pair_counts.get(pair_type, 0)

            pair_count_df = pd.concat([pair_count_df, pd.DataFrame([pair_count])

        return pair_count_df

class PredictaVie_SplitData:
    def __init__(self, dataframe):
        self.dataframe = dataframe

    def c2c_split_data(self, test_size=0.2, random_state=42):
        df = self.dataframe
        exclude_column = ['person_id', 'last_condition']
        filtered_columns = [col for col in df.columns if col not in exclude_colu
        y = df['last_condition']
        x = df.loc[:, filtered_columns].astype(int)
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=test
        return x_train, x_test, y_train, y_test

class PredictaVie_Model:
    def __init__(self, x_train, y_train, x_test, y_test):
        self.x_train = x_train
        self.y_train = y_train
        self.x_test = x_test
        self.y_test = y_test

    def c2c_train_xgb(self, param_grid=None, cv=5):
        label_encoder = LabelEncoder()
        y_train_encoded = label_encoder.fit_transform(self.y_train)

        if param_grid is None:
            param_grid = {
                'n_estimators': [100, 150, 200],
                'max_depth': [3, 4, 5],
                'learning_rate': [0.1, 0.01, 0.001]
            }

        model = xgb.XGBClassifier()
        grid_search = GridSearchCV(model, param_grid, cv=cv, scoring='accuracy')
        grid_search.fit(self.x_train, y_train_encoded)

        best_model = grid_search.best_estimator_
        best_params = grid_search.best_params_
        best_score = grid_search.best_score_

        print("Best parameters:", best_params)
        print("Best cross-validation accuracy:", best_score)
```

```
        self.best_xgbmodel = best_model

        return best_model

    def c2c_evaluate_xgb(self):
        if not hasattr(self, 'best_xgbmodel'):
            raise AttributeError("The 'best_xgbmodel' has not been generated. Pl

        best = self.best_xgbmodel

        label_encoder = LabelEncoder()
        y_test_encoded = label_encoder.fit_transform(self.y_test)

        y_pred = best.predict(self.x_test)

        accuracy = accuracy_score(y_test_encoded, y_pred)
        print("Test Accuracy:", accuracy)

        original_labels = label_encoder.inverse_transform(y_test_encoded)
        predicted_labels = label_encoder.inverse_transform(y_pred)
        cm = confusion_matrix(original_labels, predicted_labels)

        cm_df = pd.DataFrame(cm, index=label_encoder.classes_, columns=label_enc

        plt.figure(figsize=(10, 8))
        sns.heatmap(cm_df, annot=True, cmap="YlGnBu", fmt="d")
        plt.xlabel('Predicted')
        plt.ylabel('Actual')
        plt.title('Confusion Matrix')
        plt.show()

    def c2c_xgb_feature_importance(self):
        if not hasattr(self, 'best_xgbmodel'):
            raise AttributeError("The 'best_xgbmodel' has not been generated. Pl

        xgb_model = self.best_xgbmodel
        feature_importance = xgb_model.feature_importances_
        feature_names = xgb_model.get_booster().feature_names
        feature_importance_dict = dict(zip(feature_names, feature_importance))
        sorted_feature_importance = sorted(feature_importance_dict.items(), key=

        features = [x[0] for x in sorted_feature_importance[:20]]
        importance = [x[1] for x in sorted_feature_importance[:20]]

        plt.figure(figsize=(10, 8))
        plt.barh(features, importance, color='skyblue')
        plt.xlabel('Feature Importance')
        plt.ylabel('Features')
        plt.title('XGBoost Feature Importance')
        plt.gca().invert_yaxis()
        plt.show()
        top_20_feature_indices = [item[0] for item in sorted_feature_importance[
        return top_20_feature_indices

    def c2c_train_logreg(self, param_grid=None):
        logistic_reg = LogisticRegression()

        if param_grid is None:
            param_grid = {
```

```
                'penalty': ['l1', 'l2', 'elasticnet', 'none'],
                'C': [0.001, 0.01, 0.1, 1, 10, 100],
                'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']
            }

        grid_search = GridSearchCV(estimator=logistic_reg, param_grid=param_grid
        grid_search.fit(self.x_train, self.y_train)
        print("Best Parameters:", grid_search.best_params_)

        best_model = grid_search.best_estimator_

        feature_names = list(self.x_train.columns)
        feature_weights = best_model.coef_[0]
        feature_weight_dict = dict(zip(feature_names, feature_weights))
        print("\nFeature Weights in the Best Model:")
        for feature, weight in feature_weight_dict.items():
            print(f"{feature}: {weight}")

        abs_feature_weights = np.abs(feature_weights)
        top_20_indices = np.argsort(abs_feature_weights)[::-1][:20]
        top_20_features = [feature_names[i] for i in top_20_indices]

        self.best_logregmodel = best_model

        return best_model, top_20_features

    def c2c_evaluate_logreg(self):
        if not hasattr(self, 'best_logregmodel'):
            raise AttributeError("The 'best_logregmodel' has not been generated.

        best = self.best_logregmodel

        y_pred = best.predict(self.x_test)

        accuracy = accuracy_score(self.y_test, y_pred)
        print("Test Accuracy:", accuracy)

        cm = confusion_matrix(self.y_test, y_pred)
        cm_df = pd.DataFrame(cm, index=best.classes_, columns=best.classes_)

        plt.figure(figsize=(10, 8))
        sns.heatmap(cm_df, annot=True, cmap="Blues", fmt='g')
        plt.xlabel('Predicted')
        plt.ylabel('Actual')
        plt.title('Confusion Matrix Heatmap')
        plt.show()
```

In [4]:

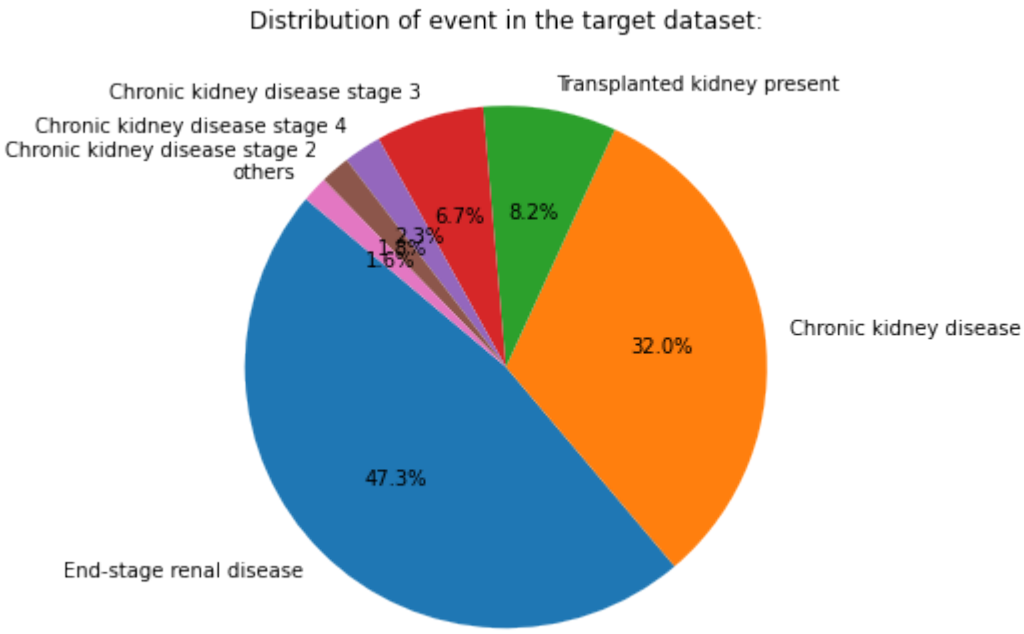
```
# load the kidney event data
from dbconns import ImpalaConnector
ic = ImpalaConnector('CHUNGAX6','Francais1418!', connection = "arch-prod-impala-
data_kidney = ic.read("SELECT * FROM app_rwd_capstone_group3.data_kidney ORDER B

data_kidney.head()
```

Out[4]:

	person_id	age_when_event	gender	event	event_date
0	6102	60	FEMALE	Chronic kidney disease	2003-02-03
1	6102	60	FEMALE	Chronic kidney disease	2003-02-01
2	11801	60	MALE	End-stage renal disease	2007-05-16
3	11801	61	MALE	End-stage renal disease	2008-08-20
4	11801	60	MALE	End-stage renal disease	2007-09-04

In [5]:
```
# load the preprocess package
preprocess = PredictaVie_Preprocess(data_kidney)
```

In [6]:
```
# show the distribution of each events in the dataset
preprocess.event_pie_chart('event')
```

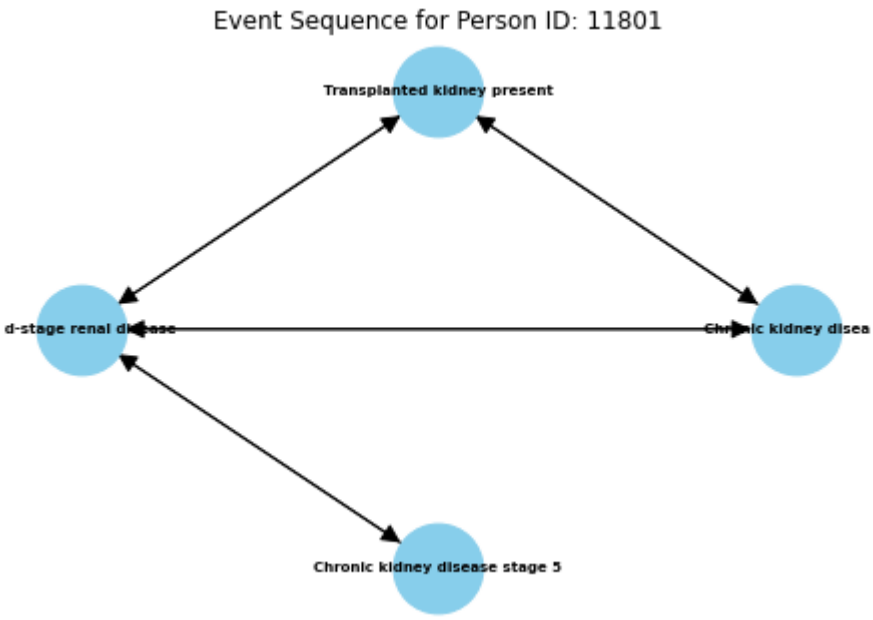


In [7]:
```
# take "patient 11801" as an example
preprocess.ind_event_sequence(11801)
```

Out[7]:

	person_id	age_when_event	gender	event	event_date
0	11801	53	MALE	Chronic kidney disease	2000-04-03
1	11801	54	MALE	Transplanted kidney present	2001-04-17
2	11801	55	MALE	Chronic kidney disease	2002-03-20
3	11801	55	MALE	Transplanted kidney present	2002-03-20
4	11801	55	MALE	Chronic kidney disease	2002-05-01
5	11801	55	MALE	Transplanted kidney present	2002-05-15
6	11801	58	MALE	Chronic kidney disease	2005-09-02
7	11801	60	MALE	End-stage renal disease	2007-02-07
8	11801	60	MALE	Transplanted kidney present	2007-02-16
9	11801	60	MALE	End-stage renal disease	2007-07-09
10	11801	60	MALE	Chronic kidney disease	2007-07-18
11	11801	61	MALE	End-stage renal disease	2008-03-28
12	11801	61	MALE	Chronic kidney disease stage 5	2008-03-31
13	11801	61	MALE	End-stage renal disease	2008-05-07
14	11801	61	MALE	Chronic kidney disease	2008-05-08
15	11801	61	MALE	End-stage renal disease	2008-05-16
16	11801	61	MALE	Chronic kidney disease	2008-05-19
17	11801	61	MALE	End-stage renal disease	2008-05-28
18	11801	61	MALE	Transplanted kidney present	2008-05-28
19	11801	61	MALE	End-stage renal disease	2008-10-20
20	11801	61	MALE	Chronic kidney disease	2008-10-21
21	11801	61	MALE	End-stage renal disease	2008-12-30

In [8]:
```
# pateint 11801's condition journey
preprocess.plot_journey(11801)
```



In [9]:
```
# filter those insignificant events for smoother model building
preprocess.filter_data_by_significant_events('event')
```

Out[9]:

	person_id	age_when_event	gender	event	event_date
0	6102	60	FEMALE	Chronic kidney disease	2003-02-03
1	6102	60	FEMALE	Chronic kidney disease	2003-02-01
2	11801	60	MALE	End-stage renal disease	2007-05-16
3	11801	61	MALE	End-stage renal disease	2008-08-20
4	11801	60	MALE	End-stage renal disease	2007-09-04
...	...	...	...	...	...
299995	98451202	52	FEMALE	End-stage renal disease	2014-03-10
299996	98451202	43	FEMALE	Chronic kidney disease	2005-05-02
299997	98451202	43	FEMALE	Chronic kidney disease	2005-03-07
299998	98451202	51	FEMALE	End-stage renal disease	2013-09-11
299999	98451202	44	FEMALE	End-stage renal disease	2006-07-24

295213 rows × 5 columns

In [10]:
```
# create the condition sequence
preprocess.create_event_sequence()
```

Out[10]:

	person_id	gender	condition_1	condition_2	condition_3	condition_4	condition
0	6102	FEMALE	NaN	NaN	NaN	NaN	N
1	11801	MALE	Chronic kidney disease	Transplanted kidney present	Chronic kidney disease	Transplanted kidney present	Chro kidr disea
2	25602	MALE	NaN	NaN	NaN	NaN	N
3	27602	FEMALE	NaN	NaN	NaN	NaN	N
4	30902	FEMALE	NaN	NaN	NaN	NaN	N
...	...	...	...	...	...	...	
8428	98441501	MALE	Chronic kidney disease stage 2	Chronic kidney disease stage 3	NaN	NaN	N
8429	98442801	FEMALE	NaN	NaN	NaN	NaN	N
8430	98446205	FEMALE	NaN	NaN	NaN	NaN	N
8431	98450502	FEMALE	Chronic kidney disease stage 3	Chronic kidney disease	NaN	NaN	N
8432	98451202	FEMALE	Chronic kidney disease	NaN	NaN	NaN	N

8433 rows × 195 columns

In [11]:
```
# take the top 2000 patients who have long condition journey (lots of event hist
preprocess.filter_long_journey()
```

Out[11]:

	person_id	gender	condition_1	condition_2	condition_3	condition_4	conditio
0	15785006	FEMALE	Chronic kidney disease stage 3	Chronic kidney disease stage 4	Chronic kidney disease stage 3	Chronic kidney disease stage 4	Chro kid dise stag
1	27543001	FEMALE	Chronic kidney disease stage 3	Chronic kidney disease stage 4	Chronic kidney disease	Chronic kidney disease stage 4	Chro kid dise stag
2	31193801	MALE	Chronic kidney disease	End-stage renal disease	Transplanted kidney present	End-stage renal disease	Transplan kid pres
3	27636101	MALE	Chronic kidney disease	Transplanted kidney present	Chronic kidney disease	Transplanted kidney present	Chro kid dise
4	14659201	FEMALE	Chronic kidney disease	End-stage renal disease	Chronic kidney disease	End-stage renal disease	Chro kid dise
...	...	...	...	...	...	...	
1995	14743102	FEMALE	Chronic kidney disease stage 2	Chronic kidney disease stage 3	NaN	NaN	N
1996	98418801	FEMALE	Chronic kidney disease	Chronic kidney disease stage 4	NaN	NaN	N
1997	98450502	FEMALE	Chronic kidney disease stage 3	Chronic kidney disease	NaN	NaN	N
1998	14710902	FEMALE	Chronic kidney disease stage 3	Chronic kidney disease stage 4	NaN	NaN	N
1999	611001	FEMALE	Chronic kidney disease	Chronic kidney disease stage 3	NaN	NaN	N

2000 rows × 195 columns

In [12]:
```
# create condition pairs on the filtered data (including duplicated pairs)
preprocess.generate_pairs()
```

Out[12]:

	person_id	gender	pair	last_condition
0	15785006	FEMALE	Chronic kidney disease stage 3 -> Chronic kidn...	End-stage renal disease
1	15785006	FEMALE	Chronic kidney disease stage 3 -> Chronic kidn...	End-stage renal disease
2	15785006	FEMALE	Chronic kidney disease stage 3 -> Chronic kidn...	End-stage renal disease
3	15785006	FEMALE	Chronic kidney disease stage 3 -> Chronic kidn...	End-stage renal disease
4	15785006	FEMALE	Chronic kidney disease stage 3 -> End-stage re...	End-stage renal disease
...	...	...	...	...
223508	14710902	FEMALE	Chronic kidney disease stage 3 -> Chronic kidn...	Chronic kidney disease stage 3
223509	14710902	FEMALE	Chronic kidney disease stage 4 -> Chronic kidn...	Chronic kidney disease stage 3
223510	611001	FEMALE	Chronic kidney disease -> Chronic kidney disea...	Chronic kidney disease stage 4
223511	611001	FEMALE	Chronic kidney disease -> Chronic kidney disea...	Chronic kidney disease stage 4
223512	611001	FEMALE	Chronic kidney disease stage 3 -> Chronic kidn...	Chronic kidney disease stage 4

223513 rows × 4 columns

In [13]:
```
# make the condition pairs into count
paircount = preprocess.pairs_to_count()
paircount
```

Out[13]:

	person_id	MALE	FEMALE	Chronic kidney disease stage 3 -> Chronic kidney disease stage 4	Chronic kidney disease stage 3 -> End-stage renal disease	Chronic kidney disease stage 3 -> Chronic kidney disease	Chronic kidney disease stage 3 -> Transplanted kidney present	Chronic kidney disease stage 3 -> Chronic kidney disease stage 2	Ch k di st Ch k di st
0	11801	1	0	0	0	0	0	0	
1	40601	1	0	0	0	0	0	0	
2	74102	0	1	0	0	0	0	0	
3	195401	0	1	0	0	0	0	1	
4	299801	1	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	
1995	98422902	1	0	0	0	0	0	0	
1996	98427601	1	0	1	1	0	0	0	
1997	98436901	1	0	0	0	0	0	0	
1998	98441501	1	0	0	0	0	0	1	
1999	98450502	0	1	0	0	1	0	0	

2000 rows × 34 columns

In [14]:
```
# make the condition pairs into binary
pairbin = preprocess.pairs_to_bin()
pairbin
```

Out[14]:

	person_id	MALE	FEMALE	Chronic kidney disease stage 3 -> Chronic kidney disease stage 4	Chronic kidney disease stage 3 -> End-stage renal disease	Chronic kidney disease stage 3 -> Chronic kidney disease	Chronic kidney disease stage 3 -> Transplanted kidney present	Chronic kidney disease stage 3 -> Chronic kidney disease stage 2	Ch k di st Ch k di st
0	11801	1	0	0	0	0	0	0	
1	40601	1	0	0	0	0	0	0	
2	74102	0	1	0	0	0	0	0	
3	195401	0	1	0	0	0	0	1	
4	299801	1	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	
1995	98422902	1	0	0	0	0	0	0	
1996	98427601	1	0	1	1	0	0	0	
1997	98436901	1	0	0	0	0	0	0	
1998	98441501	1	0	0	0	0	0	1	
1999	98450502	0	1	0	0	1	0	0	

2000 rows × 34 columns

In [15]:
```
# load the train_test_split package
paircountsplit = PredictaVie_SplitData(paircount)
pairbinsplit = PredictaVie_SplitData(pairbin)
```

In [16]:
```
# Split data into training and testing, respectively
x_train, x_test, y_train, y_test = paircountsplit.c2c_split_data() # for count a
X_train, X_test, Y_train, Y_test = pairbinsplit.c2c_split_data() # for binary da
```

In [17]:
```
# load the model package
countmodel = PredictaVie_Model(x_train, y_train, x_test, y_test)
```

```
binmodel = PredictaVie_Model(X_train, Y_train, X_test, Y_test)
```

In [18]:
```
# Use Gridsearch to build and find the best XGBoost model
binmodel.c2c_train_xgb()
```

```
Best parameters: {'learning_rate': 0.01, 'max_depth': 4, 'n_estimators': 200}
Best cross-validation accuracy: 0.6518750000000001
```

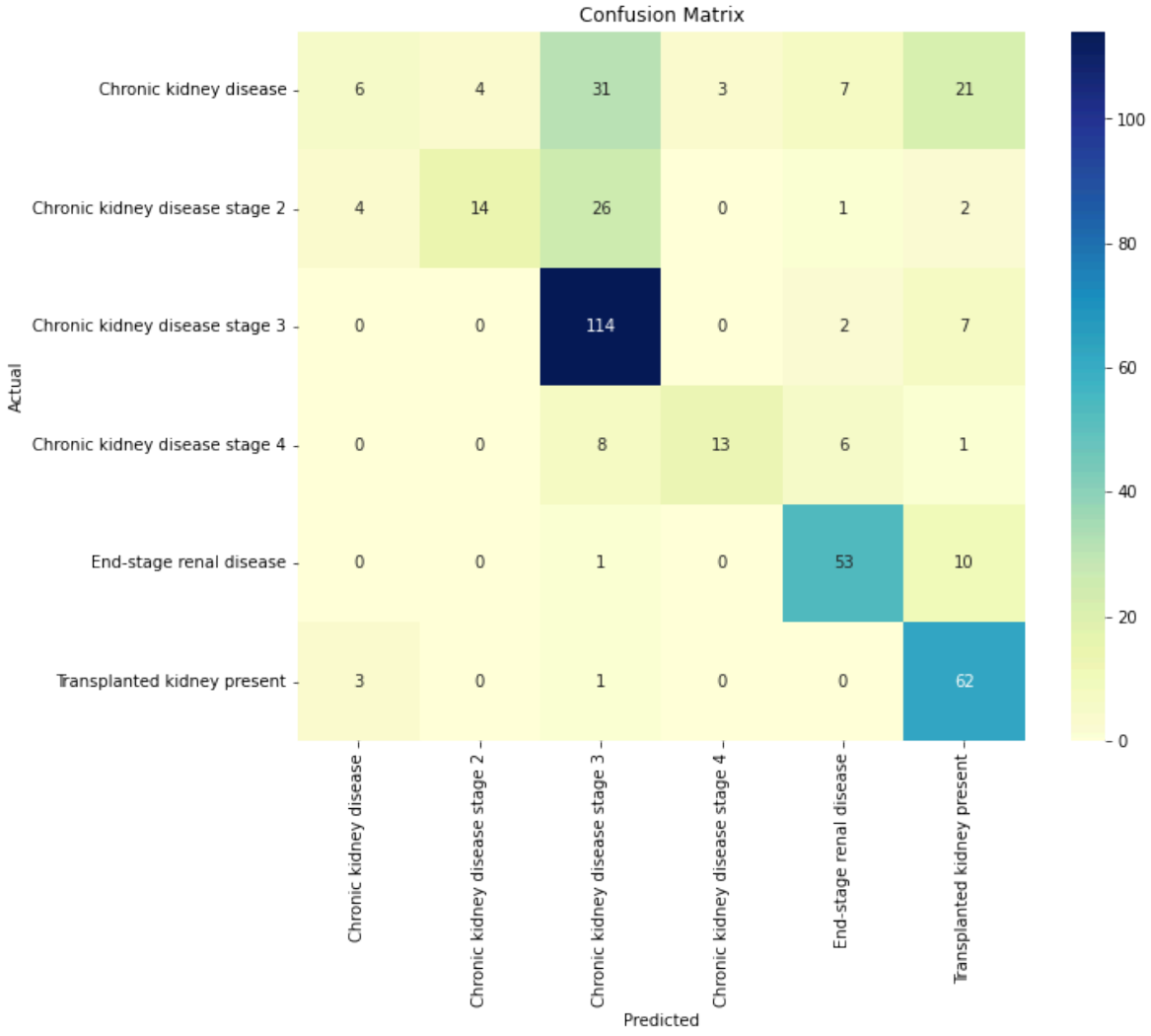
Out[18]:

XGBClassifier

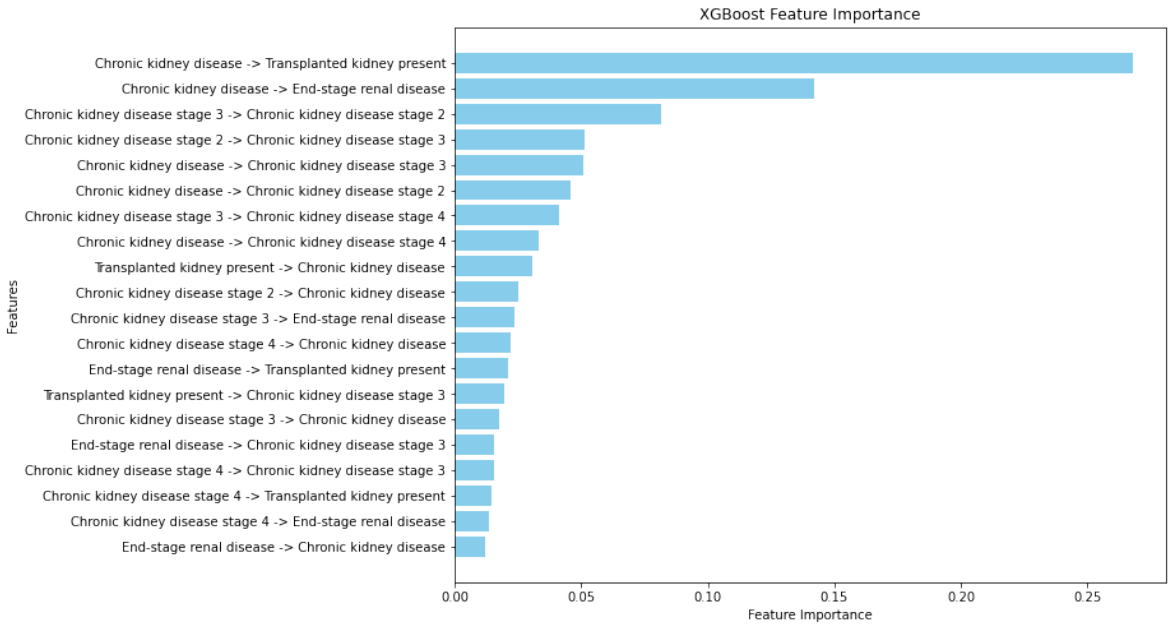
```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rou
nds=None,
              enable_categorical=False, eval_metric=None, feature_ty
pes=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.01, max_
bin=None,
```

In [19]:
```
# test the best XGBoost model to see its performance
binmodel.c2c_evaluate_xgb()

# not quite good since it only performs well in predicting CKD satge 3, ESRD, an
```

```
Test Accuracy: 0.655
```



In [20]:
```
# Check which condtion pairs are important for predictions
xgb_top_features = binmodel.c2c_xgb_feature_importance()
xgb_top_features # store the top 20 important features into a list
```



Out[20]:
```
['Chronic kidney disease -> Transplanted kidney present',
 'Chronic kidney disease -> End-stage renal disease',
 'Chronic kidney disease stage 3 -> Chronic kidney disease stage 2',
 'Chronic kidney disease stage 2 -> Chronic kidney disease stage 3',
 'Chronic kidney disease -> Chronic kidney disease stage 3',
 'Chronic kidney disease -> Chronic kidney disease stage 2',
 'Chronic kidney disease stage 3 -> Chronic kidney disease stage 4',
 'Chronic kidney disease -> Chronic kidney disease stage 4',
 'Transplanted kidney present -> Chronic kidney disease',
 'Chronic kidney disease stage 2 -> Chronic kidney disease',
 'Chronic kidney disease stage 3 -> End-stage renal disease',
 'Chronic kidney disease stage 4 -> Chronic kidney disease',
 'End-stage renal disease -> Transplanted kidney present',
 'Transplanted kidney present -> Chronic kidney disease stage 3',
 'Chronic kidney disease stage 3 -> Chronic kidney disease',
 'End-stage renal disease -> Chronic kidney disease stage 3',
 'Chronic kidney disease stage 4 -> Chronic kidney disease stage 3',
 'Chronic kidney disease stage 4 -> Transplanted kidney present',
 'Chronic kidney disease stage 4 -> End-stage renal disease',
 'End-stage renal disease -> Chronic kidney disease']
```

In [21]:
```
# ignore "convergence not found" warnings while the gridsearch is running
from sklearn.exceptions import ConvergenceWarning
import warnings
warnings.filterwarnings("ignore", category=ConvergenceWarning)


# Use Gridsearch to build and find the best Logistic Regression model
_, logreg_top_features = countmodel.c2c_train_logreg()
logreg_top_features # store the top 20 important features into a list
```

```
/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_validatio
n.py:547: FitFailedWarning:
390 fits failed out of a total of 600.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_sc
ore='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver newton-cg supports only 'l2' or None penalties, got l1 penalt
y.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver lbfgs supports only 'l2' or None penalties, got l1 penalty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver sag supports only 'l2' or None penalties, got l1 penalty.
```

```
--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver newton-cg supports only 'l2' or None penalties, got elasticnet
penalty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver lbfgs supports only 'l2' or None penalties, got elasticnet pen
alty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 75, in _check_solver
    raise ValueError(
ValueError: Only 'saga' solver supports elasticnet penalty, got solver=liblinear.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
```

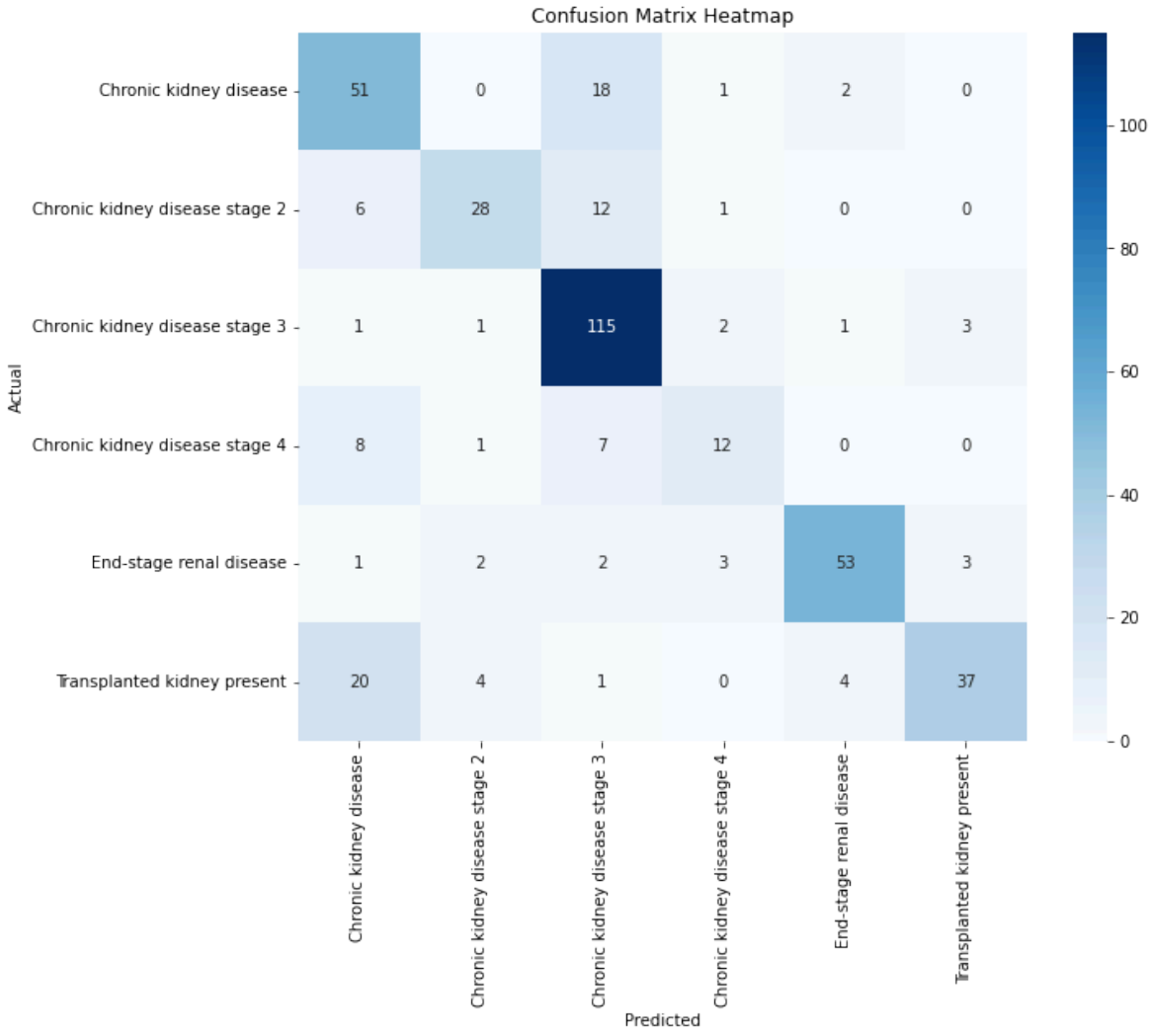
```
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver sag supports only 'l2' or None penalties, got elasticnet penal
ty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1182, in fit
    raise ValueError("l1_ratio must be specified when penalty is elasticnet.")
ValueError: l1_ratio must be specified when penalty is elasticnet.

--------------------------------------------------------------------------------
150 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 146
7, in wrapper
    estimator._validate_params()
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 666,
in _validate_params
    validate_parameter_constraints(
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/utils/_param_valida
tion.py", line 95, in validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The 'penalty' parameter of
LogisticRegression must be a str among {'l2', 'elasticnet', 'l1'} or None. Got 'n
one' instead.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_search.py:
1051: UserWarning: One or more of the test scores are non-finite: [     nan
nan 0.4325        nan 0.565625 0.535    0.52625  0.69375
 0.615625 0.599375      nan      nan      nan      nan      nan      nan
      nan      nan      nan      nan      nan      nan 0.498125      nan
 0.59625  0.685    0.694375 0.711875 0.615625 0.6           nan      nan
      nan      nan      nan      nan      nan      nan      nan      nan
      nan      nan 0.695         nan 0.599375 0.76     0.741875 0.733125
 0.615625 0.59875       nan      nan      nan      nan      nan      nan
      nan      nan      nan      nan      nan      nan 0.729375      nan
 0.599375 0.763125 0.745    0.731875 0.615625 0.6           nan      nan
      nan      nan      nan      nan      nan      nan      nan      nan
      nan      nan 0.734375      nan 0.6      0.758125 0.746875 0.73625
 0.61625  0.6           nan      nan      nan      nan      nan      nan
      nan      nan      nan      nan      nan      nan 0.730625      nan
```

```
 0.599375 0.75625  0.750625 0.715625 0.615625 0.6           nan      nan
      nan      nan      nan      nan      nan      nan      nan      nan]
  warnings.warn(
Best Parameters: {'C': 1, 'penalty': 'l2', 'solver': 'newton-cg'}

Feature Weights in the Best Model:
MALE: 0.05561987869072586
FEMALE: -0.05232769122140239
Chronic kidney disease stage 3 -> Chronic kidney disease stage 4: 0.1815698146994
6626
Chronic kidney disease stage 3 -> End-stage renal disease: -1.257395684904263
Chronic kidney disease stage 3 -> Chronic kidney disease: 1.0183694362394993
Chronic kidney disease stage 3 -> Transplanted kidney present: -0.693687173498612
2
Chronic kidney disease stage 3 -> Chronic kidney disease stage 2: -0.047328590064
39803
Chronic kidney disease stage 4 -> Chronic kidney disease stage 3: -0.197205216944
1083
Chronic kidney disease stage 4 -> End-stage renal disease: -1.0096325447988825
Chronic kidney disease stage 4 -> Chronic kidney disease: 1.253964702151482
Chronic kidney disease stage 4 -> Transplanted kidney present: 0.4855523099283287
Chronic kidney disease stage 4 -> Chronic kidney disease stage 2: -0.389739611501
3413
End-stage renal disease -> Chronic kidney disease: 0.8549535405547682
End-stage renal disease -> Transplanted kidney present: 0.436838944134676
End-stage renal disease -> Chronic kidney disease stage 2: 0.3769180161906987
Chronic kidney disease -> End-stage renal disease: -0.1599695078789491
Chronic kidney disease -> Transplanted kidney present: -0.17609400220484595
Chronic kidney disease -> Chronic kidney disease stage 2: -0.6138704047471433
Transplanted kidney present -> End-stage renal disease: -0.7139761231700354
Transplanted kidney present -> Chronic kidney disease stage 2: 0.6277815720560841
Transplanted kidney present -> Chronic kidney disease: 0.4819413548344996
Chronic kidney disease stage 2 -> Transplanted kidney present: -0.523708133100070
8
Chronic kidney disease stage 2 -> End-stage renal disease: -0.4372329327093382
Chronic kidney disease stage 2 -> Chronic kidney disease: 1.4352611854465387
Chronic kidney disease -> Chronic kidney disease stage 4: -0.5252959415166177
Chronic kidney disease -> Chronic kidney disease stage 3: -0.3794252409852231
End-stage renal disease -> Chronic kidney disease stage 3: 0.2362938438181207
Transplanted kidney present -> Chronic kidney disease stage 3: 0.7940320299680954
Chronic kidney disease stage 2 -> Chronic kidney disease stage 3: -0.548954588922
0316
End-stage renal disease -> Chronic kidney disease stage 4: 0.005425225762460063
Transplanted kidney present -> Chronic kidney disease stage 4: -0.039266682304975
63
Chronic kidney disease stage 2 -> Chronic kidney disease stage 4: -0.142566170472
9185
```

Out[21]:
```
['Chronic kidney disease stage 2 -> Chronic kidney disease',
 'Chronic kidney disease stage 3 -> End-stage renal disease',
 'Chronic kidney disease stage 4 -> Chronic kidney disease',
 'Chronic kidney disease stage 3 -> Chronic kidney disease',
 'Chronic kidney disease stage 4 -> End-stage renal disease',
 'End-stage renal disease -> Chronic kidney disease',
 'Transplanted kidney present -> Chronic kidney disease stage 3',
 'Transplanted kidney present -> End-stage renal disease',
 'Chronic kidney disease stage 3 -> Transplanted kidney present',
 'Transplanted kidney present -> Chronic kidney disease stage 2',
 'Chronic kidney disease -> Chronic kidney disease stage 2',
 'Chronic kidney disease stage 2 -> Chronic kidney disease stage 3',
 'Chronic kidney disease -> Chronic kidney disease stage 4',
 'Chronic kidney disease stage 2 -> Transplanted kidney present',
 'Chronic kidney disease stage 4 -> Transplanted kidney present',
 'Transplanted kidney present -> Chronic kidney disease',
 'Chronic kidney disease stage 2 -> End-stage renal disease',
 'End-stage renal disease -> Transplanted kidney present',
 'Chronic kidney disease stage 4 -> Chronic kidney disease stage 2',
 'Chronic kidney disease -> Chronic kidney disease stage 3']
```

In [22]:
```
# test the best Logistic Regression model to see its performance
countmodel.c2c_evaluate_logreg()

# better than XGBoost!
# performs well in predicting:
# CKD, CKD stage 2 (though some were predicted as CKD satge 3, predicting worse
```

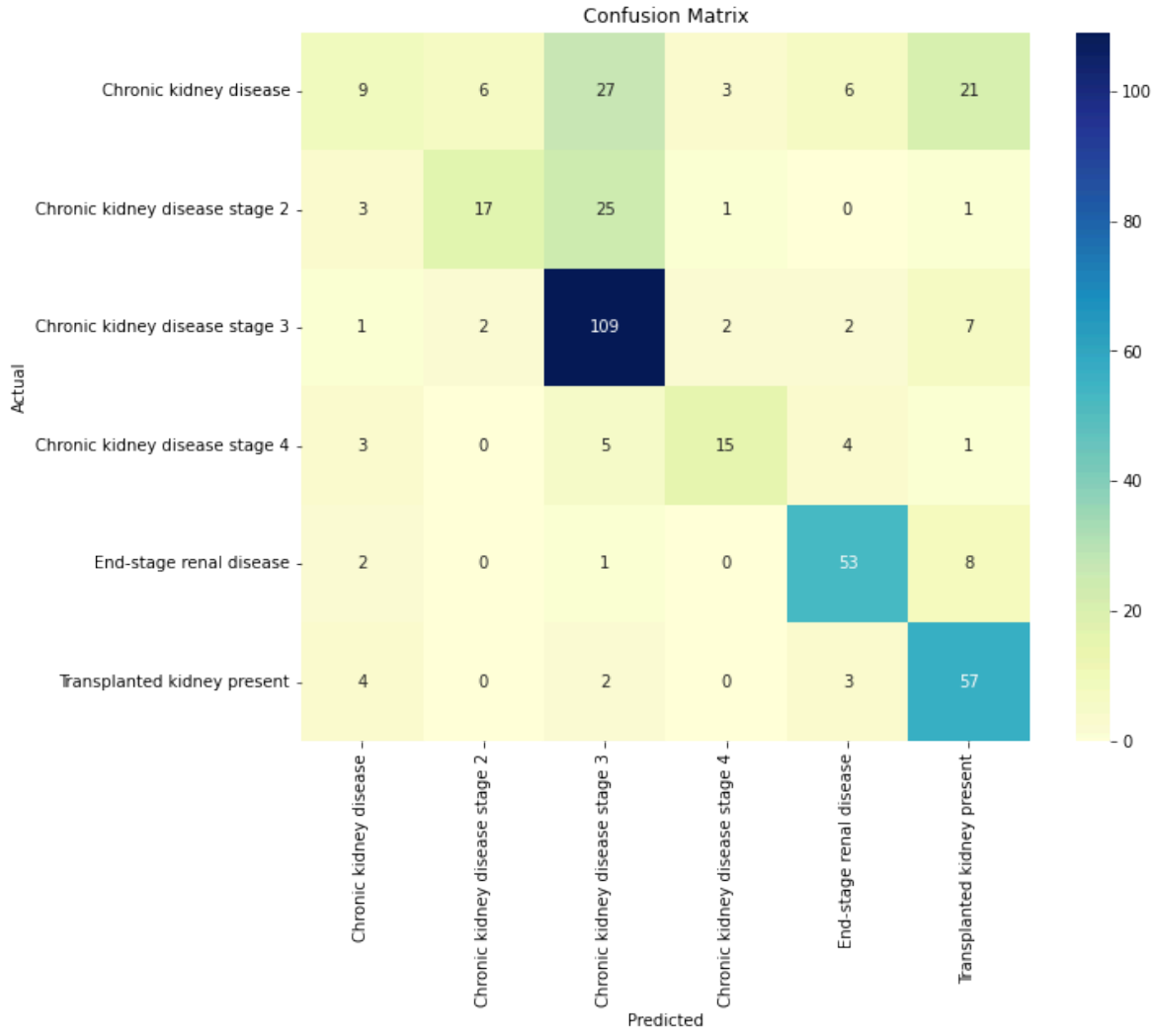
```
Test Accuracy: 0.74
```



In [23]:
```
# try using less columns (focus on the top condition pairs)
paircountsplit = PredictaVie_SplitData(paircount[['person_id', 'MALE', 'FEMALE',
pairbinsplit = PredictaVie_SplitData(pairbin[['person_id', 'MALE', 'FEMALE', 'la

# Split data
x_train, x_test, y_train, y_test = paircountsplit.c2c_split_data() # for count a
X_train, X_test, Y_train, Y_test = pairbinsplit.c2c_split_data() # for binary da

# load the model package
countmodel = PredictaVie_Model(x_train, y_train, x_test, y_test)
binmodel = PredictaVie_Model(X_train, Y_train, X_test, Y_test)
```

In [24]:
```
# Use Gridsearch to build and find the best XGBoost model
binmodel.c2c_train_xgb()

# test the best XGBoost model to see its performance
binmodel.c2c_evaluate_xgb()

# the performance seems to be a bit worse... (using the top features)
```

```
Best parameters: {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 150}
Best cross-validation accuracy: 0.653125
Test Accuracy: 0.65
```



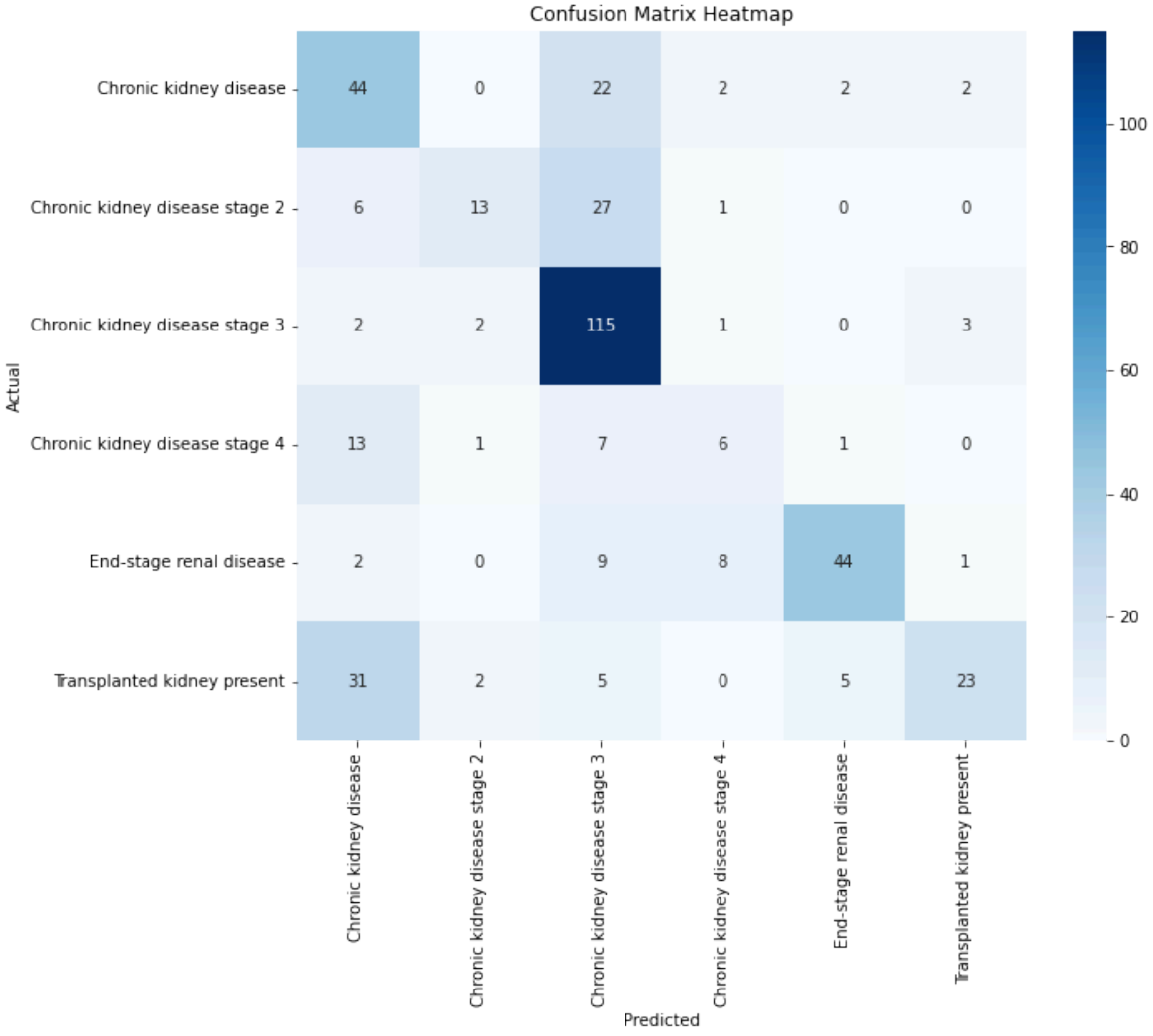
In [25]:
```
# Use Gridsearch to build and find the best Logistic Regression model
countmodel.c2c_train_logreg()

# test the best Logistic Regression model to see its performance
countmodel.c2c_evaluate_logreg()

# Focusing on the top features makes the Logistic Regression Terrible! Don't do
```

```
/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_validatio
n.py:547: FitFailedWarning:
390 fits failed out of a total of 600.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_sc
ore='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver newton-cg supports only 'l2' or None penalties, got l1 penalt
y.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver lbfgs supports only 'l2' or None penalties, got l1 penalty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver sag supports only 'l2' or None penalties, got l1 penalty.
```

```
--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver newton-cg supports only 'l2' or None penalties, got elasticnet
penalty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver lbfgs supports only 'l2' or None penalties, got elasticnet pen
alty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 75, in _check_solver
    raise ValueError(
ValueError: Only 'saga' solver supports elasticnet penalty, got solver=liblinear.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
```

```
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1172, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 67, in _check_solver
    raise ValueError(
ValueError: Solver sag supports only 'l2' or None penalties, got elasticnet penal
ty.

--------------------------------------------------------------------------------
30 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 147
4, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/linear_model/_logis
tic.py", line 1182, in fit
    raise ValueError("l1_ratio must be specified when penalty is elasticnet.")
ValueError: l1_ratio must be specified when penalty is elasticnet.

--------------------------------------------------------------------------------
150 fits failed with the following error:
Traceback (most recent call last):
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_va
lidation.py", line 895, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 146
7, in wrapper
    estimator._validate_params()
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/base.py", line 666,
in _validate_params
    validate_parameter_constraints(
  File "/home/cdsw/.local/lib/python3.9/site-packages/sklearn/utils/_param_valida
tion.py", line 95, in validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The 'penalty' parameter of
LogisticRegression must be a str among {'l2', 'elasticnet', 'l1'} or None. Got 'n
one' instead.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
/home/cdsw/.local/lib/python3.9/site-packages/sklearn/model_selection/_search.py:
1051: UserWarning: One or more of the test scores are non-finite: [     nan
nan 0.5225        nan 0.5575   0.4925   0.481875 0.608125
 0.58625  0.579375      nan      nan      nan      nan      nan      nan
      nan      nan      nan      nan      nan      nan 0.47375       nan
 0.578125 0.5675   0.575625 0.59125  0.58625  0.58          nan      nan
      nan      nan      nan      nan      nan      nan      nan      nan
      nan      nan 0.576875      nan 0.579375 0.60875  0.603125 0.595625
 0.585625 0.579375      nan      nan      nan      nan      nan      nan
      nan      nan      nan      nan      nan      nan 0.594375      nan
 0.57875  0.615625 0.608125 0.59875  0.585625 0.579375      nan      nan
      nan      nan      nan      nan      nan      nan      nan      nan
      nan      nan 0.6025        nan 0.579375 0.611875 0.608125 0.6025
 0.585625 0.579375      nan      nan      nan      nan      nan      nan
      nan      nan      nan      nan      nan      nan 0.60375       nan
```

```
 0.579375 0.61125  0.605    0.60375  0.58625  0.57875       nan      nan
      nan      nan      nan      nan      nan      nan      nan      nan]
  warnings.warn(
Best Parameters: {'C': 1, 'penalty': 'l2', 'solver': 'newton-cg'}

Feature Weights in the Best Model:
MALE: 0.02489133548427596
FEMALE: -0.023884918115527336
Chronic kidney disease stage 2 -> Chronic kidney disease: 1.2174107779304995
Chronic kidney disease stage 3 -> End-stage renal disease: -0.4412419577388694
Chronic kidney disease stage 4 -> Chronic kidney disease: 0.19928704585769222
Chronic kidney disease stage 3 -> Chronic kidney disease: 0.4445407581246135
Chronic kidney disease stage 4 -> End-stage renal disease: 0.11063335923808655
End-stage renal disease -> Chronic kidney disease: 0.1339702982783418
Transplanted kidney present -> Chronic kidney disease stage 3: 0.4569209721274815
Transplanted kidney present -> End-stage renal disease: -0.5934048929710831
Chronic kidney disease stage 3 -> Transplanted kidney present: -0.372968253013533
53
Transplanted kidney present -> Chronic kidney disease stage 2: 0.1471004049524484
4
Chronic kidney disease -> Chronic kidney disease stage 2: -0.43228146087911007
Chronic kidney disease stage 2 -> Chronic kidney disease stage 3: -0.756580845242
7949
Chronic kidney disease -> Chronic kidney disease stage 4: -0.0013632571361805548
Chronic kidney disease stage 2 -> Transplanted kidney present: 0.0489989010648524
9
Chronic kidney disease stage 4 -> Transplanted kidney present: -0.003312741177041
402
Transplanted kidney present -> Chronic kidney disease: 0.25612320134114924
Chronic kidney disease stage 2 -> End-stage renal disease: -0.6158537109232047
End-stage renal disease -> Transplanted kidney present: 0.25290493877933073
Chronic kidney disease stage 4 -> Chronic kidney disease stage 2: 0.0228278704245
44292
Chronic kidney disease -> Chronic kidney disease stage 3: -0.25969337037412243
Test Accuracy: 0.6125
```


Confusion Matrix Heatmap
Chronic kidney disease
Chronic kidney disease stage 2
Chronic kidney disease stage 3
Chronic kidney disease stage 4
End-stage renal disease
Transplanted kidney present
Actual
44 0 22 2 2 2
6 13 27 1 0 0
2 2 115 1 0 3
13 1 7 6 1 0
2 0 9 8 44 1
31 2 5 0 5 23
Chronic kidney disease
Chronic kidney disease stage 2
Chronic kidney disease stage 3
Chronic kidney disease stage 4
End-stage renal disease
Transplanted kidney present
Predicted
100
80
60
40
20
0