Online Supplement

Original Research
Identification of Severe Acute Exacerbations of Chronic Obstructive Pulmonary Disease Subgroups by Machine Learning Implementation in Electronic Health Records

Huan Li[1] John Huston[1] Jana Zielonka[1] Shannon Kay[1] Maor Sauler[1] Jose Gomez[1,2]

[1]Pulmonary, Critical Care and Sleep Medicine Section, Yale University, New Haven, Connecticut, United States

[2]Center for Precision Pulmonary Medicine (P2MED), Yale University, New Haven, Connecticut, United States

Supplementary Methods

Original Cohort Data Source and Study Population

YNHH is a tertiary-care hospital with 1541 beds and two campuses in New Haven, Connecticut, USA. We included all participants who met the following criteria during the study period: This study was limited to hospital admissions from patients 18 years and older. The International Classification of Diseases, Tenth Revision, Clinical Modification (ICD-10-CM) codes ICD-10-CM code J44.1 and also patients with any combination of acute bronchitis (J20.90) with chronic obstructive pulmonary disease, unspecified (J44.9), unspecified chronic bronchitis (J42), and emphysema, unspecified (J43.9). Patients with dual ICD-10-CM diagnosis for asthma and COPD at the time of the index or subsequent hospitalizations were also excluded from these analyses. Data from the first hospitalization that met inclusion and exclusion criteria during the study period were used for this analysis. Following patient identification, demographic data, including self-reported race and self-reported Hispanic ethnicity, comorbidities, first measured laboratory data and inpatient medication records were collected from the index hospitalization. Comorbidity data were extracted from the EHR's past medical history section. Laboratory data included complete blood count and blood chemistries.

COVID Cohort

The Yale Department of Medicine COVID-19 Explorer and Repository tool (DOM-CovX) was used to extract data on patients admitted with COVID-19 from March 1, 2020 to April 1, 2021 in Yale-New Haven Health System hospitals (https://spinup-0011f4.spinup.yale.edu/domcovx/ (2021), Accessed 27th Jun 2021). These hospitals included YNHH, Greenwich Hospital (Greenwich, CT), Bridgeport Hospital (Bridgeport, CT), Lawrence & Memorial Hospital (New London, CT), and Westerly Hospital (Westerly, RI). The patients had a positive test for SARS-CoV-2 using reverse transcriptase–polymerase chain reaction assays performed on nasopharyngeal swab specimens. The samples were collected within 14 days after admission. Following patient identification, demographics, past medical history, self-reported race and ethnicity, smoking history (in pack years), admission vital signs, laboratory tests, medications administered during hospitalization, and discharge status (to assess in-hospital mortality), were retrieved

Clustering

The 51 training features included: Age, sex, body mass index (BMI), systolic blood pressure, diastolic blood pressure, mean arterial pressure, smoking status (former, current, never), insurance, coronary artery disease (CAD), cerebrovascular disease, diabetes mellitus (DM), chronic kidney disease (CKD), heart failure (HF), allergic rhinitis, nasal polyposis, gastroesophageal reflux (GERD), lung cancer, sleep apnea, and hypertension (HTN). Inpatient administration of albuterol, inhaled corticosteroids (ICS) with long-acting beta-agonist (LABA), long-acting muscarinic antagonist (LAMA), antibiotic, ipratropium, systemic steroid, ICS-only, leukotriene receptor antagonist (LTRA), theophylline, influenza vaccine, oseltamivir, pneumococcal vaccine, and nicotine supplementation. White blood cell (WBC) count, neutrophil percentage, eosinophil percentage, basophil percentage, monocyte percentage, lymphocyte percentage, absolute neutrophil count, absolute eosinophil count, absolute basophil count, absolute monocyte count, absolute lymphocyte count, red blood cell count (RBCC), hematocrit, hemoglobin, mean corpuscular hemoglobin (MHC), mean corpuscular hemoglobin concentration (MCHC), mean corpuscular volume (MCV), platelets, and mean platelet volume (MPV).

The next step involved categorizing the variables into numerical and string types.

Statistical Analysis

The R statistical software was used for statistical analyses. The packages dplyr v.0.8.5, lubridate v.1.7.9, survival v.3.5-3, survminer v.0.4.9, ggfortify v.0.4.12, ggplot2 v.3.3.2, ggpubr v.0.4.0 were used in this study. The python packages used in this study include Pandas v. 1.2.4, Numpy v. 1.20.2, Sklearn v. 0.24.2, Pytorch v. 1.9.0+cu102,

torch_lr_finder v. 0.2.1, statsmodels v. 0.12.2, Matplotlib v. 3.4.1, pROC v 1.17.0.1, rpy2 v 3.4.5. The models were run on Yale's Milgram high performance computing cluster. Descriptive statistics used the Kruskal-Wallis and Wilcoxon Rank sum tests for continuous values, chi-square for categorical values, two-proportions Z-test for proportions between groups. Statistical analyses were performed using R [1], version 3.6.3 and Python, version 3.9.13. Univariate and multivariate survival analyses were performed with the R survival package v.3.5-3.

References

1. Ripley BD. The R project in statistical computing. *MSOR Connections. The newsletter of the LTSN Maths* [Internet] Citeseer; 2001; Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.430.3979&rep=rep1&type=pdf.

Classifier Code

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
import xgboost as xgb
from sklearn.model_selection import GridSearchCV, cross_val_score, train_test_split
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix,
f1_score, roc_auc_score
import shap

#import joblib
+*In[ ]:*+
[source, ipython3]
----
import seaborn as sns
----


+*In[ ]:*+
[source, ipython3]
----
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
from sklearn.model_selection import GridSearchCV, cross_val_score, train_test_split
from sklearn.calibration import calibration_curve
from sklearn.metrics import precision_recall_curve
from sklearn.model_selection import cross_val_score
from hyperopt import STATUS_OK
from hyperopt import hp
----

For this code, we are trying to build up a classifier based on the
previous unsupervised learning clustering outcome. We try to apply this
classifier to similar set of patient's data but with different criteria
```

for admission (covid).

1. Cross-validation for 5-fold to find the best number of rounds for an 80-20 split each time, and then select the best numbers and train the entire dataset.

+*In[2]:*+
[source, ipython3]

# loading the cluster_data
df = pd.read_csv('cluster_res.csv')#Original dataset used to identify the subgroups
df = df.drop(columns=['Unnamed: 0'])

+*In[3]:*+
[source, ipython3]
----
df.columns

+*Out[3]:*+
----Index(['Age', 'Sex', 'BMI', 'BP_Systolic', 'BP_Diastolic', 'MAP',
'Smoking.Status.Simple', 'Insurance.Simple', 'CAD', 'Cereb_VD', 'DM',
'CKD', 'CHF', 'Allergic.Rhinitis', 'GERD', 'Nasal.Polyposis',
'Lung.Cancer', 'Sleep.Apnea', 'HTN', 'ALBUTEROL', 'ICS_LABA', 'LAMA',
'ANTIBIOTIC', 'IPRATROPIUM', 'SYSTEMIC_STEROID', 'FLU_VACCINE', 'ICS',
'LTRA', 'THEOPHYLLINE', 'PNEUMOCOCCAL_VACCINE', 'NICOTINE',
'OSELTAMIVIR', 'WBC', 'Neutrophils', 'Eosinophils', 'Basophils',
'Monocytes', 'Lymphocytes', 'Abs.Neu.Count', 'Abs.Eo.Count',
'Abs.Baso.Count', 'Abs.Mono.Count', 'Abs.Lymph.Count', 'RBCC',
'Hematocrit', 'Hemoglobin', 'MCH', 'MCHC', 'MCV', 'MPV', 'Platelets',
'cluster', 'Study.ID'],
dtype='object')----

= Method 1

= Data-processing

+*In[4]:*+
[source, ipython3]
----
check the data condition to see if there is any missing data and types
XGB won't able to processing objective
so need to convert to dummy
df.dtypes

+*Out[4]:*+

----Age	int64
Sex	object
BMI	float64
BP_Systolic	int64
BP_Diastolic	int64
MAP	float64
Smoking.Status.Simple	object
Insurance.Simple	object
CAD	object
Cereb_VD	object
DM	object
CKD	object
CHF	object
Allergic.Rhinitis	object
GERD	object
Nasal.Polyposis	object
Lung.Cancer	object
Sleep.Apnea	object
HTN	object
ALBUTEROL	object
ICS_LABA	object
LAMA	object
ANTIBIOTIC	object
IPRATROPIUM	object
SYSTEMIC_STEROID	object
FLU_VACCINE	object
ICS	object
LTRA	object
THEOPHYLLINE	object
PNEUMOCOCCAL_VACCINE	object
NICOTINE	object
OSELTAMIVIR	object
WBC	float64
Neutrophils	float64
Eosinophils	float64
Basophils	float64
Monocytes	float64
Lymphocytes	float64
Abs.Neu.Count	float64
Abs.Eo.Count	float64
Abs.Baso.Count	float64
Abs.Mono.Count	float64
Abs.Lymph.Count	float64
RBCC	float64
Hematocrit	float64
Hemoglobin	float64
MCH	float64
MCHC	float64
MCV	float64

MPV float64
Platelets int64
cluster int64
Study.ID object
dtype: object----

+*In[5]:*+
[source, ipython3]
----
df.isnull().sum()

+*Out[5]:*+
----Age 0
Sex 0
BMI 0
BP_Systolic 0
BP_Diastolic 0
MAP 0
Smoking.Status.Simple 0
Insurance.Simple 0
CAD 0
Cereb_VD 0
DM 0
CKD 0
CHF 0
Allergic.Rhinitis 0
GERD 0
Nasal.Polyposis 0
Lung.Cancer 0
Sleep.Apnea 0
HTN 0
ALBUTEROL 0
ICS_LABA 0
LAMA 0
ANTIBIOTIC 0
IPRATROPIUM 0
SYSTEMIC_STEROID 0
FLU_VACCINE 0
ICS 0
LTRA 0
THEOPHYLLINE 0
PNEUMOCOCCAL_VACCINE 0
NICOTINE 0
OSELTAMIVIR 0
WBC 0
Neutrophils 0
Eosinophils 0
Basophils 0

```
Monocytes               0
Lymphocytes             0
Abs.Neu.Count           0
Abs.Eo.Count            0
Abs.Baso.Count          0
Abs.Mono.Count          0
Abs.Lymph.Count         0
RBCC                    0
Hematocrit              0
Hemoglobin              0
MCH                     0
MCHC                    0
MCV                     0
MPV                     0
Platelets               0
cluster                 0
Study.ID                0
dtype: int64----
```

+*In[ ]:*+
[source, ipython3]

```
#covid_data['Smoking.Status.Simple'] = np.nan
#covid_data['Insurance.Simple'] = np.nan
#covid_data['Allergic.Rhinitis'] = np.nan
#covid_data['Nasal.Polyposis'] = np.nan
#covid_data['Sleep.Apnea'] = np.nan
#covid_data['ICS_LABA'] = np.nan
#covid_data['FLU_VACCINE'] = np.nan
#covid_data['PNEUMOCOCCAL_VACCINE'] = np.nan
```

+*In[5]:*+
[source, ipython3]

```
col_to_drop = ['Lung.Cancer', 'FLU_VACCINE', 'Allergic.Rhinitis','ALBUTEROL', 'LTRA',
'IPRATROPIUM','Nasal.Polyposis',
            'THEOPHYLLINE', 'OSELTAMIVIR',
'cluster','Study.ID','Smoking.Status.Simple','Insurance.Simple',
            'Allergic.Rhinitis', 'Nasal.Polyposis', 'Sleep.Apnea','ICS_LABA','FLU_VACCINE',
'PNEUMOCOCCAL_VACCINE']
```

+*In[6]:*+
[source, ipython3]

```
# Identify the target column and separate it from the features
X = df.drop(col_to_drop, axis=1)
```

```
y = df['cluster']
----


+*In[7]:*+
[source, ipython3]
----
# Convert the string varaibles to one hot encoding
categorical_columns = list(X.select_dtypes(include=['object']).columns)
label_encoders = {}
for col in categorical_columns:
    label_encoders[col] = LabelEncoder()
    X[col] = label_encoders[col].fit_transform(X[col])
----


+*In[26]:*+
[source, ipython3]
----
X.columns
----


+*Out[26]:*+
----Index(['Age', 'Sex', 'BMI', 'BP_Systolic', 'BP_Diastolic', 'MAP', 'CAD',
       'Cereb_VD', 'DM', 'CKD', 'CHF', 'GERD', 'HTN', 'LAMA', 'ANTIBIOTIC',
       'SYSTEMIC_STEROID', 'ICS', 'NICOTINE', 'WBC', 'Neutrophils',
       'Eosinophils', 'Basophils', 'Monocytes', 'Lymphocytes', 'Abs.Neu.Count',
       'Abs.Eo.Count', 'Abs.Baso.Count', 'Abs.Mono.Count', 'Abs.Lymph.Count',
       'RBCC', 'Hematocrit', 'Hemoglobin', 'MCH', 'MCHC', 'MCV', 'MPV',
       'Platelets'],
      dtype='object')----


+*In[10]:*+
[source, ipython3]
----
X.isnull().sum()
----


+*Out[10]:*+
----Age                0
Sex                0
BMI                0
BP_Systolic        0
BP_Diastolic       0
MAP                0
CAD                0
Cereb_VD           0
DM                 0
```

```
CKD                 0
CHF                 0
GERD                0
HTN                 0
LAMA                0
ANTIBIOTIC          0
SYSTEMIC_STEROID    0
ICS                 0
NICOTINE            0
WBC                 0
Neutrophils         0
Eosinophils         0
Basophils           0
Monocytes           0
Lymphocytes         0
Abs.Neu.Count       0
Abs.Eo.Count        0
Abs.Baso.Count      0
Abs.Mono.Count      0
Abs.Lymph.Count     0
RBCC                0
Hematocrit          0
Hemoglobin          0
MCH                 0
MCHC                0
MCV                 0
MPV                 0
Platelets           0
dtype: int64----
```

+*In[11]:*+

```
[source, ipython3]
----
# Split training and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
----
```

+*In[12]:*+

```
[source, ipython3]
----
# Define a grid of hyperparameters to search
param_grid = {
    'learning_rate': [0.01, 0.1, 0.2],
    'n_estimators': [100, 200, 300],
    'max_depth': [3, 4, 5],
    'min_child_weight': [1, 2, 3],
    'gamma': [0, 0.1, 0.2],
    'subsample': [0.8, 0.9, 1.0],
    'colsample_bytree': [0.8, 0.9, 1.0],
```

}

+*In[13]:*+
[source, ipython3]
----
xgb_model = xgb.XGBClassifier(objective='multi:softprob')

+*In[14]:*+
[source, ipython3]

# Perform Grid Search with cross-validation
grid_search = GridSearchCV(estimator=xgb_model, param_grid=param_grid,
scoring='precision', cv=5, n_jobs=-1)
grid_search.fit(X_train, y_train)

+*Out[14]:*+
```
GridSearchCV(cv=5,
        estimator=XGBClassifier(base_score=None, booster=None,
                        callbacks=None, colsample_bylevel=None,
                        colsample_bynode=None,
                        colsample_bytree=None, device=None,
                        early_stopping_rounds=None,
                        enable_categorical=False, eval_metric=None,
                        feature_types=None, gamma=None,
                        grow_policy=None, importance_type=None,
                        interaction_constraints=None,
                        learning_rate=None,...
                        missing=nan, monotone_constraints=None,
                        multi_strategy=None, n_estimators=None,
                        n_jobs=None, num_parallel_tree=None,
                        objective='multi:softprob', ...),
        n_jobs=-1,
        param_grid={'colsample_bytree': [0.8, 0.9, 1.0],
                'gamma': [0, 0.1, 0.2],
                'learning_rate': [0.01, 0.1, 0.2],
                'max_depth': [3, 4, 5], 'min_child_weight': [1, 2, 3],
                'n_estimators': [100, 200, 300],
                'subsample': [0.8, 0.9, 1.0]},
        scoring='precision')----
```

+*In[15]:*+
[source, ipython3]

Get the best hyperparameters

```
best_params = grid_search.best_params_
print(f"Best Hyperparameters: {best_params}")
```

+*Out[15]:*+
```
Best Hyperparameters: {'colsample_bytree': 0.8, 'gamma': 0, 'learning_rate': 0.01, 'max_depth':
3, 'min_child_weight': 1, 'n_estimators': 100, 'subsample': 0.8}
```

+*In[27]:*+
[source, ipython3]
```
# Train the final model with the best hyperparameters using the full training dataset
best_xgb_model = xgb.XGBClassifier(**best_params, objective='multi:softprob')
best_xgb_model.fit(X_train, y_train)
```

+*Out[27]:*+
```
----XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=0.8, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=0, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.01, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=3, max_leaves=None,
              min_child_weight=1, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=100, n_jobs=None,
              num_parallel_tree=None, objective='multi:softprob', ...)----
```

+*In[28]:*+
[source, ipython3]
```
#cross-validation
cv_scores = cross_val_score(best_xgb_model, X, y, cv=5)
```

+*In[29]:*+
[source, ipython3]
```
print(f'Cross-validation scores: {cv_scores}')
print(f'Mean CV score: {np.mean(cv_scores)}')
```

+*Out[29]:*+

Cross-validation scores: [0.83333333 0.79827089 0.78097983 0.81268012 0.76368876]
Mean CV score: 0.7977905859750241

+*In[30]:*+
[source, ipython3]

# Predict using the final model on the test set
y_pred = best_xgb_model.predict(X_test)

+*In[31]:*+
[source, ipython3]

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy on test set: {accuracy}')

# Calculate confusion matrix
confusion = confusion_matrix(y_test, y_pred)
print(f'Confusion Matrix:\n{confusion}')

+*Out[31]:*+

Accuracy on test set: 0.7902298850574713
Confusion Matrix:
[[159 16   3]
 [27 90   5]
 [11 11 26]]

+*In[40]:*+
[source, ipython3]

Create a heatmap using seaborn
sns.heatmap(confusion, annot=True, fmt='d', xticklabels=['Predicted class 1', 'Predicted class 2',
'Predicted class 3'], yticklabels=['Actual class 1', 'Actual class 2', 'Actual class 3'], cmap='Blues')

+*Out[40]:*+
----<AxesSubplot:>
![png](output_26_1.png)

+*In[32]:*+
[source, ipython3]

# Calculate precision
precision = precision_score(y_test, y_pred, average='weighted')
print(f'Precision: {precision}')

# Calculate recall
recall = recall_score(y_test, y_pred, average='weighted')
print(f'Recall: {recall}')

+*Out[32]:*+

Precision: 0.7879799161447826
Recall: 0.7902298850574713

+*In[33]:*+
[source, ipython3]

# Calculate f1
f1 = f1_score(y_test, y_pred, average='weighted')
print(f'f1: {f1}')

+*Out[33]:*+
----
f1: 0.785246751662438

+*In[34]:*+
[source, ipython3]

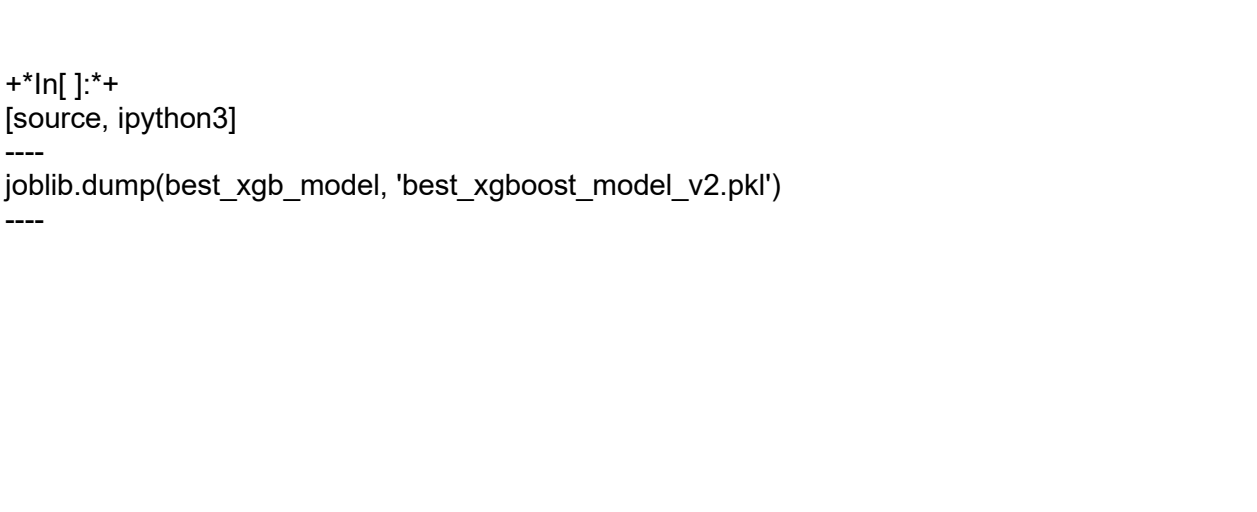
# Visualize feature importances using SHAP
# Create an Explanation object for SHAP values
explainer = shap.Explainer(best_xgb_model, X_train)
shap_values = explainer.shap_values(X_test)

+*Out[34]:*+

[15:45:33] WARNING: /Users/runner/work/xgboost/xgboost/src/c_api/c_api.cc:1240: Saving into deprecated binary model format, please consider using `json` or `ubj`. Model format will default to JSON in XGBoost 2.2 if not specified.

+*In[35]:*+
[source, ipython3]
---
shap.summary_plot(shap_values, X_test, feature_names=X.columns, plot_type='bar', max_display=X.shape[1])

+*Out[35]:*+



+*In[ ]:*+
[source, ipython3]
---
joblib.dump(best_xgb_model, 'best_xgboost_model_v2.pkl')

Supplementary Table

Cytokine	Reference	Cardio-Renal	Eosinophilic	p-value	FDR
IL-1b	1 (1-5)	1 (1-5)	5 (1-5)	0.006	0.048
IL-2	1 (1-5)	1 (1-5)	2.1 (1-5)	0.057	0.125
IL-2R	2288 (1357-3537)	2390 (1394-4300)	1562 (1057-2440)	0.013	0.048
IL-4	5 (5)	5 (5)	5 (5)	0.572	0.699
IL-5	5 (5)	5 (5)	5 (5)	0.773	0.850
IL-6	18.7 (6.4-56.3)	24.1 (7.2-68.3)	16 (5-36.5)	0.050	0.125
IL-8	22.7 (5-44.2)	23.4 (5-45.7)	5 (5-25.2)	0.012	0.048
IL-10	10.9 (6.12-20.8)	11.4 (6.35-22)	9.5 (5-19.1)	0.260	0.358
IL-12	5 (5)	5 (5)	5 (5)	0.254	0.358
IL-13	5 (5)	5 (5)	5 (5)	0.239	0.358
IL-17	5 (5)	5 (5)	5 (5)	0.867	0.867

Values are median (IQR)

STROBE Statement—Checklist of items that should be included in reports of ***cohort studies***

	Item No	Recommendation	Page No
Title and abstract	1	(*a*) Indicate the study's design with a commonly used term in the title or the abstract	1
		(*b*) Provide in the abstract an informative and balanced summary of what was done and what was found	2-3
Introduction			
Background/rationale	2	Explain the scientific background and rationale for the investigation being reported	4-5
Objectives	3	State specific objectives, including any prespecified hypotheses	4-5
Meth			
Study design	4	Present key elements of study design early in the paper	6-7 Supp
Setting	5	Describe the setting, locations, and relevant dates, including periods of recruitment, exposure, follow-up, and data collection	6-7 Supp
Participants	6	(*a*) Give the eligibility criteria, and the sources and methods of selection of participants. Describe methods of follow-up	6-7 Supp
		(*b*) For matched studies, give matching criteria and number of exposed and unexposed	
Variables	7	Clearly define all outcomes, exposures, predictors, potential confounders, and effect modifiers. Give diagnostic criteria, if applicable	6-7 Supp
Data sources/ measurement	8*	For each variable of interest, give sources of data and details of methods of assessment (measurement). Describe comparability of assessment methods if there is more than one group	6-7 Supp
Bias	9	Describe any efforts to address potential sources of bias	6-7 Supp
Study size	10	Explain how the study size was arrived at	NA
Quantitative variables	11	Explain how quantitative variables were handled in the analyses. If applicable, describe which groupings were chosen and why	6-7 Supp
Statistical methods	12	(*a*) Describe all statistical methods, including those used to control for confounding	6-7 Supp

		(*b*) Describe any methods used to examine subgroups and interactions (*c*) Explain how missing data were addressed (*d*) If applicable, explain how loss to follow-up was addressed (*e*) Describe any sensitivity analyses	Table 3
Results			
Participants	13*	(a) Report numbers of individuals at each stage of study—eg numbers potentially eligible, examined for eligibility, confirmed eligible, included in the study, completing follow-up, and analysed (b) Give reasons for non-participation at each stage (c) Consider use of a flow diagram	8-12 Tables 1 and 4
Descriptive data	14*	(a) Give characteristics of study participants (eg demographic, clinical, social) and information on exposures and potential confounders (b) Indicate number of participants with missing data for each variable of interest (c) Summarise follow-up time (eg, average and total amount)	8-12 Tables 1 and 4
Outcome data	15*	Report numbers of outcome events or summary measures over time	8-12 Tables 2 and 5

Main results	16	(*a*) Give unadjusted estimates and, if applicable, confounder-adjusted estimates and their precision (eg, 95% confidence interval). Make clear which confounders were adjusted for and why they were included (*b*) Report category boundaries when continuous variables were categorized (*c*) If relevant, consider translating estimates of relative risk into absolute risk for a meaningful time period	8-12 Tables
Other analyses	17	Report other analyses done—eg analyses of subgroups and interactions, and sensitivity analyses	Table 3, figure 2
Discussion			
Key results	18	Summarise key results with reference to study objectives	14-17
Limitations	19	Discuss limitations of the study, taking into account sources of potential bias or imprecision. Discuss both direction and magnitude of any potential bias	14-17
Interpretation	20	Give a cautious overall interpretation of results considering objectives, limitations, multiplicity of analyses, results from similar studies, and other relevant evidence	14-17
Generalisability	21	Discuss the generalisability (external validity) of the study results	14-17
Other information			
Funding	22	Give the source of funding and the role of the funders for the present study and, if applicable, for the original study on which the present article is based	Title page

*Give information separately for exposed and unexposed groups.

Note: An Explanation and Elaboration article discusses each checklist item and gives methodological background and published examples of transparent reporting. The STROBE checklist is best used in conjunction with this article (freely available on the Web sites of PLoS Medicine at http://www.plosmedicine.org/, Annals of Internal Medicine at http://www.annals.org/, and Epidemiology at http://www.epidem.com/). Information on the STROBE Initiative is available at http://www.strobe-statement.org.