Mapping Common Data Model data tables to an HDF5 file for reproducible machine learning workflows Janos G. Hajagos, Ph.D. **Department of Biomedical Informatics *** Stony Brook University Mapping OHDSI CDM data to HDF5 Jupyter Notebooks Data processing Database table 38000200 {nul 192450 3/12/2010 78820 319835 3/12/2010 3/13/201 38000200 {null} 1 4280 437827 3/12/201 3/13/2010 38000200 {null} 1 2720 38000200 {null} 1 73300 80502 3/12/2010 38000200 {null} 1 E9330 437469 3/12/2010 38000230 {null} 3 27541 435510 10/14/2009 10/14/2009 From the HDF5 container of the mapped OHDSI inpatient data a subset of the data is extracted into two separate matrices 439502 10/14/2009 10/14/2009 38000230 {null} 3 33818 independent/core array and /dependent/core arra 4 29606 4148842 4/1/2010 4/1/2010 38000230 {null} 38000230 {nul 4 20680 import numpy as n ohdsi file name = "synpuf inpatient combined readmission.hdf5" # Edit this for your file = h5py.File(ohdsi file name, # Define helper function for joining labels together def flatten_column_annotations(f5, base_path, abbreviation=None, field_separator="|", first_part=2, secon 🕮 janos_hajagos@ohdsi-docker-instance-1: ~/data/ohdsi2hdf5 - Google Chrome column_annotations = f5[base_path + "column_annotations"][... number_of_columns = column_annotations.shape[1] if abbreviation is not None abbreviation = field_separator + abbreviation abbreviation = flattened list = [column annotations[first part, i] + field separator + column_annotations[second_part, i] + abbreviation for i in range(number_of_columns)] JSON Document cleaned flattened list = for name in flattened list if name[-1] == field separator name = name.strip()[:-1] cleaned_flattened_list += [name] return np.array(cleaned flattened list, dtype=column annotations.dtyp measurement names = flatten column annotations(f5.measurement nath, abbreviation="M", second part=0an day": 2454855, pt_id": 38000200, ': "Inpatient header - 1st position" /_id": "SNOMED", measurement_names[0:10] code": "244.9", name": "Unspecified acquired hypothyroidism' ry id": "ICD904", Abnormal results of cardiovascular function studies 137989 'Urinalysis, by dip stick or tablet reagent for bilirubin, glucose, hemoglobin, ketones, leukocyt s, nitrite, pH, protein, specif' 'Increased blood lymphocyte number|320074| 66700 ocardiogram abnormal|320536|M Band neutrophil count above reference range|40481861|M Type II diabetes mellitus uncontrolled|40482801|M' Type 1 diabetes mellitus uncontrolled 40484648 M HDF5 'Body mass index 25-29 - overweight 4060705 [M'] bservation names = flatten column annotations(f5, observation path, abbreviation="M", second part=0) observation names[0:10 y(['No matching concept|0|M nilateral recurrent femoral hernia with obstruction but no gangrene|196731 'Unilateral recurrent inguinal hernia with obstruction but no gangrene|197022|M 'Unilateral recurrent inguinal hernia 201899 'Unilateral partial vocal cord paralysis 261047 synpuf inpatient combined readmis Unilateral complete paralysis of vocal cords|261888 Patient need for|4011950|M', 'Palliative care|4014023| Consultation|4014829|M', 'Vaccination required|4015724|M' dtvpe='|\$128') List View Icon Vi person names = flatten column annotations(f5, person path, first part=0 core_array HDF5 Dataset HDF5 Dataset array(['gender concept name|FEMALE', 'gender concept name|MALE ace_concept_name|Black or African American', ace_concept_name|No_matching_concept', 'race_concept_name|Whi ethnicity concept name Hispanic or Latino 'ethnicity_concept_name Not Hispanic or Latino', 'birth_julian_day' 'birth date'l 4000 dtype='|\$128' isit names = flatten column annotations(f5, visit occurrence path, first part=0 rray(['visit concept name|Inpatient Visi isit_type_concept_name|Visit derived from encounter on claim' No matching concept 'age_at_visit_start_in_years_int', 'age_at_visit_start_in_days', categorical_list isit start julian day', 'visit end julian day' Gingival and periodontal disease categorical_lis - 0) 'visit_start_datetime', 'visit_end_datetime'], condition_concept 132392 Staphylococcal scalded skin syndrome categorical_lis condition_concept Pressure ulcer stage 1 categorical_lis condition_concept 132412 Post-laminectomy syndrome categorical_lis condition_concept Chronic osteomyelitis of hand categorical_lis condition_concept Congenital anomaly of skin categorical_lis condition_concept Contusion of scapular region categorical list 421: 1114223 condition_concept Chronic myeloid leukemia in remissio categorical lis condition_concept 132583 Postablative hypothyroidism categorical lis condition_concept 132584 Thyroid hemorrhage and infarction categorical_lis 43]: f5.close() SCII String (128 character condition_concept Diplegic cerebral palsy categorical_lis ff5.close(condition_concept 132659 Acute apical periodontitis of pulpal or categorical_list condition_concept 132703 Lichen planus categorical_list condition_concept 132706 Disorder of nail categorical_list 🏠 🔘 🔘 🕂 🛒 🔮

Objective:

Predicting which inpatients are at highest risk for a 30-day readmission is an important factor for allocating care management resources. There is a rich literature of building machine learning models for predicting hospital readmissions [1]. This poster demonstrates the first steps in building a reproducible workflow for predicting inpatient readmission based on normalized data stored in the OHDSI (CDM) Common Data Model.

Key Concepts:

OHDSI Common Data Model allows healthcare data from various sources to be stored in a single schema with a standardized vocabulary. It grew out of the work to rigorously evaluate methods and data sets for detecting adverse drug events.

HDF5 is a flexible file container for storing arrays in an organized structure. The concept of groups which is similar to file paths allows the data to be stored in a hierarchy. It supports a range of data types and compression methods. It has been used for storing and analyzing the data for the LIGO experiment to detect gravitational waves, see: (https://losc.ligo.org/s/events/LVT151012/LOSC_Event_tutorial_LVT151012.html).

JSON (Javascript Object Notation) is a format for storing data in a machine independent way. Here it is used as an intermediary data format between a relational database and HDF5.

Jupyter notebooks grew out of the iPython project. It is used for creating reproducible and documented computations using a range of backends (Python, R, Julia).

CMS SYNPUF (Synthetic Public Use File) was created to encourage developers to build applications that utilize CMS's Medicare data. It does not contain any PHI (Protected Health Information). It is complex and large enough to demonstrate the feasibility for a project. It was mapped to the OHDSI CDM in 2016.

Methods:

A 100,000 encounter subset of CMS's SYNPUF (Synthetic Public Use File) and OHDSI vocabulary files were loaded into a PostGreSQL (version 9.6) relational database system. Tables were first denormalized at the visit occurrence level and the results were stored in temporary database tables. A total of 66,700 inpatient visits were extracted from the relational database as a set of JSON (Javascript Object Notation) documents. The JSON documents were mapped to a single HDF5 file. The mapped file contained multiple matrices where each row represents a separate inpatient visit and a column a feature associated with a domain. Both mapping steps are controlled by separate configuration files. The generated HDF5 file is then post processed and a 30-day readmission flag is appended to the HDF5 file as a separate dataset.

Using a Jupyter/IPython3 notebook features are selected across multiple groups and assembled into a single matrix (66,700 rows by 5,686 columns). All conditions, observations, procedures, and measurements were included. Additionally, the gender (female), length of stay in days, age in years, and past history of readmissions were included. In total there were 6,241 30-day hospital readmissions. A random forest model was trained to predict 30-day readmission following discharge. To measure performance of the classifier the AUC (Area Under the Curve) for the ROC (Receiver Operating Curve) was calculated

Results:

The HDF5 file generated from the SYNPUF inpatient visits set is 7.59 Mb (Megabytes), the post processed file is 9.40 Mb, and the file used in the predictive model fitting is 4.85 Mb. The complete analysis of the data can be found in the Github project: https://github.com/SBU-BMI/MappingOHDSI2HDF.

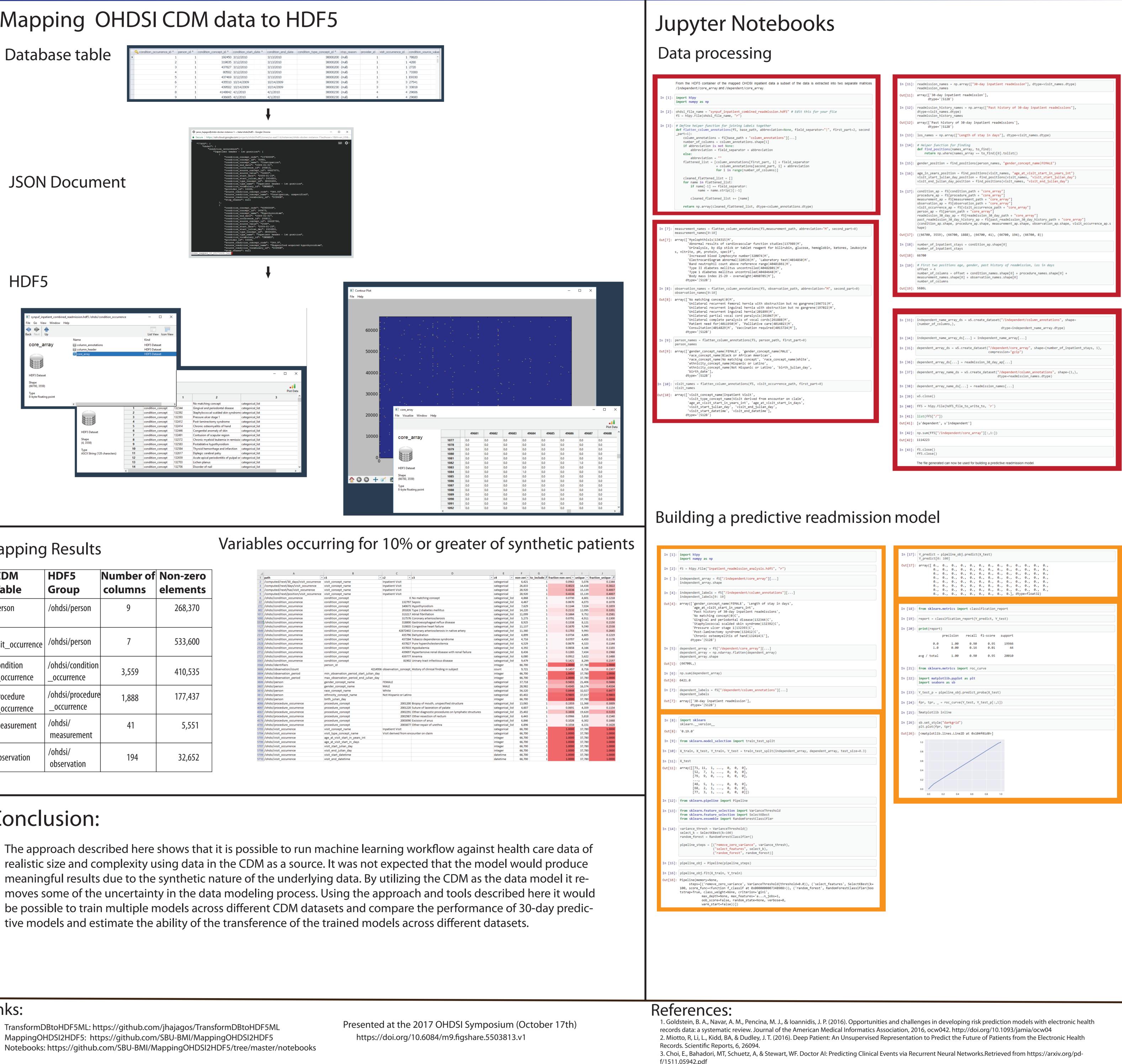
A random forest model with a population of 500 trees and two feature selection steps: remove zero variance features and select K best features (ANOVA F-score with 250 features) were utilized to predict 30-day inpatient readmission. On the testing set which was not utilized for training the total (AUC) area under the curve was 0.53. The predictive ability of the model trained on the CMS's synthetic data to predict 30-day readmission is poor.

Abstract:

Synthetic inpatient claims data in the OMOP Common Data Model were mapped to matrices in a HDF5 file. The content of the HDF5 were further manipulated to build a matrix for a 30-day post discharge readmission model. A random forest model was built to predict a patient's 30-day readmission risk. While the model's results were not predictive (AUC = 0.53) the modeling approach and pipeline can be applied to data in the Common Data Model (version 5.0). This opens up the possibility of rigorously comparing predictive performance of readmission models across different datasets.

Mapping Results

CDM Table	HDF5 Group	Number of columns	Non-zero elements
person	/ohdsi/person	9	268,370
visit_occurrence	/ohdsi/person	7	533,600
condition occurrence	/ohdsi/condition occurrence	3,559	410,535
procedure occurrence	/ohdsi/procedure occurrence	1,888	177,437
measurement	/ohdsi/ measurement	41	5,551
observation	/ohdsi/ observation	194	32,652



June 2007.

Conclusion:

tive models and estimate the ability of the transference of the trained models across different datasets.

Links:

TransformDBtoHDF5ML: https://github.com/jhajagos/TransformDBtoHDF5ML MappingOHDSI2HDF5: https://github.com/SBU-BMI/MappingOHDSI2HDF5 Notebooks: https://github.com/SBU-BMI/MappingOHDSI2HDF5/tree/master/notebooks

ion_analysis.hdf5", "r")
nt/core_array"][]
ent/column_annotations"][]
', 'Length of stay in days', s_int', atient readmissions',
<pre>Lsease 132344 C', n syndrome 132392 C', 393 C', L32412 C', and 132414 C'],</pre>
ore_array"][] en(dependent_array)
column_annotations"][]
on'],
t train_test_split
<pre>train_test_split(independent_array, dependent_array, test_size=0.3)</pre>
ð], ð], ð],
9], 9], 9]))
ine
ort VarianceThreshold ort SelectKBest mForestClassifier
4()
fier()
riance", variance_thresh),
s", select_k), , random_forest)]
s", select_k),
s", select_k), , random_forest)]
s", select_k), , random_forest)]

In [17]:	<pre>Y_predict = pipeline_obj.predict(X_test) Y_predict[0: 100]</pre>		
Out[17]:	array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,		
In [18]:	<pre>from sklearn.metrics import classification_report</pre>		
In [19]:	<pre>report = classification_report(Y_predict, Y_test)</pre>		
In [20]:]: print(report)		
	precision recall f1-score support		
	0.0 1.00 0.90 0.95 19966 1.0 0.00 0.16 0.01 44		
	avg / total 1.00 0.90 0.95 20010		
In [21]:	<pre>from sklearn.metrics import roc_curve</pre>		
In [22]:	<pre>import matplotlib.pyplot as plt import seaborn as sb</pre>		
In [23]:	Y_test_p = pipeline_obj.predict_proba(X_test)		
In [24]:	<pre>fpr, tpr, _ = roc_curve(Y_test, Y_test_p[:,1])</pre>		
In [25]:	%matplotlib inline		
In [26]:	<pre>sb.set_style("darkgrid") plt.plot(fpr, tpr)</pre>		
Out[26]:	[<matplotlib.lines.line2d 0x104f01d0="" at="">]</matplotlib.lines.line2d>		
	1.0		
	0.6		
	0.4		
	0.2		
	0.0 0.2 0.4 0.6 0.8 1.0		

4. Fernando Pérez, Brian E. Granger, IPython: A System for Interactive Scientific Computing, Computing in Science and Engineering, vol. 9, no. 3, pp. 21-29, May/-