Paper PP23

From Data Iron to Data Gold

Ashwini Yermal Shanbhogue, Norwood, MA, USA

ABSTRACT

In recent years, Real World Data (RWD) has gained importance in the clinical trial space. One of the sources of RWD is claims data. Claims data is usually present in large datasets, which until recently were hard to process and gain any meaningful insights from. However, recent advances in Machine Learning (ML) have made this task easier. In this poster, I will demonstrate how an R package that includes several ML algorithms can be used to make predictions and thus gain insights from a claims data set, the CMS 2008-2010 Data Entrepreneurs' Synthetic Public Use File (DE-SynPUF).

INTRODUCTION

Real World Data (RWD) is defined as, 'data relating to patient health status and/ or the delivery of health care routinely collected from a variety of sources like electronic health records (EHRs), medical claims and billing data, data from product and disease registries, patient-generated data, including from in-home-use settings, and data gathered from other sources that can inform on health status, such as mobile devices' (FDA, 2018, CDISC, n.d.). Among these sources, claims data will be the focus of this paper.

Claims data consists of billing codes that a patient's healthcare provider submits to the patient's insurance provider so that the healthcare provider may get reimbursed for the services provided to the patient. Claims data is a valuable source of RWD due to several reasons. First among those is the diversity of the data. It offers a broad representation across different demographics, unlike clinical trial data. Second, is the ability to follow a patient's journey across multiple healthcare providers as long as the insurance provider remains constant, which can be very advantageous for longitudinal studies. Thirdly, since every encounter with a healthcare provider generates billing codes, by nature, claims data sets are very large. This is advantageous because it can be used to gain insights into rare diseases (Stein et al., 2014, Wilson & Bock, n.d.). This large size may also become a disadvantage since very large data sets can be difficult to process at the patient level. Due to recent advances in Machine Learning (ML) and Cloud Computing, however, this is becoming easier to accomplish.

Although patient-level prediction is now possible for large data sets, it may be hard to share or validate ML models across studies or data sets since the data may not have uniform structure or vocabulary i.e. it may not be standardized. The models may also not have been developed or reported transparently or while following the best principles for development and reporting and therefore not standardized. The use of data in a standardized format or common data model (CDM) like Observational Medical Outcomes Partnership (OMOP) along with standardized ML models can circumvent these problems.

Having claims data in the OMOP CDM is also advantageous while submitting it to regulatory agencies since regulatory agencies require clinical trial data to be submitted in the Study Data Tabulation Model (SDTM) standard structure and it is easier to Extract- Transform- Load (ETL) data in a CDM to SDTM format compared to non-standard data (Shanbhogue, 2022).

'PatientLevelPrediction' is an R package developed by the Observational Health Data Sciences and Informatics (OHDSI) program while following the Transparent Reporting of a multivariable prediction model for Individual Prognosis Or Diagnosis (TRIPOD) statement, which aims to improve the transparency of reporting of a prediction model study (Reps et al., 2018, Collins et al., 2015). 'PatientLevelPrediction' can be used to build and validate patient-level predictive models using data in the OMOP CDM. The package thus helps predict which patients among a population at risk will develop a particular outcome during a future time at risk based on information about the patients in a window of time before the moment of prediction. The ML algorithms available within 'PatientLevelPrediction' include Regularized logistic regression, Random forest, Gradient boosting machines, Decision tree, Naive Bayes, K-nearest neighbors, Neural network, AdaBoost, and Support vector machines. Custom ML algorithms may also be added.

Advantages of using the 'PatientLevelPrediction' package:

1. Developed following TRIPOD, which has clear guidelines for prediction model development, validation, and reporting, thus encouraging process transparency.
2. Developed to be used on data sets in the OMOP CDM. Since CDMs have standardized structure and vocabulary, using the standardized 'PatientLevelPrediction' package to analyze data present within a CDM will result in models and results that are standardized and easy to comprehend and share among researchers.
3. Makes external validation easier since validation can be carried out on other data sets in the OMOP CDM, the number of which is growing by the day.

CMS 2008-2010 Data Entrepreneurs' Synthetic Public Use File (DE-SynPUF) is a realistic set of synthetic claims data released by the Centers for Medicare and Medicaid Services (CMS) to facilitate the development of research tools that may then be applied to actual CMS data (CMS, 2013). This data set has been converted into the OMOP CDM by the open-source community (CMS Working Group of OHDSI & University of New Mexico, 2018, May 7).

I will demonstrate how the 'PatientLevelPrediction' R package can be used to create an ML model using CMS 2008-2010 DE-SynPUF data in OMOP CDM to answer the following question-

Which patients with new onset Type II Diabetes Mellitus (T2DM) will go on to develop Coronary arteriosclerosis in native artery (CA) within 2 years?

METHOD

STUDY DESIGN

Following the best practices recommended by OHDSI in several places including in the documentation for 'PatientLevelPrediction' (OHDSI, 2021), I developed the following study design.

Target cohort: New Onset Type II Diabetes Mellitus (T2DM)
Outcome cohort: Diagnosis of Coronary arteriosclerosis in native artery
Time at risk (TAR): 2 years
Model: Regularized Logistic Regression
Covariates: Age, Sex, and Race

STUDY DATA

I downloaded the CMS 2008-2010 DE-SynPUF in OMOP CDM 100k data set from The Registry of Open Data on AWS (AWS, n.d.) and uploaded it to a PostgreSQL database within the 'public' schema. Since the 'concept' table for the data set was not available on AWS, I downloaded it from the Google Cloud console, where the rest of the data is also available as a public data set (Google Cloud console, n.d.).

CUSTOM COHORTS USING SQL

I then created custom cohorts for my study using the following SQL code and stored them in the 'dmcohorts' table of the 'results' schema of my PostgreSQL database.

```
INSERT INTO results.dmcohorts (cohort_definition_id,
                               subject_id,
                               cohort_start_date,
                               cohort_end_date)
SELECT 1 AS cohort_definition_id,
DM.person_id AS subject_id,
DM.condition_start_date AS cohort_start_date,
observation_period.observation_period_end_date AS cohort_end_date
FROM
```

```
(
    SELECT person_id, MIN(condition_start_date) AS condition_start_date
    FROM public.condition_occurrence
    WHERE condition_concept_id= 201826
    GROUP BY person_id
) DM
INNER JOIN public.observation_period
ON DM.person_id= observation_period.person_id
AND DM.condition_start_date >= observation_period.observation_period_start_date
AND DM.condition_start_date <= observation_period.observation_period_end_date;


INSERT INTO results.dmcohorts (cohort_definition_id,
                           subject_id,
                           cohort_start_date,
                           cohort_end_date)
SELECT 2 AS cohort_definition_id,
CA.person_id AS subject_id,
CA.condition_start_date AS cohort_start_date,
observation_period.observation_period_end_date AS cohort_end_date
FROM
(
    SELECT person_id, MIN(condition_start_date) AS condition_start_date
    FROM public.condition_occurrence
    WHERE condition_concept_id= 42872402
    GROUP BY person_id
) CA
INNER JOIN public.observation_period
ON CA.person_id= observation_period.person_id
AND CA.condition_start_date >= observation_period.observation_period_start_date
AND CA.condition_start_date <= observation_period.observation_period_end_date;
```

MODEL BUILDING

I then used the 'PatientLevelPrediction' R package and CMS 2008-2010 DE-SynPUF data in OMOP CDM to create an ML model in the following manner.

```
#installing remotes package
```{r}
install.packages("remotes")
```


#installing FeatureExtraction and PatientLevelPrediction packages
```{r}
remotes::install_github("OHDSI/FeatureExtraction")
remotes::install_github("OHDSI/PatientLevelPrediction")
```
```

#loading required packages
```{r}
library(FeatureExtraction)
library(PatientLevelPrediction)
library(magrittr)
```

#connecting to PostgreSQL, defining the database containing the raw data tables, the cohort table, and the version of OMOP CDM being used
```{r}
connectionDetails <- createConnectionDetails(server="localhost/postgres", port=5433, dbms="postgresql", user="postgres", password="XXXXXX", pathToDriver ="c:/temp/jdbcDrivers")
cdmDatabaseSchema <- "public"
cohortsDatabaseSchema <- "results"
cdmVersion <- "5"
```

#creating covariates using a function of 'FeatureExtraction' package
```{r}
covariateSettings <- createCovariateSettings(useDemographicsGender = TRUE, useDemographicsAge = TRUE, useDemographicsRace = TRUE)
```

#defining database details i.e conncection details, where CDM data, target and outcome cohorts are stored and what the version of CDM used is
```{r}
databaseDetails <- createDatabaseDetails(
  connectionDetails = connectionDetails,
  cdmDatabaseSchema = cdmDatabaseSchema,
  cdmDatabaseName = 'CMS',
  cohortDatabaseSchema = cohortsDatabaseSchema,
  cohortTable = 'dmcohorts',
  targetId = 1,
  outcomeDatabaseSchema = cohortsDatabaseSchema,
  outcomeTable = 'dmcohorts',
  outcomeIds = 2,
  cdmVersion = 5
)
```

#defining any additional restrictions and creating plpData object. getPlpData function extracts all the data defined
```{r}
restrictPlpDataSettings <- createRestrictPlpDataSettings(sampleSize = NULL)

plpData <- getPlpData(
  databaseDetails = databaseDetails,
  covariateSettings = covariateSettings,
  restrictPlpDataSettings = restrictPlpDataSettings
)
```

#saving plp data
```{r}
savePlpData(plpData, 'ca_in_dm_data')
```

#defining additional inclusion criteria
```{r}
populationSettings <- createStudyPopulationSettings(
  removeSubjectsWithPriorOutcome = TRUE,
  priorOutcomeLookback = 730,
  riskWindowStart = 1,
  riskWindowEnd = 730,
  minTimeAtRisk = 729,
  includeAllOutcomes = TRUE
)
```

#splitting data into training and test datasets
```{r}
splitSettings <- createDefaultSplitSetting(
  testFraction = 0.2,
  trainFraction = 0.8,
  splitSeed = 23,
  nfold = 10,
  type = "stratified"
)
```

#defining sample and feature engineering settings. Default settings were used for this study
```{r}
sampleSettings <- createSampleSettings()
featureEngineeringSettings <- createFeatureEngineeringSettings()
```

defining preprocessing settings
```{r}
preprocessSettings <- createPreprocessSettings(
  minFraction = 0.01,
  normalize = TRUE,
  removeRedundancy = TRUE
)
```

#creating settings for a LASSO Logistic regression model. Default settings were used for this study
```{r}
lrModel <- setLassoLogisticRegression()
```

#running Patient level prediction and saving it in 'lrResults' object
```{r}
lrResults <- runPlp(
  plpData = plpData,
  outcomeId = 2,
  analysisId = 'PP23',
  analysisName = 'CA in DM from CMS synthetic data',
  populationSettings = populationSettings,
  splitSettings = splitSettings,
  sampleSettings = sampleSettings,
  featureEngineeringSettings = featureEngineeringSettings,
  preprocessSettings = preprocessSettings,
  modelSettings = lrModel,
  logSettings = createLogSettings(),
  executeSettings = createExecuteSettings(
        runSplitData = T,
        runSampleData = T,
        runfeatureEngineering = T,
        runPreprocessData = T,
        runModelDevelopment = T,
        runCovariateSummary = T
  ),
  saveDirectory = file.path(getwd(), 'pp23')
)
```

#saving model
```{r}
savePlpModel(lrResults$model, dirPath = file.path(getwd(), "model"))
```

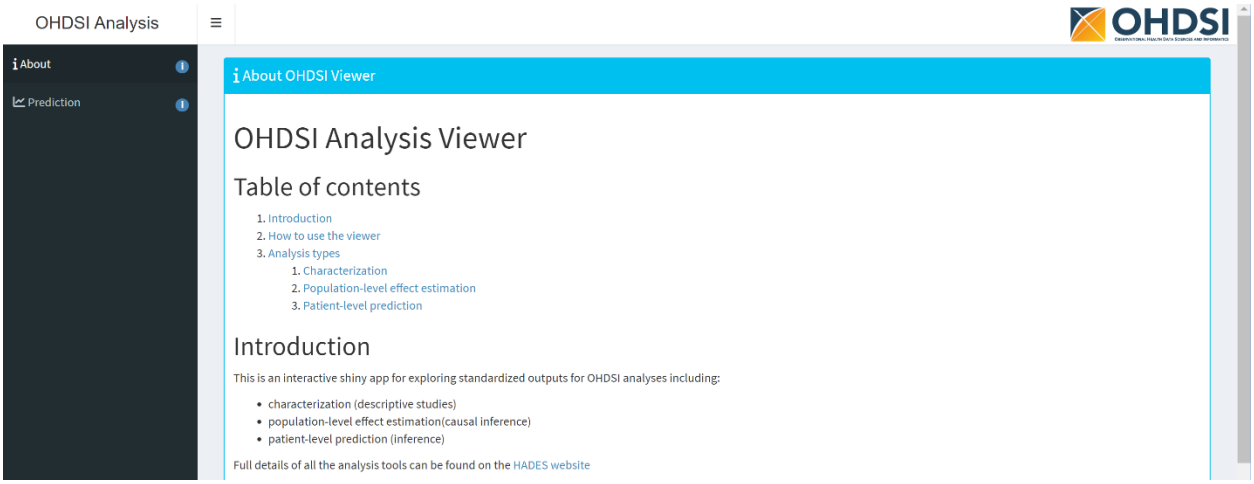
#saving full results structure
```{r}
savePlpResult(lrResults, file.path(getwd(), "lr"))
```

#viewing prediction details including 'value' column, which is predicted risk of outcome
```{r}
View(lrResults$prediction)
```

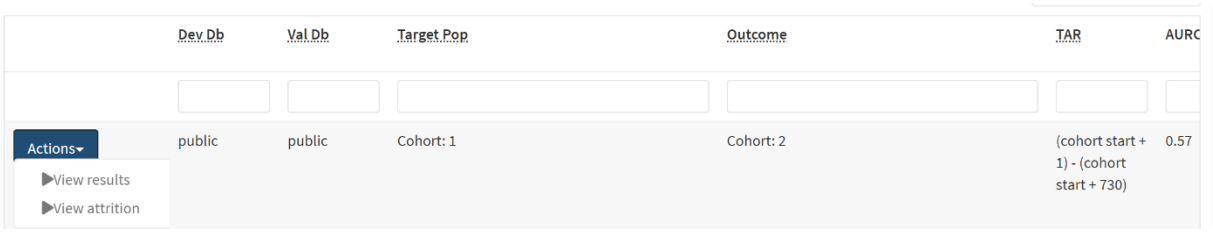
#viewing results in a Shiny app
```{r}
viewPlp(lrResults)
```

RESULTS

I was then able to view my results in the OHDSI Analysis Viewer Shiny application (pictured below).



Clicking on the Prediction tab on the left leads to a page where all the models are listed by design and summarized. Selecting the 'View Model' option on the 'Action' button to the left leads to that model's summary page.



Here, we can see that my model had an AUROC (an indicator of model performance) of 0.57 and that out of my cohort size (T size) of 3389, 2291 showed the outcome (O size), which is an incidence rate of 67.6% (O Incidence (%)).

Clicking on the 'View Attrition' option on the 'Action' button brings up a summary of the attrition of subjects in my cohort due to the inclusion criteria chosen.

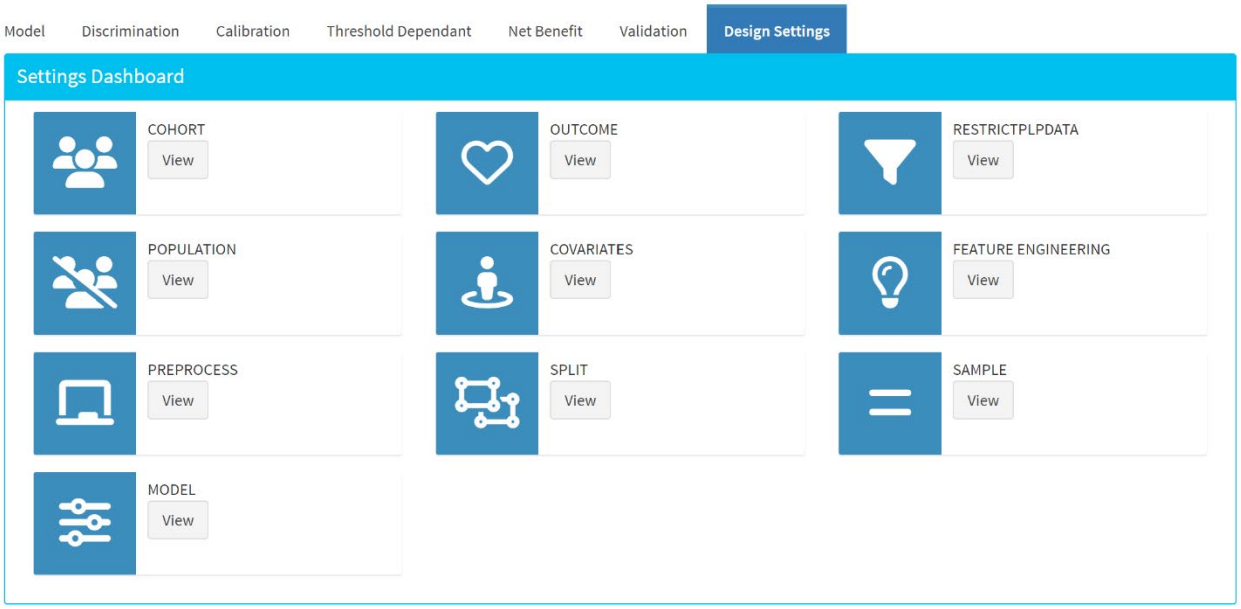
Attrition

Show 10 entries Search:

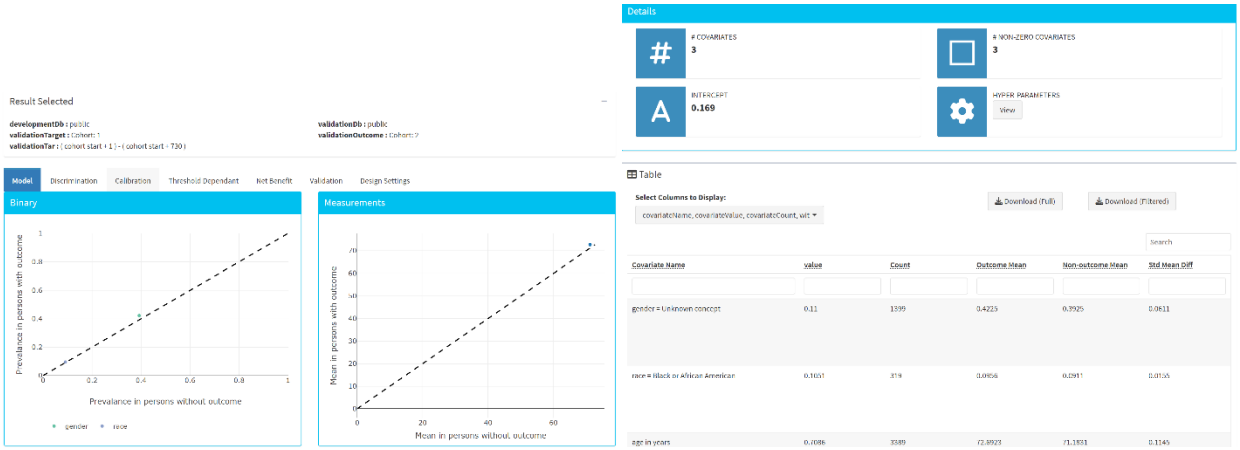
	description	targetCount	uniquePeople	outcomes
1	Original cohorts	5602	5602	3605
2	Initial plpData cohort or population	5602	5602	2291
3	No prior outcome	4457	4457	2291
4	Removing non-outcome subjects with insufficient time at risk (if any)	3389	3389	2291

Clicking on the 'View Results' option on the 'Action' button brings up a Results page with several tabs.

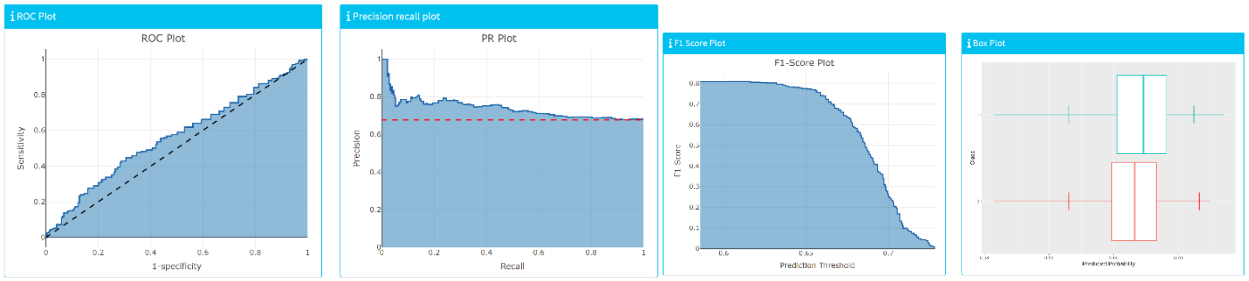
On the 'Design Settings' tab, we can see all the design options I had selected for my model.

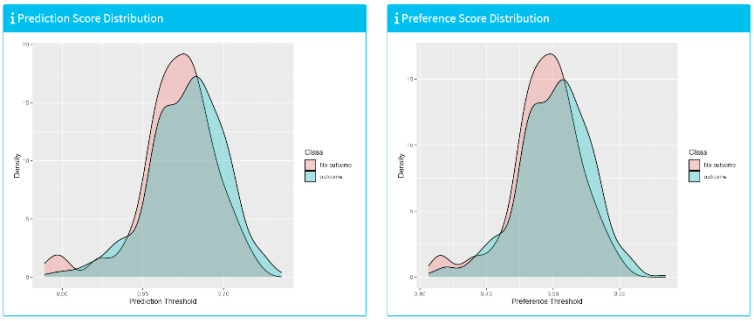


On the 'Model' tab we can see the coefficients of all the covariates and how common each covariate is among the people who had the outcome versus those who didn't (Outcome Mean and Non- Outcome Mean).

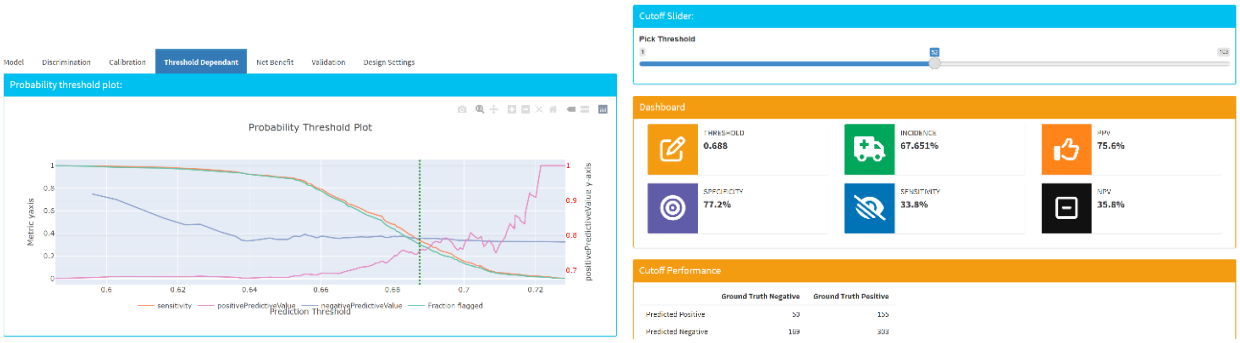


On the 'Discrimination' tab, we can see how the model is performing with the help of several plots- the ROC plot, which shows the tradeoff between specificity and sensitivity, the Precision-Recall curve, which shows the tradeoff between Positive Predictive Value (PPV) and sensitivity across thresholds, the F1 score plot, a box plot showing predicted risk for those with (class 1) and without (class 0) the outcome, prediction score distribution and preference score distribution, showing how well the model is able to discriminate between those with and without outcome. The metrics on this tab can be used to fine-tune and optimize the model.





The 'Threshold dependent' tab has the Incidence, Sensitivity, Specificity, Positive Predictive Value (PPV), and Negative Predictive Value (NPV) at different thresholds. We can pick a threshold to say- if the predicted risk is above that value, the outcome should be considered positive. The cutoff slider can be used to pick the threshold.



Once the threshold has been picked, the ML model we have trained so far can be applied to a fresh data set to create patient-level predictions.

CONCLUSION

In this paper and poster, I have demonstrated how an R package developed following TRIPOD guidelines, the 'PatientLevelPrediction' can be used along with data in a standard format- CMS 2008-2010 DE-SynPUF in OMOP CDM- to develop an ML model which is easy to share and validate across data sets.

Patient-level prediction could have several applications in the realm of clinical trials, some of which are-

1. Clinical Trial Design and Recruitment: To enhance the efficiency of clinical trials by identifying candidates who are more likely to benefit from a given intervention, thus streamlining trial design and recruitment.
2. Drug re-purposing: To determine if a drug the patient is being given already is likely to show an effect that was previously unknown. If the effect is therapeutic in nature, clinical trials can be carried out to gain regulatory approval for the new indication.
3. Post-marketing safety surveillance: To monitor patients who are prescribed specific drugs, in order to predict and mitigate the risk of adverse events, enhancing patient safety.
4. Personalized medicine: To determine if a patient's profile before a certain point in time is likely to make them react in a particular way to a drug/ therapy beyond that point and to tailor medical treatment to optimize outcomes.

Through these diverse applications, the integration of ML models with standardized clinical data has the potential to revolutionize the field of clinical trials, making it more data-driven, efficient, and personalized.

REFERENCES

1. Framework for FDA's Real World Evidence Program. (2018, December). FDA. https://www.fda.gov/media/120060/download
2. CDISC. (n.d.). Real World Data | CDISC. Www.Cdisc.org. Retrieved January 6, 2024, from https://www.cdisc.org/standards/real-world-data

3. Stein, J. D., Lum, F., Lee, P. P., Rich, W. L., & Coleman, A. L. (2014). Use of health care claims data to study patients with ophthalmologic conditions. Ophthalmology, 121(5), 1134–1141. https://doi.org/10.1016/j.ophtha.2013.11.038
4. Wilson, J., & Bock, A. (n.d). The benefit of using both claims data and electronic medical record data in health care analysis [White paper]. Optum. https://www.optum.com/content/dam/optum/resources/whitePapers/Benefits-of-using-both-claims-and-EMR-data-in-HC-analysis-WhitePaper-ACS.pdf
5. Shanbhogue, A. Y. (2022). RWD (OMOP) to SDTM (CDISC): A primer for your ETL journey. In www.pharmasug.org. PharmaSUG 2022, Austin, TX, United States of America. https://www.pharmasug.org/proceedings/2022/RW/PharmaSUG-2022-RW-160.pdf
6. Reps J. M., Schuemie M. J., Suchard M. A., Ryan P. B., & Rijnbeek P. (2018). Design and implementation of a standardized framework to generate and evaluate patient-level prediction models using observational healthcare data. Journal of the American Medical Informatics Association, 25(8), 969-975. https://doi.org/10.1093/jamia/ocy032.
7. Collins, G. S., Reitsma, J. B., Altman, D. G., & Moons, K. G. (2015). Transparent Reporting of a Multivariable Prediction Model for Individual Prognosis or Diagnosis (TRIPOD): the TRIPOD statement. Annals of Internal Medicine, 162(1), 55–63. https://doi.org/10.7326/M14-0697
8. Centers for Medicare & Medicaid Services. (2013, January 15). CMS 2008-2010 Data Entrepreneurs' synthetic public use file (DE-SynPUF). CMS.gov. https://www.cms.gov/data-research/statistics-trends-and-reports/medicare-claims-synthetic-public-use-files/cms-2008-2010-data-entrepreneurs-synthetic-public-use-file-de-synpuf
9. CMS Working Group of OHDSI & University of New Mexico. (2018, May 7). OHDSI/ETL-CMS: Workproducts to ETL CMS datasets into OMOP Common Data Model. GitHub. https://github.com/OHDSI/ETL-CMS
10. AWS (n.d) CMS 2008-2010 Data Entrepreneurs' Synthetic Public Use File (DE-SynPUF) in OMOP Common Data Model. Retrieved on December 29, 2023 from https://registry.opendata.aws/cmsdesynpuf-omop
11. Google Cloud console. (n.d.). Synthetic Patient Data in OMOP. Retrieved on December 29, 2023 from https://console.cloud.google.com/marketplace/product/hhs/synpuf
12. OHDSI. (2021, December 16). PatientLevelPrediction/inst/doc/BuildingPredictiveModels.pdf at main · OHDSI/PatientLevelPrediction. GitHub. https://github.com/OHDSI/PatientLevelPrediction/blob/main/inst/doc/BuildingPredictiveModels.pdf

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name: Ashwini Yermal Shanbhogue

Email: ash23shan@yahoo.com

GitHub: https://github.com/ash23shan/From-Data-Iron-to-Data-Gold