[2023/2024]

READY TO BE DISCHARGED:

EXAMINING HOSPITAL READMISSIONS

Group 33
Maria Batrakova - 20230739
João Gonçalves - 20230560
Nicolau Dulea - 20230544
Matheus Felisberto - 20230585
Francisco Campos - 20230565

Abstract

This machine learning project aimed to predict hospital readmissions, a critical factor in healthcare outcomes and cost management. Hospital readmission prediction is a challenging task due to data imbalances, privacy concerns, and the complexity of healthcare data. Despite these challenges, the primary objective was to develop accurate predictive models to enhance patient care quality, reduce preventable readmissions, and improve overall patient outcomes.

The project encompassed two classification tasks: binary classification to predict readmission within 30 days post-discharge and multiclass classification to categorize readmission durations into "No," "<30 days," and ">30 days." These models aimed to empower healthcare providers with actionable information for proactive and patient-centric care.

One of the most surprising revelations was that minimal preprocessing often outperformed more complex approaches including extensive feature selection, and that challenged initial expectations a lot. Moreover, specifically for our setup with the dataset given we were able to identify a unique pattern encoded in patients' encounter_ids that contributed a lot to the models performance.

In conclusion, best results were obtained by carefully choosing the models taking into account some technical characteristics of their implementation.

This project showcased the potential of machine learning in predicting hospital readmissions. While the outcomes diverged from initial expectations, they shed light on the intricacies of healthcare analytics and paved the way for further research and model refinement.

Introduction

This machine learning project focuses on predicting hospital readmissions, a key factor influencing patient well-being and contributing to more sustainable healthcare spending.

Hospital readmission prediction can prove to be difficult. From analyzing research with similar objectives we found that the problems for predictions stem from data-related issues, primarily characterized by data imbalances, privacy concerns, and the inherent complexity of healthcare data. For example, even if patients' medical records are gathered and presented in detail, the class distributions are often imbalanced meaning readmission visits are much fewer than normal visits. Having this in mind we are convinced to be skeptical in achieving great prediction results.

Nevertheless the primary objective is to develop an accurate predictive model that healthcare providers can seamlessly integrate into their decision-making processes, thus:

- enhancing patient care quality;

- reducing preventable readmissions by identification of high-risk patients;

- improving overall patient outcomes

The project includes 2 classification tasks:

- *binary classification:* develop a classification model capable of precisely forecasting whether a patient will experience hospital readmission within 30 days post-discharge. A reliable prediction empowers healthcare providers to proactively implement preventive measures and offer timely interventions, potentially resulting in substantial savings in healthcare costs;

- *multiclass classification*: create a multiclass classifier that anticipates the duration until a patient's readmission, categorizing them into "No," "<30 days," and ">30 days." This model offers more detailed insights into patient risk levels, assisting hospitals in customizing their post-discharge care and follow-up procedures based on the predicted time frame.

In summary, our machine learning project on hospital readmissions aligns with the mission of empowering healthcare professionals with actionable information, fostering a more proactive, patient-centric, and efficient healthcare ecosystem.

Data Exploration and Preprocessing

We started by exploring the train dataset to check if there were some problems or incoherences to be fixed, duplicates, missing values and more. To achieve this, we looked at [**Figure 1**] and [**Table 1**] to [**Table 2**]. During the initial analysis, we also concluded that the data is not well balanced in terms of the target variable in the binary classification, having 63286 "*False*" and only 7950 "*True*" as expected.

Exploratory Data Analysis (EDA)

Our train dataset has 71236 rows of data and 31 variables, 2 of them being target variables.

As a first analysis we observed that the features *glucose_test_result, a1c_test_result* and *weight* have a lot of missing values with a lot of other features also having missing values. These missing values were mostly identified with the ”?” sign or just blank. As far as duplicate values go, we have 0 exact duplicates, but we observed ~20% of patients that have more than 1 encounter, totally covering ~25% of all encounters.

For the categorical variables, in [**Table 1**], we observed that diagnoses and medication have a lot of categories, which can be a problem, the “?” sign appears a lot of times for certain features and for the variable *country* there is 1 constant value.

For the metric features in [**Figure 1**] we observed that the features *outpatient_visits_in_previous_year, emergency_visits_in_previous_year* and *inpatient_visits_in_previous_year* do not have a lot of variability, most of their values being 0.

All of these problems and more inconsistencies were tackled after in the preprocessing stage of the project.

Extra Block of Analysis

Some proportion of patients (~20%) have more than 1 encounter. Taking this into consideration we can try to get some insights into how the encounter_ids are organized, and if there is any pattern that can contribute to making our predictions more accurate. Moreover, we assume that encounter_ids aren't assigned randomly in the hospitals.

This extra block consists of the following parts:

- unite train and test datasets to get a big picture on our encounter_ids;

- calculate 'distances' to the next and previous encounters if a patient has any, assuming that there might be a time component;

- calculate number of encounters per patient;

- plot histograms of distance to next encounters using multiclass labels in order to better understand our potential new feature based on *encounter_id*.

After such transformations on the united dataset and plotting distances to the next encounter for labeled objects from the train we can observe 4 peaks [**Figure 2**]. They are too obvious to consider them artifacts.

At first the probability density that the person will be readmitted drops with 'distance' passing after discharge. That makes a lot of common sense. But later it starts growing again. And these peaks have a periodic pattern.

One hypothesis is that it might be connected with annual periodicity of some diseases (infectious or winter traumas). But the distribution of medical speciality and diagnoses of patients "around the peaks"

doesn't support it. Also we can try to deduce the ratio between distances in encounters_id and time by looking at humps borders of the classes of patients readmitted in the first 30 days and after the first 30 days. Then 1 month will correspond to around 2700 units of the encounter_id distance. But then 1 year is about 32000, while the distance between peaks is about 260000. It is also very suspicious that the period between peaks is not just around the same, but totally the same. Finally, it is surprising that we do see distances to the next encounter for the patients from the class of not readmitted. And, moreover, they group around the peaks, but only from the left rising side.

So, there should be another explanation. The period between peaks is around 262000, it suspiciously resembles 2^{18}. So, let's check, what if the first 18 bits of *encounter_id* are monotonically increased with more or less the same speed corresponding to the time of admission. But the meaning of the 2 highest bits is still unclear. [**Figure 3**]

After interpreting only the first 18 bits of *encounter_id* as a time-correlated component (t_e) we do not see repeated peaks any more. Also the amount of encounters labeled “not readmitted” but with positive distance to the next encounter significantly dropped. Unfortunately, it didn’t drop to 0, and we see a lot of such artifacts at the end of our t_e axis. [**Figure 4**]. The length of this interval is about 30000, which may correspond to 1 last year of data in the dataset. Maybe, there is some incorrectness of labeling or maybe the coding of *encounters_id* was changed in the last year.

So, now we definitely can extract a new feature: distance to the previous encounter. It might be useful. (and it will be proven later). For the training process we didn't ‘cheat’ and calculated this feature based only on the data from the train dataset.

Cheating

We also can extract another new feature. The one which we were plotting in the previous section to decode *encounter_id* - the distance to the next encounter. This new feature is extremely powerful, with it we were able to easily get phenomenal results for a f1 score of about 0.712, which is twice bigger than any result without it. But we consider it a cheat feature, because it looks into the future. In the real world we will have neither train nor test data from the future to predict readmission.

By the way, making a new feature *encounters_num*, by just summing the number of all encounters of the patient is also some sort of cheating. Because on average half of such encounters will be again from the future. We also refuse to use such a feature in further analysis.

Preprocessing

In the preprocessing phase of our project we divided the process into two major parts: Soft (minimal) preprocessing and Hard preprocessing .

This section focuses on the Soft preprocessing steps undertaken to prepare the data.

Soft (Minimal) Preprocessing

1. **Outliers Handling**
   Our method involves using the Interquartile Range (IQR) to find outliers in our metric variables such as *outpatient visits, emergency visits, inpatient visits, average pulse rate, hospital stay length, lab tests, non-lab procedures, number of medications,number of diagnoses*. We visually identify outliers through boxplots and then manually filter to attenuate their impact on the analysis [**Figure 5**].

2. **Data Cleaning**

   We have created a function that identifies columns containing specific symbols or strings indicating missing data and replaces them with NaN values. Our aim is to normalize the representation of missing values for later processing and analysis, creating a cleaner and more consistent data set.

3. **Data Type Conversions**

   We encountered the necessity for data type conversions to accurately interpret variables initially presented as strings.

   Boolean variables values in *change_in_meds_during_hospitalization, prescribed_diabetes_meds* we mapped accordingly into 0 and 1.

   Applying a similar approach to our output labels, *readmitted_binary* mirrored the transformation of *prescribed_diabetes_meds*. The *readmitted_multiclass* label saw 'No' encoded as 0, '<30 days' as 2, and '>30 days' as 1.

   Further adjustments were made to the variables *glucose_test_result* and *a1c_test_result*, assigning numerical values to 'Norm', '>200', '>300', as well as 'Norm', '>7', '>8', respectively. With all of these transformations being detailed in [**Table 3**].

   An additional transformation involved categorizing *Age* into ten distinct categories, as outlined in [**Table 4**], enhancing the compatibility with models specifically designed for numerical analysis.

4. **Feature Engineering**

   We noticed a variety of types within the medication characteristic. To simplify, we created a new binary feature for each medication type, assigning a value of 1 if prescribed to the patient and 0 if not.

   Moreover, we implemented an additional feature highlighting the most frequent combinations within the medication column.

Hard preprocessing

1. **Data Type Conversions**

   Building on the soft preprocessing steps, we extend the approach to categorical features like *discharge_disposition, medical_specialty, admission_type,* and *admission_source*. Our strategy involves adopting a categorization approach to group similar discharge dispositions, medical specialties, admission types, and admission sources into larger categories. The semantic mappings were created with the help of ChatGPT.

   Furthermore, the extensive variety of diagnoses in *primary_diagnosis*, *secondary_diagnosis,* and *additional_diagnosis* posed challenges. To address this, we opted to group the diagnoses into broader categories based on the official ICD-9 Classification. Moreover, we have mixed data types in those columns, so we need to convert them accordingly.

   All these transformations not only reduce dimensionality and balance subcategories, but also provide a more manageable set of features, as outlined in [**Table 4**, **5**, **6**] and [**Table 7**].

2. **Feature Engineering**

   We create new features such as *has_payer_code* (transformed into a binary variable), specifying if the individual possesses a payer code or not, and *total_visits* that represents the aggregation of all possible visit types simplifying the representation of an individual's overall visit history.

   The *diabetes_diagnosis* feature discerns whether a patient has a diabetes diagnosis, assigning 1 for affirmative and 0 for negative outcomes based on the presence of code 250 in the diagnoses, whereas the *onco_diagnosis* feature acts as a binary (1 for malignant neoplasms, 0 otherwise), determined by examining whether any diagnosis code falls within the 140 to 209 range.

   For mental health considerations, the *psycho_diagnosis* feature was designed as a binary indicator (1 for mental disorders, 0 otherwise). This determination was made by checking for the presence of the label 'Mental Disorders' in primary, secondary, and additional diagnosis groups.

   Both the *diabetes_diagnosis* and *onco_diagnosis* features were derived from the ICD-9 Classification, providing valuable insights into diabetes and malignant neoplasms(cancer), respectively.

3. **Incoherence Checking**

   Regarding Incoherence, we found that there are "expired" patients in the dataset. In this

case, they have no predictive value, because being "expired" automatically means "not being readmitted", since the patient is no longer alive.

Feature Selection

Feature selection plays an important role in the development of machine learning models, as it influences both their performance and interpretability. It helps reduce overfitting, thereby enhancing models' generalization capabilities, especially in models sensitive to data noise. This process also decreases training time due to less computational demand.

In our project, we implement a systematic approach to feature selection, aiming to identify the most relevant features while discarding those that are redundant. To finally decide on whether to keep a feature or discard it, we perform some sort of 'voting' among all the methods used. We repeat the process of feature selection twice, separately for the binary classification task and then for the Multiclass Classification problem. The methods we utilize are described below.

Filter-Based methods

Filter-based methods in feature selection are techniques that rely on statistical measures to evaluate the significance of different features independently of any machine learning model. These methods assess the relationship between each feature and the target variable using specific statistical tests or measures. They provide a preliminary feature selection process that can be further refined with other methods.

For categorical features we use a *chi-squared test* that evaluates the independence between features and the target with the default alpha=0.05. *Spearman rank correlation* with the threshold=0.5 is employed to assess monotonic relationships between our metric features and the target, capturing both linear and non-linear associations. To examine the relationship between a binary target and a continuous variable, *point biserial correlation* (a specialized form of Pearson correlation) is applied. Note that we apply *point biserial correlation* only during the binary classification task. Lastly, *mutual information* quantifies the amount of information one variable shares with another, effectively capturing all types of dependencies, including non-linear ones. Collectively, these methods should offer a robust and comprehensive approach to filtering out irrelevant or redundant features based on their statistical relationships with the target variable.

Wrapper-Based methods

This family of methods include assessing the importance of features using specific machine learning algorithms and help us evaluate the impact of different subsets of features on the model's performance.

We use *Recursive Feature Elimination with Cross Validation (RFECV)* with a Logistic Regression as a base estimator. It identifies the optimal number of features that yields the best performance during the

cross-validation. It provides an insight into which features contribute the most to predicting the target variable and how many features are needed before the model's performance plateaus or decreases.

Embedded methods

Embedded methods incorporate feature selection as part of the learning algorithm. For this purpose we implement *Lasso Regression with Cross-Validation*. Lasso Regression is effective for feature selection as it can reduce the coefficients of less significant features to zero, thereby implicitly selecting the most important features.

Binary Classification

Task Pipeline

We consider using 4 models for the binary classification task: *Logistic Regression, HistGradientBoostingClassifier, Random Forest Classifier and Voting Classifier.*

Validation of the models is performed with StratifiedKFold Cross-Validation with 5 folds in order to preserve the percentage of samples for each class.

These models are based on totally different principles. Thus, for being able to work and perform at their best they require different approaches for final preprocessing of features, especially categorical ones. So, for the sake of convenience and free-error process we create two major final-preprocessors, separately for Logistic Regression and Tree-Based algorithms.

To evaluate models' performances in a binary classification task we use f1 score, balanced accuracy and 'area under the ROC curve' (ROC AUC) score.

We start with a 'simple' *Logistic Regression algorithm. Logistic Regression* can't natively cope with categorical features. That's why we need to transform them either with one hot encoding, target encoding, or both according to features' cardinality (number of unique values).

Another requirement is that Logistic Regression can't work with NaNs or missing values, so we use *SimpleImputer* to fill such values with the mean. Moreover, *Logistic Regression* is sensitive to differences in feature scales, thus we apply *MinMax* scaling to the metric features (outliers were already removed on the previous steps). We've also noticed that for this dataset we get slightly better results if we apply *Log (np.log1p) Transformation* to metric features.

A few words about categorical features encoders mentioned a moment before:

One-Hot Encoding transforms categorical variables into binary vectors, representing each category as a separate column with binary values (0 or 1). This method, suitable for nominal categories, can significantly increase the dataset's dimensionality but avoids any implicit ordinal relationship.

Target Encoding replaces categorical values with the mean of the target variable for each category. It's effective for capturing more information in a feature and doesn't expand the feature space like *One-Hot*

Encoding. However, it can lead to overfitting and potential target information leakage, requiring careful implementation and validation.

The result of *Logistic Regression* with different preprocessing and setup is shown in [**Table 8**].

Then we try a more sophisticated *Hist Gradient Boosting Classifier*. It natively supports dealing with NaNs or missing values. As a tree-based algorithm it is insensitive to metric features scales. We can also give a hint to this algorithm which features consider categorical ones. The only technical requirement is that cardinality should be less than 255.

A Tree-Based algorithms preprocessor encodes categorical features consisting of features of low and high cardinality either with ordinal encoding, target encoding or both.

Above-mentioned *Ordinal Encoding* is a technique used to convert categorical variables into numerical values, where the categories are ordered from 0 to max_cat in a decreasing order of their frequencies. Infrequent categories are mapped to one category.

Metric features are passed to a Tree-Based algorithms preprocessor as they are.

The result of *Hist Gradient Boosting Classifier* with different preprocessing and setup is shown in [**Table 9**].

Subsequently, we implement a variety of feature selection methods to identify the most pertinent features for our models. With the refined feature set, we retrain *Logistic Regression and HistGradientBoostingClassifier* and analyze the outcomes [**Table 10**].

Additionally, we train a *Random Forest Classifier.* This model can't natively deal with missing or NaN values. So, we use keepna=False mode of our Tree-Based algorithm preprocessor to convert NaNs or missing values into special "NaN2Missing" values in categorical features. For metric features we replace NaNs or missing values with mean.

It's not straightforward which hyperparameters (tree max_depth and amount of pruning) might lead us to a higher f1 score using this algorithm, that's why we conduct hyperparameter tuning using *GridSearch with Cross-Validation.* [**Table 11**].

Finally, we train a *Voting Classifier* consisting of two *HistGradientBoosting Classifiers* with 2 different and equally best setups and a best tuned *Random Forest Classifier* [**Table 12**].

Findings

We got the results quite opposite of what we were expecting. Comparing minimal preprocessing and hard preprocessing with feature engineering while training *Logistic Regression* and *HistGradientBoosting Classifier* with different encoders, we can conclude that minimal preprocessing in both cases led us to a

slightly higher f1 score. Moreover, both models performed even worse on a feature set obtained as a result of the feature selection process.

One more interesting insight we got while training *HistGradientBoosting Classifier.* If we use target encoder for both low cardinal and high cardinal categorical features specifying low cardinal categorical features as categorical features inside *HistGradientBoosting Classifier,* we get very low results of f1 score (comparing with other setups on both minimal and hard preprocessing).

Also we can conclude that best results we obtain on *HistGradientBoosting Classifier* in two different setups. First, if we use ordinal encoder for low cardinal categorical features (specifying them as categorical features inside *HistGradientBoosting Classifier)* and target encoder for high cardinal categorical features. Second, using only ordinal encoder for all categorical features can give us almost the same result (without looking at the target at all). But only if we don't treat high cardinal features as categorical features. So, the grouping by target mean gives similar performance as grouping by just frequency.

One more thing to mention is that we use class_weight = {0: 0.14, 1: 1} instead of "balanced" in all our models. We proved it practically, that models' performance can benefit from changing class_weight hyperparameter from "balanced" to {0: 0.14, 1: 1} giving a bit more focus to class 0 (Not Readmitted) [**Table 13**]**.**

Multiclass Classification

For multiclass classification purposes we divide our train dataset into train and test sets. We validate models performance using StratifiedKFold cross validation.
We check performance of *Logistic Regression as a baseline and* those of our models that gave us best results during binary classification: *HistGradientBoosting Classifier and Random Forest Classifier.*
We can not use *Target Encoding* in a straightforward way for multiclass classification. So, the set of setups of our models is smaller than in binary classification: we check them only with ordinal encoding for all categorical features.
As in the case with binary classification, a small shift of focus from the readmitted in 30 days class to readmitted later or never classes also gives us small improvement. We use weights: {0: 0.23, 1: 0.34, 2: 1}, instead of "balanced".
Because of the limitation with *Target Encoding* this time only *HistGradientBoosting Classifier* becomes unconditional leader. So, we add 3 copies of it (with different random_state) to the *Voting Classifier.* Just as with the binary classification case, it also gives an extra small increase in performance. Results can be found in the table [Table 14]
We evaluate our best model - *Voting Classifier*, based on the *HistGradientBoosting Classifiers* on the test part of our train dataset, not shown during fitting. And the results correspond with the results obtained during cross-validation, as shown in the table [Table 15].
We also checked hard preprocessing with the best setup of *HistGradientBoosting Classifier.* And the results are the same, as in the binary classification part: it decreases the performance.

Conclusion

The findings of this project did not align with the initial expectations, as it was anticipated that feature engineering and hard preprocessing would lead to improved model performance. However, minimal preprocessing proved to be more efficient for the chosen models with particular setups. The reasons for that can stem from the fact that our hard preprocessing was too tough leading to the information loss while doing, e.g. different types of mappings for categorical features.

One of the limitations of this work include the usage of vanilla scikit-learn algorithms only. Even with the most powerful algorithms, e.g. *HistGradientBoosting Classifier* that can cope easily with different types of features, missing values it was difficult to get an f1 score of more than 0.3. With the help of identified patterns from encounter_ids we were able to get an f1 score of almost 0.72 on Kaggle. But again using that feature has limitations described in the corresponding part.

References

Shuwen Wang, X. Z. (n.d.). *Predictive Modeling of Hospital Readmission:.* Retrieved from https://arxiv.org/pdf/2106.08488.pdf

Goudjerkan, T. a. (n.d.). *Predicting 30-day hospital readmission for diabetes patients using multilayer perceptron.* Retrieved from https://researchonline.ljmu.ac.uk/id/eprint/11685/1/Paper_36-Predicting_30_Day_Hospital_Readmission_for_Diabetes_Patients.pdf

wikipedia. (n.d.). *List of ICD-9 codes*. Retrieved from wikipedia: https://en.wikipedia.org/wiki/List_of_ICD-9_codes

Yinan Huang, A. T. (n.d.). *Application of machine learning in predicting hospital readmissions.* Retrieved from https://link.springer.com/content/pdf/10.1186/s12874-021-01284-z.pdf

Annexes

	count	unique	top	freq
country	71236	1	USA	71236
race	67682	6	Caucasian	50693
gender	71236	3	Female	38228
age	67679	10	[70-80)	17359
weight	71236	10	?	68990
payer_code	71236	18	?	28201
admission_type	67530	7	Emergency	37742
medical_specialty	71236	69	?	34922
discharge_disposition	68646	25	Discharged to home	42256
admission_source	66518	16	Emergency Room	40319
primary_diagnosis	71236	687	428	4776
secondary_diagnosis	71236	699	276	4694
additional_diagnosis	71236	747	250	8070
glucose_test_result	3688	3	Norm	1806
a1c_test_result	11916	3	>8	5705
change_in_meds_during_hospitalization	71236	2	No	38326
prescribed_diabetes_meds	71236	2	Yes	54890
medication	71236	303	['insulin']	21715
readmitted_binary	71236	2	No	63286
readmitted_multiclass	71236	3	No	38405

Table 1

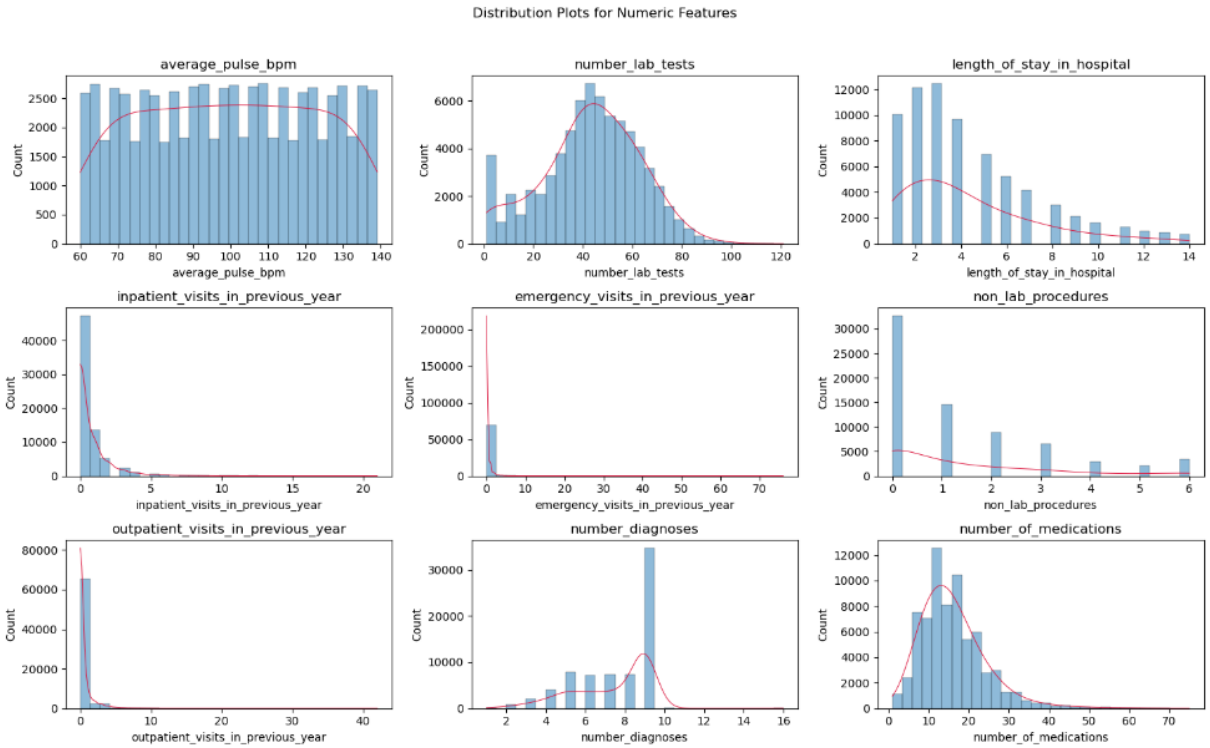

Figure 1

	count	mean	std	min	25%	50%	75%	max
encounter_id	71236	551634.3	259387	100011	326583	552684	775671	999979
patient_id	71236	54302279	38795850	135	23396510	45305631	87558374	1.9E+08
outpatient_visits_in_previous_year	71236	0.369588	1.287469	0	0	0	0	42
emergency_visits_in_previous_year	71236	0.196249	0.910854	0	0	0	0	76
inpatient_visits_in_previous_year	71236	0.640154	1.267271	0	0	0	1	21
average_pulse_bpm	71236	99.48662	23.0988	60	79	100	119	139
length_of_stay_in_hospital	71236	4.391024	2.988739	1	2	4	6	14
number_lab_tests	71236	43.09565	19.64292	1	31	44	57	121
non_lab_procedures	71236	1.340923	1.706664	0	0	1	2	6
number_of_medications	71236	15.99545	8.122347	1	10	15	20	75
number_diagnoses	71236	7.421023	1.937809	1	6	8	9	16

Table 2

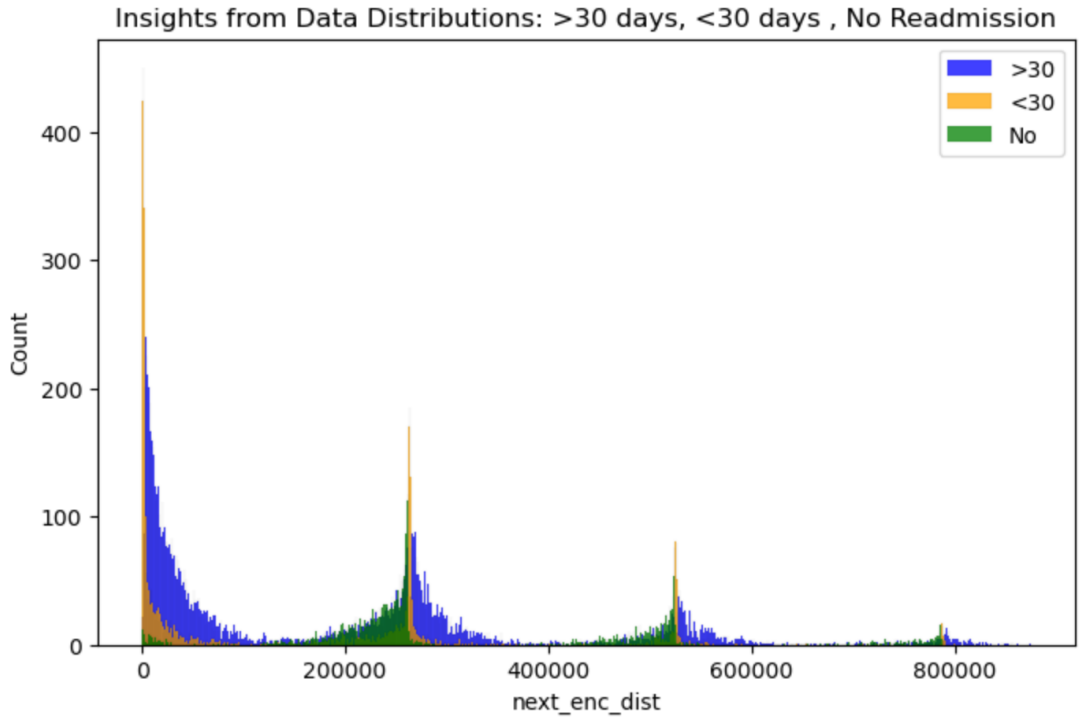

Figure 2

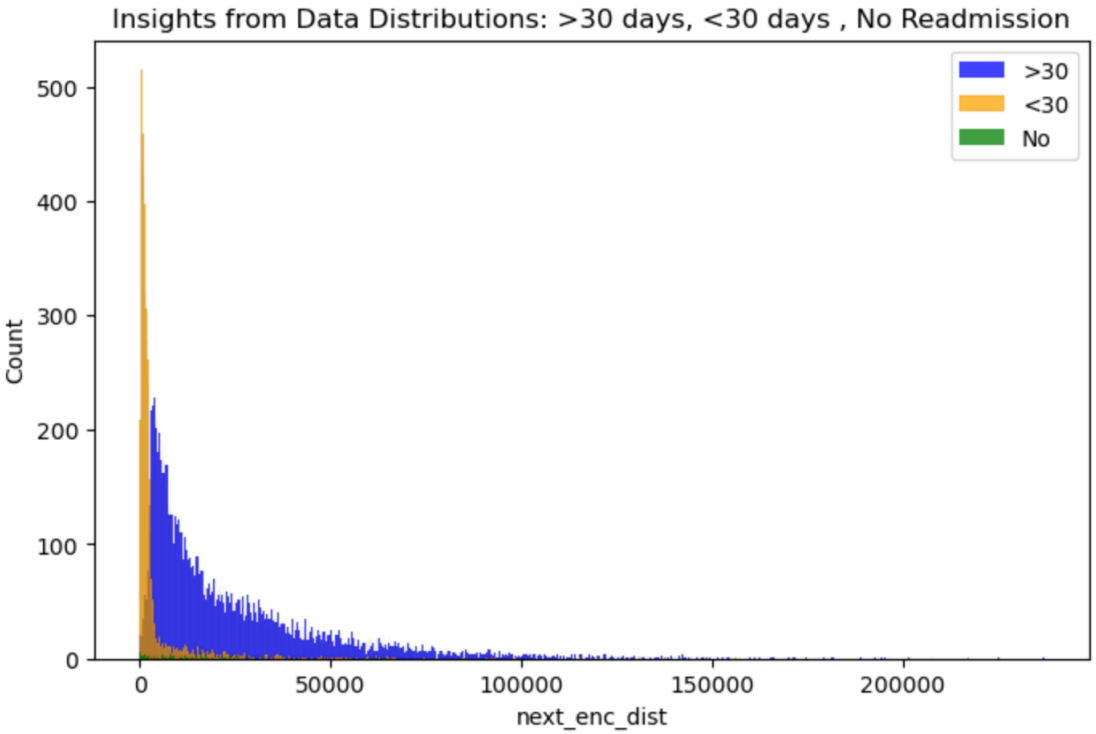

Figure 3

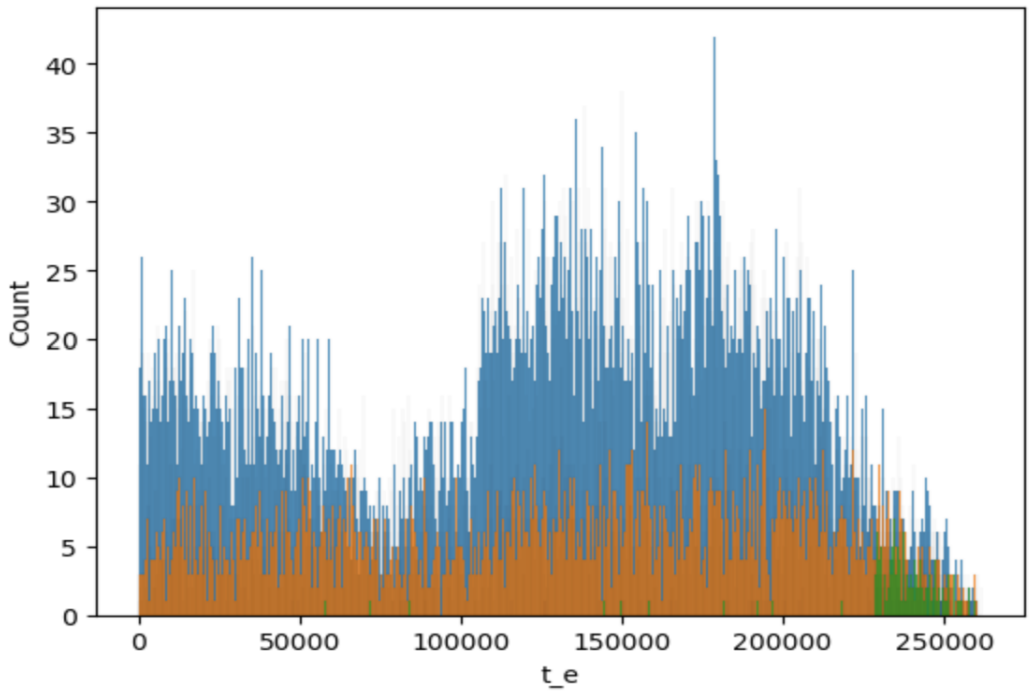

Figure 4

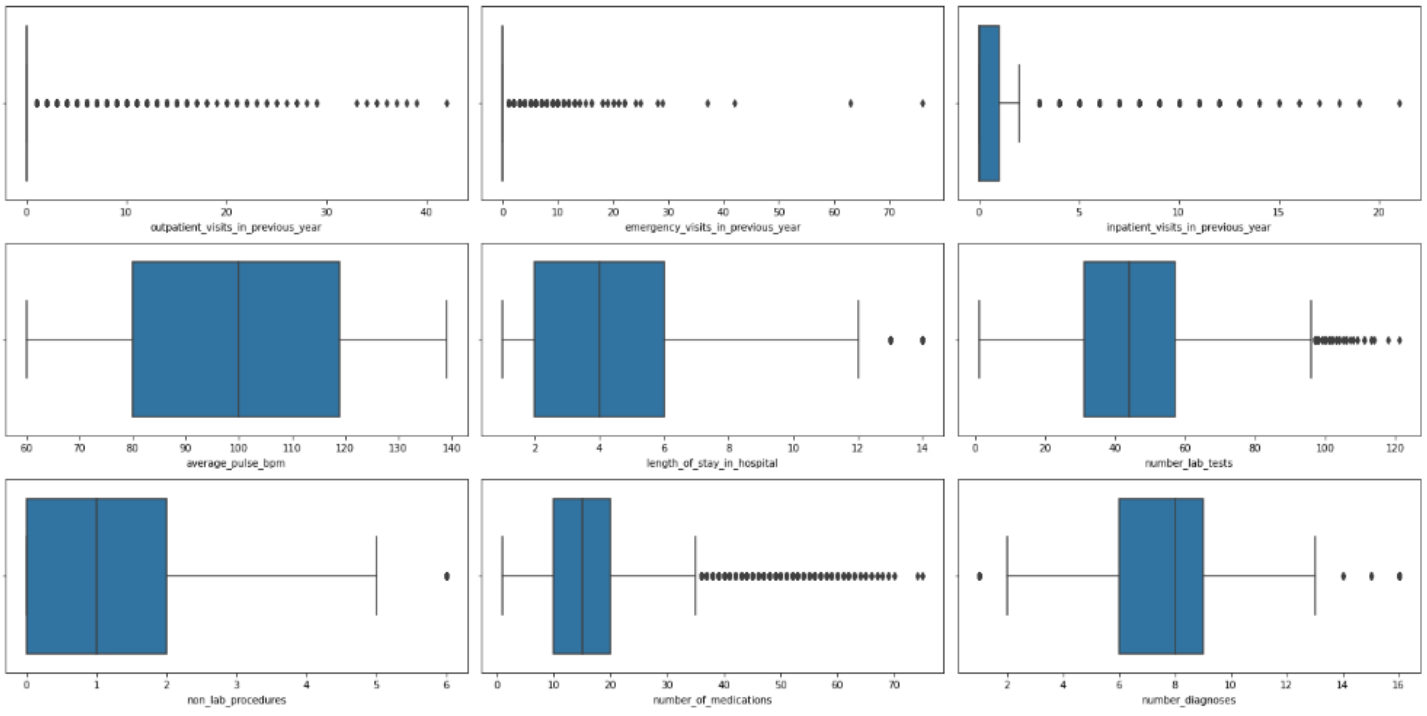

Figure 5

Feature	Old Value	New Value
change_in_meds_during_hospitalization	Ch	1
	No	0
prescribed_diabetes_meds	Yes	1
	No	0
readmitted_binary	Yes	1
	No	0
readmitted_multiclass	<30 days	2
	>30 days	1
	No	0
glucose_test_result	>300	2
	>200	1
	Norm	0
a1c_test_result	>8	2
	>7	1
	Norm	0

Table 3

Feature	Old Value	New Value
admission_type	Emergency	Emergency
	Elective	Elective
	Urgent	Urgent
	Newborn	Newborn
	Trauma Center	Trauma Center
	Not Available	missing
	Not Mapped	missing
admission_source	Not Available	missing
	Not Mapped	missing
	Emergency Room	Emergency Room
	Physician Referral	Physician Referral
	Clinic Referral	Clinic Referral
	Transfer from a hospital	Transfer from another medical facility
	Transfer from another health care facility	Transfer from another medical facility
	Transfer from a Skilled Nursing Facility (SNF)	Transfer from another medical facility
	Transfer from hospital inpt/same fac reslt in a sep claim	Transfer from another medical facility
	Transfer from critial access hospital	Transfer from another medical facility
	Transfer from Ambulatory Surgery Center	Transfer from another medical facility
	Extramural Birth	rare
	Normal Delivery	rare
	Sick Baby	rare
	HMO Referral	rare
age	[0-10)	0
	[10-20)	1
	[20-30)	2
	[30-40)	3
	[40-50)	4
	[50-60)	5
	[60-70)	6
	[70-80)	7
	[80-90)	8
	[90-100)	9

Table 4

Feature	Old Value	New Value
primary_diagnosis , secondary_diagnosis, and additional_diagnosis	1 <= icd9 <= 139	Infectious and Parasitic Diseases
	140 <= icd9 <= 239	Neoplasms
	240 <= icd9 <= 279	Endocrine, Nutritional, and Metabolic Diseases, and Immunity Disorders
	280 <= icd9 <= 289	Diseases of the Blood and Blood-Forming Organs
	290 <= icd9 <= 319	Mental Disorder
	320 <= icd9 <= 389	Diseases of the Nervous System and Sense Organs
	390 <= icd9 <= 459	Diseases of the Circulatory System
	460 <= icd9 <= 519	Diseases of the Respiratory System
	520 <= icd9 <= 579	Diseases of the Digestive System
	580 <= icd9 <= 629	Diseases of the Genitourinary System
	630 <= icd9 <= 679	Complications of Pregnancy, Childbirth, and the Puerperium
	680 <= icd9 <= 709	Diseases of the Skin and Subcutaneous Tissue
	710 <= icd9 <= 739	Diseases of the Musculoskeletal System and Connective Tissue
	740 <= icd9 <= 759	Congenital Anomalies
	760 <= icd9 <= 779	Certain Conditions Originating in the Perinatal Period
	780 <= icd9 <= 799	Symptoms, Signs, and Ill-Defined Conditions
	800 <= icd9 <= 999	Injury and Poisoning
	icd9 < 1 , icd9 > 999	Other

Table 5

Feature	Old Value	New Value
discharge_disposition	Discharged to home	Home Discharge
	Discharged/transferred to home under care of Home IV provider	Home Discharge with Health Service
	Discharged/transferred to home with home health service	Home Discharge with Health Service
	Discharged/transferred to SNF	Facility Transfer
	Discharged/transferred to another short term hospital	Facility Transfer
	Discharged/transferred to another rehab fac including rehab units of a hospital	Facility Transfer
	Discharged/transferred to another type of inpatient care institution	Facility Transfer
	Discharged/transferred to a federal health care facility.	Facility Transfer
	Discharged/transferred to ICF	Facility Transfer
	Neonate discharged to another hospital for neonatal aftercare	Facility Transfer
	Discharged/transferred to a long term care hospital	Facility Transfer
	Discharged/transferred/referred to a psychiatric hospital of psychiatric distinct part unit of a hospital	Facility Transfer
	Discharged/transferred within this institution to Medicare approved swing bed	Facility Transfer
	Discharged/transferred to a nursing facility certified under Medicaid but not certified under Medicare	Facility Transfer
	Admitted as an inpatient to this hospital	Other
	Not Mapped	Other
	Left AMA	Other
	Discharged/transferred/referred to this institution for outpatient services	Outpatient Services
	Discharged/transferred/referred another institution for outpatient services	Outpatient Services
	Still patient or expected to return for outpatient services	Outpatient Services
	Hospice / medical facility	Hospice
	Hospice / home	Hospice
	Expired in a medical facility. Medicaid only, hospice.	Expired
	Expired	Expired
	Expired at home. Medicaid only, hospice.	Expired

Table 6

medical_specialty	Surgery-General	Surgery
	Surgery-Thoracic	Surgery
	Surgery-Vascular	Surgery
	Surgery-Neuro	Surgery
	Surgery-Cardiovascular/Thoracic	Surgery
	Surgery-Maxillofacial	Surgery
	Surgery-Cardiovascular	Surgery
	Surgery-Plastic	Surgery
	SurgicalSpecialty	Surgery
	Surgeon	Surgery
	Surgery-Colon&Rectal	Surgery
	Surgery-Pediatric	Surgery
	InternalMedicine	Internal Medicine
	Hospitalist	Internal Medicine
	Family/GeneralPractice	Primary Care
	Cardiology	Cardiology
	Cardiology-Pediatric	Cardiology
	Pediatrics	Pediatrics
	Pediatrics-Endocrinology	Pediatrics
	Pediatrics-CriticalCare	Pediatrics
	Pediatrics-Neurology	Pediatrics
	Pediatrics-Pulmonology	Pediatrics
	Pediatrics-EmergencyMedicine	Pediatrics
	Pediatrics-AllergyandImmunology	Pediatrics
	Pediatrics-Hematology-Oncology	Pediatrics
	Pediatrics-InfectiousDiseases	Pediatrics
	Gastroenterology	Gastroenterology
	Radiology	Radiology
	Radiologist	Radiology
	Orthopedics	Orthopedics
	Orthopedics-Reconstructive	Orthopedics
	Nephrology	Nephrology
	Emergency/Trauma	Emergency Medicine
	Psychiatry	Psychiatry
	Psychiatry-Child/Adolescent	Psychiatry
	Pulmonology	Pulmonology
	ObstetricsandGynecology	Obstetrics and Gynecology
	Gynecology	Obstetrics and Gynecology
	Obsterics&Gynecology-GynecologicOnco	Obstetrics and Gynecology
	Obstetrics and Gynecology	Obstetrics and Gynecology
	Urology	Urology
	Oncology	Oncology
	Hematology/Oncology	Oncology
	PhysicalMedicineandRehabilitation	Rehabilitation

Table 7 (continues next page)

medical_specialty	Neurology	Rare
	Neurophysiology	Rare
	Endocrinology-Metabolism	Rare
	Endocrinology	Rare
	Otolaryngology	Rare
	Podiatry	Rare
	PhysicianNotFound	Rare
	Osteopath	Rare
	AllergyandImmunology	Rare
	DCPTEAM	Rare
	Pathology	Rare
	Dentistry	Rare
	OutreachServices	Rare
	Speech	Rare
	SportsMedicine	Rare
	Proctology	Rare
	Resident	Rare
	Psychology	Rare
	Hematology	Rare
	Ophthalmology	Rare
	InfectiousDiseases	Rare
	Anesthesiology	Rare
	Anesthesiology-Pediatric	Rare
	Rheumatology	Rare

Table 7

		f1			balanced accuracy			roc auc		
		test max	test avg	train avg	test max	test avg	train avg	test max	test avg	train avg
Preprocessing	**Setup**									
Minimal	**low_card: one hot; high_card: target**	0.289302	0.281978	0.299949	0.631432	0.624001	0.642720	0.684620	0.680725	0.707005
	low_card: target; high_card: target	0.290129	0.282852	0.299385	0.630399	0.622794	0.639535	0.682261	0.679066	0.703531
	low_card: one hot; high_card: one hot	0.287047	0.280554	0.292383	0.630821	0.623472	0.635897	0.676557	0.675558	0.696103
Hard Preprocessing + Feature Engineering	**low_card: one hot; high_card: target**	0.287639	0.274670	0.288090	0.630494	0.616113	0.630667	0.682357	0.668025	0.687615
	low_card: target; high_card: target	0.288171	0.272275	0.283392	0.629695	0.612030	0.623668	0.677019	0.665514	0.681853
	low_card: one hot; high_card: one hot	0.285523	0.274476	0.285758	0.628432	0.615951	0.627969	0.682367	0.665393	0.682421

Table 8: Logistic Regression Results on Minimal and Hard Preprocessing

		f1			balanced accuracy			roc auc		
		test max	test avg	train avg	test max	test avg	train avg	test max	test avg	train avg
Preprocessing	Setup									
Minimal	low_card: ordinal->categorical; high_card: target	0.305181	0.296107	0.336668	0.647782	0.638302	0.680886	0.698237	0.694422	0.755965
	low_card: target->categorical; high_card: target	0.124932	0.036686	0.037716	0.529889	0.508716	0.509049	0.498599	0.458451	0.456224
	low_card: target; high_card: target	0.299145	0.292450	0.329083	0.641106	0.634274	0.672037	0.698039	0.692790	0.743830
	low_card: ordinal->categorical; high_card: ordinal	0.305147	0.295448	0.331227	0.644646	0.635062	0.670932	0.695228	0.691179	0.746595
	low_card: ordinal_max20->categorical; high_card: ordinal_max20->categorical	0.302161	0.293482	0.343517	0.640620	0.633627	0.684193	0.698723	0.689721	0.758227
	low_card: ordinal_max15->categorical; high_card: ordinal_max15->categorical	0.301165	0.293261	0.336784	0.641414	0.633415	0.677699	0.697613	0.690332	0.750266
Hard Preprocessing + Feature Engineering	low_card: ordinal->categorical; high_card: target	0.304711	0.289513	0.339116	0.644647	0.629597	0.682054	0.698427	0.684727	0.756625
	low_card: target->categorical; high_card: target	0.203613	0.203581	0.203581	0.500000	0.500000	0.500000	0.507740	0.544680	0.546760
	low_card: target; high_card: target	0.301563	0.285622	0.320618	0.644117	0.627083	0.664899	0.697047	0.681712	0.733600
	low_card: ordinal->categorical; high_card: ordinal	0.304612	0.289406	0.336315	0.644100	0.629139	0.678320	0.695197	0.681403	0.751948
	low_card: ordinal_max20->categorical; high_card: ordinal_max20->categorical	0.298271	0.289414	0.340132	0.637023	0.628969	0.681888	0.692663	0.680295	0.757036
	low_card: ordinal_max15->categorical; high_card: ordinal_max15->categorical	0.304450	0.288903	0.340642	0.643318	0.628110	0.682128	0.692891	0.680385	0.756766

Table 9: HistGradientBoosting Results on Minimal and Hard Preprocessing

		f1			balanced accuracy			roc auc		
		test max	test avg	train avg	test max	test avg	train avg	test max	test avg	train avg
Preprocessing	Setup									
[Logistic Regresion] Hard Preprocessing + Feature Engineering + Feature Elimination	low_card: ordinal->categorical; high_card: ordinal	0.287205	0.273182	0.286990	0.630043	0.614429	0.629513	0.682022	0.665744	0.684443
[HistBoost] Hard Preprocessing + Feature Engineering + Feature Elimination	low_card: ordinal->categorical; high_card: target	0.296076	0.279937	0.324759	0.636772	0.620805	0.669106	0.688092	0.673382	0.742032
	low_card: ordinal->categorical; high_card: ordinal	0.291185	0.278632	0.318382	0.632694	0.620054	0.663318	0.684752	0.669313	0.732103

Table 10: Logistic Regression and HistGradientBoosting Results with Selected Features

	mean_fit_time	mean_score_time	p__ccp_alpha	p__max_depth	mean_test_balanced_accuracy	mean_test_f1_score	mean_test_roc_auc	max_test_f1
0	11.940917	0.603164	0.0001	9	0.624121	0.292988	0.688347	0.297908
4	10.972632	0.551493	0.00015	9	0.624859	0.291392	0.688860	0.301993
8	11.063648	0.517130	0.0002	9	0.630299	0.295114	0.689710	0.301257
12	10.888598	0.508262	0.00025	9	0.630738	0.292881	0.689083	0.299048
1	12.705297	0.598336	0.0001	10	0.621272	0.293768	0.688871	0.298029
5	12.964232	0.611698	0.00015	10	0.623868	0.293382	0.690455	0.301594
9	13.198853	0.608425	0.0002	10	0.628707	0.295343	0.690887	0.301721
13	12.158508	0.538402	0.00025	10	0.629198	0.292168	0.689911	0.301108
2	13.264609	0.609103	0.0001	11	0.615099	0.291707	0.689129	0.296152
6	14.644383	0.614959	0.00015	11	0.622548	0.295517	0.689949	0.303753
10	14.144627	0.586976	0.0002	11	0.627430	0.295620	0.691960	0.302411
14	13.506212	0.549604	0.00025	11	0.630730	0.294674	0.690290	0.302485
3	14.672791	0.719648	0.0001	12	0.609377	0.289299	0.689127	0.294872
7	15.371581	0.619303	0.00015	12	0.615159	0.290597	0.690236	0.294850
11	14.845317	0.581468	0.0002	12	0.624446	0.294577	0.691502	0.299742
15	15.329561	0.569971	0.00025	12	0.628835	0.293320	0.690428	0.300664

Table 11: GridSearchCV results of Random Forest Classifier

	f1			balanced accuracy			roc auc		
	test max	test avg	train avg	test max	test avg	train avg	test max	test avg	train avg
0	0.305213	0.296984	0.34474	0.642368	0.633883	0.680025	0.699135	0.696539	0.758275

Table 12: Voting Classifier Results

		f1			balanced accuracy			roc auc		
		test max	test avg	train avg	test max	test avg	train avg	test max	test avg	train avg
Preprocessing	Setup									
Minimal	Logistic Reg: {0:0.14, 1: 1}	0.287047	0.280554	0.292383	0.630821	0.623472	0.635897	0.676557	0.675558	0.696103
	Logistic Reg: balanced	0.279695	0.274867	0.287091	0.630742	0.624462	0.638980	0.679984	0.675085	0.696169
	HistBoost: {0:0.14, 1: 1}	0.305147	0.295448	0.331227	0.644646	0.635062	0.670932	0.695228	0.691179	0.746595
	HistBoost: balanced	0.295843	0.290402	0.325613	0.647824	0.641407	0.682852	0.699214	0.694755	0.752957

Table 13: Class Weight

		f1			balanced accuracy			roc auc		
		test max	test avg	train avg	test max	test avg	train avg	test max	test avg	train avg
Preprocessing	Setup									
Minimal	[HistBoost] Weights: balanced; low_card: ordinal->categorical; high_card: ordinal	0.555123	0.550860	0.595105	0.498336	0.496395	0.569816	0.701000	0.696516	0.761385
	[HistBoost] Weights {0: 0.23, 1: 0.34, 2: 1}; low_card: ordinal->categorical; high_card: ordinal	0.566783	0.562071	0.607168	0.501168	0.498342	0.569477	0.702891	0.698558	0.764712
	[Random Forest] Weights {0: 0.23, 1: 0.34, 2: 1}; low_card: ordinal->categorical; high_card: ordinal	0.545038	0.541932	0.574829	0.473338	0.471128	0.528880	0.679538	0.677936	0.722780
	[Logistic Regression] Weights {0: 0.23, 1: 0.34, 2: 1}; all=one_hot	0.542473	0.536321	0.547584	0.478650	0.474976	0.493066	0.679273	0.675121	0.693768
	[Voting on 3 similar HistBoost]; Weights {0: 0.23, 1: 0.34, 2: 1}; low_card: ordinal->categorical; high_card: ordinal	0.566416	0.563520	0.609894	0.502181	0.498916	0.572577	0.698039	0.700244	0.767595

Table 14: Multiclass Classification Results on CV

	f1	balanced accuracy	roc auc
Y_test	0.557584	0.497893	0.696684

Table 15 : Best f1 score obtained on Multiclass with Voting Classifier