PROJECT REPORT

TEAM DETAILS:
ASHISH CHOUDHARY : 28929337
RATHNAINTHRI HANNAH DEVANAND: 64207299

1.INTRODUCTION

We aim to predict if a diabetes patient would be readmitted to a hospital within 30 days,after 30 days or if they were readmitted at all.In general , the dataset provides data related to demographics , diagnoses ,medications and in-hospital procedures.

2.FEATURE ENGINEERING

We focused on cleaning the features which had missing values in them and also bringing all the features on to the same scale. For example, categorical data in the form of strings was converted to an integer value using map-based encoding. Below we have listed in detail the preprocessing we performed based on **missing values, recoding and rescaling.**

2.1 PREPROCESSING BASED ON MISSING VALUES:

The following columns were dropped because they had missing data. The reason for each is listed below.

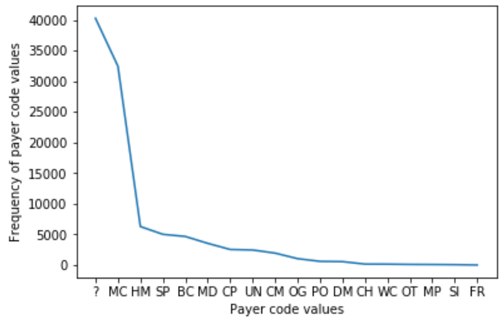


Fig 1: Payer code frequency

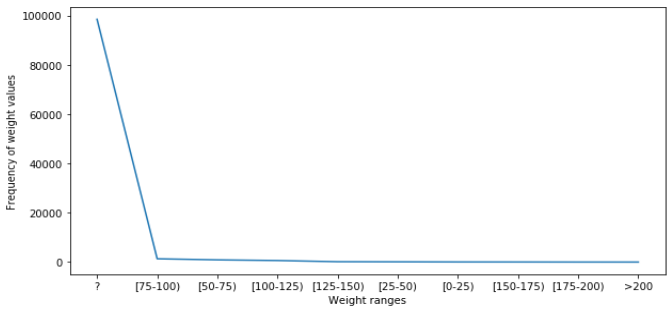


Fig 2: Weight frequency

- **Payer_code**: Indicated how the patient paid(through insurance etc) .50% of the records were missing as indicated by Fig 1.
- **Weigh**t:Indicated the weight of the patient. 90 % rows of this column had no values in them as indicated by Fig 2.
- **Medical Speciality**: This indicated to which speciality the patients went to after their admission. It had many distinct values and 50% missing records.
- **Diag_2 and Diag_3** : Indicated secondary and tertiary diagnoses and contained 923 and 954 distinct values, so this column was dropped.

- **Patient_nbr**: This indicated the patient's id and seemed irrelevant to the prediction of a readmission of a patient.
- **Citoglipton, Examide**: Indicated a medication provided to a patient.They had identical values and so it did not seem to have any bearing on the prediction of readmission.

2.2 PREPROCESSING BASED ON RECODING:

2.2.1.A1C HAEMOGLOBIN TEST RESULT:

The values of this column indicate the haemoglobin A1C levels of the patient. From the original research paper we gathered that this was an important measure of the effectiveness of the treatment. If during readmission , the value is within the normal range , it means that the current therapy is working.The range of values are :

> 7 and >8: Current therapy not working.

< 7 , Normal Value: 0

The cells with values > 7 and > 8 are grouped together and replaced with the value 1.

2.2.2.MAX_GLU_SERUM LEVELS TEST RESULT:

The values of this column indicate the max_glu_serum levels of the patient. Similar to A1C haemoglobin , this was an important measure of the effectiveness of the patient.

> 200 and >300 : Current therapy not working.

< 200 , Normal Value: 0

The cells with values > 200 and >300 are grouped together and replaced with the value 1

2.2.3.AGE:

Indicated the age of the patient within a range for example : [0-10], so we replaced these column values with a single value indicating the upper limit.

2.2.4.RACE:

Indicated the race of the patient. Was originally string based data. Based on the 5 distinct values,we then changed the column values from String based categorical values to integers using Map Based Encoding .

2.2.5.MEDICINE-RELATED RECODING:

In our dataset, there were medicine related columns which indicated if the medicine was provided to a patient or not. The values of these columns were:

- **No** : indicating that the medicine was not provided.
- **Up**: Indicating that dosage amount was increased.
- **Down**: indicating the dosage amount was decreased.
- **Steady**: indicating the dosage amount was maintained.

2.2.5.1 : ADDITION OF NEW FEATURE : NUM_CHANGE:

Using these values , we came up with a new feature indicating the number of dosage changes provided to the patient by studying the Up/Down values across these features. We name this new feature as num_change.

2.2.5.2 : ADDITION OF NEW FEATURE : NUM_MEDS:

In the original columns , the values ' Up' ,'Down','Steady' are replaced by 1 and 'No' was replaced by 0, indicating whether the medicine was given to the patient or not.
Using this replacement , we then created a new column named num_meds which indicated the number of medicines provided to a patient,encapsulating the changes in medication to the patient in 2 columns.

2.3.PREPROCESSING OF PRIMARY DIAGNOSIS(DIAG_1):

In the original dataset, the primary , secondary and tertiary had a wide range of unique values. We decided to take into account only the primary diagnosis. The values fall between various ranges. On looking at the ranges we decided to use each range to indicate a particular primary diagnoses. Each range was re-coded to an integer value. For example

- *Code 1:Circulatory*

  ......
- *Code 0:Others*

2.4. RESCALING OF DATA:

Since classification data perform better when all the data values are on one scale, using scikit learn's Standard Scaler, we have attempted to bring the data onto one scale.

2.5. Final Data Size

Training	**Validation**	**Testing**
`(80493, 43)`	`(8959, 43)`	`(9888, 43)`

3. CLASSIFICATION MODELS

3.1: LOGISTIC REGRESSION:

This was our example of a simple model. Since our classification problem was a multinomial class and large dataset, we used a 'SAGA' based solver with L1 penalty. The parameters which we worked with are :

- **Penalty:L1, Solver: saga**

- **Multiclass: Multinomial, Max_iterations : 300**

OBSERVATIONS & ANALYSIS:

- In our first iteration , we provided the model with both saga and sag solvers. We achieved better accuracy scores with saga and hence maintained that as our solver.
- We then used the same solver and retrained the model with both L1 and L2 penalty. We saw better accuracy results with L1 penalty.
- Setting the bias=true/false did not have any bearing on our predictions.

3.2: SUPPORT VECTOR MACHINE (SVM):

This was another example of our base model. In general we played around with the hyperparameters kernel, weight , decision_shape_function and max_iterations. The parameters we worked with are

- **Kernel: Sigmoid, Weight: balanced**
- **Max_iterations:400, Decision_shape_function: ovo**

OBSERVATIONS & ANALYSIS:

- In our first iteration we worked around with the kernel function of poly and rbf. Both of these kernel functions gave very low results with respect to accuracy.
- In our second iteration we used a combination of kernel function sigmoid and weight set to balanced. We chose our weight=balanced since our data samples are limited with respect to class 2(readmission greater than 30). We then chose the decision_shape_function as ovo since our problem was a multiclass classification.
- In our third iteration, we added the max_iterations=200, on increasing the number of iterations to 400, we saw an increase in accuracy.

3.3. COMPLEX MODEL: RANDOM FOREST CLASSIFIER :

A random forest is an estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

For our project, we ran a gridsearch on a list of ranges for these 6 parameters:

- **N_Estimators, Min_samples_leaf**
- **Split, Max_depth**
- **Max_features, Min_samples_leaf**

On running grid search , we chose the following 5 parameters , since they seemed to have the most effect on the accuracy:

- **N_Estimators, Max_Depth**
- **Min_Samples_Split, Max_features**

- **Max_leaf_nodes**

OBSERVATIONS & ANALYSIS:

- Initially we started out with a **max depth of 30,min_sample split set to 0.1 and no_of_estimators set to 10**. Increasing the depth of the tree above 30 , did not make any difference to our accuracy results and in fact led to overfitting. Whereas the number of estimators (number of trees) improved our accuracy results.
- In our second iteration, we added the parameter **max_features=sqrt,** (choosing only number of features=square root). At this point we observed over fitting.
- In our third iteration, we then added the **max_leaf_nodes=5 ,** which then reduced the overfitting and achieved a stable accuracy score of ~53 %
- The figure below indicates the most important features in our dataset.

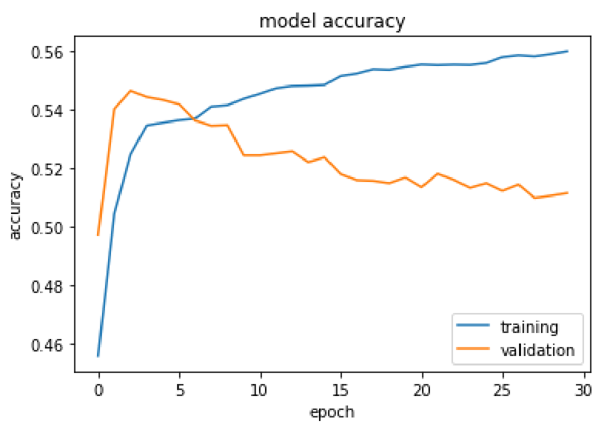
3.4. ADVANCED MODEL: NEURAL NETWORK CLASSIFIER:

A simple neural network endeavors to find the underlying relationship between the various features in a given dataset by mimicking human behavior. For this project we worked with the activation_function, number of layers and number of nodes in each layer.

- **Activation function:Relu**
- **Number of layers: 3**
- **Number of nodes: 64**

OBSERVATIONS & ANALYSIS:

- In our first iteration we chose a layer size =3 , number of nodes=32 and activation function = logistic.
- In our second iteration to gain better accuracy , we changed our activation to sigmoid and observed better accuracy.
- In our third iteration , we changed the number of nodes to 64, activation function=relu and again saw an increase in accuracy. This combination gave us a stable model.
- Finally we used batch normalization and dropout layer to further improve our results.
- We achieved a final accuracy of 51% for training and validation. The below plot shows the training and validation accuracy with respect to no of epochs trained.



4. ERROR ANALYSIS:

During the initial splitting phase ,the column 'readmitted' was re-coded to these values:**No :0,< 30 :1 and > 30 : 2**

4.1.1 : TRAIN-TEST-SPLIT CLASS SAMPLE SIZES

Dataset Type	**Class 0**	**Class 1**	**Class 2**
Training	42577	28753	9163
Validation	4798	3151	1010
Testing	5321	3457	1110

4.1.2. REASONS FOR MODEL SELECTION:

- Initially we started out with Naive Bayes and KNN models. With Naive Bayes we saw low accuracy of ~10 %. In the case of KNN, the time for execution was very long and also the accuracy was ~30 %. We decided not to go ahead with these models.
- In the case of complex models , we trained Xgboost and CNN on our dataset. Both the classifiers were overfitting.
- We also trained an ensemble classifier of models individually giving better accuracies , but it did not make an impact on the overall accuracy.

4.1.3. CONCLUSIONS

- After the train/test/split invocation it can be seen that the data samples for class 2(> 30) is very less for all datasets in the ratio 1:5. (Class 2: Class 0) & 1:3 (Class 2 & Class 1).

- By observing the confusion matrices, for all our models we observe that the models were not able to predict correctly class 2 since the amount of training data for class 2 is very less.
- According to our analysis, this was the main reason for observing low accuracy across all the models.

Model Used	**Training accuracy**	**Validation Accuracy**	**Test Accuracy**
Logistic Regression	56.8%	49.6%	48.6%
SVM	52.2%	46.05%	45.2%
Random Forest	54.4%	52.80%	51.8%
Neural Network	56%	51.1%	51%

5. REFERENCES:

- **https://medium.com/berkeleyischool/how-to-use-machine-learning-to-predict-hospital-readmissions-part-1-bd137cbdba07**
- **https://towardsdatascience.com/predicting-hospital-readmission-for-patients-with-diabetes-using-scikit-learn-a2e359b15f0**
- **https://www.hindawi.com/journals/bmri/2014/781670/**
- **Sklearn's Documentation for each model**

DISTRIBUTION OF WORK :

Rathnainthri Hannah Devanand: Logistic Regression and Random Forest
Ashish Choudhary: SVM and Neural Network

APPENDIX :

To run the project in local machine -

- Comment the first two cells which is to be used to run in colab.
- Make sure that the dataset is in the same location as the notebook.
- Run the Notebook.

** For your reference, the notebook also contains an extra commented code section which has the extra models we used for our experiments.

Libraries which needs to be installed -

- Pandas

- Numpy
- Matplotlib.pyplot
- Sklearn
- Keras
- Pickle