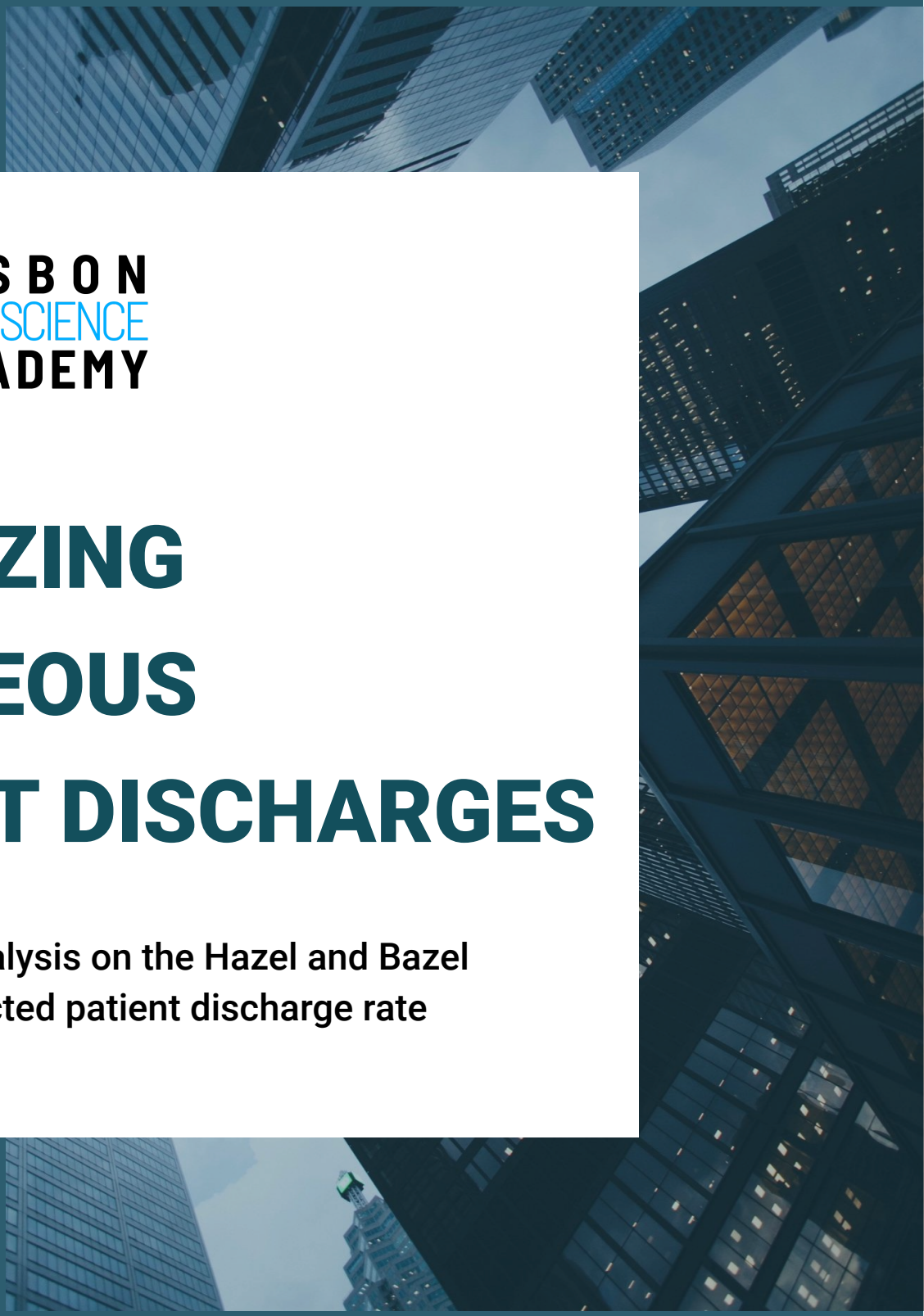


LISBON DATA SCIENCE ACADEMY

MINIMIZING ERRONEOUS PATIENT DISCHARGES

A data science analysis on the Hazel and Bazel Hospital's unexpected patient discharge rate

Prepared for:
Hazel and Bazel Hospital

Prepared by:
Rafael Gil
Data Scientist
Awkward Problem Solutions™

6 of March 2022

Contents

1 Business Conclusions

1.1 Summary

Awkward Problem Solutions™ provided a REST API to validate patient discharges at the Hazel and Bazel hospital. This was part of an effort to minimize the erroneous discharge rate the hospital was noticing. The service run in production for a whole week validating a total of 9694 medical encounters. Besides predicting whether a patient would be readmitted at the hospital within the following 30 days, the service also collected data for future iterations of the model, including the real outcome of each medical encounter, i.e. whether the patient was actually readmitted within a 30 day time-frame after discharge.

The service was stable throughout the whole week in production, requiring no interventions or updates. The application was able to reject malformatted requests, informing the end user of the error, and to pass valid data onto the predictive model. The predictive model itself handled all kinds of data successfully, by outputting a result at every request.

There was minor a issue regarding data gathering, causing 200 medical encounters not to receive their true label, i.e. whether that medical encounter resulted in readmission or not.

Regarding performance, the model did match the expectations regarding accuracy, but failed on other expectations and requirements. The model showed an 68 % accuracy, having correctly estimated the outcome of slightly more than 2 out of 3 medical encounters. However, it failed the minimum 50 % precision, having achieved only 18 % precision. This means that only 18 % of the medical encounters classified as a readmission were actually a readmission, in other words only 18 % of the patients were actually sick.

The present results indicate that out model may have overfitted to the training data, as warned in the precious report. An evidence of overfitting is that the real performance drastically dropped when compared to the expected performance. The unsatisfactory performance, could also have been related to a population shift, thus rendering the old data as obsolete, and consequently the model provided. However, a population analysis reveal no population shift, thus enhancing the overfitting probability.

Finally, not all discrimination requirements were met with the exception of gender and insurance status discrimination criteria, as expected in the previous report. Nonetheless, analogously to the initial training data, the collected data revealed multiple discrimination issues on many medical specialties across different subgroups.

2 Results Analysis

2.1 Model Performance

The model performance in production did not meet the expectations laid out by on the test set. As detailed in table 1, the accuracy is the only metric that was met, even though it is the least important of the set. Accuracy is the least important metric because the problem at hand is highly imbalanced. For instance, consider a model to distinguish dogs from cats where 99 % of the observations are dogs. Such model would have a 99 % accuracy if it always outputs the class dog. However, that model would not be any good at distinguishing dogs from cats, which is its main purpose. On the current scenario most of the observations are non-readmissions, thus the 68 % could have been achieved by mostly identifying the non-readmissions correctly.

Table 1: Model performance on the test set and also in production.

	Accuracy	Recall	Precision	F1 Score
Test set	68 %	79 %	65 %	0.71
Production	68 %	52 %	18 %	0.27

The main goal of the predictive model was to maximize recall, which is the ability of the model to find all the medical encounters that will be readmitted, i.e. to minimize the erroneous patient discharge rate. The expected recall of 79 % on the test set contrasts with the 52 % obtained in production. This result means that more than half of medical encounters that would result in a readmission were detected by the model. It is, nonetheless, a positive outcome even though there is room to improve in future iterations.

Furthermore, the result regarding precision is not enough to meet the minimum 50 % business requirement. The current results show that only 18 % of the medical encounters classified as readmissions were actually readmissions.

The production deployment allowed the collection of 9694 new medical encounters, and their true outcome is known for all but 200. These 200 medical encounters were discarded on the present analysis, since their readmission outcome is not known. Furthermore, it is worth noting that none of the collected medical encounters was already available at the training set, i.e. all the collected data is new.

2.2 Success on requirements

The first requirement regarding the model performance is that at least 50 % of the patients identified for readmission should actually be sick. This topic was mentioned in the previous section, since this requirement translates into a minimum 50 % precision score on the overall model. As mentioned this threshold was not met, the model achieved a precision score of 18 % according to table 1.

Nonetheless, there was one more business requirement stating that the wrongful discharge rate, or readmission rate, should not vary more than 10 percentage points between sub-groups, and less than 5 percentage points between medical specialties. As mentioned, the readmission rate can be measured using the precision score for each sub-group and medical specialty.

Figure 2 plots the precision score of the deployed predictive model for the sensitive sub-groups. Two sub-groups meet the criteria: gender, and insurance status with precision scores ranging a maximum of 0.94 % and 6.88 %, respectively. However, the criteria is not met for age, and race with with precision scores ranging a maximum of 26.53 % and 12.99 %, respectively. Nonetheless, the observed precision variance is lower than expected on report 1.

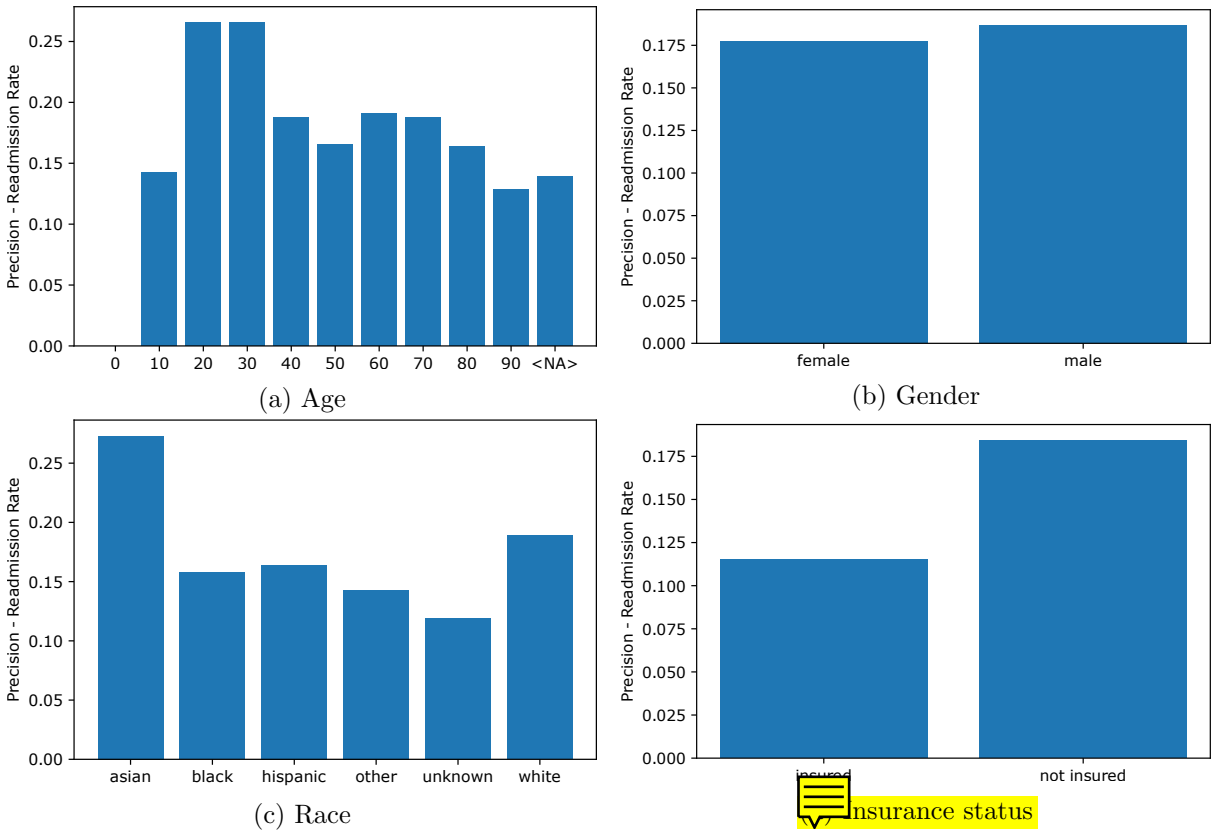


Figure 1: Model readmission rate for sensitive sub-groups

Regarding, the medical specialty the criteria is not met, since the precision scores ranges a maximum of 27.5 %. Once again, the observed precision variance, for the medical specialties, is lower than expected on report 1.

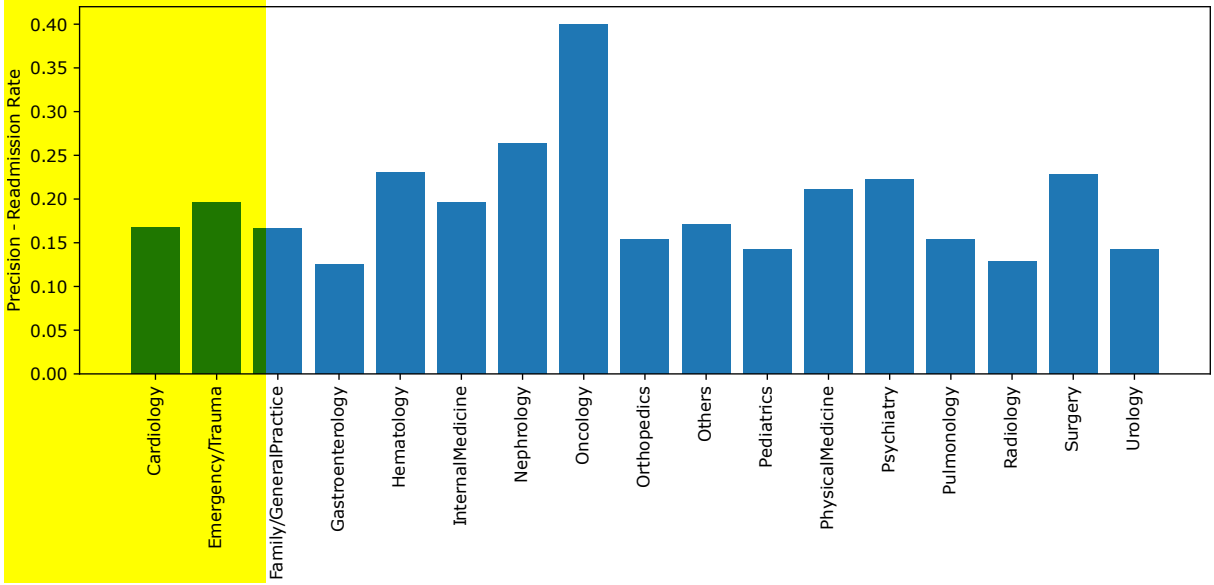


Figure 2: Model readmission rate for medical specialties

However, as also discussed on report 1, it is worth noting that the sub-groups where the

criteria failed are also the sub-groups with a higher number of categories. Empirically, spreading the existing medical encounters throughout a higher number of categories, increases the preponderance of those medical encounters. Thus, even though the criteria failed, immediate conclusions should not be drawn because a higher volume of data could prove otherwise.

2.3 Population Analysis

The new dataset has 50 different medical specialties. Analogously to the first dataset, roughly 50 % of the medical encounters are missing a medical specialty. As discussed in the previous report, `medical_specialty` had a high volume of unique classes, and to mitigate that the classes were merged into the main medical specialty category. Out of the 50 different medical specialties, only did not exist previously: *Surgery-PlasticwithinHeadandNeck*. Consequently, it was mapped into the *Others* category, even though it belong to the *Surgery* class. Thus, this new class was added to the pre-processing logic.

After this update to the medical specialties mapping, figure 3 was plotted with the distribution of medical encounters per medical specialty. The figure shows that the medical specialty distribution is consistent with the initial dataset. The top-7 medical specialties are exactly the same both in the initial dataset and in the current collected data from the REST API.

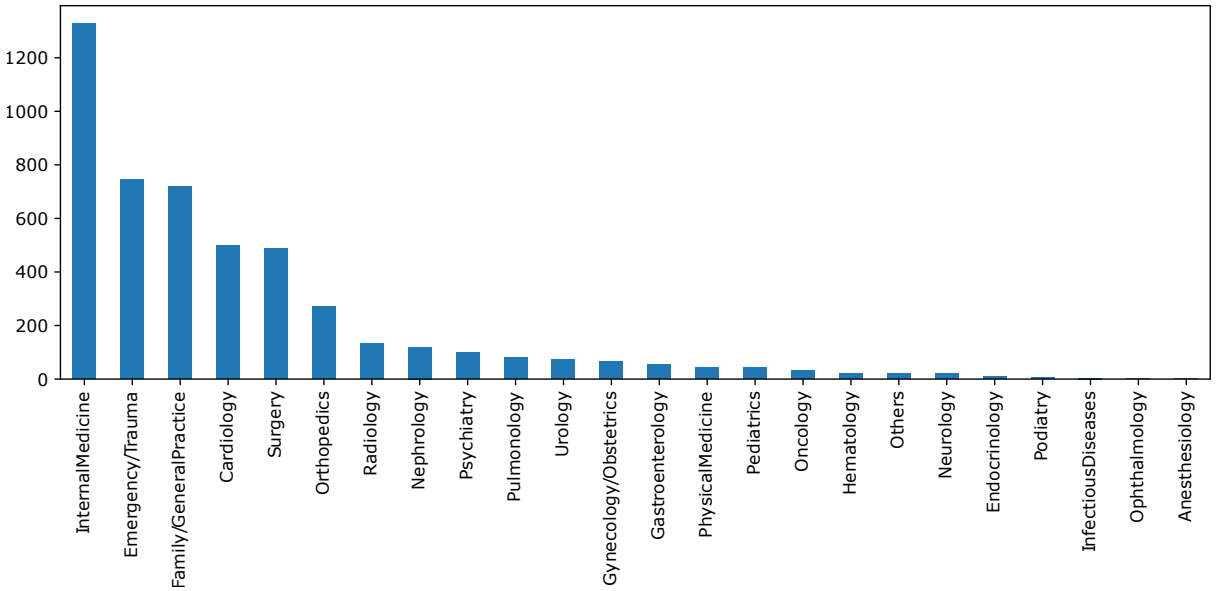
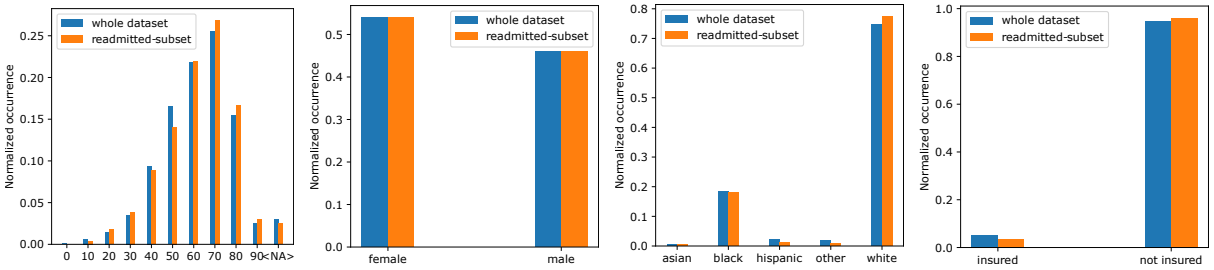


Figure 3: Number of medical encounters per medical specialty after processing

Analogously to the previous report, figure 4 plots the occurrence of each sensitive class on the new dataset, including age, gender, race, and insurance status. In blue its plotted the occurrence of each sensitive class on the whole dataset, whilst in orange its plotted the occurrence of the same sensitive class across the readmission-only subset.

The graphs on figure 4 are highly similar to the graphs from the previous reports. There are only slight variations, for instance, on figure 4c, the `black` race on the new data set have a slightly higher occurrence rate on the whole dataset when compared to the readmitted subset. Nonetheless, these slight variations are neglectable when looking at the overall distribution. The new data, extracted during the REST API production week, looks like a careful replication of the population from the initial dataset.

Furthermore, as suggested by table 2, there are sill evidences of discrimination on several medical specialties against different subgroups, such as age, gender, and race. Nonetheless,

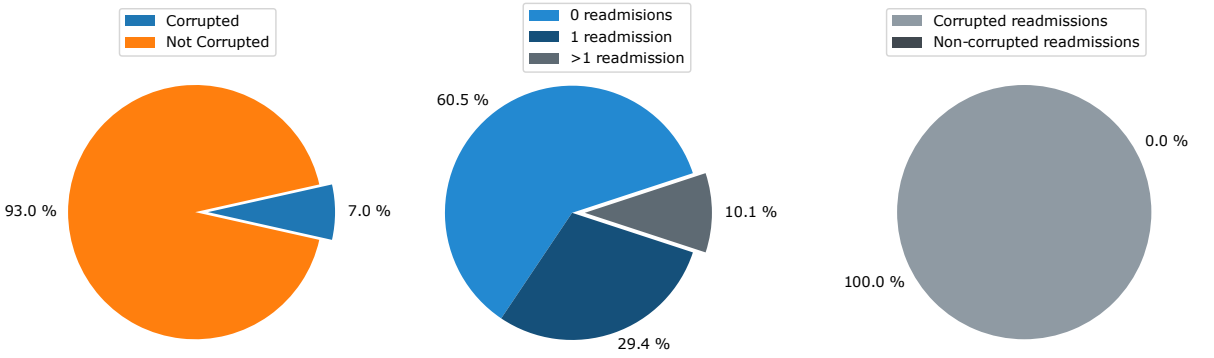


(a) Age (b) Gender (c) Race (d) Insurance status

Figure 4: Sensitive classes distribution across the whole dataset and a readmitted-only subset

there are no evidences of discrimination based on the insurance status of the patient on any of the medical specialties.

Proceeding with the corruption analysis, figure 5 reveals the analysis to the new collected dataset. Recall that a medical encounter is considered corrupted if there is a second one where the same patient has a different `blood_type`. Figure 5a shows that 7.0 % of the data set is corrupted, i.e. 7.0 % of the medical encounters have at least another medical encounter where `blood_type` does not match for the same patient.



(a) Corrupted medical encounters across the whole dataset

(b) Target/Readmission distribution across the corrupted data

(c) Corrupted distribution on a readmitted-only subset from 5b

Figure 5: Data integrity analysis of based on patients who visited the hospital at least twice

Figure 5b drills down into the corrupted entries and focus on the target variable: `readmitted`. It shows that 60.5 % of those corrupted medical entries belong to patients that were never readmitted into the hospital. 29.4 % belongs to patients that were readmitted once. The remaining, 10.1 % are entries of patients that were readmitted into the hospital at least twice.

Finally, figure 5c focus on the 10.1 % of corrupted medical entries that belong to patients readmitted more than once. Assuming that a medical encounter is handled with extra care when a readmission occurs, and knowing that these patients were readmitted more than once, this chart checks if the all their readmitted medical encounters have a matching `blood_type`, i.e. if the re-admissions are corrupted when only compared to re-admissions. It turns out that all of those medical encounters have mismatched `blood_type` observations for the same patient.

Table 2: Detailed per-specialty discrimination analysis on the new collected data. All values are normalized to the overall medical specialty readmission occurrence on the last column. Values above the 25 % threshold are marked in **bold**, values below the threshold are underlined. Dashes mean there were no occurrences for that class, whilst zeros mean there no re-admissions for that class.

Medical Specialty	Age											Gender		Race					Insured		Spec. Rate
	0	10	20	30	40	50	60	70	80	90	NA	F	M	asian	Black	Hisp.	Other	White	Yes	No	
Endocrinology	-	-	**11.00**	-	0	0	0	0	-	-	-	**2.20**	0	-	0	-	-	**1.57**	-	1.00	0.09
Others	0	0	**1.52**	**1.28**	0.84	0.97	1.02	0.96	1.04	**1.50**	0.77	1.04	0.96	0	0.99	0.71	0.31	1.05	0.80	1.01	0.13
Family/GeneralPractice	-	0	0.90	0.56	1.14	0.76	1.02	1.08	1.19	0.50	**1.30**	1.01	0.99	**1.35**	1.24	0	0.48	1.01	0.87	1.01	0.15
Radiology	-	-	-	-	**2.56**	**1.33**	0.67	**1.38**	0	0	0	**1.32**	0.72	-	**2.05**	-	0	0.92	1.14	0.99	0.10
Orthopedics	-	-	0	0	**2.08**	0.74	0.94	0.97	**1.40**	**6.78**	0	**1.33**	0.41	-	0.88	0	-	0.94	0	1.03	0.07
InternalMedicine	-	0	0.54	0.96	0.86	0.70	1.15	1.07	1.16	0.34	1.22	0.95	1.07	**1.56**	0.83	**1.30**	**1.34**	1.07	0.23	1.02	0.14
Emergency/Trauma	-	0	1.23	1.13	**1.35**	0.62	**1.29**	1.02	0.84	0.89	0.70	0.89	1.14	1.23	0.70	0.74	0.82	1.05	0.82	1.04	0.14
Pulmonology	-	-	-	**4.20**	0	0.65	0.42	**1.46**	0	**4.20**	**2.80**	0.84	1.15	**4.20**	1.10	0	0	0.66	0	1.05	0.12
Gynecology/Obstetrics	-	**5.83**	0.73	0.65	0.73	**1.67**	0	0	0	**11.67**	-	1.00	-	0	**2.75**	0	0	0.53	0	1.01	0.09
Surgery	-	**4.05**	**3.24**	0.37	**1.42**	1.18	0.83	1.06	0.90	0	0.40	0.93	1.06	0	0.80	1.16	**1.35**	1.02	1.19	0.99	0.12
Cardiology	-	-	**9.94**	**1.99**	0.95	0.82	0.54	**1.35**	**1.40**	0	0.66	0.88	1.10	**9.94**	**1.36**	0	0.66	0.93	0.62	1.01	0.10
Pediatrics	0	**1.76**	-	-	0	0	0	0	-	-	0	**1.69**	0	0	**4.89**	-	-	0	0	1.02	0.02
Nephrology	-	-	**5.30**	0.76	0.48	1.16	0.78	1.15	**2.12**	-	0	0.76	1.21	-	0.98	-	0	1.06	0	1.02	0.19
PhysicalMedicine	-	-	-	0	0	0	0.72	1.21	**1.44**	-	-	0.88	1.15	-	0	0	**5.75**	1.06	-	1.00	0.17
Gastroenterology	-	-	-	0	0	0.85	1.24	**1.87**	1.17	0	0	1.00	1.00	-	**1.56**	-	-	0.93	-	1.00	0.11
Hematology	-	-	-	-	-	0	0	**1.28**	**2.56**	-	-	1.02	0.96	-	-	-	-	1.00	0	1.05	0.26
Psychiatry	-	**1.73**	0.74	0	0.92	1.20	1.22	**1.60**	0.74	-	0	0.92	1.11	-	0.60	-	0	1.16	0	1.07	0.19
Neurology	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00
Urology	-	-	-	0	0	**2.21**	0.55	**2.15**	0	-	0	**1.28**	0.90	-	**2.70**	0	-	0.81	0	1.01	0.08
Oncology	-	-	-	**3.40**	**1.70**	0.97	0.57	0.68	**1.36**	-	0	1.00	1.00	-	0.97	-	0	1.09	-	1.00	0.29
InfectiousDiseases	-	-	-	**2.00**	-	-	0	-	1.00	-	-	**1.33**	0	-	1.00	-	-	1.00	-	1.00	0.50
Podiatry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00
Anesthesiology	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00
Ophthalmology	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00

3 Next Steps

3.1 Next Steps

Looking at the results drawn from the collected data during production, there are areas of improvement regarding the predictive model. The model failed some of the key business requirements and those issues should receive most of the attention.

The model seems to have overfitted to the training data, which was one of the dangers raised on the first report. The main reason for this issue is probably related to the unbalanced dataset that forced an undersampling of the training data available. Perhaps, alternatives to the undersampling should be considered, such as oversampling. Oversampling could provide better results, given the current scenario, since no data is discarded, thus helping the model to better fit the problem.

Nonetheless, an investigation should be conducted in order to identify measures that would prevent the *Gradient Boosting Classifier* from overfitting. The previous report ignored hyper parameters such as `min_samples_leaf` and `min_samples_split` that seem beneficial to prevent overfitting. Increasing these hyper-parameters makes it harder for the model to memorize single observations, and potentially preventing this issue.

The discrimination requirements, as mentioned above, are hard to achieve on features with a high number of categories. This is related to the spread of observations across a high number of categories leading to a scarce amount of observations per category. This phenomenon will skew metrics to extremes, either very high or very low, making it harder to meet criteria.

Furthermore, the Hazel and Bazel hospital seems to be dealing with an issue of discrimination on different medical specialties, as warned on the first report and repeated on the new data on the present report. The hospital should focus on analysing the root causes of such issues on a per medical specialty basis. Higher resources should be applied trying to solve the issue at the root, i.e. on the patient care itself, rather than on the readmission validation, since these criteria lower the model performance. Naturally, lowering the model performance results in a model that is less accurate compared to what it could possibly be. In the end it, the Hazel and Bazel hospital should evaluate if the discrimination issues outweigh a wrongful discharge.

4 Deployment Issues

4.1 Re-deployment

The deployment went smoothly from start to finish with no major warnings during the production week. Naturally, not having access to the complete logs of the application forced constant monitoring to catch unexpected errors. Nonetheless, there is the risk of having missed some error messages, but taking into account the amount of data collected and the expected volume of traffic, it seems unlikely.

Given the model's under-performance based on the collected data analysis, a new version aiming towards a performance improvement was indispensable to face the second week in production. Thus, the new collected data was merged with the old data in order to create a new bigger dataset. The final merged data set has a total of 90 906 observations, where only 10 146 are readmitted medical encounters.

In order to balance the data, and avoid data loss, the under-represented class was re-sampled 8 times, resulting in a balanced dataset with a total of 161 928 medical encounter where 50.13 % are readmitted.

Afterwads, a grid search was conducted in order to find the best hyper-parameters for the *Gradient Boosting Classifier*. This time, as suggested in the previous section, the search also included `min_samples_leaf` and `min_samples_split` parameters in an effort to prevent overfitting. The grid search result is listed in table 3.

Table 3: Retrained model's hyper-parameters.

Hyper-Parameter	
`random_state`	42
`n_estimators`	100
`learning_rate`	30 %
`max_depth`	10
`min_samples_leaf`	100
`min_samples_split`	125

The retrained model expected performance improved when compared to the previous model. Its new performance metrics, based on the test set, are available on table 4.

Table 4: Retrained model's performance on the test set (20 % of the whole merged dataset).

	Accuracy	Recall	Precision	F1 Score
Test set	81 %	92 %	75 %	0.83

Finally, after a discrimination analysis for the final model similar to the one on section 2.2, the discrimination requirements also improved. The new model is now expected to fulfill discrimination threshold for the race, gender, and insurance status sub-groups with a maximum variance of 7.48 %, 1.90 %, and 0.56 % respectively. It almost meets the age discrimination threshold with a maximum expected variance of 11.12 %. Regarding, the medical specialty the maximum discrimination threshold is still overrun, but the variance has drastically improved to a maximum 18.09 % when compared to the previous model.

4.2 Unexpected problems

As mentioned, the deployment went smoothly from start to finish during the production week. There were no errors and both the app and predictive model were able to provide an answer to all the requests, either with a prediction or a detailed explanation of the error.

Nonetheless, it is known that the production deployment allowed the collection of 9694 new medical encounters, but their true outcome is known for all except 200. The issue that led to these 200 missing outcomes is suspected to be related with extra fields on the request that were not explicitly removed, such as `id` and `index`. These fields were indeed removed for the `/predict` endpoint, but not for the `/update` endpoint, thus leading to this issue. This minor issue was known during production week, but the risk of re-deploying at the time outweighted the benefits, since a re-deployment could impose permanent data loss and also cause a system unavailability.

Furthermore, it is worth noting a small issue imposed by the `heroku` platform. At the middle production week, `heroku` relocated the server where the application was running as detailed by the logs on figure 6.

```
2022-02-09T13:45:18.192202+00:00 heroku[router]: at=info method=POST path="/predict" host=e6281d438ef244b3b42f775aae.herokuapp.com request_id=84b040fa-c60e-4b69-
b86d-5d4a7d63b5b3 fwd="52.208.22.146" dyno=web.1 connect=0ms service=104ms status=200 bytes=172 protocol=https
2022-02-09T13:46:02.197842+00:00 heroku[router]: at=info method=POST path="/predict" host=e6281d438ef244b3b42f775aae.herokuapp.com request_id=50cd5650-
c750-4d78-9faf-6add43d88c4a fwd="52.208.22.146" dyno=web.1 connect=0ms service=121ms status=200 bytes=172 protocol=https
2022-02-09T13:46:24.125698+00:00 heroku[web.1]: Relocating dyno to a new server
2022-02-09T13:46:24.219296+00:00 heroku[web.1]: State changed from up to down
2022-02-09T13:46:24.229539+00:00 heroku[web.1]: State changed from down to starting
2022-02-09T13:46:46.634709+00:00 heroku[web.1]: Starting process with command `/bin/sh -c /bin/bash\ -c\ \'gunicorn\ app:app\'`
2022-02-09T13:46:48.504742+00:00 app[web.1]: [2022-02-09 13:46:48 +0000] [5] [INFO] Starting gunicorn 20.0.0
2022-02-09T13:46:48.505281+00:00 app[web.1]: [2022-02-09 13:46:48 +0000] [5] [INFO] Listening at: http://0.0.0.0:54413 (5)
2022-02-09T13:46:48.505439+00:00 app[web.1]: [2022-02-09 13:46:48 +0000] [5] [INFO] Using worker: sync
2022-02-09T13:46:48.526524+00:00 app[web.1]: [2022-02-09 13:46:48 +0000] [8] [INFO] Booting worker with pid: 8
2022-02-09T13:46:48.967890+00:00 heroku[web.1]: State changed from starting to up
2022-02-09T13:46:59.058159+00:00 heroku[web.1]: Stopping all processes with SIGTERM
2022-02-09T13:47:00.434356+00:00 app[web.1]: [2022-02-09 13:47:00 +0000] [5] [INFO] Handling signal: term
2022-02-09T13:47:00.434630+00:00 app[web.1]: [2022-02-09 13:47:00 +0000] [8] [INFO] Worker exiting (pid: 8)
2022-02-09T13:47:00.735883+00:00 heroku[web.1]: Process exited with status 143
2022-02-09T13:47:27.044795+00:00 heroku[router]: at=info method=POST path="/predict" host=e6281d438ef244b3b42f775aae.herokuapp.com request_id=6c29a823-3047-4c0e-
8642-2177e18b4518 fwd="52.208.22.146" dyno=web.1 connect=0ms service=107ms status=200 bytes=173 protocol=https
2022-02-09T13:48:10.100357+00:00 heroku[router]: at=info method=POST path="/predict" host=e6281d438ef244b3b42f775aae.herokuapp.com request_id=01583019-5b94-489e-
b120-b1fed91e4706 fwd="52.208.22.146" dyno=web.1 connect=0ms service=296ms status=200 bytes=173 protocol=https
```

Figure 6: Logs of heroku dyno relocation to a new server

This issue does not seem to have impacted the application, however one can note that it took 1 min and 30 s to complete the whole application relocation. Knowing the rate of requests was roughly one per minute, it is possible that some data was lost in the process.

This issue was imposed by the `heroku` platform, and as highlighted on the first report this kind of issues were out of the Awkward Problem Solutions™ control.

Other than that, no unexpected error occurred during the whole week in production.

Learnings and Future Improvements

The application deployment was perhaps the part of the current project that was better executed. Some of the key takeaways that revealed important are related to the overall app's robustness.

First of all, it is highly important to develop a broad model pipeline capable of receiving all types of data, specially `NaN`, and still output a result.

Then, the app itself has to be able to interpret the data that the REST API user provides. During this project an effort was made to interpret the data, discarding requests only as the last resort. For instance, when a field type is an integer, but the REST API user provides the string *"11"* the app should be able to convert into the number 11, without discarding the request as invalid. Relying on the database to do this kind of heavy lifting was an important learning from

this project. Since the database is, by definition, able to do this type of conversions leveraging this functionality improved the app robustness and increased the development speed.

Moreover, an important feature that was overlooked during the preparation for the production week is access to logs. This was a known limitation from `heroku`, but the information it provides is highly valuable and should never be missed in a production application.

Given the limitation, one alternative would be adding a second table with the triggered exceptions. However, this alternative was not viable for the current project, since it imposed a data loss risk due to the 10 000 rows limitation from `heroku`. A different alternative would be creating a local log file to be accessed later.

Finally, `DBeaver` revealed to be an important tool during the deployment and production stage. It allowed accessing the remote database hosted by `heroku` and proved to be a valuable monitoring tool. On top of that, it made the extraction of the database into a `csv` file seamless and quick.