Predicting Planning Decisions from Open Data: A Reproducible Analysis

Eimann Dekkar
Advanced Machine Learning

November 5, 2025

1 Introduction

I study whether short planning descriptions and simple location signals from an open local government dataset can help predict grant decisions. The data were obtained from the Dún Laoghaire–Rathdown (DLR) planning applications published on the Irish national open-data portal [7]. Each record includes free text describing the proposal, a decision string, and point coordinates. I define a binary target by mapping the decision text to `GRANTED` or `NOT_GRANTED`, and build a compact feature set that mixes the TF–IDF of the description with a description-length proxy and basic geography. The question is straightforward: With a reproducible pipeline and course-aligned methods, how much signal can these public fields provide for early triage?

The design follows a standard defensible practice. I placed all preprocessing inside pipelines to prevent leakage and keep the experiments comparable [11]. For text, I use TF–IDF with unigrams and bigrams [12]; for dense models, I reduce dimensionality with truncated SVD (LSA-style) to a compact topic space [4]. I evaluated four lightweight classifiers that cover linear, instance-based, tree-based, and neural baselines, and ran stratified 10-fold cross-validation with macro-averaged metrics due to class imbalance [13]. I also report the ROC–AUC to assess ranking quality across thresholds [10]. To move beyond the trivial application of algorithms, I add a feature-selection study on the linear stack—univariate chi-square filtering and L1-based model selection—so that I can quantify the effect of vocabulary reduction versus a full TF–IDF space on this task.

This setup provides three contributions. First, I define a clear, reproducible target, and transparent feature set that a practitioner can audit and explain. Second, I compare models under identical folds and transformations, which isolates modelling gains from data handling. Third, I quantified the impact of feature selection relative to a strong linear baseline. The headline result previews the story: a small Random Forest trained on reduced text topics and simple location features offers the best balance by macro F1 (0.671) and ROC–AUC (0.741), whereas a compact MLP attains the highest accuracy and precision at the cost of lower recall. In practice, this means that open text already carries enough information to prioritize a subset of applications for a second look, provided we calibrate thresholds to local preferences and accept the limits of historical labels.

2 Methodology

2.1 Data

In this project, I used the Dún Laoghaire–Rathdown (DLR) planning applications open dataset from the Irish national open-data portal [7]. The published CSV includes core application fields (references, dates, and decision text), free-text descriptions, and point geography (Latitude/Longitude plus Irish Transverse Mercator eastings/northings). The portal notes that the point represents the application site in a broad sense and may be missing for some records; this matches what was observed in the raw file [6]. The dataset was released under CC-BY 4.0, which permits reuse with attribution [3].

Target definition. The predictive task is binary: whether a planning application is granted. I map the original `decision` strings to a clean label `decision_binary` with two values: `GRANTED` if the text begins with "GRANT," otherwise `NOT_GRANTED`. Rows with missing data were dropped. After cleaning, I have 7,139 rows and 8 columns.

Feature set. I maintain a small, transparent feature set that mixes text, geography, and a lightweight area token: (i) the raw description `descrptn` for TF–IDF, (ii) the description length `descr_len` (characters), (iii) geography `lat`, `long`, `itm_east`, `itm_north`, and (iv) a coarse location token extracted from the last comma-separated element of `location` and normalized (`location_token`). Table 1 summarises roles and preprocessing.

Table 1: Features used for modelling.

Feature	**Type**	**Role / preprocessing**
`descrptn`	text	Main signal; TF–IDF with unigrams/bigrams. Missing filled with "na" to keep rows.
`descr_len`	numeric	Proxy for content volume; median imputation for missing.
`lat`, `long`	numeric	Location; parsed to numeric, median imputation; used as dense scalars and for sanity checks.
`itm_east`, `itm_north`	numeric	Alternative coordinates; same parsing and imputation as above.
`location_token`	categorical	Coarse area token from `location`; lowercased and cleaned; missing as "unknown".
`decision_binary`	target	`GRANTED` vs `NOT_GRANTED`.

Cleaning and imputation. I standardize column names to lowercase, strip whitespace, and coerce non-numeric geography to `NaN`. For text, I fill missing descriptions with a single placeholder token to avoid dropping rows during vectorisation and keep missing location as the explicit category "unknown." For numeric fields, I use median imputation; it is robust to skew and keeps the baseline simple. I also compute a Dublin bounding-box flag to capture obvious coordinate errors during EDA; I do not use this flag as a feature.

Why this pipeline. The goal is a reproducible baseline that avoids leakage while staying faithful to course guidance: transformations used by the models (TF–IDF, scaling, one-hot) live inside `Pipeline/ColumnTransformer` for cross-validation, and fixed imputations are applied once to stabilize downstream steps. This keeps the modelling stack clean and makes each experiment easy to compare and repeat.

Exploratory findings. The class distribution was imbalanced with 5,619 `GRANTED` (78.7%) and 1,520 `NOT_GRANTED` (21.3%) [Fig. 1]. Description lengths are right-skewed; most records fall between 100 and 500 characters with a long tail to roughly 5k+. The granted cases are slightly denser in the 150–400 character band [Fig. 2]. Geography sits in the expected DLR window (longitude −6.30 to −6.10, latitude 53.21 to 53.31) with heavy overlap between classes, suggesting limited standalone separation from coordinates [Fig. 3]. The top tokens confirm domain consistency—for example, *ground* (9,353), *level* (7,817), *entrance* (7,260), *dwelling* (6,976), and *construction* (6,126)—which supports the use of TF–IDF over `descrptn` [Fig. 4].

Relevance to the overall analysis. These choices set the constraints for the modelling section: I will use stratified 10-fold cross-validation and macro-averaged metrics to respect the 79/21 split; I will keep `descr_len`, the compact location token, and dense geography alongside text features; and I will compare linear and non-linear learners under identical preprocessing so differences come from the models, not from inconsistent data handling. Limitations are clear: the site point does not capture the full site geometry, and simple median fills do not borrow neighborhood structures; I address the first by not over-interpreting spatial effects and the second by reporting robust, fold-wise estimates.

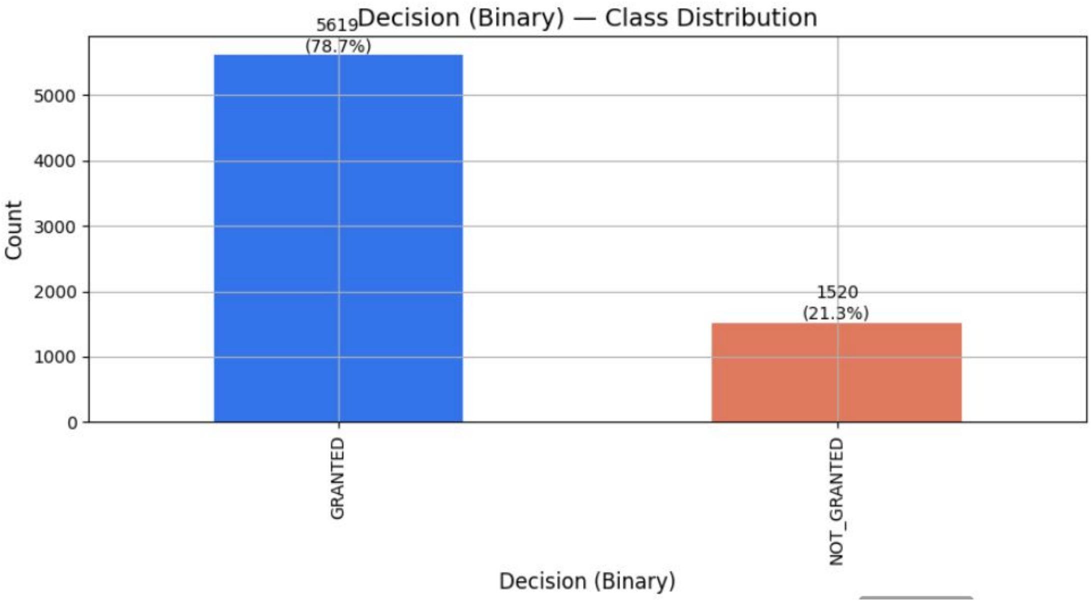

Figure 1: Binary decision class counts with percentages (GRANTED 5,619 / 78.7%, NOT_GRANTED 1,520 / 21.3%).

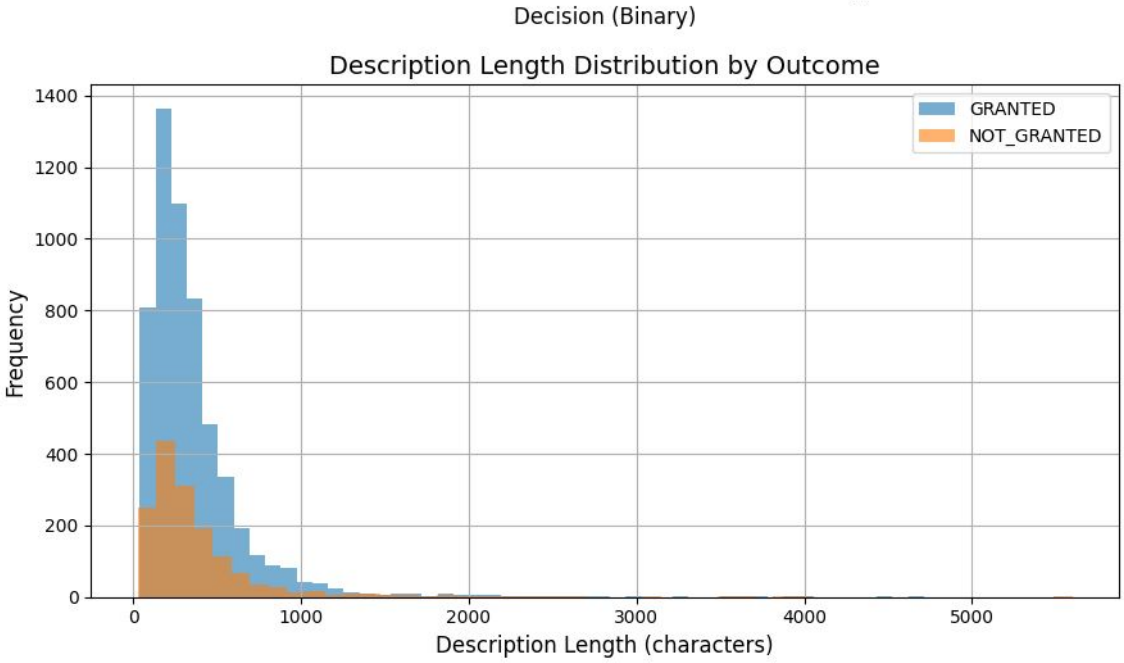

Figure 2: Description length distribution by outcome. Right-skewed overall with a denser mass for GRANTED around 150–400 characters.

2.2 Modelling

The task was cast as a binary classification problem with GRANTED=1. To avoid leakage and keep the experiments comparable, every transformation lives inside a single Pipeline with a

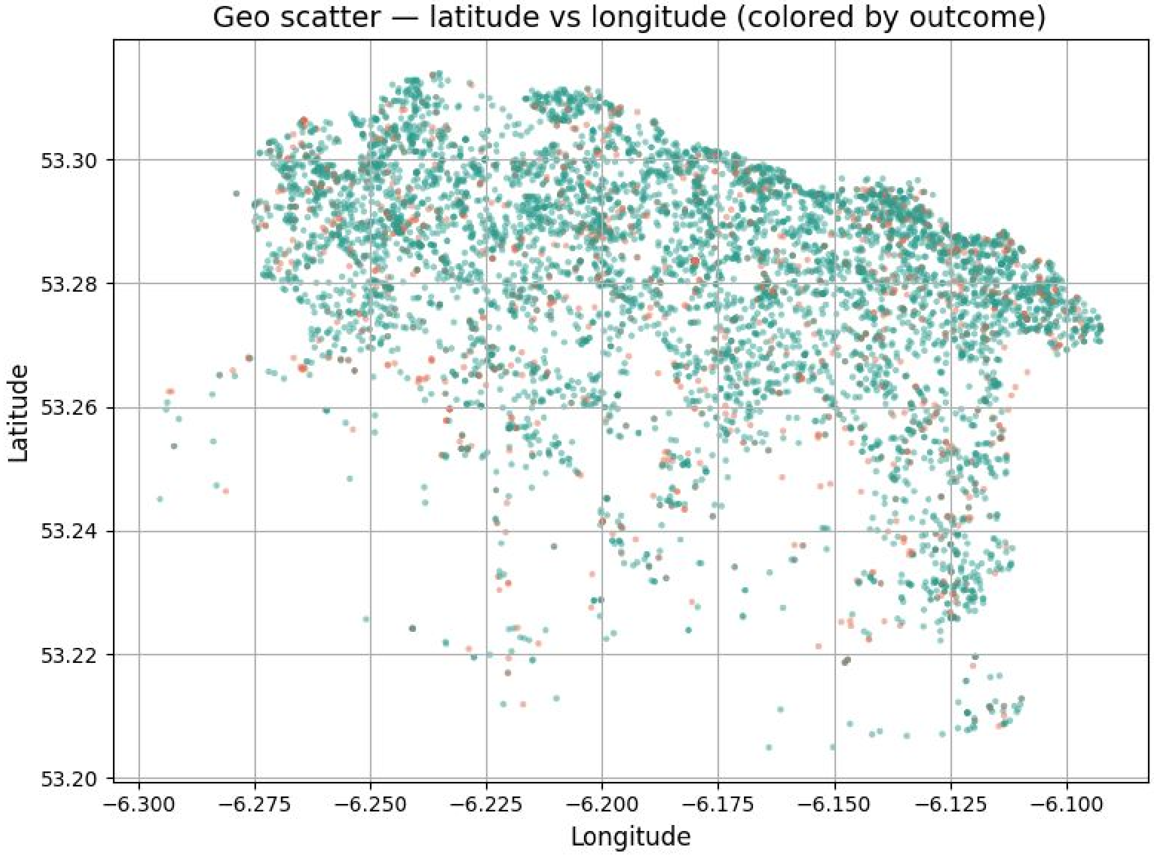

Figure 3: Longitude vs latitude coloured by outcome. Points lie within the DLR window (−6.30 to −6.10, 53.21 to 53.31); classes overlap strongly.

`ColumnTransformer` [11]. Two pre-processing paths were used. The *sparse* path vectorizes the description with TF–IDF using unigrams and bigrams (cap 8,000 terms) [12] and maintains numeric and categorical features in sparse form. The *dense* path applies TF–IDF and then reduces the text space with truncated SVD to 60 components (LSA-style) [4], scales numeric features, and one-hot encodes the area token in dense form. I chose 60 components as a compact representation that preserves broad topics while keeping the feature budget sufficiently small for distance-based and neural models to train reliably.

I adopted stratified 10-fold cross-validation with shuffling and a fixed seed for out-of-fold (OOF) estimates (`StratifiedKFold`, `random_state=42`). Because the classes are imbalanced (approximately 79/21), I report macro-averaged precision, recall, and F1, plus ROC–AUC from probabilistic outputs when available [10], [13]. Confusion matrices were plotted per model to show the trade-off in the minority class.

For the linear baseline, I use Logistic Regression on the *sparse* path with `liblinear`, `class_weight=balanced`, and a log-spaced search over $C \in [10^{-2}, 10]$ (eight values). This baseline tests whether a linear separator over a high-dimensional TF–IDF is already competitive. For KNN, I switch to the *dense* path so distances operate in a compact space, and search `n_neighbors` $\in \{5, 10, 15, 25\}$, `weights` $\in$\{uniform, distance\}, and ℓ_p with $p \in \{1, 2\}$ [2]. For Random Forest, I keep the dense path and enable `class_weight="balanced_subsample"` to stabilise splits under skew; I search `n_estimators` $\in \{150, 250, 400\}$, `max_depth` $\in \{\text{None}, 10, 20\}$, `min_samples_leaf` $\in \{1, 2, 5\}$, and `max_features` $\in \{\text{"sqrt"}, \text{"log2"}, 0.5\}$ [1]. For the MLP, I use the dense path with early stopping and tune `hidden_layer_sizes` $\in \{(100,), (200,), (100, 50)\}$, `activation` $\in \{\text{relu}, \text{tanh}\}$, `alpha` $\in \{10^{-4}, 10^{-3}, 10^{-2}\}$, and `learning_rate` $\in \{\text{constant}, \text{adaptive}\}$ [11]. All

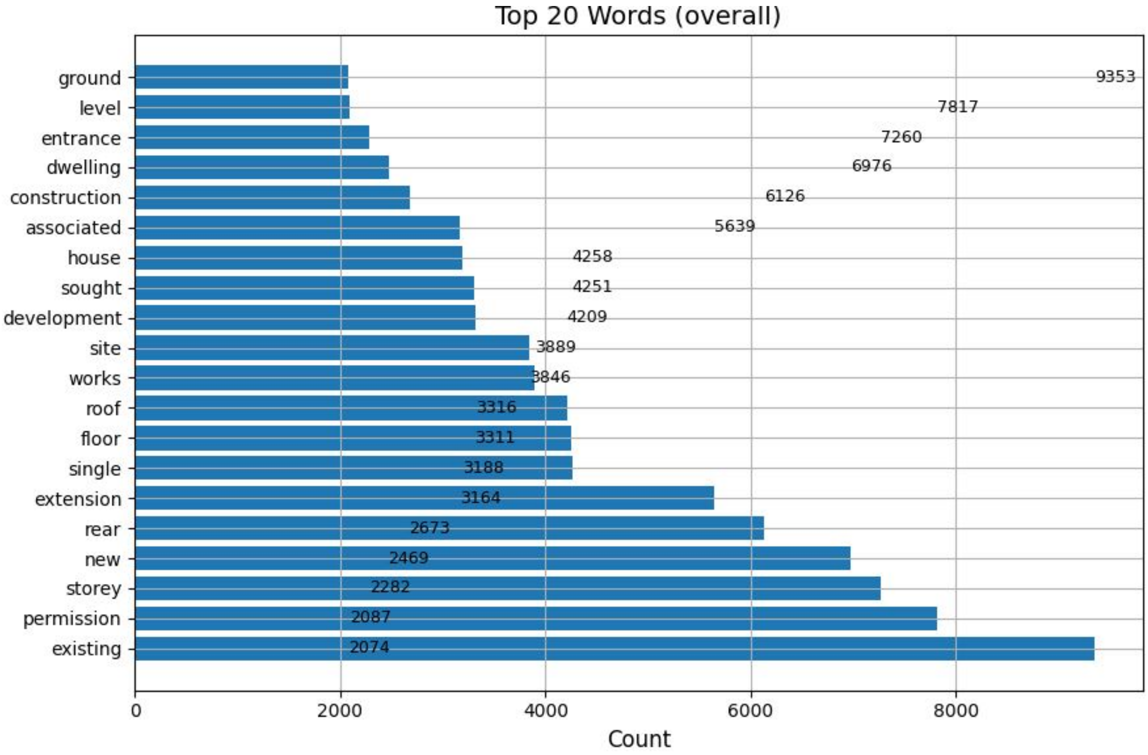

Figure 4: Top 20 tokens in `descrptn`. Frequent tokens match planning lexicon (e.g., *ground*, *level*, *entrance*, *dwelling*, *construction*).

searches use 5-fold `RandomizedSearchCV` on `f1_macro`, and 10-fold OOF metrics are computed by refitting the selected configuration inside the cross-validation loop.

I keep the modelling choices simple on purpose. The sparse linear baseline provides a transparent reference; the dense path allows non-linear models to mix reduced text topics with geography and the area token; and the metrics emphasize balanced performance. This setup supports the comparative analysis in the next section, where I contrast models on the same folds and examine their confusion matrices and OOF summary tables.

2.3 Feature Selection

Two supervised selectors that operate inside the cross-validation pipeline were compared to avoid leakage [11]. The first is a univariate filter on the text branch: *SelectKBest* with the chi-square statistic over TF–IDF features ($k = 800$) [8], [15]. Chi-square ranks tokens by how strongly their presence is associated with the class; it requires non-negative inputs, which TF–IDF satisfies. I apply this selector *only* to the text branch and pass numeric (`descr_len`, coordinates) and categorical area tokens unchanged (dense scaling and one-hot, respectively). The resulting representation is compact (10% of the 8,000-term cap) yet expressive: it maintains the highest-signal unigrams/bigrams while reducing noise and training costs for linear models.

The second selector is a multivariate model-based approach on the full sparse linear space. I train an L1-penalised Logistic Regression (`penalty=L1`, `solver=SAGA`) with class balancing to induce sparse coefficients [5], [14]. I then wrap it with *SelectFromModel* using a *median* coefficient-importance threshold, which retains features whose absolute weights exceed the median across all nonzero weights. After selection, I refit a standard L2-regularised Logistic Regression on the reduced set. This two-step "L1 for selection, L2 for estimation" pattern is a common way to stabilize coefficients after sparse screening while preserving linear interpretability [9].

Both selectors live inside `Pipeline/ColumnTransformer` so that for each training fold, the selector fits on the fold's training data and transforms the validation split with parameters learned from that fold only. I keep the evaluation protocol identical to the modelling section: stratified 10-fold OOF estimates with macro-averaged precision, recall, and F1, plus ROC–AUC when probabilistic outputs are available. The comparison in Section 3 contrasts (i) the no-selection linear baseline, (ii) chi-square token filtering on text, and (iii) L1-based model selection, holding all other steps fixed. This isolates the effect of selection from confounders, such as re-vectorization or altered scaling.

3 Results & Discussion

I evaluated all models with stratified 10-fold out-of-fold (OOF) predictions and reported the accuracy, macro precision, macro recall, macro F1, and ROC–AUC. Macro metrics matter here because the target is imbalanced (78.7% `GRANTED` vs. 21.3% `NOT_GRANTED`). Table 2 summarizes the comparisons; I bold the best value in each column.

Table 2: Model comparison (10-fold OOF). Best per column in **bold**.

Model	Acc	P_{macro}	R_{macro}	$F1_{\text{macro}}$	ROC–AUC
Logistic Regression	0.677	0.596	0.629	0.597	0.682
KNN	0.788	0.668	0.613	0.627	0.663
Random Forest	0.797	0.691	**0.658**	**0.671**	**0.741**
MLP	**0.806**	**0.727**	0.589	0.602	0.708

The Random Forest was the best overall by macro F1 (0.671) and ROC–AUC (0.741). This indicates a stronger balance between classes and a better ranking of positives across thresholds than the other models. The best configuration uses 150 trees, a maximum depth of 10, minimum samples per leaf of 5. The MLP records the highest accuracy (0.806) and macro precision (0.727) but lower macro recall (0.589), which suggests a conservative behavior: it commits fewer positive predictions overall and misses more minority `NOT_GRANTED` cases. KNN benefits from the dense SVD pathway and lands between the linear and tree models in F1 (0.627). The sparse linear baseline is the simplest model; it sets a useful reference but underfits the nonlinear alternatives.

These differences were aligned with the feature mix. Dense SVD components capture broad topics from the descriptions and interact with geography and area tokens. Trees can exploit these interactions without requiring manual feature engineering. The MLP can learn non-linearities as well, but with limited data and a modest network, it prefers safer boundaries that trade recall for precision. This trade-off is important in a planning support setting. If the goal is to *flag likely refusals* for further scrutiny, the recall of `NOT_GRANTED` is more valuable than a small gain in overall accuracy. Under this objective, the Random Forest is a better default because it recovers more minority cases while keeping the precision acceptable. If the goal is *to minimize false alarms* in a high-volume triage, the MLP's higher precision may be preferable, but threshold calibration would be needed to maintain minority recall at an acceptable level.

I also compare feature selection on the linear stack to show "with vs without" effects. Table 3 reports the OOF metrics for the baseline sparse Logistic Regression, chi-square token filtering ($k = 800$) on the text branch, and L1-based selection with `SelectFromModel` followed by an L2 refit. Neither selector improves macro F1 over the no-selection baseline; the L1 route preserves the accuracy slightly better, whereas chi-square filtering drops F1 further.

This outcome is consistent with the manner in which the selectors work. The text branch already uses TF–IDF with regularization in the classifier; aggressive univariate filtering at $k = 800$ can drop informative bigrams that help separate edge cases, which explains the decline. L1 screening is more targeted but, with a median threshold, it can become too sparse on this

Table 3: Feature selection comparison on logistic regression (10-fold OOF). Best per column in **bold**.

Variant	Acc	P_{macro}	R_{macro}	$F1_{\text{macro}}$	ROC–AUC
LogReg_base	0.677	**0.596**	**0.629**	**0.597**	**0.682**
L1Select+LogReg	**0.683**	0.592	0.620	0.595	0.674
KBestChi2+LogReg	0.656	0.585	0.617	0.581	0.673

dataset and hurt calibration a little. The practical benefit of selection is operational: both routes reduce the feature count and training time for linear models. For deployment on constrained hardware, the L1 approach is a reasonably compact surrogate with minimal loss. For accuracy, the no-selection baseline remained the strongest among the linear variants.

What the results mean in practice. The feature space derived from short planning descriptions contains sufficient signals to rank applications reliably. A lightweight Random Forest trained on reduced text topics plus simple location features can support early triage: for example, flagging a subset of applications for a second look because their profile resembles past refusals. The MLP's higher precision suggests that it is suitable when false positives carry a real review cost. In both cases, threshold calibration against a held-out set and class-specific analysis (e.g., focusing on `NOT_GRANTED` recall) would allow a planner to tune the system to local priorities.

Limitations and next steps. The labels reflect past decisions and may encode policy shifts over time; temporal drift is not modelled here. The coordinates are single site points and do not capture parcel geometry or neighborhood context; richer spatial features could help. I use simple median imputation and do not tune decision thresholds or calibrate probabilities; both are important before deployment. Finally, all estimates were obtained using internal cross-validation. A temporal split (train on older cases, test on newer ones) or an external county would provide a harder test of generalization.

4 Conclusion

This work shows that a modest, fully reproducible pipeline can extract useful signals from public planning records. TF–IDF over short descriptions, a coarse area token, and basic coordinates are sufficient for a Random Forest to achieve balanced performance (macro F1 0.671; ROC–AUC 0.741), with a linear baseline close behind and a cautious MLP trading recall for precision. Neither chi-square filtering nor L1-based screening improved the linear model's F1 over the full TF–IDF space, which suggests that for this dataset, regularization and the original vocabulary already strike a good bias–variance balance. The practical takeaway is simple: if the goal is to flag likely refusals for review, prefer Random Forest, and tune its threshold for minority recall; if the goal is to minimize false alarms, start from the MLP and calibrate probabilities before deployment.

This analysis has clear limitations. Labels encode past decisions and may drift over time; I did not model temporal changes. Coordinates are single points and omit neighborhood contexts; richer spatial features (zoning, ward effects, proximity to amenities) could help. I used median imputation and did not optimize class-dependent thresholds or perform probability calibration; both are important for a live system. Finally, all figures come from internal cross-validation; a temporal split (training on older applications, testing on newer ones) or an external county would provide a stronger generalization check. The next steps follow directly: add calibrated decision thresholds, run a time-aware evaluation, engineer simple spatial context, and document operating points that match planning office priorities. With these pieces in place, the same pipeline can support transparent, low-cost triage on open data without heavy infrastructure.

References

[1] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[2] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[3] Creative Commons, *Creative commons attribution 4.0 international public license (cc by 4.0)*, `https://creativecommons.org/licenses/by/4.0/`, License text; Accessed: 2025-10-28, 2013.

[4] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[5] A. Defazio, F. Bach, and S. Lacoste-Julien, "Saga: A fast incremental gradient method with support for non-strongly convex composite objectives," in *NeurIPS*, 2014, pp. 1646–1654.

[6] Dún Laoghaire–Rathdown County Council, *Dlr planning applications csv — resource notes*, `https://data.gov.ie/en_GB/dataset/dun-laoghaire-rathdown-county-council-planning-applications`, Field descriptions and coordinate availability; Accessed: 2025-10-28, 2025.

[7] Dún Laoghaire–Rathdown County Council, *Dún laoghaire–rathdown county council planning applications*, `https://data.gov.ie/en_GB/dataset/dun-laoghaire-rathdown-county-council-planning-applications`, Accessed: 2025-10-28, 2025.

[8] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *Journal of Machine Learning Research*, vol. 3, pp. 1289–1305, 2003.

[9] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[10] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[11] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[12] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.

[13] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[14] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[15] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *ICML*, 1997, pp. 412–420.

Video Recording of project:

https://1drv.ms/v/c/81b22a501d41db0f/EfGDI95fXr5NgC5wdJFmwzIB5Rwz9mokIdIp6_p4dbtw8g?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZy1MaW5rIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXcifX0%3D&e=GBDetb