Andrew Feng, Alex Lim, Krish Sarkar, Haofeng Mei
Catboost Report

Section 1: Algorithms Used

During Part 3 of our project, we utilized CatBoost, an algorithm based on the gradient boosting framework, to train models on both the Fashion MNIST dataset (Multiclass classification) and the California Housing dataset (Regression). How CatBoost works is as follows: CatBoost, like other gradient boosting algorithms, trains an ensemble of weak decision trees sequentially, each attempting to reduce the errors made by the previous ones. After each tree is trained, CatBoost computes residuals (the gradient of the loss), and the next tree focuses on correcting the patterns that were least well captured so far. Repeating this process many times yields a strong model with low prediction error.

What differentiates CatBoost from other gradient boosting algorithms is its use of ordered boosting and native categorical feature handling. Ordered boosting ensures that, when CatBoost computes target statistics and residuals for a training example, the model and encodings are built only from data points that come earlier in a random permutation of the dataset. As a result, each sample's residual is calculated by a model not trained on that sample. This avoids target leakage and reduces overfitting. Native categorical handling allows CatBoost to automatically convert categorical features into high-quality numeric representations using target statistics and permutations, eliminating the need for manual preprocessing such as one-hot encoding.

In addition to CatBoost, we also used a few simplistic baseline models in order to compare and measure how much CatBoost is learning. We used the mean predictor, a model which always predicts the mean value of the training dataset, in order to quantify the amount of improvement in root mean squared error that a CatBoost model has over a trivial prediction strategy. We also used a basic linear regression algorithm, which fits a hyperplane to the data and minimizes error via the least-squares method, in order to demonstrate the improvement in performance that CatBoost, a nonlinear model, has compared to the standard linear best fit.

Finally, we use certain algorithms in order to transform datasets, reduce dimensionality, and enhance our understanding of our models and predictions. One such algorithm is PCA dimensionality reduction, which chooses a number of dimensions which preserves the most variance from the original dataset when projected onto the lower space. We included this algorithm in our project in order to observe the effects of training CatBoost models on PCA features, compared to a full set of features. We also used 5-fold cross-validation, which involved partitioning the validation set five different ways and averaging the RMSE and R^2 values across all sets, in order to further determine the viability of CatBoost models in making predictions on varying datasets.

Section 2a: Regression Dataset

We selected the California Housing Prices dataset as our regression task for Part 3 of this project. The target variable that we trained the model to predict is median house value, and the feature set includes longitude, latitude, housing age, room counts, population, households, median income, and ocean proximity. The dataset contains 20,640 samples with 9 input features and 1 target variable. The data was split using a 60% training, 20% validation, and 20% testing strategy to allow for proper model selection and unbiased evaluation. Categorical features were one-hot encoded for baseline models, while CatBoost handled categorical data natively. Median imputation was applied where needed to address missing values, and all experiments were run with a fixed random seed for reproducibility.

Two baseline models were implemented for performance comparison. The first model we used was the mean predictor baseline. This baseline predicts the mean of the training targets for all samples. The metrics that this baseline produced were:

- Validation RMSE: 117,190.26
- Test RMSE: 114,480.56 1.2.2

After encoding the categorical feature and imputing missing values, we trained a linear regression model on the newly embedded data. The metrics that this baseline produced were:

- Validation RMSE: 68,645.28
- Test RMSE: 70,084.57
- Validation R2 : 0.6568
- Test R2 : 0.6252

This shows a large improvement over the naive mean predictor, but performance is still far below what a nonlinear model can achieve.

For our main model, we chose CatBoost, a gradient boosting framework. Its strong performance on tabular data and native handling of categorical features make it particularly useful for our chosen datasets, and being a forest-based model it is quite powerful. A study of possible hyperparameters was conducted over the following parameters and values:

- Number of trees (iterations): 50, 100, 200, 300, 400, 500, 700, 1000
- Tree depth: 3, 4, 5, 6, 8, 10

- Learning rate: .01, .03, .05, .1, .2

We also plotted RMSE against different numbers of iterations, which clearly shows that the iteration numbers we tested were still barely low enough that the model was not overfit at any point.

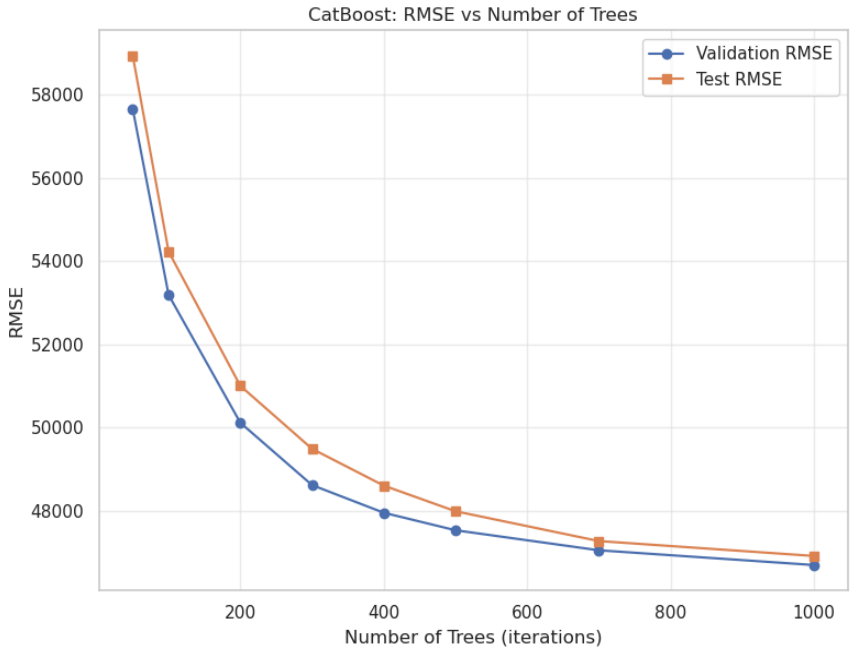

Fig. 1: RMSE vs. Number of Trees

The baseline CatBoost model configuration we decided on, which was 500 iterations, a learning rate of .1, and a tree depth of 6, achieved the following metrics:

- Validation RMSE: 47,531.58
- Test RMSE: 47,988.57
- Cross-Validated RMSE : 47978.35 +- 690.73
- Validation R2 : 0.8355
- Test R2 : 0.8243
- Cross-Validated R2 : .83 +- .01

This represents a major improvement over both baselines, demonstrating the advantage of nonlinear ensemble models for this dataset.

As an additional experiment, Principal Component Analysis (PCA) was applied to the numeric features only. Before PCA, numeric features were standardized and median-imputed to satisfy PCA requirements. PCA retained 95% of the variance, reducing the numeric

dimensionality from 8 to 4 components. A new CatBoost model was trained on the PCA-compressed features, which yielded the following metrics:

- Validation RMSE: 69,029.20
- Test RMSE: 68,952.84
- Cross-Validated RMSE : 69,365.35 +- 795.16
- Validation R2 : 0.6530
- Test R2 : 0.6372
- Cross-Validated R2 : .64 +- .01

Although PCA reduced feature dimensionality and marginally sped up the training process, it significantly degraded performance compared to the full CatBoost model. This indicates that CatBoost benefits from the original feature representation and does not require dimensionality reduction for this task.

Model	Val RMSE	CV RMSE	Test RMSE	Val R2	CV R2
Mean baseline	117190.261467	115092.94 ± 1358.13	114480.562375	NaN	-0.00 ± 0.00
Linear Regression (one-hot + impute)	68645.277900	68621.32 ± 1443.57	70084.566581	0.656826	0.64 ± 0.02
CatBoost (tuned, no PCA)	47531.579510	47978.35 ± 690.73	47988.571545	0.835465	0.83 ± 0.01
CatBoost + PCA (4 comps)	69029.196687	69369.35 ± 795.16	68952.838271	0.652977	0.64 ± 0.01

Fig. 2: Final regression performance comparison for Part 3.

The tuned CatBoost model without PCA achieved the best overall performance for the regression task, with the lowest RMSE and highest R2 on both validation and test sets. The PCA experiment showed that dimensionality reduction was not beneficial for this dataset when using CatBoost. This confirms that CatBoost is the optimal model choice for the California Housing regression task in Part 3.

Section 2b: Classification Dataset

For our classification task, we selected the MNIST Fashion dataset to train our model on. The goal for this dataset was to train the CatBoost model to classify 28 x 28 greyscale images of various articles of clothing, which were labelled as "shirt", "dress", etc. The pixels of each image serve as the dataset's features. The dataset contains 70,000 samples, which we split into a training set of 50,000, a validation set of 10,000, and a test set of 10,000. Pixel values were normalized to [0, 1] for the sake of more consistent and efficient training.

In order to ensure a high validation accuracy, we studied several hyperparameters including iterations, depth, learning rate, L2 regularization, bootstrap type, and subsampling. A grid search with a fixed number of iterations (200) was combined with early stopping using the

validation set; we then optimized the number of iterations during actual training, since the nature of gradient boosting allows us to measure accuracy at each iteration. This procedure allows the model to stop training automatically once validation accuracy stops improving. Below are the hyperparameters we optimized using this approach; the rest were selected based on online research, due to time constraints.

- Number of trees (iterations): 50, 100, 200, 400, 800
- Tree depth: 4, 6, 8
- Learning rate: .03, .1
- L2 Regularization Coefficients: 1, 3, 5

After researching the algorithm, running the grid search, and training the final model, we found that the values of the optimal parameters (among those we considered) were as follows:

- Number of trees (iterations): 800
- Tree depth: 8
- Learning rate: .1
- L2 Regularization Coefficients: 1
- Bootstrap Type: Boolean
- Subsampling: .8

Below is a plot of validation error vs. number of iterations. The curve trends monotonically downwards, implying that the model does not suffer from overfitting at the numbers of iterations we tested.

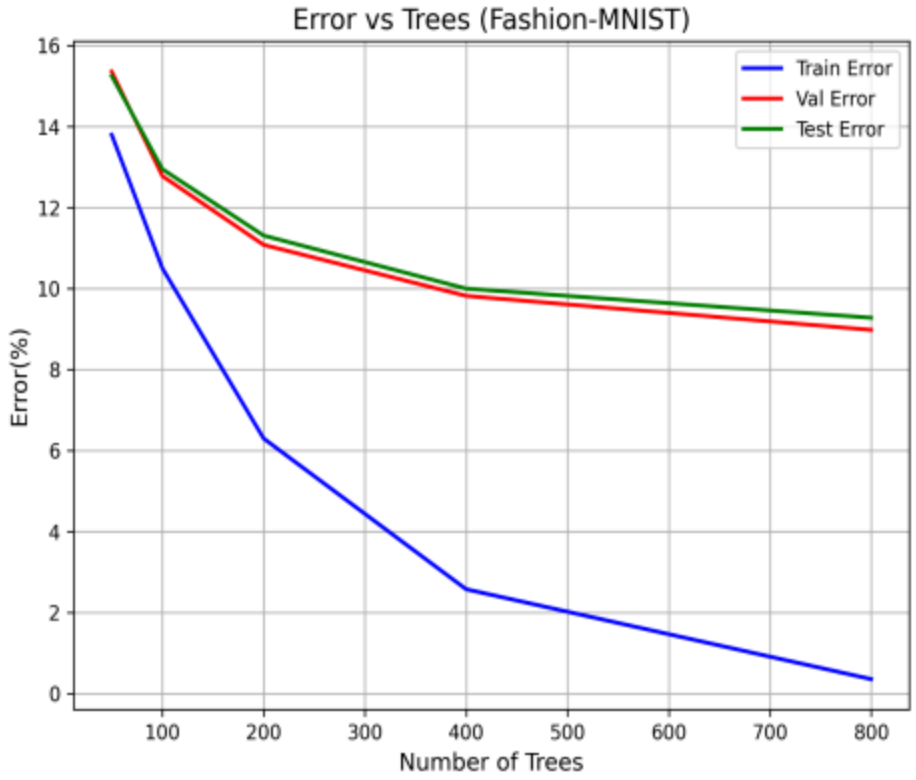

Fig. 3: Validation Error vs. Number of Trees.

Below are the metrics for the standard CatBoost model. Overall, its accuracy was fairly high, both in terms of the standard validation error and the cross-validated error, highlighting our model's consistency across varied datasets.

- Train Accuracy: 0.9694
- Validation Accuracy: 0.9032
- Cross-Validation Accuracy: 0.9088 +- 0.017
- Test Accuracy: 0.8985
- Test Error: 10.15%

Additionally, as with the regression dataset, we experimented with applying the model to a PCA-flattened dataset. For this one, we flattened the data to 50 dimensions. Though the training time for the PCA model was marginally shorter than for the model trained on the full dataset, the resulting test error and accuracy were lower than the standard model, implying that CatBoost performs better when given the full set of features from the dataset.

- PCA Test Accuracy: 0.8699
- PCA Test Error: 13.01%

Finally, in order to gain a more focused understanding of the data, we plotted both confusion matrices and top feature importances.

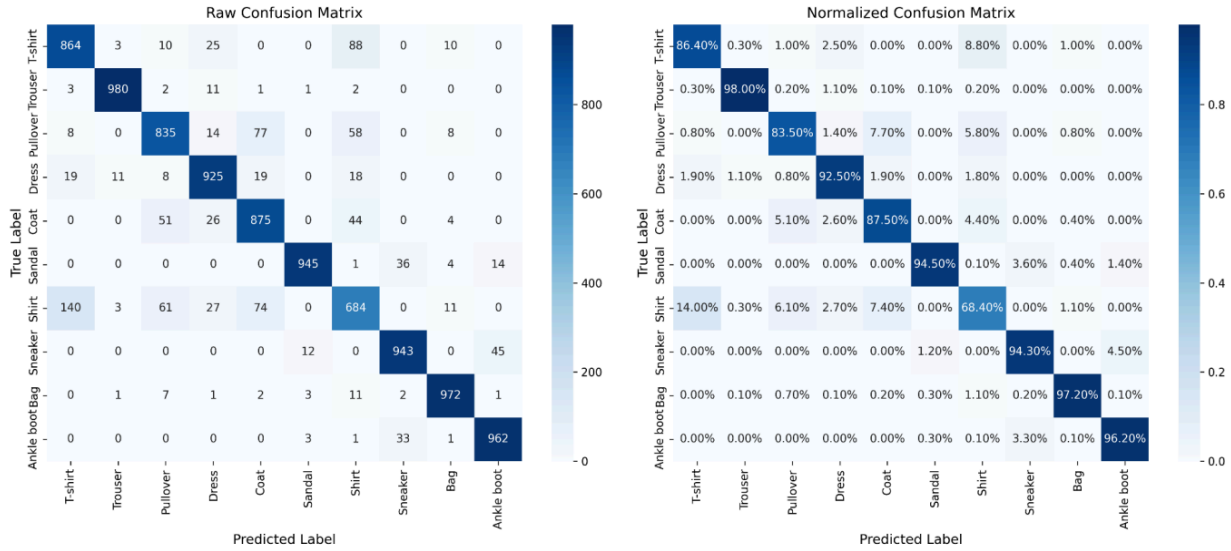

Fig. 4: Confusion Matrices.

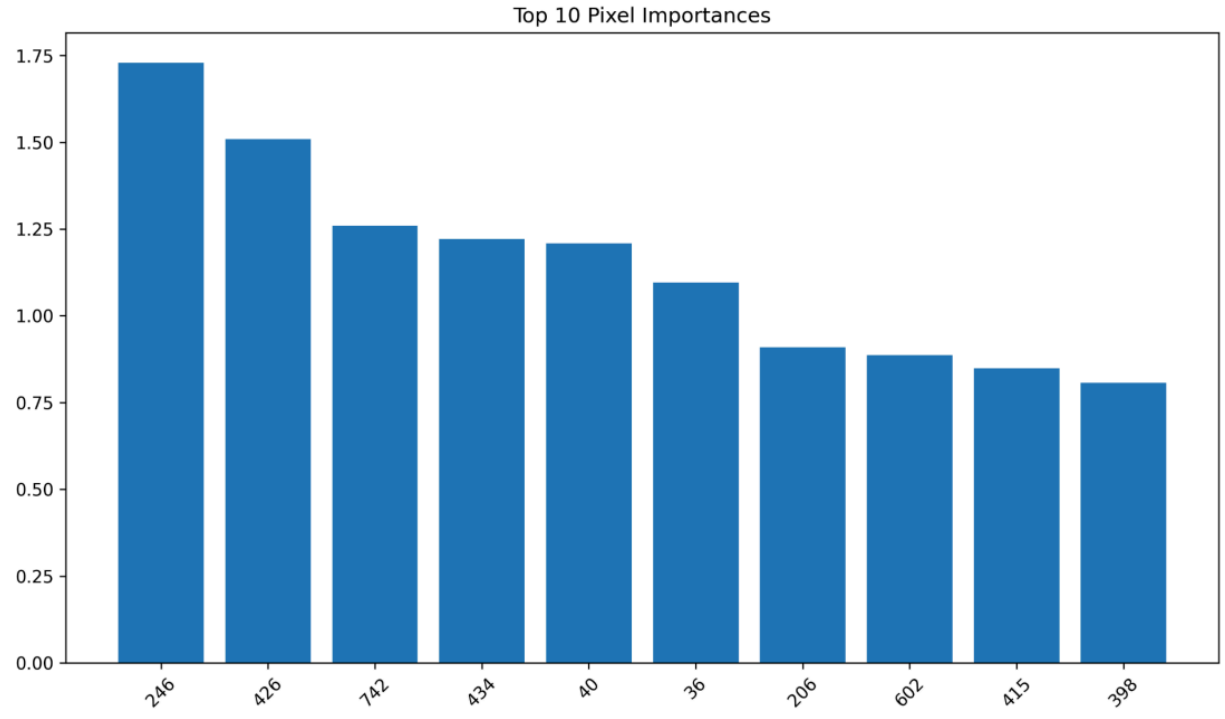

Fig. 5: Top 10 Important Features.

From these visualizations, we were able to gather that shape and contour- that is, the pixels closer to the outer edges of whatever garment was pictured- was the most relevant factor in determining classifications. This is because we observed that similarly-shaped garments were

the most easily confused, and because we saw that the most important pixels were closer to the edges of the images, based on the pixel number.

Overall, CatBoost showcases strong performance on Fashion-MNIST, reaching almost 90% accuracy with clear interpretability through feature importance analysis and stable generalization when trained with a sufficient number of trees. While this demonstrates that boosted decision trees can handle high-dimensional image-like data surprisingly well, the model itself still falls short of convolutional architectures due to its lack of spatial inductive bias and limited ability to exploit local pixel structure.

Section 3: Conclusions

The main thing we learned about CatBoost is that its unique ordered boosting characteristic allows for high numbers of iterations without overfitting- as such, we initially underestimated the optimal number of iterations. With more time, our team would have likely attempted to test the limits of this model by increasing the number of iterations to the point of nearly overfitting. Our validation accuracy of around 90% would likely be higher with a greater number of trees.

We also learned that CatBoost is not compatible with dataset manipulations and simplifications like PCA. CatBoost treats each feature individually, and as such extracts any and all useful information from features individually. Removing any feature or reducing dimensionality thus has a noticeable effect on performance.

Finally, we learned that the nonlinear nature of CatBoost made it highly adaptable- it was able to model nonlinear data, such as the California Housing dataset, with far more fidelity compared to linear baselines.

In conclusion, CatBoost proved to be a robust and useful framework for both regression and classification tasks. Its built-in native handling of categorical data and data imputation made it possible to use many types of datasets as training data, with some tolerance for missing features. Its resistance to overfitting and stable cross-validation scores lend itself towards diverse and highly varied datasets. All of these features make it a unique framework that fills a specific niche in contemporary machine learning.