Week 7 - Develop Second Modeling Approach

Maria Alice Fagundes Vieira

Aritra Ray

- **The Model Approach**

The model approach we chose for this week is the XGBoost model, which is a gradient boosting method based on decision trees. This is a powerful and efficient method that is effective for tabular data and its capabilities allows it to handle sparse data while providing high accuracy predictions with optimized memory usage, which was one of our concerns since the data we are working with is big. The Santander dataset is complex with multiple product labels (24 columns), and XGBoost's handling of non-linearity and feature interactions can prove to be beneficial to us. Its gradient boosting framework iteratively improves predictions by minimizing errors of previous models, leading to strong predictive power. This makes it particularly effective for ranking items and generating personalized recommendations.

XGBoost is designed for speed and scalability, making it suitable for large-scale recommendation systems and it can provide insights into feature importance, which is valuable for understanding factors driving recommendations and refining the system. This interpretability can help in feature selection and model optimization.

- **Complexity of the modeling approach**

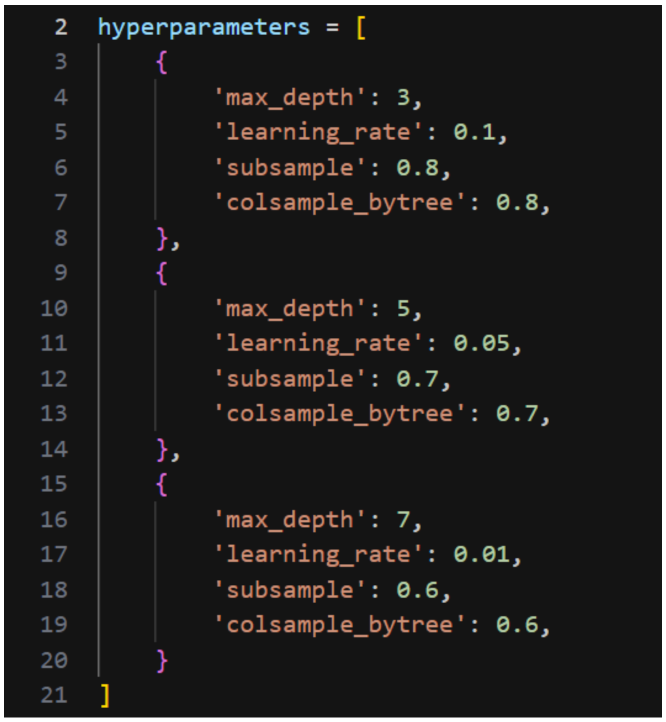
The modeling approach demonstrates a moderate level of complexity, utilizing for loop to create a separate XGBoost models for each product and incorporating early stopping to prevent overfitting, which adds a layer of complexity to the training process. The model builds a series of decision trees sequentially, with each new tree correcting errors made by previously trained trees. This naturally adds complexity compared to simpler models like logistic regression or single decision trees.

The implementation of early stopping further sophisticates the training process. By monitoring the model's performance on a validation set and stop training when performance stops improving, we're adding an adaptive element to the learning process. This dynamic approach to determining the optimal number of boosting rounds for each product model adds another layer of complexity and helps in preventing overfitting. The hyperparameters also have one extra parameter than when compared to last week while the evaluation process was similar, where we calculated ROC AUC using probabilities and F1 Score and Confusion Matrix based on the binary predictions.

- **Hyperparameters Evaluated**

This week we evaluated 4 different parameters: max_depth, learning_rate, subsample, and colsample_bytree.

1. max_depth: Controls the maximum depth of trees. Evaluating different depths (3, 5, 7) helps balance between model complexity and potential overfitting.
2. learning_rate: Affects how quickly the model learns. Testing different rates (0.1, 0.05, 0.01) helps find the optimal learning speed. A smaller learning rate often leads to better performance but requires more iterations to converge.
3. subsample: Determines the fraction of samples used for training in each boosting round. This helps in controlling overfitting by introducing randomness in the training process. We used 0.8, 0.7, 0.6.
4. colsample_bytree: Controls the fraction of features used at each split. Adjusting this (0.8, 0.7, 0.6) can help create more robust models by introducing randomness, like the subsample.



```
hyperparameters = [
    {
        'max_depth': 3,
        'learning_rate': 0.1,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
    },
    {
        'max_depth': 5,
        'learning_rate': 0.05,
        'subsample': 0.7,
        'colsample_bytree': 0.7,
    },
    {
        'max_depth': 7,
        'learning_rate': 0.01,
        'subsample': 0.6,
        'colsample_bytree': 0.6,
    }
]
```

- **Model Performance Metrics**

This week we used the same metrics as last week: ROC AUC, F1 Score and Confusion Matrix

ROC AUC: Suitable for binary classification tasks like product recommendations. It measures the model's ability to distinguish between classes across various thresholds, which is crucial in a recommendation system where you might want to adjust the threshold for different products.

F1 Score: This balances precision and recall, which is important in a recommendation system where you want to minimize both false positives and false negatives.

Confusion Matrix: This provides a detailed breakdown of true positives, false positives, true negatives, and false negatives. In a product recommendation context, this can help us understand which products are being over or under recommended.

- **Training and Validation Metrics Across All 3 Variations**

Variation 1: Shows a Train ROC AUC of 0.918933 and a Val ROC AUC of 0.904372. The F1 Scores are relatively low, with 0.144641 for training and 0.132815 for validation. Variation 1 showed the worst performance between all 3 variations, but still shows a good balance between training and validation performance.

Variation 2: Exhibits the highest Train ROC AUC of 0.926318 and Val ROC AUC of 0.923959, along with the highest F1 Scores (Train: 0.150042, Val: 0.139032). This suggests that variation 2 achieves the best balance between precision and recall, maintaining strong performance on unseen data.

Variation 3: Has the highest Train ROC AUC at 0.933340 but a lower Val ROC AUC of 0.884652, indicating potential overfitting. The F1 Scores are also lower (Train: 0.135798, Val: 0.117693), suggesting less effective predictions compared to other variations. The loss between train and validation predictions are also the highest among all 3 variations.

Variation	Train ROC AUC	Val ROC AUC	Train F1 Score	Val F1 Score
Variation 1	0.918933	0.904372	0.144641	0.132815
Variation 2	0.926318	0.923959	0.150042	0.139032
Variation 3	0.93334	0.884652	0.135798	0.117693

- **Identify the Best Model For the Week**

The best model for this week is Variation 2, since it performs better in terms of validation performance, achieving the highest scores in both F1 Score and ROC AUC, indicating better fit an generalization to unseen data. This model shows good performance across all metrics and demonstrates a good balance between training and validation scores, indicating it generalizes well to unseen data.

Overall, we see this week that we had a small loss between training and validation dataset predictions which shows a good generalization of the model.

APPENDIX

Training Dataset				
Variation	**Product**	**ROC AUC**	**F1 Score**	**Confusion Matrix**
Variation 1	savings_acct	0.993083	0	[[706792, 0], [74, 0]]
	guarantees	0.999966	0	[[706850, 0], [16, 0]]
	current_acct	0.808806	0.804876	[[171753, 105207], [69525, 360381]]
	derivada_acct	0.967359	0	[[706603, 0], [263, 0]]
	payroll_acct	0.896745	0.013931	[[667803, 58], [38731, 274]]
	junior_acct	0.999975	0.966054	[[699891, 394], [64, 6517]]
	mas_particular_acct	0.981853	0.015199	[[701120, 1], [5701, 44]]
	particular_acct	0.925143	0.438586	[[602473, 17660], [57410, 29323]]
	particular_plus_acct	0.921532	0.001422	[[677355, 3], [29487, 21]]
	short_term_depo	0.969419	0	[[705857, 0], [1009, 0]]
	medium_term_depo	0.94511	0	[[705817, 0], [1049, 0]]
	long_term_depo	0.936072	0.363833	[[672909, 4435], [21971, 7551]]
	e_acct	0.899611	0.274052	[[642067, 5918], [48592, 10289]]
	funds	0.93075	0	[[694099, 0], [12767, 0]]
	mortgage	0.95753	0	[[702807, 0], [4059, 0]]
	pension	0.936456	0.001558	[[700454, 0], [6407, 5]]
	loans	0.962701	0.065306	[[705207, 3], [1600, 56]]
	taxes	0.882262	0.000105	[[668644, 0], [38220, 2]]
	credit_card	0.907053	0.004812	[[675770, 8], [31013, 75]]
	securities	0.921937	0.002945	[[689233, 2], [17605, 26]]
	home_acct	0.935767	0	[[704156, 0], [2710, 0]]
	pensions_2	0.887281	0.012208	[[664206, 54], [42344, 262]]
	direct_debt	0.900405	0.158468	[[612621, 4243], [81892, 8110]]

Variation 2	savings_acct	0.993083	0	[[706792, 0], [74, 0]]
	guarantees	0.999966	0	[[706850, 0], [16, 0]]
	current_acct	0.808806	0.804876	[[171753, 105207], [69525, 360381]]
	derivada_acct	0.967359	0	[[706603, 0], [263, 0]]
	payroll_acct	0.896745	0.013931	[[667803, 58], [38731, 274]]
	junior_acct	0.999975	0.966054	[[699891, 394], [64, 6517]]
	mas_particular_acct	0.981853	0.015199	[[701120, 1], [5701, 44]]
	particular_acct	0.925143	0.438586	[[602473, 17660], [57410, 29323]]
	particular_plus_acct	0.921532	0.001422	[[677355, 3], [29487, 21]]
	short_term_depo	0.969419	0	[[705857, 0], [1009, 0]]
	medium_term_depo	0.94511	0	[[705817, 0], [1049, 0]]
	long_term_depo	0.936072	0.363833	[[672909, 4435], [21971, 7551]]
	e_acct	0.899611	0.274052	[[642067, 5918], [48592, 10289]]
	funds	0.93075	0	[[694099, 0], [12767, 0]]
	mortgage	0.95753	0	[[702807, 0], [4059, 0]]
	pension	0.936456	0.001558	[[700454, 0], [6407, 5]]
	loans	0.962701	0.065306	[[705207, 3], [1600, 56]]
	taxes	0.882262	0.000105	[[668644, 0], [38220, 2]]
	credit_card	0.907053	0.004812	[[675770, 8], [31013, 75]]
	securities	0.921937	0.002945	[[689233, 2], [17605, 26]]
	home_acct	0.935767	0	[[704156, 0], [2710, 0]]
	pensions_2	0.887281	0.012208	[[664206, 54], [42344, 262]]
	direct_debt	0.900405	0.158468	[[612621, 4243], [81892, 8110]]

Variation 3	savings_acct	0.993083	0	[[706792, 0], [74, 0]]
	guarantees	0.999966	0	[[706850, 0], [16, 0]]
	current_acct	0.808806	0.804876	[[171753, 105207], [69525, 360381]]
	derivada_acct	0.967359	0	[[706603, 0], [263, 0]]
	payroll_acct	0.896745	0.013931	[[667803, 58], [38731, 274]]
	junior_acct	0.999975	0.966054	[[699891, 394], [64, 6517]]
	mas_particular_acct	0.981853	0.015199	[[701120, 1], [5701, 44]]
	particular_acct	0.925143	0.438586	[[602473, 17660], [57410, 29323]]
	particular_plus_acct	0.921532	0.001422	[[677355, 3], [29487, 21]]
	short_term_depo	0.969419	0	[[705857, 0], [1009, 0]]
	medium_term_depo	0.94511	0	[[705817, 0], [1049, 0]]
	long_term_depo	0.936072	0.363833	[[672909, 4435], [21971, 7551]]
	e_acct	0.899611	0.274052	[[642067, 5918], [48592, 10289]]
	funds	0.93075	0	[[694099, 0], [12767, 0]]
	mortgage	0.95753	0	[[702807, 0], [4059, 0]]
	pension	0.936456	0.001558	[[700454, 0], [6407, 5]]
	loans	0.962701	0.065306	[[705207, 3], [1600, 56]]
	taxes	0.882262	0.000105	[[668644, 0], [38220, 2]]
	credit_card	0.907053	0.004812	[[675770, 8], [31013, 75]]
	securities	0.921937	0.002945	[[689233, 2], [17605, 26]]
	home_acct	0.935767	0	[[704156, 0], [2710, 0]]
	pensions_2	0.887281	0.012208	[[664206, 54], [42344, 262]]
	direct_debt	0.900405	0.158468	[[612621, 4243], [81892, 8110]]

Validation Dataset				
Variation	Product	ROC AUC	F1 Score	Confusion Matrix
Variation 1	savings_acct	0.904347	0	[[200325, 0], [8, 0]]
	guarantees	0.029764	0	[[200331, 0], [2, 0]]
	current_acct	0.813402	0.818449	[[51050, 27102], [18774, 103407]]
	derivada_acct	0.900264	0	[[200289, 0], [44, 0]]
	payroll_acct	0.897229	0.009591	[[194107, 12], [6184, 30]]
	junior_acct	0.999902	0.933043	[[199106, 108], [46, 1073]]
	mas_particular_acct	0.955139	0.000712	[[197526, 0], [2806, 1]]
	particular_acct	0.952627	0.404784	[[182926, 2344], [10646, 4417]]
	particular_plus_acct	0.94412	0	[[195857, 0], [4476, 0]]
	short_term_depo	0.947906	0	[[200009, 0], [324, 0]]
	medium_term_depo	0.901511	0	[[200152, 0], [181, 0]]
	long_term_depo	0.947198	0.266093	[[195094, 578], [3857, 804]]
	e_acct	0.914806	0.146672	[[190199, 574], [8758, 802]]
	funds	0.928412	0	[[198246, 0], [2087, 0]]
	mortgage	0.96995	0	[[199812, 0], [521, 0]]
	pension	0.941016	0	[[199454, 0], [879, 0]]
	loans	0.931207	0.023121	[[199991, 4], [334, 4]]
	taxes	0.886356	0	[[195396, 0], [4937, 0]]
	credit_card	0.932572	0.001829	[[195963, 6], [4360, 4]]
	securities	0.924884	0	[[197281, 0], [3052, 0]]
	home_acct	0.926991	0	[[200015, 0], [318, 0]]
	pensions_2	0.891483	0.007692	[[193599, 10], [6698, 26]]
	direct_debt	0.905907	0.09495	[[183640, 592], [15269, 832]]

Variation 2	savings_acct	0.904347	0	[[200325, 0], [8, 0]]
	guarantees	0.029764	0	[[200331, 0], [2, 0]]
	current_acct	0.813402	0.818449	[[51050, 27102], [18774, 103407]]
	derivada_acct	0.900264	0	[[200289, 0], [44, 0]]
	payroll_acct	0.897229	0.009591	[[194107, 12], [6184, 30]]
	junior_acct	0.999902	0.933043	[[199106, 108], [46, 1073]]
	mas_particular_acct	0.955139	0.000712	[[197526, 0], [2806, 1]]
	particular_acct	0.952627	0.404784	[[182926, 2344], [10646, 4417]]
	particular_plus_acct	0.94412	0	[[195857, 0], [4476, 0]]
	short_term_depo	0.947906	0	[[200009, 0], [324, 0]]
	medium_term_depo	0.901511	0	[[200152, 0], [181, 0]]
	long_term_depo	0.947198	0.266093	[[195094, 578], [3857, 804]]
	e_acct	0.914806	0.146672	[[190199, 574], [8758, 802]]
	funds	0.928412	0	[[198246, 0], [2087, 0]]
	mortgage	0.96995	0	[[199812, 0], [521, 0]]
	pension	0.941016	0	[[199454, 0], [879, 0]]
	loans	0.931207	0.023121	[[199991, 4], [334, 4]]
	taxes	0.886356	0	[[195396, 0], [4937, 0]]
	credit_card	0.932572	0.001829	[[195963, 6], [4360, 4]]
	securities	0.924884	0	[[197281, 0], [3052, 0]]
	home_acct	0.926991	0	[[200015, 0], [318, 0]]
	pensions_2	0.891483	0.007692	[[193599, 10], [6698, 26]]
	direct_debt	0.905907	0.09495	[[183640, 592], [15269, 832]]

Variation 3	savings_acct	0.904347	0	[[200325, 0], [8, 0]]
	guarantees	0.029764	0	[[200331, 0], [2, 0]]
	current_acct	0.813402	0.818449	[[51050, 27102], [18774, 103407]]
	derivada_acct	0.900264	0	[[200289, 0], [44, 0]]
	payroll_acct	0.897229	0.009591	[[194107, 12], [6184, 30]]
	junior_acct	0.999902	0.933043	[[199106, 108], [46, 1073]]
	mas_particular_acct	0.955139	0.000712	[[197526, 0], [2806, 1]]
	particular_acct	0.952627	0.404784	[[182926, 2344], [10646, 4417]]
	particular_plus_acct	0.94412	0	[[195857, 0], [4476, 0]]
	short_term_depo	0.947906	0	[[200009, 0], [324, 0]]
	medium_term_depo	0.901511	0	[[200152, 0], [181, 0]]
	long_term_depo	0.947198	0.266093	[[195094, 578], [3857, 804]]
	e_acct	0.914806	0.146672	[[190199, 574], [8758, 802]]
	funds	0.928412	0	[[198246, 0], [2087, 0]]
	mortgage	0.96995	0	[[199812, 0], [521, 0]]
	pension	0.941016	0	[[199454, 0], [879, 0]]
	loans	0.931207	0.023121	[[199991, 4], [334, 4]]
	taxes	0.886356	0	[[195396, 0], [4937, 0]]
	credit_card	0.932572	0.001829	[[195963, 6], [4360, 4]]
	securities	0.924884	0	[[197281, 0], [3052, 0]]
	home_acct	0.926991	0	[[200015, 0], [318, 0]]
	pensions_2	0.891483	0.007692	[[193599, 10], [6698, 26]]
	direct_debt	0.905907	0.09495	[[183640, 592], [15269, 832]]