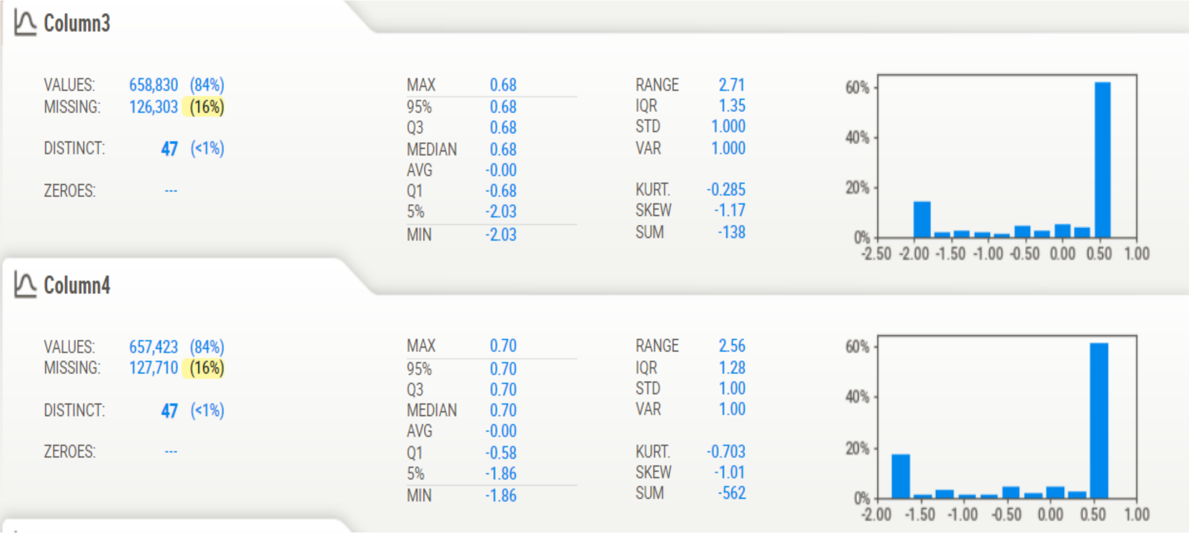
Intelligent Finance Function Predictor: **Insight Report.**

-Ashmita Singh & Aarti Thakre

Solving the Column Conundrum:



Column3

VALUES:	658,830 (84%)	MAX	0.68	RANGE	2.71
MISSING:	126,303 (16%)	95%	0.68	IQR	1.35
		Q3	0.68	STD	1.000
DISTINCT:	47 (<1%)	MEDIAN	0.68	VAR	1.000
		AVG	-0.00		
ZEROES:	---	Q1	-0.68	KURT.	-0.285
		5%	-2.03	SKEW	-1.17
		MIN	-2.03	SUM	-138

Column4

VALUES:	657,423 (84%)	MAX	0.70	RANGE	2.56
MISSING:	127,710 (16%)	95%	0.70	IQR	1.28
		Q3	0.70	STD	1.00
DISTINCT:	47 (<1%)	MEDIAN	0.70	VAR	1.00
		AVG	-0.00		
ZEROES:	---	Q1	-0.58	KURT.	-0.703
		5%	-1.86	SKEW	-1.01
		MIN	-1.86	SUM	-562

Some columns were categorical for sure, but who was feigning? Column 3 and 4. Let us show you this in more detail,

Training data:

'Column3': '-2.028572085775468 - 0.6781394378315789',

'Column4': '-1.855728261270304 - 0.7014034666794821'

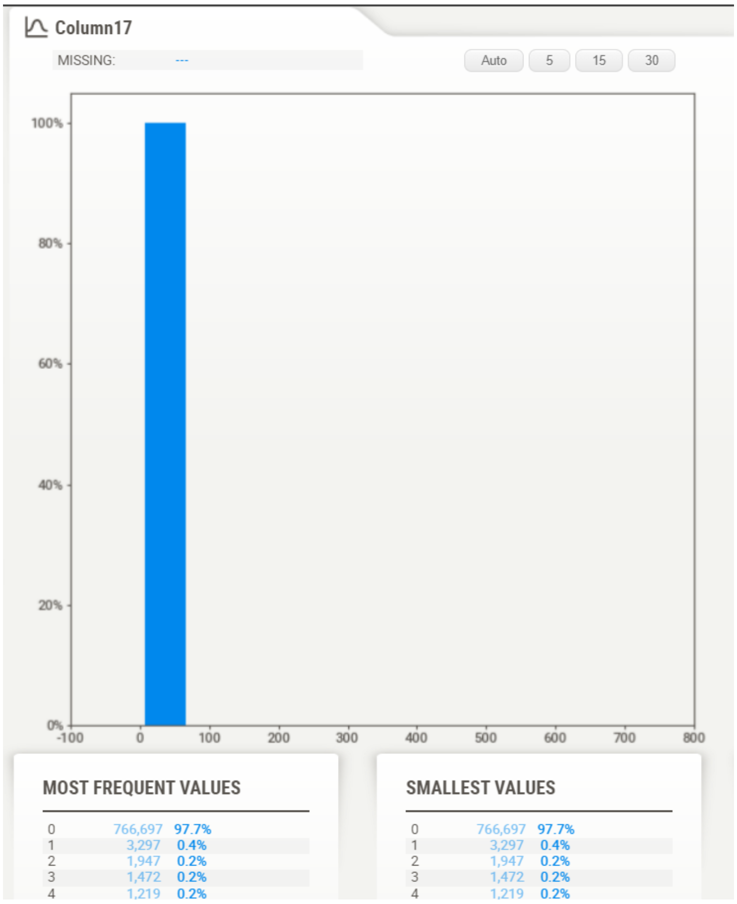
Testing data:

'Column3': '-2.028572085775468 - 0.6781394378315789',

'Column4': '-1.855728261270304 - 0.7014034666794821'

Found the stunning connection and that’s why we treated them as co-dependent categories.

And then we looked at Column 17 which had 153 and 124 unique values in training and testing dataset respectively, with the range as 0-728, so on further analysing, we came to the conclusion that Column 17’s default state must be 0, and the other very high values could possibly be rare occurrences.



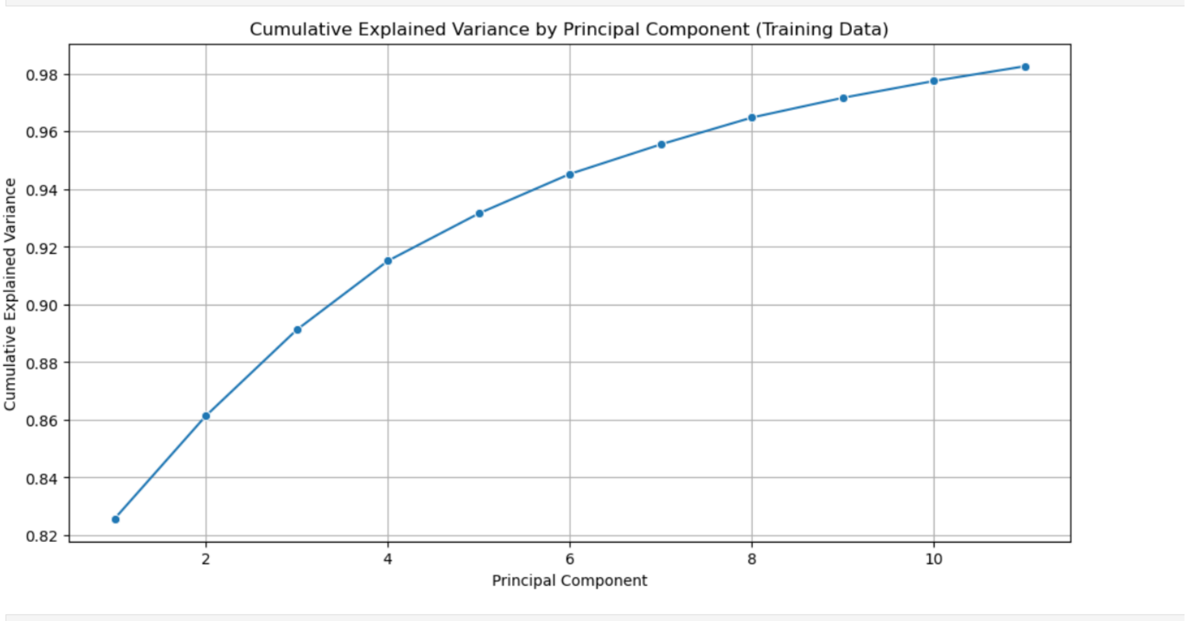
MOST FREQUENT VALUES

0	766,697	97.7%
1	3,297	0.4%
2	1,947	0.2%
3	1,472	0.2%
4	1,219	0.2%

SMALLEST VALUES

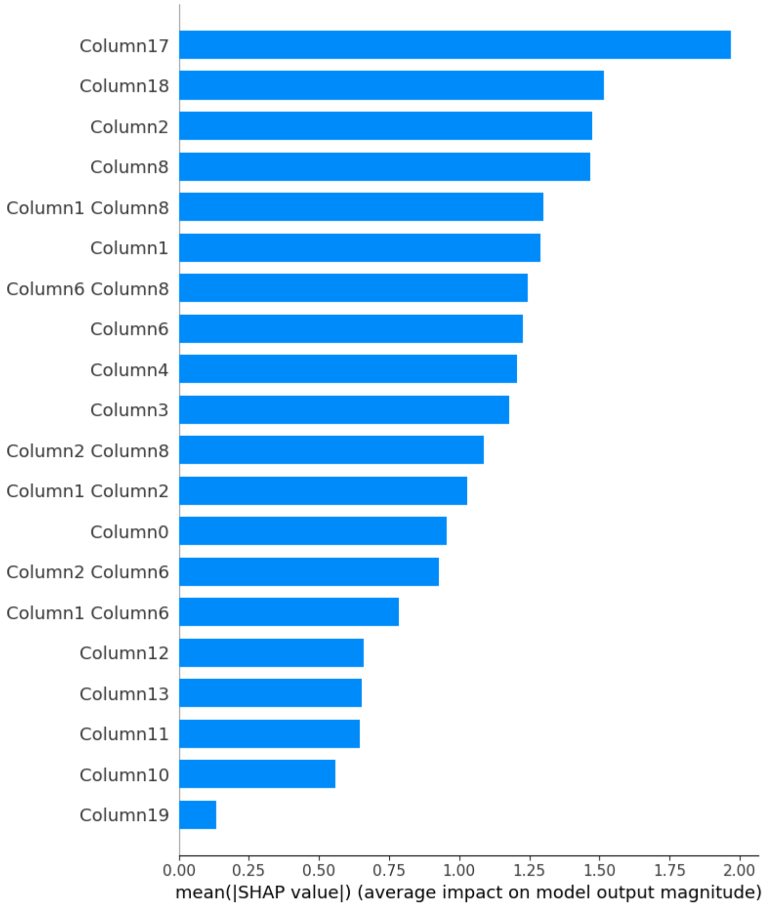
0	766,697	97.7%
1	3,297	0.4%
2	1,947	0.2%
3	1,472	0.2%
4	1,219	0.2%

PCA Component's important information retaining, amount to 98% percent like we chose to.



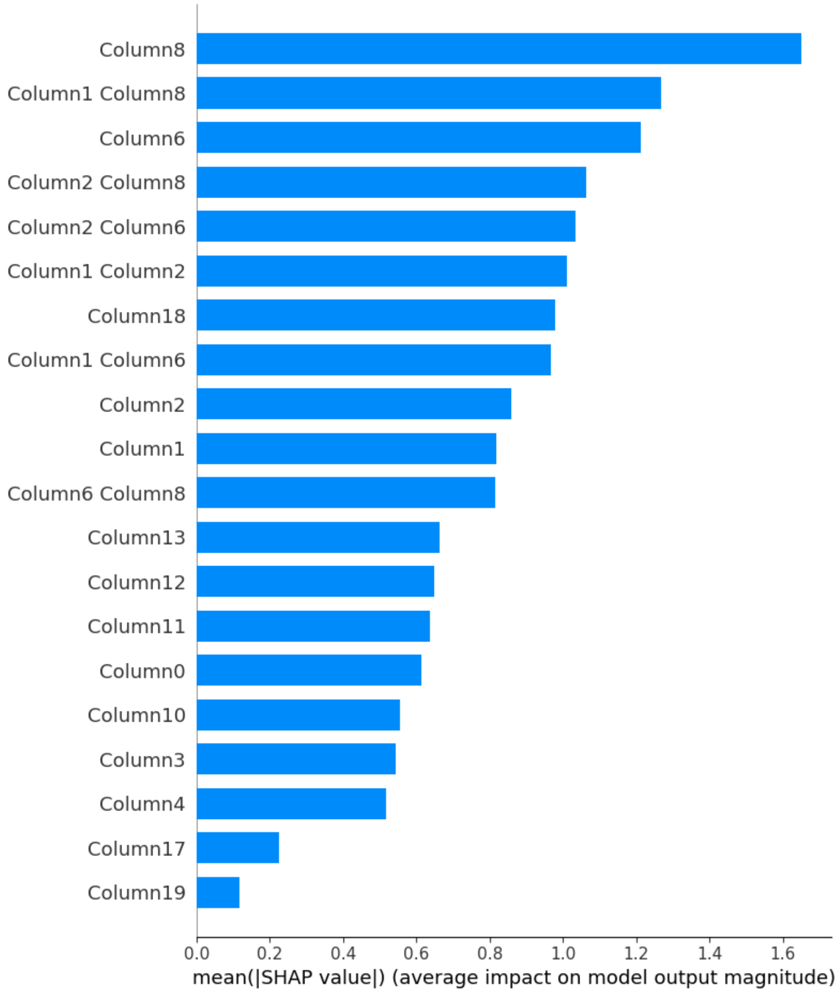
SHAP Interpretations:

For XGBoost, Column 17 really played an important role in the prediction of our target variable. Likewise Column 18, Column 2 etc.



For SGDClassifier however, the differences were there.

WHY?



XGBoost:

- **Tree-Based Method**: XGBoost builds decision trees, which capture complex interactions between features. It evaluates feature importance based on how much each feature splits the data and reduces impurity.
- **Non-Linear Relationships**: XGBoost can capture non-linear relationships, meaning it might highlight different features that interact in complex ways to predict the target variable.

SGDClassifier:

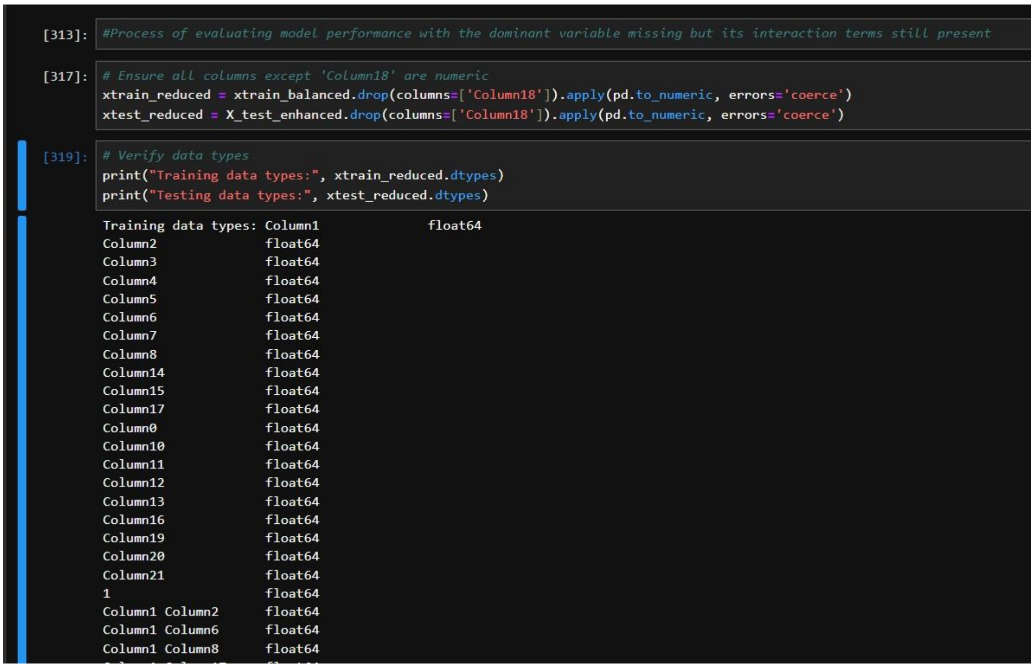
- **Linear Method**: Stochastic Gradient Descent is a linear model, meaning it evaluates feature importance based on linear relationships with the target variable.
- **Coefficient Magnitude**: Features with the highest absolute coefficients in the linear equation are considered the most important.

Key Differences:

- **Nature of Relationships**: XGBoost's ability to capture non-linear interactions can lead it to prioritize different features compared to the linear focus of SGDClassifier.

- **Evaluation Criteria**: The criteria for feature importance in tree-based models (splitting ability) versus linear models (coefficient magnitude) inherently differ, causing a variation in feature prioritization.

Another highlight is that our model performed really well even with the absence of the most dominant variable which was column 18. (Performed in another file)

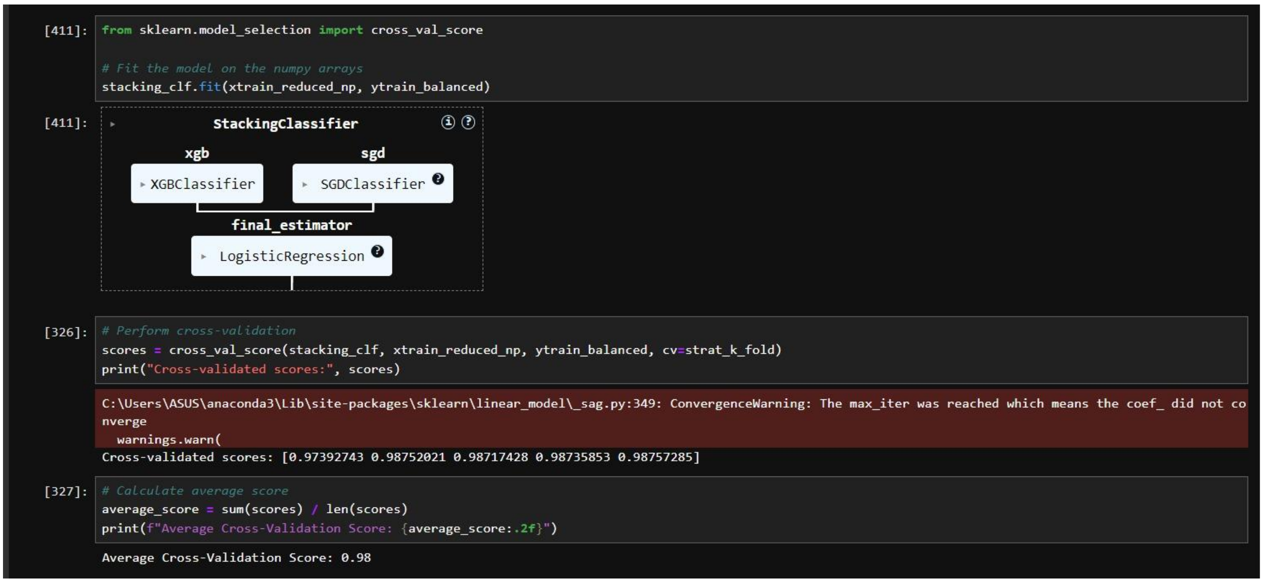


```
[313]: #Process of evaluating model performance with the dominant variable missing but its interaction terms still present

[317]: # Ensure all columns except 'Column18' are numeric
       xtrain_reduced = xtrain_balanced.drop(columns=['Column18']).apply(pd.to_numeric, errors='coerce')
       xtest_reduced = X_test_enhanced.drop(columns=['Column18']).apply(pd.to_numeric, errors='coerce')

[319]: # Verify data types
       print("Training data types:", xtrain_reduced.dtypes)
       print("Testing data types:", xtest_reduced.dtypes)

       Training data types: Column1             float64
       Column2              float64
       Column3              float64
       Column4              float64
       Column5              float64
       Column6              float64
       Column7              float64
       Column8              float64
       Column14             float64
       Column15             float64
       Column17             float64
       Column0              float64
       Column10             float64
       Column11             float64
       Column12             float64
       Column13             float64
       Column16             float64
       Column19             float64
       Column20             float64
       Column21             float64
       1                    float64
       Column1 Column2      float64
       Column1 Column6      float64
       Column1 Column8      float64
```



```
[411]: from sklearn.model_selection import cross_val_score

       # Fit the model on the numpy arrays
       stacking_clf.fit(xtrain_reduced_np, ytrain_balanced)

[411]:
```

```
[326]: # Perform cross-validation
       scores = cross_val_score(stacking_clf, xtrain_reduced_np, ytrain_balanced, cv=strat_k_fold)
       print("Cross-validated scores:", scores)

       C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\linear_model\_sag.py:349: ConvergenceWarning: The max_iter was reached which means the coef_ did not co
       nverge
         warnings.warn(
       Cross-validated scores: [0.97392743 0.98752021 0.98717428 0.98735853 0.98757285]

[327]: # Calculate average score
       average_score = sum(scores) / len(scores)
       print(f"Average Cross-Validation Score: {average_score:.2f}")

       Average Cross-Validation Score: 0.98
```

So, these are the insights overall:

Insights from Our Model Development:

1. **Accurate Data Classification**:

   - **Insight**: Differentiating between categorical and numerical columns using missing values, range, and entropy ensured precise preprocessing.
   - **Impact**: This step improved data integrity and facilitated appropriate handling of each data type.

2. **Effective Missing Value Handling**:
   - **Insight**: Using median imputation for numerical columns and mode imputation for categorical columns maintained data consistency without introducing bias.
   - **Impact**: Enabled robust data processing, preserving the essential information while managing incomplete data.
3. **Balanced Outlier Management**:
   - **Insight**: Removing 5.60% outliers using the Z-Score method struck the right balance, filtering out noise while keeping significant data.
   - **Impact**: Enhanced model focus on central patterns, leading to more accurate predictions.
4. **Selective Feature Enhancement**:
   - **Insight**: SelectKBest helped in pinpointing crucial features, while polynomial features captured complex interactions, enriching the dataset.
   - **Impact**: Boosted model’s ability to understand intricate relationships, improving overall predictive power.
5. **Addressing Class Imbalance**:
   - **Insight**: Implementing SMOTE to generate synthetic samples for the minority class balanced the dataset effectively.
   - **Impact**: Improved model generalization and performance, particularly for the minority class.
6. **Optimal Dimensionality Reduction**:
   - **Insight**: Applying PCA retained 98% of variance while reducing the feature space.
   - **Impact**: Enhanced computational efficiency and reduced the risk of overfitting, ensuring the model remained focused on significant features.
7. **Diverse Model Selection**:
   - **Insight**: Choosing XGBoost and SGDClassifier as base models combined with Logistic Regression as the meta-model provided a robust and balanced ensemble.

- **Impact**: Leveraged the strengths of different algorithms, resulting in a highly accurate and versatile model.

8. **Efficient Hyperparameter Tuning**:

    - **Insight**: Conducting hyperparameter tuning on a 10% subset of data using grid search for SGD and random search for XGBoost balanced precision and computational efficiency.
    - **Impact**: Optimized model performance without exhaustive resource consumption.

9. **Comprehensive Model Evaluation**:

    - **Insight**: Using classification reports, AUC-ROC, and SHAP values for PCA components provided a well-rounded evaluation.
    - **Impact**: Ensured a thorough understanding of the model's performance and transparency in predictions.

References

1. Microsoft Copilot, personal communication, 2024.

Plagiarism Declaration

I hereby declare that this project report is my own work and has been written by me in my own words. Any information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is provided. I have not copied in part or in whole or otherwise plagiarized the work of other students and/or persons. The primary source of assistance was Microsoft Copilot, which provided guidance and insights without copying any proprietary content.