Explainable Validation and Feature Reduction in Machine Learning

Fenrir Badorf
Dept. of Computer Science
Loyola University Maryland
Baltimore, USA
bkbadorf@loyola.edu

Francis Durso
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, USA
francis.durso@nist.gov

Callie Walker
Dept. of Computer Science
Loyola University Maryland
Baltimore, USA

Megan Olsen
Dept. of Computer Science
Loyola University Maryland
Baltimore, USA
mmolsen@loyola.edu

M S Raunak
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, USA
raunak@nist.gov

D. Richard Kuhn
National Security Institute
Virginia Tech
Arlington, VA, USA
rickkuhn@vt.edu

***Abstract*—Although there are many available approaches for designing machine learning models and experiments, not all of these approaches are explainable or interpretable. Combination frequency differencing (CFD) provides information on the relationship between feature values and predicted class in the form of distinguishing combinations. In this paper we investigate how distinguishing combinations in the training and validation datasets may affect the effectiveness of the trained model. We also investigate how distinguishing combinations can be used for a more explainable approach to feature reduction. Notably, results showed that CFD performed comparably to principal component analysis, an established method for reducing features, while providing better explainability. Thus CFD shows promise for providing a unified, interpretable framework for dataset construction, model validation, and feature reduction.**

***Index Terms*—Combinatorial analysis, machine learning, validation, feature reduction, explainability**

I. Introduction

Machine learning models increasingly underpin critical decisions in domains such as healthcare, finance, and cybersecurity, where predictive performance, reliability, and interpretability all play critically important roles. As models grow more complex and datasets larger and noisier, it becomes harder to understand which data points and features actually drive the learned decision boundaries. Traditional approaches focus on global metrics like accuracy or F1 score, on generic pre-processing steps such as random sampling, and on standard dimensionality reduction approaches like Principle Component Analysis (PCA). These methods, however, provide limited insight into why particular instances matter or how individual feature interactions contribute to model behavior. There is thus a strong need for techniques that not only improve efficiency and robustness, but also expose the structure of the data in ways that support more rigorous validation and interpretability.

Combinatorial approaches such as Combination Frequency Differencing (CFD) [1], along with other related measures and processes, have been applied to dataset improvement [2] and understanding [3], [4] as well as towards model *explainability* in machine learning [5], [6]. Explainability is beneficial and desired in machine learning as it can increase the trustworthiness of predictions when it is clearer how the model has made its predictions, and can also make it clearer when something has gone wrong [7]–[9]. For many modern machine learning models, however, explainability can be difficult to achieve.

In our prior work, we showed that combination frequency differencing can aid in determining which rows of data are most useful for training a supervised machine learning model [2]. We used the dataset's Distinguishing Combinations (DC), identified as value combinations associated more strongly with a particular class, to determine which rows to keep and which to discard, creating smaller datasets that train models equally well or better than the original dataset. We showed that DCs can guide instance selection by prioritizing rows that include these combinations when constructing reduced training datasets, thereby preserving or even improving model performance while substantially shrinking dataset size. In addition, DC-based selection provides a direct link between specific feature interactions and the examples retained, making the reduction process more transparent and explainable than other black-box algorithmic approaches. While DCs have been used to decide which instances to keep, their broader role within the full lifecycle of a dataset used for machine learning remains underexplored.

In this paper, we ask how DCs function not just as a filter for training data, but as structural elements that shape model behavior across both training and validation. We investigate the relative importance of distinguishing combinations being present in the training set versus in the validation or test sets. Building on these insights, we then explore whether DCs can be leveraged as a more explainable approach to dimensionality reduction. Rather than eliminating features based on opaque criteria, we analyze how feature interactions embodied in DCs

NSF #1626262 funded the computing resources used in our experiments.

can guide feature selection or aggregation. By systematically relating DCs to both instance selection and feature-level explanations, our goal is to position CFD-based analysis as a unified, interpretable framework for dataset construction, model validation, and feature reduction.

II. Combination Frequency Difference

Combination Frequency Differencing (CFD) [10] [11] is a data analysis approach that leverages feature–value interaction patterns in the dataset to deduce some characteristic information about the data. For example, it can be used to identify training instances that are most informative for class discrimination. The method builds on combinatorial coverage concepts by examining the frequency of t-way feature-value combinations across class partitions of a dataset and identifying those combinations whose frequency of occurrence differs meaningfully between classes. CFD was developed from the intuitive notion that classes may be identified by clusters of feature values that are more strongly associated with one class than they are with other classes.

Let a dataset contain n features, $\{f_1, f_2, \ldots, f_n\}$ where each feature f_i takes values from a finite domain $\mathcal{D}_i$.

t-way Combination: A *t-way combination* is any subset of t distinct features selected from $\{f_1, \ldots, f_n\}$.

t-tuple of Values: Given a t-way combination $\{f_{i_1}, \ldots, f_{i_t}\}$, a t-tuple is a specific assignment of values

$$x = (v_{i_1}, \ldots, v_{i_t}), \quad v_{i_j} \in \mathcal{D}_{i_j}.$$

For a fixed interaction strength t, the number of possible value tuples grows combinatorially with the number of features and their domain sizes. Traditional combinatorial coverage measures whether such tuples appear in the dataset; however, simple coverage calculation does not quantify how frequently they occur within each class. CFD addresses this limitation by incorporating class-conditional frequency information.

Consider a binary classification problem with class partitions C (class of interest) and N (non-class). Let $fr(x, Y)$ denote the frequency (i.e., number of instances) in which tuple x appears in dataset partition $Y \in \{C, N\}$. Let us now define distinguishing combination.

Distinguishing Combination: A t-tuple x is said to be *distinguishing* for class C if

$$fr(x, C) > T \cdot fr(x, N) \tag{1}$$

where $T \geq 1$ is a user-defined threshold parameter controlling the required strength of class bias.

When $T = 1$, any tuple that occurs more frequently in C than in N is considered distinguishing. Larger values of T restrict selection to tuples with stronger discriminatory skew.

We collectively refer to the following combinations as *Distinguishing Combinations (DCs)*:

- **Unique Distinguishing Combinations**: tuples for which $fr(x, N) = 0$ and $fr(x, C) > 0$, or for which $fr(x, C) = 0$ and $fr(x, N) > 0$.
- **Notable Distinguishing Combinations**: tuples present in both classes and satisfying the frequency threshold inequality.

CFD identifies higher-order feature interactions that exhibit asymmetric distributions across class partitions. Unlike marginal feature ranking methods or geometric boundary-based instance selection approaches, CFD explicitly models multi-feature interaction structure.

Some properties of the CFD metric include:

1) **Model-agnostic operation**: The metric works on raw training data. No intermediate classifier is required, and it is not necessary to tailor the approach for different ML models.
2) **Frequency sensitivity**: It allows differentiation between exclusive (unique) patterns and frequent class-skewed interactions.
3) **Interaction awareness**: Can theoretically support an arbitrary interaction strength t. Current tool implementation supports up to 6-way interactions.

Thus, CFD can capture localized regions of the feature space where class separation is expressed through interaction effects.

A. From Distinguishing Combinations to Instance Selection

After identifying DCs, each training instance can be evaluated according to:

- Whether it contains at least one DC,
- The total number of DCs it contains,
- The interaction strength t associated with its DCs.

Instances containing DCs encode interaction patterns that are statistically biased toward a class and are therefore hypothesized to contribute directly to defining discriminative structure. Selecting such instances yields reduced datasets that may preserve class-separating interaction information while eliminating redundant or weakly informative samples [2].

Features of the CFD tool include CFD computation for different strengths ($t = 1, 2, \ldots, 6$) and the ability to optionally exclude lower-order DCs from higher-order DCs to isolate interaction-specific effects ("filtering"). Later we will refer to unfiltered and filtered dataset to specify whether lower ordered DCs were removed from the dataset or not. A limitation of CFD is that it expects discrete data values. Continuous features are required to be discretized prior to the frequency analysis.

CFD outputs a structured set of distinguishing combinations by class, along with associated frequency statistics. These outputs provide both interpretability via explicit interaction patterns and actionable guidance for dataset construction.

III. Dataset

We use the UCI Adult Census dataset [12] for all experiments. The dataset contains 48,842 instances and 14 features. It has binary labels and illustrates several common sources of dataset complexity: the presence of mixed feature types (categorical and numerical), class imbalance, and non-linear class separability. The labels denote income categories: class 0 corresponds to income at or below \$50K, and class 1 corresponds to income above \$50K.

We perform a series of pre-processing steps to clean the data: filtering to include only U.S. samples, dropping the *education* column while retaining *education_num*, removing the *capital_gain*, *capital_loss*, and *country* columns, and discarding rows with missing values. After these steps, the dataset contains 27,504 rows. We then divide the data into training and test sets, with the initial training set containing 22,003 rows (16,407 in Class 0; 5,596 in Class 1). The test set is held out until assessment of the final models. All analysis in this paper is on this training data.

Discretization ("binning") is applied solely for conducting CFD analysis to identify distinctive feature combinations. Four numerical columns are binned: *age*, *fnlwgt*, *education_num*, and *hrs_worked_per_week*. Six categorical features are one-hot encoded for machine learning:

Sex (2 values): Male (14,863), Female (7,140)

Race (5 values): White (19,368), Black (2,105), Asian-Pac-Islander (221), Amer-Indian-Eskimo (219), Other (90)

Relationship (6 values): Husband (9,125), Not-in-family (5,700), Own-child (3,348), Unmarried (2,322), Wife (982), Other-relative (526)

Occupation (14 values): Exec-managerial (2,994), Prof-specialty (2,972), Craft-repair (2,937), Adm-clerical (2,728), Sales (2,709), Other-service (2,219), Machine-op-inspct (1,334), Transport-moving (1,193), Handlers-cleaners (946), Farming-fishing (719), Tech-support (680), Protective-serv (499), Priv-house-serv (66), Armed-Forces (7)

Marital status (7 values): Married-civ-spouse (10,213), Never-married (7,105), Divorced (3,201), Separated (677), Widowed (610), Married-spouse-absent (180), Married-AF-spouse (17)

Type of employer (7 values): Private (16,107), Self-emp-not-inc (1,877), Local-gov (1,591), State-gov (966), Self-emp-inc (761), Federal-gov (692), Without-pay (9)

One-hot encoding transforms each categorical variable into separate binary features. Although it increases the dimensionality, it prevents misleading ordinal relationships between categories. For instance, encoding occupation categories as integers 0–13 would incorrectly suggest ordered relationships such as "occupation 2 is half of occupation 4," which do not exist. None of the numerical or categorical features exhibit strong correlations with each other or with the label.

IV. DC in Validation

Our goal in this section is to determine the role of distinguishing combinations in model training and validation. In prior work, following standard practice, we used cross validation in training [2]. Consequently, each dataset in that study likely had Distinguishing Combinations (DCs) in both data training and validation steps. This setup raises the question if DCs are equally useful for both training and validation or if they are more important in one versus the other.

A. Approach

To understand the role of DCs in validation, we create the validation data by dividing the full training data into two sets:

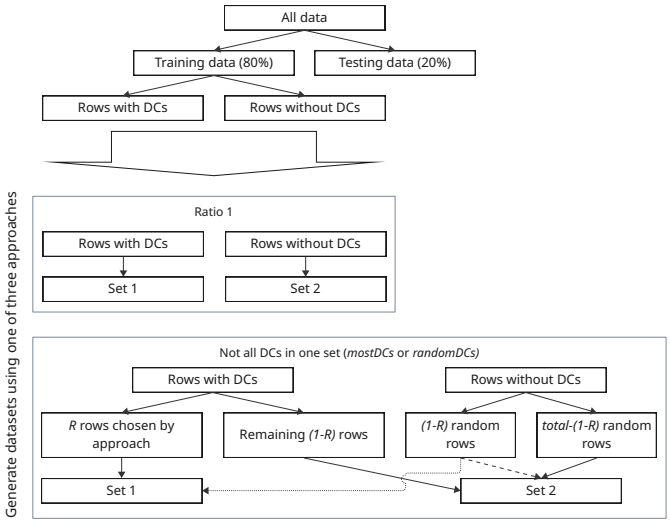


Fig. 1: Five alternative processes for generating paired sets of the training data. Each set 1 and set 2 that is generated together is used as a pair for training/validation. Dotted lines represent data that is moved only when *non-DC transfer* is utilized. Dashed lines represent data that is moved only when *non-DC transfer* is not used. All other data are used in the same way with or without *non-DC transfer*.

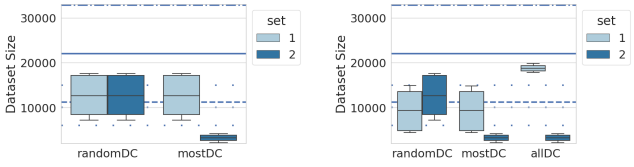
Set 1 and Set 2. First, we determine the DCs in the full training data using our CFD tool [2]. Since numerical data must be discretized, we try 4 approaches of binning: 3 or 6 bins, of either equal frequency or equal size. We only use unfiltered DCs. We use these DCs to determine which rows belong in the training data or the validation data using three approaches (Figure 1) based on our prior work [2]. Every row of the full training data is placed into either Set 1 or Set 2 for each approach. The two sets created by each approach are treated as corresponding pairs. One set of the pair will be used for training and the other for validation.

We examine the number of t-way DCs in each row. In the simplest approach (*allDCs*), rows with DCs are moved into Set 1, and those without DCs are moved into Set 2, so that Set 1 has all DC rows and the Set 2 has none. This data pair is thus used to examine an all or nothing approach.

To further examine the importance of DCs in validation, we create datasets such that both Set 1 and Set 2 have some number of rows with DCs, varied by the ratio kept in Set 1. The ratio represents a percentage of rows with DCs, resulting in R rows kept in Set 1. We generate these variations of the datasets by moving $1 - ratio$ percentage of rows with DCs from the Set 1 dataset to the Set 2 dataset:

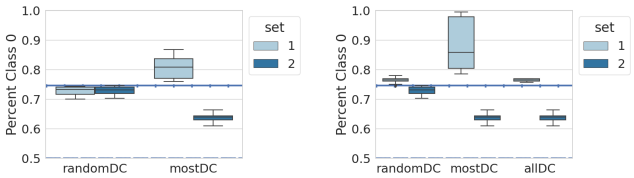
- *mostDCs*: Some number of the rows $1-R$ with the lowest number of DCs are moved from the Set 1 dataset to the Set 2 dataset.
- *randomDCs*: Some number of the rows $1 - R$ are randomly chosen to move from the Set 1 dataset to the Set 2 dataset.

Based on the above process, all rows in Set 1 will have DCs.



(a) With non-DC transfer (b) Without non-DC transfer

Fig. 2: Dataset sizes



(a) With non-DC transfer (b) Without non-DC transfer

Fig. 3: Dataset Class Imbalance

However, we also want to know if non-DC rows are valuable in conjunction with DC rows. Thus, we also implement a *non-DC transfer* strategy where the number of DC rows moved from Set 1 to Set 2 are replaced by transferring an equal number of non-DC rows from Set 2 to Set 1.

After the datasets are generated in this fashion, we train support vector machines to determine the most important role for rows with DCs in training and validation.

B. Dataset Generation Results

The process of using CFD to identify DCs and using DCs to select and move certain rows to create the pairs of datasets are summarized in Table I. The names for each approach represent the role of DCs in Set 1 of the pair. We generate *mostDCs* and *randomDCs* datasets using three ratios of the number of DC rows to keep in Set 1: $[0.25, 0.5, 0.75]$. Each of those approaches use five random number seeds to generate five instances of each pairing, as randomness is used for determining which rows to transfer.

The resulting datasets have varied dataset sizes and class imbalances. In Figure 2 we see that when non-DC transfer is used, the *randomDCs* sets are the same size, but Set 1 is noticeably bigger for *mostDCs*. *allDCs* datasets are larger in Set 1, since all rows with DCs remain in Set 1 only. Figure 3 shows that the Set 1 *mostDCs* datasets are the most imbalanced, and the Set 2 datasets without non-DC transfer are the most balanced. For the *randomDCs* approach, transfer does not affect balance for Set 2. *mostDCs* Set 1 has high class imbalance, and both *mostDCs* and *allDCs* Set 2 has low class imbalance, implying that the rows with the most DCs tend to be Class 0. In these figures we also see the size and imbalance of the comparison datasets: full training data (solid line), downsampled data (dashed), SMOTE (dotted), and random samples (dot dash). Only the down/up sampled datasets are evenly balanced.

C. Machine Learning Experimental Design

Each pair of Set 1 and Set 2 is used for training a model: first Set 1 as training and Set 2 as validation, and then Set 2 as training and Set 1 as validation. These dataset pairs are used to train a support vector machine (SVM) through a grid search in scikit-learn of potential hyperparameters: $C = [1, 10, 100, 1000]$, $\gamma = [10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}]$, kernel of radial basis function or polynomial (with a degree of 1, 2, or 3). A grid search trains models on every combination of hyperparameters, and then chooses the hyperparameters that resulted in the best validation score to train the final model. The data is scaled using the standard scaler, and processed with principal components analysis (PCA), keeping the number of components that explain 95% of the variance. The grid search uses F1 score to choose the best set of hyperparameters based on validation scores. The SVM uses balanced class weight in its calculations to offset the impact of the class imbalance.

We also create five additional datasets to compare with our datasets generated using the above process. These five datasets comprise of 1) the full training data, 2) a SMOTE upsampled dataset, 3) a randomly downsampled dataset, and 4) three random samples of data (sizes 15000, 10000, and 6000). These datasets are trained using a similar grid search but with 10-fold cross validation, which we expect to perform better than dedicated training/validation sets.

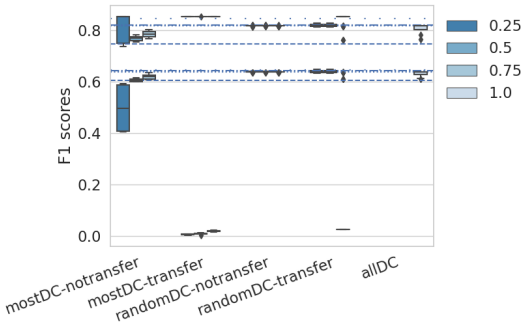
D. Machine Learning Results

Figure 4 displays the F1 scores of each trained model, with comparison datasets using the same line format as previously described. Most models perform better on Class 0 than Class 1, meaning the preference is the majority class. In general, a good model would improve on the minority class as the class imbalance is a key factor in model performance on this dataset. Set 1 as training data is more likely to outperform or meet the performance of comparison models despite generally higher class imbalance than Set 2. *mostDCs* with non-DC transfer performs the worst for Class 1 for both Sets, but particularly for Set 1 as training data. Likely it performs better with Set 2 as training data due to moving rows without DCs out of the training set and into the validation set. It is surprising that *mostDCs* without non-DC transfer works as well as it does given how strong the class imbalance is in those datasets. The *randomDCs* approach performs most consistently, implying that having a mix of rows with high and low numbers of DCs present in both training and validation datasets aids in model training. The comparison datasets all have a mix of DCs throughout training and validation as they are trained using cross validation on datasets with a mix of rows with and without DCs.

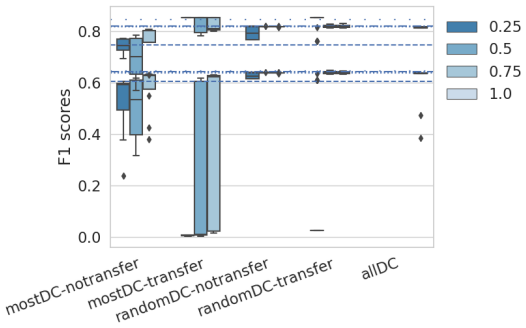
In Figure 5 we further analyze the impact of CFD choices: t-way analysis and discretization (binning) strategy. When non-DC transfer is used, most models have a higher F1 score than the full training data for Class 0 but not Class 1 (Figure 5a). The *mostDCs* models perform better when Set 1 is used as training data, especially with higher ratios. *randomDCs* can train models that are better in both classes, especially when

TABLE I: Description of the types of data existing in each generated dataset. Each row of the original training data are in either Set 1 or Set 2 in each pairing.

	allDCs	mostDCs	randomDCs
Set 1 without non-DC transfer Set 2 without non-DC transfer	all DC rows All non-DC rows	rows with most DCs All non-DC rows + rows with fewest DCs	rows with varying DCs All non-DC rows + rows with varying DCs
Set 1 with non-DC transfer Set 2 with non-DC transfer	Not applicable	rows with most DCs + some non-DC rows rows with fewest DCs + Some non-DC rows	rows with varying DCs + some non-DC rows rows with varying DCs + some non-DC rows



(a) Set 1 as Training



(b) Set 2 as Training

Fig. 4: F1 scores by ratio for Class 0 (top boxes on each graph) and Class 1 (lower boxes on each graph) with comparison lines.

3 bins are used. Even though Set 1 is more class imbalanced than Set 2 with transfer, the random strategy is clearly not as strongly impacted by that imbalance. The spread of rows with high numbers of DCs and the rows without DCs seems to augment the class imbalance impact for specific ratios.

As already seen, when non-DC transfer is not used (Figure 5b) the *mostDCs* approach can only improve on Class 0, the majority class. Since *mostDCs* rarely outperforms the full training data, it implies that keeping the rows with the most DCs in one set and putting the rows with the fewest DCs in the other set is unsuccessful. The results hold when the sets are used for training and validation in either combination.

The *allDCs* approach can improve on Class 1 with 2-way analysis and equal frequency bins in either set, or on both classes with 3-way analysis and 3 equal sized bins with Set 1 as training data (Figure 5b). As already seen the *randomDCs* approach performs the best in general, and we see that almost every setup can result in a model that outperforms the full training data for both classes. The ratio, binning strategy, and t-way analysis therefore seems to not play a large role when random DCs are chosen to be put in each set. This *randomDCs* approach may therefore be the most promising for a new explainable dimensionality reduction approach.

In Figure 6 we see that the majority of trained datasets without transfer do not overfit, as defined by a difference between testing and validation F1 scores within $[-0.1, 0.1]$; most datasets using transfer overfit. The *mostDCs* approach has the largest level of overfitting of all trained models, as the models perform well on training data and poorly on test data. In addition, a total of 9 models were found to underfit as defined as having both training and testing scores below 0.5; all of these models were created with *mostDCs*.

E. Discussion

The analysis demonstrates that distinguishing combinations should be present in both training and validation data for training a good model. The number of DCs within a row does not correspond to the best rows for training, however, as the best datasets had a random set of rows with DCs as opposed to the rows with the highest number of DCs.

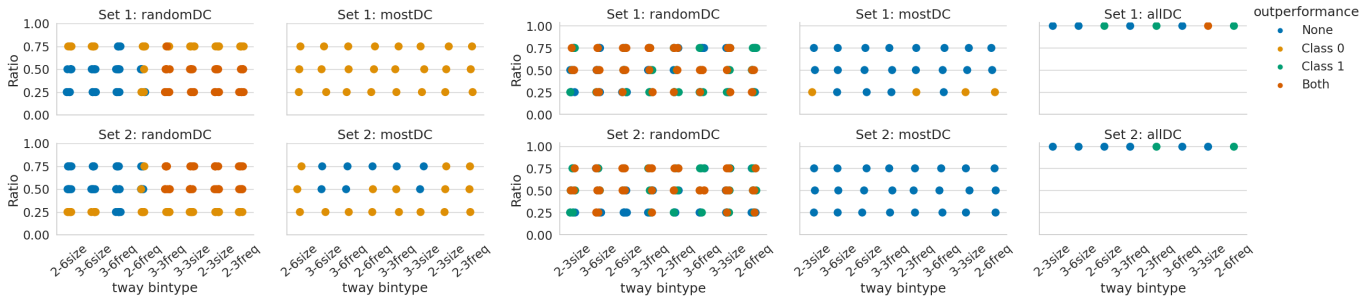
Overall these results imply that distinguishing combinations do play a role in the suitability of a dataset to train or validate a model. It appears that DCs are most valuable in the training dataset, but the relative number of DCs per row is not the most important metric. Although our models were trained without cross validation, most of the models with Set 1 without non-DC transfer were able to perform better than the full training data that used cross validation. These results imply the power of distinguishing combinations to affect the model predictions.

V. DC in Feature Reduction

We have seen above that distinguishing combinations (DCs) are important in training and validating datasets. We previously saw that we could use DCs to determine which rows could be removed from a dataset as well [2]. We now investigate if DCs can be used to inform which features, or combinations of features, are the most important in a dataset. The results of CFD are intuitively explainable; can we use this data to create a more explainable feature reduction approach?

A. Approach

To determine which features to keep in our feature reduction approach we find the distinguishing combinations in the data via CFD analysis. We experiment with two approaches to use that information:



(a) With Transfer

(b) Without Transfer

Fig. 5: Which models outperform the full training data model in F1 score for a particular class, neither class, or both classes. The x-axis denotes the t-way analysis, number of bins, then binning approach.

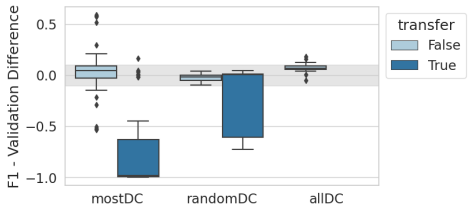


Fig. 6: Overfitting as defined by the difference between the test score and the validation score. Values below the gray box are considered to overfit.

- *Base importance*: The number of times a feature appears in a DC corresponds to its importance.
- *Normalized importance*: The number of times a feature appears in a DC is divided by the number of values in that feature to calculate importance.

We focus on the relative ranking of features instead of raw importance value. When reducing features, we drop features based on the resulting ranking.

There are a number of choices that must be made in how the data is prepared for CFD. These choices include:

- Binning strategy: Numerical values must be discretized, and this choice may affect the analysis
- Dropping duplicates: Duplicate rows can be removed prior to analysis or retained in the analysis. Significant duplicates may skew the calculation of DCs.
- Removing common entries: Entries that are common between both classes (e.g. duplicated other than label) may be removed or kept.
- Filtering: As previously described, if filtering is used then t-tuples that include lower strength t-tuples will not be included in the calculation of DCs. Without filtering, all t-way DCs are provided even if they include a lower strength t-way DC.
- Threshold: What frequency difference makes a t-tuple distinguishing.

We will experiment on the impact of these choices for using CFD for feature importance, and analyze what feature importance orders are created across the varied approaches.

B. Experimental Design

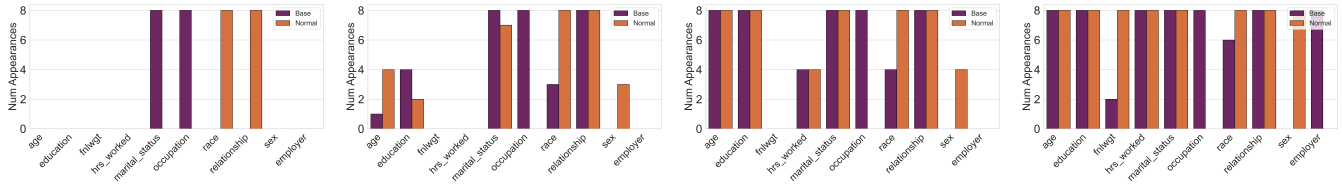
Distinguishing combinations are found via CFD using 3-way analysis. We experiment with thresholds of 5 and 10, and use 6 equal frequency bins to prepare the numerical features. We examine both dropping and keeping duplicates and common entries. We also use both base and normalized importance approaches. These setups result in 32 different combinations and thus 32 sets of DCs.

Each of these approaches then calculates a feature ranking, which we use to build reduced datasets by keeping the top k features, with $k \in \{8, 6, 4, 2\}$. Each dataset is used to train an SVM using the same grid search that is described in Section IV-C except without PCA and with 10-fold cross validation.

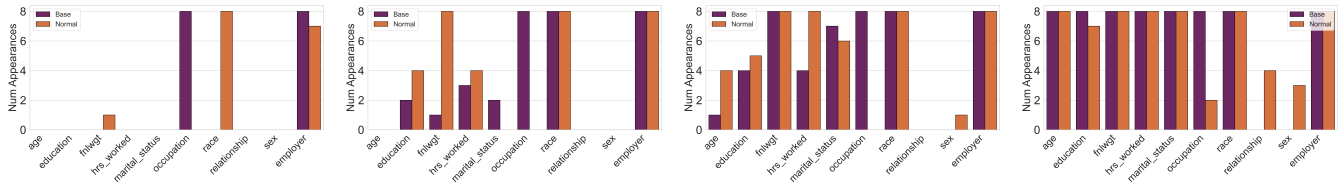
We compare our results to dimensionality reduction with either PCA or random forests on the original training data with the same number of components as our approach keeps features. PCA serves as a performance baseline due to its popularity in machine learning [13], but at the cost of losing direct feature interpretability. Random forests serves as a baseline explainable approach that does not also alter the data as PCA does. If CFD is useful for dimensionality reduction, we should see performance degrade slowly (or not at all) as low-ranked features are removed, and with performance similar to the comparison results.

C. Feature Selection Results

Figure 7 shows the number of datasets in which each feature is used by component count, filtering, and base vs. normalized importance approaches. There is no overlap of complete feature combinations between filtered and unfiltered CFD analysis for any component count, or between base and normalized importance. In Figures 7a and 7e the top 2 features clearly differ by importance tallying approach, and filtering approach. Base importance always chooses *occupation* as a top feature, and normalized importance always chooses *race*. Figures 7b and 7f show that our varying feature reduction approaches choose different features for third and fourth most important. With no filter, base adds the features most likely to be chosen as the top 2 features by normalized importance, and normalized always includes *marital_status*. With filtering,



(a) 2 components no filter (b) 4 components no filter (c) 6 components no filter (d) 8 components no filter



(e) 2 components, filter (f) 4 components, filter (g) 6 components, filter (h) 8 components, filter

Fig. 7: Frequency at which each feature is chosen as a top 2/4/6/8 feature, divided by importance approach and filtering.

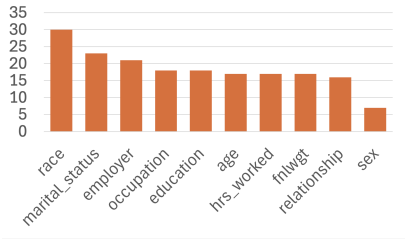


Fig. 8: Number of unique feature combinations in which a feature appears.

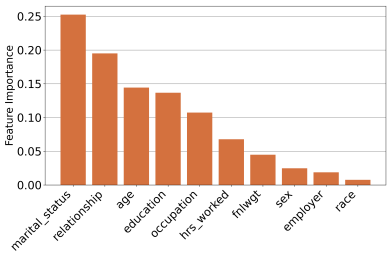


Fig. 9: Feature importance ranking of random forests.

fnlwgt appears in all normalized datasets, and either *education* or *hrs_worked* is also added. For base importance with filtering, *race* is most commonly added. With 6 components we see that base and normalized choose almost identical features when filtering is not used (Figure 7c), but vary more with filtering. *Relationship* is present in all unfiltered datasets, but no filtered datasets; *employer_type* and *fnlwgt* appear in all filtered datasets, but no unfiltered ones. For eight components (Figure 7d) it is easiest to see which components are the bottom ranking and thus never chosen: *occupation* for unfiltered normalized importance, *sex* for unfiltered base importance, and *relationship* and *sex* for filtered base importance.

Not all of the created combinations are unique, however. For 2 components there are 5 unique feature combinations created; for 4 components there are 13; for 6 components there are 11; and for 8 components there are 7. Of these 36 unique feature combinations, we see in Figure 8 that *race* is the most common and *sex* is the least common across all component amounts. *Race* is chosen for almost all normalized importance datasets and a large percentage of base importance datasets, whereas *sex* is only used in normalized datasets and primarily only with unfiltered CFD. While *marital_status* is popular for all unfiltered base importance datasets, it is primarily only used in higher component count datasets with filtering, and thus it is only the second most common feature overall in unique feature combination datasets.

We can directly compare our chosen features with the features chosen by random forests. In Figure 9 we see that random forests choose *marital_status* and *relationship* as the most important features, so that they will be in every dataset. The 4-8 component datasets also include *age* and *education*; the 6-8 also include *occupation* and *hrs_worked*; and the 8 component adds *fnlwgt* and *sex*. Our datasets frequently include the two features that random forests has determined are least important. The majority of our datasets include the random forest's most important feature, and about half include the second most important feature. Our least commonly occurring feature (*sex*) is only included for the random forest in the 8 component dataset. We cannot directly compare to PCA since PCA projects the data onto new axes, thus defining new features instead of only keeping specific existing features.

D. Machine Learning Results

Most of the pre-CFD processing does not affect the results (Figure 10); whether we remove duplicates, which cutoff we choose, and if we remove common entries does not have a strong effect on performance. The main differences come in filtering when finding DCs, and base vs. normalized importance calculations. Analyzing the data further, we see that duplicate rows and common entries are not frequent in the dataset, which likely explains why removing them does

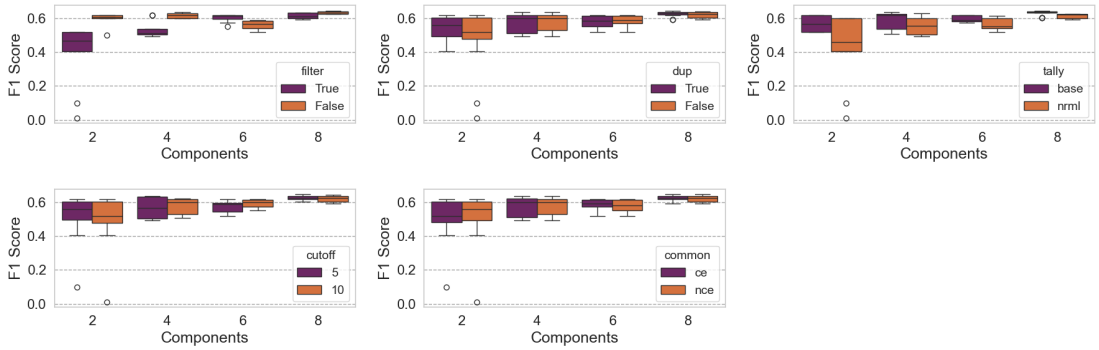


Fig. 10: Comparing each setup variable to number of components shows that F1 score is most affected by filtering and importance calculations.

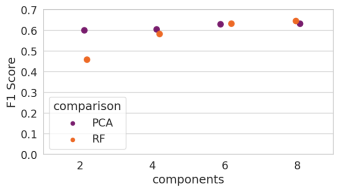
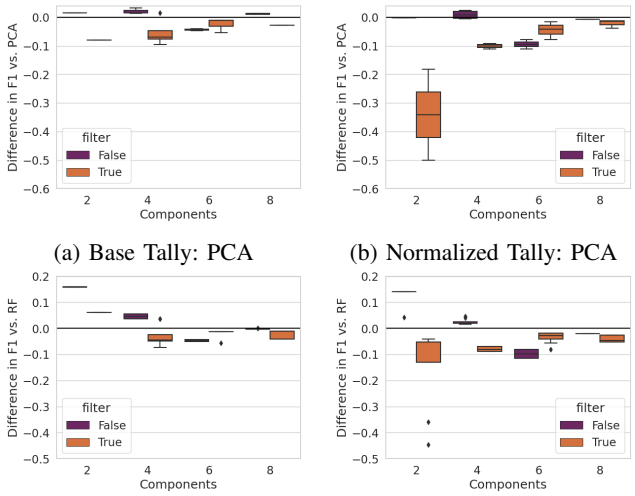


Fig. 11: Comparison approach results.



(a) Base Tally: PCA (b) Normalized Tally: PCA

(c) Base Tally: RF (d) Normalized Tally: RF

Fig. 12: Difference in F1 scores for CFD models vs. PCA and RF models for only unique feature combinations.

not affect which distinguishing combinations are found. There are 1,219 duplicates in Class 0 and 3,155 duplicates in Class 1, out of a dataset with just over 22,000 rows overall.

For our comparisons, PCA performs better than random forests (RF) for lower feature numbers, but slightly worse for higher feature counts (Figure 11). The feature count affects the RF approach more than it affects PCA. Our unique feature combination models have more feature combinations that under perform than over perform against PCA (Figure 12); 9 combinations perform better than PCA, and 27 perform worse. Of the ones that perform better than PCA, two are 8 component, one is 2 component, and six are 4 component; all were generated with unfiltered CFD analysis; 7 were generated with base tallying. All but one includes *marital_status*, all but two include *occupation*, only one contains either *fnlwgt* or *sex*, and only two include *hrs_worked*.

In Figures 12c and 12d we see that almost all generated datasets perform better than random forests for 2 or 4 components. For 6 or 8 components, random forests perform slightly better. It therefore appears that our approach is better for removing a large number of features, and random forests is better for removing only a few. These results imply that random forests' choice of the most important features is worse than ours, as the larger the number of features kept the less important the actual feature ranking becomes as most features are included in those datasets.

Figure 13 shows how likely each feature is to exist in a dataset with a particular number of components, as well as how likely they are to exist in datasets that perform better than PCA or random forests (RF). For 2 component datasets, *marital_status* being included is highly correlated to outperforming PCA with 2 components. Weaker correlations are shown with *occupation* positively correlated and *race* negatively correlated with outperforming PCA. There are no strong correlations with outperforming random forests, the strongest is a positive correlation with *occupation* (0.67).

For 4 components, there are no features strongly correlated with outperforming PCA. However, both *marital_status* and *relationship* have a strong postive correlation with outperforming RF. For 6 components there are no models that outperform PCA, but we can see that *age* and *sex* being in the dataset are negatively correlated with the F1 score, indicating that their inclusion may be part of the reason 6 components models perform poorly. For 8 components, no feature is strongly correlated with outperforming PCA. *Race* is strongly negatively correlated with outperforming RF, indicating that its inclusion in 8 component datasets may be part of the reasons we do not perform as well; it is not included in any RF models.

(a) 2 components (b) 4 components (c) 6 components (d) 8 components

Fig. 13: Correlations between the presence of a feature and the model's performance for only the unique feature combinations generated by CFD; a strong positive correlation to "outperform" (first two columns) indicates the feature's presence correlates with outperforming PCA or RF. For 6 components we instead compare to the F1 score, as no 6 component models outperform.

E. Discussion

When using unfiltered CFD analysis our approach performs better than random forests when removing more than half of the features. Our approach appears to select the most important features better than random forests, but is worse at choosing which features are least important. When using a base importance approach, our approach performs at least as well as PCA for most feature counts. CFD appears capable of identifying a smaller subset of features that preserves most of the predictive signal. PCA does not offer a clear performance advantage, which makes CFD appealing primarily for its improved interpretability. At each number of features except six, CFD-informed dimensionality reduction matched or outperformed PCA while also being more explainable, as individual features remained intact.

Based on these results, we recommend using the base importance approach with unfiltered CFD to reduce dimensions, as it is most likely to outperform both comparison approaches. Removing duplicates or common entries does not appear to affect the results in this dataset, but further work is needed to determine if other datasets have similar results.

There are several worthy avenues for further research in applying CFD or other combinatorial methods to feature reduction. In particular, would higher t-way settings (ex, 4-way) change the feature ranking meaningfully? Should feature ranking be based upon t-tuples of values rather than individual numbers of occurrences in DCs? On a higher-dimensional dataset, would CFD start to diverge more clearly from PCA or random forests in either performance or feature choice?

We also propose determining if our approach could pair well with PCA, using our approach to reduce features and PCA for projection onto new axes only. This combination approach may combine the best of each approach: a more explainable choice of features, while still retaining the benefit of modifying features before training to better separate classes.

There are a small number of threats to validity in this approach: examining a single dataset, training a single machine learning model type, and comparing to only two dimensionality reduction approaches. By focusing on a single dataset we are able to more thoroughly examine how well the approach works on that dataset, but the work will benefit from application to additional data. We focused on SVMs as they benefit from feature reduction, but further work could determine if other models also benefit. Another interesting question is how CFD compares with other approaches to feature reduction, such as sparse regularization methods like LASSO.

VI. Related Work

A common principle in machine learning is that increasing the size of the training dataset often leads to improved model performance. However, larger datasets also incur higher computational costs in terms of training time and resource consumption. Moreover, simply adding more data without considering its relevance or quality does not necessarily result in a better-performing model. For reasons of efficiency, cost, and scalability, there is often a need to reduce the size of training datasets while preserving their informative content. Existing approaches for reducing training data can be broadly categorized into four groups: feature selection, data summarization, instance generation, and instance selection.

Feature selection methods focus on reducing the dimensionality of training data in order to accelerate model training and improve interpretability [14]. There are many techniques for feature reduction such as Principal Component Analysis (PCA) [15] and sparse regularization methods like Least Absolute Shrinkage and Selection Operator (LASSO) [16] that are commonly used to retain only the most informative features [13]. These approaches typically achieve dimensionality reduction through a projection of original features or through regression-based feature selection.

Data summarization methods aim to produce compact representations of datasets, often through clustering or data distillation. Cluster-based approaches such as K-Means and hierarchical clustering reduce data volume by summarizing similar samples into cluster centroids. Dataset distillation has emerged as a promising direction, in which small synthetic datasets are optimized to approximate the distribution and learning behavior of the original data [17]. Instance generation techniques, by contrast, focus on synthesizing new examples to modify or balance datasets. A well-known example is the Synthetic Minority Over-sampling Technique (SMOTE) [18], which generates artificial samples to address class imbalance while preserving the underlying data distribution.

Explainability and interpretability in machine learning is an ongoing issue as many models are black boxes, where it is difficult to know why a model is making a particular prediction. Models can be explainable generally either because the outputs are interpretable in terms of the inner workings of the model regarding what factors cause a prediction, or due to a post hoc approach that explores the reasons for predictions separately after training [7], [8]. An explainable model's output may be more trustworthy, and easier to understand when something goes wrong [9].

As noted in the Introduction, combination frequency analysis has been used to address the problem of explainability in machine learning. In [19] and [20], it was shown how classification decisions could be explained or justified using combinations of features that are present in elements of the identified class but absent or rare in different classes. This earlier work also included a trial study of using frequency differences directly as a machine learning method for categorical identification problems. Performance was similar to conventional algorithms such as decision trees; this potential may be investigated in future work. Fault localization, i.e., the problem of identifying the particular value combinations that trigger a specific fault [21], is another possible application that may be studied in future work.

VII. Conclusions

In this paper we examined the role of distinguishing combinations (DCs) in explainable machine learning via two approaches: using DCs to create separate training and validation datasets to determine the importance of DCs in each; and using DCs to determine which features to keep during feature reduction, with comparison to PCA and Random Forests.

We found that distinguishing combinations should be present in both training and validation data to train a good model, and are most valuable in the training dataset. We also found that DCs can be used to choose which features should be kept when performing dimensionality reduction. The most effective approach for reducing dimensions is using unfiltered CFD results, and choosing the features that appeared in the highest number of distinguishing combinations.

Across our experiments it has become clear that filtering should in general not be used in combination frequency difference analysis. Filtering out lower strength t-way DCs may be useful for determining what data is included only in t-way DCs and not in lower strength DCs (for example in fault localization), but when using the DCs for any decisions regarding rows or features of data to use, the full list of t-way DCs provided by unfiltered analysis is best.

Our approach of using CFD for feature reduction performs well, and is more explainable than commonly used approaches such as PCA. For PCA, the data are projected onto new axes, which creates new features that are not as easily related to the original features. It can therefore be difficult to determine the roles of the original features in those trained machine learning models. For CFD, it is directly clear which features are included as they are not projected into a new space, and the reason for each feature being kept or removed is understandable. Our CFD approach also outperforms random forests, another explainable approach, for smaller feature counts.

References

[1] D. R. Kuhn, M. S. Raunak, and R. N. Kacker, "Combination frequency differencing," National Institute of Standards and Technology, Tech. Rep., December 2021.

[2] M. Olsen, M. S. Raunak, D. Richard Kuhn, H. Van Lierop, F. Badorf, and F. Durso, "A combinatorial approach to reduce machine learning dataset size," in *2025 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2025, pp. 248–257.

[3] E. Lanus, L. Freeman, D. R. Kuhn, R. N. Kacker, and Y. Lei, "Combinatorial testing metrics for machine learning," in *2021 IEEE Intl Conf on Software Testing, Verification and Validation*, 2021.

[4] T. Cody, E. Lanus, D. D. Doyle, and L. Freeman, "Systematic training and testing for machine learning using combinatorial interaction testing," *arXiv preprint arXiv:2201.12428*, 2022.

[5] D. R. Kuhn, R. N. Kacker, Y. Lei, and D. E. Simos, "Combinatorial methods for explainable ai," in *2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2020, pp. 167–170.

[6] J. Chandrasekaran, Y. Lei, R. Kacker, and D. Richard Kuhn, "A combinatorial approach to explaining image classifiers," in *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2021, pp. 35–43.

[7] M. Du, N. Liu, and X. Hu, "Techniques for interpretable machine learning," *Commun. ACM*, vol. 63, no. 1, p. 68–77, Dec. 2019. [Online]. Available: https://doi.org/10.1145/3359786

[8] F. Yan, S. Wen, S. Nepal, C. Paris, and Y. Xiang, "Explainable machine learning in cybersecurity: A survey," *Int. J. Intell. Syst.*, vol. 37, no. 12, p. 12305–12334, Dec. 2022. [Online]. Available: https://doi.org/10.1002/int.23088

[9] G. Rjoub, J. Bentahar, O. Abdel Wahab, R. Mizouni, A. Song, R. Cohen, H. Otrok, and A. Mourad, "A survey on explainable artificial intelligence for cybersecurity," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 5115–5140, 2023.

[10] D. R. Kuhn, M. Raunak, and R. N. Kacker, "Combinatorial coverage difference measurement," National Institute of Standards and Technology, Tech. Rep., June 2021.

[11] D. R. Kuhn, M. S. Raunak, C. Prado, V. C. Patil, and R. N. Kacker, "Combination frequency differencing for identifying design weaknesses in physical unclonable functions," in *2022 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2022, pp. 110–117.

[12] B. Becker and R. Kohavi, "Adult," UCI Machine Learning Repository, 1996, DOI: https://doi.org/10.24432/C5XW20.

[13] C. O. S. Sorzano, J. Vargas, and A. P. Montano, "A survey of dimensionality reduction techniques," 2014. [Online]. Available: https://arxiv.org/abs/1403.2877

[14] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[15] I. Jolliffe, "Principal component analysis," *Encyclopedia of statistics in behavioral science*, 2005.

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

[17] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros, "Dataset distillation," *arXiv preprint arXiv:1811.10959*, 2018.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[19] R. Kuhn and R. Kacker, "An application of combinatorial methods for explainability in artificial intelligence and machine learning," 2019.

[20] D. R. Kuhn, R. N. Kacker, Y. Lei, and D. E. Simos, "Combinatorial methods for explainable ai," in *2020 IEEE Intl Conf on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2020, pp. 167–170.

[21] L. Kampel, D. E. Simos, D. R. Kuhn, and R. N. Kacker, "An exploration of combinatorial testing-based approaches to fault localization for explainable ai," *Annals of Mathematics and Artificial Intelligence*, vol. 90, no. 7, pp. 951–964, 2022.