Impact of Parameter Dependency in Machine Learning Model

Abhishek Shah | Ehsan Khan

University of Michigan

Machine learning relies on data for model's performance and ability to predict depends on it’s dependency. Data dependencies can cause a variety of challenges, including model training overfitting and generalization issues. Thus, in this paper, we will discuss how to cope with the complexities of parameters and their data, dependencies, how they affect the machinery model, and how we can mitigate them. The performance of a machine learning model depends on parameter values, which are variables that the model learns during training. We can examine the regression model or neural network model that modifies the printer's weight, for instance. The machine learning model's performance can be significantly influenced by the parameters, training method, and properties of the data used to train the model.

This dependency, especially an issue when the model is not generalized well and has been forfeited due to overtraining. For example, in that case, when the new data is introduced and the parameters are not tuned well, the model will perform poorly, which can lead to an inaccurate prediction. [1] Also, in the edge cases, this dependency can lead to overfitting, where the models perform so well on the training data but also fail to generalize to unseen data, which is caused by model noise, learning noise, and outliers in the training data. This also means that the motto has become overly complex. Technically, the complexity of the model can be as high as the number of data points, so, for example, if we have 100 data points, the degree of freedom for the model can be 100. The model would fit very well for the training data, but as soon as we introduced new data said it would be useless.

There are many techniques with which we can address the parameter dependency. We sometimes use regularization, which is a commonly used technique that adds a penalty term to the last function of the model and prevents it from becoming overly complicated. This helps to make sure the model doesn’t overfill the data by effectively reducing the values of the hyperparameters, which allows the model to generalize well. We can also use the Bayesian method, which introduces a certain level of uncertainty into the model parameters, allowing the model to consider a wide range of different parameter values to improve the robustness of the changes in the data.

One of the main issues in machine learning is also data dependency as the models get highly affected by the training data. Some of the factors that affect data dependencies include the data size, it’s distribution and noise in the data set. If a model is trained on a small or biased dataset, it will not perform well to the new data set. Also, it will capture a lot of noise which will cause the model to underperform leading to inaccurate predictions. We can devise different strategies to address the data dependencies. One of those strategies is called data augmentation, which can be used to artificially increase the available data. This is done by creating modified versions of existing data, such as rotating images in the image in a classification model and adding semantic noises in the natural language processing. We can also use transfer learning, where a model is trained on one task and then transferred to another. This allows the model to generalize well based on the learn feature, which improves its overall performance. Understanding dependencies in machine learning is crucial, just as it is in feature engineering. Finding these dependencies can be very helpful in feature engineering because some features by themselves might not have the projected power when used, however employing a combination with other features considerably increases the model performance.

High interdependencies between variables, on the other hand, might result in multicollinearity, a situation in which two or more variables are so tightly associated that one can be predicted linearly from the others. This could have a detrimental effect on the model's performance, making it unstable and challenging to understand. Multicollinearity is addressed using dimensionality reduction techniques like Principal Component Analysis (PCA) [2], which allow to remove unnecessary features without sacrificing crucial data. Data leaking occurs when information that isn't supposed to be available is used for model training because of data dependencies. This frequently results in excessively optimistic performance estimates during training and validation but fails when used to make predictions about the real world. It's essential to carry out thorough data sanitization, make sure the data used for model training is representative, apply cross-validation techniques, and regularly track the model's performance in order to prevent data leakage. Predictions may be biased because of data dependencies, particularly if the dependent variables include delicate characteristics like race or gender. It's critical to assure fairness as AI systems are increasingly utilized in high-stakes decision-making. To do this, meticulous data gathering, and utilization, the adoption of impartial and fair algorithms, and ongoing bias monitoring are all necessary.

Relationships between two or more data points are known as data dependencies. [3] They might exist, might not exist, or might be either. Positive dependencies signify a correlation between the two data points, meaning that if one increases, the other is probably going to rise as well. Negative dependencies signify an inverse relationship between the two data points, meaning that if one increases, the other is likely to decrease. Dependencies that don't exist indicate that there is no connection between the two data pieces. Data dependencies can have a variety of effects on the accuracy of machine learning predictions. Inaccurate predictions could be made if a model is unaware of a positive correlation between two data items. For example, if the model is attempting to estimate the price of a house and is aware that the size of the house is a component but is unaware that the location of the house is also a factor, it may produce false predictions for houses in different places.

Inaccurate forecasts can also result from negative correlations between data points. For instance, if the model is attempting to forecast the likelihood that a customer will default on a loan and is aware that the customer's income affects the risk but is unaware that the customer's credit score also affects the risk, it may make incorrect predictions for customers with various credit scores. Machine learning models must find and include data dependencies to improve prediction accuracy. This can be achieved by using techniques like feature engineering and feature selection. Choosing which traits to include in a model is the procedure known as feature selection while the process of creating new features that are not directly available in the data is referred to as feature engineering. With the inclusion of data dependencies into machine learning models, it is possible to increase prediction accuracy and make better decisions. This can be accomplished by combining existing features, altering them, or creating whole new ones. The purpose of feature engineering is to create features that outperform raw data in predicting the target variable.

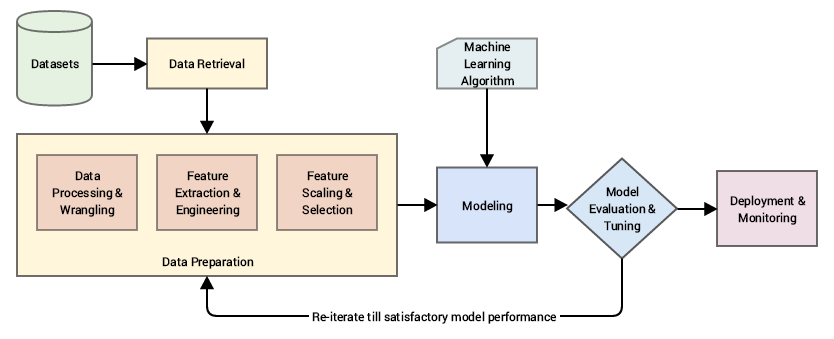


Figure .Image from Practice Machine Learning with Python, Appress/Springer

Understanding data dependencies can help with feature engineering [3]. Certain characteristics may not be valued on their own, but they become relevant when combined with others, demonstrating their interdependence. For example, if we try to anticipate if a client would churn, we may discover that the client's age alone is not very predictive. Combining the client's age and the number of years as a customer may be even more predictive of churn. Another application of dependencies in feature engineering is the building of features based on the links between distinct features. For example, we might develop a feature that demonstrates the relationship between the customer's income and spending patterns. This trait may be beneficial in predicting whether a consumer will fall behind on a loan. Understanding data relationships can be a useful technique for feature engineering. You may generate features that are more predictive of the target variable than raw data by identifying the links between specific characteristics.

Feature Engineering Techniques to deal with data dependencies.

* Interaction Features: we can create new features by combining existing features by using mathematical model. For example, if we are trying to predict the value of the house using the house size either, it is big or small and the location. If it is good or bad, we have too many features to work with. In this case, we can combine the house size and locations as a single feature, which would reduce the number of features. We can also try to add these terms or multiply them if we can capture non-linear relationships, or dependencies, as compared to leaving those features by themselves. [4]

A picture containing line, diagram, font, mathematics

Description automatically generated

Figure .•Interaction Features: we can create new features by combining existing features by using mathematical model. [5]

* Polynomial feature engineering: just like the interaction features we can also use the polynomial features by raising each feature to two different powers. For example, if we have a feature x, we can create a new feature by x^2 or x^3. This can also reduce the number of degrees, which would also help in featuring engineering. [5]
* Binning/Discretization: The process of turning continuous values of variables into discrete variables is known as data discretization or binning. This is frequently utilized in data mining, data science, and the development of AI models. Numerous regression and classification models, including decision trees and Naive Bayes, perform better with discrete values and may benefit from discretization. [6] Equal-width and equal-frequency discretization are the most frequently utilized discretization algorithms; such unsupervised methods determine interval bounds without considering the target into account. Contrarily, cut-points and the ideal number of divisions can be picked out automatically using decision tree-based discretization approaches. By incorporating values that are strongly connected to various classes of the target values into the discretization process, however, information loss can happen.

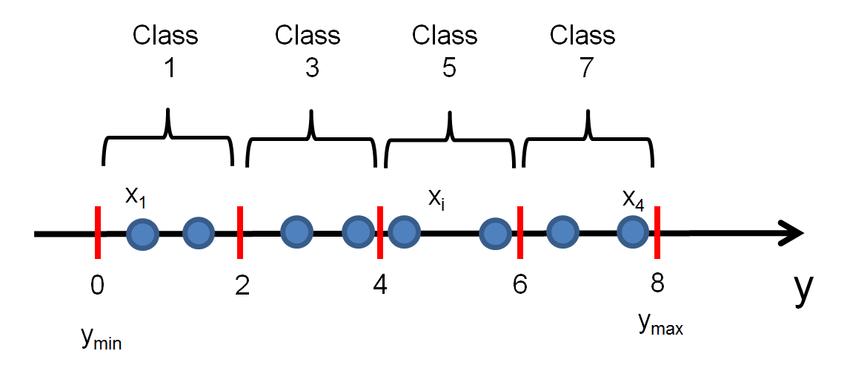


Figure . Example of discretization by binning a continuous label y into Q = 4 equal-length intervals. Each interval is associated to a unique class label. In this example, the class label for each interval is equal to the mean in each interval.

* Lag Features: The concept of lag is used in time series forecasting. To build a dataset from scratch that may be utilized for supervised learning, lag features are built by moving the time series data by a particular number of time steps. Predicting the value at the following time step based on the value at the prior time step is the basic method. The shift() method in the Pandas package can be utilized to extract these shifted or delayed characteristics from a time series collection. Lag features are produced using the sliding window method, where the window's width determines how many previous values are incorporated in the original data set. [7] A wider window could allow the model access to more data; however, it also generates a model more complex. It is recommended to perform a sensitivity analysis to determine the ideal window width for a specific problem.
* Categorical Encoding: Category-variable encoding is the process of transforming category variables to numerical variables so that machine learning models can decode and extract useful information. Since most machine learning algorithms accept just numerical variables, this is necessary. There are multiple types of categorical data encoding methods, and choosing the appropriate one is essential for the performance of models and the development of features. [8] Categorical variables are frequently expressed as strings or subcategories that contain a finite number of entries. If they have an inherent order, they can either be ordinal or nominal. Ordinal data should keep the order in which the categories are presented, but nominal data should consider the existence or absence of a feature. For nominal data, one hot encoding is employed, while label encoding is used for ordinal data. Each of the levels of a categorical feature corresponds with a binary variable having either 0 or 1, and these newly created binary features are known as dummy variables in one hot encoding.

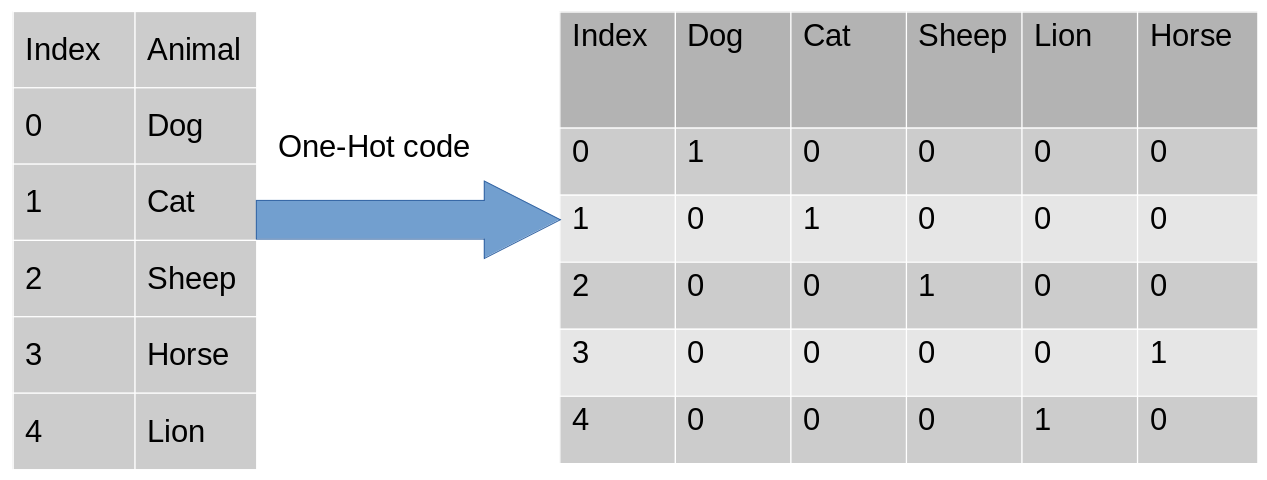


Figure . One-Hot Encoding

* Feature Scaling/Normalization: A typical phase in machine learning pipelines is feature scaling. The standardization of independent variables or data characteristics to have the same range of values is known as feature scaling. Normalization and standardization are two popular scaling strategies. Normalization rescales features to the range [0, 1], whereas standardization ensures that each feature's values have a mean of zero and a variance of one. Power transformer, quantile transformer, and robust scaler are some more scaling approaches. The choice between normalization and standardization depends on the distribution of the data and the algorithm utilized. If data fails to conform to a Gaussian distribution, normalization is advantageous; when data adheres to a Gaussian distribution, standardization is favorable.
* Feature Selection/Dimensionality Reduction: Feature selection is the process of choosing a subset of input features that are most important to the model. This can be performed by making use of task-dependent or task-independent techniques. The method of selecting features based on how important are to a particular important to learning goal, such as either regression or classification, is referred to as task-dependent feature selection. Independent approaches, such as PCA or clustering, select features based on their inherent characteristics instead of the labels they generate. Dimensionality reduction refers to the technique of reducing the number of input features while maintaining as much information as possible. This is possible using techniques like principal component analysis (PCA) [9]or clustering. PCA work by determining the underlying structure of the data and then creating a smaller set of features that represent the bulk of the variance in the original data. Clustering is grouping similar characteristics and identifying a representative feature for each group.

There are other features as well such as time-based features which can identify temporal relationships and seasonality patterns in the data because they are generated from date or time variables. Applying procedures specifically designed for the problem area is what domain-specific transformations do. They can also capture certain dependencies based on prior knowledge. Tokenization, stemming, and TF-IDF [10] are a few feature extraction techniques that are used to transform text or unstructured data into a numerical representation that captures dependencies within the text. These methods aid in identifying significant relationships and patterns in textual data.

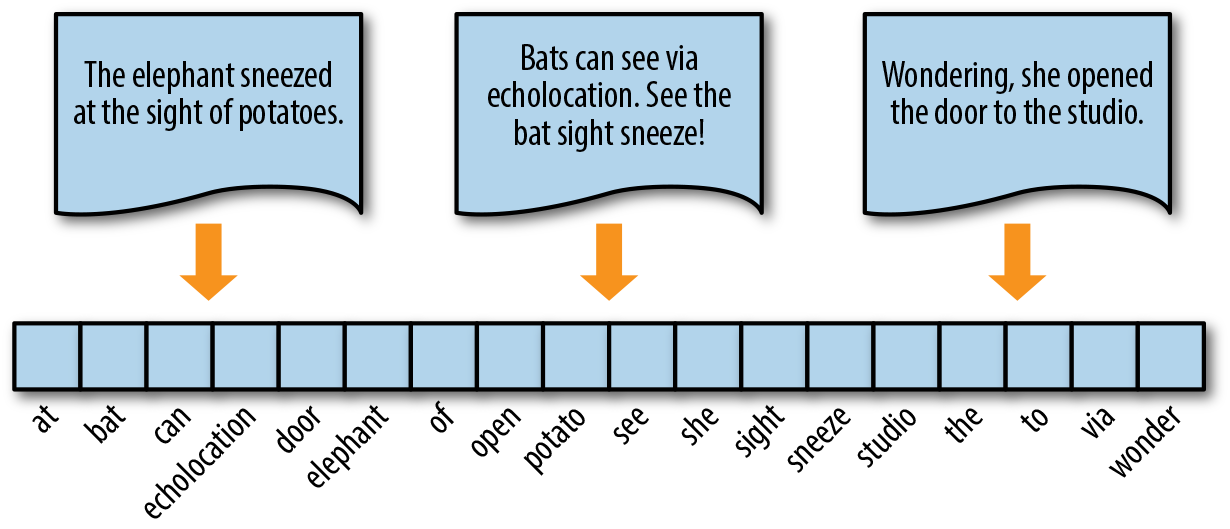


Figure . Encoding documents as vectors [10]

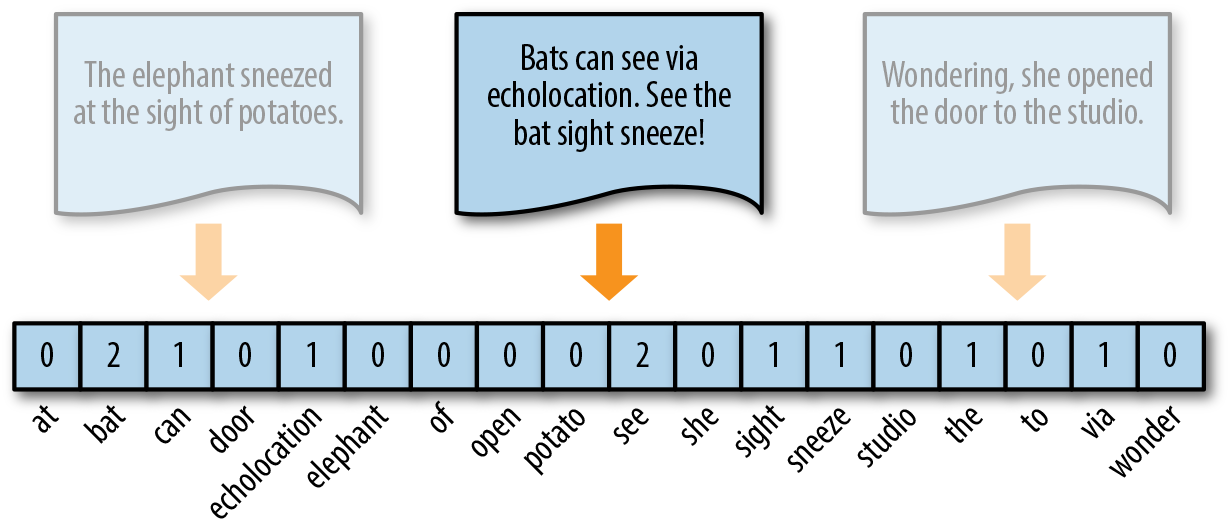


Figure .Token frequency as vector encoding [10]

A picture containing sketch, design

Description automatically generated

Figure . Data Impact

**Features**

One more problem related to data dependency is feature selection. Usually, features are dependent on the combination of multiple variables, if the engineer is not aware of the data set background and data expectations, it requires them to get the understanding from research and technical people which requires a lot of time and resources. Identifying relevant features that capture the data dependencies becomes crucial. Engineers need to determine which variables or combinations of variables contribute most to the target outcome. This process can be particularly difficult when dealing with a large number of features or when the dependencies are non-linear or subtle. Feature engineering techniques, domain knowledge, and automated feature selection algorithms can help mitigate this challenge.

A picture containing text, screenshot, font, diagram

Description automatically generated

Complex feature analysis is big issue as well. In many cases, features exhibit complex interactions and dependencies with multiple variables. These interactions may not be immediately apparent and require careful analysis and exploration of the data. Understanding these interactions is crucial for selecting the most informative features that capture the underlying patterns effectively. Complex feature analysis requires a combination of data exploration, domain knowledge, and analytical techniques to uncover the underlying dependencies and relationships. It assists in selecting the most informative features, designing effective feature transformations, and gaining a deeper understanding of the data, ultimately improving the performance and interpretability of AI models.

There is one more factor which is dependent on feature complexity is coorelated features. Correlated features refer to variables in a dataset that exhibit a strong linear or non-linear relationship with each other. When two or more features are highly correlated, they tend to contain similar information, which can lead to redundancy and potentially impact the performance of machine learning models.

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Dependency of Mins played vs Points Scored

**Conclusion**

The main purpose of this project is to help engineers and developers in the perspective of saving time and resources. There are some tools available in the market, so we have analyzed them. It is important to note that not all correlated features should be removed. Some correlations may carry important information and removing them could result in a loss of valuable insights. Therefore, a careful analysis of the specific problem domain, the nature of correlation, and the impact on model performance are necessary to make informed decisions regarding correlated features in feature selection. A thoughtful analysis of the problem domain, the nature of correlation, and the impact on model performance are necessary to make informed decisions regarding correlated features in feature selection. It requires a combination of domain expertise, experimentation, and trade-off considerations to strike the right balance between interpretability and performance. We are confident about this idea, right now it's a basic implementation of concepts and information work. But for market-level applications, we need time, so we can test it on more diverse and complex variant data sets.

# Works Cited

|  |  |
| --- | --- |
| [1] | J. M. E. H. S. Adam Cannon, "Machine Learning with Data Dependent Hypothesis Classes," *Journal of Machine Learning Research,* vol. 2, pp. 335-358, 2002. |
| [2] | Z. Jaadi, "A Step-by-Step Explanation of Principal Component Analysis (PCA)," Builtin, 29 May 2023. [Online]. Available: https://builtin.com/data-science/step-step-explanation-principal-component-analysis. [Accessed 17 June 2023]. |
| [3] | Google, "Data Dependencies," google, 18 July 2022. [Online]. Available: https://developers.google.com/machine-learning/crash-course/data-dependencies/video-lecture. [Accessed 17 June 2023]. |
| [4] | C. Molnar, Interpretable Machine Learning, Munich: Mucbook Clubhouse, 2022. |
| [5] | M. P. N. J. S. e. a. Jiu, "Sparse Hierarchical Interaction Learning with Epigraphical Projection," *Journal of Signal Processing Systems ,* vol. 4, no. 92, p. 637–654 (2020), 2020. |
| [6] | J. Brownlee, "How to Use Polynomial Feature Transforms for Machine Learning," Machine Learning Mastery, 29 May 2020. [Online]. Available: https://machinelearningmastery.com/polynomial-features-transforms-for-machine-learning/. [Accessed 17 June 2023]. |
| [7] | Train in Data, "Data discretization in machine learning," Train In Data, 04 2022 Jul. [Online]. Available: https://www.blog.trainindata.com/data-discretization-in-machine-learning/#:~:text=Data%20discretization%2C%20also%20known%20as,train%20models%20for%20artificial%20intelligence.. [Accessed 17 June 2023]. |
| [8] | J. Brownlee, "Basic Feature Engineering With Time Series Data in Python," Machine Learning Mastery, 14 Sept 2019. [Online]. Available: https://machinelearningmastery.com/basic-feature-engineering-time-series-data-python/. [Accessed 17 June 2023]. |
| [9] | S. Saxena, "Here’s All you Need to Know About Encoding Categorical Data (with Python code)," Analytics Vidhya, 13 Aug 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/08/types-of-categorical-data-encoding/. [Accessed 17 June 2023]. |
| [10] | M. Hauskrecht, "Dimensionality reduction Feature selection," 17 June 2023. [Online]. Available: https://people.cs.pitt.edu/~milos/courses/cs2750-Spring04/lectures/class20.pdf. [Accessed 17 June 2023]. |
| [11] | R. B. T. O. Benjamin Bengfort, Applied Text Analysis with Python, Sebastopol: O'Reilly Media, Inc, 2018. |